

Virtual Institutions



A dissertation submitted for the degree of
Doctor of Philosophy in Computing Sciences

by

Anton Bogdanovych

Sydney, Australia
2007

© Copyright by
Anton Bogdanovych
2007

CERTIFICATE OF AUTHORSHIP/ORIGINALITY

I certify that the work in this thesis has not previously been submitted for a degree nor has it been submitted as a part of requirements for a degree except as fully acknowledged within the text.

I also certify that the thesis has been written by me. Any help that I have received in my research work and the preparation of the thesis itself has been acknowledged. In addition, I certify that all information sources and literature used are indicated in the thesis.

Signature of Candidate

With gratitude

to my mother, who introduced me to life;

to my grandmother, who introduced me to writing;

to my father, who introduced me to research

and to my beautiful wife, who introduced me to love.

Table of Contents

Chapter 1 Introduction	1
1.1 Motivation	1
1.2 Taxonomy of Virtual Institutions	9
1.3 Research Problem	11
1.4 Objectives	13
1.5 Research Method	13
1.6 Contributions and Significance	15
1.6.1 Contributions	15
1.6.2 Significance	16
1.7 Structure	18
Chapter 2 Background	20
2.1 Virtual Worlds	20
2.1.1 Interactivity	20
2.1.2 Collaboration	21
2.1.3 Definitions and Terminology	21
2.1.4 Benefits of Using Avatars	23
2.1.5 Historical Overview of Virtual Worlds	24
2.1.6 Enabling Technologies	26
2.2 A Need for a Methodology	39
2.2.1 Methodologies in Computer Science	41
2.2.2 Methodologies for Virtual Worlds	42
2.3 Virtual Worlds as Multiagent Systems	44
2.4 Need for Institutions in Virtual Worlds	47
2.5 Electronic Institutions	49
2.5.1 Interactions of Participants in Electronic Institutions	50
2.5.2 Specification	51
2.5.3 EIDE Framework	53
2.5.4 Example: Trading Institution	55
2.5.5 Humans and Agents in Electronic Institutions	65
2.6 Summary	67

Chapter 3	Virtual Institutions	68
3.1	The Concept	69
3.2	The Metaphor	70
3.3	Visual Interaction Layer	73
3.3.1	3D Interaction Space	73
3.3.2	Garden	74
3.3.3	Institutional Buildings	74
3.3.4	Avatars	75
3.3.5	Rooms	77
3.3.6	Doors	78
3.3.7	Map	78
3.3.8	Backpack with obligations	78
3.3.9	Events/Actions/Messages	79
3.4	Normative Control Layer	79
3.4.1	Federations	81
3.4.2	Institutions	81
3.4.3	Scenes and Transitions	81
3.4.4	Performative Structure	82
3.4.5	Connections	83
3.4.6	Obligations	83
3.4.7	Data Types in Ontology	83
3.4.8	Illocutions and Messages	84
3.4.9	Synchronization Issues	84
3.5	Concept Illustration: Trading Institution	85
3.6	Formalizing the Concept using Z Specification Language	90
3.6.1	Virtual Institution Terms	94
3.6.2	Virtual Institution Model	95
3.6.3	The State of a Virtual Institution at Run Time	108
3.6.4	Initialization of the States	115
3.6.5	Operations	119
3.7	Summary	125
Chapter 4	Approach and Methodology	127
4.1	Virtual Institutions Methodology	127
4.2	Technological Solution	134
4.2.1	Step 4. Automatic Generation of Virtual Institutions	135
4.2.2	Step 5. Annotation	145
4.2.3	Step 6. Integration	148

4.3	Deployment	152
4.3.1	Normative Control Layer	155
4.3.2	Visual Interaction Layer	155
4.3.3	Communication Layer	156
4.4	Summary	164
Chapter 5 Learning Aspects		165
5.1	Related Work	168
5.2	Implicit Training	171
5.2.1	Scenario	172
5.2.2	Visual and Normative Levels of Execution	174
5.2.3	Learning to Imitate the Human	175
5.2.4	Assessing the Cognitive State	177
5.2.5	Distributed User Modeling	178
5.3	Implementation Details	181
5.3.1	Comparing the Trajectories	181
5.3.2	Recording	184
5.3.3	Using the Learning Graph	186
5.3.4	Nearest Neighbor Classifier	187
5.3.5	Detecting the Visibility of Objects	188
5.4	Experiments	190
5.4.1	Experiment 1: Trajectory Recognition	191
5.4.2	Experiment 2: Training the External Agent	195
5.5	Training the Internal Agent	201
5.6	Summary	204
Chapter 6 Virtual Institutions in E-Commerce		206
6.1	Issues in Existing E-Commerce Solutions and the Potential of Virtual Institutions in Addressing Them	206
6.1.1	Drawbacks of Existing E-Commerce Solutions	207
6.1.2	Beneficial Aspects of Virtual Institutions	217
6.2	Developing a Virtual Institution for E-Tourism	221
6.2.1	Selecting the Application Domain	221
6.2.2	Eliciting Specification Requirements	228
6.2.3	Developing the Normative Control Layer	238
6.2.4	Developing the Visual Interaction Layer	244
6.3	Summary	246

Chapter 7 Conclusion and Future Work	247
7.1 Future Work	248
7.1.1 Design Grammars	249
7.1.2 Gestures	249
7.1.3 Institutional Rules and Natural Language	249
7.1.4 Introducing Implicit Social Conventions	250
7.1.5 Evolution of Virtual Institutions	250
7.1.6 Normative Virtual Environments	251
7.1.7 Learning	251
7.1.8 Agent Programming	253
7.1.9 World Trotter Institution Prototype	253
7.1.10 Other Application Domains	254
Appendix A Z Specification of Electronic Institutions	255
A.1 Electronic Institution Static Data Structures	255
A.1.1 Basics: Variables, Constants, Terms, Ontology, Language	255
A.1.2 The Social Model	259
A.1.3 The Communication Model	260
A.1.4 The Normative Model	262
A.1.5 The Performative Model	262
A.1.6 Standard Constraints on Performative Models	268
A.1.7 The Electronic Institution	271
A.2 The State of an Electronic Institution at Run Time	272
A.2.1 Properties and Environments	272
A.2.2 Social Model State	273
A.2.3 Normative Model State	274
A.2.4 Performative Model State	274
A.2.5 The Electronic Institution State	278
A.3 Electronic Institution Operations	278
A.3.1 Operations on Scene Instances	279
A.3.2 Operations on Transition Instances	280
A.3.3 Dialogue Moves	280
A.3.4 Operations on the Performative Model State	283
Appendix B ISLANDER Specification Example: Trading Institution	285
Bibliography	305

List of Tables

2.1	Technologies Supporting Virtual Worlds: Feature Overview.	37
3.1	Mapping between 3D Virtual Worlds and Electronic Institutions	80
4.1	Action/Message Table for Trading Institution.	149
4.2	Scene Bounds.	151
4.3	Transition Bounds.	151
4.4	User Permissions for Trading Institution.	152
5.1	The Steps of the Levenshtein Distance Algorithm.	183
5.2	Attributes Used during the Training Session.	197
5.3	A Fragment of Data Used in Recording.	198
5.4	A Fragment of Data Used in the Experiments.	200

List of Figures

1.1	Concept Taxonomy of Virtual Institutions	10
2.1	Inside the Cybertown Virtual World	28
2.2	An Environment of the Active Worlds Universe: Example	29
2.3	Adobe Atmosphere Builder	32
2.4	Inside the Virtual World of Second Life	33
2.5	Maldives Embassy in Second Life	36
2.6	Electronic Institutions Development Environment	53
2.7	Roles in the Trading Institution	55
2.8	Ontology used for the Trading Institution.	56
2.9	Illocutions, Content Language and Communication Language	58
2.10	A Performative Structure of the Trading Institution	59
2.11	Scene Protocol for RegistrationRoom	61
2.12	Scene Protocol for MeetingRoom	62
2.13	Scene Protocol for TradeRoom	63
2.14	Norm Example	65
3.1	Virtual Institutions Metaphor	72
3.2	The Virtual World of the Trading Institution	85
3.3	Registration Room Inside the Trading Institution	87
3.4	Meeting Room Inside the Trading Institution	88
3.5	Trading Room Inside the Trading Institution	89
3.6	Basic Formalization Components.	92
3.7	Schema Relationship Diagram for Z-Specification.	93
4.1	Methodology steps.	128
4.2	Example of a Room generated using World Generator.	136
4.3	Euclidian representation. Human knows: Room 1 must be behind Door 4.	138
4.4	A planar triangulated graph and two possible rectangular layouts.	140
4.5	A rectangular dual representation computed by OCoRD. Shaded rectangles are due to breaking points.	141
4.6	A planar embedding of a graph with its input file.	142
4.7	Algorithm: Rectangular Dual Construction	143

4.8	Generating the 3D representation of a Performative Structure graph. . .	144
4.9	An Example of a Shape Rule.	146
4.10	Trading Institution Generated and Annotated Using Shape Grammars. .	147
4.11	Room Design Using Atmokits.	148
4.12	Runtime Architecture.	153
4.13	Causal Connection.	158
4.14	The Itchy Feet System.	159
4.15	The Architecture of the Causal Connection Server.	160
4.16	Component Interaction: Sequence Diagram	163
5.1	Interaction between Autonomous Agents and their Principals.	166
5.2	Visual and Normative Levels of Execution.	173
5.3	Distributed User Modeling	180
5.4	A Fragment of the Learning Graph.	185
5.5	Detecting the Visibility.	189
5.6	Trajectories Used for Training.	192
5.7	Comparing the Trajectories: Experiments.	194
5.8	Training the Guest Agent in the Garden.	196
5.9	Experiments: Avatar Positions and Eye Direction	199
5.10	The Scene Protocol for Meeting Room.	201
5.11	The Fragment of Assistant’s Learning Graph.	203
6.1	Age and Location of Participants.	233
6.2	Virtual Amsterdam Tour as Implemented in Second Life Technology. . .	241
6.3	Hotel Visualization Service.	242
6.4	Performative Structure of the World Trotter Institution.	244
6.5	The Map of the World Trotter Institution.	245
6.6	Booking Room.	245

List of Definitions

1	Virtual Environments	9
2	Immersive Normative Virtual Environments	10
3	Virtual Worlds	21
4	Avatars	22
5	3D Virtual Worlds	22
6	Open Systems	44
7	An Agent	44
8	Autonomous Agents	44
9	Principle	44
10	Multiagent Systems	45
11	Distributed Artificial Intelligence	45
12	Normative Multiagent Systems	46
13	Institutions	47
14	Electronic Institutions	49
15	Virtual Institutions	69
16	Implicit training	171
17	Cognitive State	177

Acknowledgments

This thesis would not be possible without all of the people who have helped me with its completion.

First of all, I would like to thank my four supervisors. Four initially sounded like a frightening number, but resulted in a bright and exciting collaboration enriched with four colorful personalities, all of whom I consider friends.

Great thanks are due to Simeon Simoff, who encouraged me to “fly in the air” and generate these crazy ideas, and who was always ready to explore them with me.

To Carles Sierra, who was always keen on interrupting my flights of ideas and grounding me. Who, on the one hand, is one of the most intelligent people I know and, on the other hand, one of friendliest and most fun-loving professors I have ever met.

To Helmut Berger, who has added a touch of “Ordnung” into my research, and who has helped with every single aspect of my work and also is a fantastic friend.

To John Debenham, who was always ready to provide me with any kind of help and whose great sense of humor often made my day.

My gratitude goes also to John Hughes and the Institute for Information and Communication Technologies for accepting my scholarship application... as well as to the University of Technology Sydney, for waiving my tuition fees, funding conference trips and simply for giving me the opportunity to enjoy being a part of one of the best universities in Australia.

To my colleagues Paul Bogg, Les Green, Igor Cregol and Ante Prodan, as well as to other friendly people from “Red Square”, who were always there when I needed to hear a friendly human voice or when I was keen to share one of my new ideas with them.

Thanks to Olga Voronina and Natalie Taranec for helping to edit some of the chapters.

Thank you to all of the people from other universities who collaborated with me: Sara Drago, Massimo Ancona, Gianluca Quercini, Mary Lou Maher, and Ning Gu.

Thanks to Marc Esteva for creating the specification of the “World Trotter” Institution and for helping to produce Z-specification of Virtual Institutions; to Simon Biber for developing some of the prototypes and to Bruno Rosell i Gui for his help with EIDE.

And finally, thanks to the imaginary girl from our office, who was very quiet, but was always ready to listen and accepted full responsibility for any mess that was present at any given time. You rock, Ming!

Abstract

This thesis establishes Virtual Institutions as a comprehensive software engineering technology for the development of 3D Virtual Worlds that require normative regulation of participants' interactions (such as the commercially-oriented Virtual Worlds).

3D Virtual Worlds technology currently offers somewhat unregulated environments without means to enforce norms of behavior and interaction rules on their inhabitants. Furthermore, existing methodologies for Virtual Worlds development focus primarily on the design side of the "look-and-feel" of the inhabited space. Consequently, in current 3D Virtual Worlds it is difficult to keep track of the deviant behavior of participants and to guarantee a high level of security and predictable overall behavior of the system.

The Virtual Institutions Methodology proposed by this dissertation is focused on designing highly secure heterogeneous Virtual Worlds (with humans and autonomous agents participating in them), where the participants behave autonomously and make their decisions freely within the limits imposed by the set of norms of the institution. It is supported by a multilayer model and representational formalisms, and the corresponding tools that facilitate rapid development of norm-governed Virtual Worlds and offer full control over stability and security issues.

An important part of the Virtual Institutions Methodology is concerned with the relationship between humans and autonomous agents. In particular, the ways to achieve human-like behavior by learning such behavior from the humans themselves are investigated. It is explained how formal description of the interaction rules together with full observation of the users' actions help to improve the human-like believability of autonomous agents in Virtual Institutions. The thesis proposes the concept of implicit training, which enables the process of teaching autonomous agents human characteristics without any explicit training efforts required from the humans, and develops the computational support for this new learning method.

The benefits of using Virtual Institutions are illustrated through applying this technology to the domain of E-Commerce. It is demonstrated that providing shoppers with a normative environment that offers immersive experience and supports important real world attributes like social interaction, location awareness, advanced visualization, collaborative shopping and impulsive purchases can improve existing practices in E-Commerce portals.

Chapter 1

Introduction

In this chapter we outline the key reasons that motivated us to conduct this research and problems it attempts to address.

Our initial motivation was very practical in nature. We were concerned with the problem of improving the shopping experience of the customers in online shops. One of the possibilities to achieve this is provided with 3D¹ product presentation.

The use of 3D product presentation on the Web was a hot topic in late nineties. Many businesses tried to enhance their Internet sites with 3D models of the auctioned goods to offer customers better product presentation facilities. Unfortunately, the majority of investments in 3D shopping didn't pay off. The lack of success was usually coupled with issues like costs of 3D modeling, visualization performance and slow Internet connection [103].

Nevertheless, recent developments prove that in the near future 3D models may be faster and cheaper to create than quality photographs [72]. Furthermore, broadband Internet connection is becoming much more popular than dial-up and the performance of computers as well as the visualization algorithms have improved quite significantly. These factors turned the attention of retailers back to 3D technology.

Further in this chapter we explore the key topics that in our opinion should motivate the researchers in the area of 3D E-Commerce, highlight the important characteristics of 3D environments that improve the shopping experience and describe how those can be integrated into future E-Commerce portals.

1.1 Motivation

One of the reasons why researchers and developers in the nineties started to work on providing 3D visualizations of the products in the E-Commerce portals is because they saw this new technology having a potential to create more realistic experience about the product. With the help of the Internet they expected this experience to be easily transmittable to the customers, so that customers have a similar understanding of the appearance

¹stands for 3-dimensional

and the features of the product as they would have gotten in a real store. Unfortunately, beyond these simple expectations not much research ground was laid, which became one of the reasons for the temporary retreat of 3D technology in E-Commerce.

Some studies analyzing the benefits of *3D product presentation* started to appear when the idea of 3D shopping was already well abandoned. Proving the usefulness of 3D visualization some researchers [56] claim that a virtual experience (3D product presentation) has the potential to be richer than both direct (manipulation with a real product) and indirect (the “classical” catalogue type of product presentation) experience because it can be simulated, framed, annotated and contextualized. However, the conducted study [56] showed that when tactile affordances are the most relevant for the product (e.g. touching the fabric), a virtual experience may have the same effect as indirect experience.

The possibility of *scanning physical interaction behaviors* of 3D objects proposed by [151] draws an even more optimistic picture. The scientists have managed to produce a fully automated device for scanning the interactive 3D-models of real world objects that include deformation response, contact texture and contact sound. During the scanning procedure a robotic arm (supplied with a microphone) moves around the object and applies different kinds of deformation to the object, as well as records the sounds that are produced on contact with the object. The scanned information is then attached to the resulting 3D model and helps to make the interaction with the virtual object as close as possible to the interaction with the real one. Using this technology in combination with haptic devices that allow users to “feel” virtual objects [213] may result in the ability of physical interactive behaviors (i.e. the sensation of touching a fabric) to be easily transmittable through the Web. So, with the help of 3D technology the product presentation on the Web can be advanced almost to the level of product presentation in the real world.

Although, the future of 3D product presentation looks optimistic we are convinced that in order for 3D technology to succeed in commercial applications it has to offer much more than just product visualization. In our opinion, isolated 3D visualization can not satisfy all the demands of the customers and on its own is unable to change the face of E-Commerce. One of the key goals behind using 3D technology in commercial applications is to introduce the beneficial aspects that are present in brick and mortar environments into computer-based solutions. The possibility to fully experience the product in brick and mortar stores is quite important, however, there are many more attributes that contribute to the shopping experience in traditional commercial environments.

One of such attributes is *impulsive decision making* of the customers (or unplanned purchases). Impulsive decisions play a highly important role in traditional commerce. Some researchers even take an extreme view saying that if we would go into stores only when we needed to buy something, and if once there we would buy only what

we needed, the economy would collapse [206]. Such a view is rather too pessimistic, but the importance of unplanned purchases is quite significant and taking it away from traditional commerce would most certainly have a severe economic impact.

Today's economy is extremely vibrant. We have an enormous variety of products present on the market, and with so many information sources available it gets harder and harder for consumers to plan their purchases. That's why many purchasing decisions are made, or can be heavily influenced, on the floor of the store itself. As an example of this, a study conducted by [206] showed that up to 70% of all the purchases in a supermarket are unplanned. This trend is also strong in other industries, i.e. [135] highlights the significance of impulsive purchases in tourism. Despite the existing evidence stressing the importance of the unplanned purchases in brick and mortar environments, there is no clear mechanism present in nowadays E-Commerce systems to support this trend. One of the reasons for this is that the form-based nature of the web sites makes them very limited in presenting large volumes of information. In contrast, the 3D representation in this respect is as rich as the real world and can easily support impulsive purchase decisions of the consumers [43].

Another important set of drawbacks of E-Commerce in its present form is tightly connected with the fact that human beings are "social animals" [178]. The real world is a highly social place and *social interactions* is one of the basic natural needs of human beings [78]. It is false to assume that commercial activities of some kind do not require social interactions. Even such a simple activity as shopping usually involves (and is highly valued for) social interactions with fellow shoppers, sales assistants, clerks etc. These social interactions are not only useful for the exchange of important information about the products but also to satisfy the social needs not directly connected with the product [206]. Social interactions play an important role in traditional commerce and will definitely be an important factor in the future of E-Commerce [162]. They are, in our opinion, as important as product presentation.

One of the most popular kinds of social interactions present in brick and mortar stores is the interaction between buyers and sales assistants. *Sales assistants* there have a function to offer help in a store, provide additional information on products and simplify the decision making process by helping to find a good that satisfies the customer's requirements and various constraints. One of the major drawbacks that E-Commerce is facing today is the lack of such sales clerks. There is strong evidence that in physical stores customers find interaction with a sales person very beneficial [47]. People value and are willing to pay for the reduction of perceived risk, the optimal configuration of the transaction for their specific usage context, and the enhancement of the in-use experience, which shopping assistants can provide them with [47].

The technology that is capable of enriching E-Commerce with social interactions and sales assistants is *3D Virtual Worlds*. 3D Virtual Worlds are immersive environments that combine 3D visualization with the notion of physical presence and have an emergent feature of being highly social. These characteristics make 3D Virtual Worlds a very promising technology for commercial activities in the Internet.

Apparently, 3D Virtual Worlds represent one of the few successful online businesses that are making money on the Web [102]. Millions of people around the world spend an average of 20 hours per week in Virtual Worlds based computer games. Usually those users have to pay a monthly fee of around US\$ 10 just for being able to participate [102]. Moreover, not only are participants paying for the experience of being there, but also to trade virtual goods. Surprisingly enough, in some games virtual items are sold for real money and in many other games the virtual goods are exchanged for virtual money (which still can be easily converted into real world currencies). Virtual items may be quite expensive and range from simple artifacts to virtual properties. A virtual property may end up being nearly as expensive as a real one.

In December 2004 the Project Entropia multi-user game entered the Guinness Book of Records when it sold a virtual treasure island for an equivalent of US\$ 26500 [197]. This was considered the most expensive virtual item at the time and many financial analysts considered this purchase being a very good investment. Just less than one year after this, the sale of a virtual asteroid in the same game for US\$100,000 set a new record for the most valuable virtual item [138].

Every day more and more people join Virtual Worlds and get involved with the variety of virtual trading activities, contributing to the development of the phenomenon of *virtual economies* [38]. The extend of virtual economies can be observed on the example of Norrath, the online world created by Sony, which in 2003 had more residents than Miami and higher per capita GNP than Bulgaria. The inhabitants of the Norrath game literally live and work in the virtual environment of the game. They spend their time and in return receive money, products, services or experiences. The economic study that analyzed a number of different economical parameters [38] has revealed that in 2000 Norrath could have been 77th richest country in the world (roughly equal to Russia).

If selling virtual items in Virtual Worlds is that profitable and widely used, why not sell real items there? Some retailers are already selling their services in Virtual Worlds and some even make a living out of it. Veronica Brown, a fashion designer, earned an equivalent of US\$ 60K in 2005 by running a fashion shop in the virtual environment of Second Life [192]. She created designer dresses for the avatars in the Virtual World and had to quit her real world job to continue doing it on a full time basis. Although, this service is still concerned with virtual items it is actually not any different from all

the other services being offered in the real world. We are convinced that our society is well prepared for the merge of the two economies and will benefit from trading with real world services and goods in the Virtual Worlds. The virtual trading environments are already well populated with enthusiastic customers who are simply waiting for the appearance of new products from the real world.

The major reason for the enormous attractiveness of the Virtual Worlds technology as the facilitator of selling real products is the unlimited number of possibilities they provide to stage experiences for the inhabitants. Staging experiences is highly important today. Our economy has successfully evolved from product economy through service economy to the *experience economy* [160].

This economical evolution can be best observed on an example of a birthday cake. As a vestige of the agrarian economy, mothers made birthday cakes from scratch, mixing farm commodities (flour, sugar, butter and eggs) that together cost mere dimes. As the goods-based industrial economy advanced, mothers started to pay a bit more (around \$1-\$2) for premixed ingredients. Later, when the service economy took hold, busy parents stopped cooking themselves and ordered cakes from bakeries or grocery stores. The cakes there cost about ten times as much as the packaged ingredients. Now, in the experience economy times, parents neither bake the birthday cakes nor even throw the party. Instead, they pay at least ten times the price of the cake to “outsource” the entire event to some business (like Chuck E. Cheese’s, the Discovery Zone, the Mining Company, etc.) that stages a memorable event for the kids – and often throws the cake for free [160].

As shown by the birthday cake example, one of the paradoxes of the phenomenon of experience economy is that experiences today are often much more valuable (expensive) than the products or services they adhere to [159]. The commerce world of today is extremely competitive. Many retailers do not open the stores to create new markets anymore, they open them to steal someone else’s customers [206]. In such tough conditions providing an exciting experience is one of the major factors of success. In many industries like entertainment, business, travel and even health care, more and more experiences today are bought and sold together with products and services or even separately from them. And this trend is going to continue [159].

Virtual Worlds are quite efficient in staging experiences. They are capable of simulating the experience associated with a particular product and can also recreate (and even enhance) the experience associated with the process of purchasing the product. Ultimately, Virtual Worlds extend our experience economy capabilities [80].

Studies in South Korea have recently shown that users prefer Virtual Worlds to television [210], which so far was the most popular experience provider. People visit Virtual Worlds and are ready to spend their time and money there just for the sake of the experi-

ence they receive. Most of the participants would explain that they are so involved with a virtual activity that nothing else seems to matter, the experience itself is so enjoyable that they are ready to do it just for the sheer sake of doing it. Such a state is defined in the literature as the *optimal experience* or the *flow* [52].

The concept of the flow appeared as the result of a study on the topic of what makes people happy. The term flow refers to a state of consciousness that is sometimes experienced by individuals who are deeply involved in an enjoyable activity [52]. According to the authors [52] the state of the ultimate happiness (the flow) can be achieved while having a high degree of concentration on some task and if a person involved with the task is continuously presented with challenges that are up to this person's skills and making the challenges harder reflecting the growth in skills. Video games based on the metaphor of the Virtual Worlds are very good flow facilitators. Flow is one of the key reasons why people enjoy playing such video games and are ready to devote so much of their spare time to visiting the corresponding Virtual Worlds [182].

Many brick and mortar stores are very eager to embrace the flow, because in this state customers enjoy the shopping and are very likely to make a purchase and, more importantly, are more likely to come back to the same store again in a hope to experience the flow once more [188]. It is remarkable that most of the participants of the Virtual Worlds seem to continuously experience this state. While in brick and mortar stores significant efforts are required to generate the flow, in Virtual Worlds it is something that seems to be quite easily achievable. Even the participants of non-gaming Virtual Worlds (where no clear challenges are expressed and no specific tasks are assigned to the users) are still able to experience this state. It is believed that interactivity and immersion, which are integral properties of Virtual Worlds, are the significant flow facilitating factors [97].

The aforementioned benefits associated with the Virtual Worlds technology attracted the attention of many researchers and developers. A group of researchers participating in the international "Metaverse Roadmap" project, focused on predicting the future of 3D Virtual Worlds, highlight the significance of using Virtual Worlds in non-gaming activities [37]. The outcomes of the predictive study conducted within the frame of this project suggest that many of the ordinary activities we today associate with form-based interfaces will migrate into Virtual Worlds. In particular, it is predicted that rapid prototyping, customized and decentralized production, logistics and transportation will soon have a strong presence in Virtual Worlds. The project also stresses the importance of the notion of *mirror worlds*, which are informationally-enhanced virtual models or "reflectors" of the real world. They hypothesize that mirror worlds similar to the currently available Google Earth², but supplied with detailed 3D models of significant buildings

²<http://earth.google.com>

and surrounding objects in the real world, will become an every day reality by 2016. They outline a scenario of a Virtual Town, which contains a virtual replication of popular social locations of the associated real town. In the virtual town users virtually meet each other, efficiently review their joint entertainment options and coordinate entertainment plans. According to the report presented in [37] such scenario will become feasible in 2012.

Another important report highlighting the significance of 3D Virtual Worlds was recently released by Gartner. *Gartner predicts that 80% of the Internet users will be actively participating in non-gaming Virtual Worlds by the End of 2011.* Gartner advises businesses that this is a trend that they should investigate and experiment with [75].

Despite the enormous potential of 3D Virtual Worlds, there is a number of unsolved problems. One such problem is the *lack of a proper methodology*, which would specifically target Virtual Worlds. Most of the Virtual Worlds today are developed on an adhoc basis or utilize methodologies that are not specifically designed for Virtual Worlds and are mostly concerned with the design side of the “look-and-feel” of the inhabited space. Because of this many Virtual Worlds are poorly programmed, hard to maintain and do not offer efficient and safe mechanism to introduce new functionality [111].

Another important drawback of 3D Virtual Worlds is that with its current chaotic nature (which is due to the lack of a proper methodology) there is *no clear mechanism to regulate the interactions among participants in the Virtual Worlds*. The problem of regulating interactions of participants received significant attention after the collapse of the Ginko Financial – the Second Life’s best-known virtual bank [74]. This bank initially promised very high returns (around 44% p.a.) on virtual deposits, but ended up bankrupt, unable to repay approximately USD 750,000 to its investors. Existing mechanisms employed by Second Life proved to be unable to protect the inhabitants from losing their investments and to punish the Grinko founders, despite the fact that many people predicted Grinko’s insolvency straight after it’s appearance [74].

Virtual Worlds are open systems visited by a huge number of people and so designers, developers and users are faced with a number of safety and security risks such as malevolent or unintentional overstraining of computational resources, deviant behavior, destruction of data and code, etc. Having no well established means of interaction regulation is a serious concern for businesses, which are very eager to invest in Virtual Worlds.

Selling real goods and services will immediately attract high volumes of real money, so those risks will become much more costly and the need to regulate the interactions will dramatically increase. Even now many disputes occur in Virtual Worlds, which due to the chaotic nature of the interactions can only be regulated via a post factum

code change by the developers. Such changes usually take a significant amount of time, simply because the Virtual World in question wasn't initially designed following a proper methodology and the interactions of the participants were not well specified in advance. This means that in its current form Virtual Worlds are not yet quite ready to become a widely acceptable platform for online trading of real goods and services.

Both Gartner's report [75] and the "Metaverse Roadmap" report [37] suggest that there is a strong need to establish institutional control over the behavior of participants in Virtual Worlds before active business involvement is possible.

Researchers, who investigate different aspects of *virtual law* [28, 102, 123], are also concerned with this issue. Some authors consider extending the reach of national legal frameworks [136], but also express the concern that such approach is likely to backfire. Not only will national regulation increase competition among Virtual World providers operating in different jurisdictions, but it will also push Virtual Worlds along the same path that the regulation of Napster pushed music sharing – towards a decentralized peer-to-peer model in which providers will eventually disappear and there will be no hope for the real-world law makers to directly influence the governance inside such Virtual Worlds [136]. Other researchers [28] stress the significance of the attempts of Virtual Worlds inhabitants to establish social order (calling this phenomenon *Virtual Institutions*), but also mention that such Virtual Institutions are so far only capable of introducing a set of norms of socially acceptable behavior, but offer no mechanism to enforce these norms.

In this thesis we address the aforementioned problems by extending the notion of Virtual Institutions with a strong conceptual and technological basis. We define Virtual Institutions as 3D Virtual Worlds with normative regulation of participants' interactions and propose the *Virtual Institutions* methodology to be used for their development.

For the control of participants' interactions in Virtual Institutions we use the Electronic Institution [67] technology, which is seen as an application that defines and enforces a set of rules limiting the behavior of participants inside Virtual Worlds. The specification of the Electronic Institution is the initial step of the methodology that helps to clearly define the activities that the participants can engage in and the rules of the interactions amongst them (interaction protocols). With such an approach all of the security aspects of the system are controlled by the institution, which ensures the validity of interactions, meaning that the specified rules can not be violated. Moreover, the proposed approach creates a possibility of quickly changing the rules on demand and even opens the future perspective for inhabitants to have influence over the rule changes and the evolution of the rules.

The proposed methodology is supplied with a number of tools that help in specification, verification and rapid development of commercially-oriented 3D Virtual Worlds.

Some of the implementation steps can be automatically completed once the specification is created. Moreover, there are tools for deployment and runtime verification of the interaction validity.

An important aspect of the implemented infrastructure for Virtual Institutions is the availability of facilities for implicit training of autonomous agents by humans. By implicit training we mean that agents are trained to act like humans by observing them, without any explicit training efforts required from the humans. Such an approach is particularly useful in creating computer-operated sales assistants in virtual environments that can be trained to emulate human-like intelligence. The implicit training concept is one of the central parts of the architecture and we suggest to employ it more often than the explicit programming of the agents.

The motivation presented in this section interlinks a number of different research areas and operates with some terms that might be unfamiliar to a general reader. To provide better understanding, next we present the concept taxonomy of Virtual Institutions.

1.2 Taxonomy of Virtual Institutions

The concept of Virtual Institutions originates from various different areas of Computer Science. In order to illustrate these areas Figure 1.1 outlines the taxonomy of concepts related to Virtual Institutions and positions Virtual Institutions within these fields.

The highest level concept in the taxonomy is the concept of Virtual Environments. Virtual Environments are systems covered by the following definition.

Definition: *Virtual Environments* are imaginary spaces often manifested through a medium [187]. Such spaces may exist solely in the mind of its originator or be broadcasted so that it is shared with others [187].

For the purpose of this dissertation we distinguish between the following two types of Virtual Environments: Normative Virtual Environments (which employ a set of rules to constrain the behavior of their participants) and Immersive Virtual Environments (which offer immersive experience to their participants).

Providing Immersive experience means supporting a high degree of immersion. Immersion is a metaphorical term derived from the physical experience of being submerged in water. We seek the same feeling from a psychologically immersive experience as we do from a plunge in the ocean or swimming pool; the sensation of being surrounded by a completely other reality; as water is from air, that takes all of our attention, our whole perceptual apparatus [143].

There are two different types of immersion:

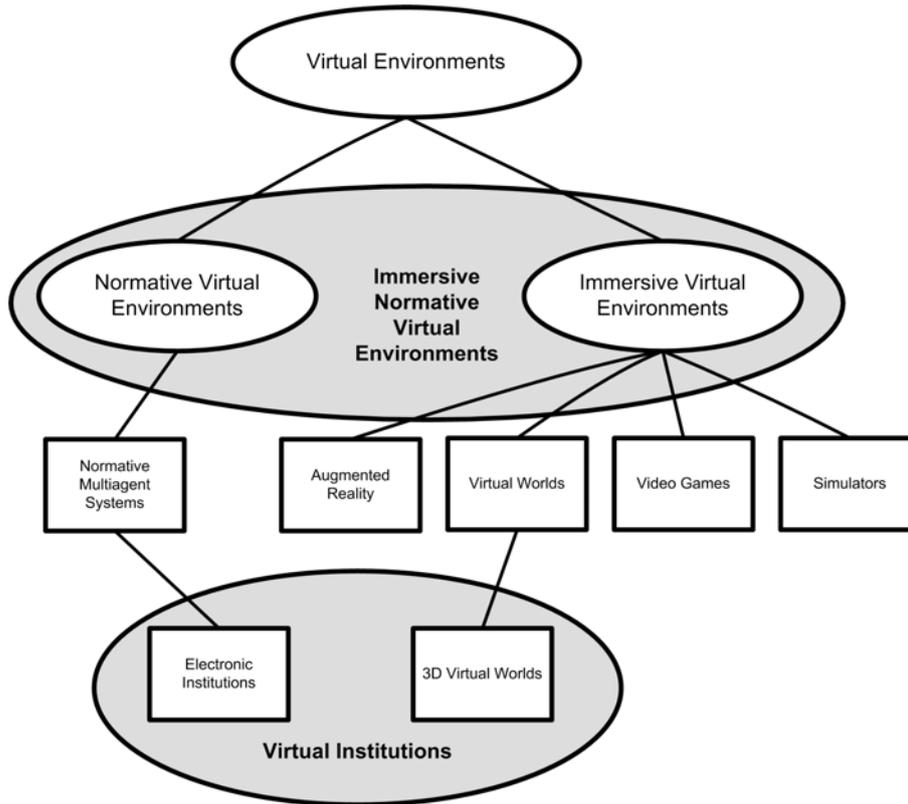


Figure 1.1: Concept Taxonomy of Virtual Institutions

- **mental immersion** state of being deeply engaged; suspension of disbelief; involvement [187]
- **physical immersion** bodily entering into a medium; synthetic stimulus of the body's senses via the use of technology [187]

Both mental and physical immersion can be applied to Virtual Environments. It is possible to mentally immerse a user into an imaginary environment, meaning that the user can be deeply involved with the experience provided by this environment. Physical immersion has a more complicated relation to Virtual Environments. Being physically immersed means that the user is experiencing the environment through available sensors (i.e. able to see, touch, smell the Virtual Environment).

The prime focus of our research is on Immersive Normative Virtual Environments (INVE). This concept originates from the overlap of Immersive Virtual Environments and Normative Virtual Environments. We define INVE as follows:

Definition: *Immersive Normative Virtual Environments are Virtual Environments, which provide participants with immersive experience and employ a set of rules to control the validity of their interactions.*

This thesis is concerned with a particular instance of the Normative Virtual Environments class, called Virtual Institutions. Virtual Institutions originate from the area of Electronic Institutions (a subclass of Normative Multiagent Systems) and 3D Virtual Worlds (a subclass of Virtual Worlds). Further definitions of each of the aforementioned concepts are given in Chapter 2.

Next, we identify the research problem this thesis is addressing and describe the evolution of ideas that happened through the process of conducting this research.

1.3 Research Problem

The initial research problem we faced was: how to enhance E-Commerce with social interactions, support of implicit decisions, advanced visualization of products, collaborative shopping, attracting as many customers as possible and letting them spend as much time as possible in the store, learn personal information about the customers and provide them with a personalized shopping experience?

This problem can be addressed by moving E-Commerce into believable environments closely imitating the real world. We selected 3D Virtual Worlds for this task as we find them being an appropriate technology for supporting the above mentioned features.

But, as we know that Virtual Worlds are missing a clear way to regulate the interactions between participants, the next challenge we had is how to regulate participants' interactions in a 3D Virtual World. We found the answer to this challenge in the area of Normative Multiagent Systems, namely employing the Electronic Institutions methodology for the development of Virtual Worlds.

It is the first attempt to introduce normative regulation of interaction into Virtual Worlds and there is a number of issues we had to take into consideration. Not only a Virtual Institution has to offer both immersive experience and be consistent with a correct execution of the institution, but it also has to ensure that the enforcement of the institutional norms is not violated by the visualization. Based on this, the next challenge was how to establish a close conceptual, as well as technological, relationship between Virtual Worlds and Electronic Institutions? To address this challenge we developed the concept of Virtual Institutions, which was formalized into a number of specification requirements that can be used for the development of the Virtual Institutions technology.

In order to ensure technological consistency of Virtual Institutions and utilize best software engineering practices for their implementation, it was necessary to produce a corresponding methodology. So, the next challenge was how to create a methodology that supports the development of Virtual Institutions and consistently enforces the strict adherence of the visualization to the rules set by the Electronic Institution.

As a result, we developed the “Virtual Institutions Methodology”, described all the necessary steps that have to be followed in the development of Virtual Institutions and proposed a technology to support each of the methodology steps, as well as the deployment of the implemented systems.

The next problem we faced was the agent-oriented nature of the underlying Electronic Institutions technology used in the implementation. Electronic Institutions are usually concerned with autonomous agents, not humans. Direct human inclusion into Electronic Institutions was not properly investigated. Moreover, efficient mechanisms for direct inclusion of autonomous agents into Virtual Worlds are also not available. So, the next challenge was how can the Virtual Institutions methodology, on the one hand, support the direct inclusion of the autonomous agents into Virtual Institutions and, on the other hand, support direct human involvement into Electronic Institutions.

Regarding the human inclusion into Electronic Institutions, one of the conceptual difficulties we faced was that most of the normative environments (including Electronic Institutions) regulate the interactions in a way that every action that is allowed in the system is described in the specification and any other action is prohibited. Such an approach is not acceptable for the development of Virtual Worlds. Most of the features that are automatically supported by most of the Virtual Worlds visualization platforms concerned with Virtual Worlds do not usually require any institutional regulation. Such features include walking, jumping, changing head position, slowing down zooming up etc. Therefore, we decided that for Virtual Institutions we needed a different approach, namely to allow executing any action apart from the actions that Electronic Institution prohibits to execute.

The inclusion of autonomous agents into Virtual Worlds presented another significant challenge. One of the practical motivations behind having autonomous agents as participants of Virtual Institutions was to create autonomous sales clerks that help human shoppers in the store. Such an approach would help to save human resources and, in end effect, would become a significant financial gain for the E-Commerce retailers. The problem is that with the current state of Artificial Intelligence it is not realistic to produce fully intelligent and believable autonomous agents for this task. As a trade off we came up with an idea of implicit training, where humans and agents interchange each other to be able to satisfy all the customer enquires and the agent constantly extends its intelligence by learning from this principle (so that future human involvement can be reduced). The research challenge associated with this issue is how to develop an infrastructure which would support implicit training?

Summarizing all of the above challenges, we formulate the main research problem this thesis addresses:

Problem: *How to build believable and heterogeneous normative environments for commercial activities in the Internet?*

In the context of this research problem the term “heterogeneous environments” refers to environments populated by both humans and autonomous agents.

Next, we present the objectives this thesis sets for solving the research problem and describe the research method that is used to achieve the objectives.

1.4 Objectives

The aim of the thesis is the development of the Virtual Institutions concept and exploration of research issues associated with it.

Therefore, the high level objective of our research is the establishment of the concept of Virtual Institutions. This objective is decomposed into a number of lower level objectives presented below:

- To define and formalize the concept of Virtual Institutions.
- Experimental validation of Virtual Institutions concept.
- To develop a methodology for design of Virtual Institutions.
- To develop a technology that supports a methodology for Virtual Institutions development.
- To provide facilities for programming autonomous agents in Virtual Institutions through training mechanisms.
- To apply Virtual Institutions to the domain of electronic business.

1.5 Research Method

To illustrate how the identified research objectives are addressed and how the research problem is solved we again present the list of objectives and identify corresponding research hypotheses and the research method selected to achieve each of the objectives:

- *To define and formalize the concept of Virtual Institutions.* The research hypothesis associated with this objective is that the concept of Virtual Institutions, as an instance of the Immersive Normative Virtual Environments class, can be successfully

established as a combination of two metaphors: 3D Virtual Worlds and Electronic Institutions. In order to validate this hypothesis the basic exploratory research was conducted, where the reasons for the establishment of the new metaphor are described and the advantages over previous solutions are explored. By means of descriptive research the new concept was formalized (using Z Specification language), the features of Virtual Institutions were summarized and described, and the mapping between concepts in 3D Virtual Worlds and Electronic Institutions was established.

- *Experimental validation of Virtual Institution metaphor.* The research hypothesis associated with this objective is that the formalization of Virtual Institutions is indeed valid and functional. This hypothesis was investigated by means of a prototype that was developed following the requirements expressed in the concept formalization. The prototype provided a proof of concept for Virtual Institutions.
- *To develop a methodology for Virtual Institutions.* The research hypothesis associated with this objective is that a platform independent software engineering methodology for development of Virtual Institution can be created. This hypothesis was validated by means of applied exploratory and applied descriptive research. As a result, a set of practices required for development of Virtual Institutions were collected.
- *To develop a technology that supports a methodology for Virtual Institutions development.* The research hypothesis associated with this objective is that applying the Virtual Institutions methodology can be facilitated through employment of technological facilities. By means of applied descriptive research the basic architecture supporting the execution of the methodological steps, as well as further runtime maintenance of Virtual Institutions were presented. A proof of concept for the technologies used on each of the steps of the methodology, as well as for the deployment architecture was obtained through a number of implemented prototypes.
- *To provide facilities for programming autonomous agents in Virtual Institutions through implicit training mechanisms.* First hypothesis associated with this objective is that it is possible to develop methods for learning from user behavior in Virtual Institutions. Basic descriptive as well as applied exploratory research was conducted to find the appropriate method for co-learning between autonomous agents and humans. On this basis the developed prototype was used to prove the usefulness of the implicit training and to demonstrate how the use of Virtual Institutions can improve the training of autonomous assistants. Applied descriptive research was used to explain how the developed prototype should be improved.

Second hypothesis is that it is possible to develop mechanisms for collection of data about the behavior of the participants of Virtual Institutions. The means of applied exploratory research were used to understand what data should be collected for observation, what the data format is and how this data should be used. The integrated data sources which arise with using Virtual Institutions were determined. A prototype was developed to validate the correctness of the selected mechanism of data collection.

- *To apply Virtual Institutions to the domain of electronic business.* The hypothesis associated with this objective is that Virtual Institutions can be successfully applied to the domain of E-Commerce. This hypothesis was validated by means of applied explanatory research. The detailed literature review highlighted existing evidence and provided new supporting information in favor of using Virtual Institutions in E-Commerce. The analysis of the literature helped to select the most promising application domain for Virtual Institutions within E-Commerce, namely E-Tourism. A conducted user study identified the key issues of E-Tourism that the new technology can solve. The benefits of Virtual Institutions were then illustrated on a number of scenarios associated with the “World Trotter” prototype.

1.6 Contributions and Significance

In this section we show the major contributions of this thesis and highlight the significance of these contributions in various areas of Computer Science.

1.6.1 Contributions

With this thesis we made the following contributions to science.

- Defined the class of Immersive Normative Virtual Environments and its instance – Virtual Institutions.
- Developed a methodology for software engineering and design of Virtual Institutions.
- Provided a technological solution and a proof of concept for each of the methodology steps.
- Developed a formal specification of the Virtual Institutions concept expressed in the Z Specification Language. This specification provides detailed implementation

requirements for Virtual Institutions independent of the actual technologies that are used for implementation.

- Developed a 3-layered deployment solution for Virtual Institutions.
- Developed an implicit training method to support the believability aspects of autonomous agents participating in a Virtual Institution.
- Demonstrated how the Virtual Institutions technology can support commercial activities in the Internet.

1.6.2 Significance

Virtual Institutions is a new research paradigm which significantly contributes to global knowledge in the following research areas: Virtual Worlds, Multiagent Systems, Machine Learning and E-Commerce (E-Tourism in particular). Below we highlight the significance of the thesis contributions for each of the aforementioned domains.

- ***Virtual Worlds***: In this thesis we highlight the similarity of Virtual Worlds with Multiagent Systems, present scientific evidence in favor of using specific MAS methodologies and techniques for the development of Virtual Worlds and demonstrate the advantages of such view of Virtual Worlds.

The concept of Virtual Institutions proposed here, on the one hand, is capable of introducing structured interactions into the chaotic nature of Virtual Worlds. The formal representation of the rules of the virtual society enables automatic transformation of these rules into a desired natural language. This helps in creating an explanation of the rules to human users as well as in describing the reasons why a desired action cannot be performed, together with providing guidelines on possible states and actions in a given situation.

On the other hand, Virtual Institutions allow for semiautomatic generation of the 3D Virtual World visualization from the corresponding Electronic Institutions specification and facilitate runtime maintenance of the generated Virtual World. Such an improvement has a high potential to increase the speed of Virtual Worlds development in the future and make online communities more secure.

Furthermore, to our best knowledge, Virtual Institutions is the first formal methodology for the development of 3D Virtual Worlds.

- ***Multiagent Systems***: Virtual Institutions is a reliable methodology for human-centered institutions, as it doesn't impose limitations on the participants' architec-

ture (as most of the multiagent methodologies do) but restricts the interactions, introduces norms and specifies the common ontology instead. This approach is perfectly suitable for both humans and software agents. Surprisingly enough, software agents so far are mostly considered as the only participants of Electronic Institutions. Combining 3D Virtual Worlds with Electronic Institutions enhances Electronic Institutions with a visual representation and creates a possibility to “open” Electronic Institutions to humans. Visualization of Electronic Institutions as Virtual Worlds also provides conceptual and technological facilities for direct human inclusion into Multiagent Systems.

- **Machine Learning:** The architecture of Virtual Institutions proposed in this thesis enables a novel approach to user modeling and Machine Learning in 3D Virtual Worlds that follow the metaphor of Virtual Institutions. This approach is called “implicit training”. Through implicit training mechanisms autonomous agents can implicitly learn to imitate human behavior inside Virtual Institutions without any explicit training efforts required from the humans. The implicit training is particularly useful for training virtual assistants in various commercial applications.

Through the use of the third dimension and the formal specification of the interaction rules Virtual Institutions open new possibilities for user modeling and learning. More precisely, Virtual Institutions provide better means to observe their participants than 2D solutions. Full immersion and the same embodiment that humans and agents share helps to achieve a more consistent observation of the environment than in form-based interfaces, where most of the human actions are happening outside the environment.

Moreover, limiting possible human and agent actions by the rules of the institution significantly helps to reduce the complexity of the world and knowing the institutional specification may help autonomous agents to execute actions that their principals have never trained them to execute.

In the model we propose each avatar in a 3D space is controlled by either a human or an autonomous agent. When the human drives the avatar the agent learns how to represent human in the 3D Virtual World. The architecture of Virtual Institutions framework automatically supports this feature as the human is always assigned with the corresponding agent and stimulates using implicit training instead of explicit programming of the agent behavior.

- **E-Commerce:** The conducted study, literature review and developed prototypes present important evidence in favor of using Virtual Institutions in E-Commerce.

The 3D representation of products in E-Commerce that was a buzz word of late nineties proved not to provide enough benefit for the users to accept this technology. Our research suggests that social interactions, support of impulsive decisions, collaborative shopping, advanced marketing, easy and non-intrusive customer behavior analysis as well as training of virtual assistants are the key benefits of 3D shopping, which were missed by previous researchers.

Detailed literature review and a conducted user study identified online travel agents as the most promising domain for application of Virtual Institutions.

1.7 Structure

The remainder of the thesis is structured as follows:

Chapter 2 outlines the main components of the problem domain: Virtual Worlds and Electronic Institutions and their key characteristics. It also presents detailed motivation for using Electronic Institutions in conjunction with 3D Virtual Worlds and highlights the need for employing formal methodologies for Virtual Worlds' development.

Chapter 3 presents the new metaphor developed in this thesis – Virtual Institutions. The main logical components of this metaphor are identified and the concept is illustrated on the example of the Trading Institution. The detailed proposal for implementation is expressed in terms of Z-Specification language.

Chapter 4 describes the actual methodology that should be applied to the development of Virtual Institutions and suggests a technological approach that could be utilized on each of the steps of this methodology.

Chapter 5 examines the relationship between humans and autonomous agents in Virtual Institutions. It is shown how the architecture developed in Chapter 4 facilitates implicit training of autonomous agents. The implicit training method is verified through a set of experiments, which test the possibility to teach autonomous agents imitating human-like movement with no explicit training efforts required from the humans.

Chapter 6 is concerned with applications of Virtual Institutions technology and explores how this technology can be used in E-Commerce. The drawbacks of existing E-Commerce solutions are identified and it is shown how Virtual Institutions can address them. Based on the acquired knowledge and detailed literature review, E-Tourism is selected as the sub domain of E-Commerce that has the highest potential for application of Virtual Institutions. The advantages of using Virtual Institutions in E-Tourism are illustrated on the example of the World Trotter travel agency. This example also illustrates how user requirements can be analyzed and how the specification of a Virtual Institution fulfilling these requirements can be derived from the results of the analysis.

Chapter 7 presents some concluding remarks and directions of future work. In particular, in the future work section we identify the most promising areas of research that arise with Virtual Institutions and suggest how these should be approached.

Chapter 2

Background

As the previous chapter has introduced, this research is concerned with structuring the interactions of participants inside 3D Virtual Worlds. Below we present some background information that helps in detailed understanding of the Virtual Worlds concept and the reasons for the importance of structuring the interactions of participants inside them. We suggest seeing Virtual Worlds as Multiagent Systems, and show how through the application of the Electronic Institutions Methodology it is possible to establish the interaction rules and ensure their enforcement. The concept of Electronic Institutions, its relation to Multiagent Systems as well as the corresponding methodology are explained in details and illustrated on an example of the Trading Institution.

We start the explanation by introducing the concept of Virtual Worlds.

2.1 Virtual Worlds

The concept of virtual worlds is synonymous to the concept of Virtual Environments defined in Chapter 1, Section 1.2. This concept is quite broad and covers all kinds of imaginary spaces. One example of such a space could be a theater play, where the script and enacting the script by the actors are two different ways to broadcast the virtual world to others.

As it was previously described, in this thesis we are concerned with a particular instance of this broad class. To produce a more accurate definition of the area we are focused on, we will now introduce some additional terms.

2.1.1 Interactivity

Some of the virtual worlds support interactivity. Such environments differ from theaters or cinemas, where audiences are just passive spectators, by allowing users to actively participate in the experience and have influence over this experience.

Interactivity comes more readily with the addition of computers to the equation [187]. With the use of computers it is possible to design environments that would respond to

user commands, giving a participant a feeling of being an active part of these environments. One of the simplest forms of interactivity that can be achieved via computer intervention is changing the user's viewpoint. This functionality provides users with a possibility to observe the environment from different positions and under different view angles.

Although computer graphics is not necessarily required for interactivity, the most popular interactive environments are supported by this technology. Within the environments visualized via computer graphics facilities the interactivity is usually achieved through keyboard and mouse clicks. It involves changing directions, picking up objects and acting upon these objects.

2.1.2 Collaboration

Another important feature available in some virtual worlds is collaboration. Collaboration is the act of working together on a common task or process [22]. It is related to interactivity, however collaboration is explicitly concerned with interactions between people.

To be able to collaborate in imaginary environments people need certain facilities to communicate with each other. Depending on the media used to represent the virtual world different kinds of collaboration facilities can be utilized. Collaboration can happen in written form, audio form, video, interactive 3D etc.

2.1.3 Definitions and Terminology

All the aforementioned features must be present in the type of environments we are focusing on in this thesis. Therefore, we define "Virtual Worlds" as follows:

Definition: *Virtual Worlds are immersive imaginary spaces supporting sensory feedback, visualized through computer simulations and designed for their users to inhabit, collaborate and interact.*

To distinguish between different definitions of virtual worlds in this thesis we use the following notation. Where we talk about "virtual worlds" (no capital letters) we refer to the very general concept, which is synonymous with Virtual Environments. Where "Virtual Worlds" are used (in capital letters) we refer to a particular class of virtual worlds covered by the former definition.

The key differences between Virtual Worlds and other similar environments are outlined in [191]. According to the authors the following features of Virtual Worlds are important:

- The ability to support multiple users differentiates Virtual Worlds from standard virtual reality and game engines
- The ability to share and manipulate Virtual World objects differentiates Virtual Worlds from traditional chat rooms.
- The ability to support real-time interactions differentiates Virtual Worlds from email services and traditional web browsing.

To be able to see each other and collaborate, the inhabitants of Virtual Worlds are usually represented in the form of avatars. Avatars are defined as follows:

Definition: *Avatars are one, two or three-dimensional graphical representations of humanoids [187].*

One of the most popular kinds of Virtual Worlds available today is the class of 3D Virtual Worlds. The key characteristic of 3D Virtual Worlds is that they are visualized using 3-dimensional computer graphics. A precise definition is given below.

Definition: *3D Virtual Worlds are multiuser Virtual Worlds designed using the metaphor of architecture and visualized using 3-dimensional computer graphics.*

In 3D Virtual Worlds the world being computer-simulated typically appears similar to the real world, with real world rules such as gravity, topography, locomotion, real-time actions, and communication. However, 3D Virtual Worlds are often supplemented with features not present in the physical world, like teleportation¹ or flying.

Immersion in 3D Virtual Worlds is supported by representing users as embodied avatars. Such avatars are usually 3-dimensional models of humanoids that have an operational set of all body parts. Participants navigate through a Virtual World by manipulating their avatars and making them physically move inside the simulated environment.

Collaboration in 3D Virtual Worlds has, until recently, been mostly conducted in the form of text. The interface of each Virtual World is usually supplied with a chat window, through which participants can communicate with each other. Some Virtual Worlds solutions offer mechanisms to select the recipient of the textual message; others try to closely simulate the real world only allowing avatars in close proximity to the speaker to receive the message. Many of the recent Virtual Worlds platforms also utilize real-time voice communication through VOIP (Voice Over IP) [99].

3D Virtual Worlds bring new terminology into computer-mediated environments. While in form-based interfaces humans are often referred to as “users”. In 3D Virtual

¹The process of moving objects/avatars from one place to another instantaneously, without passing through the intervening space.

Worlds (mostly due to their computer games origins) humans are called “players”. In this thesis we suggest a new name: “participants”. With this term we want to stress that the application of 3D Virtual Worlds in this work extends beyond the area of computer games. Additionally, we are not only concerned with “human-players” but also take into account the fact that software-driven avatars may also actively participate. The word “participant” is thus used for both humans and software-driven players.

Autonomous software-driven components of computer generated environments are studied in the area of Distributed Artificial Intelligence [108]. There such components are called “agents”. To maintain consistency with this terminology we also use the word “agent” to refer to software-driven participants of 3D Virtual Worlds.

In 3D Virtual Worlds the participants (both humans and agents) are embodied as avatars and can collaborate with each other in an environment which closely replicates the real world. The fact of embodiment is quite important for human beings, and next we explain some of the reasons behind this importance.

2.1.4 Benefits of Using Avatars

Through the use of avatars 3D Virtual Worlds challenge the disembodied paradigm of the Internet by focusing on the importance of the body [28]. Having a body is important for our brain to better understand our actions as well as actions of the others. This issue is carefully studied by neurobiologists, who have recently discovered the phenomenon of “mirror neurons” [109].

Mirror neurons were discovered by accident while recording the brain activity of a primate who grabbed a banana. During the experiment one of the researchers grabbed a banana as well. The primate’s neurons fired in a similar pattern if the monkey itself was grabbing a banana. In a sense the primate’s brain was treating the actions of any embodied primate in a similar way as its own actions [109].

Neurobiologists believe that mirror neurons not only help the brain to identify and cognitively recreate the local actions of primates in the environment, but also help to predict the intended actions of others based on visual cues. The fact that mirror neurons fire regardless of whether one’s body or an outside body performs an action suggests that primates cognitively internalize the actions, akin to guessing “what behavior would I be doing if I were this individual?” [109, 107].

Mirror neurons are thought to be located in the part of the brain where vocalization patterns are controlled as well. For humans this has an obvious advantage: we can give words to the actions (and the meanings) that we perceive. What either our body or the body of someone else does – we can predict, identify and name [109, 107].

The absence of a body distracts from shared group experiences [28], while its pres-

ence (even in a form of an avatar) stimulates such activities. From body representation forms currently available in the Internet only video can compete with 3D representation. Video, however, can only be used for a very limited number of participants and requires a high bandwidth as well as additional equipment. 3D avatars have much richer possibilities in this respect. Hundreds of avatars can simultaneously participate in a conversation and the required bandwidth for such a conversation would be much smaller than it is required for a video conference between two people.

Another big advantage of using avatars is anonymity. Many people online do not feel comfortable with giving away their true identity. Being hidden behind an avatar helps them to protect it. Moreover, when selecting avatars, participants visualize their ideal-self image (or alter ego) [193, 187], that is, how participants would like to see themselves. This image can potentially convey even more information about a participant than their regular real world appearance. Such information is useful for establishing trust amongst participants and for a better understanding of their preferences.

The use of avatars stimulates shared group experiences, facilitates social interactions, allows for more natural observation of human actions and helps to achieve the feeling of being in the World Wide Web rather than on it. While most of the Virtual Worlds present today are 3-dimensional, some of them are not. In such Virtual Worlds the avatars may be visualized as 2-dimensional cartoon characters or even text symbols. Of course, the possibilities to benefit from mirror neurons are much lower there, which is probably one of the reasons why 3D Virtual Worlds are dominating today. Despite this fact, some of the one-dimensional and two-dimensional Virtual Worlds are still actively used. To complete the presentation of the Virtual Worlds metaphor it is necessary to explore all of the available varieties of Virtual Worlds and demonstrate the evolution of the concept.

Next, we show how the concept of Virtual Worlds and the supporting technologies changed over time and provide a detailed description of the popular Virtual Worlds and their features.

2.1.5 Historical Overview of Virtual Worlds

The idea of what is currently called a Virtual World was first presented in the science fiction novel *Neuromancer* [77]. In this novel, the term *cyberspace* was introduced as “a consequential hallucination experienced daily by billions of legitimate operators in every nation... a graphical representation of data abstracted from the banks on every computer in the human system”. Another novelist [198] described a shared world called “metaverse”, which allows participants to globally connect through computer-mediated interfaces and interact virtually.

The technology supporting Virtual Worlds started to appear in the 1980’s. At the

early stage Virtual Worlds were implemented as multi-user text based virtual environments called MUDs and MOOs.

MUD stands for Multi User Domain. MUDs usually served as places for role-playing games. Some of the early MUDs ran on a bulletin board system, while others used an Internet server. The Virtual World representation was fully text driven. During the game session the players would read the descriptions of rooms, objects, events, other characters, and computer-controlled creatures or non-player characters (NPCs) in a virtual world. To interact with the game and with each other players would type commands in a natural language. “Dungeons and Dragons” game is one of the most popular MUDs. Plato², one of the first games that can be considered MUDm appeared in 1977.

MOO is the abbreviation for MUD Object Oriented. These more recent text-based worlds were enhanced with objects: participants bodies and objects around them and verbs for activating behaviors. MOO, along with all of its nephews, started out with text based adventure games. With the advent of the Internet, MUD was formed as a networked version of one of those games. Eventually it developed into three different types of MUDs, one of them being MOO. The first version of the MOO server, called “AlphaMOO”, was released on May 2 1990. One of the most popular MOOs was developed by Pavel Curtis, an employee of Xerox PARC, who took the basic design, language, and code of Alpha Moo, fixed bugs and added features to release the first version, called “LambdaMOO” on the 30th October 1990 [215].

After the text-based MOOs and MUDs next step in Virtual Worlds evolution were graphical games. Habitat [140] was the very first networked virtual world in which people were represented as avatars able to communicate and form a “virtual community”. It started running on Commodore 64 computers back in 1985. The essential lesson from the experience with Habitat is that Cyberspace is defined more by the interactions among the actors within it than by the technology with which it was implemented.

Today, 3D Virtual Worlds have replaced MUDs, MOOs and 2-dimensional graphical Virtual Worlds and became the most frequently seen forms. This type of Virtual Worlds is now particularly common in massively multiplayer online games (The Sims Online³, There⁴, Entropia Universe⁵) and in massively multiplayer online role-playing games (World of Warcraft⁶, EverQuest⁷, Ultima Online⁸, etc.).

Along with game-based 3D Virtual Worlds other forms also started to appear. In

²<http://en.wikipedia.org/wiki/PLATO>

³<http://thesims.ea.com/>

⁴<http://www.there.com/>

⁵<http://www.entropiauniverse.com/>

⁶<http://www.worldofwarcraft.com/>

⁷<http://everquest.station.sony.com/>

⁸<http://www.uoherald.com/>

the environments like Active Worlds or Second Life there is no clear goal set for the participants and it is not required to collect points or fight virtual opponents. People enter such Virtual Worlds to socialize and experiment with their creativity. They invest time and money just to share the experience of being there with others. Many researchers (i.e. [38, 37]) predict that such Virtual Worlds are a new phase in the evolution of software systems. Due to the fact that with 3D Virtual Worlds an individual may easily access a far broader set of experiences as well as vastly larger social network, it is suggested that many of the activities that we associate today with 2D Web will migrate into such Virtual Worlds [37]. It is believed that tasks like E-Commerce [43], rapid prototyping [37], logistics [37], online education [101] etc. will be widely conducted in 3D Virtual Worlds in the future.

To understand the possibilities non game-based 3D Virtual Worlds may provide we will outline key enabling technologies and major features of such Virtual Worlds.

2.1.6 Enabling Technologies

In this section we do not intend to present a comprehensive overview of all of the existing Virtual Worlds platforms. Instead, the goal of this section is to select the most outstanding technologies that are not created around a game scenario, but are self-sufficient. In contrast to game-based Virtual Worlds the technologies we focus on are not structured around an expressly developed plot or storyline, but allow for general purpose scenarios.

VRML

The beginning of the Virtual Worlds era is tightly connected with VRML. The term VRML was coined by Dave Raggett in a paper submitted to the First International Conference on the World-Wide Web in 1994. It stands for Virtual Reality Modeling Language. As a technology it offers a standard file format for representing 3-dimensional interactive vector graphics, designed particularly with the World Wide Web in mind [217].

VRML is a text file format where each of the objects is specified as a set of its attributes and parameters. For example, the vertices and edges of a 3D polygon can be specified through their coordinates along with surface color, image-mapped textures and other attributes.

To give an impression about VRML format an example of a shape description is given below:

```
#VRML V2.0 utf8
```

```
# Red cone
```

```

Shape {
  appearance Appearance {
    material Material {
      diffuseColor 1 0 0
    }
  }
  geometry Cone {
    bottomRadius 0.75
    height 1.6
  }
}

```

In order to view VRML files, it is necessary to install one of the VRML browsers currently available on the market. The most popular browsers are “Cortona VRML Client” and “Cosmo Player”.

Initially, VRML language (VRML 1.0) was limited to the development of static, single-user 3D scenes. There was no possibility to see other users and interact with them, as well as there was no possibility to interact with the visualized objects. These problems were addressed in the release of the VRML 2.0 language that was announced in 1997. In this version it became possible to [183]:

- include a system of events to communicate among VRML objects;
- insert programs by: Java, JavaScript and VRML scripts;
- establish a connection between a VRML world and an external program.

The announcement of VRML 2.0 was the height of VRMLs popularity. At that time VRML was used on some personal home pages and sites such as “CyberTown”, which offered 3D chat functionality.

The format had been further supported by SGI’s Cosmo Software, however, after a restructure in 1998 the division was sold to Platinum Technologies and VRML was no longer supported or developed.

One of the reasons for VRML’s lack of success was that its rendering capabilities remained largely the same, while real-time 3D graphics kept improving and other platforms were able to better keep up with this trend. Because of this, many users abandoned VRML and moved to Virtual Worlds with more appealing visual characteristics. To illustrate the rendering capabilities of the VRML browsers, Figure 2.1 shows a participant inside the CyberTown Virtual World.



Figure 2.1: Inside the Cybertown Virtual World

After an extensive period of inactivity VRML Consortium eventually changed its name to the Web3D⁹ consortium, and began to work on the successor to VRML X3D [217].

While VRML still enjoys widespread use, particularly in education and research where an open specification is most valued, it has now been superseded by X3D. The MPEG-4 Interactive Profile (ISO/IEC 14496) was based on VRML (now X3D), and X3D is largely back-compatible with it. VRML is also still used as a file format for the interchange of 3D models, particularly from CAD systems [217].

Active Worlds

Active Worlds¹⁰ (AW) is one of the oldest 3D Virtual Worlds platforms. It was first created in 1994. Since this time the management structure of Active Worlds has changed several times and it is currently operated by Activeworlds Incorporated.

AW was one of the first technologies where members were highly involved in developing the world by building objects that visually enhance each world's setting but appear to do little else. Unlike many other platforms the Active Worlds platform is not focused on futuristic themes but is full of modern-day cities with a new age utopian twist. It features quite an international place with many worlds based on real-world countries. All the participants are represented as avatars, can chat with each other and build complex objects from basic primitives.

⁹<http://www.web3d.org>

¹⁰<http://activeworlds.com>

To enter Active Worlds, participants launch a web-based application – “AW browser”. Through this application they log into the Active Worlds Virtual World universe, and explore the environments that other people have built.

Figure 2.2 presents the interface of the AW browser.

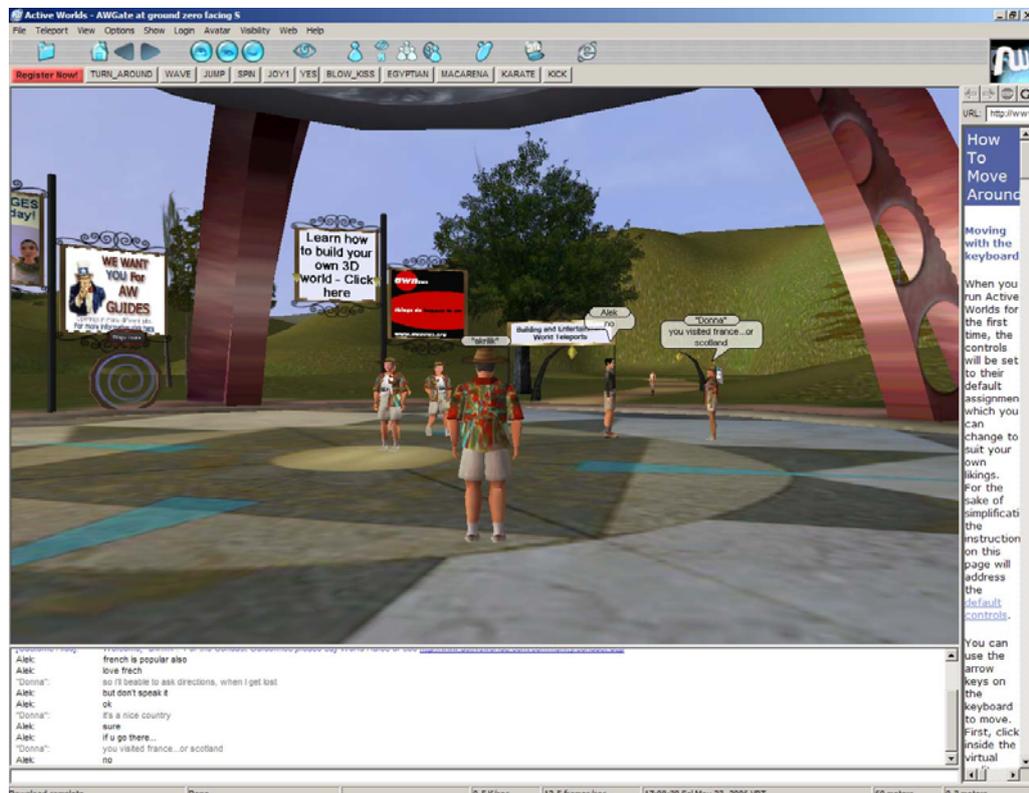


Figure 2.2: An Environment of the Active Worlds Universe: Example

The AW browser provides two ways of entering its universe: as a free tourist or as a paid citizen [14]. Tourists have limited capabilities, they can not enter some of the worlds, can not reserve a name for their avatars and the content they create can be deleted. Paid citizens are allowed to own worlds and universes, use various communications facilities (such as voice chat, file exchange etc.), use personalized avatars with reserved names. The content they create can not be deleted.

A unique feature of Active Worlds are autonomous agents called *bots*. These agents are created using the AW Software Development Kit. Some bots allow participants to automate some tasks such as weather and chat relay [28]. Each citizen is allowed to employ up to three such bots.

Active Worlds' world organization is similar to the real world. “Meter” is the measurement unit inside this virtual environment. The world is divided into “logic” squares, each measuring 10 x 10 square meters. There is a limit of how many objects one can

place within a “logic” square. Navigation is based on the world directions and coordinate value within each direction. East, West, North and South Orientation (facing direction) follow the four main directions and four intermediate coordinations. To support synchronization of geographically dispersed communities, Active Worlds employ Virtual World and real world time simultaneously.

One of the widely used features of Active Worlds is the inclusion of inline web links into virtual spaces and reverse teleports into worlds. With their help, instant movements through virtual space can be achieved through clicks on the links. Scripting languages and the AW Software Development Kit can be used for enabling complex behaviors of objects and avatars.

As with the majority of 3D Virtual Worlds to reduce the internet traffic, most of the costly computations occur on the client side. The server constantly transmits the basic packet: (x, y, z, heading, velocity, communications) and each of the clients performs the rendering.

Adobe Atmosphere

Adobe Atmosphere is a 3D visualization platform originally developed by Attitude Software. In November 1999, Adobe Systems purchased the technology. The product spent the majority of its lifetime in beta testing. The last version of Atmosphere, version 1.0 build 216, was released in February 2004. In December 2004 the software was discontinued [214].

Atmosphere distinguished itself from pre-existing technologies like VRML and ActiveWorlds in several ways. Unlike VRML, which is focused on the language for creating three-dimensional “models” and wasn’t really concerned with anything else, Atmosphere focused on explorable “worlds” (later officially called “environments”), which were linked together by “portals”, analogous to the World Wide Web’s hyperlinks. These portals were represented as spinning squares of red, green, and blue that revolved around each other and floated above the ground [214].

Portals were indicative of the Atmosphere team’s desire to mirror the functionality of webpages. Although the world itself was described in the .aer (or .atmo) file, images and sounds were kept separately, usually in the GIF, WAV or MP3 format. Objects in worlds were scriptable using a modified version of JavaScript, allowing a more immersive environment, and worlds could be generated dynamically with the help of PHP. Using JavaScript, a world author could link an object to a webpage, so that a user could, for example, launch a webpage by clicking on a billboard advertisement (Ctrl+Shift+Click in earlier versions). By version 1.0, Atmosphere also boasted support for Flash animations and WMP movies [214].

Atmosphere-based worlds consisted mainly of parametric primitives, such as floors, walls, and cones. These primitives could be painted a solid color, given an image-based texture, or made “subtractive”. Invisible, “subtractive” primitives could be used to cut “holes” in other primitives, to build even more complex shapes. Many worlds also contained animated polygon meshes made possible by Atmosphere’s implementation as a subcomponent of Viewpoint Corporation’s Viewpoint Media Player. However, Viewpoint stopped supporting the Atmosphere subcomponent some time prior the discontinuation of Atmosphere [214].

Unlike the more centralized structure of ActiveWorlds, in which environments are primarily built within AlphaWorld, Atmosphere worlds were spread throughout the Internet, usually hosted on the author’s own website as .aer files. (The .aer format was later used solely for building, once the binary .atmo format was created.) As with ActiveWorlds, the user navigated an avatar; in later builds, an option allowed the user to see his or her own avatar. With three-dimensional wireframe views and multiple floating tool windows, the Atmosphere Builder was reminiscent of both CAD software and professional graphics applications like Adobe Photoshop [214].

Figure 2.3 shows the interface of the Atmosphere Builder.

In ActiveWorlds it is only possible to communicate with users within a 200-meter radius, whereas Atmosphere users could chat with all the users in the world. This was more appropriate for Atmosphere, considering the smaller sizes of most worlds. Technically, users could chat with anyone in the same YACP¹¹ channel, a reference to the IRC¹² protocol. The exception to this is when worlds would receive too many visitors, as was often the case at HomeWorld¹³: worlds would “clone”, creating duplicate channels for the same world, which would often cause confusion for users. Some world developers wrote scripts that limited communication to users within a certain distance, to create an effect of enhanced realism [214].

A built-in Havok physics engine, detailed rendering, and dynamic lighting also contributed to the realism of Atmosphere worlds. Many world authors wanted to create large worlds, in order to build more realistic cities, for example, but such worlds would often take an excessive amount of time to load in the visitor’s web browser, especially if the visitor was using a slower dial-up connection. To alleviate this issue, Atmosphere supported loading sections of the world subworlds or models as they became needed; a city could then be loaded block by block, rather than all at once. One of Atmosphere’s problems, however, was excessive memory usage, which was exacerbated by the use of advanced features such as embedded models and Flash movies in its many worlds [214].

¹¹Yet Another Chat Protocol

¹²see <http://www.irc.com/>

¹³A public Virtual World created by Adobe.

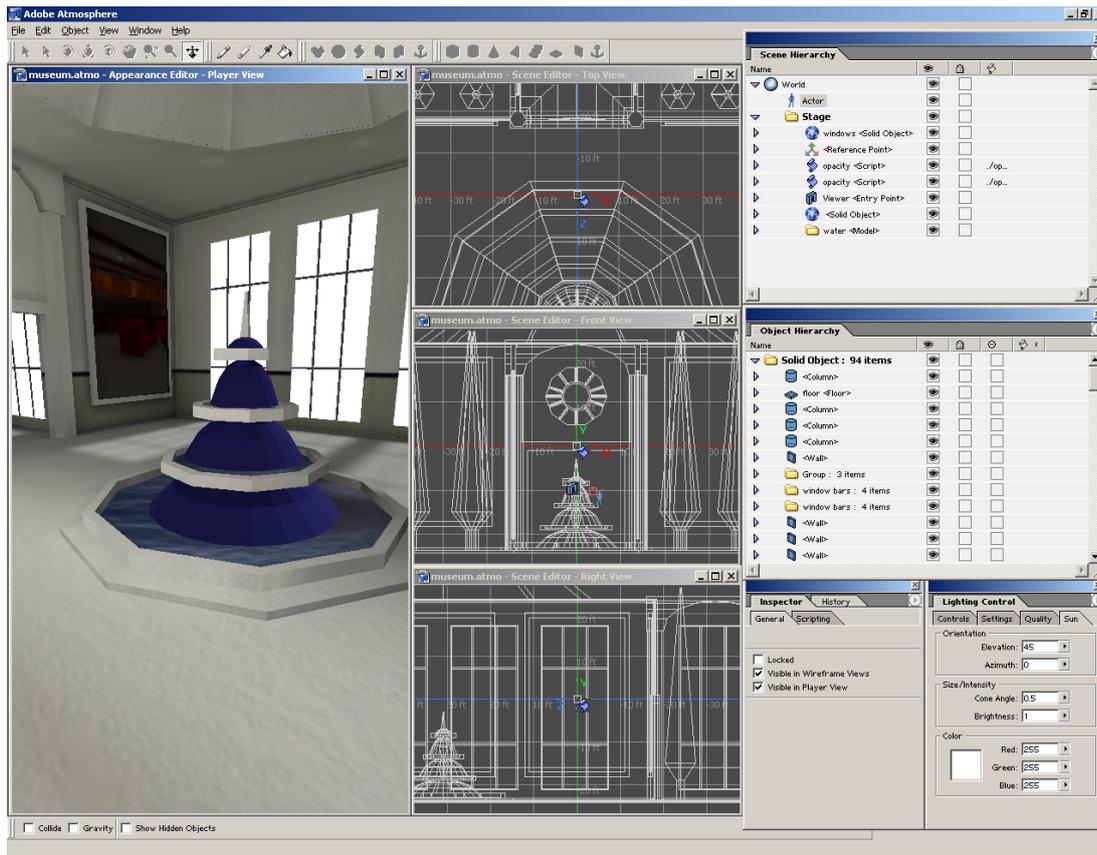


Figure 2.3: Adobe Atmosphere Builder

Second Life

Second Life¹⁴ is one of the most popular non game-based 3D Virtual Worlds platforms available today. It was launched in 2003 by Linden Lab as a partly subscription-based Virtual World.

Despite the fact that its rendering engine is not as effective as in Adobe Atmosphere it gained enormous popularity. As of November 2006, there were over 1,700,000 registered participants of Second Life, and the world's population is growing at a rate of 30% monthly [28].

The approach taken by Linden Lab is different from ActiveWorlds and Adobe Atmosphere. Instead of being distributed amongst private parties the world of Second Life is owned and maintained by Linden Lab; no private Second Life servers are allowed.

A downloadable client program enables its participants, called "Residents", to interact with each other through motional avatars, providing an advanced level of a social network service combined with general aspects of a metaverse described in the novel

¹⁴<http://secondlife.com>



Figure 2.4: Inside the Virtual World of Second Life

“Snow Crash” by Stephenson [198]. Residents can explore, meet other residents, socialize, participate in individual and group activities, create and trade items, virtual property and services from one another [216].

Figure 2.4 demonstrates the Virtual World of Second Life.

Similar to Active Worlds, residents can be of two types depending on the type of their Second Life account: basic and premium. Basic account users have no recurring fee, but lack the right to own land within Second Life. Premium account users paid US\$9.95 per month, as of February 2007 [216]. Premium members can own land (up to 512 m^2 without additional fees). Owning larger areas of land incurs additional fees ranging from US\$5 a month up to US\$295 a month for an individual island [216]. Premium accounts receive a weekly stipend (paid in Linden dollars) which somewhat offsets the membership payment. This stipend has reduced with time; as of February 2007, this was at L\$300 per week [216]. Basic account users registered before 29 May 2006 receive a stipend of L\$50 for every week in which they log into Second Life, but no stipend is provided to basic accounts registered after that time [216].

Within Second Life, there are two main methods of text-based communication: local chat, and global “instant messaging” (known as IM). Chatting is used for public localized

conversations between two or more avatars, and can be “heard” within 25 m. Avatars can also “shout” (“audible” within 96 m) and “whisper” (“audible” within 18 m). IM is used for private conversations, either between two avatars, or between the members of a group. Unlike chatting, IM communication does not depend on the participants being within a certain distance of each other. Voice communication is currently in beta testing [216].

The most basic method of moving around the Second Life environment is by foot. To travel more rapidly, avatars can also fly up to about 170 m over the terrain (meaning 270 m if ground level is 100 m) without requiring any special equipment, and with scripted attachments, there is currently no limit to how high an avatar can fly (although once past several thousand meters, the rendering of the avatar mesh starts to be affected) [216].

Avatars can also ride in vehicles. Many vehicles are available in the object library and there are many resident-made vehicles available freely and for purchase including helicopters, submarines and hot-air balloons. Airborne vehicles can fly up to about 4000 m high (the maximum altitude allowed for any object) [216].

For instantaneous travel, avatars can teleport (commonly abbreviated to “TP”) directly to a specific location. An avatar can create a personal landmark (often called “LM”) at their current location, and then teleport back to that location at any time, or give a copy of the landmark to another avatar. There’s also a map window that allows direct “teleportation” to any other location [216].

There are some external web sites that allow Residents to locate each other from outside of the virtual world, and SLurl.com allows external links through the Second Life World Map to locations in-world [216].

One of the distinguishing characteristics of Second Life is that the residents, not Linden Lab, create most of the content of the world. There is a 3D modeling tool in Second Life that allows any Resident with the right skills to build virtual buildings, landscape, vehicles, furniture, and machines to use, trade, or sell. This is a primary source of activity in the Second Life economy. Any Resident can also utilize gestures from small animations and sounds from the standard library. Outside Second Life, Residents can use various graphics, animation, and sound tools to create more elaborate items, and upload them into the world. Once the creation is in the world of Second Life, the system makes efforts to help protect the exclusive rights of the content creator [216].

Second Life also includes a scripting language called Linden Scripting Language (LSL). LSL is used to add autonomous behavior to many of the objects in Second Life, such as doors that open when approached. LSL has been used to create relatively advanced systems, such as the artificial life experiment on the island of Svarga, where a complete ecology runs autonomously (featuring clouds, rain, sunshine, bees, birds, trees and flowers) [216].

Objects created by residents can be either donated to the world or can be sold for “Linden Dollars”. Linden Dollars (Linden, or L\$) are exchangeable for US Dollars in a marketplace consisting of residents, Linden Lab and real life companies [216].

A Resident who creates an item and the Resident that owns an item may retain certain rights, rather like copyright in the real world. The creator can mark an item as “no copy”, which means that no copies of it may be made by others, “no mod”, which means that others may not modify the item’s characteristics, and “no trans”, which means that the current owner may not give it to another [216].

In 2006 Linden Lab started a massive promotional campaign, positioning Second Life as a business platform of the future. It was heavily advertised in the popular media: television, radio, newspapers and magazines. As the result of this campaign Second Life began to attract the attention of all sorts of businesses and non profit organizations.

Reuters has created a news agency in Second Life. Recruitment companies, escort agencies, games developers, and clothing designers are also well established there [158].

Live music performances take place in Second Life on a regular basis, in the sense that vocal and instrumental music by Second Life Residents can be provided from their homes and studios. The sound is recorded via microphones, uploaded to audio streams, and played in-world for the enjoyment of other Residents. Popular bands like Duran Duran and Suzanne Vega often perform in Second Life [216].

There is a large virtual community of artists and designers. They use Second Life not only as platform to demonstrate their art from real life, but also to express themselves and create new (virtual) art. The modeling tools from Second Life allow the artists to create new forms of art, that in many ways are not possible in real life due to physical constraints or high associated costs. The virtual arts can be viewed, for example, in the Second Life Louvre, a virtual representation of the Louvre Museum [216].

On September 22, 2007 Maldives opened the first embassy in Second Life (Figure 2.5) [150]. Just 8 days later Sweden did the same [149]. Macedonia and Philippines have also announced their intention to follow this trend [150]. The visitors of existing virtual embassies are able to talk face-to-face with a computer generated ambassador about visas, trade and other issues, but no real or virtual visa services are offered so far.

Second Life has also recently emerged as one of the cutting-edge virtual classrooms for major colleges and universities, including the Open University (UK), Harvard, Vassar, Pepperdine, University College Dublin, Elon University, Ohio University, Ball State, New York University, University of Houston, Stanford University, Delft University of Technology and AFEKA Tel-Aviv Academic College of Engineering. Second Life fosters a welcoming atmosphere for administrators to host lectures and projects online, selling more than 100 islands for educational purposes [216].



Figure 2.5: Maldives Embassy in Second Life

Major corporations have also taken notice of the marketing and training opportunities provided by this Virtual World. Technology companies like IBM and Cisco Systems are investing time and money to create an environment for their employees and potential customers [116]. IBM uses Second Life to conduct corporate events and job interviews. Ultimately, IBM hopes to lower programmers' travel expenses by conducting meetings and training sessions on the Second Life's islands [116].

An interesting innovation proposed by IBM is replication of the Australian Open championship in Second Life. The data is taken from the real world tournament through the employment of game-tracking devices on the field. These devices plot shots and ball trajectories and clock the players' serves. Based on the acquired data the positioning of the ball inside the virtual stadium is calculated. The avatars in Second Life can then simulate strokes made by the players. The spectators can pick the seats and be a part of the virtual audience or can choose to watch the game from a player's perspective [104].

Second Life is also known for its virtual economy [38]. An average of US\$600,000 is exchanged amongst Second Life's virtual characters daily [28]. Around US\$100,000 exchanged into L\$ at the official Second Life currency exchange portal called Lindex [28]. These money are further spent in the Virtual World for virtual items and services.

To manage the virtual money banks have also been established. Meta Bank is a Virtual Bank that operates entirely in Second Life. Clients of this bank can deposit or withdraw virtual funds and even borrow some money from the bank to buy virtual property. One of the real world banks "Wells Fargo" has a presence in Second Life,

though its focus is on teaching virtual participants the value of saving and maintaining a good credit rating in the real world [28].

To show the scale of the virtual economy of Second Life below are some numbers extracted from the official statistics¹⁵ provided by Linden Lab. As of Monday, July 23, 2007 there are 8,329,767 total residents in Second Life. Out of all the residents there were 305,551 participants who actively spent money in Second Life in July 2007 and 1,031,293 users who logged-in at least once in July 2007. The exchange rate in July 2007 was around L\$268 per US\$1. The total of US\$6,790,904 were exchanged into L\$ in June 2007.

Comparison of Technologies Supporting Virtual Worlds

Next we illustrate the key features of the aforementioned technologies and highlight the differences between them. Before continuing with the presentation we would like to point out the key conceptual difference that sets VRML apart from the rest of the presented technologies. VRML – is a language standard for description of 3D models. While there are some standard tools available for the visualization of the models described using this language as well as there is standard software for supporting other features like multiuser collaboration, chat etc. – this software is not controlled by VRML community (Web3D) and practically anyone can develop it. In contrast, the rest of the technologies are fully fledged and are owned and controlled by their developers.

Table 2.1 outlines the key differences between VRML, Active Worlds, Adobe Atmosphere and Second Life.

Table 2.1: Technologies Supporting Virtual Worlds: Feature Overview.

	VRML	Active Worlds	Adobe Atmosphere	Second Life
Rendering	Poor	Average	Very Good	Good
Standardized	Yes	No	No	No
Central Hosting	No	Partially	No	Yes
Object Building	Hard	Easy	Moderate	Easy
Dynamic Building	No	Yes	Rare	Yes
Model Import	Good	Average	Good	No
Supported Architecture	Any	Client-Universe/World Server	Client-Server	Client-Server Grid
Client Software	Open Source	AW Browser	Web Browser + Plug-in	SL Client
Space Limit	No	No	No	Yes
Level of Control	Full	High	High	Partial

First comparison criterion is the rendering quality¹⁶. VRML as a stagnating technology is the worst performer in this respect. Active Worlds’s rendering is also not very impressive, in particular the visualization of the avatars is not very good. The quality of rendering in Second Life is quite reasonable, but despite the popularity of this platform

¹⁵http://secondlife.com/whatis/economy_stats.php

¹⁶Talking about rendering (as well as about other characteristics) in context of VRML we refer to the rendering quality of the popular VRML browsers.

and enormous development efforts put into it, the quality is still lower than of Adobe Atmosphere. In particular, Adobe Atmosphere platform is known for its lighting (which Second Life doesn't support).

Only one of the four technologies is a standard. This technology is VRML.

Comparing by the "central hosting" criterion we analyze whether the Virtual World visualized by the technology in question can only be visited through the central server of the hosting company or not. Only Second Life is fully hosted by the Linden Lab and doesn't support external servers. Most of the Active Worlds Universes and Worlds are also hosted by the developer, but there are private Universes and Worlds too. Adobe Atmosphere and VRML based Virtual Worlds can be hosted on any server.

Another important characteristic is how hard it is to build objects using the tools provided with a particular technology. In general, creating 3D models with VRML is done by specifying them in terms of text. Practically, there are third-party graphical tools available, but they are not as good as the tools supplied with other technologies. For development of Virtual Worlds Adobe Atmosphere provides Atmosphere Builder, a tool capable of producing very sophisticated designs. Active Worlds and Second Life offer in-world building facilities, which are more primitive than Atmosphere Builder, but are also much easier to use.

The unique feature of Active Worlds and Second Life is the possibility of dynamic building of the objects inside the Virtual World. This feature is most valued by the virtual communities, as dynamic building can be conducted by a number of people simultaneously and the result is immediately visible. Adobe Atmosphere doesn't directly support it, but with the help of additional software (like AtmoKits) dynamic building can also be achieved. VRML has no support of this feature.

Only Adobe Atmosphere and VRML support all the major 3D formats and can import most of the models from them. A more limited number of import formats is supported by Active Worlds. Second Life, to our best knowledge, doesn't provide import of external models, but some efforts in this direction are being made.

There are no particular guidelines on software architecture for platforms visualizing VRML. Most of the visualization tools are supplied with the server application (which controls the movements of participants and distributes chat messages between connected clients) and a client tool (which constantly communicates with the client to update the state of the Virtual World and visualize the updated state). Similar Client-Server architecture is employed by Adobe Atmosphere. Each Virtual World visualized with Adobe Atmosphere is usually supplied with a private server. Active Worlds employ the concept of a Universe (which is a server that contains a list of other servers hosting individual Virtual Worlds) and a concept of a World (a server fully controlling one Virtual World).

Most of the Universes and Worlds are hosted by ActiveWorlds Corporation, but any private party is also allowed to host their private server. Second Life follows a different approach. It uses a large number (grid) of servers, each of which is responsible for visualizing a particular part of the Second Life Universe. One server is physically controlling a virtual space of a given volume and no private servers are allowed.

All 4 technologies require installing special software to visualize their worlds. In the case of Adobe Atmosphere this software is simply a downloadable plug-in for the standard web browser (Internet Explorer), while other technologies need their own browser.

VRML, Adobe Atmosphere and Active Worlds do not apply any limitations on the space that a Virtual World can occupy. Second Life, due to its specific architecture, has the space of the Virtual Worlds limited by the number of servers that are currently used for the visualization of the Second Life Universe.

With the employment of VRML developers have full control over the Virtual Worlds they create. They can use any technology they like to visualize the objects and a private server to control the interactions of participants. The developers employing Adobe Atmosphere and Active Worlds can use private servers, but have to rely on the visualization software they are supplied with. Second Life only provides some control over the physics engine and weather for the developers building their objects on private islands, but neither private servers nor custom visualization software can be used there.

The comparison of the characteristics of the Virtual Worlds enabling technologies highlighted the diversity of the contemporary Virtual Worlds solutions in terms of their features. No single Virtual Worlds platform is optimal in terms of the features, none of them dominates the market and each of the technologies has unique and promising characteristics. It is hard to predict which of the aforementioned platforms is going to survive the competition and which ones are going to fail. Hence, an optimal solution to the development of Virtual Worlds is to come up with an approach that remains technology independent on as many development levels as possible, as well as to offer mechanisms for quick replacement of the selected visualization platform on demand. Next, we focus on analyzing the development of the Virtual Worlds, discovering the unique engineering trends associated with them and investigating the necessity of having a general (technology independent) methodology supporting the development of Virtual Worlds.

2.2 A Need for a Methodology

Virtual Worlds today have already extended beyond the computer game environments. With versatile activities currently happening there it is necessary to ensure that Virtual Worlds are developed based on best software engineering practices. In contrast, the over-

all design and development of Virtual Worlds has emerged as a phenomenon shaped by a home computer user rather than by research and development activities at universities or companies. As a result, Virtual Worlds are more or less unregulated environments and continue to be developed on an adhoc basis.

Developing a Virtual World application is a rather difficult task and is often more challenging than building a form-based application. The main difficulty lies in the management of complexity. The complexity arises from the fact that the application developer must “design” three things that are interrelated, namely, “form”, “function” and “behavior” at the same time. Form refers to the outer appearance of virtual objects, their structure (for composite objects) and the scene structure of the Virtual World. Other physical properties (which may be required for physical simulation) such as mass, material property, velocity, acceleration may be a part of form information. *Function* basically refers to encoding of what virtual objects do to accomplish their behavior (either autonomously or in response to some external stimuli or event). *Behavior* refers to how individual virtual objects (including participants) dynamically change and carry out different functions over some period of time, usually expressed through states, exchange of data/events, and inter-object constraints [111].

Using adhoc approach for such complex environments results in unstructured code, which is expensive, unreliable and takes a long time to implement . The technique widely used in software engineering for avoiding this kind of problems is employment of software methodologies. Software methodologies are codified sets of practices that may be repeatedly carried out to produce software. With their help, the development becomes predictable, easily distributable and the resulting code is usually of higher quality [111].

To our knowledge no widely acceptable methodology has been proposed for the development of Virtual Worlds. Because of this many Virtual Worlds are poorly programmed; the information regarding function, behavior, and constraints among them are all mixed in together, and not readily visible for easy future maintenance and reuse [111].

The closest attempt to address this issue was made by [111]. Authors came up with a set of useful methods that could potentially be employed, in particular for the modeling of form, function and behavior of the virtual objects present in the Virtual World. Unfortunately, neither a methodology, which would structure the use of these methods, nor appropriate software tools for automating the application of the described methods were offered by the authors. Furthermore, the proposed methods are only concerned with the behavior of virtual objects and ignore the behavior of participants. We see the inability to regulate the behavior of participants as one of the most important problems Virtual Worlds face today. Having no methodology that could help to solve this problem is a significant drawback in Virtual Worlds design.

We will now explore possible solutions and build the argument in favor of using a formal methodology in Virtual Worlds development. We shall commence by providing more specific details about the use of methodologies in the field of Computer Science.

2.2.1 Methodologies in Computer Science

Edsger Dijkstra's controversial letter "GoTo Statement Considered Harmful" sent in 1968 to the editor of the Communications of the Association for Computing Machinery (CACM), as well as his famous book "A Discipline of Programming" [58], have revolutionized our way of thinking about software development. The proposed idea of structured programming started the ongoing discussion on the need for software methodologies in Computer Science. As a result of this discussion, which has developed into some kind of a "religious war" between academics and practitioners, the Software Engineering field was born, with many formal methodologies from the academic world ending up being ignored by the majority of software developers. The "battle of giants" still continues, however the overall number of people relying on informal methodologies like 4M [32] and unified notations such as UML [177] for creating their software significantly outweighs the number of followers of the formal methodologies [205].

Formal methodologies employ the same principles of structured programming [58] as UML and 4M. The difference is that the formal methodologies use *formal methods*, which in software design comprise two main activities: formal specification and verified design, that is the precise specification of the behavior of a piece of software, so that such software can be subsequently produced, and the proof of whether the implementation complies with the specification [49].

The reason why programmers are reluctant to use formal methodologies are as follows: most of these methodologies have too many rules and practices [179, 137, 2]; those rules are too complex, hard to follow and not well understood [212]; the amount of documentation is way out of control [205]. Having the decreasing of the development time as one of the main objectives, sophisticated methodological tools became so time consuming that programmers have to "cut corners" to meet the schedule targets. Moreover, the majority of formal methodologies do not provide developers with adequate frameworks, forcing them to use obscure notations and primitive CASE tools instead of convenient IDEs [205].

Despite the denial by the majority of the developer community, formal methodologies are catching up in the area of life critical software applications (airplane control, army defense etc.), where errors in code can cost people's lives. For such software it is infeasible to compensate for possible faults and uncertainties in its design, so the selected methodology has to guarantee beforehand that the software "does the right thing".

There it is crucial to increase the level of understanding of the system by reducing its complexity and by employment of mathematically based formal languages for system specification together with theorem provers for the verification of its correctness [179]. The use of formal methodologies does not a priori guarantee correctness, however, it significantly helps in revealing inconsistencies, ambiguities, and incompleteness that might otherwise go undetected [49].

Apart from the verification of the code, formal methodologies offer some additional benefits. With the use of formal methodologies it is possible to abstract away from the way the system is programmed, but still be precise about some aspect of its behavior. Of course, informal methodologies do the same, but with them it is impossible to say precisely whether or not a real system satisfies the description. In contrast, when formal methodologies are employed one can perform precise analysis on the description itself, knowing that any conclusions one comes to, will be true of the real system [59].

One value of this level of precision is that it exposes design decisions which otherwise might not be noticed until the system is being implemented. It is clear in many systems that obscure interface behavior could not have been designed that way, but has occurred as the result of a specific programming decision. The specification of an interactive system should not determine the algorithms and data structures used to build the system that is the proper domain of the programmer. But, it should describe precisely the behavior of the system, while the programmer may not be qualified to make such decisions and the level of commitment at the time that the issue is uncovered may mean that the design choice has been determined by foregoing implementation choices [59].

2.2.2 Methodologies for Virtual Worlds

Although it looks like only critical systems still continue to utilize formal methodologies, we advocate that Virtual Worlds should also apply them. The main reason behind this statement is that only formal methods and methodologies are capable of handling the growing complexity of interactions inside Virtual Worlds and, at the same, time guarantee the correctness of these interactions.

Despite the fact that active support of human interactions is one of the key characteristics that sets Virtual Worlds apart from other technologies, there are currently almost no technological facilities available to control these interactions.

As the number of inhabitants in the artificial societies established in Virtual Worlds constantly grows, the level of immersion increases and the participants become more and more involved with the experience. As a result, the need for structuring their interactions becomes more explicit. This need is identified by researchers [37], who argue that existing Virtual Worlds lack a clear mechanism of expressing the interaction rules, and stress

that techniques for accountability of the participants to these rules are strongly required.

As the drift away from games to commercial applications continues in Virtual Worlds, the degree of interaction and the complexity of these interactions dramatically increases. Big and small businesses moving into Virtual Worlds and offering their products and services there will soon employ quite complex protocols for interaction with customers [37]. With direct money involvement the validity of these protocols is at stake.

Even now, there is already constant pressure on the developers to hardcode the interaction rules into their Virtual Worlds by changing the programming code of the environment. Doing so, however, in a system that was built without a methodology centered on structuring users' interactions is a very challenging task. Moreover, such changes would require terminating the execution of the Virtual World and restarting it after the changes have been introduced, which for many businesses is unacceptable. This is one of the reasons why Virtual Worlds are currently used on a very limited set of domains (mostly in computer games), where structuring the interactions of participants is not necessarily useful and the consequences of errors in the code are not significant. In order to extend the scope of Virtual Worlds technology to be applied to a wider range of problems, exploit the benefits of Virtual Worlds and deal with their growing complexity, methodologies regulating the interactions of participants and improving reliability and security issues need to be applied.

Informal methodologies are unable to cope with complex interactions and ensure their validity. Techniques, like simulation and testing used for the validation of the code in such methodologies can only explore *some* of the possible behaviors of the system. If the system supports complex interaction protocols – running comprehensive tests and simulations becomes extremely difficult. Developers in this case can never be sure whether an unexplored execution paths may contain fatal errors. The existence of such errors in commercial applications may have significant consequences [155].

In contrast, formal methodologies usually employ formal verification mechanisms, which exhaustively explore *all* possible behaviors of the system and prove their validity. Formal verification is a very attractive alternative to simulation and testing. If the corresponding methodology is supplied with convenient specification and verification tools it is an obvious choice for complex systems with a high degree of interaction.

As Virtual Worlds lack such a methodology we decided to search for it in other domains. One of the domains which is conceptually very close to Virtual Worlds is the domain of *open systems* [96]. Similar to Virtual Worlds, open systems usually require a high degree of interaction and a high level of security. Social and visualization issues are at stake there and support of efficient collaboration between participating components is necessary. In [96] open systems are defined as follows:

Definition: *Open Systems* are software environments whose components are unknown beforehand, can change over time and can be either humans or software entities developed by different parties.

Such open systems have been carefully studied by researchers working in the area of Distributed Artificial Intelligence (DAI). We suggest viewing Virtual Worlds in the DAI context and adapt the formal methodologies available in this area to the development of Virtual Worlds.

2.3 Virtual Worlds as Multiagent Systems

As the name suggests – DAI is a subset of Artificial Intelligence. The main goal of the Artificial Intelligence (AI) field is to create *intelligent agents* [180]. An agent is defined as follows:

Definition: *An Agent* is anything that can be viewed as perceiving its environment through sensors and acting upon this environment through actuators [180].

The concept of an agent in AI simply provides the context for research, while the primary focus of this discipline is on making agents *intelligent*. In contrast, the notion of an agent is central for Distributed Artificial Intelligence. In this subfield of AI the agents are normally seen as software entities and their most important characteristic is autonomy rather than intelligence. Therefore, in DAI context the agents are often referred to as “autonomous agents”. The following definition presents the DAI view on the notion of agents.

Definition: *Autonomous Agents* are computational entities such as software programs or robots that can be viewed as perceiving and acting upon its environment and that are autonomous in that their behavior at least partially depends on their experience within the environment [211].

Autonomous agents often act on behalf of humans (or a human). Such humans are called “principles”.

Definition: *Principle* in DAI context is a human whose requirements the corresponding agent tries to fulfill by acting in the software environment on human’s behalf.

While in other subfields of Artificial Intelligence an agent is often mentioned in singular form, in DAI this form is almost never used. DAI is not aiming at achieving human-like intelligence and is rather concerned with studying various aspects of interactions of autonomous agents. Therefore, a single agent is rarely interesting for DAI

scholars and most of DAI research is concerned with Multiagent Systems. Multiagent Systems are defined as follows.

Definition: *Multiagent Systems* (or MAS) are systems in which several interacting, autonomous agents pursue some set of goals or perform some set of tasks [211].

Having introduced the concept of Multiagent Systems we can now define DAI.

Definition: *Distributed Artificial Intelligence* (DAI) is a study, construction, and application of Multiagent Systems [211].

The most important question DAI is trying to answer is when and how should agents interact to successfully meet their objectives. There are two different approaches that DAI researchers take in answering this question: agent-centered (bottom up) and system-centered (top down) approaches. The bottom up approach, which dominates the DAI community, aims to search for specific agent-level capabilities that result in appropriate interaction at the overall group level [211]. The top down approach deals with searching for specific group-level rules – conventions, norms, and so on – that appropriately constrain the interaction repertoire at the level of the individual agents [211].

The Multiagent view is perfectly suitable for open systems (as well as for Virtual Worlds based on this metaphor), the components of which are also independent and autonomous. However, the design and implementation of open systems introduces some additional complications that must be addressed:

- *Heterogeneity.* Open systems must be capable of accommodating heterogeneous software and human agents representing different parties, with different purposes (objectives) and preferences, and endowed with varying degrees of sophistication [172].
- *Reliability.* Open systems must be reliable. They must be designed so that failures of individual components can be accommodated by operating components while the failed components are being repaired and replaced [12].
- *Accountability and legitimacy.* Since the inhabitants may possibly exhibit either deviant or fraudulent behavior, their actions must be monitored in order to detect and prevent dysfunctional behaviors which may jeopardize the overall functioning of the system. Thus, only those actions that are regarded as legitimate should be permitted [172].
- *Societal change.* The societies in open systems are not static; they evolve over time by altering, eliminating or incorporating rules. Hence, there is a need for demanding flexible societies capable of accommodating future changes [172].

In general, agent-centered approaches assume that participants are known beforehand, along with agents' benevolence and cooperation. These assumptions clearly contradict the above-mentioned complications making agent-centered approaches unfeasible for open systems. That's why, an important part of research in Open Multiagent Systems followed the top down approach, where it is not the internal architecture of participants, but the rules and structure of the interactions between them, that are modeled. This means that a system is designed as a set of limitations, which impose restrictions on the behavior of participants, while their internal architecture is not a part of system design. Only such a solution permits the inhabitants to behave autonomously and to make their decisions freely up to the limits imposed on them by the system.

Specifying all those limitations and protocols requires the use of formal methodologies and formal languages. The class of open systems that employ these is called Normative Multiagent Systems (Normative MAS).

Definition: *Normative Multiagent Systems are sets of agents whose interactions are norm-governed. The norms in this case prescribe how the agents ideally should and should not behave. Importantly, the norms allow for the possibility that actual behavior may at times deviate from the ideal, i.e., that violations of obligations, or of agents rights, may occur [24].*

Two of the most prominent methodologies for normative MAS are: Gaia [220] and Electronic Institutions [67]. Gaia proposes to build the system as a set of organizations and stresses that open systems must be aware that some participants can show self-interested or competitive behaviors. This problem is clearly expressed, but the methodology itself does not commit to any mechanism to deal with this issue. Another problem with Gaia is that it only covers the specification of the system and is not concerned with the rest of the development process. In contrast, Electronic Institutions methodology suggests a clear solution to dealing with self-interested participants, namely that an open Multiagent System requires an internal mediator between participants that verifies the validity of their actions against a set of rules, protocols and norms specified by systems designers. It also facilitates the whole development process (from specification to the deployment) and is supplied with the set of graphical tools [7] for each of the methodological steps. Thus, we consider Electronic Institutions as a better choice than Gaia.

Electronic Institutions are thought to play the same role in virtual societies as traditional institutions do in human societies. Electronic Institutions define a set of rules which establishes what participating agents are permitted as well as forbidden to do.

As Virtual Worlds reflect human societies, the use of institutions is an obvious choice there. Before elaborating the technical details of the Electronic Institutions methodology,

we provide additional motivation for needing to employ institutions to regulate the behaviors of the participants inside Virtual Worlds as well as give a detailed description of the notion of institutions.

2.4 Need for Institutions in Virtual Worlds

The concept of institutions is something that has existed for a very long time. In human societies there are situations in which individuals interact in ways that involve commitment, delegation, repetition, liability and risk. This is particularly true for the whole range of commercial activities, where each individual is trying to maximize his/her own utility function and minimize the utility of their opponent. In such situations the participants are mostly self-interested, non-benevolent and often non-reliable. No assumptions can be made about their cooperative behavior. If some degree of trust can't be achieved, participation in such interactions becomes too risky and rational participants would refuse to take part in them. One of the mechanisms used in human societies to control the interactions involving self-interested participants is employment of trusted third parties, which establish the *rules* of the interactions and administer the strict *control* in regards to their *enforcement*. In the literature, such trusted parties are called institutions [147] and are defined as follows.

Definition: *Institutions are structures and mechanisms of social order and cooperation governing the behavior of two or more individuals [147].*

Institutions are used to regulate the interactions amongst the participating individuals by enforcing strict rules, norms and conventions on their behavior [184]. The establishment of the institutions helps in reducing the complexity in the decision making of the participants as well as in increasing the trust between individuals [184].

The most prominent example of an institution is a government. The rules it tries to enforce are various laws present in the country the government is representing.

Virtual Worlds represent an alternative to real world human societies and reflect many of their components. They require the use of institutions to the same (or possibly even higher) degree as human societies in the real world. In its current form Virtual Worlds are rather anarchical environments with a few rules suggested by the developers of the corresponding software. These rules are usually expressed in the terms of services of a particular Virtual World. Developers usually try to enforce rules themselves by issuing warnings to the rule violators or even banning them if the violation reoccurs [117].

It is even harder with Virtual Worlds that allow self organization, where users create their own content. In such environments there are no clear mechanisms for allowing the

users to easily establish the institutional rules in relation to the visualization that they have created.

In reality most of the rules within human societies, which help establish socially acceptable behavior of human beings are called social conventions. Social conventions are a fundamental precondition for the stability, efficiency, and inner coherence of a social system [17]. There exists a distinction between implicit and explicit social conventions [17]. In the case of *explicit conventions* the rules are articulated by explicit agreements, or even laws, which have been established by institutions or responsible persons. The *implicit conventions* are not explicitly stated in any form and are learned by the individuals as the part of the common sense and cultural background. Social rules function not only for comprehension but also for coherence within a social system by establishing a common context and a common normative basis for its participants [17].

According to a research study conducted by [17], the participants of Virtual Worlds are successful in establishing implicit social conventions, as most of them share similar cultural background. There is, however, no clear mechanism present in the Virtual Worlds that helps in establishing explicit social conventions. In its present form the violation of the implicit community rules requires constant moderation from either trusted members of the community or the publishers of Virtual Worlds, while violations of explicit conventions are transferred to real world courts.

Many participants of Virtual Worlds expect that the rules that govern fraud in virtual communities are explicitly set in the terms of service or end-user license agreement of the Virtual World they are participating in, but it is not usually the case. One of the main reasons for it is that contract law cannot regulate participants' interactions with each other [203]. Because of this there is not much a publisher can do to stop such behavior. Fraud could be prevented from happening at the design phase; but as it was already said, there are no technological mechanisms available for establishing interaction rules in Virtual Worlds.

Similar to the real world, the participants of the Virtual Worlds desire security and justice [134], and have requested more clearly anticipated lines of demarcation, but publishers often reply that this is too hard to incorporate it into their products [203].

One of the reasons why the developers are reluctant to introduce the interaction rules into their environments is that human institutions seem to evolve over time, so that the rules change very quickly. Supporting rule evolution in software is very hard to achieve without formal description of the institutional rules and a clear separation of the specification of the institutional rules from the rest of the development cycle.

Although, Electronic Institutions (as well as other MAS oriented methodologies we are familiar with) do not consider Virtual Worlds as the area of application they do have

a high potential to address the problem of interaction regulation in Virtual Worlds. They provide mechanisms for formal description of the interaction rules and, more importantly, their validation. The rule specification phase is clearly separated from the rest of the development cycle. The tools supplied with Electronic Institutions help to conduct rule specification and validation very efficiently. Furthermore, using the provided technological apparatus it is possible to quickly change the interaction rules without a need to modify the program code of the developed system.

Next, we outline the metaphor of Electronic Institutions and the corresponding methodology in more detail.

2.5 Electronic Institutions

In human societies there are two possible rule enforcement mechanisms available: preventive measures, which prevent the rule violation from happening; and sanctions, which are used to punish the rule violator [71]. While all the possible efforts are put into developing preventive measures, the environment humans are living in is so complex that it is physically impossible to prevent all the rule violations from occurring. Therefore, much of the rule-control mechanisms used in human societies are sanction based. The rule enforcement mechanisms of the government, for example, mostly include the employment of armed forces like police or the army.

In this thesis we are concerned with open societies embedded into computer mediated environments. In such societies it is much easier to control the interactions of participants as the complexity of such environments is much lower than the complexity of the real world. For such societies it is more efficient to employ preventive rule enforcement measures rather than use sanctions. One of the possibilities of preventive rules enforcement is provided by Electronic Institutions.

Definition: *Electronic Institutions* are software systems composed of autonomous entities, i.e. humans or autonomous agents, that interact according to predefined conventions on language and protocol and that guarantee that certain norms of behavior are enforced. This view permits that participants of Electronic Institutions behave autonomously and make their decisions freely up to the limits imposed by the set of norms of the institution. An Electronic Institution is in a sense a natural extension of the social concept of institutions as regulatory systems which shape human interactions [172].

Electronic Institutions are created following the Electronic Institutions Methodology [67]. In this methodology the development is divided into two basic steps:

1. *Formal specification* of institutional rules.

2. *Execution* via an infrastructure that mediates agents interactions while enforcing the institutional rules.

The formal specification focuses on macro-level (rules) aspects of agents, and not on their micro-level (participants) aspects. The infrastructure is required to be of general purpose so that it can interpret any formal specification.

2.5.1 Interactions of Participants in Electronic Institutions

In Multiagent Systems most of the agent interaction is considered to be conducted through speech acts [185]. As independent software-driven components agents have limited capabilities to interact in any other way, as other agents have unknown architectures and sometimes different embodiments. Therefore, most of the Multiagent theories consider communication as the only form of agent interaction [211] and employ the speech act theory [185] as the basis for interaction.

Speech act theories are pragmatic theories of language (i.e. theories of language use). They attempt to account for how language is used by people every day to achieve their goals and intentions. As it was noticed by [11], some utterances are rather like “physical actions” that appear to change the state of the world. More generally, everything we utter is uttered with the intention of satisfying some of our internal goals. The theory of how utterances are used to achieve intentions is the speech act theory. The key approach taken by this theory is that each of the utterances consists of two components:

- A performative verb; and
- Propositional content

A performative verb explicitly contains the type of action the “speaker” is trying to achieve by pronouncing the propositional content. In human communication such a separation is not required, as the performative verb can be easily recognized from context, non verbal clues etc. In agent communication such separation is necessary to be able to correctly interpret the utterances of the opponent.

To illustrate both components of an utterance, below are three possible utterances a teacher may “send” to a student in the classroom:

1. request(“Door closed”)
2. inquire(“Door closed”)
3. inform(“Door closed”)

While the propositional content (“Door closed”) is always the same, the action that is expected from the student is different in every case. In (1), the student is requested to close the door. In (2), the expected result is to answer the question as to whether the door is closed or not. In (3), there is no result expected as the door is already closed and the goal is to inform the student about it.

Communications of the participants of Electronic Institutions are based on the speech act theory. It is conducted in the institution by participants exchanging utterances with each other. In the terminology used with Electronic Institutions such utterances are called *illocutions*. Each of the illocutions consists of two parts: illocutionary particle (performative verb) and message content (propositional content).

In contrast to other communication platforms used in the DAI community (i.e. FIPA ACL, KQML) Electronic Institutions do not limit the set of possible illocutionary particles. System designers are free to use as many as they feel necessary for the scenario that they are trying to implement.

A detailed explanation of illocutions is given in Section 2.5.4.

2.5.2 Specification

After presenting various aspects of participants’ interactions in Electronic Institutions we shall continue by explaining the way these interactions are specified.

The notion of institutions introduces a dramatic difference to the development of computer mediated systems compared to the majority of present solutions. Instead of focusing on the implementation details of each participant, a system-oriented view is taken. We assume that participants may be heterogeneous and self-interested, and we cannot rely on their correct behavior. Therefore, the institution is designed as a set of limitations, which every participant has to comply with. The aforementioned limitations apply along the following four dimensions [67, 172]:

- Conventions on language, the *Dialogical Framework*.
- Conventions on activities, the *Performative Structure*.
- Conventions on interactions, the *Scenes*
- Conventions on behavior, the *Norms*.

Next we describe each of them in details.

Dialogical Framework

This dimension determines which language, ontology and illocutionary particles agents should use and the roles they are supposed to play within the institution.

As Electronic Institutions deal with open societies it can not be predicted who is going to be participating, how many participants will be present in the institution and what are they going to do there. To be able to deal with this uncertainty each of the participants is assigned with a role. The basic functionality for each of the roles contributes to fixing the organizational structure of the society of agents (that is, which roles agents can play, and which incompatibilities and relationships exist among the roles).

There are two basic types of roles: internal roles and external roles. Internal roles are played by internal agents (institutional employees) that represent the institution. These roles can not be played by the visitors of the institution. External roles are played by external agents (visitors of the institutions).

Performative Structure

This dimension determines in which types of dialogues agents can engage.

Each different activity an agent may perform is associated to a dialogue among the group of agents involved in that activity. These (structured) dialogues are called scenes. (The term is borrowed from theater scripts, where actors incarnate characters and follow strictly pre-fixed dialogues).

The Performative Structure fixes which protocol (possible dialogues) can be enacted in each scene, which sub-language of the overall institutional language can be used in each scene, and which conventions regulate the in and out flux of agents in scenes.

Agents can join and leave scenes at certain points of the dialogue, for instance, at the end of a round in an auction scene. These points are marked as exact states in the finite-state machine that represents the protocol.

Arcs of the finite-state machine are labeled with illocutionary patterns that make the conversation state evolve when correctly matched agent illocutions are uttered.

Finally, the minimum and maximum number of participants is limited by the specification of scenes. Scenes are interconnected to form a network in order to represent a sequence of activities, concurrency of activities or dependencies among them. Agents leave scenes where they have been playing a given role and enter other scenes to play the same or a different role. This transit of agents is regulated by the special type of scenes called *transitions*.

Transitions are responsible for re-routing agents. They are also places where synchronization with other agents (if needed) occurs. Sometimes new scenes can only be

enacted by a group of agents, or agents can only join scenes as members of a group. Such cases are controlled through synchronization mechanisms employed inside transitions.

Scenes

A scene defines a conversation protocol for a group of roles. Each scene requires the definition of its participating roles and their population, its conversation protocol, and the states at which the agents can either leave or join the conversation. The scenes of an institution define the valid interactions that the agents are allowed to have and set the context wherein the exchanged illocutions amongst the agents must be interpreted [67].

Norms

Institutions impose restrictions on the agents' actions within scenes. These actions are basically restricted to: illocutions and scene movements. Norms determine the commitments that agents acquire while acting within an institution. These commitments restrict future activities of the agent. They may limit the possible scenes to which agents can go, and the illocutions that can henceforth be uttered.

2.5.3 EIDE Framework

In order to support the specification, design and deployment of Electronic Institutions, *EIDE* (Electronic Institutions Development Environment) is used [7]. As outlined in Figure 2.6 the *EIDE* comprises *ISLANDER*, *SIMDEI*, *aBUILDER* and *AMELI* tools.

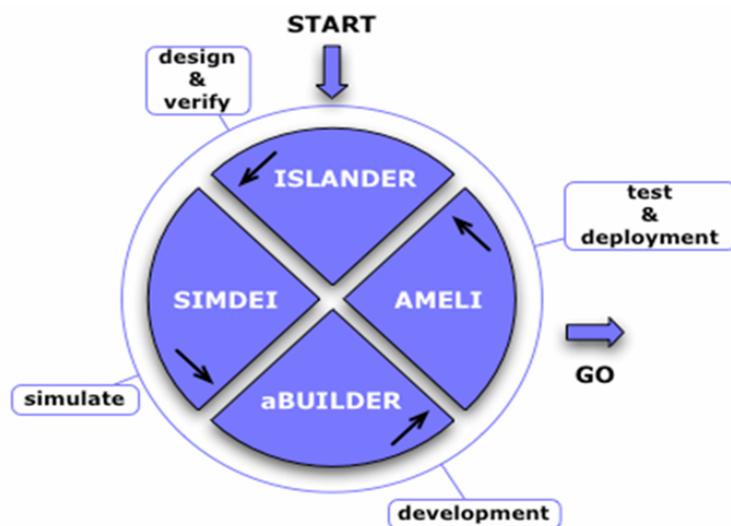


Figure 2.6: Electronic Institutions Development Environment

ISLANDER [68] is a graphical specification tool. It provides a convenient means to design Electronic Institutions through a graphical interface. It supports the specification of an institution along the four dimensions that we described above: Dialogical Framework, Performative Structure, Scenes and Norms. The tool also performs several verifications on the specified institution (integrity, protocol correctness, and norm correctness).

SIMDEI provides a simulation environment for quick testing of *ISLANDER* specifications. It is a graphical tool that can animate institutional specifications and helps in analyzing them. With *SIMDEI* institutions can be tested with different agents populations and under different circumstances [7].

aBUILDER is used to facilitate programming of the agents that participate in an Electronic Institution. While the *ISLANDER* specification of an institution provides the instructions about what can be done by the agents within the institution, it doesn't specify how the agents should do it. Therefore, programming of the agents can be conducted using any possible framework, platform or language. The *aBuilder* is a cross platform java framework specifically concerned with the development of the agents that participate in Electronic Institutions. It structures the development of agents along the specificational dimensions and automatically generates agent "skeletons" [7].

AMELI [69] loads institution specifications (.xml file created with *ISLANDER*) and acts as the infrastructure that mediates participant's interactions while enforcing the institutional norms. To execute an Electronic Institution, *AMELI* is launched up-front and then external participants can join the institution through a socket connection to this infrastructure. Each participant that is connected to the infrastructure communicates in the Electronic Institution via a *Governor*. The *Governor* serves the purpose of "safeguarding" the institutions: it checks whether a particular message is inline with the institutional protocol or not, and only then passes it to the institutional infrastructure for more in-depth verification. Governors are also capable of answering the requests of the participants about their current state and possibilities to change it. A more detailed overview of the EIDE components can be found in [7].

The specification of an institution with *ISLANDER* is important to understand for further comprehension of the concept of Virtual Institutions introduced in the following chapter. To gain such an understanding next we illustrate the process of specifying an institution in *ISLANDER* on an example. The Trading Institution used in the example is simple enough to avoid distracting the reader with unnecessary complications and, at the same time, complex enough to illustrate the key aspects of *ISLANDER*.

2.5.4 Example: Trading Institution

We will explain the Electronic Institutions metaphor by using an example of the Trading Institution. This institution comprises three main activities: admission of participants, social interactions of participants and conducting different kinds of auctions. Below we present the process of specifying this institution using ISLANDER editor. The output of the editor is in an .xml file. Appendix B gives an impression about what is stored in this file. It outlines the completed ISLANDER specification of the Trading Institution.

We start the explanation of the ISLANDER specification process by describing the components of the Dialogical Framework.

Dialogical Framework

The specification starts by defining the roles of the accepted participants. Each role defines a pattern of behavior within the institution. The participants of an Electronic Institution can play multiple roles at the same time and can change their roles. There are two types of roles: internal roles – played by the staff agents to which the institution delegates its services and tasks and external roles – played by external agents.

Figure 2.7 outlines the relationships amongst the roles in the Trading Institution.

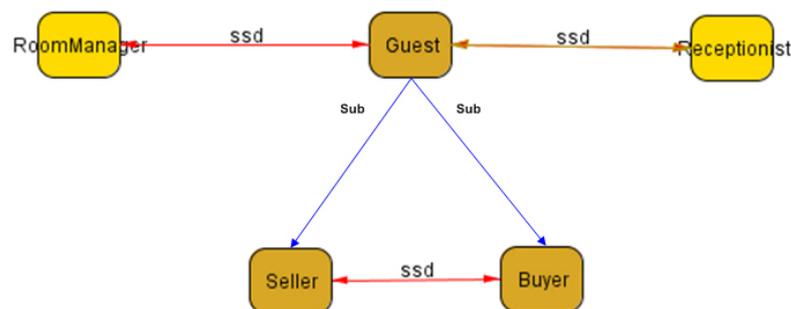


Figure 2.7: Roles in the Trading Institution

“Receptionist” and “RoomManager” are internal roles in this institution, while the rest are external roles. The Receptionist is an institutional employee, whose task is to greet the external participants and verify their registration details. The RoomManager is another employee which is responsible for starting the interactions in the Meeting Room and controlling the execution of each auction conducted in the Trading Room.

All of the external participants are playing the role “Guest”. This role has two sub-roles: “Buyer” and “Seller”. This fact is expressed in Figure 2.7 by two blue arrows marked as “Sub”. Having subroles for a given role means that in the Trading Institution

each Guest can become a Buyer or Seller as a result of some action (i.e entering correct registration details) and these roles will be accepted everywhere where the “Guest” role is accepted.

The last component on the picture that requires explanation are the two-directional arrows marked with “ssd” (static separation of duties). These arrows define incompatibility between roles. In our case, once a user entered the Trading Institution as a Guest it is not possible for this user to become a Receptionist or RoomManager. And vice-versa, a Receptionist or RoomManager can not change their roles to become a Guest or any of its subroles. The same condition is true for Buyer and Seller. A Guest who has registered as a Buyer can not be a Seller anymore.

Once the roles of the participants are defined in order for them to be able to interact with each other we need to define common ontology, acceptable illocutionary particles, valid communication language expressions and content language.

The ontology defined within the Dialogical Framework specifies the acceptable data types and functions (structuring the message content) that can be used inside the illocutions uttered by participants. There are standard data types: Integer, Float, String, Agent, etc. which can be instantly used. Other data types, which are compositions of the basic data types, can also be defined. The ontology used for the Trading Institution is shown in Figure 2.8.

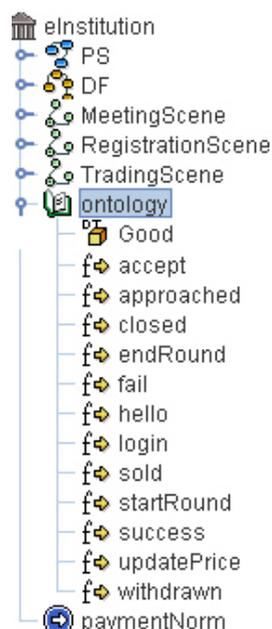


Figure 2.8: Ontology used for the Trading Institution.

This ontology specifies a single custom data type “Good”, which is a tuple of the

following form: ($goodID : float, goodName : String$). The rest of the ontology components are functions to be used in the illocutions inside scene protocols. These functions in our example are marked with an “f” symbol and include: accept, approached, closed, etc.

The illocutionary particles for this institution include: “inform” and “request”.

The communication language used by the agents consists of the expressions of the following form:

$$(i(\alpha_i, r_i)\beta, \gamma, \tau) \quad (2.1)$$

In this expression:

i is an illocutionary particle (e.g. request, inform);

α_i can be either an agent variable or an agent identifier;

r_i can be either a role variable or a role identifier;

γ is an expression in the content language;

τ can be either a time variable or a time-stamp;

β represents the addressee(s) of the message and can be:

- (α_k, r_k) the message that is addressed to a single agent.
- r_k the message that is addressed to all the agents playing the role r_k .
- “all” the message that is addressed to all the agents in the scene.

An example of an expression pattern sent by a Guest agent to a Receptionist in the Registration Room looks like:

$$(request(?x Guest)(!y Receptionist)(login ?user ?pwd)) \quad (2.2)$$

This particular expression means that agent “x” playing the role Guest sends a login request to agent “y” with user identifier and email address as the parameters of this request.

In this expression $?x$ stands for a free occurrence of the variable and $!x$ is its an application occurrence, which means that it has to be replaced by the last bound value of variable “x”.

Figure 2.9 shows the components of this regular expression.

The dialog window on the left hand side illustrates the possible illocutionary particles that are defined in the Trading Institution. These include “request” and “inform”. For an agent to be able to construct an expression as in the equation 2.2 the “request” illocutionary particle should be defined in the Dialogical Framework of the institution.

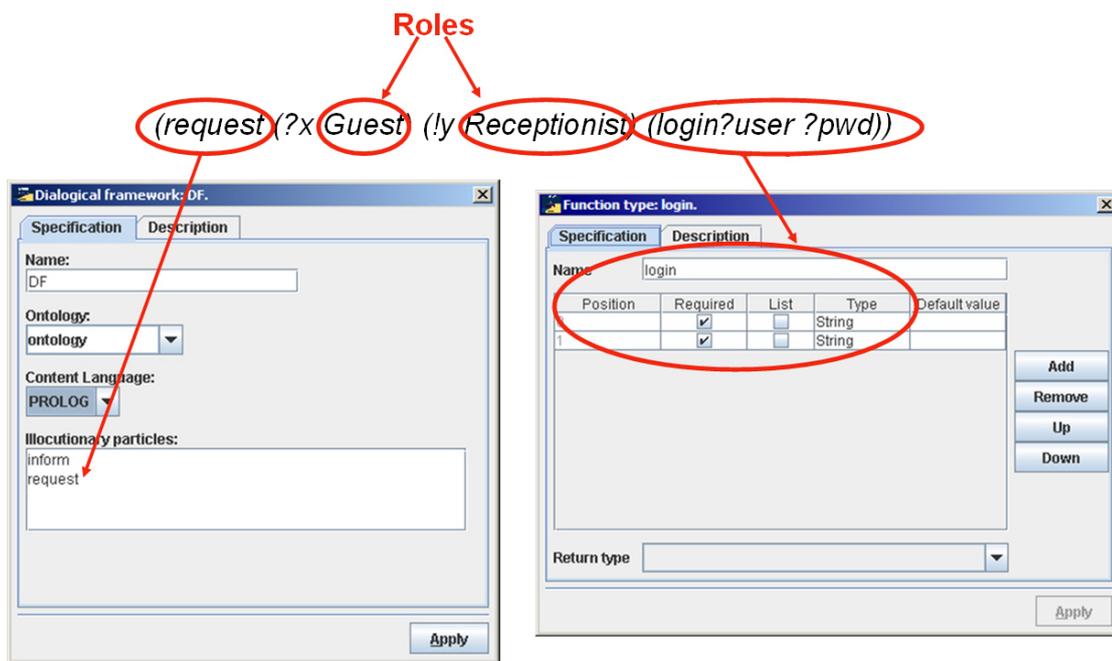


Figure 2.9: Illocutions, Content Language and Communication Language

The dialog window in the right hand side of the Figure 2.9 illustrates the definition of the “login” message inside the ontology. This definition simply specifies the number of parameters (two) and their type (both are of type String).

Performative Structure

The Performative Structure defines the relationships between basic activities (scenes) of the institution. These relationships can be:

- causal dependency (e.g. a Guest agent must go through the registration scene before going to any further scenes);
- synchronization points (e.g. synchronize a Buyer and a Seller before starting a negotiation scene);
- parallelization mechanisms (e.g. a Buyer agent can go to multiple auction scenes);
- choice points (e.g. a Buyer leaving the registration scene can choose which auction scene to join);
- the role flow policy (which participants can access which scenes depending on their roles).

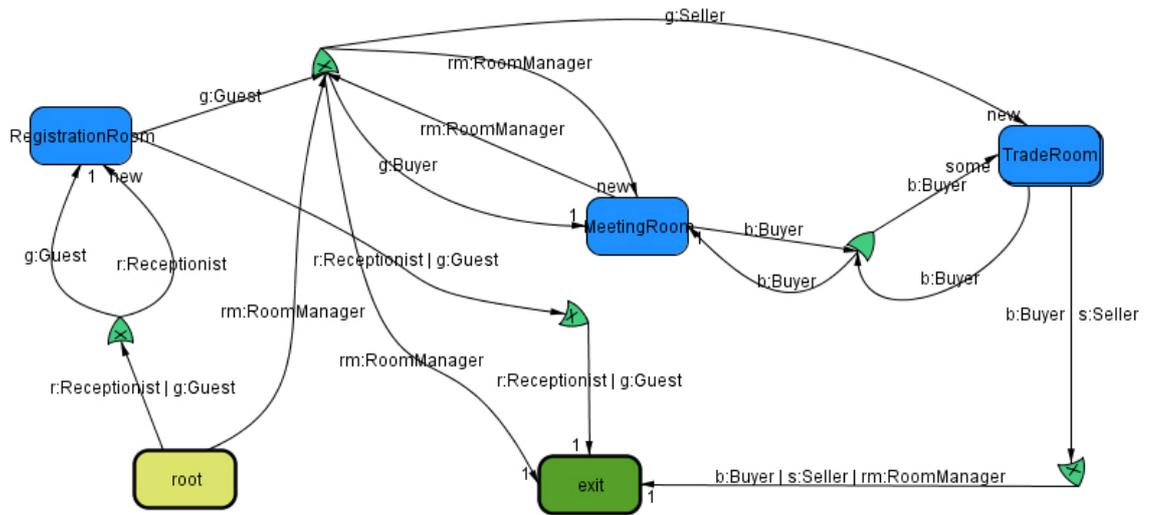


Figure 2.10: A Performative Structure of the Trading Institution

Figure 2.10 outlines the Performative Structure of the Trading Institution.

The rectangular shapes in Figure 2.10 represent scenes, the arcs connecting them are connections and triangular shapes are transitions. Three of the scenes “Registration”, “Meeting” and “Trade” in this performative structure are functional components. Another two scenes: “root” and “exit” are only used to show the entrance point (root scene) and exit point (exit scene). When an agent joins the institution it is immediately moved to the root scene. Reaching the exit scene means leaving the institution.

The transitions are used for rerouting agents between scenes and for agent synchronization. Most of the transitions in Figure 2.10 are marked with an “x” symbol (exclusive choice point), which means that the agents can only follow one path from such a transition. One of the transitions doesn’t have this symbol (choice point). This is because with this transition it is permitted for an agent to follow multiple paths. In the particular case of the Trading Institution the choice point transition connects MeetingRoom and TradeRoom scenes. A Buyer agent is allowed to perform the stay-and-go operation in the TradeRoom (which will be further shown in the protocol of this scene). This means that the Buyer agent can split into alteroids¹⁷ inside this transition. There is also another type of transitions – the synchronization and parallelization point available in Electronic Institutions (this type wasn’t used in our example). Such transitions can be used to force splitting agents into alteroids or force synchronization for agents with different roles.

The TradeRoom scene’s appearance differs slightly from the rest of the scenes and appears to be slightly “bumpy”. This illustrates that for this scene it is allowed to have

¹⁷The concept of an alteroid is similar to the concept of a thread in object-oriented programming. Splitting into alteroids for an agent means that two instances of the given agent are created inside a transition, which can simultaneously move into two different scenes and act there independently.

multiple executions. To have multiple executions means that a number of instances of a scene with the same protocol is created, each of them is associated with a state and the agents can join or leave these instances depending on the state and permissions.

The labels indicated above the connections define the role flow of participants. In the case of the Trading Institution the following role flow dynamics are possible:

- Agents with the “Guest” role: can access the RegistrationRoom scene from the root through a corresponding transition. After successful admission in the RegistrationRoom in the next transition Guests can change their role and become either Buyer or Seller. If the admission wasn’t successful – the Guests are not allowed to enter any further scenes and can only leave the institution.

If admitted as Buyers Guests change their role and can enter the MeetingRoom. From the MeetingRoom Guests can move to the TradeRoom and then either leave the institution or return back to the MeetingRoom. The “some” marking present on the arc connecting the TradeRoom with the corresponding transition shows that a Buyer can enter more than one instance of the TradeRoom scene.

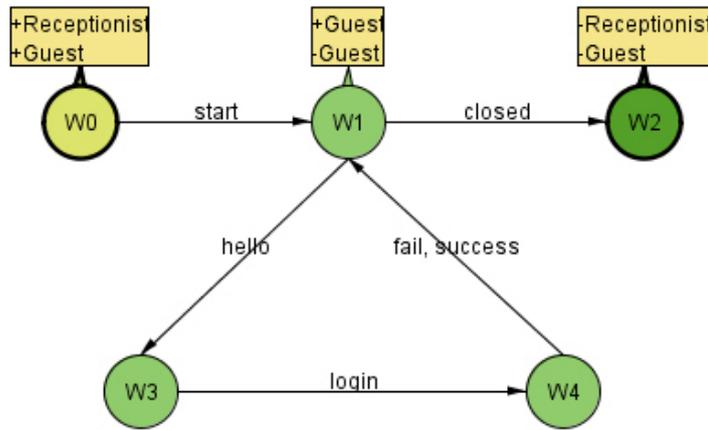
If a Guest agent is admitted as a Seller, it can create a new instance of the Trade room and conduct an auction there. The Sellers are not allowed to enter the MeetingRoom and from the RegistrationRoom can only go to the TradeRoom or exit.

- Agents with the “Receptionist” role activate the RegistrationRoom scene. This is expressed by the “new” marking on the incoming connection into the RegistrationRoom. Before the scene is activated other participants cannot enter it. As the task of the Receptionist is to control the registration of the guests, from the RegistrationRoom it can only exit the institution and can not access any other scene.
- Agents with the “RoomManager” role: activate the MeetingRoom. They can only access this scene. When the MeetingRoom scene is terminated the RoomManager agent can exit the institution through the corresponding transition.

Scene Protocols

The *RegistrationRoom scene* is used for letting the Guest agents identify themselves. It accepts only one “Receptionist” agent and twenty “Guest” agents to be simultaneously present within this scene. The protocol for the RegistrationRoom scene is presented in Figure 2.11. This figure also shows the illocutions that may trigger the state changes in the scene.

This protocol outlines 5 different states through which the RegistrationRoom scene can evolve (W0 – W4). State W0 is the initial state. As soon as the scene is activated



```

start: (timeout[0])
hello: (request(?x Receptionist)(?y Guest) hello)
login: (inform (!y Guest) (!x Receptionist) (?login ?pwd))
success: (inform (!x Receptionist) (!y Guest) success)
failure: (inform (!x Receptionist) (!y Guest) (fail?reason))
closed: (inform (?x Receptionist) (all Guest) closed)

```

Figure 2.11: Scene Protocol for RegistrationRoom

it is switched into W0. In this state the agents with roles: “Receptionist” and “Guest” can enter the scene. This is expressed through “+Receptionist” and “+Guest” markings above the state.

As a result of a timeout (“start” illocution) the scene will immediately change its state to W1. While the scene is in this state all the agents with the role “Guest” can enter or leave this scene.

The final state of the scene (the state in which scene is destroyed and can not be accessed anymore) is “W2”. This state can be reached from state “W1” through “closed” illocution. This illocution occurs when a Receptionist sends the “closed” message to all Guests. In the final state all the agents are forced to leave. The fact that all of them are indeed able to do so is expressed by “-Receptionist” and “-Guest” markings above the state W2.

The registration process happens between states W1, W3 and W4. Each of the Guest agents, that enter the scene while it is in W1 state will receive the identification request from the Receptionist (“hello” illocution). As the result of this illocution the scene will evolve to state W3. The Guest should respond with its login and password by sending the “login” illocution. This will change the state of the scene to W4.

If the registration details of the Guest are acceptable – the Receptionist will inform the Guest about it by executing the “success” illocution. If the registration details sent by the Guest are incorrect – the Guest will be notified about the failure to register (“fail” illocution) and can try to register once again. Either of these two illocutions will change the state of the scene to W1.

Notice that while RegistrationScene is in states W3 and W4 it is impossible for any agent to leave or enter the scene.

The *MeetingRoom scene* is to be used for social interactions between Buyers. The protocol and illocutions for this scene are presented in Figure 2.12.

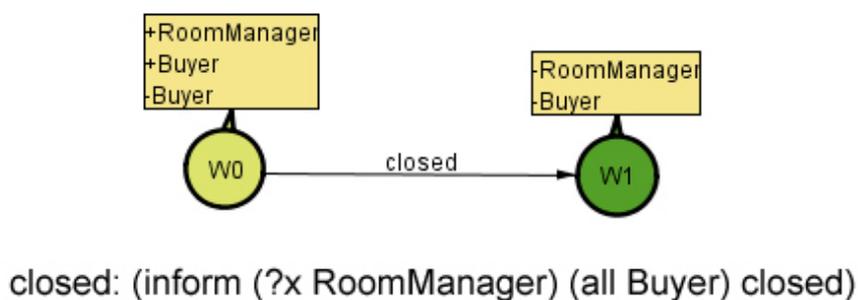


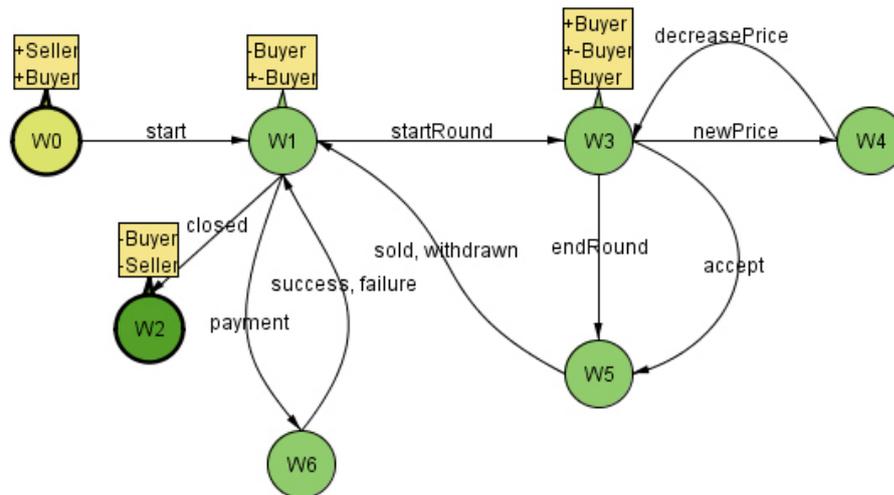
Figure 2.12: Scene Protocol for MeetingRoom

It consists of two states (W0 and W1). In state W0 agents with roles “RoomManager” and “Buyer” can enter the scene and agents with the role “Buyer” can leave the scene. It is allowed to have 1 receptionist agent and 50 Buyers inside the scene. The scene will evolve to the state “W1” when a RoomManager agent will decide to destroy the scene by sending the “closed” illocution. In this case all the agents will be forced to leave the scene and the scene will be terminated. Such simple protocol is due to the fact that in the particular implementation we are not concerned with controlling the socially acceptable behavior of the agents and no restrictions are imposed on how they interact in the MeetingRoom scene.

The *TradeRoom scene* is used for conducting different kinds of auctions following the downwards bidding protocol. The protocol for the TradeRoom scene and the corresponding illocutions are presented in Figure 2.13.

This scene has 6 possible states. In the initial state (W0) agents with the role “Buyer” and agents with the role “Seller” can enter the scene. From the initial state the scene can evolve to the state W1. This happens as the result of a Seller notifying all the Buyers that the auction is about to start by executing the “start” illocution. While the scene is in this state Buyers can enter and leave the scene.

The final state of this scene may be reached from state W1 through the “closed”



start: (inform (?x Seller) (all Buyer) start)
 closed: (inform (!x Seller) (all Buyer) closed)
 startRound: (inform (!x Seller) (all Buyer) (startRound ?good ?price ?bidding_time))
 endRound: (inform (!x Seller) (all Buyer) (endRound))
 sold: (inform (!x Seller) (all Buyer) (sold !good !y !price))
 withdrawn: (inform (!x Seller) (all Buyer) (withdrawn !good))
 newPrice: timeout[!bidding_time/100]
 decreasePrice: (inform (!x Seller) (all Buyer) (updatePrice ?price))
 accept: (request (?y Buyer) (!x Seller) accept)
 payment:(request (?y Buyer) (!x Seller) (payment ?price))
 success: (inform (!x Seller) (!y Buyer) success)
 failure: (inform (!x Seller) (!y Buyer) (failure ?reason))

Figure 2.13: Scene Protocol for TradeRoom

illocution. This illocution is executed in case a Seller has no more products to sell and decides to close the auction and destroy the corresponding scene.

When a Seller decides to submit a good to be sold at the auction, the state of the scene is changed from W1 to W3 by sending the “startRound” illocution. Here the good being sold, initial price for the good and the time for the auction are announced. In the downward bidding auction the price is constantly decreasing from the initial price to the lowest price a Seller is ready to accept. This process is expressed through “newPrice” and “decreasePrice” illocutions. The bidding time is divided into 100 segments. Each 1/100 of the bidding time the timeout (“newPrice” illocution) is fired and the scene evolves to W4. The price is then updated by the Seller and the scene reverts to W3.

The buyers are to react to the price changes and as soon as they see the price they are ready to pay – they should notify the Seller that they are ready to accept the current price. If a buyer agent accepts the price, it executes the “accept” illocution and terminates the

auction. As the result, the scene evolves to the state W5. It will also evolve into W5 if none of the buyers have decided to accept the good within the given time frame or if the seller decides to finish the round for some other reason (“endRound” illocution).

At the end of the auction the seller either announces the winning Buyer (“sold” illocution) or withdraws a good from the auction (“withdraw” illocution). As a result of this, the scene reverts to the state W1, where Buyers can enter and leave and a new auction round for another good can be initiated by the seller.

Notice that in states W1 and W3 all participants playing the role Buyer can exercise the stay-and-go operation (this is expressed in the scene protocol as “+-Buyer” marking above the corresponding states). These markings mean that the scene is in an acceptable state for agents to split into alteroids. If an agent decides to execute the stay-and-go operation then one of the alteroids remains in the original scene and all other alteroids can leave the scene and move somewhere else.

Norms

Norms define the consequences of agents’ actions within the institution. Such consequences are captured as obligations. “ $Obl(x, \phi, s)$ ” means that agent x is obliged to do ϕ in scene s .

Norms are a special types of rules specified by three elements:

1. Antecedent: the actions and boolean expressions over illocution scheme variables that provoke the activation of the norm.
2. Defeasible antecedent: the actions that agents must carry out in order to fulfil the obligations.
3. Consequent: the set of obligations expressed as pairs of scene and illocution schema.

In the trading institution there is only one Norm present (paymentNorm). This norm is presented in Figure 2.14.

Payment norm expresses the obligation of a buyer to pay for a good that was purchased in the TradeRoom. The antecedent in this norm specifies that the norm should be activated in the TradeRoom scene as soon as a seller announces that a particular good was sold to a given buyer. Once the antecedent was fired – the buyer will collect the obligation to commit the action expressed in the defeasible antecedent. The defeasible antecedent outlines the actions that should be done to withdraw the acquired obligation. In our case the obligation can only be withdrawn if in the TradeRoom the buyer agent

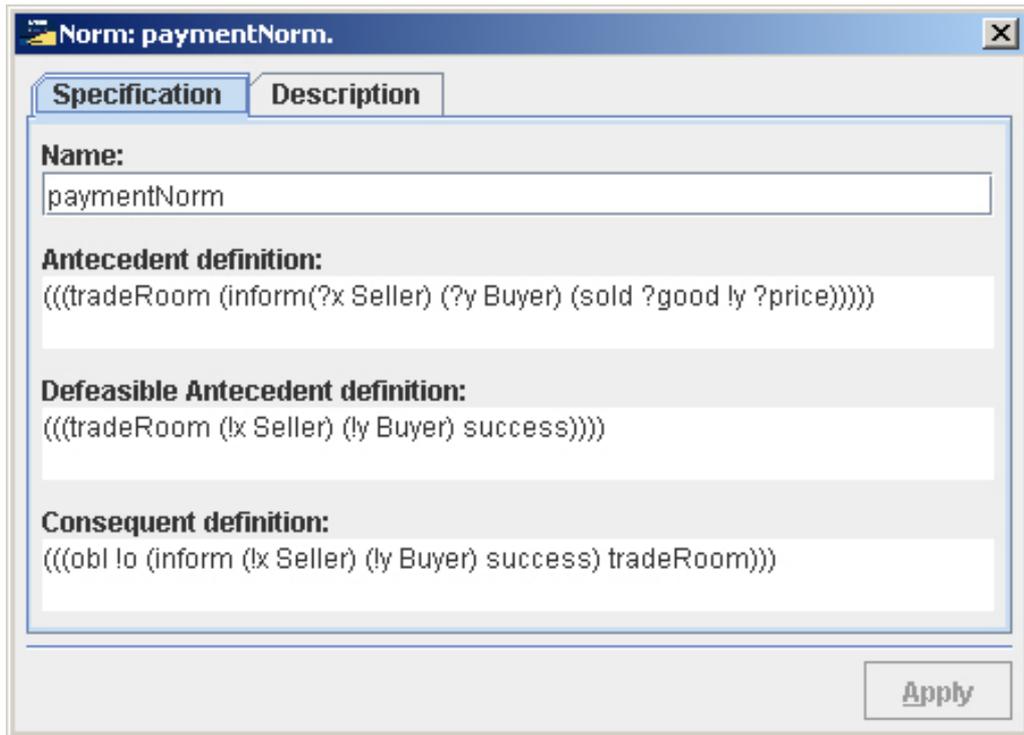


Figure 2.14: Norm Example

receives a confirmation about successful payment. The consequent shows the actual obligation the agent acquires when the antecedent is fired. In our case this obligation is not something the buyer agent has to do, but something its actions will trigger.

Norms provide an additional mechanism to control the validity of agent interactions. One of the intentions behind introducing this formalization element into Electronic Institutions was to facilitate control over the obligations an agent may acquire while moving through different scenes. For example, it is possible to have an institution similar to the Trading Institution, where payment would be completed not in the TradeRoom but in another scene. Apart from norms in this case there is no mechanism to actually control that the payment was made, while with the use of norms it is possible to establish such inter-scene validation.

2.5.5 Humans and Agents in Electronic Institutions

Electronic Institutions originate in the area of Distributed Artificial Intelligence. As a subset of Artificial Intelligence, DAI deals with issues related to autonomous agents and human-like intelligence, and not particularly with having humans as active participants of the simulated environments. The aspects of human inclusion into MAS have not received proper attention from DAI researchers and are understudied.

In general, Electronic Institutions do not prohibit human participation, as the restrictions on the internal architecture of participating agents are not imposed. The founders of the EI methodology even mention that humans are considered as active participants [172]. However, Electronic Institutions Methodology, is not supplied with the necessary technological facilities for active human involvement. As in most MAS methodologies and applications in Electronic Institutions the role of the humans is limited to interacting behind the scenes, setting up agents' parameters and interrupting agents' execution. In order for the humans to be able to participate they are required to interact with the system and other participants through instructing the corresponding agents to execute desired illocutions on their behalf. Giving such instructions can currently be only done through typing in the text of the illocution in the corresponding form. Such way of interaction with the system is inconsistent with the immersive nature of 3D Virtual Worlds.

Moreover, to be able to interact with the system in this way the human is required to have clear understanding of the Electronic Institutions concept, be able to monitor the state of the institution and know how to decode the institutional responses. Such an understanding requires a well rounded Software Engineering background and specific training. Clearly, such a situation is not acceptable as we want the created environments to be accessible for general audiences.

An alternative to direct instructing agents on every illocution is to give them a high degree of autonomy. The agents could be provided with a set of high level instructions and requested to decide themselves how these can be executed. Currently such a scenario seems rather unrealistic. Several MAS industrial applications (i.e. MASFIT project [53]) have pointed out that human beings participating in complex decision making tasks quite reluctantly delegate these activities to a completely autonomous entity. This is mostly due to the fact that there no reliable techniques yet available capable of achieving an acceptable level of agent intelligence for performing the requested task in a way the principal would desire.

Another drawback of Electronic Institutions' MAS origins is that MAS research deals with either agent-to-agent or agent-to-human interactions, and is not concerned with the support of human-to-human interactions. Software systems that enable such a feature should carefully examine the additional needs of humans and materialize the distinctive characteristics of human societies. One of the key features that is very important for humans and is not present in the agent societies is the demand for social interactions, which is completely ignored by DAI researchers.

We insist that a better understanding and modeling of the relationship between humans and the agents that make decisions on their behalf is needed to be able to integrate humans into Electronic Institutions.

To deal with aforementioned problems we suggest the following approach. Firstly, for maintaining the conceptual consistency *humans* should be seen as *heterogeneous, self interested agents, whose internal architecture is unknown*. This type of agent is acceptable in Normative Multiagent Systems (which Electronic Institutions are concerned with) and the existing theories are fully applicable to such agents. Technological dimension of human inclusion into Electronic Institutions, however, requires further investigation.

To achieve technological consistency and efficient collaboration between humans and autonomous agents we combine Electronic Institutions and 3D Virtual Worlds into the metaphor of Virtual Institutions. In the next chapter we will introduce this metaphor and show how such combination can be achieved. The new metaphor, on the one hand, helps to “open” Electronic Institutions to humans by supplying them with the necessary facilities to be directly integrated into Electronic Institutions. On the other hand, it helps to bring structured interactions and social order into Virtual Worlds.

The new metaphor also enables extending the intelligence of the autonomous agents through observing their principles, so that humans can eventually be able to instruct their agents to execute some task and be certain that the set of actions the agent select goes inline with humans’ requirements. Chapter 5 reveals some details about this issue.

2.6 Summary

This chapter has introduced the concepts of Virtual Worlds, Electronic Institutions as well as all related terms and technologies. In the present form Virtual Worlds are unregulated environments with no widely acceptable methodology that would be capable of structuring the interactions of participants and guarantee that those interactions do not create any possibilities of system abuse. Such a guarantee can only be achieved by the employment of formal methodologies.

Electronic Institutions is a formal methodology, which originates in the area of DAI, and is concerned with structuring the interactions of autonomous agents. The novelty proposed in this thesis is the application of this methodology to the development of Virtual Worlds. To realize this we need to see Virtual Worlds as Multiagent Systems and treat humans as autonomous agents with unknown internal architecture.

One of the problems with Electronic Institutions is that direct human inclusion is not technologically supported. The demand for human inclusion is strongly expressed by Virtual Worlds, which are mostly oriented towards support of human-to-human interactions with very little or no agent involvement. In the next chapter we present Virtual Institutions Methodology that is originally based on Electronic Institutions, but was further extended to include specific requirements expressed by Virtual Worlds.

Chapter 3

Virtual Institutions

Every day in the real world we participate in a number of institutions. Once we enter a work place, shop or university we realize the change of the context and start obeying the rules of the environment we have entered. Our behavior is highly influenced by these rules, which range from not strictly enforceable and rather implicit social conventions (like etiquette) to more explicit and usually strictly controlled rules or instructional norms (like walking through a metal detector in an airport or queuing before paying for the products in a shop).

Virtual Worlds represent a replication of the real world with a difference that most of them allow some degree of anonymity, have no physical adherence to the culture of a particular region and provide richer facilities to observe and control the behavior of participants. To avoid anarchy, that usually comes together with anonymity, Virtual Worlds require clear ways to define and enforce the rules of the interactions. Moreover, in virtual societies as established in Virtual Worlds it is highly desirable to be able to easily explain the institutional rules to the carriers of different cultural backgrounds and representatives of different language groups. The most efficient way to achieve these goals is to have the institutional rules expressed in a formal way (using mathematical equations) and create natural language translations for each of the desired languages.

The concept of Virtual Institutions proposed in this chapter helps in bringing structuring of interactions into virtual spaces by formalizing the interaction rules. Although in the current form it is only concerned with establishing explicit rules it can also be applied for establishing implicit social conventions. The proposed approach is not only useful for helping humans to establish social order in Virtual Worlds but also for helping autonomous agents to reduce the uncertainty about the world, understand and learn the rules of the interactions and interpret the actions of the others. Further we present the concept in more details, illustrate it with an example and outline the implementation requirements.

3.1 The Concept

The concept of Virtual Institutions is defined as follows.

Definition: *Virtual Institutions are 3D Virtual Worlds with normative regulation of interactions.*

More precisely, we propose to separate the development of Virtual Worlds based on the concept of Virtual Institutions into two independent phases: specification of the interaction rules and design of the 3D Interaction environment. For producing more efficient designs such separation is widely used in architecture [133], whose metaphor inspires Virtual Worlds. Apart from design efficiency, in our case this separation has the following advantages:

- it helps in achieving the clear distribution of the development tasks between system analysts and designers;
- explicitly focusing the attention of system analysts on the interactions forces them to inspect the system in details before creating the visualization, which is useful for detecting the critical points and errors at an early stage;
- it makes the specification of the interaction rules independent of particular Virtual Worlds technology used for the visualization of the system, permitting a quick and easy portability to new visualization platforms.

To be able to support the conceptual separation between the design of a Virtual World and normative control of the interactions within this space we subdivide the Virtual Institutions concept into two conceptual layers: Visual Interaction Layer and Normative Control Layer.

The *Visual Interaction Layer* maps to the domain of 3D Virtual Worlds. It is concerned with audio and visual aspects of the multimedia, as well as with visualization of the interactions of participants in the 3D Virtual World.

The *Normative Control Layer* maps to the domain of Electronic Institutions and is concerned with the institutional control of interactions that happen in the Visual Interaction Layer.

For the purpose of normative control of the interactions we suggest employing the Electronic Institutions methodology, which is supplied with facilities for rules specification and tools for helping to ensure the validity of the specified rules and their correct execution. However, we would like to point out that in Virtual Institutions these rules are used in a slightly different manner. In contrast to the majority of normative Multiagent

Systems (and Electronic Institutions in particular) the normative part of a Virtual Institution does not represent all the activities that are allowed to be performed in a Virtual World. On the contrary, the normative part can be seen as defining what is prohibited to do. Its absence means that any action is granted immediate execution.

Not every Virtual World requires normative control of interactions as well as not every real world institution needs 3D Visualization. Only systems that have a high degree of interactions, and only if these interactions need to be structured in order to avoid violations) may need institutional modeling. And only the institutions where 3D visualization of active components is possible and beneficial should be visualized in Virtual Worlds.

Systems that could benefit from both interaction control and 3D visualization, provided by the concept of Virtual Institutions, should be built following the Virtual Institutions metaphor presented in the next section.

3.2 The Metaphor

The concept of Virtual Institutions requires the development of a new metaphorical basis. Choosing an appropriate metaphor for a new concept is very important and is a critical factor in users' overall acceptance of this concept [181]. Metaphors provide useful abstractions for technological concepts and help in explaining the users something unfamiliar (and usually complicated) in terms of something they are well familiar with [129].

The original metaphor behind the concept of Electronic Institutions is a "theater". As in a theater, Electronic Institutions have a set of scenes, and only the "players" with specific roles can appear in particular scenes. The conversations inside each of the scenes follow a strict protocol [172]. Although, metaphorically they originate from this limited domain, Electronic Institutions have much higher expressive power and are applicable to much wider range of problems than the original theater metaphor suggests. Moreover, combining Electronic Institutions with 3D Virtual Worlds makes the theater metaphor rather inappropriate.

Virtual Worlds do not only suggest a metaphorical explanation of the concept, but also provide a concept visualization in terms of the proposed metaphor. Visualizing activities like E-Commerce in the theater metaphor could cause more confusion to the users than bring actual benefits. Therefore, we see a strong need to use another more general metaphor for Virtual Institutions.

While choosing a new metaphor for software it is important to select amongst those, which meaning is known to very general audiences and doesn't change in different cultures. Building on the metaphors users deal with very often, and especially on the

metaphors reflecting physical structures, increases the chance of the metaphor being assimilated by the users [129].

Virtual Worlds are inspired by the metaphor of architecture. They often employ physical structures like buildings, rooms, walls, etc. to represent different kinds of activities and to separate them from one another [181]. The metaphor of architecture is universal as humans are mostly familiar with the concept of a building. It is also culturally independent. Therefore, the metaphor of architecture seems like a good choice for Virtual Institutions and can be applied to the same (or an even wider) range of problems that we are concerned with in this thesis. However, the fact that we are dealing not with arbitrary Virtual Worlds, but with Virtual Worlds that support normative regulation of interactions, requires us to use a constrained version of the metaphor used by the Virtual Worlds.

We propose seeing Virtual Institutions as a virtual space called *3D Interaction Space*. This space can correspond to an arbitrary 3D Virtual World populated by avatars and various objects. Inside the 3D Interaction Space a set of buildings are located, where each of the buildings represents an Electronic Institution. The appearance of the 3D Interaction Space outside the buildings can be arbitrary and the behavior of avatars is not controlled by the institutional norms. However, there are restrictions on appearance and interactions inside the buildings.

As it is hard to provide a physical metaphor for the 3D Interaction space we find it necessary to introduce some sort of substitute. Thus, we present the metaphor of a garden. A garden should be seen as a place surrounding the institutional buildings. This metaphor is well known to humans as gardens often surround residential buildings in the real world. In this way the 3D Interaction Space can be described as the combination of the garden and institutional buildings.

Employing the building metaphor for the visualization of an institution is motivated by the fact that many institutions familiar to the participants from the real world (like universities, courts, banks etc.) also have a brick and mortar representation. As in the real world the walls of a virtual building create visible boundaries for norm enforcement.

Each institutional building is associated with its unique set of interaction rules, which are controlled by the specification of the corresponding Electronic Institution. The participants are visualized as avatars and each of them is assigned with at least one role. Only participants with specific roles can enter the institutional buildings and once there should act according to the specification of the corresponding institution. The concept of a role is widely used in Virtual Worlds. In many game based Virtual Worlds a role reflects the fact of being a part of a selected group and determines different abilities of the participants associated with it. In non-game based Virtual Worlds the role is normally used to distinguish between fee paying subscribers and participants with trial membership.

Further elaborating the metaphor, we see each of the institutional buildings being divided into a set of rooms that are separated from each other by walls and doors. The doors are opened or closed for a participant depending on the role and the institutional state. Again, this choice of metaphors is fully consistent with real world institutions. Walls and doors are often employed in the real world to restrict access to some activities.

Figure 3.1 outlines the details of the Virtual Institutions metaphor presented so far.

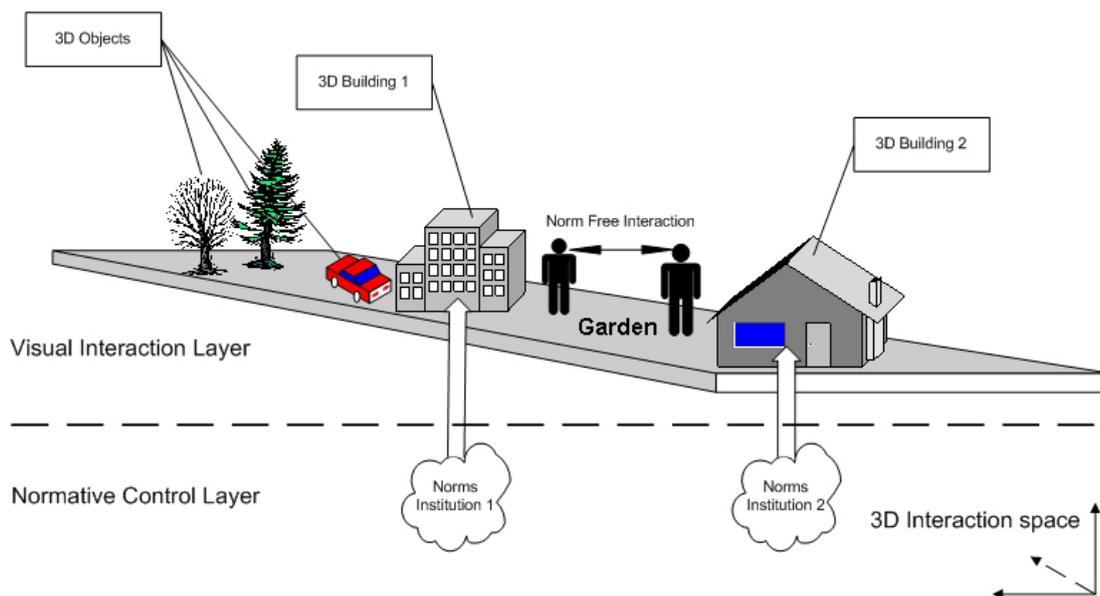


Figure 3.1: Virtual Institutions Metaphor

With the help of the 3D Virtual Worlds technology the metaphor of Virtual Institutions can be visualized. All of the concepts contributing to the metaphor (Garden, Buildings, Rooms, etc.) will have a graphical representation inside the resulting 3D Virtual World.

The findings in the area of Virtual Worlds design research suggest that during the construction of a 3D representation of a Virtual World, it is important to keep the benefits of traditional 2D interface design in mind [27]. Participating in a 3D environment, where users can manipulate 3D objects, does not necessarily mean exclusion of 2D interface elements. In fact, the interaction with 2D interface elements offers a number of advantages over a 3D representation for particular tasks [146]. Most efficient selection techniques, for instance, are widely realized in 2D, whereas, the selection process in a 3D user interface must consider the *user's viewpoint and distance* to the object. Combining the advantages of 2D and 3D design is a very powerful and intuitive approach for the construction of virtual environments [146].

Based on these findings we suggest enhancing the Virtual Institutions metaphor with 2 additional components (which in the resulting Virtual World will be visualized as ad-

ditional 2-dimensional elements). The first component is a map of the 3D Interaction space, which inside a building is transformed into a building map. The map metaphor is widely used in all kinds of Virtual Worlds to improve the navigation of participants. Virtual Worlds can be harder to navigate than the real worlds, and employment of maps proved to be very useful in Virtual Worlds [55]. As Virtual Institutions are visualized as Virtual Worlds, we find it necessary to introduce, maps into Virtual Institutions as well.

Another additional component that contributes to the overall metaphor and which is also visualized in a 2-dimensional way is what we call “backpack with obligations”. Electronic Institutions provide technological facilities for participants to collect obligations while acting inside the institution. Not fulfilling the obligation might lead to access restriction to some activities by the participant. To inform the participants about the reasons for imposed restrictions and to provide them with easy facilities to visualize their commitments (so that enforcement of restrictions can be promptly avoided) we introduce the backpack metaphor. This metaphor is taken from military-oriented games, in which it is usually the case, that a player is given a mission, which in turn consists of a set of submissions. Any time during the game, the player can click on his/her backpack to see the mission related details. In a similar way, the backpack can be used in Virtual Institutions to show the acquired obligations to the participants.

Next we describe the components of each of the Virtual Institutions conceptual layers in terms of the new metaphor.

3.3 Visual Interaction Layer

Visualization of the Virtual World and its participants as well as providing the participants with interaction facilities is associated with the following set of concepts.

3.3.1 3D Interaction Space

The 3D Interaction Space is the entrance point into the visualized 3D Virtual World. It is represented as a graphical 3-dimensional area and associated with an Euclidean 3-dimensional coordinate system, where $(0, 0, 0)$ is the reference point for locating the objects. Most of the laws of physics usually apply within this space, however, their effect may be slightly different to the corresponding effect in the real world and some of the rules may be absent or purposely allowed to be violated.

All the interactions of the participants are limited to interacting within this space. There is no possibility to move beyond it and the only way to leave is by disconnecting from the Virtual World. Once someone enters it, he/she will become embodied as an avatar and will be physically located inside.

To enhance the believability of the visualization the space is usually populated with a number of various 3D Objects. The most typical case is that a 3D Interaction Space is decorated with grass, trees, bushes, cars, etc.

The special type of objects located inside the 3D Interaction Space are buildings. Each of the buildings is metaphorically seen as an institution. It is important for every building to collide with the participants trying to enter it and there should be no way to switch the collision off. Most of the rendering engines utilized by 3D Virtual Worlds usually support forcing the collision, and for those which do not it is required to introduce the necessary changes to force the collision before using them for Virtual Institutions.

3.3.2 Garden

Anywhere outside the institutional buildings the interactions of participating avatars are not regulated and every event that happens inside this space is immediately visualized without any prior validation. This part of the 3D Interaction Space is called the *Garden*.

The Garden is the entrance point into the Virtual World. Exiting any institutional building will also result in the avatar being placed inside the Garden.

3.3.3 Institutional Buildings

Institutions are excluded from the uncontrolled interactions. Each institution is seen as a separate normative structure, and for its visualization we use the “building” metaphor. The 3D model of a building for every institution is present in the system located within the 3D Interaction Space and for regulating the interactions each building is associated with an Electronic Institution specification. The Electronic Institution is seen as an infrastructure that establishes a set of norms on the behavior of participants, who can be either humans or autonomous agents. The institutional buildings do not necessarily follow an Euclidean model. The rooms inside the building may not be involved in any kind of spatial relationship and the total size of the building as perceived from the outside, may be significantly different to the total size of the internal space.

The enforcement of the rules is achieved via strict institutional control of the actions of participants inside the institutional buildings. Every event that a participant requests by pressing keys on the keyboard or by operating the mouse is first sent to the institutional infrastructure for validation. If the institution permits the event execution then the corresponding action is visualized, otherwise the event is ignored. It is also possible for the institution to provide context-based explanations for the reasons why a particular event cannot be processed. This can be done by the institution requesting an action to be visualized for a desired participant. This action may be just a sound which is played

when a participant is trying to open a door but has no permission to enter it, or it may be a more sophisticated action, such as an employee of the institution appearing in front of the participant and explaining the reasons for the imposed restrictions.

Each of the institutional buildings has a single entrance door, through which the participants can enter it. The entrance to each institution is restricted and only participants with specific roles will be granted entrance to the institutional building. Whether or not a particular avatar is allowed to enter a given institution is determined by matching the role initially assigned to the corresponding participant against the set of possible roles accepted by the Electronic Institution, responsible for regulating the interactions inside the given building. The initial roles are assigned before entry to the 3D Interaction Space and can be further changed through participants' interactions inside the institutional buildings. The avatars with roles that are not accepted in a given institution will experience the locked entrance door. For the avatars with a role accepted by the institution the entrance door will be unlocked or open.

One of the most typical cases is that after entering the 3D Interaction space all of the participants are initially given the "Guest" role and most of the institutional buildings accept the participants with this role, letting them enter the registration room. In this room, the participants may decide to explore the building further, but to do this, they are required to enter their registration details (login/password) and only then, will be assigned a new role to be able to enter other rooms.

3.3.4 Avatars

The participants of the 3D Interaction Space are visualized as avatars¹. We distinguish between the following two types of avatars: avatars for visitors and avatars for the institutional employees. The visitors are all agents or human playing external roles. Internal agents, humans playing internal roles and governors are institutional employees.

Visitors' avatars are provided with an initial set of default appearances, but these can be changed later. The institutional employees are assumed to have a similar appearance which is in line with the dress code of the institution that they are employed with. The governors are usually not displayed, but they may appear as avatars when a participant tries to violate the institutional rules. Governors may also be visualized in case a participant experiences navigation problems and requires help (the trajectory of the avatar's movement may clearly indicate whether the participant is lost in the Virtual World).

According to [46], following an embodied agent may dramatically improve the navigation. For this reason if the participant appears to experience navigational problems (to

¹An electronic representation of one's self in a form of a graphical character.

determine this, the trajectory of each avatar is constantly observed) the embodied governor may appear embodied as an avatar to guide the participant to a desired destination. The default appearance of the governor is a police officer dressed in uniform.

We assume that some of the participants of Virtual Institutions are autonomous software agents (not humans). Such agents are visualized as avatars and may replace humans playing any institutional role and try to believably act on their behalf. The fact that they have similar embodiment as human participants and, therefore, may be mistakenly believed to be humans raises an important question: whether to explicitly notify the humans that they are talking to an autonomous agent or hide this fact? On the one hand, hiding this fact has a number of practical benefits connected with improving the acceptance and the development of trust in the actions of such avatars [189]. On the other hand, such approach poses a number of ethical questions. Can software agents influence human relations [144]? Can humans develop a false sense of trust towards agents and can the agents abuse this trust [19]? Can humans develop personal relationships with agents mistakenly accepting them as humans and what are the consequences of this [57]?

We do not have a preferred position in the debate regarding the need to notify humans that they communicate with an agent and believe that the questions raised above require a detailed further investigation. Therefore, we propose that the system architect should be solely responsible for making the final decision on this topic. As an overall guideline, however, we propose to pay attention to the ethical issues connected with the presence of agent-driven avatars and introduce the following mechanism for indication that the avatar is controlled by an autonomous agent. Each time an autonomous agent gains control over an avatar, this avatar is forced to change its appearance to the standard “bot” appearance, which will become an indication for other participants that this avatar is not controlled by a human anymore. The only exception to this is the appearance of the institutional employees, which should not be changed. The institutional employees are institutional representatives and the institution takes full responsibility for their actions, so they are not likely to express deviant behavior and will not provide misleading information to other participants. Some of the institutional employees may act, for example, as sales assistants and behind each of the assistants humans and autonomous agents may constantly interchange each other following the implicit training approach presented in Chapter 5. Changing the appearance of the sales assistant avatar on a regular basis may become highly frustrating for the participants interacting with such avatars and we, therefore, suggest to keep their appearance unchanged.

Another important issue concerning avatars is that in some of the rooms it is allowed by the institution to split the participant into several alteroids² (avatars). The concept

²We borrow this metaphor from computer games, where a player is often in control of several avatars

of alteroids is important as in the virtual space we are not physically restricted to being just one actor. This fact received a lot of attention in computer games and other kinds of software. In order to support this feature in Virtual Institutions, each time a new alteroid is created a human participant should decide which avatar to choose to control further and the remaining avatars will be controlled by autonomous agents. This functionality allows a human to employ autonomous agents for performing some routine tasks on a human's behalf, while the human may be involved in other (more complex) activities.

Finally, it is necessary to impose restrictions on the avatar movement inside the 3D Interaction Space. These restrictions are associated with the role played by the corresponding avatar. For being assigned with a role before entering the 3D Interaction Space the participants are first required to type in their identification details. This normally includes entering the unique nickname and a password that the system can map to an acceptable role. After the identification process, the corresponding avatar appears in a specified location within the 3D Interaction Space, outside any of the institutional buildings. While outside, the avatars are free to execute any possible actions and their communication is not moderated by any of the institutions. Entering the institutional buildings imposes all the limitations associated with the chosen role.

3.3.5 Rooms

Every institutional building consists of a set of rooms. The rooms are supposed to be represented as rectangular boxes closed by walls from every side. Each room has at least one door, through which it can be entered. The collision inside each of the rooms is forced and cannot be switched off. Once a participant enters a room, the only possibility for the corresponding avatar to move outside the room boundaries is to walk through one of the doors embedded into the walls of the room. For a participant to be able to instantly move from one room to another, it is necessary for those two rooms to be connected via a door. This connection is not required to be spatial. A door may act as a teleport, instantly moving an avatar to the initial position in the room that is being entered through the door.

Unlike in most of the typical Virtual Worlds, in Virtual Institutions it is not always the case that a participant can enter the room through a door and then will be able to exit through the same door. On the contrary, when the room is entered, the entrance door may become automatically locked for the user. This depends on the specification of the underlying Electronic Institution, whether it allows the backward movement between the rooms or not. In order to provide a consistent experience for the participants we suggest that the system designers should include the backward movement where possible.

simultaneously.

3.3.6 Doors

The Doors are used to connect different rooms in the institutional building. Each door is associated with a number of execution states (a state per avatar). Each state logically explains whether a door is open or closed for a particular avatar. Due to this fact, at the same point of time, the same door may be open for one participant, but closed for another one. In order to avoid breaking the immersion of participants by the fact that some of them will be observing the others walking through closed doors, we use the following approach. Every time an avatar changes the state of a door by opening it – every other avatar which is currently observing the door will also see that the door is open but will not necessarily be able to enter it. Technically this is achieved by making the door invisible when it is open and only switching off the collision for the avatars that are allowed to proceed through it.

3.3.7 Map

In order to simplify the navigation of the participants every institution is supplied with a map of the 3D Interaction Space. Once a participant enters an institutional building, this map is replaced by the map of the entered building. The map usually appears in the upper-right corner of the screen as a semitransparent schematic plan. Each of the available rooms in the building is displayed on the map and the human-like figures show every participant the positions of all the associated alteroids. While moving throughout the institution the positions are updated accordingly.

The human individual can use the map to find the desired alteroid and may click on the corresponding figure to take control over it. Once an alteroid is selected the human fully controls it and the alteroid that the human has previously controlled is assigned to an autonomous agent attached to the selected alteroid.

3.3.8 Backpack with obligations

While acting in an institution a participant may acquire some commitments. An example of such a commitment may be that a participant who has won an auction round will not be able to directly leave the institution, but is committed to visit the payment room before leaving. Commitments are expressed in the specification of the underlying Electronic Institution and their fulfillment is controlled by the system. In order to have a simple way to present those commitments to a human we use the metaphor of a backpack utilized in many computer games. The backpack icon is displayed in the lower right part of the screen and a participant may decide to hide it or reveal it. Clicking on the backpack will result in displaying the list of commitments acquired by the participant.

3.3.9 Events/Actions/Messages

Although, we anticipate that participants may use all sorts of different devices for navigating virtual worlds, in a standard case a participant in a 3D Interaction Space is able to control the avatar and change the state of the Virtual World by pressing keyboard buttons, moving a mouse or clicking mouse buttons. These physical actions executed by a human in the real world generate events inside the Virtual World, which are then visualized as actions executed within the 3D Interaction Space. The events that a participant is trying to execute inside an institutional building are not directly visualized. Before visualization every event is transformed into a message understandable by the institution and sent to the institutional infrastructure for validation. Only if the message is consistent with the current state of the institution and it is not against the institutional rules to visualize the corresponding action – the action is performed.

3.4 Normative Control Layer

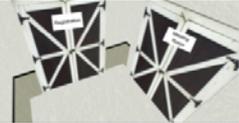
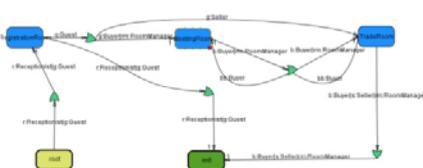
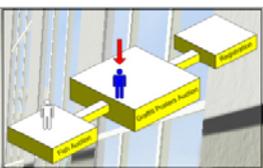
The description of the Visual Interaction Layer provided the explanation of the Virtual Institutions metaphor in terms of the Virtual Worlds domain. In order to explain how the institutional control of the processes inside the Virtual World is achieved we next present the mapping of the concepts presented above to the domain of Electronic Institutions. This mapping serves 2 conceptual purposes:

- explaining the Electronic Institutions metaphor in terms of Virtual Worlds using the concepts familiar to most of the humans. We see this part as being useful for providing richer explanation of the Electronic Institutions to people without Multiagent Systems background.
- explaining the Virtual Worlds in terms of institutional specification of the underlying processes. This part is useful for the designers who will be able to see the direct connection between the designs and the activities that these designs are thought to support.

Further in this section we provide an explanation how each of the concepts described in the previous section is expressed in terms of Electronic Institutions.

Table 3.1 presents a short overview of the mapping between the concepts of Electronic Institutions and the corresponding concepts of the Virtual World.

Table 3.1: Mapping between 3D Virtual Worlds and Electronic Institutions

Specification Element	Example	Virtual World Element	Example
Scene		Room	
Transition		Room/No representation	
Connection		Door	
Number of participants		Size of the room	
Agent		Avatar	
Message	<code>(inform (? RoomManager) (all Buyer) closed)</code>	Action	
Root Scene		Room/No representation	
Exit Scene		Garden	
Obligations	<code>((obl !x (inform (!x buyer) (?y buyer_ac)(payment !price)) buyer-settlement))</code>	Backpack	
Performative Structure		Map	
Data Types in Ontology		3D Objects or No Representation	

3.4.1 Federations

The part of the 3D Interaction Space called Garden, in its current form, has no conceptual relationship to the Electronic Institutions metaphor, although the developers of the Electronic Institutions are working on introducing the notion of *federations* as the unity of institutions that serve a similar purpose and, therefore, the 3D Interactions Space can be seen as the Federation. A good real world example for a federation would be a university. Most of the universities usually have a campus (federation), which contains a number of faculties (institutions) united under the common purpose of education and under the same umbrella institution. In the case of the university the institutional rules are very much the same across different faculties and the rules and policies of the university are mentioned much more frequently than the rules of the particular faculty.

3.4.2 Institutions

In Virtual Institutions every building is seen as an institution. This metaphor is borrowed from the real world, where most of the institutions are brick and mortar. Entering or exiting such an institution changes the behavior of participants and their conversation style. Most of the processes inside each brick and mortar institution are controlled by the institutional infrastructure.

3.4.3 Scenes and Transitions

A scene in Electronic Institutions is seen as the protocol that describes a basic activity. By giving a brick and mortar representation to this concept we suggest seeing each of such basic activities as a separate physical room inside the institutional building. As in a theater there is usually a different set of decorations for different scenes – the same way we see it being represented in Virtual Worlds. The only difference is that at the current stage each of the rooms in Virtual Institutions is physically separated by a set of walls from all other rooms.

Transitions in Electronic Institutions serve the purpose of a middle point between two different rooms and sometimes are used for synchronization of agents. In a case when a transition is used for synchronization purposes we suggest to visualize it as a room of a special kind. Probably the most appropriate appearance for such a room would be a waiting room similar to the waiting rooms in the airports. That's the place where participants will have to wait for someone else to join them if the institution demands so.

In the most general case, when a transition is used as a simple middle point between two rooms we propose not to visualize it at all, to avoid forcing the participants to move through the spaces that do not correspond to any behavioral process.

In the Electronic Institutions specification for each of the scenes the number of participants for each of the accepted roles is always specified. We use this information to determine the size of the room. The more avatars can enter the scene the bigger it should be. So, the size of the room is proportional to the sum of the maximum number of participants in all the accepted roles.

Electronic Institutions support having multiple executions of a scene. This may, for example, be useful for the case when the maximum number of participants in a scene has been reached and a new participant wants to enter the scene. A classic solution to this problem is to create another instance of the scene and place the new participant inside of this new scene instance. For dealing with the issue of multiple scene executions in Virtual Worlds we use the metaphor of a floor. Every time it is required to create another instance of an already existing scene – another instance of the room that corresponds to this scene is created, placed on top of the previous instance and the building is extended to have an additional floor. So, the height of the buildings that contain scenes with multiple execution will be dynamically changing. There are cases when a participant is allowed to choose which scene execution to enter. In order to support entering a different execution of a scene in the 3D Virtual World, transitions function as elevators, which allow for choosing the room to be entered in terms of pressing the corresponding button.

3.4.4 Performative Structure

The original idea behind the Performative Structure is to define the main activities and specify the role flow of participants between those activities (scenes). The closest concept present in the Virtual Worlds domain is the map of the building. Although it is not usually the case that the map contains information regarding who is allowed to enter which room, this information becomes unnecessary after personalizing the interface. As it naturally happens in the Virtual Worlds, the map for each of the participants is personalized. The participants are usually only interested in the places that they can access and have much less interest in the rest. This allows us to personalize the map in a way that only the scenes that a participant is able to access are visualized there.

Every performative structure contains special types of scenes called “root” and “exit”. These scenes are not associated with any processes inside the institution and cannot have a protocol specified for them. The root scene only serves the purpose of being the entrance point into the institution. In some cases when a root scene is connected to more than one transition it is useful to visualize it as a small room with a set of doors. Otherwise, when there is only one transition connected to it we suggest to avoid creating unnecessary rooms and do not visualize it. The entrance door of the institution in this

case will lead straight into the room connected to the root scene via a first transition.

The exit scene defines the exit point. It is never visualized. Reaching the exit scene means leaving the institutional building. So, once a participant walks through a door that corresponds to the connection leading to the exit scene, the corresponding avatar will be located outside the institutional building and the exit door will be closed (as there can be no return connection from the exit scene).

3.4.5 Connections

The connections in the Performative Structure graph are used to determine the flow of participants between a scene and a transition. In the Virtual World we see them visualized as doors connecting different rooms. Many connections should not be visualized at all. For example, when in the specification there are several incoming connections that define entering a scene by the agents playing different roles, there is no need to create a separate door inside the corresponding rooms for each of them. In such a case only one door will be present and all the agents will use the same door for entering the room.

In the case when a transition is not visualized all the connections leading to it should not be visualized too. In such situation the rooms that are connected with each other via this transition will have a direct door connection.

3.4.6 Obligations

As it was already mentioned before, the obligations the participants acquire in an institution are represented as the backpack, which on opening displays the textual description of the obligations.

3.4.7 Data Types in Ontology

To be able for the agents operating in an Electronic Institution to “understand” each other it is necessary for them to establish common language. Therefore, the ontology defines a set of data types that the agents should operate with (by sending messages to each other). The majority of the Virtual Worlds inhabitants are humans, who usually are pretty good in establishing a common language without any ontological help. But it doesn’t mean that the ontology is absolutely useless for visualization. The data types in the ontology are used to describe objects that the participants operate while interacting inside the institution. Although some data types may refer to intangible concepts, the majority of them should be represented as 3-dimensional objects.

Associating data types from ontology with 3D models in the Virtual World helps

autonomous agent to recognize these objects, manipulate them, track actions upon them and refer to them in conversations with humans.

3.4.8 Illocutions and Messages

The notion of illocutions originates from the speech act theory [185]. It is suggested that every time a human speaks there is always some intended action behind each uttered illocution. For example, in an extreme case such as the a president of the United States declaring the war in Iraq – the result is not just a set of words spoken out by someone, but a set of actions that change the state of the world around us. That’s why in Electronic Institutions every message sent between two agents contains an illocution. Consider the example of a classroom with an open door. Here two possible illocutions wrapped around the same message can be understood in a very different way:

- **Illocution 1:** Request(Student, closed(door))
- **Illocution 2:** Inform(Teacher, closed(door))

Illocution 1 has the clear expectation on the side of the agent (a student) who receives it to perform the action of closing the door, while Illocution is probably not associated with any action. Notice that it is possible to have a simple dialogue using those two illocutions, where a teacher executes Illocution 1 and the student replies with Illocution 2. Being concerned with visualization of the illocutions we have a set of actions for every message, where the illocution used in the dialogue defines which action of this set to visualize.

3.4.9 Synchronization Issues

The Electronic Institution specification permits the modeling of situations where several agents have to enter a scene together. To do that, agents must synchronize. While synchronization is easily supported by the Electronic Institution infrastructure, it requires detailed thinking when applied to Virtual Worlds.

We want to provide an immersive environment where humans are “driving” their avatars throughout different rooms. Moving to another room in a Virtual World is expressed by opening a door, walking through the transition and opening the entrance door of the next room. When more than one participant is required to leave the room (open the exit door) we propose the following approach: each participant approaches the door and tries to open it (showing the intention to leave). The last agent to synchronize approaches the door and is able to open it. When the door is opened all the waiting agents will be moved through it *automatically* (synchronized).

3.5 Concept Illustration: Trading Institution

In order to illustrate the concept of Virtual Institutions we propose the reader to consider the following scenario.

Scenario: *Imagine a businessman who is very interested in contemporary art. He is a regular customer of a Virtual Institution and uses its fish market auction for buying and selling fish. One of the rooms in this institution serves as the gallery for graffiti posters. During the vernissage the artist is present in the room and is looking forward to conversations with visitors. The businessman enters the poster exhibition and spends his time browsing through the art works, while his another alteroid, driven by an autonomous agent, participates in the fish market auction and buys fish on his behalf.*

In Virtual Institutions the above scenario can be mapped onto the norms of the Trading Institution presented in section 2.5.4. The specification of the Trading Institution forms a basis for the Normative Control Layer. The Visual Interaction Layer of this institution corresponds to the 3D Virtual World outlined in Figure 3.2.



Figure 3.2: The Virtual World of the Trading Institution

The Virtual World contains a landscaped garden, within which the institutional building is located. The building consists of five rooms, three of which correspond to Regis-

tration, Meeting and Trading scenes in the Electronic Institution specification. The other two rooms represent the transitions connecting these scenes in the specification. The rooms are separated by doors, where doors are either locked or open for a participant depending on its role and the state of the institution.

The Virtual World for this institution is automatically generated from the Electronic Institution specification. In the Visual Interaction Layer the Electronic Institution specification determines the skeleton of the Virtual World. In the Normative Control Layer the specification serves as the basis for the execution of the infrastructure, enforcing the interactional constraints, controlling the state of conversations, and providing permissions for different roles.

The performative structure forms the basis for the map of the institution and the 3D model of the institutional building. Each of the scenes (“Registration Room”, “Meeting Room” and “Trade Room” in this case) lay the foundations for the generation of the 3D models of the rooms. The size of each room is determined by the maximal number of participants specified for each scene. The *root* and *exit* scenes do not have any visual representation. Transitions are transformed into special types of rooms (corridors) connecting the scenes. Connections (lines connecting the elements of the performative structure) are represented as doors. Each door is initially locked, and will be opened as soon as the participant is granted the permission to enter the corresponding scene or transition by the institution.

After the automatic generation the rooms in the newly created Virtual Institution are not furnished, however, the doors, transitions and door labels are present. This institution is fully functional, which means that all the security issues of the institution will be imposed (e.g. permissions, protocols, obligations). The agents are able to freely interact and take part in conversations; the consistency of those conversations and interactions with the institutional rules is guaranteed by the infrastructure and the possibility to split into alteroids is also granted. To make the institutional building visually appealing the textures and additional objects are added at a later stage.

Each participant enters the system appearing as an avatar in the garden. The initial role given to each of the participants is “Guest”. As defined in the specification any “Guest” can enter the institution. Therefore, the entrance door of the institutional building is always open for all avatars. Entering the institutional building for an avatar means entering the Registration Room. The “root” scene and the transition connecting it to the “Registration” scene are not visualized. After successful registration in the Registration Room the participant’s role changes and, depending on this new role, appropriate doors are open. Opening of the doors happens independently for each particular avatar, and the same doors may be locked for a different avatar if the permission is not granted.

Once inside the institutional building a participant moves throughout its different rooms. The Registration Room serves as a reception desk. Once the desk is approached a participant is welcomed into the institution by the Receptionist agent and asked to enter login and password details to be able to proceed further. Figure 3.3 demonstrates the Registration Room.

From the Registration Room every participant can either leave the institutional building by walking back through the entrance door into the garden or enter the corridor (transition) behind the registration room, which connects it to the Meeting Room. The door that connects the corridor with the Meeting Room is initially locked. Only when the user is correctly identified as a “Buyer” this door will be unlocked. If the user is identified as a “Seller” the door will not be open as Sellers are not permitted to enter the Meeting Room. The exit door from the corridor in this case will contain a number of buttons on it (as in an elevator). Each button is marked with the name of the auction currently conducted inside the Trading Room. Pressing a button results in teleporting the Seller into the selected instance of the Trading Room.



Figure 3.3: Registration Room Inside the Trading Institution

The view inside the Meeting Room is presented in Figure 3.4. To match the scenario presented above the room is decorated as a graffiti poster gallery. The posters are presented in a conventional way (hanging on the walls) as it is usually done in real world galleries. The participants of the gallery are represented as avatars. The avatar controlled by an autonomous agent has a robot-like appearance. All the other avatars are human participants with customizable appearances.

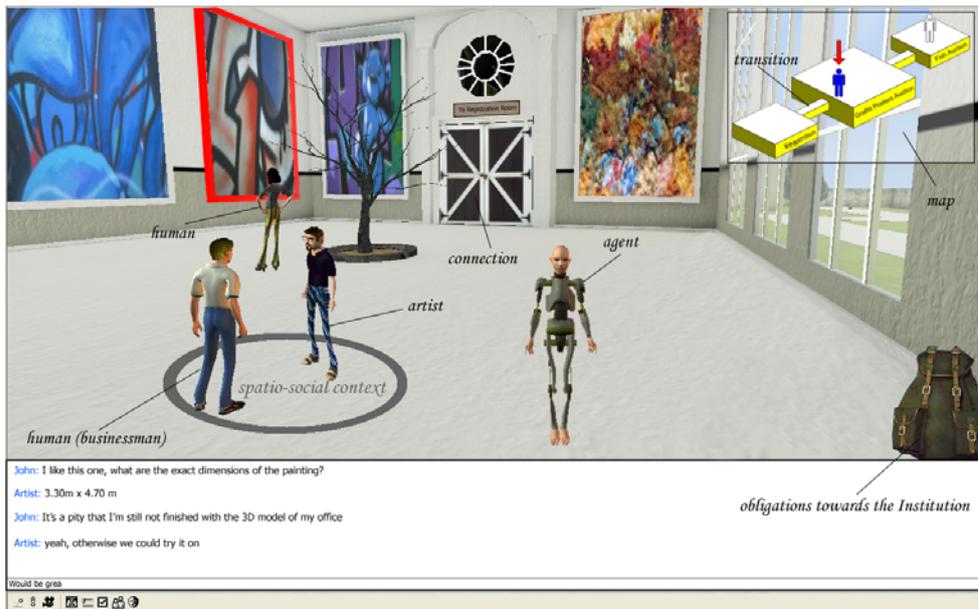


Figure 3.4: Meeting Room Inside the Trading Institution

Additionally to the main 3D part, the Virtual World is enhanced with some 2-dimensional elements. The chat window serves the purpose of the communicator between participants. To reduce the information overload and engage people into spatial interactions the maximum *audibility distance* is specified. Only the avatars within the audibility distance can participate in a conversation. The avatars that are not within this distance can not “hear” the conversation. Due to this fact, the female avatar present in the left part of the window is not disturbed with the conversation between the artist and the businessman. This approach also provides means to address privacy issues: humans can clearly observe the participants of each conversation and may not give away secure information if there are undesirable participants present.

Another important 2D part of the interface is the *map* of the institution. It is only visible if the mouse pointer is moved to the right border of the screen. The large rectangular blocks represent rooms and the smaller ones correspond to transitions. The solid figure with the arrow on top of it displays current location of the human-driven avatar within the institution. The non-solid figures represent the autonomous agents that are in the participant’s subordination. As it was already mentioned, Electronic Institutions permit situations where a participant can split himself/herself into a number of *alteroids*. Only one of the alteroids can be controlled by a human and all the others are controlled by autonomous agents. Autonomous agents act autonomously trying to fulfill the task specified by a human. They may move around and even walk between different rooms. If in some situation an autonomous agent is unable to proceed with the given task due

to the lack of intelligence, the figure representing this agent on the map starts to blink attracting the attention of the human.

The human is able to control any of the alteroids at any time by clicking the corresponding figure on the map. This will lead to switching to a different view (determined by the position and head rotation of the alteroid). The control over the avatar that the human was controlling before is automatically passed on to an autonomous agent and the appearance of this avatar is changed to the default appearance of an autonomous agent (robot look). In the given scenario the autonomous agent (non-solid figure) represents the businessman in the fish market auction, while the businessman drives the avatar (solid figure) through the Meeting Room.

The *Backpack* with obligations is another 2-dimensional element of the user interface. It helps the human to remember the obligations towards the institution that have to be fulfilled. The backpack automatically opens and the pending obligation is displayed if the situation of not fulfilling an obligation makes it impossible to proceed to another scene or state in the institution. The participant can also see the obligations on demand by clicking on the backpack icon.



Figure 3.5: Trading Room Inside the Trading Institution

As it is expressed in the institutional specification, the Trading Room is allowed to have multiple executions. Therefore, it is represented as a number of similar rooms placed on top of each other. Inside each of these rooms a different type of auction is conducted. One such auction is outlined in Figure 3.5. This room instance is currently functioning as a fish auction. A seller is conducting the auction with lobsters being the

current offer. The buyers surrounded the seller's desk and wait for an acceptable price. The red robot on the left hand side is the businessman's autonomous agent from our scenario.

Once the offer is announced, the big screen behind the seller is updated to show the picture of the current product, the remaining time left for bidding and current price. As specified in ISLANDER the auction follows the Dutch auction protocol. In this protocol the auction is held for a given period of time and the initial price for a good is set to be slightly higher than the desired price of the seller. Within the given time-frame for a given auction round the price decreases to the minimal acceptable price. As soon as the price drops to the point when one of the buyers is ready to accept it – the buyer raises his hand to notify the auctioneer that he wants to purchase the advertised product. The auctioneer immediately announces the winner and if there are still any goods left to be sold – continues with the next round. After a successful purchase the buyer who won the previous round is required to approach the seller and finalize the purchase. The obligation to pay for the good is assigned to the buyer by the institution and can be observed through the buyer's backpack. Until this obligation is fulfilled this buyer will be unable to leave the institution. The product that was not sold during the given time frame is withdrawn from the auction.

3.6 Formalizing the Concept using Z Specification Language

In the previous section we described the concept of Virtual Institutions. Here we formalize the concept using Z specification language (see [126] for details). The main task behind the formalization is to create precise requirements for implementation of Virtual Institutions and their deployment. For providing such requirements, each of the concepts for both Visual Interaction and Normative Control layers is formalized and some additional components are introduced.

Z is a formal specification language that is widely used for creating precise specification of computer systems. It utilizes standard mathematical notations used in set theory, lambda calculus, and first-order predicate logic. The main reason for selecting Z for Virtual Institutions is that Z notation used in this language is commonly preferred by many researchers and the language itself is quite successful in communicating general ideas about software architectures from researchers to developers. Formal Z specifications using mathematical notations help to precisely describe the properties which an information system must have, without unduly constraining the way in which these properties are to be achieved [195]. Z specification helps to describe what the system must

do without saying how it is to be done. This abstraction makes formal specifications useful in the process of developing a computer system, because they allow questions about what the system does to be answered confidently, without the need to disentangle the information from a mass of detailed program code, or to speculate about the meaning of phrases in an imprecisely-worded prose description [195].

As the result of applying this technique, the Z-specification of Virtual Institutions is quite general and independent of a particular technology. Despite that generality, it provides enough information for system developers to be able to gain overall understanding of the computer system and implement it.

In general, a Z specification of a system consists of the following steps [195, 126]:

- Defining the terms, constants and variables. Here constants, global variables and compositional types used throughout the specification are defined. The variables here should be seen as mathematical data types rather than variables seen in programming languages.
- Defining the basic schemas. Z schemas are small pieces of specification in mathematical form linked together with text commentaries. They help to decompose the formalization of the system into basic components. In our case, each of the schemas corresponds to a basic concept from either the Normative Control Layer or Visual Interaction Layers.
- Defining the runtime state. This dimension of the formalization defines how the schemas defined on the previous step should evolve at runtime.
- Defining the state initialization mechanisms. Here the initialization procedures for each of the schemas are described.
- Defining the operations allowed at runtime. The schemas represent the key architectural components and the operations define possible actions that can change the state of the system.

All of the above steps were followed to create the Z-specification of Virtual Institutions. Before presenting a detailed overview of these steps, we will outline the key components that were identified on each of the conceptual levels. Figure 3.6 shows the two layers and each of them includes the logical components that correspond to this layer. The Normative Control Layer contains only one specification element *EISpec* as the top level abstraction of the institutional rules. The reason for this abstraction is that the formalization of the Normative Control Layer was completed by the developers of

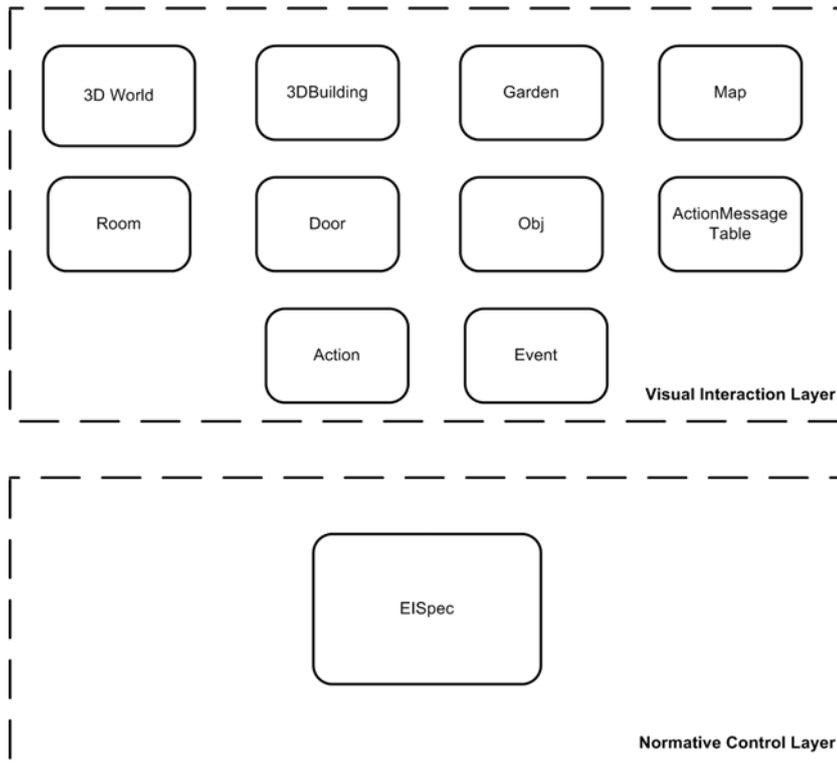


Figure 3.6: Basic Formalization Components.

Electronic Institutions and, therefore, we do not present it here. Interested readers will find the specification of the EISpec in Appendix A.

The Visual Interaction Layer consists of *3D World* – corresponding to 3D Interaction Space; *3DBuilding* – the institutional building; *Garden* – the part of the 3D Interaction space surrounding the institutional buildings; *Map* – the schematic representation of an institutional building; *Room* – a room inside an institutional building; *Door* – a door connecting two rooms; *Obj* – an object located anywhere inside the 3D Interaction Space; *Action/Message Table* – an abstraction required to achieve mapping between the actions inside an institutional building and corresponding messages sent inside the Normative Control Layer. *Action* represents any change of the 3D Interaction Space that forces the change in the Normative Control Layer, while *Event* corresponds to a request to conduct an action inside an institutional building.

The extended version of this diagram is presented in Figure 3.7. There the relationships between the presented components are outlined and the basic Z-schemas used in the formalization are introduced. Each of the schemas are represented by a rectangle separated into 3 sections.

1. the name of the schema;

2. the names of other schemas that a given schema depends on;
3. the operations defined for the schema.

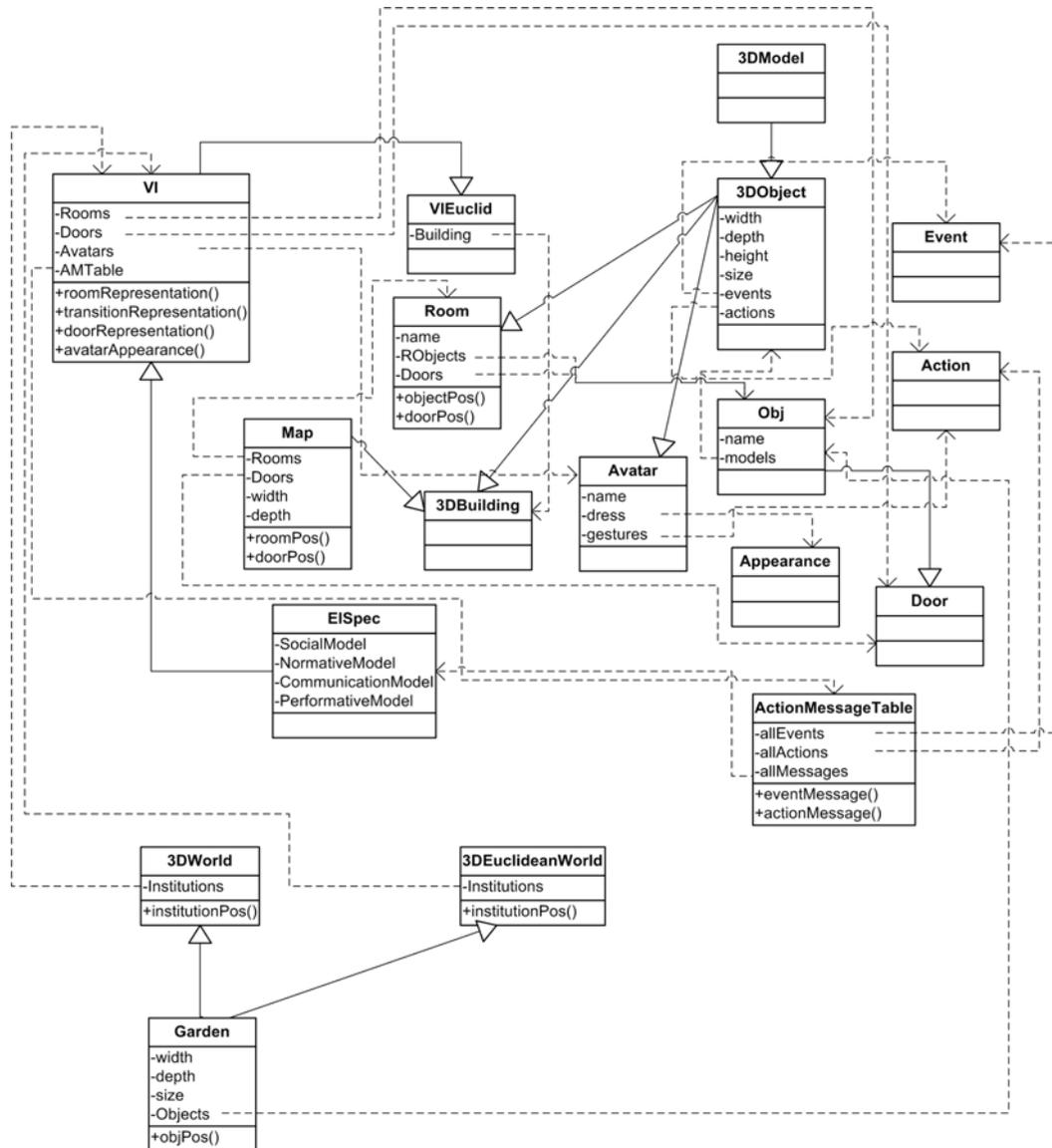


Figure 3.7: Schema Relationship Diagram for Z-Specification.

Readers familiar with object oriented programming should consider an explanation of the specification in terms of the class metaphor. The name of the schema can be seen as a name of the class. The names of the dependent schemas correspond to member variables. The operations can be treated as member functions. The connecting arrows presented in the picture represent the inheritance, and dotted lines show the use of one class by another.

Next we present the details of each of the Z specification steps.

3.6.1 Virtual Institution Terms

Variables

We use different sets of variables for different classes of elements that we wish to represent.

$[RoomVar, DoorVar, AvatarVar, ObjectVar, TimeVar, ActionVar, EventVar]$

Constants

The constants are the identifiers associated with avatars, rooms, doors, objects and natural numbers relating to time and a constant to specify the minimum avatar space in a room.

$Time == \mathbb{N}$
 $MinPSpace : \mathbb{N}$
 $DoorDepth : \mathbb{N}$
 $[RoomID, DoorID, AvatarID, ObjectID, ActionID, EventID]$

There are special constants, signifying the set of all rooms, all doors, all avatars, all objects, all actions and all events.

$\left\{ \begin{array}{l} roomall : RoomID \\ doorall : DoorID \\ avatarall : AvatarID \\ objectall : ObjectID \\ actionall : ActionID \\ eventall : EventID \end{array} \right.$

Terms

Terms can be either variables or constants (identifiers).

- *RoomTerm* is the set of room variables and room identifiers.
- *DoorTerm* is the set of door variables and door identifiers.
- *AvatarTerm* is the set of avatar variables and avatar identifiers.
- *ObjectTerm* is the set of object variables and object identifiers.
- *TimeTerm* is the set of time variables and time values.

$$\begin{aligned}
RoomTerm & ::= roomconst\langle\langle RoomId \rangle\rangle \mid roomvar\langle\langle RoomVar \rangle\rangle \\
DoorTerm & ::= doorconst\langle\langle DoorId \rangle\rangle \mid doorvar\langle\langle DoorVar \rangle\rangle \\
AvatarTerm & ::= avatarconst\langle\langle AvatarID \rangle\rangle \mid avatarvar\langle\langle AvatarVar \rangle\rangle \\
ObjectTerm & ::= objectconst\langle\langle ObjectID \rangle\rangle \mid objectvar\langle\langle ObjectVar \rangle\rangle \\
TimeTerm & ::= timeconst\langle\langle Time \rangle\rangle \mid timevar\langle\langle TimeVar \rangle\rangle
\end{aligned}$$

3.6.2 Virtual Institution Model

The Virtual Institutions formalism extends the *EISpec* schema, originally specified for the Electronic Institutions. While we present only a brief overview of the components of *EISpec* relevant to Virtual Institutions, the full specification can be found in Appendix A.

The first component of *EISpec* is *SocialModel*, where the roles of the participants and the relationships between those roles are specified. *PerformativeModel* determines which scenes and transitions can be enacted in the institution and which types of participants can access them. *CommunicationModel* introduces the communication language for the participating agents. *NormativeModel* is concerned with controlling the obligations of participants towards the institution.

<i>EISpec</i> <i>SocialModel</i> <i>NormativeModel</i> <i>CommunicationModel</i> <i>PerformativeModel</i>

Virtual Institutions enhance the formalism of Electronic Institutions with the specification of the 3D Interaction Space and its visualization.

3D Interaction Space extends the scope of the Electronic Institution Specification (*EISpec*) by being able to include a number of different institutions (which relates to the concept of “Federation” in the Electronic Institution methodology). In contrast, *EISpec* is only concerned with one institution.

Inside the 3D Interaction Space different institutions are visually represented as 3-dimensional buildings, which are all surrounded by a Garden.

There are two possible ways to visualize the content of the 3D Interaction Space: Euclidean and non-Euclidean. To express this fact we separate the concept of the 3D Interaction Space into two independent schemas. The schema that represents an Euclidean 3D Interaction Space is called *3DEuclideanWorld*. The schema that covers the specification of a non-Euclidean 3D Interaction Space is called *3DWorld*.

Both *3DEuclideanWorld* and *3DWorld* are top level concepts in the specification hierarchy. We present this hierarchy in a top down manner. First we describe the *3DWorld*

and 3DEuclideanWorld schemas and then continue expanding them by describing other components they include. For the comfort of the reader we suggest using Figure 3.7 as a guide through the presentation.

The *3DEuclideanWorld* schema represents a case when all the visualization elements of a Virtual Institution are located within an Euclidean space. Euclidean space is a space that complies with the notion of Euclidean distance and the related concepts of length and angle. Such a space is also associated with an orthogonal coordinate system in any number of dimensions. It is the “standard” example of a finite-dimensional, real, inner product space. In our case, we are dealing with three-dimensional spaces. Every institution is visualized as a building and all buildings share the same orthogonal space with unique coordinates. Any two buildings can be uniquely identified by their coordinates. Moreover, the rooms inside each of the buildings share this property too. Any room has a unique position determined by its coordinates inside the 3D Interaction Space.

The *3DEuclideanWorld* schema extends the *Garden* schema, meaning that it encapsulates all of its functionality. The *Garden* is the space surrounding the institutional buildings. The *3DEuclideanWorld* schema also includes a set of *Institutions*. Each of the items in this set is an instance of VI schema.

To associate each institution with unique coordinates inside the Euclidean space this schema is supplied with the *institutionPos* relationship. This relationship could be seen as a function, where a given institution returns its coordinates. With the help of the *institutionPos* relationship it is possible to create a global map of the 3D Virtual World. This map outlines the Garden and the institutional buildings.

The 3DEuclideanWorld schema is associated with a number of constraints (also called predicates). The constraints define the limitations that each of the schemas 3DEuclideanWorld extends should comply with. These constraints are as follows:

- The *institutionPos* function is defined on the *Institutions* set;
- The position of each institution is bigger than (0, 0, 0) – which is the starting position of the garden. Also an institution cannot be placed outside the *Garden* and cannot extend beyond it;
- None of the institutions can overlap.

Below is the formal description of the 3DEuclideanWorld schema in Z-specification language. This description consists of two parts separated by a horizontal line. The upper part lists the schemas that are extended by 3DEuclideanWorld and introduces new variables and relations.

The lower part of the schema description specifies the constraints that the introduced variables and functions have to comply with.

3DEuclideanWorld

Garden

Institutions : \mathbb{P} *VIEuclid*

institutionPos : *VIEuclid* \leftrightarrow $R \times R$

$\text{dom } \textit{institutionPos} = \textit{Institutions}$

$\forall i : \textit{Institutions} \mid \{x, z\} = \textit{institutionPos}(i) \bullet$

$x \in (0, \textit{width} - i.\textit{width}) \wedge z \in (0, \textit{depth} - i.\textit{depth})$

$\forall i_1, i_2 : \textit{Institutions} \mid$

$\{x_1, z_1\} = \textit{institutionPos}(i_1), \{x_2, z_2\} = \textit{pos}(i_2) \bullet$

$[x_1, x_1 + \textit{width}] \cap [x_2, x_2 + \textit{width}] = \emptyset \vee$

$[z_1, z_1 + \textit{depth}] \cap [z_2, z_2 + \textit{depth}] = \emptyset$

The *3DWorld* schema presents a case when a Virtual Institution doesn't have an Euclidean representation. This means that the buildings and the rooms inside them are not necessarily sharing the same Euclidean space. In fact, they may not be associated with any Euclidean space at all.

One of the important characteristics of an Euclidean space is that a distance between two points is a function of their coordinates. In a non-Euclidean space this property is not necessarily satisfied. One example of such a situation may be the use of *teleportation*, where the distance between two places connected by a *teleport* is equal to zero. A *teleport* in our case is represented as a door. Instead of opening the door when permission to enter it is granted, and walking through it when required to leave a room, a different strategy is employed. A door is always closed and colliding with the door will result (if it is not against the institutional rules) the disappearance of the participant in the source room and instant appearance in the target room.

The predicates of the *3DWorld* schema define the following set of limitations:

- The *institutionPos* function is defined on the *Institutions* set (meaning that it can only be applied to the members of this set);
- Any two institutions are uniquely positioned (we consider an (x,y,z) coordinate of the bounding box surrounding the institutional building being the position of an institution);
- As the institutional buildings are placed inside the Garden, the position of any institution is limited by the dimensions of the *Garden*.

$3DWorld$ <hr/> <i>Garden</i> <i>Institutions</i> : $\mathbb{P} VI$ <i>institutionPos</i> : $VI \leftrightarrow R \times R$
<hr/> $\text{dom } institutionPos = Institutions$ $\forall i_1, i_2 : Institutions \mid \{x_1, z_1\} = institutionPos(i_1) \mid$ $\{x_2, z_2\} = institutionPos(i_2) \bullet x_1 \neq x_2 \wedge z_1 \neq z_2$ $\forall i : Institutions \mid \{x, z\} = institutionPos(i) \bullet$ $x \in (0, width) \wedge z \in (0, depth)$

The *Garden* is a virtual space decorated with 3D objects, which surrounds the institutions. The *width* and *depth* set the dimensions of the *Garden* in the virtual space. Some objects are used for decorating purposes and may include, for example, trees, fountains, grass. Another kind of objects are institutional buildings. Institutional buildings have a deeper functional purpose and act as a graphical representation of the institutional infrastructure.

The only predicate of this schema defines *size* as *width* multiplied by *depth*.

$Garden$ <hr/> <i>width</i> : \mathbb{N} <i>depth</i> : \mathbb{N} <i>size</i> : \mathbb{N} <i>Objects</i> : $\mathbb{P} Obj$
<hr/> $size = width * depth$

Next, we specify how the concepts of the Electronic Institutions relate to the concepts of Virtual Institutions and how the latter are visualized.

For the case of a non-Euclidean representation (3DWorld) the *VI* schema is used. It inherits the *EISpec* schema, extends it with new concepts and specifies the relations between those new concepts and building blocks of the *EISpec*.

The extension of the Electronic Institutions metaphor consists of the following components:

- Rooms – the 3D representation of the Scenes in the Performative Model. Each *Scene* is associated with a *Room* via *roomRepresentation* function;
- Doors – the 3D representation of connecting arcs between scenes and transitions in the performative model. Each *Door* is associated with one or more *Arcs* connecting a *Room* with a *Transition* in the Performative Model. For the case when a transition and a scene have more than one arc connecting them, there will be only one *Door*

created;

- The size of the door (width*depth) as well as its volume is equal to zero (the door is a flat 3D object);
- Objects – additional elements of the 3D World used to enhance the immersive experience of the users;
- Avatars – 3D objects representing the participants. Each *Role* in the Social Model is associated with an *AvatarType*;
- AMTable – the table that stores the relationships between all the events, actions and messages in the institution.

In addition to the introduced data types and the relations between them VI schema also defines some constraints:

- The *sceneRepresentation* and *transitionRepresentation* functions result in the *Rooms* set;
- The *sceneRepresentation* function is defined on the *allscenes* set;
- The *transitionRepresentation* function is defined on the *alltransitions* set;
- The *doorRepresentation* function is defined on the *allarcs* set;
- The *avatarAppearance* function is defined on the *allroles* set;
- The *doorRepresentation* function results in the *Doors* set;
- A room representation of a scene is never equal to a room representing a transition;
- The size of each room should be big enough to be able to contain the maximum number of participants for the corresponding scene;
- For each connected scene and transition there is one corresponding door in the VI. The room representations of these transition and scene will share the same door object;
- Each room has a set of associated doors. This set is a subset of *Doors* initially generated from the performative model;
- The appearance of the agents that play internal roles is different from the appearance of the agents with external roles;

- All the possible events (*allEvents*) of the *AMTable* is the union of all the events of all the objects in every room;
- All the possible actions (*allEvents*) of the *AMTable* is the union of all the actions of all the objects in every room;
- All the messages of the *AMTable* are obtained via *illocutionParticles* variable from *EISpec*.

VI

EISpec

AMTable : *ActionMessageTable*

Rooms : \mathbb{P} *Room*

Doors : \mathbb{P} *Door*

Avatars : \mathbb{P} *AvatarType*

sceneRepresentation : *Scene* \leftrightarrow *Room*

transitionRepresentaion : *Transition* \leftrightarrow *Room*

doorRepresentation : *Arc* \leftrightarrow *Door*

avatarAppearance : *Role* \leftrightarrow *AvatarType*

ran *sceneRepresentation* \cup **ran** *transitionRepresentation* = *Rooms*

dom *sceneRepresentation* = *allscenes*

dom *transitionRepresentaion* = *alltransitions*

dom *doorRepresentation* = *allarcs*

dom *avatarAppearance* = *allroles*

ran *doorRepresentation* = *Doors*

$\forall s : \text{allscenes}, t : \text{alltransitions} \bullet$

$\text{sceneRepresentation}(s) \neq \text{transitionRepresentation}(t)$

$\forall \text{scene} : \text{allscenes} \bullet \text{sceneRepresentation}(\text{scene}).\text{size} >$

$\text{scene}.\text{maxNumberOfParticipants} * \text{MinPSpace}$

$\forall s : \text{allscenes}, t : \text{alltransitions} \bullet$

$\text{connectingArcs} = s.\text{outArcs} \cap t.\text{inArcs} \neq \emptyset \Rightarrow$

$\exists_1 d \in \text{Doors} : d = \text{doorRepresentation}(\text{connectingArcs}) \wedge$

$d \in \text{sceneRepresentation}(s).\text{Doors} \wedge$

$d \in \text{sceneRepresentation}(t).\text{Doors}$

$\forall r : \text{Rooms} \bullet r.\text{Doors} \subseteq \text{Doors}$

$\forall r_1 : \text{InternalRole}, r_2 : \text{ExternalRole} \bullet$

$\text{avatarAppearance}(r_1) \neq \text{avatarAppearance}(r_2)$

$\text{AMTable}.\text{allEvents} = \bigcup_{r \in \text{Rooms}} r.\text{RObjects}.\text{events}$

$\text{AMTable}.\text{allActions} = \bigcup_{r \in \text{Rooms}} r.\text{RObjects}.\text{actions}$

$\text{AMTable}.\text{allMessages} = \text{iparticles}$

The *VI Euclid* schema is used in the specification of the *3DEuclideanWorlds* schema. It corresponds to the case where each Electronic Institution has an Euclidean representation in the 3D space. In such a situation all the connected rooms are physically located

next to each other and an Euclidean map (schematic plan) of the institution can be generated.

The Map is basically an alternative representation of the Performative Model. Every *Building* is the 3D representation of the EISpec in the form of a 3-dimensional house. A number of such houses are contained inside a Garden. The *Building* variable is responsible for the appearance of the institutional building. To make it look similar to a house this variable is associated with a 3-dimensional model of the house and is enhanced by additional 3D components and textures.

The constraints of this schema are:

- The Rooms of the *Building* are the same as in the *VI* schema;
- The Doors of the *Building* are the same as in the *VI* schema;
- The summary of all the room sizes is smaller than the Building size.

<i>VI</i> Euclid
<i>VI</i>
<i>Building</i> : <i>3DBuilding</i>
<i>Building.Rooms</i> = <i>Rooms</i>
<i>Building.Doors</i> = <i>Doors</i>
$\sum_{r \in \text{Rooms}} r.size < \text{Building.size}$

The schema *3DModel* refers to the 3D representation of any Object in the Virtual World. The details of this schema are not specified due to the fact that we want to keep it independent from any particular technology.

[3DModel]

The *Appearance* schema controls the appearance of the avatars associated with different agents. This may include textures, 3D meshes etc. Again, we want to keep it independent from any technology and, therefore, do not specify the details here.

[Appearance]

The *3DObject* schema extends *3DModel* schema. *3DObject* provides an easy access to the dimensions of the object via *width*, *depth* and *height*. It also specifies the *size* of the object, which is defined as (*width* * *depth*). One of the examples of the *size* usage is

presented below for identifying doors. The *size* of a door is equal to zero, which means a door is flat. *Events* and *actions* are used to specify standard events the object receives and the responses to these events that it can produce. An event does not necessarily generate a response (action). If the institutional infrastructure prohibits the execution of an action then the event is ignored.

$3DObject$ $3DModel$ $width : \mathbb{N}$ $depth : \mathbb{N}$ $height : \mathbb{N}$ $size : \mathbb{N}$ $events : \mathbb{P} Event$ $actions : \mathbb{P} Action$
$size = width * depth$

The *Event* schema is attached to each *3DObject* to control its dynamic behavior. It specifies possible events that can happen in the 3DWorld that may result some response from the *3DObject*. The examples of such events include clicks, collisions etc.

[Event]

The *Action* schema specifies a response that a *3DObject* can generate to an event. The examples of actions include opening the doors, changing the Object shape etc.

[Action]

The *Obj* schema is used for specifying any type of 3D components. The *name* of the *Obj* assigns a unique identifier to every instance. In contrast to *3DObject* the *Obj* schema is extended with the possibility of having multiple appearances (*models*). This may help if during the execution phase an object has to experience some transformation in response to different events. The rest of the variables are the same as in the *3DObject* but have a different meaning.

The predicates of this schema specify that:

- The *width* is equal to the maximum width between all the *models*;
- The *depth* is equal to the maximum depth between all the *models*;
- The *height* is equal to the maximum height between all the *models*;

- The *size* is equal to $\text{width} * \text{depth}$;
- The *events* set is the union of all the *events* sets of all the models;
- The *actions* set is the union of all the *actions* sets of all the models.

<i>Obj</i>
$\text{name} : \text{ObjID}$ $\text{models} : \mathbb{P} \text{3DObject}$ $\text{width} : \mathbb{N}$ $\text{depth} : \mathbb{N}$ $\text{height} : \mathbb{N}$ $\text{size} : \mathbb{N}$ $\text{events} : \mathbb{P} \text{Event}$ $\text{actions} : \mathbb{P} \text{Action}$
$\text{width} = \max(\text{models.width})$ $\text{depth} = \max(\text{models.depth})$ $\text{height} = \max(\text{models.height})$ $\text{size} = \text{width} * \text{depth}$ $\text{events} = \bigcup_{m:\text{models}} m.\text{events}$ $\text{actions} = \bigcup_{m:\text{models}} m.\text{actions}$

The *Door* schema provides a specification for the concept of a door. Every *Door* is simply a special type of *Obj* with special characteristics. Therefore, the *Door* extends the *Obj* schema. The *doorType* specifies the look and feel of the door. If the *doorType* is *normalDoor* – the door will be a flat rectangle with some texture. In the case of the *teleport* the *Door* may have any possible appearance. This type should be used for non-Euclidean institutions. The *Door* of type *lift* should be used for the cases when there are multiple executions of a *Scene*. In this situation the corresponding Rooms will be put on top of the original room (creating different floor). The appearance of the *Door* should be similar to the appearance of an elevator, where pressing a button will mean moving to another floor (which represents a particular execution of the corresponding scene).

The only limitation this schema applies is that the size of any *Door* of type *flatDoor* has to be equal to zero, meaning that the *Door* is flat on either *width* or *depth*.

<i>Door</i>
Obj $\text{doorType} : \{\text{flatDoor}, \text{teleport}, \text{lift}\}$
$\text{doorType} = \text{flatDoor} \Rightarrow \text{size} = 0$

The *Room* schema defines the concept of a room in a Virtual Institution. A *Room*

is a 3-dimensional space separated by walls. It inherits the *3DObject* scheme and uses it to specify the 3D model of the room, name, width, depth and height of the room. The perimeter of the room is accessible via *size*. Each room may contain a number of different 3D Objects (*RObject*), which help to enhance the immersive experience of the users. It also contains a set of doors, which connect this room with transitions that are connected to the corresponding scene in the *PerformativeModel*. The *doorPos* function is used to position doors inside the room. The limitations specified for the *Room* schema indicate that:

- The *doorPos* function is defined on the *Doors* set;
- Any door has to be placed into one of the room's walls, and its *y* coordinate has to be equal to zero (so that the door is positioned on the ground level).

<p><i>Room</i></p> <hr/> <p><i>3DObject</i> <i>name</i> : <i>RoomID</i> <i>RObjects</i> : \mathbb{P} <i>Obj</i> <i>Doors</i> : \mathbb{P} <i>Door</i> <i>doorPos</i> : <i>Obj</i> \leftrightarrow $R \times R \times R$</p> <hr/> <p>dom <i>doorPos</i> = <i>Doors</i> dom <i>objectPos</i> = <i>RObjects</i> \forall <i>door</i> : <i>Doors</i> $\{x, y, z\} = \text{doorPos}(\text{door}) \bullet$ $(x = \text{width} \vee x = 0) \vee (z = \text{depth} \vee z = 0) \wedge (y = 0)$</p>
--

The *3DBuilding* schema is a 3D representation of one institution in terms of a 3-dimensional house. The height is used to increase the dimensions of the *Map*. The schema extends the *3DObject* in order to specify the facade of the house. The *entrance* is the particular *Door* through which the building can be entered.

The limitations this schema applies are:

- The height of each room is equal to the height of the building;
- The height of each door is equal to the height of the building;
- The entrance door into the building belongs to the set of all doors of the institution.

3DBuilding

Map

3DObject

height : R

entrance : *Door*

$\forall r : \text{Room} \bullet r.\text{height} \leq \text{height}$

$\forall d : \text{Door} \bullet d.\text{height} \leq \text{height}$

$\text{entrance} \subseteq \text{Doors}$

The *Map* schema, which is inherited by every *3DBuilding* represents the schematic plan of the institution. For visualization purposes, this plan stores the positions of the doors and rooms inside the corresponding institutional building. The *Rooms* and *Doors* are independent objects generated by the *VI Euclid* schema. The *Map* schema only sets the positions for these objects and provides their 2-dimensional representation. This information is further used for the visualization of the actual schematic plan of the institution. But in this specification we are not concerned with the technical details of the visualization and do not present any visualization of related information.

The limitations for this schema include the following:

- The *roomPos* function is defined on the *Rooms* set;
- The *doorPos* function is defined on the *Doors* set;
- The set of doors associated with any room is a subset of all doors of the map;
- The doors inside every room (local coordinates) are located on the same position as corresponding doors on the map (global coordinates);
- The *width* and *depth* of each room is bigger than the standard depth of the door (*DoorDepth* constant);
- All rooms are uniquely positioned and can only overlap by sharing a wall;
- Each two connected rooms are physically placed next to each other, meaning that those two rooms at least share a part of a wall and this joint wall should be big enough to locate a door;
- All doors are uniquely positioned and do not overlap;
- A door has to be placed inside the shared wall of the two connected rooms. Note that if this wall is horizontal – the depth of the door is equal to “0”, if the wall is vertical – the width of the door is equal to “0”.

Map

Rooms : $\mathbb{P} \text{ Room}$

Doors : $\mathbb{P} \text{ Door}$

width : \mathbb{N}

depth : \mathbb{N}

roomPos : $\text{Room} \leftrightarrow R \times R$

doorPos : $\text{Door} \leftrightarrow R \times R$

dom *roomPos* = *Rooms*

dom *doorPos* = *Doors*

$\forall r : \text{Rooms} \bullet r.\text{Doors} \subseteq \text{Doors}$

$\forall r : \text{Rooms} \mid \text{roomDoors} = r.\text{Doors} \mid \{x_r, y_r\} = \text{roomPos}(r) \bullet$

$\forall d : \text{roomDoors} \mid \{x_{abs}, y_{abs}\} = \text{doorPos}(d) \mid$
 $\{x_{rel}, y_{rel}\} = r.\text{doorPos}(d) \bullet$

$x_{abs} = x_r + x_{rel} \wedge y_{abs} = y_r + y_{rel}$

$\forall r : \text{Rooms} \bullet r.\text{width} > \text{DoorDepth} \wedge r.\text{depth} > \text{DoorDepth}$

$\forall r_1, r_2 : \text{Rooms} \mid \{x_1, y_1\} = \text{roomPos}(r_1) \mid \{x_2, y_2\} = \text{roomPos}(r_2) \bullet$

$(x_1, x_1 + r_1.\text{width}) \cap (x_2, x_2 + r_2.\text{width}) = \emptyset \vee$

$(y_1, y_1 + r_1.\text{depth}) \cap (y_2, y_2 + r_2.\text{depth}) = \emptyset$

$\forall r_1, r_2 : \text{Rooms} \mid r_1.\text{Doors} \cap r_2.\text{Doors} \neq \emptyset$

$\{x_1, y_1\} = \text{roomPos}(r_1) \mid \{x_2, y_2\} = \text{roomPos}(r_2) \bullet$

$\#[[x_1, x_1 + r_1.\text{width}] \cap [x_2, x_2 + r_2.\text{width}]] > \text{DoorDepth} \vee$

$\#[[y_1, y_1 + r_1.\text{depth}] \cap [y_2, y_2 + r_2.\text{depth}]] > \text{DoorDepth}$

$\forall d_1, d_2 : \text{Doors} \mid \{x_1, y_1\} = \text{doorPos}(d_1) \mid \{x_2, y_2\} = \text{doorPos}(d_2) \bullet$

$[x_1, x_1 + d_1.\text{width}] \cap [x_2, x_2 + d_2.\text{width}] = \emptyset \vee$

$[y_1, y_1 + d_1.\text{depth}] \cap [y_2, y_2 + d_2.\text{depth}] = \emptyset$

$\forall r_1, r_2 : \text{Room} \mid \text{jointDoors} = r_1.\text{Doors} \cap r_2.\text{Doors} \neq \emptyset \mid$

$\{x_1, y_1\} = \text{roomPos}(r_1) \mid \{x_2, y_2\} = \text{roomPos}(r_2) \bullet$

$\exists d : \text{jointDoors} \mid \{x_d, y_d\} = \text{doorPos}(d) \bullet$

$x_2 = x_1 + r_1.\text{width} \Rightarrow x_d = x_2 \wedge d.\text{width} = 0$

$x_2 = x_1 - r_2.\text{width} \Rightarrow x_d = x_1 \wedge d.\text{width} = 0$

$y_2 = y_1 + r_1.\text{depth} \Rightarrow y_d = y_2 \wedge d.\text{depth} = 0$

$y_2 = y_1 - r_2.\text{depth} \Rightarrow y_d = y_1 \wedge d.\text{depth} = 0$

The *AvatarType* schema connects the avatar name with the appearance (dress) of the corresponding 3D Object. Initially each role in an institution is associated with an *AvatarType*. This schema extends the *3DObject* schema (which specifies the 3D mesh of the avatar, possible *actions* and *events*).

The *actions* are standard behaviors an avatar can perform: move, jump, etc.

The *events* represent all the possible actions that may change the avatar state and may generate a response in terms of an *Action*.

The *gestures* is a set of standard articulation related behaviors connected to each particular *AvatarType*. This can be bowing, hand waving, etc.

The constraints of this schema are as follows:

- The sets of *gestures* and *actions* do not overlap;
- The *size* of an avatar is always smaller than the *MinPSpace* constant. This constant defines the minimum space needed inside the room for each participant.

<i>AvatarType</i>
<i>3DObject</i> <i>name</i> : <i>AvatarID</i> <i>dress</i> : <i>Appearance</i> <i>gestures</i> : \mathbb{P} <i>Action</i>
<hr/> <i>gestures</i> \cap <i>actions</i> = \emptyset <i>size</i> < <i>MinPSpace</i>

The *ActionMessageTable* creates a runtime connection between the Electronic Institution and its 3D visualization. It defines the relationships between some events and actions in the 3D Virtual World and messages of the Electronic Institution. Only the events and actions that have an institutional impact are participating in these relationships.

The predicates contained in this schema specify the following constraints:

- Only a subset of *allEvents* is involved in the *eventMessage* relationship;
- Only a subset of *allActions* is involved in the *actionMessage*
- The *actionMessage* and the *eventMessage* relationships define the values for every item in *allMessages*;
- There is no event or action that will have the same message assigned to it via *eventMessage* or *actionMessage*.

<i>ActionMessageTable</i>
<i>allEvents</i> : \mathbb{P} <i>Event</i> <i>allActions</i> : \mathbb{P} <i>Action</i> <i>allMessages</i> : \mathbb{P} <i>IllocutionParticle</i> <i>eventMessage</i> : <i>Event</i> \leftrightarrow <i>Message</i> <i>actionMessage</i> : <i>Action</i> \leftrightarrow <i>Message</i>
<hr/> $\text{dom } eventMessage \subseteq allEvents$ $\text{dom } actionMessage \subseteq allActions$ $\text{ran}(eventMessage) \cup \text{ran}(actionMessage) = allMessages$ $\forall e : allEvents, a : allActions \bullet$ $eventMessage(e) \neq actionMessage(a)$

3.6.3 The State of a Virtual Institution at Run Time

The state of any Virtual Institution is a combination of the *ElectronicInstitutionState* and either the *3DEuclideanWorldState* or *3DWorldState*, depending on whether the Euclidean or non-Euclidean model was selected for the visualization.

The *ElectronicInstitutionState* is controlling the runtime state of the institutional infrastructure. Here we do not present the complete Z-specification of it but only show the most basic components (the complete specification is presented in Appendix A). The *ElectronicInstitutionState* defines the states for all the data types specified in the *ElectronicInstitution* schema. These include:

- *SocialModelState* – the state of the *SocialModel*
- *NormativeModelState* – the state of the *NormativeModel*
- *CommunicationModelState* – the state of the *CommunicationModel*
- *PerformativeModelState* – the state of the *PerformativeModel*

<i>ElectronicInstitutionState</i> <i>SocialModelState</i> <i>NormativeModelState</i> <i>CommunicationModelState</i> <i>PerformativeModelState</i>

The *3DEuclideanWorldState* is used for Virtual Institutions visualized in a fully Euclidean way. It extends the *GardenState* and stores the states of all the institutions inside the *Garden*.

<i>3DEuclideanWorldState</i> <i>GardenState</i> <i>InstitutionStates</i> : \mathbb{P} <i>VIeuclidState</i>
--

The *3DWorldState* has similar functionality as the *3DEuclideanWorldState* schema. The only difference is that it assumes that the Virtual Institution follows a non-Euclidean model and, therefore, the *InstitutionStates* are of type *VIState*.

<i>3DWorldState</i> <i>GardenState</i> <i>InstitutionStates</i> : \mathbb{P} <i>VIState</i>

The *GardenState* schema defines how the *Garden* should be enacted at runtime. It extends the original *Garden* specification and adds *ObjectStates*, *AvatarStates* and *objPos*. The *ObjectStates* stand for the states of all of the objects contained inside the *Garden*. The *AvatarStates* refer to the states of participants that are inside the *Garden*. The *objPos* function determines the positions of the objects inside the *Garden*. The objects can change their positions over time.

The predicates of this schema say that:

- The *objPos* function is defined for *Objects* set;
- All the objects should be positioned inside the *Garden*;
- None of the garden objects overlap with an institution;
- The unique *name* identifiers are not assigned to the participants until they enter any of the institutions; Before that the ID of each participant is set to “0”.

<i>GardenState</i>
<p><i>Garden</i></p> <p><i>ObjectStates</i> : $\mathbb{P} \text{ObjState}$</p> <p><i>AvatarStates</i> : $\mathbb{P} \text{AvatarState}$</p> <p><i>objPos</i> : $\text{Obj} \leftrightarrow R \times R \times R$</p> <hr/> <p>$\text{dom } objPos = \text{Objects}$</p> <p>$\forall o : \text{Obj} \mid \{x, y, z\} = objPos(o) \bullet$ $x \in [0, width - o.width] \wedge z \in [0, depth - o.depth]$</p> <p>$\forall i : \text{Institutions}, o : \text{Objects} \mid$ $\{x_i, z_i\} = institutionPos(i) \mid \{x_o, z_o\} = objPos(o) \bullet$ $[x_i, x_i + i.width] \cap [x_o, x_o + o.width] = \emptyset \vee$ $[z_i, z_i + i.depth] \cap [z_o, z_o + o.depth] = \emptyset$</p> <p>$\forall a_s : \text{AvatarStates} \bullet a_s.name = 0$</p>

The *BasicVISate* schema specifies the runtime state of a Basic Institution (not taking into account whether the Euclidean or non-Euclidean model is used). It extends the *ElectronicInstitutionState* enhancing it with a set of additional data types. The *RoomStates* control the runtime execution of the set of rooms of the institution. The *agentAvatar* function connects the agents of the *ElectronicInstitutionState* level to the avatars of the *VIEuclidState* level by setting the names of the avatars equal to the names of the corresponding agents and setting the correct avatar type (accordingly with the role of the corresponding agent). The *entryRoom* specifies which *Room* corresponds to the entry scene in the Institution. The *exitRoom* defines the room from which the user exits the institution. The *entryDoor* defines the *Door* through which the Virtual Institution can be

entered. The *exitDoor* defines the *Door* through which the user can leave the building and appear in the *Garden*. The *sceneRoom* function creates the mapping between the *Scenes* of the Normative Control Layer to *Rooms* of the Visual Interaction Layer. The *transitionRoom* function creates the mapping between the *Transitions* of the Normative Control Layer to *Rooms* of the Visual Interaction Layer. The *roomScene* function maps the set of *Rooms* to the set of *Scenes*. The *roomTransition* function maps the set of *Rooms* to the set of *Transitions*.

The predicates of this schema define the following:

- The domain of the *agentAvatar* function is all agents present in the Normative Control Layer;
- The *agentAvatar* function returns values from the set of all names of avatars present in all the rooms of the Visual Interaction Layer;
- Every agent in the Electronic Normative Control Layer has a corresponding avatar in the Visual Interaction Layer with the same name and role;
- For every agent present in a *Scene*, the corresponding avatar is present in the *Room*, that is the visualization of this *Scene*;
- For every agent present in a *Transition*, the corresponding avatar is present in the *Room*, that is the visualization of this *Transition*;
- The *entryRoom* should be one of the rooms of the institution;
- The *exitRoom* should be one of the rooms of the institution;
- The *entryDoor* should be one of the doors in the *entryRoom*;
- The *exitDoor* should be one of the doors in the *exitRoom*;
- The *entryDoor* is always open for any participant;
- The domain of the *sceneRoom* function is *allscenes*;
- The domain of the *transitionRoom* function is *alltransitions*;
- The *transitionRoom* and the *sceneRoom* functions result in the *RoomStates* set.
- The *roomScene* function results in the *allscenes* set;
- The *RoomTransition* function results in the *alltransitions* set;

BasicVIState

ElectronicInstitutionState
RoomStates : \mathbb{P} *RoomInstance*
agentAvatar : *AgentID* \leftrightarrow *AvatarState*
entryRoom : *RoomInstance*
exitRoom : *RoomInstance*
entryDoor : *DoorState*
exitDoor : *DoorState*
sceneRoom : *SceneInstance* \leftrightarrow *RoomInstance*
transitionRoom : *TransitionInstance* \leftrightarrow *RoomInstance*
roomScene : *RoomInstance* \leftrightarrow *SceneInstance*
roomTransition : *RoomInstance* \leftrightarrow *SceneInstance*

dom *agentAvatar* = *allagents*
ran *agentAvatar* = \cup *RoomStates.AvatarStates.name*
 \forall *agentID* : *allagents* | *avatarState* = *agentAvatar(agentID)* •
 avatarState.name = *agentID* \wedge
 avatarState.role = *agentsroles(agentID)*
 \forall *s* : *SceneInstance* | *a* : *s.Agents* | *r* = *sceneRepresentation(s)* •
 \exists_1 *a_s* \in *RoomStates.AvatarStates* = *agentAvatar(a)*
 \forall *t* : *TransitionInstance* | *a* : *t.Agents* | *r* = *transitionRepresentation(t)* •
 \exists_1 *a_s* \in *RoomStates.AvatarStates* = *agentAvatar(a)*
entryRoom \in *Rooms*
exitRoom \in *Rooms*
entryDoor \in *entryRoom.DoorStates*
exitDoor \in *exitRoom.DoorStates*
 \forall *agentID* : *allagents* • *entryDoor.isDoorOpen(agentID)* = *true*
dom *sceneRoom* = *allscenes*
dom *transitionRoom* = *alltransitions*
ran *sceneRoom* \cup **ran** *transitionRoom* = *RoomStates*
ran *roomScene* = *allscenes*
ran *roomTransition* = *alltransitions*

The *3DEuclideanState* schema specifies the runtime state of a Virtual Institution, which uses the Euclidean model. It extends the *BasicVIState* schema enhancing it with *VIEuclid*.

VIEuclidState

BasicVIState
VIEuclid

The *VIState* is similar to the *VIEuclidState* with the only difference that it is used to specify the Virtual Institutions that do not follow the Euclidean model. Therefore *VIState* uses *VI* instead of *VIEuclid*.

<i>VIState</i>
<i>BasicVIState</i>
<i>VI</i>

The *Matrix* schema defines the transformations that can be applied to any *Obj* and its subclasses. These transformation include rotation, twisting, scaling etc. As we do not limit Virtual Institutions by the use of a particular technology for the implementation and visualization purposes, we will not elaborate on the specific details of the *Matrix*.

[Matrix]

The *AvatarState* is used to specify the details that change the state of every participant of a Virtual Institution. These include:

- The *name* is a unique identifier of the participant;
- The *role* is the role the Avatar plays in the institution;
- The *AvatarPosition* is the location of each participant that changes over time;
- The *AvatarTransform* includes a set of vectors that define the head rotation, body yaw, scaling factor etc;
- The *lastGesture* stores the last gesture executed by a participant;
- The *lastAction* refers to the last action that was performed by an avatar;
- The *lastEvent* stands for the last event a participant responded (or was about to respond) to;
- In case the participant is currently executing a gesture or an action they can be accessible via *currentGesture* or *currentAction*.

The predicates of this schema apply the following limitations:

- The *currentAction* is a subset of all possible actions for this type of participants.
- The *currentGesture* is a subset of all possible gestures for this type of participants.
- The *lastEvent* is a subset of all possible events for this type of participants.
- The *lastAction* is a subset of all possible actions for this type of participants.
- The *lastGesture* is a subset of all possible gestures for this type of participants.

AvatarState

AvatarType

name : *AvatarID*

role : *RoleID*

AvatarPosition : $R \times R \times R$

AvatarTransform : *Matrix*

lastGesture : *Action*

lastAction : *Action*

lastEvent : *Event*

currentGesture : *Action*

currentAction : *Action*

$currentAction \subseteq actions$

$currentGesture \subseteq gestures$

$lastEvent \subseteq events$

$lastAction \subseteq actions$

$lastGesture \subseteq gestures$

The *DoorState* schema defines the state of every *Door* in a Virtual Institution. As participants with different roles may have totally different permissions to access a room that the *Door* leads to, the *DoorState* is different for every participant. The *isDoorOpen* function tells whether the *Door* should be open for a participant with a particular *AvatarID* or not. The only predicate of this schema limits the domain of the *isDoorOpen* function to the set of all participants.

DoorState

Door

$isDoorOpen : AvatarID \leftrightarrow \{true, false\}$

$dom\ isDoorOpen = allAvatars$

The *ObjState* schema specifies how different Objects (inside the *Garden* or inside different *Rooms*) behave at runtime. It extends the *Obj* schema and similar to the *AvatarState* tracks *currentAction*, *lastEvent*, *lastAction*. Additionally, it also stores the runtime information about the transformation applied to the object in *ObjTransform*. The visibility of the Object is determined by the value of the *visible* variable. Note that each *Obj* can be manipulated (transformed) at runtime, so its dimensions may change. The values of *width*, *depth*, *height* and *size* are subject to change after the *ObjTransform* function is applied. After each transformation the lower left corner of the bounding box is still considered to be the position of the *Obj*.

The predicates of this schema apply the following limitations:

- The *currentAction* is a subset of all possible actions for this type of participants;

- The *lastEvent* is a subset of all possible events for this type of participants;
- The *lastAction* is a subset of all possible actions for this type of participants.

ObjState <hr/> Obj $\text{ObjTransform} : \text{Matrix}$ $\text{visible} : \{\text{true}, \text{false}\}$ $\text{currentAction} : \text{Action}$ $\text{lastEvent} : \text{Event}$ $\text{lastAction} : \text{Action}$ <hr/> $\text{currentAction} \subseteq \text{actions}$ $\text{lastEvent} \subseteq \text{events}$ $\text{lastAction} \subseteq \text{actions}$
--

A *RoomInstance* represents a runtime instance of a room. A room instance consists of a copy of the room data structure from which it was instantiated, the set of avatars and their roles (which are *functional* since within a room instance an avatar can only have one role) along with a unique room identifier. Also we introduce a counter that records how much time has elapsed since the last successful avatar action.

Function *objectPos* is used for positioning the objects inside the room.

The predicates of this schema specify that:

- The *objectPos* function is defined on the *RObjects* set.
- Any *RObject* has to be positioned within the bounds of the room

RoomInstance <hr/> $\text{orig} : \text{Room}$ $\text{ObjectStates} : \mathbb{P} \text{ObjState}$ $\text{AvatarStates} : \mathbb{P} \text{AvatarState}$ $\text{DoorStates} : \mathbb{P} \text{DoorState}$ $\text{objectPos} : \text{Obj} \mapsto R \times R \times R$ $\text{avatarState} : \text{AgentID} \mapsto \text{AvatarState}$ <hr/> $\text{dom objectPos} = \text{orig.RObjects}$ $\forall o : \text{orig.RObjects} \mid \{x, y, z\} = \text{objectPos}(o) \bullet$ $(x \in [0, \text{orig.width} - o.width] \wedge y \in [0, \text{orig.height} - o.height] \wedge$ $z \in [0, \text{orig.depth} - o.depth])$
--

The *MapState* schema specifies how the map should evolve at runtime. It contains *activeRooms* and *activeDoors*, which determine which rooms and doors are active at any given moment.

The predicates of this schema only assign the *activeRooms* and *activeDoors* with the appropriate sets.

$MapState$ $orig : Map$ $activeRooms : \mathbb{P} RoomInstance$ $activeDoors : \mathbb{P} DoorState$
$activeRooms \in Rooms$ $activeDoors \in Doors$

We do not specify the state of the *3DBuilding* as it is not supposed to change at runtime and should remain constant.

3.6.4 Initialization of the States

In this section we provide the details of the initialization of the states of the specified schemas. Note that we do not describe the schemas used for the initialization of the Electronic Institutions here (but their description can be found in Appendix A).

The *Init3DWorld* schema defines how the initialization of the 3DWorld is going to occur. It basically indicates that the Garden has to be initialized via *InitGarden* and each institution has to be initialized via *InitVI*.

$Init3DWorld$ $InitGarden$ $InitVI$

The *Init3DEuclideanWorld* schema is specified in a similar way to *Init3DWorld* with the only difference being that it is concerned with Euclidean institutions.

$Init3DWorld$ $InitGarden$ $InitVIEuclid$

The *InitGarden* schema specifies the initial state of the *Garden*. It extends the *GardenState* schema and defines two additional functions. The *objInitialPos* function defines the initial positions of all the objects located inside the *Garden* and the *objInitialTransform* function defines the initial transformations of these objects.

The predicates of this schema apply the following limitations:

- The *objInitialPos* and *objInitialTransform* are defined on the *orig.Objects* set;

- The initial position of every object should be within the bounds of the *Room*;
- All objects are on their initial positions defined by the values of *objInitialPos* function;
- The initial transformation of every object in the *Garden* is defined by the values of the *objInitialTransform* function;
- In the initial state there are no participants in the *Garden*.

$\begin{array}{l} \textit{InitGarden} \\ \textit{GardenState} \\ \textit{objInitialPos} : \textit{Obj} \mapsto R \times R \times R \\ \textit{objInitialTransform} : \textit{Obj} \mapsto \textit{Matrix} \\ \text{dom } \textit{objInitialPos} = \text{dom } \textit{objInitialTransform} = \textit{Objects} \\ \forall o : \textit{Objects} \mid \{x, y, z\} = \textit{objInitialPos}(o) \bullet \\ \quad (x \in [0, \textit{width} - o.\textit{width}] \wedge y \in [0, \textit{height} - o.\textit{height}] \wedge \\ \quad z \in [0, \textit{depth} - o.\textit{depth}]) \\ \forall o : \textit{Objects} \bullet \textit{objPos}(o) = \textit{objInitialPos}(o) \\ \forall os : \textit{ObjectStates} \bullet os.\textit{ObjTransform} = \textit{objInitialTransform}(os) \\ \textit{AvatarStates} = \emptyset \end{array}$
--

The *InitVIEuclid* schema defines the initial state of an institution visualized in a fully Euclidean way. It extends the *VIEuclidState*. The predicates of the schema define that on the initialization step the institution consists of only two rooms: entry and exit.

$\begin{array}{l} \textit{InitVIEuclid} \\ \textit{VIEuclidState} \\ \textit{RoomStates} = \{\textit{entryRoom}, \textit{exitRoom}\} \end{array}$

The *InitVI* schema defines the initial state of an institution visualized in a non-Euclidean way. It extends the *VIState*. The only predicate of the schema means that on the initialization step the institution consists of only two rooms: entry and exit.

$\begin{array}{l} \textit{InitVI} \\ \textit{VIState} \\ \textit{RoomStates} = \{\textit{entryRoom}, \textit{exitRoom}\} \end{array}$

The *InitRoomInstance* schema provides the specification of the initial state of each room. It extends the *RoomInstance* schema and defines two additional functions. The

objInitialPos function defines the initial positions of all the objects located inside the *Room* and the *objInitialTransform* function defines the initial transformations of these objects.

The predicates of this schema apply the following limitations:

- The initial instance of any room has no avatars;
- All the doors are closed for any participant;
- All objects are on their initial positions defined by the values of *objInitialPos* function;
- The initial transformation of every object is defined by the values of *objInitialTransform* function;
- The *objInitialPos* and *objInitialTransform* are defined on the *orig.RObjects* set;
- The initial position of every object should be within the bounds of the *Room*.

<p style="margin: 0;"><i>InitRoomInstance</i></p> <hr style="border: 0.5px solid black;"/> <p style="margin: 0;"><i>RoomInstance</i></p> <p style="margin: 0;"><i>objInitialPos</i> : <i>Obj</i> \leftrightarrow $R \times R \times R$</p> <p style="margin: 0;"><i>objInitialTransform</i> : <i>Obj</i> \leftrightarrow <i>Matrix</i></p> <hr style="border: 0.5px solid black;"/> <p style="margin: 0;"><i>AvatarStates</i> = \emptyset</p> <p style="margin: 0;">$\forall d : \text{DoorStates}, a = d.\text{Avatars} \bullet$ $a = \emptyset \vee d.\text{isDoorOpen}(a) = \text{false}$</p> <p style="margin: 0;">$\forall o : \text{ObjectStates} \bullet o.\text{currentEvent} = \emptyset \wedge o.\text{currentAction} = \emptyset \wedge$ $o.\text{lastEvent} = \emptyset \wedge o.\text{lastAction} = \emptyset$</p> <p style="margin: 0;">$\forall o : \text{Obj} \bullet \text{objPos}(o) = \text{objInitialPos}(o)$</p> <p style="margin: 0;">$\forall o : \text{Obj} \bullet \text{objTransformation}(o) = \text{objInitialTransform}(o)$</p> <p style="margin: 0;">dom <i>objInitialPos</i> = dom <i>objInitialTransform</i> = <i>orig.RObjects</i></p> <p style="margin: 0;">$\forall o : \text{orig.RObjects} \mid \{x, y, z\} = \text{objInitialPos}(o) \bullet$ $(x \in [0, \text{orig.width} - o.\text{width}] \wedge y \in [0, \text{orig.height} - o.\text{height}] \wedge$ $z \in [0, \text{orig.depth} - o.\text{depth}])$</p>
--

The *InitAvatar* schema specifies how the initialization of the instances of *AvatarState* should happen. This schema extends the *AvatarState*. It sets the avatar position to the starting position in the *Garden*, clears the *Avatar* transformation and forces all the other variables being empty.

<i>InitAvatar</i>
<i>AvatarState</i>
<i>role</i> = 0 <i>name</i> = 0 <i>AvatarPosition</i> = <i>InitPos</i> <i>AvatarTransform</i> = 0 <i>lastGesture</i> = \emptyset <i>lastAction</i> = \emptyset <i>lastEvent</i> = \emptyset <i>currentGesture</i> = \emptyset <i>currentAction</i> = \emptyset

The *InitDoor* schema specifies how the initialization of the *Doors* should happen. It extends the *DoorState* schema. The only predicate forces the *Door* to be initially closed for every participant of the Virtual Institution.

<i>InitDoor</i>
<i>DoorState</i>
$\forall a : AvatarID \bullet isDoorOpen(a) = false$

The *InitObj* schema specifies how the *Obj* instances should be initialized. It extends the *ObjState* schema.

The predicates of this schema specify the following limitations:

- All the objects are initially visible.
- There is no *currentAction* being executed on the initialization stage.
- There is no *lastEvent* yet received on the initialization stage.
- There is no *lastAction* yet executed on the initialization stage.

<i>InitObj</i>
<i>ObjState</i>
<i>visible</i> = <i>true</i> <i>currentAction</i> = \emptyset <i>lastEvent</i> = \emptyset <i>lastAction</i> = \emptyset

The *InitMap* scheme specifies the initialization of the Map for the Euclidean case of an institution.

The predicates of this schema specify that:

- Initially only entryRoom and ExitRoom are present.
- Initially only entranceDoor and exitDoor are present.

<i>InitMap</i>
<i>MapState</i>
$activeRooms = \{entryRoom, exitRoom\}$ $activeDoors = \{entranceDoor, exitDoor\}$

3.6.5 Operations

Next we define which operations that change the state of a Virtual Institution.

1. *Entering the Garden*

This schema defines what changes happen in the environment when an avatar enters the Garden. The only change to the environment caused by this operation is the inclusion of the new avatar into the *AvatarStates* set of the *GardenState* schema.

<i>EnterGarden</i>
$\Delta GardenState$ $av? : AvatarState$
$av? \notin AvatarStates$ $av'.name = \#Avatarstates + 1$ $AvatarStates' = AvatarStates \cup \{av'\}$

2. *Moving Garden Object*

We do not prohibit moving the objects around the garden. The *MoveGardenObject* schema controls the execution of this operation. It changes the *ObjPos* in the *GardenState* schema to specify how the position is updated.

<i>MoveGardenObject</i>
$\Delta GardenState$ $obj? : ObjectStates$ $pos? : ObjPos$ $newPos? : R \times R \times R$
$obj? \in ObjectStates$ $ObjPos'(obj) = newPos$

3. *Walking in the Garden*

The only thing that changes when an avatar moves within the garden is its position. The change of the position changes the state of the Garden, so the *GardenState* schema is updated accordingly.

<i>AvatarMove</i>
$\Delta GardenState$
$av? : AvatarState$
$avPos? : R \times R \times R$
$av? \in AvatarStates$
$av'.AvatarPosition = avPos?$

4. *Activating a Room*

Before visualization, each of the rooms has to be activated. The *ActivateRoomInstance* schema shows how this process happens. This operation is applied to a *MapState* schema. It changes the *activeRooms* set by including the given room and also adds the doors associated with this room to the *activeDoors* set of the *MapState* schema. This operation should be executed as a result of creating the corresponding scene in the Normative Control Layer.

<i>ActivateRoomInstance</i>
$\Delta MapState$
$r? : RoomInstance$
$activeRooms? = activeRooms \cup r?$
$activeDoors? = activeDoors \cup r?.DoorStates$

5. *Requesting to Open a Door*

The *RequestOpenDoor* schema defines the operation for an avatar to ask to enter or leave a room. Before leaving a current room and moving to another one, the door connecting these two rooms has to be opened. The process of opening and closing doors is controlled by the Electronic Institution. The institution is accessed via the *BasicVIState* schema. First, we find the *currentRoom* (representing the room where the avatar is currently located) by accessing the *RoomStates* set and searching for the item in this set that contains the given avatar. Then we map this room to either a scene or a transition in the Normative Control Layer. The room connected to it (and the corresponding scene or transition) are discovered by finding a room that contains the same door as the door the avatar is requesting to

open. After this, the door will only be open under the following conditions:

- *if the current room is associated with a transition*, then the scene the agent is trying to enter is not allowed to contain the same agent; this scene should be in one of the states where the agents with the given role are permitted to enter; and the number of agents already present in this scene should be less than the maximum number of agents playing the given role allowed to be present in this scene;
- *if the current room is associated with a scene*, then the current scene should be in one of the states, in which the agents playing the given role are permitted to leave the scene and the avatar should not be already present inside the transition that it tries to enter.

The *RequestOpenDoor* operation can, for example, be executed as a result of an avatar colliding with a door (and in this way expressing an intention for the door to be opened).

<i>RequestOpenDoor</i>
$\Delta DoorInstance$ $\Xi BasicVIState$ $av? : AvatarState$ $d? : DoorInstance$
$\exists_1 currentRoom : RoomStates \bullet av \in currentRoom.AvatarStates$ $currentScene = roomScene(currentRoom)$ $currentTransition = roomTransition(currentRoom)$ $d? \in currentRoom.DoorStates$ $\exists_1 nextRoom : RoomStates \bullet$ $currentRoom.DoorStates \cap nextRoom.DoorStates = d?$ $nextScene = roomScene(nextRoom)$ $nextTransition = roomTransition(nextRoom)$ $currentTransition \neq \emptyset \Rightarrow$ $av? \notin nextScene.AvatarStates \wedge$ $nextScene.currentconvstate \in$ $nextScene.sorig.type.accessconvstates(av?.role) \wedge$ $\#nextScene.agents <$ $mymax(nextScene.sorig.type.minmaxagents(av?.role)) \wedge$ $d'.isDoorOpen(?av) = true$ $currentScene \neq \emptyset \Rightarrow$ $currentScene.currentconvstate \in$ $currentScene.sorig.type.leavingconvstates(av?.role) \wedge$ $av? \notin (\mathbf{dom} nextTransition.currentroles) \wedge$ $d'.isDoorOpen(?av) = true$

6. Changing an Avatar

The *AvatarTransform* schema is responsible for changing the avatar on the fly during runtime. This includes scaling the avatar, rotating it etc. The avatar transformation is achieved through updating the *AvatarTransform* associated with the *AvatarState* set.

<i>AvatarTransform</i>
$\Delta AvatarState$
$m? : AvatarTransform$
$AvatarTransform' = m$

7. Responding to an Event

Every time a user generates an event (i.e. pressing a key on the keyboard or a mouse button, colliding, etc.) we consult with the *ActionMessageTable* whether such an event requires institutional verification. If the verification is required then the corresponding message for this event is sent to the institutional infrastructure for verification. The result of the verification will be sent back to the Virtual World in terms of a message. Only when the appropriate message is received should the action that corresponds to a generated event be visualized. This action is again extracted from the *ActionMessageTable*. The *EventResponse* schema specifies this process. It is assumed that *MakeIllocution* is defined in *ElectronicInstitutionState*.

<i>EventResponse</i>
$\exists ActionMessageTable$
$\Delta ElectronicInstitutionState$
$e? : Event$
$\forall i = eventMessage(e?) \mid i \neq \emptyset \Rightarrow$ $MakeIllocution(i)$

8. Responding to a Message

When a message is received from the institutional infrastructure we need to find the action that should be executed in response to this message. The *MessageResponse* schema defines how this should occur.

$ \begin{array}{l} \textit{MessageResponse} \\ \exists \textit{ActionMessageTable} \\ \Delta \textit{BasicVIState} \\ i? : \textit{Illocution} \end{array} $
$ \forall a = \textit{actionMessage}(i?) \mid a \neq \emptyset \Rightarrow \\ \textit{VisualizeAction}(a) $

9. Visualizing an Action

The way the actions are visualized is highly dependent on the visualization platform. Therefore, we do not define how the actions should be visualized in this specification and let the system engineer decide on the details of this process.

[VisualizeAction]

10. Creating an Alteroid

The *CreateAlteroid* schema defines how new alteroids are created. The alteroid in our case is simply another copy of the given *AvatarState* put into the target room. As this *AvatarState* is a duplicate of the original schema, it also has the same id as the original. The operation can only be executed under the following conditions:

- the given *AvatarState* is present the current room;
- the current room corresponds to a scene;
- the target room corresponds to a transition;
- there is an open door between the current room and the target room;
- the state of the scene associated to the current room permits to perform the stay-and-go operation.

If all the above conditions are satisfied – a copy of the *AvatarState* is put inside the target room and the corresponding agent joins the associated transition.

CreateAlteroid

Δ *Basic VIState*

$av? : AvatarState$

$currentRoom? : RoomInstance$

$targetRoom? : RoomInstance$

$av? \in currentRoom.AvatarStates$

$currentScene = roomScene(currentRoom)$

$targetTransition = roomTransition(desiredRoom)$

$currentScene \neq \emptyset$

$targetTransition \neq \emptyset$

$currentScene.currentconvstate \in currentScene.stgoconvstates$

$\exists d : targetRoom.DoorStates \mid d.isDoorOpen(av?) = true \Rightarrow$

$targetRoom?.AvatarStates' = targetRoom?.AvatarStates' \cup \{av?\} \wedge$

$JoinTransitionInstance(av?.name, av?.role, targetRoom?.torig)$

11. *Exiting a Room*

A room can only be exited if the exit door is open. On the level of the Electronic Institution this is more complicated, as whether the door is open or closed is determined by the state of the corresponding scene, but this process is controlled by the *RequestOpenDoor* schema. The *ExitRoom* schema specifies that the given avatar is removed from the *AvatarStates* set if the exit door is open. It also propagates the change of the state to the Normative Control Layer by removing the corresponding agent from the scene or transition associated to the given room.

ExitRoom

Δ *Basic VIState*

$av? : AvatarState$

$r? : RoomInstance$

$currentScene = sceneRoom(r?)$

$currentTransition = sceneTransition(r?)$

$av? \in AvatarStates$

$\exists d : DoorStates \mid d.isDoorOpen(av?) = true \Rightarrow$

$AvatarStates' = AvatarStates' / \{av?\}$

$currentScene \neq \emptyset \Rightarrow$

$LeaveSceneInstance(currentScene, av?.name)$

$currentTransition \neq \emptyset \Rightarrow$

$LeaveTransitionInstance(currentTransition, av?.name, av?.role)$

12. *Entering a Room*

A room can be entered by an avatar if the entrance door into this room is open. The

fact that the door is open or closed for a given avatar is strictly controlled by the underlying Electronic Institution. The *EnterRoom* schema specifies that the given avatar is included into the *AvatarStates* set if the room entrance door is open. It also propagates the change of the state to the Normative Control Layer by adding the corresponding agent into the scene or transition associated to the given room.

<i>EnterRoom</i>
$\Delta BasicVIState$ $av? : AvatarState$ $r? : RoomInstance$
$currentScene = sceneRoom(r?)$ $currentTransition = sceneTransition(r?)$ $av? \in AvatarStates$ $\exists d : DoorStates \mid d.isDoorOpen(av?) = true \Rightarrow$ $AvatarStates' = AvatarStates' \cup \{av?\}$ $currentScene \neq \emptyset \Rightarrow$ $JoinScene(currentScene, av?.name, av?.role)$ $currentTransition \neq \emptyset \Rightarrow$ $JoinTransitionInstance(av?.name, av?.role, currentTransition)$

3.7 Summary

We have introduced the concept of Virtual Institutions as 3D Virtual Worlds with normative regulation of interactions. Virtual Institutions consist of two independent conceptual layers: the Visual Interaction Layer (responsible for the visualization of participants and for providing them with interaction facilities inside the Virtual World) and the Normative Control Layer (responsible for establishing and controlling the enforcement of the interaction rules).

These two layers are integrated through the employment of the Virtual Institutions metaphor. This metaphor suggests seeing the concept of Virtual Institutions as a set of buildings surrounded by a garden. Each of the buildings represents an Electronic Institution and the interactions of participants inside the building are governed by the norms of the institution.

The participants are visualized as avatars, scenes and transitions from the Electronic Institutions Specification are represented as 3-dimensional rooms (located inside the corresponding institutional building), connections in the Performative Structure are visualized as doors and the number of participants allowed in a scene determines the size of a room. The permissions of participants to access different rooms are controlled via doors separating them. The doors are locked or unlocked for each of the participants depending

on the state of the associated Electronic Institution.

The concept of Virtual Institutions and the proposed metaphor have been illustrated with the example of the Trading Institution. The Normative Control Layer of this institution corresponds to the Trading Institution example outlined in the previous chapter.

The concept of Virtual Institutions is formally specified using Z Specification language. This formal specification provides implementation requirements for system architects, who plan to develop their systems as Virtual Institutions.

Chapter 4

Approach and Methodology

The implementation requirements expressed in the previous chapter provide the ground for the implementation of Virtual Institutions. They specify the components required for both Visual Interaction and Normative Control layers as well as describe how the system should be deployed.

In this chapter we demonstrate the proof of Virtual Institutions concept by presenting our proposal for implementation.

We start with introducing the Virtual Institutions Methodology, which is quite generic and can be applied to the development of any Virtual Institution. Next, we present the technical details behind some of the steps of this methodology and demonstrate how existing technologies can assist in completing them. Towards the end of the chapter we demonstrate the generic architecture that should be used for deployment of Virtual Institutions and outline the technological solution that was used.

4.1 Virtual Institutions Methodology

For building Virtual Institutions we propose using Virtual Institutions methodology outlined in Figure 4.1. This methodology covers the whole development process and is also supplied with the tools for deployment of Virtual Institutions. This methodology should be employed by system architects and software developers (both are further called users). In general, applying Virtual Institutions methodology requires 7 steps to be accomplished:

1. Eliciting Specification Requirements.
2. Specification of an Electronic Institution.
3. Verification of the specification.
4. Automatic Generation of the corresponding 3D environment (if needed).
5. Annotation of the Electronic Institution specification with components of the 3D Virtual World.

6. Integrating the 3D Virtual World into the institutional infrastructure.

7. Enabling Implicit Training

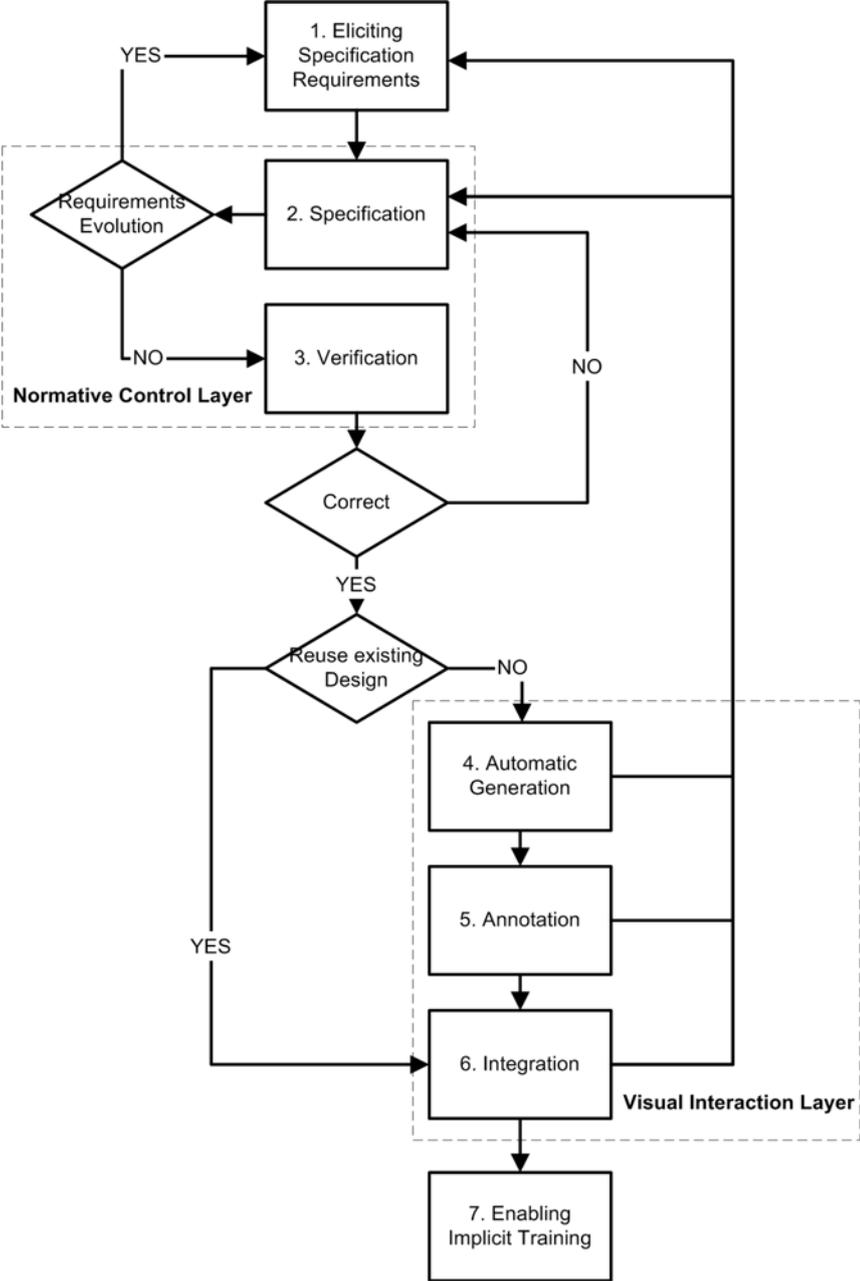


Figure 4.1: Methodology steps.

Completing the above steps will result in defining both Normative Control Layer and Visual Interaction Layer of the corresponding Virtual Institution. As shown in Figure 4.1, the specification requirements for the Normative Control Layer are derived on Step 1 of the methodology. The Normative Control Layer is created on Step 2 and Step 3. The

development of the Visual Interaction Layer is completed after applying Step 4, Step 5 and Step 6. In case a system designer wishes to enable programming agents through implicit training mechanisms Step 7 should also be completed. Next we present a detailed overview of each of these steps.

Step 1. Eliciting Specification Requirements. The initial step of the methodology is the analysis of the application domain by system architects with the goal to elicit specification requirements. This step should result in the creation of the Software Requirements Document. In this document the key activities, roles of the participants and basic scenarios are outlined. The suggested methods for eliciting system requirements are interviews, questionnaires as well as other means of applied exploratory and descriptive research. Chapter 6 (Section 6.2.2) presents an example of how this step of the methodology could be performed. Once the specification requirements are established, Step 2 should be used to formalize them.

Step 2. Specification. This step is conceptually the same as specifying the institution using the Electronic Institutions methodology [7] and should also be executed by the system architects. It establishes the regulations that govern the behavior of the participants. This process is supported by ISLANDER [68] which permits to specify most of the components graphically, hiding the details of the formal specification language and making the specification task transparent. As it was described in Chapter 2 the specification is determined by the three types of conventions:

- Conventions on language, the *Dialogical Framework*. Here the roles the participants are allowed to play, the relationships amongst these roles and common language are specified.
- Conventions on activities, the *Performative Structure*. This dimension determines in which types of dialogues the participants can be engaged. First the key activities (scenes) are identified. Next the role flow between different scenes is set. Each of the scenes is further associated with an interaction protocol, where a protocol is defined as a final state machine.
- Conventions on behavior, the *Norms*. Norms determine which consequences the participants' actions in different scenes have. These consequences are regarded as commitments acquired while acting in the environments and have to be fulfilled later on. These commitments restrict future activities of the participants. They may limit the possible scenes that can be entered and possible actions that can henceforth be executed.

- Interaction protocols, the *Scenes*. Scenes define conversation protocols for a group of roles. Each scene requires a definition of its participating roles and their population, its conversation protocol, and the states at which the participants can either leave or enter the conversation. The scenes of an institution define the valid interactions that the participants are allowed to have and set the context wherein the exchanged illocutions must be interpreted.

If during formalization it became evident that the system requirements elicited on step 1 are insufficient or if for some other reason their evolution is required, the system architect has a choice to return to step 1 for refining the system requirements. If no requirements evolution is necessary and the specification of the system is complete, its validity should be ensured on Step 3.

Step 3. Verification. One of the advantages of the formal nature of the methodology is that the specification produced on the previous step can be automatically verified for correctness by ISLANDER. The tool verifies the scene protocols, the role flow among different scenes and the correctness of norms. This verification is static in nature, meaning that the specification has to be finalized before the verification can take place (in contrast to trying to verify on the fly during the specification process). The verification starts with the validation of the correctness of the protocol defined by each scene. This includes checking that the state graph of each scene is connected, that each state is reachable from the initial state and that there is a path from each state to a final state. It is also ensured that the messages associated to the arcs of the state graph are correct with respect to the Dialogical Framework.

The Performative Structure establishes how the participants can legally move among the different scenes. As we do not want them to get blocked in any scene or transition it is verified that from each scene and transition the participants always have a path to follow, and that from any scene or transition there is a path to the final scene (so that all participants can leave the institution).

Finally, ISLANDER checks whether the norms are correctly specified and whether the participants can fulfill their commitments. As commitments are expressed in terms of actions that have to be carried out in the future, it is verified that those actions can be performed. For instance, if there is a norm that defines that a participant has to pay for acquired products, it is checked that from the scene where the products are purchased there is a path that allows reaching the payment scene.

The verification permits to detect errors in the institutional rules before starting the design and development of the 3D visualization. If such errors are found, the developers should go back to step 2 to correct those. If the specification contains no errors there

are two options how to proceed. If the 3D Visualization of the environment is already created (reuse of the existing design) then the developers may skip the next two steps and continue with Step 6. Otherwise, the generation step (Step 4) should be executed.

Step 4. Automatic Generation. On the generation step the 3D Virtual World and its floor plan are created in a fully automatic way. Not only the institutional specification defines the rules of the interactions, but also helps to understand which visualization facilities are required for participants to operate in the institution. Some elements of the specification have conceptual similarities with building blocks in 3D Virtual Worlds, which makes it possible to create an automatic mapping between those. On this step of the methodology the Electronic Institution Specification is automatically transformed into the skeleton of the Virtual World.

The scenes and transitions are transformed into 3D rooms, connections become doors, and the number of participants allowed to be in a scene determines the size of each room. The generation can function in two different modes: Euclidean and non-Euclidean. In the first case the rooms on the generated floor plan are positioned so that each room and transition connected in the Performative Structure graph by an arc are physically placed next to each other (connected via a door). In the non-Euclidean case the rooms may be located anywhere and are not necessarily involved in any sort of spatial relationship. The movement between connected rooms in the non-Euclidean Virtual World will then be conducted using teleportation.

Next, all the rooms are resized to be able to include the maximum number of participants allowed in the corresponding scene. Another outcome of this step is the schematic plan (map) of the institution.

After finishing this step the generated visualization has to be annotated on Step 5.

Step 5. Annotation. The Electronic Institutions metaphor is quite generic and can be applied for solving a wide range of problems. The Electronic Institutions Specification was not considered to be visualized in 3D Virtual Worlds. It defines the rules of the interaction and has nothing to say about the appearance of the specified elements. Therefore, a fully immersive and visually rich Virtual World can not be automatically created just on the basis of this specification. In order to make the generated skeleton appealing it has to be enriched with additional graphical elements on the annotation step. These additional elements include textures and 3D Objects like plants, furniture elements etc.

Apart from the elements defined in the Performative Structure most of the components of the Electronic Institutions specification do not require visualization. Such components are ignored on the generation step. In some situations, however, the visualization

of these components is useful. For example, data types defined in the ontology can be represented by 3-dimensional objects. If this is the case, on the annotation step a system designer should create 3D models for these objects and manually insert them into the corresponding rooms.

One of the central ideas behind the Virtual Institutions methodology is to facilitate the development of 3D environments so that most of the tasks can be achieved automatically or semi-automatically. The absence of the visualization related information inside the Electronic Institutions specification decreases the degree of automation that can be achieved. Therefore, certain trade offs between full automation and the variety of possible visualizations should be made depending on a particular situation.

Full automation of the annotation step can only be achieved under the condition that the target domain for the application of Virtual Institutions technology is very limited. For example, if the methodology is considered to be applied only to the development of virtual shopping malls, only then it is suggested to automate the annotation step. Such automation can be achieved by limiting the range of possible designs and employing a technology (i.e. design grammars [86]) for selection of an appropriate design for a particular specification element).

If the Virtual Institutions methodology is required to be applied for a very wide range of problems – fully automatic generation is not feasible. In such a case the annotation should be done manually. Further we show how this manual annotation can still be done quite efficiently with the help of tools like AtmoKits¹.

This step of the methodology does not usually require the involvement of the system architects and should rather be executed by software developers. After this step the user can return to Steps 1 and 2 to refine the specification requirements or the specification itself or can continue with Step 6.

Step 6. Integration. On the integration step the execution state related components are specified. This includes the creation of the set of scripts that control the modification of the states of the 3D Virtual Worlds and mapping of those scripts to the messages, which change the state of the Electronic Institution. Firstly, the scripts that correspond to the messages from the agent/institution protocol need to be defined. These include entering or leaving a scene or transition, entering or leaving an institution, etc. Next, the scripts that correspond to the specific messages defined in the ontology on the specification step must be created. As most of these scripts are required to be present in every Virtual Institution the standard supply of them is provided for every visualization platform. However, in some specific situations these scripts may have to be modified, which

¹<http://www.atmokits.com>

should be done on the integration step.

In case there were any 3D objects representing the ontology data types, the actions upon which require validation – the mapping between these objects and the corresponding data types in the ontology has to be established. At the end, the correspondences between the messages and scripts (actions) are created by filling in the Action/Message table. In case the previous step of the methodology was automatically completed through the application of design grammars, these objects may have already been introduced and there is no further integration necessary. The Action/Message Table in this case should be automatically filled.

Making the integration a separate step of the methodology stimulates the development of the scripts in the form of design patterns, that are generic enough to be reused in other systems.

After accomplishing this step the generated 3D Virtual World is ready to be visualized and the Virtual Institutions infrastructure will be executed to take care of the validity of interactions between participants, verify the permissions of participants to access different scenes and make sure that all the institutional norms and obligations are imposed. This step is particularly important for the case when the system requires to use an already existing design. For existing designs integration can not happen automatically and manual integration is the only possible way to enforce the technological connection between the Visual Interaction and Normative Control layers.

Similar to the previous step, integration should be conducted by software developers. After this step the user can return to Steps 1 and 2 to refine the specification requirements or the specification itself or can continue with Step 7.

Step 7. Enabling Implicit Training. Virtual Institutions provide unique facilities for development of autonomous agents. We propose the method, called Implicit Training, which is suggested to be one of the central technologies behind the decision making of the autonomous agents in Virtual Institutions. With the help of Implicit Training the agents can learn sophisticated human-like behaviors from observation of human actions in the 3D environment. Following this approach in many cases it becomes much more efficient to train the autonomous agents than to program them. For some application domains like E-Commerce, where the demand for assistant agents is quite high, Implicit Training is particularly important. It is suggested that the assistant agents for such applications would constantly observe the behavior of human assistants and learn from them how to answer the customer enquiries (without any explicit training efforts required from the humans). In contrast to programming, such approach is much less resource consuming and a lot more flexible.

In case of Implicit Training the Normative Layer of Virtual Institutions forms the basis for the decision tree of the agent, where possible illocutions become the nodes of this tree. For each of those nodes it is possible to specify whether implicit training is conducted or not. This process is completed on the Enabling Implicit Training step of the methodology. The remaining details of the implicit training are presented in Chapter 5.

4.2 Technological Solution

The methodology presented above should be used for generating the technological infrastructure of Virtual Institutions. The specification produced on Step 2 and verified on Step 3 forms the Normative Control Layer. The technology for achieving those steps is already available. The ISLANDER [68] tool supplied with Electronic Institutions Development Environment [7] is a graphical editor used for creating the specification and verifying it for correctness. The completed specification can be stored in an .xml file. XML format is generic enough and there are many parsers available for it. Therefore, for Step 2 and Step 3 of the Virtual Institutions Methodology we suggest utilizing ISLANDER without any modifications.

The steps of the methodology that correspond to the development of the Visual Interaction Layer as well as enabling implicit training are not supported with existing tools. Further in this section we propose a technological solution for supporting Steps 4, 5, 6 of the methodology and the details regarding the Step 7 are given in Chapter 5.

Before continuing with the presentation we find it necessary to discuss some limitations of the implementation. The Virtual Institutions Methodology assumes that an Electronic Institutions specification covers all the aspects of the Normative Control Layer. In particular, it is assumed that one specification may include the formalization of a number of institutions. Unfortunately, the current version of ISLANDER is unable to define multiple institutions inside one specification (although, the developers are currently working on the implementation of this feature). Because of this, the technology we propose here only applies to the development of one institutional building.

Another remark we would like to make is that creating a fully functional framework covering all the steps of the Virtual Institutions Methodology was beyond the scope of our research. Instead, we only focused on creating a set of working prototypes that are capable of demonstrating how each of the steps could be facilitated through various existing technologies and provide a proof of concept. We also tried, where possible, to reuse existing solutions and collaborate with other research groups to share their expertise. Therefore, some of the tools described here were adapted from (or fully created by) other collaborators.

Ultimately, we aim at providing a technology independent technological solution, however, for the purpose of quick prototyping we had to select a particular technology to be used for the visualization of 3D Virtual Worlds. The technology chosen for this task was Adobe Atmosphere visualization platform. In particular, the Trading Institution prototype presented throughout the thesis was implemented using this software. At the time of selection Adobe Atmosphere was still supported and it was chosen for quick prototyping purposes based on the comparison presented in Table 2.1. Its extraordinary rendering capabilities, comprehensive object builder and the possibility to import from major 3D formats (for the case if existing design is required to be reused) have determined our choice. No additional software is required to be explicitly installed for using Adobe Atmosphere; the necessary plug-in will be installed automatically once the user opens the corresponding page with the web browser. We also required a high degree of control, so that the rules of the institution can be strictly enforced and Adobe Atmosphere was capable of providing such control.

To show that the approach we took is general enough and to a high degree independent of a particular visualization platform we also implemented some of the prototypes using other technologies.

Next we outline our technological solution that aims at facilitating efficient application of Steps 4–6 of the methodology.

4.2.1 Step 4. Automatic Generation of Virtual Institutions

To make the visualization of Virtual Institutions more efficient we propose to transform the Performative Structure graph into the corresponding 3D Virtual World in a fully automatic way. There are two possible solutions for achieving this automation. The first approach is the non-Euclidean visualization, where the resulting 3D Virtual World does not have an Euclidean representation. This means that the rooms present in the Virtual World are not involved into any kind of spatial relationship with each other and are independent objects, loaded on demand when a user needs to enter them (through a door). Second approach is to make the visualization fully Euclidean, meaning that whole institution is located within some virtual space, which is abide by the Euclidean laws. The connected rooms in this case are physically located next to each other and separated by doors inside this space.

Non-Euclidean Generation

Automatic generation of a Virtual Institution using the non-Euclidean approach is technically simpler. The institutional building and each of the rooms are represented as 3D



Figure 4.2: Example of a Room generated using World Generator.

models each stored in a separate file. The associations between these files are created with the help of the door objects. Colliding or clicking on such a door will result in uploading the 3D model of the room the user is trying to enter into and positioning the user inside this room.

The non-Euclidean automatic generation method was implemented and tested in the lab by Simon Biber. The “World Generator” software he produced takes an .xml description of an Electronic Institutions specification and transforms each of the scenes and transitions, present there, into 3-dimensional room objects that can be further used by the Half Life 2 game engine². Figure 4.2 outlines the Trading Room of the Trading Institution generated using this mechanism.

In the generated Virtual Institution the movement between rooms can only be achieved by teleports³. Each room is constrained by the walls from every side and to leave it a user is required to collide with one of the doors (teleports) present there.

As the notion of distance and size is not relevant for the non-Euclidean spaces the creation of a schematic plan of the Virtual World is quite a difficult task. However, without such a plan it will be very hard for participants to navigate the Virtual World. Therefore, the Performative Structure of the underlying Electronic Institution is used as the plan (map) of the institution.

²<http://half-life2.com>

³3-dimensional objects that on collision force teleportation to a given position inside the Virtual World.

Even with the Performative Structure acting as the map we found it quite problematic to navigate a non-Euclidean Virtual World. Next we present some research that supports our concerns.

Importance of Euclidian Representation

Navigation is an important issue in the design of 3D Virtual Worlds [55, 90, 207, 26]. Navigational problems may break the immersive experience and lead to the rejection of the system by end users [39, 43]. Humans live in an Euclidian world: distances and angles help humans to navigate it. Some researchers [26] hypothesize that the better a Virtual World imitates the real world, the more potential it has to offer consistent experience and support social factors like communication and collaboration.

3D Virtual Worlds could certainly be programmed without an Euclidian model in mind. For instance, we could create a series of rooms interconnected by teleportation. In this case the distance between each two connected rooms will be equal to zero. Technically, such a solution poses much less problems for development, as there is no need to control the positioning of every room while generating the building layout. Instead, each of the rooms can simply be uploaded on demand from a separate file.

Despite the technological simplicity of the non-Euclidean way of representing an institution as a Virtual World, such an approach may cause navigational problems for the participants inside. Moreover, having no Euclidian representation of the Virtual Institution may have negative effect on learning of the institutional structure by the participants.

To demonstrate the confusion a participant may experience navigating a non-Euclidean Virtual World consider the situation (Figure 4.3) where a participant walks from *Room 1* to *Room 4* following the arrows. In *Room 4* the human should correctly expect that *Room 1* is behind *Door 4*, otherwise the believability of the interface may be lost. This expectation of *Room 1* is based on both the consistency of the navigational layout and intuitive feeling about the size of the visited rooms. In a non-Euclidean case the human can not expect that after entering *Door 4* he/she will appear inside *Room 1*.

Although, the problems of Euclidean and non-Euclidean visualizations of Virtual Worlds are understudied, there is some research evidence in favor of our initial hypothesis that motion techniques which instantly teleport users to new locations are correlated with increased user disorientation. In [26] the authors present the results of their user study, where one of the questions was whether teleportation can cause navigational problems. The study clearly shows that teleportation (or jumping techniques, as it is called in the paper) can reduce the user's spatial awareness. With teleportation there is no sensation of motion, only that the world has somehow changed around the user. It is a technique, which has no analog in the physical world. Moreover, authors came to a

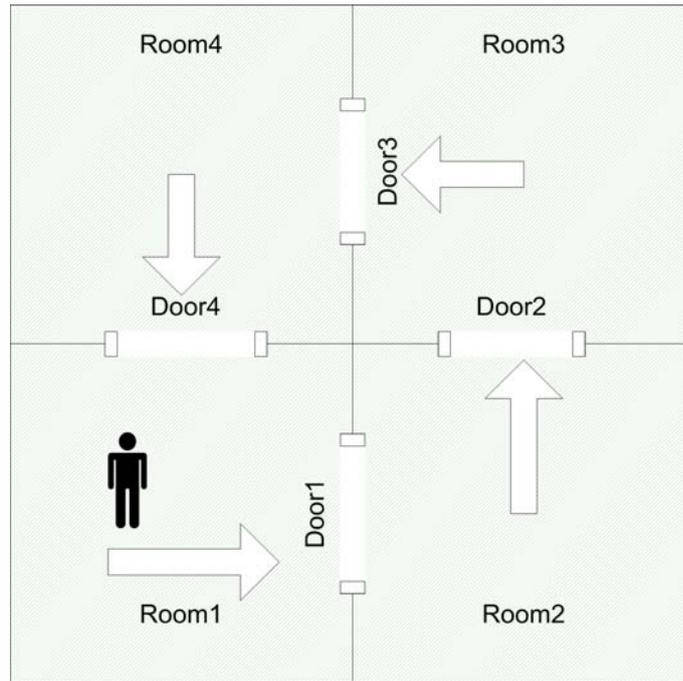


Figure 4.3: Euclidian representation. Human knows: Room 1 must be behind Door 4.

conclusion that frequent teleportation may even reduce the sense of presence in a Virtual World, which would eliminate one of the most important benefits of the 3D technology.

Another research [174] provides some support in favor of the assumption behind the example presented on Figure 4.3. This user study analyzed the navigational efficiency of participants in the overlapping Virtual Worlds. The results suggest that some users had great difficulties navigating them. Even the experimenter, who informally observed participants as they travelled through the Virtual Worlds, was often unsure of which door to go through to enter a particular room.

Additionally to the above presented evidence in favor of fully Euclidean Virtual Worlds there is another important issue. Many popular Virtual Worlds platforms simply do not allow for non-Euclidean models. Engines like Second Life make profit from selling virtual land, the price of which depends on the physical area of the purchased space. Uploading of additional models on demand from separate files is not supported there. It is not in the interest of the platform developers to facilitate purchasing a minimal space required to include the 3D Model of the building and then infinitely extend this space by uploading the rooms on demand. Due to the growing popularity of Second Life and alike we can not ignore the issue of automatic generation of Euclidean spaces. Furthermore, inside an environment like Second Life it is highly desirable to minimize the space occupied by the institutional building to be able to minimize the price of the land purchases.

Euclidean Generation

One of the existing approaches that can be used to achieve automatic transformation of the Performative Structure graph of the existing Electronic Institution Specification into a space optimal building inside a Virtual World is called “rectangular dualization”.

Rectangular dualization method is successfully employed by Massimo Anacona and Sara Drago from the University of Genoa, Italy [60] to achieve space optimal transformation of graphs into Euclidean maps. Through our collaboration with Massimo’s group their OCoRD tool that is used for generation of rectangular duals of the input graphs was further extended with the possibility to parse an .xml specification of an Electronic Institution and use it to create a graph that satisfies the conditions expressed by their method. As an output of OCoRD a 2-dimensional map of the institution is produced. This map can then be further used by the World Generator software to generate the 3-dimensional representation of the institutional building.

Although most of the work presented further in this section was conducted by Sara Drago and Massimo Anacona [60], we find it important to present the details of the rectangular dualization method here. The presentation is based on [60].

Rectangular Dualization

Rectangular Dualization was originally intended for generating rectangular topologies for floor planning of integrated circuits: by a floor plan, we partition a rectangular chip area into rectilinear polygons corresponding to the relative location of functional entities of the circuit [94]. In spite of the specialized problems that motivated its origin, rectangular dualization contributes to the resolution of many other visualization problems having in common with circuits the condition that objects and their interoccurring relations are represented by means of a planar graph. An example is given by network configuration issues, when human interventions of design or topology adjustment are needed and a physical or logical layout representation becomes essential for the human operator.

In fact, a very serious problem to cope with in graph drawing is how to represent edges in such a way that they do not appear too close together. The aim is to enhance the readability of the drawing, making easier to find out which nodes are connected by an edge. The very first solution to this problem is to avoid edge crossings, and this motivates the interest for *planar graphs*, that are precisely those graphs that can be drawn in the plane with no edge crossings. The choice for planar graphs is not only a representation facility but is primarily validated by real world examples where the presence of crossing links may produce technical drawbacks.

Further on, since a major optical effort is encountered in the proximity of vertices, where adjacent edges need to meet in a point, several studies have been focused on

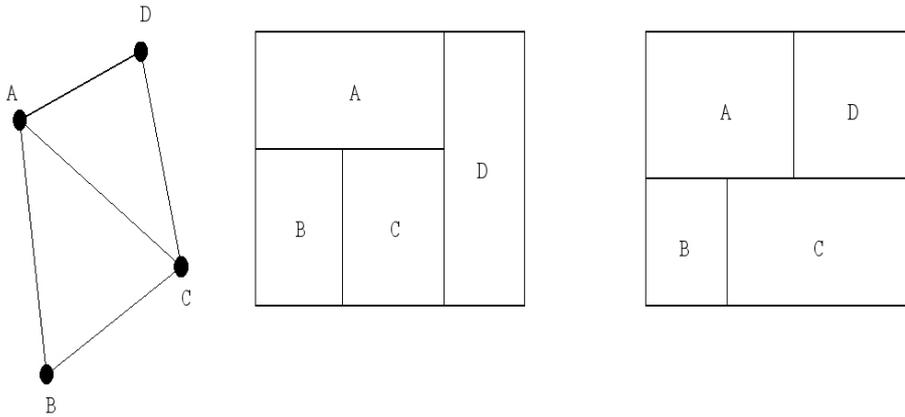


Figure 4.4: A planar triangulated graph and two possible rectangular layouts.

devising drawing algorithms capable of maximizing *angular resolution*, i.e. the smallest angle between adjacent edges, in such a way that lines representing connections are kept as separate as possible; rectangular dualization results to be an effective visualization method since only orthogonal lines occur.

Definitions

A *rectangular dual* of a planar graph $G = (V, E)$ is a rectangle R with a partition of R into a set $\Gamma = R_1, \dots, R_n$ of non overlapping rectangles such that:

- no four rectangles meet at the same point;
- there is a one-to-one correspondence $f : V \longrightarrow \Gamma$ such that two vertices u and v are adjacent in G if and only if their corresponding rectangles $f(u)$ and $f(v)$ share a common boundary.

It is easy to see that if a graph admits a rectangular dual, it may not be unique (see Figure 4.4).

On the other hand, some graphs do not admit rectangular dual. Kozminski and Kinnen present necessary and sufficient conditions under which a plane graph G has a rectangular dual [115]: the most important point is the absence of *separating triangles* (i.e. 3-vertex cycles with at least one vertex in their interior), a condition that in planar triangulations is equivalent to *4-connectivity* whose meaning is that the removal of any set of 3 vertices leaves the remainder of G connected. A *matching* in G is a subset M of edges such that for every vertex v , at most one edge e covers v , that is v is an endvertex of e . A graph is *k-regular* if every vertex has degree k , that is k incident edges. A *maximum matching* is a matching with largest possible cardinality. If the graph is weighted, we

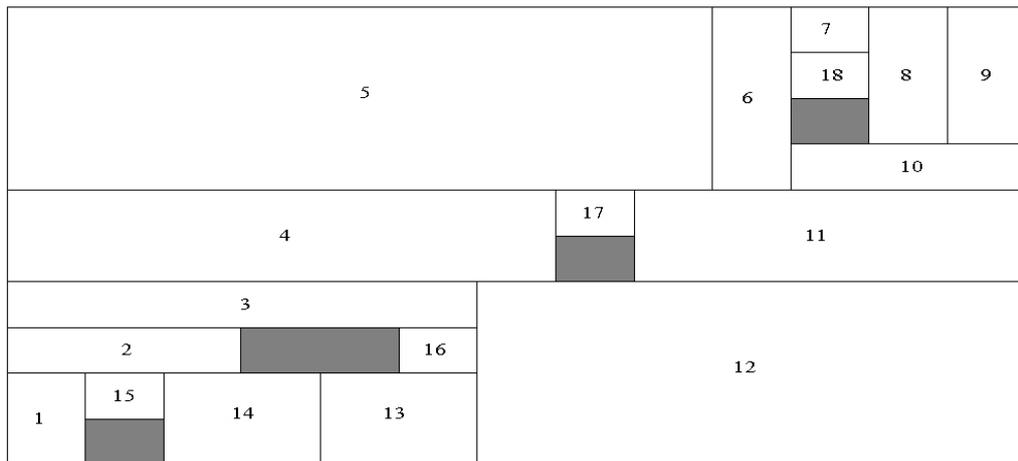


Figure 4.5: A rectangular dual representation computed by OCoRD. Shaded rectangles are due to breaking points.

may even consider a *maximum weight matching*. A *bridge* is an edge whose removal disconnects G . Whenever we speak of a planar graph, we assume that some planar embedding has been fixed, which corresponds to the idea of depicting an existent physical connection among real objects (in this perspective, it would be more accurate to speak of *plane graphs*, i.e. planar graphs with a fixed embedding in the plane). A *structured graph* is a form of abstraction applied to a large graph in order to make it modular and more manageable. The abstraction consists in collapsing a subgraph to a single vertex (called a *macrovertex*), or to a single link (called a *macrolink*) thus obtaining a simpler and hierarchically described graph. The structuring operation is usually iterated recursively until a large graph is decomposed into relatively small and manageable components and sub-components defined at several levels of nesting (see Figure 4.5) adopting a methodology that is usually applied to every large project (software and hardware design) involving hundreds or thousands of components: i.e. “modularity”.

OCoRD Software

The OCoRD software is a tool aiming at the solution of floorplanning problems and at the orthogonal drawing of planar networks. Given a planar embedding of a graph, it accepts a numeric adjacency lists and, if the graph admits a rectangular dual, it returns its coordinates in the plane and a drawing in *fig* format. Moreover, OCoRD transforms a graph not admitting a rectangular dual into a 4-connected supergraph satisfying Kozminski and Kinnen criterion. In fact, it is possible to eliminate separating triangles by adding crossover vertices on an edge of each separating triangle [119]. The implemented breaking method is optimal, in other words it only adds a minimum number of such crossover

vertices and inserts them in strategic positions, such that a single vertex may break two triangles or more. The method, described in [3], performs this transformation in the following steps:

1. four external vertices are added according to the construction presented in [110];
2. the geometrical dual of the graph is computed and faces belonging to a separating triangle are detected and clustered together in a single macrovertex;
3. a covering affecting macrovertices is computed; the effect is that all separating triangles are optimally broken by inserting new vertices in some strategic places along some of their constituting edges;
4. the resulting graph is triangulated with the algorithm described in [21].

We have so far obtained a graph satisfying triangularity and four-connectivity. Faces belonging to a separating triangle are detected in linear time [40]. Separating triangles are broken by solving a minimum unweighted macro-covering problem on the geometrical dual of the input graph. The macro-covering can be computed by solving a sequence of minimum *weighted* edge covering problems on each simple graph of the structured dual. In [154], Parekh showed how to reduce a minimum weighted edge cover of a specified subset of the vertices of G to a maximum weighted b -matching, a well solved problem [63] that is worked out by implementing the $O(mn \lg n)$ algorithm presented in [73]. The input graph is biconnected and it is described through adjacency lists satisfying the following properties (see Figure 4.6):

```

16  _____
5  2 3 6 16 12
7  7 1 15 10 6 9 8
3  1 7 4
3  3 5 6
2  4 6
7  1 4 5 7 2 10 16
4  3 2 9 6
2  2 9
3  2 7 8
5  2 15 11 16 6
4  10 13 12 16
3  1 11 13
4  11 15 14 12
2  13 15
4  2 14 13 10
4  1 6 10 11

```

adjacency lists

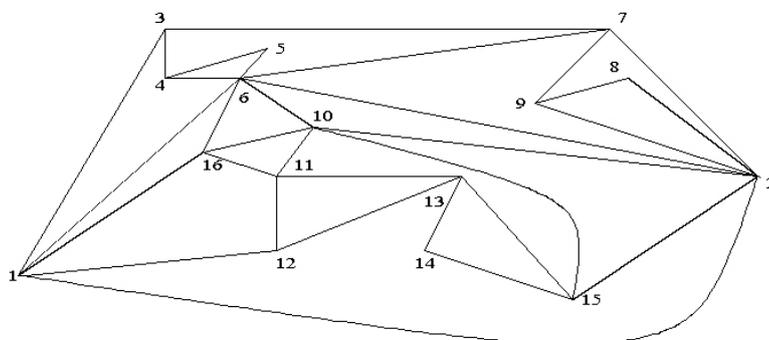


Figure 4.6: A planar embedding of a graph with its input file.

- vertices are indexed by non negative integers $(1, \dots, n)$;
- for each internal vertex, its adjacencies are clockwise listed, according to a fixed planar embedding of the graph, starting from the vertex having the lowest index;
- for each vertex belonging to the graph contour, its adjacencies are clockwise listed, starting from the vertex on the contour which precedes it in a counterclockwise direction with respect to the contour.

Algorithm: Rectangular Dual Construction

Input Planar biconnected graph G

Output A rectangular dual of G

Add four external vertices

Compute the geometrical dual graph G^*

Detect faces belonging to a separating triangle, for any

Collapse them into a macrovertex in G^*

Solve the macro-covering problem in G^*

Add new vertices in G

Triangulate G [21]

Compute a REL

Compute the rectangular dual coordinates

Delete external vertices and draw

Figure 4.7: Algorithm: Rectangular Dual Construction

We note that in the modification perspective, also input graphs that are not under the biconnection constraint may be processed: a preliminary step can be implemented to provide this degree of connectivity [170]. Figure 4.7 gives an overview of the implemented rectangular dual construction method. As a refinement of the described method, we may say that the logarithmic cost of the method is due to the fact that the aforementioned matching algorithm holds for general graphs, a much wider class of graphs than the one we deal with in our planarity assumption. Instead, a matching in a 3-regular bridgeless graph can be found in linear time [20]. Since the collection of all planar 3-regular bridgeless graphs is exactly the collection of duals of planar triangulations where the outside face is a triangle, we may tighten the bound by solving the matching problem on the dual of a planar triangulation and this can be obtained by producing a triangular outer boundary and by running the algorithm [21] (which triangulates without adding new separating triangles) before the computation of the geometrical dual.

Adaptation of OCoRD to Virtual Institutions

The algorithm presented in Figure 4.7 as well as the OCoRD software had to be further adapted for the visualization of Virtual Institutions.

From the Performative Structure graph of the institution both the 3D representation of the institutional building and the map of the institution are created. The process of automatic generation of the layout of the institutional building as well as the corresponding map from this graph is depicted in Figure 4.8.

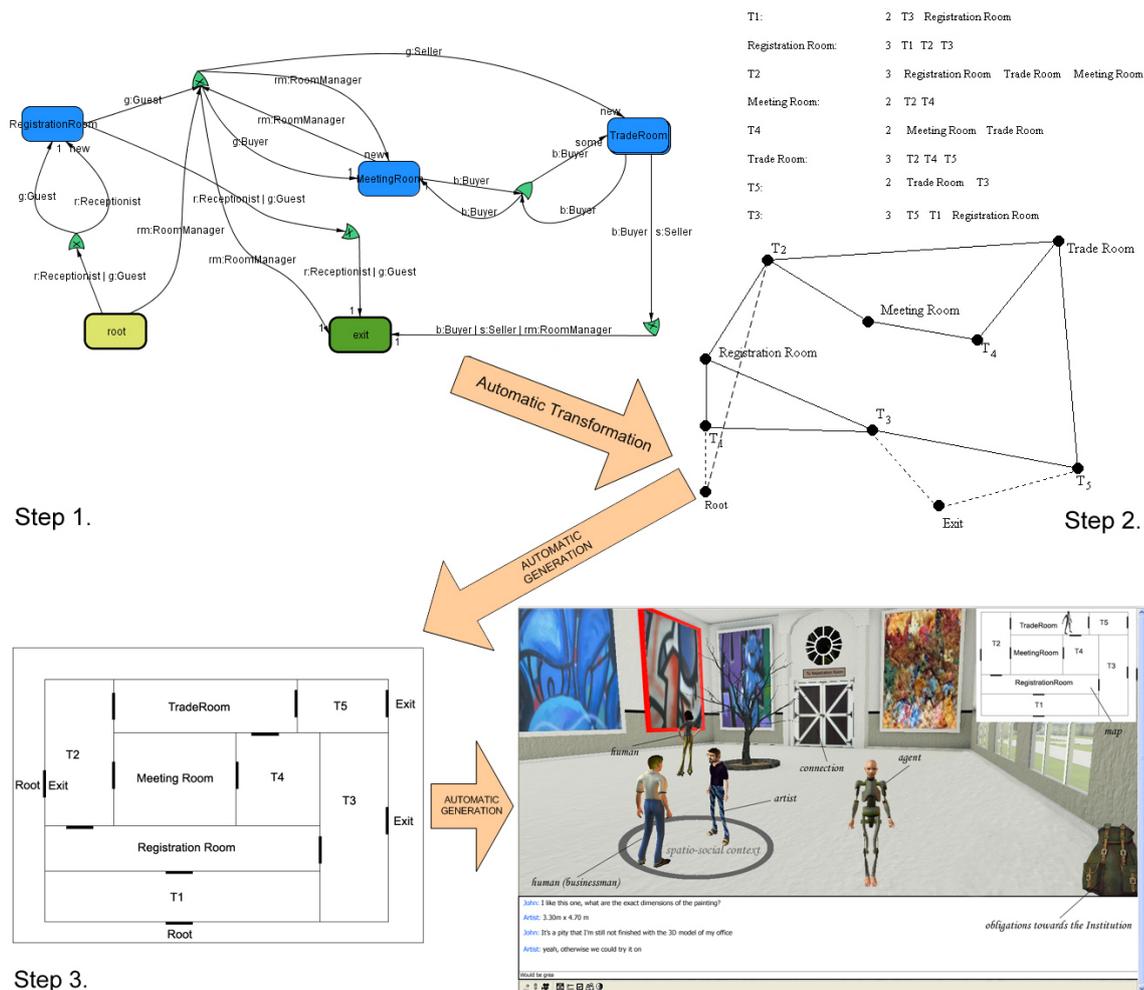


Figure 4.8: Generating the 3D representation of a Performative Structure graph.

In the upper left corner in Figure 4.8 the source Performative Structure graph is presented. This graph corresponds to the Trading Institution explained in Section 3.2. Rectangular shapes represent scenes, triangular shapes are transitions and directed arrows (arcs connecting nodes) are connections.

The automatic generation is done in 3 steps. On the first step the Performative Structure graph is transformed into the format understandable by OCoRD software and the

unnecessary information is filtered out. If some nodes of the graph are connected with more than one arc, only one randomly selected arc is left and all the others are deleted.

After the transformation is completed the step 2 is executed, on which the OCoRD software reads the transformed graph and creates its rectangular dual. The rectangles, which were introduced because of the breaking points, are removed and the adjacent rooms are reshaped to the size of removed rectangles. The Root and Exit scenes are always present in the performative structure graph. Moreover, the root scene is not permitted to have incoming arcs and the exit scene doesn't have any outgoing arcs. As those scenes are not visualized, the corresponding graph nodes are ideal candidates to be two of the four external vertices. In this way the garden will be automatically created as the rectangle surrounding the graph's rectangular dual. When the rectangular dual is produced, the only thing that is left is placing the doors between connected rooms. The outcome of this step is the map of the institution, which is presented in the lower left corner of Figure 4.8.

Step 3, transforming a 2D map of the institution into a 3D Virtual World, is the task of the World Generator software. The coordinates of the map are first transformed into the coordinates of the 3D Virtual World. Then, every room is scaled so that it can physically contain the maximum number of participants that is defined for it. Later on, the corresponding 3D objects are reshaped and put inside the 3D Virtual World. The result of this step as applied to the Trading Institution will look similar to the bottom right part of the Figure 4.8.

4.2.2 Step 5. Annotation

In order to make the generated institutional building appealing, it has to be annotated with additional visual elements. There are two possibilities in conducting this annotation: fully automatic annotation and semi-automatic annotation. As it was mentioned before, if a system architect plans to employ the Virtual Institutions methodology in a limited domain, it is possible to have the annotation happening automatically through the utilization of design grammars [173].

Automatic Annotation

The essence of this design grammars approach is to capture the style of the facade and rooms inside the institutional building by so-called design rules. Design rules illustrate and describe the forms and functions of each design element. These rules are represented by different combinations of the basic primitives existing in the 3D Virtual Worlds.

One of the most popular types of design grammars are shape grammars [200]. Shape

grammars are a specific class of production systems that generate geometric shapes. A shape grammar consists of shape rules and a generation engine that selects and processes them. A shape rule defines how an existing shape can be transformed. It consists of two parts separated by an arrow pointing from left to right (i.e. $A \rightarrow A + B$). The left hand side (LHS) of the rule depicts a condition for room execution in terms of a shape. The right hand side (RHS) determines how the LHS shape should be transformed and outlines the result of the room execution [200, 173].

To give an impression of what shape rules look like, Figure 4.9 presents a simplified version of the rule for positioning a table inside a room.

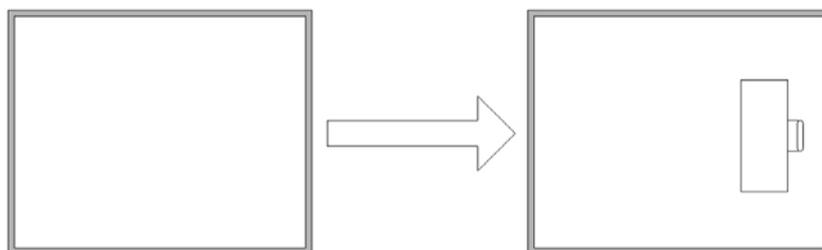


Figure 4.9: An Example of a Shape Rule.

Shape grammars can be successfully utilized to generate suitable 3D designs for the institutional rooms. They provide a linkage that translates the “specification language” of the institution into the “design language” of 3D Virtual Worlds.

The shape grammar approach has a number of advantages. Firstly, the changes of the specification and of the runtime environment can be automatically addressed through the grammar application (by alternating the choices and order of the design rules during the application). Secondly, the use of the shape rules makes the system being platform independent. Using a different 3D Virtual World platform for design implementation will only require a simple re-mapping between the elements of the database and basic primitives of the Virtual World, without undergoing any major system changes.

A successful attempt to use shape grammars for enhancing the appearance of Virtual Institutions was made by Owen Macindoe and Ning Gu from the University of Sydney [87]. Figure 4.10 outlines the visualization of the Trading Institution they created following the shape grammar approach [86]. The resulting design is rendered by the Second Life visualization platform. Each of the rectangular spaces connected to the poll on this picture represents one of the scenes in the Trading Institution.



Figure 4.10: Trading Institution Generated and Annotated Using Shape Grammars.

Semi-automatic Annotation

Automatic designs are, obviously, not always suitable for developers. There are situations when a particular design solution is required for a particular institution. In such a situation each (or some) of the rooms generated on the Automatic Generation step of the methodology will be originally empty and the design of the rooms will have to be enriched manually (annotation phase).

Even for this kind of situations there are tools available on the market that could significantly improve the speed of the annotation. One of such tools is AtmoKits software. This software is compatible with the Adobe Atmosphere platform that was used for the Trading Institution prototype developed for this thesis. It is capable of uploading the model of the Virtual World (produced on the generation step), enriching it with additional elements and saving the result in Adobe Atmosphere compatible format.

AtmoKits is supplied with the predefined sets of objects, but also supports including new objects into existing sets. The most popular standard set of objects is AtmoKits Household Kit. It can be successfully utilized to annotate the majority of commercially oriented institutions. The set contains textures, furniture pieces, plants and household objects. Figure 4.11 gives a rough idea about how the designs of the rooms can be changed via Atmokits and illustrates some of the objects available in the Household Kit.

The upper left part of the picture shows the currently selected set of objects (Household Kit) and supports the navigation through different types of objects available there. Further below the schematic plan of the uploaded institution is outlined. The bottom



Figure 4.11: Room Design Using Atmokits.

part of the interface is used for selecting an appropriate object from the list, moving it, rotating and scaling. The movement and rotation of the objects is done by clicking the corresponding arrow buttons. The status line at the very bottom of the window shows the detailed information about the currently selected object. The “Remove Model” and “Lock Model in Place” buttons support erasing the current object from the room or locking it in the current position. The right hand side of the interface shows the created design in real time. All the objects transformations and position changes are immediately reflected onto the interface. The system designer is represented as an avatar with default appearance. It is possible to fully explore every room and move between different rooms by changing the position of this avatar.

Another very useful feature of AtmoKits software is the possibility of collaborative room annotation. Each of the designers in this case is visualized as an avatar, so that a number of designers can work on the design simultaneously, observe the design evolution in real time and communicate with each other through their avatars.

4.2.3 Step 6. Integration

The Virtual Institutions model proposed in this thesis assumes that there is a tight connection between the Visual Interaction Layer and the Normative Control Layer. To make this connection happen every action of the Visual Interaction Layer that requires verifi-

cation has to be validated by the Normative Control Layer. Most of the 3D visualization platforms (including Adobe Atmosphere that is employed in our prototypes) support representing these actions in terms of scripts. In case of Adobe Atmosphere these scripts are blocks of text structured following the Java Script notation⁴. Such a view permits to express every action of the Virtual World as a text block uniquely identifiable by its name. Adobe Atmosphere offers straightforward mechanisms to execute such a script if its name is provided.

An efficient way to link the two layers is to create a mapping between the actions (scripts) that change the state of the Visual Interaction Layer and the actions (illocutions) that change the state of the Normative Control Layer. In the current implementation this mapping is achieved by employing the Action/Message table, where each script from the Visual Interaction Layer is assigned with a corresponding message from the Normative Control Layer (where a message can be either a part of the illocution sent by the agent to the institution or a part of the institutional response).

The actual methodology employed for conducting this step may be different and depends on the visualization platform being used. Therefore, it is not the goal of this dissertation to provide specific technical details as to executing this step. As a general recommendation we suggest employing the Action/Message table approach described above.

Table 4.1 presents a simplified fragment of the Action/Message table for the Trading Institution. The table specifies that “addNewAvatar” script should be executed as the result of receiving the “EnteredAgentInstitution” message from the institutional infrastructure, which corresponds to entering the institutional building. The “leftSceneRoom” script will be launched before the institution notifies the agent that it has successfully exited a scene. The “enteredSceneRoom” script will be started when the institution updates its state letting the agent to move inside a scene, etc.

Table 4.1: Action/Message Table for Trading Institution.

Action	Message
addNewAvatar	EnteredAgentInstitution
leftSceneRoom	ExitedScene
enteredSceneRoom	MovedToScenes
leftTransitionRoom	ExitedTransition
enteredTransitionRoom	MovedToTransition
raiseHand	Bid
removeAvatar	ExitedAgentInstitution

⁴<http://www.javascript.com>

All of the institutional messages shown in the table represent the responses of the institution to the agent requesting to perform an action. For example, the “Moved-ToScenes” message will be sent as the result of the institutional infrastructure receiving an “EnterScene” message from the agent. This message will be sent back to the agent by the institution after the “EnterScene” message is validated and accepted by the institution, and the state of the institution is updated to reflect the fact that the agent entered the desired scene.

At the current stage of the implementation the script creation and population of the Action/Message Table with data has to be done manually by a system designer (although some standard script templates are already available). However, we plan to automate some stages of this process in the future.

The creation of the Action/Message table is the main task that should be completed on the integration step. The population of the table with data should proceed as follows. First, the name of each of the scripts from the Visual Interaction Layer that requires institutional verification has to be inserted into the table and the corresponding message has to be assigned to it. Next, each of the message names from the agent-governor protocol has to be inserted into the table and associated with the corresponding script name.

During the integration process it is also required to explicitly specify which scenes or transitions in the Normative Control Layer correspond to which rooms in the Visual Interaction layer and which door should be opened in order to access a desired scene or transition. This is necessary for being able to control the avatar movement inside the Visual Interaction Layer and to determine whether an avatar is trying to enter or leave a scene or transition (so that the state of the Normative Control Layer can be updated accordingly). It is also important to have this information to be able to force the movement of an avatar into a particular scene or transition in case such movement is requested in the Normative Control Layer.

In case the 3D representation of the institutional building was automatically generated from the specification, the mapping between scenes and transitions to rooms and connections to doors is readily available and there is no need for explicit manual efforts from the system designer. Otherwise, if the existing design is reused for the 3D model of the institutional building the automation of this process is not possible. The aforementioned mapping has to be conducted manually. For this purpose the following two tables have to be populated with data.

The first table specifies the details of the scene/room mapping in the Virtual Institution. An example of such a table created for the Trading Institution is presented in Table 4.2. The *Name* column determines the Name of a scene as set in the specifica-

tion of the Normative Control Layer. The *Bounds Object* column specifies the name of the Visual Interaction Layer object that should be used to determine the bounds of the scene. The *Entrance Door* sets the name of the door objects from the Visual Interaction Layer. The bounds of the room objects (the coordinates of the imaginary cube surrounding the room) in Adobe Atmosphere can be easily determined by calling the “roomName.bounds()” function. In case another visualization platform is used and such a function is not available it may be required to populate the table with actual coordinates of the vertexes of the cube rather than the names of the corresponding objects.

Table 4.2: Scene Bounds.

Name	Bounds Object	Entrance Door
meetingRoom	Scene/ceiling	advDoor
egistrationRoom	Scene/registration/registrationRoom	entrDoor
tradeRoom	Scene/tradingfloor/ceiling	alleyDoor

Table 4.3 outlines the details of the transition/room mapping in the Trading Institution. The columns in this table have the same meaning as in Table 4.2.

Table 4.3: Transition Bounds.

Name	Bounds Object	Entrance Door
toMeetingRoom	Scene/main hall/reg2Adv	regDoor
toTradeRoom	Scene/main hall/Adv2Trading	tradingDoor
toOutputFromTradeRoom	Scene/infoRoom/info	exitDoor

As a good practice of the Virtual Institutions development we propose that every institution accepts the entrance of “guest” agents into the registration room where they will be able to request the information about the activities conducted inside and receive authorization to enter other rooms. To be able for a participant to proceed to further rooms he/she should be assigned with an acceptable role. For secure role assignment we suggest using the identification of each user by entering login and password details. We see this process being a common practice and, therefore, each institution is provided with the authorization mechanism. The registration details for different users in the institution are stored in the permissions table. Table 4.4 presents a fragment of such table used by the Trading Institution.

The *User Name* column stores a unique identifier of a user. The *Password* field stores the hashed value of the password for each of the participants. The *Role* field contains the role that should be assigned to a participant in the Normative Control Layer after successful completion of the registration process.

Table 4.4: User Permissions for Trading Institution.

User Name	Password	Role
testBuyer	bdPUEWiFycSfg	Buyer
testSeller	A8c/KTq5QcktA	Seller
roomMgr	n55uVIwxkbHew	roomManager
janedoe	N/70mMPI3B/3g	receptionist

Depending on the situation the permissions table can either be manually filled on the integration step or populated with data during deployment.

Populating the aforementioned tables with data creates a link between the Visual Interaction and Normative Control layers and makes it possible to maintain the tight layer connection at run time and be certain that the Virtual Institution is enacted as expected. In the next section we present the technology that was used to deploy the system.

4.3 Deployment

Applying the Virtual Institutions Methodology described in the previous section results in the creation of the Electronic Institution Specification (the central component of the Normative Control Layer) and the 3D model of the Virtual World (containing the institutional building that represents this specification). The logical connection between the two layers is achieved on the integration step. In order to maintain this connection and make the Virtual Institution functional an additional programming effort was required. In this section we present the result of this effort – the architecture for deployment of Virtual Institutions.

From the software engineering prospective, the deployment of Virtual Institutions can be achieved by designing a single system, where Normative Control Layer and Visual Interaction Layer will remain logically separated, but technologically united. Alternatively, it is possible to make the technological separation and try to reuse existing technologies for implementation of each of the layers.

The reality imposes second choice. On the one hand, such approach is justifiable from the perspective of saving programming resources. Implementing a new architecture for each of the layers would cost significant programming effort, which in case of reusing existing technological solutions can be saved. On the other hand, this technological separation seems to be the most efficient way to deal with rapid development of visualization platform for Virtual Worlds. As it was shown in Chapter 2 it is often the case that one technology used for the visualization of 3D Virtual Worlds quickly changes another and the former eventually disappears. Relying on a particular technology for de-

ployment of Virtual Institutions may have a consequence that the potential users may abandon it. Additional benefit of having the two layers technologically separated is the fact that the Normative Control Layer is quite general and it is not necessary for it to be used only with conjunction with Virtual Worlds. It should be possible for other technologies to access it directly and bypass the Visual Interaction Layer. Such an approach, for example, can be useful for connecting a Virtual World and the real world. Finally, technological layer separation is more flexible in terms of development and allows for distributed programming.

In the architecture we propose the technological separation of the two layers is supported through the introduction of the middle layer (Communication Layer) that is used for technological connection of the Normative Control and Visual Interaction layers. This 3-layered infrastructure is illustrated in Figure 4.12. First layer here is the *Normative Control Layer*. It uses the AMELI system [7] to regulate the interactions of participants by enforcing the institutional rules established on the specification step. AMELI maintains the execution state of the institution and uses it along with the specification to guarantee that participants' actions do not violate any of the institutional constraints.

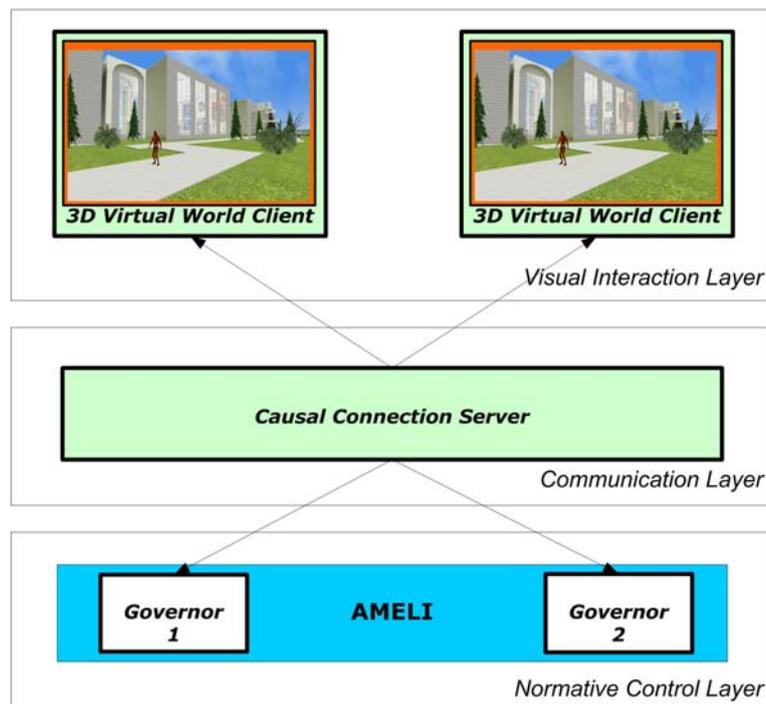


Figure 4.12: Runtime Architecture.

Second layer is the *Communication Layer*. Its task is to causally connect the institutional infrastructure with the visualization system and transform the actions of the visualization system into messages, understandable by the institutional infrastructure and

the other way around. This causal connection is achieved via the Causal Connection Server, which uses the Action-Message table created on the integration step to establish the mapping between actions of the Visual Interaction Layer and messages of the Normative Control Layer. The causal connection is happening in the following way: an action executed in the 3D Virtual World (that requires institutional verification) results in a change of the institutional state in the AMELI layer, as well as every change of the institutional state is reflected onto the 3D Virtual World and changes its state.

The third layer is called *Visual Interaction Layer*. It is used to visualize the 3D Virtual World for the users. The advantages of separating of the runtime architecture into three different layers are listed below:

1. The interactions inside the 3D Virtual World become structured, secure and predictable, as everything that needs control is verified by AMELI and will happen as specified.
2. The Visual Interaction Layer can be easily replaced (i.e. when a more advanced visualization platform appears on the market) with minimal changes required for the rest of the system.
3. The changes in the Normative Control Layer can be automatically reflected onto the Visualization layer or will require minimal manual adjustment.
4. A number of different visualization platforms (possibly implemented via different technologies) can be simultaneously connected to the Causal Connection Server and share the same institution.
5. Some participants (i.e. software agents) can bypass the 3D Virtual World and directly connect to the institution via the Normative Control Layer, while other participants (humans) will be able to observe their presence and actions in the 3D Virtual World.

It is important to understand that through the 3-layered architecture Virtual Institutions change the philosophy behind the security control initially introduced by Electronic Institutions. Direct connection to AMELI system restricts the actions of the user to those that are allowed in the current state of the institution for the particular role a user is playing, nothing else can be done. Having 3D Virtual World as a part of the system creates the possibility to have a different view. The institutional infrastructure does not anymore dictate what the users are allowed to do in the system, but rather controls some critical issues and permits everything else. A 3D Virtual World client usually supports standard actions like walking, chatting, turning around, jumping, etc. Having no institutional rules

present in the system means that everyone is allowed to do everything, because nothing prohibits users to execute any of those actions. The Electronic Institutions Specification enforces the rules of the interactions, which should be obeyed where present.

Next we present a detailed explanation of the technological solution used for deployment of each of the three presented layers.

4.3.1 Normative Control Layer

The Normative Control Layer is Deployed using the AMELI tool provided with the Electronic Institutions Development Environment [7]. AMELI is capable of reading the .xml file with the institutional specification and enforcing the institutional rules specified there. After reading the specification it opens a socket connection for participants and for every new connection done through this socket assigns a new governor agent, which will further control the actions of the connected participant.

The AMELI system maintains the run time state of the institution and updates it as the result of the illocutions sent by a participant to the corresponding governor agent.

4.3.2 Visual Interaction Layer

The Visual Interaction Layer is currently supported by Adobe Atmosphere player embedded into an HTML page accessible through a web interface by web browsers. Each of the participants initially connects to the system by accessing a web form, where a user is requested to type in the identification details (user name and password). After pressing the “Submit” button the connection request is sent to the Communication Layer. The Communication Layer creates a corresponding agent and sends the “EnterAgentInstitution” request to the Normative Control Layer. If the permission to enter the institution is granted – the Web interface is notified and as the result the web browser loads the corresponding .aer file with the 3D model of the Virtual World.

The Atmosphere Player is unable to directly communicate with the Causal Connection Server, therefore, the web interface is supplied with a Java Applet. The Java Applet connects to it using the java socket connection.

The .aer file, which the Atmosphere Player operates with, stores the model of the Virtual World produced as the result of applying the Virtual Institutions Methodology. This model is represented by the set of mathematical equations associated with each object present in the 3D Virtual World. To visualize this file the Atmosphere client applies algorithms for translation of this mathematical models into 2-dimensional images, which are constantly updated as the result of actions of the participant associated with the client or as the result of the actions performed by other participants.

As most of the game engines or other Virtual Worlds platforms, Adobe Atmosphere has a separate component that is used to maintain the states of the participants inside the Virtual Worlds and mechanisms for changing this state. This component is called Adobe Atmosphere Community Server. Every participant entering the system establishes a two directional connection with the Community Server. Every 50 Ms the coordinates of each of the avatars and corresponding transformation vectors are communicated to the Community Server and the Community Server distributes this values to each of the connected clients. In this way each of the participants is able to observe the updated state changes requested by other participants every 50 Ms.

4.3.3 Communication Layer

The task of *causally* connecting the Electronic Institutions runtime environment *AMELI* with the Virtual World visualized by the Visual Interaction Layer is accomplished via the Communication Layer. A system is said to be “causally connected” to its representation if whenever a change is made in the representation, the system itself changes to maintain a consistent state with the changed representation, and whenever the system evolves, its representation is modified to maintain a consistent relationship [130]. *Reflective systems* are a particular case in which the representation of the system is part of the system itself. The Electronic Institution (execution) has a representation of itself in the form of a 3D environment consisting of rooms, avatars, doors and other graphical elements. The 3D visualization provides a possibility for humans to interfere with the agent representing their interests (that runs over the institution infrastructure) through actions in the 3D environment. These actions and the agent’s actions have to be consistent. Thus, a causal connection between the Electronic Institution and the Virtual World seems to be a must.

To achieve the Causal Connection between the Normative Control Layer and Visual Interaction Layer each participant can connect to the system through any of these layers and the participants’ representation in another layer will be automatically created. If a human enters the system as an avatar in the Virtual World – the corresponding agent is created in the Normative Control Layer to represent the human there and communicate the actions of the human to the institutional infrastructure. On the other hand, if an autonomous agent directly connects to the Normative Control Layer – the avatar representing this agent is created in the Visual Interaction Layer.

The causal connection has to materialize in two directions. *First*, the state changes of the Normative Control Layer must have an immediate impact on the Visual Interaction Layer. Movements of an agent between scenes, for instance, must make the corresponding avatar “move” in the 3D Virtual World accordingly. *Second*, actions performed by the human in the Visual Interaction Layer are understood as made by the agent in the

Normative Control Layer. The movement of the human between different rooms inside the Virtual World must result in the changes of the AMELI state so that the associated agent “moves” to the corresponding scenes or transitions. This has a consequence that the actions that the agent is not allowed to perform in the current execution state cannot be permitted in the 3D environment. For instance, if in the current state of the Normative Control Layer it is prohibited for an agent to leave a scene, opening an exit door of the corresponding room must be prohibited to the avatar in the Virtual World.

The architectural component that connects the two layers is the *Causal Connection Server*. For every human that enters the Virtual World through the Visual Interaction Layer the Causal Connection Server sends a request to the AMELI infrastructure for creating an agent in the Normative Control Layer. It can also manipulate this agent and send messages on its behalf. The Causal Connection Server is also able to create avatars in the Visual Interaction Layer if an autonomous agent desires to directly connect to the Normative Control Layer.

Each agent that connects to the Normative Control Layer is virtually associated with a *Governor* inside the AMELI infrastructure. The Governor acts as the third party that communicates the desires of the agent to the institutional infrastructure and answers the requests of the agent about the institutional rules. The messages received by the Governor are forwarded to AMELI for validation. More precisely, AMELI verifies whether a particular message goes in line with the specified rules or not. If a positive validation response is given by AMELI, the requested action gets the permit to be performed. The Governor sends the response of the institutional infrastructure to the Causal Connection Server. As the result of this, the corresponding action is then reflected at the Visual Interaction Layer.

Figure 4.13 outlines the mechanism for achieving the causal connection and illustrates the difference between events, actions and messages.

An event is generated as the result of a human performing an action in the physical world with the goal in mind to change the state of the Virtual World. In the particular case illustrated in Figure 4.13 this physical action is positioning the mouse pointer over the door handle and clicking the left mouse button (requesting the avatar in the Virtual World to open a door by pushing the door handle). Each event that requires institutional verification must have an associated script and the name of this script has to be present in the Action/Message table. If this is the case the name of the script is forwarded to the Communication Layer, where the Causal Connection Server consults with the Action/Message Table to find the message that represents this event in the Normative Control Layer. In case such a message is accepted in the Normative Control Layer the response message is sent back to the Communication Layer. The Causal Connection

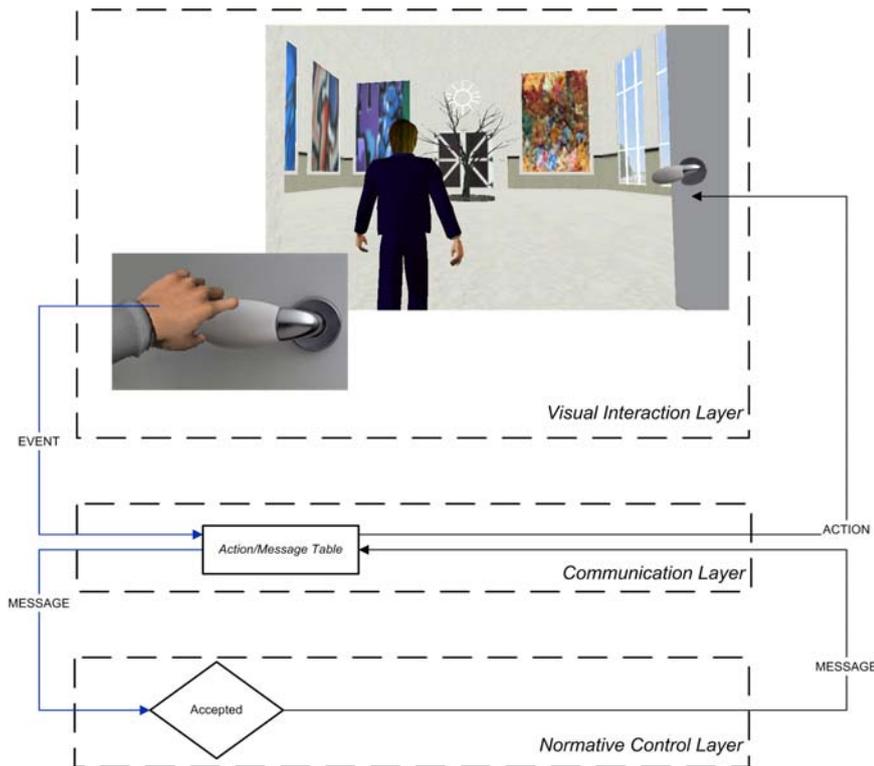


Figure 4.13: Causal Connection.

Server again consults the Action/Message table and transforms this response into the name of the action that has to be executed in the Visual Interaction Layer and the action is performed by executing the corresponding script. In the given example this action will result in opening the door and moving the avatar through the door.

The Causal Connection Server

The development of the Causal Connection Server is one of the key technological contributions of this thesis. While for the deployment of the Visual Interaction Layer and the Normative Control Layer we adapted and employed existing technologies, the Causal Connection Server had to be developed from scratch. The architecture of the Causal Connection Server enables the Communication Layer to be independent from the technologies used in the other two layers. As the proof of this the group of Helmut Berger⁵ in Austria developed a prototype of the Itchy Feet system [18] that was built based on the Virtual Institutions methodology and the Causal Connection Server was successfully employed to connect the AMELI to the Visual Interaction layer based on the *Torque Game Engine* by *GarageGames*⁶. Figure 4.14 demonstrates the interface of the Itchy Feet sys-

⁵<http://ispaces.ec3.at>

⁶<http://www.garagegames.com/>

tem. This system implements a 3D online tourism portal. The institution is represented as a building located inside the garden.



Figure 4.14: The Itchy Feet System.

Further in the thesis we show how the Causal Connection server is used in combination with AMELI and Adobe Atmosphere, however, the reader should keep in mind that these two technologies can be replaced by some other technologies and the Causal Connection Server would still be capable of performing the delegated task.

Figure 4.15 outlines the key components of the three deployment layers with the detailed focus on the architecture of the Causal Connection Server. The AMELI system represents the Normative Control Layer and is featured with two additional components: Federation Monitor and Institution Monitor. These components are used to capture the responses of all governors to all the participants connected to a particular execution of AMELI. The difference between these two components is that the Institution Monitor only controls the responses to the actions inside the institutional building, while the Federation Monitor is capable of capturing the messages that correspond to the actions in the garden, outside any of the institutional buildings. The Institution Monitor and Federation Monitor offer an interface to AMELI, allowing the observation of all messages within a single Electronic Institution platform. The Causal Connection Server is connected to AMELI through sockets provided by these monitors and collects available messages. The collected messages assist in maintaining the synchronized and consistent relation between the Normative Control Layer and the Visual Interaction Layer.

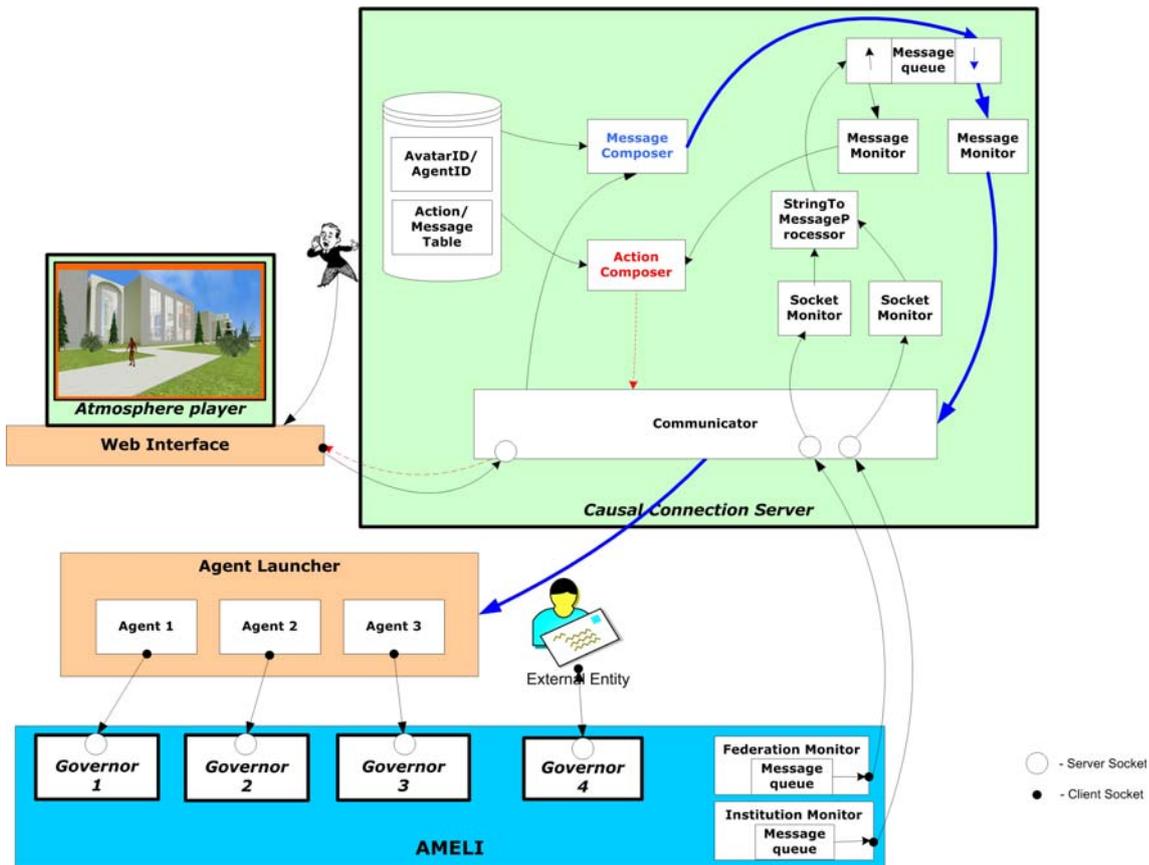


Figure 4.15: The Architecture of the Causal Connection Server.

The Visual Interaction Layer is represented by the Atmosphere Player embedded into a Web interface. A human can only act in the Virtual Institution by controlling the avatar inside the Atmosphere Player.

The Communication Layer is represented by the Causal Connection Server and Agent Launcher. The task of the Agent Launcher is to create agents on request of the Causal Connection Server and to force these agents to send the desired messages to the corresponding AMELI governors. The Causal Connection Server consists of the number of components. First component is the Communicator. Its task is to communicate with AMELI, Agent Launcher and Web Interface. Next, there are two socket monitors, which constantly observe the sockets of the communicator for messages received from Institution and Federation monitors. To be able to communicate the messages through a socket these messages have to be first transformed into string form. The task of the String-To-Message-Processor is to translate them back into a special Message class, that can be used for easy access to the message parameters. As the number of messages that can be received can be quite high and all of them can not be processed in real time, all the messages are first put into the Message queue. In Figure 4.15 two different message

queues are illustrated by two directed arrows pointing in opposite directions. One queue contains the messages received from AMELI and another one – the messages that should be sent to AMELI. Two Message Monitors constantly observe these queues and make sure that all the messages are further delivered for processing.

The processing of the messages received from AMELI is done by the Action Composer. This component consults the Action/Message table to transform a message into an action, which is then further passed to the Atmosphere Player through the Communicator and the Web Interface. The Message Composer does the opposite: it transforms events received from the Atmosphere Player into messages that should be sent to AMELI for verification. In order to create the correspondence between avatars and agents, the AvatarID/AgentID table stores their identifiers and helps in making the mapping between them. Such a mapping is necessary for being able to correctly dispatch the messages and actions to the recipients.

Two types of participants are considered in Figure 4.15, namely humans and autonomous agents. Humans connect to the system via the web interface, through which a user validates the access to the system and if admission to the institution is granted, the Adobe Atmosphere Player [4] is loaded and starts to visualize the 3D Virtual World. At the same time, a message is sent via the Causal Connection Server to the *Agent Launcher* that, in turn, spawns a new software agent. This software agent represents the human in the Normative Control Layer and communicates human's requests to the corresponding Governor, which is created by the institutional infrastructure as soon as the autonomous agent requests to enter the institution.

The second type of participants are autonomous agents (marked as External Entity on the picture) that contact AMELI directly. The agents are unaware of the availability of the Visual Interaction Layer and only know how to interact with AMELI. Each such agent requests access to enter the institution and if the access is granted – communicates with the institution via a newly assigned Governor. The figure demonstrates how the actions of both types of participants are handled by the Causal Connection Server.

An arbitrary event, e.g. a mouse click on a door handle, caused by a human participant leads to a sequence of processing steps. First, the event is caught by the Atmosphere Player and transmitted in terms of a 2-tuple $\langle \text{AvatarID}, \text{Event} \rangle$ to the Causal Connection Server. Then the event tuple is stored in the *Event Queue* which is observed by the *Event Monitor*. As soon as the Event Monitor notices the arrival, it translates the event by means of the *Event/Message* mapping table into the corresponding message. In analogy to that, the *AvatarID* is mapped onto the *AgentID* by means of the *AvatarID/AgentID* mapping table. A 2-tuple $\langle \text{AgentID}, \text{Message} \rangle$ is composed and stored in the *Message Queue*. This time the *Message Monitor* detects the arrival

and sends it to the corresponding agent via the *Communicator*. Finally, the agent passes the message to the corresponding governor. The governor validates whether the received message adheres to the institutional rules and generates an adequate response.

Messages originating from AMELI need to be reflected onto the Virtual World and are processed in exactly the opposite way. Every institutional message is first received by the Communicator, which puts it into the incoming message queue. When the Action Composer is ready to process it, the message is extracted from the queue and is transformed into an action (a script name together with a list of parameters). This action is forced to be executed inside the Adobe Atmosphere Player. As the result of the action the state of the Visual Interaction Layer is updated by the Atmosphere Community Server, that communicates the state update to all the connected clients.

The Figure also illustrates the types of connections that are made on each side. It is shown that the Causal Connection Server opens two server sockets for incoming communication requests from Federation Monitor and Institution Monitor. Both monitors use the client socket connection type to establish the connection with the Communicator. Each of the agents started by the Agent Launcher communicates with the server socket of the corresponding Governor via client socket mechanism. Finally, the Web Interface acts as a client and connects to the server socket of the Communicator component inside the Causal Connection Server.

To better explain the collaboration between different components of the run time architecture Figure 4.16 presents the sequential view of the architecture.

The main architectural components presented here are: Web Interface, Causal Connection Server, AMELI and Governor with the actors: human and autonomous agent.

First it is illustrated how the registration of the human participant is conducted. The human registers through the Web Interface by filling in the form on the web site. As the result of pressing the “Submit” button the web interface requests AMELI to allow the human to enter the institution. AMELI validates this request and assigns a governor. The Web interface is notified about the successful institutional registration as it receives the address of the newly created governor from AMELI. This address is then forwarded to the Causal Connection Server, which creates an autonomous agent for communication with the governor and establishes a socket connection with it. Next, the Causal Connection Server sends the correct URL for the Atmosphere Player and the Web interface loads the document behind this URL and displays the Virtual World.

The sequence diagram also specifies the details of the process of the human trying to execute an action in the Virtual World. As the result of the human generating an event (pressing a key on the keyboard or clicking the mouse) the action request is sent to the Atmosphere Player. The Atmosphere Player requests the Causal Connection Server to find

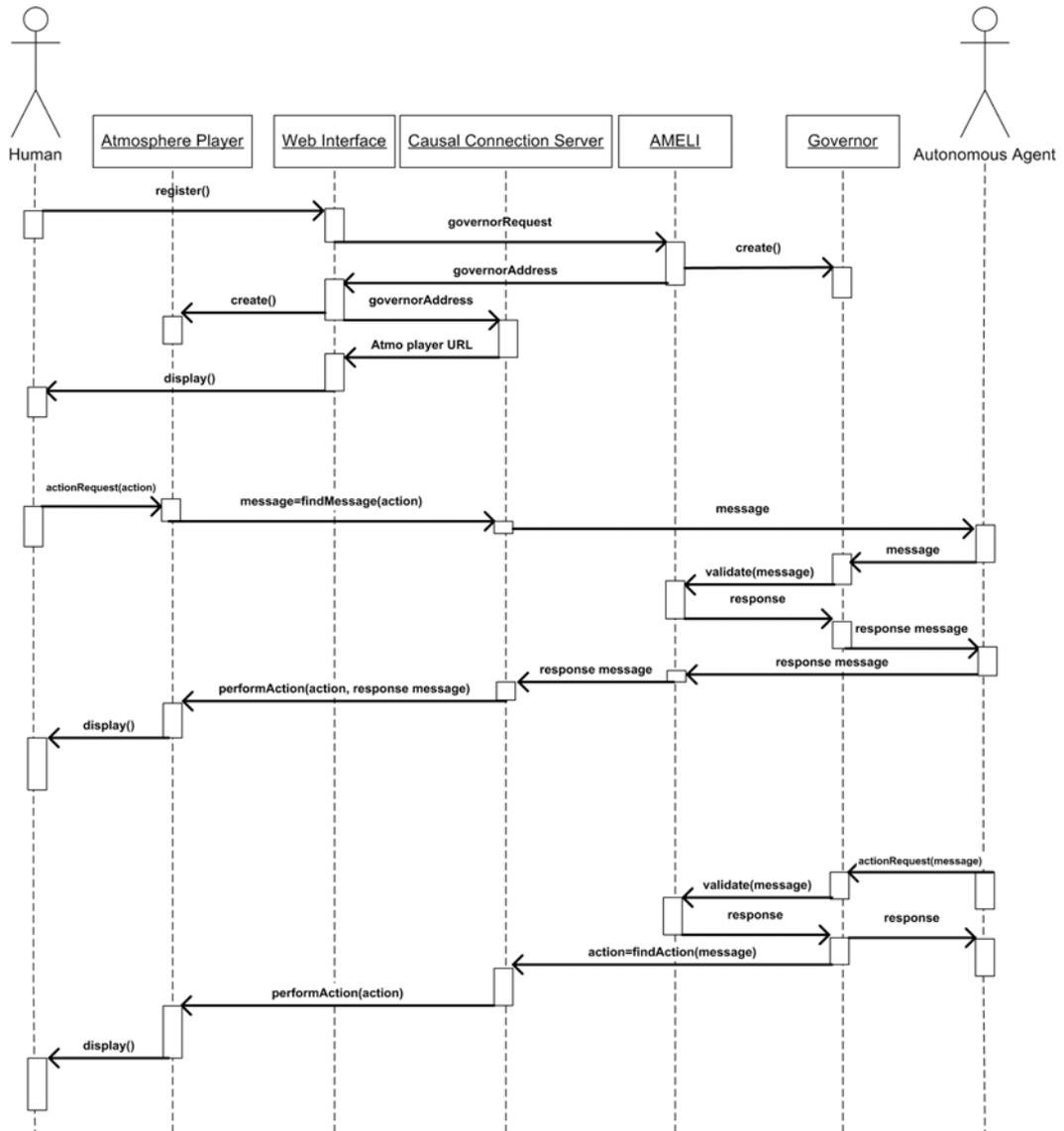


Figure 4.16: Component Interaction: Sequence Diagram

the message that corresponds to this action. When the message is found it is forwarded to the corresponding autonomous agent, which posts this message to the corresponding governor. The message is then verified by the AMELI and the response is sent back to the agent as another message. The agent forwards the response message to the Causal Connection Server, which consults the Action/Message table to find the name of the corresponding script from the Visual Interaction Layer. The resulting script is executed by the Atmosphere Player and the action is visualized through the Web Interface.

The last example illustrated by this diagram is the case of the agent directly acting in the institution on human's behalf. This agent may not even be aware of the existence of the Visual Interaction Layer and may bypass the Causal Connection Server and talk

directly to the AMELI. If the agent desires to execute an action in the Visual Interaction Layer it sends the corresponding request to the associated governor. The governor then validates this message with AMELI's help. The institutional response is then sent to the agent itself and also to the Causal Connection Server. The Causal Connection Server finds the action that corresponds to the response message and then this action is performed.

4.4 Summary

In this chapter we have introduced the Virtual Institutions Methodology. Applying this methodology requires the following seven steps to be completed: eliciting specification requirements, specification of an Electronic Institution, verification of the specification, automatic generation of the corresponding 3D environment, annotation of the Electronic Institution specification with components of the 3D Virtual World, integrating the 3D Virtual World into the institutional infrastructure and enabling implicit training.

Each of the methodological steps has been described in detail and the technological facilities that help automating each of the steps were outlined.

The formal specification of Virtual Institutions described in Chapter 3 has been used for the development of the technological solution presented here. We illustrated the technologies suggested for automatic generation of Virtual Institutions together with tools facilitating annotation and integration steps of the methodology.

The major technological contribution presented in this chapter is the development of the Causal Connection Server, which is used for deployment and acts as a middle layer between Electronic Institutions and Virtual Worlds. The Causal Connection Server is capable of translating the messages of the Normative Control Layer into actions of the Visual Interaction Layer and vice-versa.

The notion of the Causal Connection Server also makes a significant conceptual contribution. It supports connecting the institutional infrastructure to a number of different visualization platforms simultaneously and can even potentially be used for merging the real world and a Virtual World.

Chapter 5

Learning Aspects

The architecture of Virtual Institutions presented in the previous chapter is on the one hand human oriented and, on the other hand, features high agent involvement. The Communication Layer is supplied with the Agent Launcher, which is responsible for the execution of the agents that communicate the events requested in the Virtual World by a human to the institutional infrastructure and receive the institutional responses to those events. In this way a human is integrated into the system always being assigned to at least one agent. The agents are present not only for the sake of the simplicity of implementation and not just because Multiagent Systems based technology (Electronic Institutions) is used for the development of Virtual Institutions. The agent based approach also has another important benefit. Considering the agent/human couple as a joint conceptual and technological block behind each avatar makes it possible to introduce a new way of collaboration between these two entities. One of the key aspects of this collaboration is what we call *co-learning*.

By the term “co-learning” we understand the ability of an agent to learn from a human (also further referred as “principal”) and, at the same time, the ability of the agent to extend the intelligence of the human by presenting him/her with useful information.

Our approach to co-learning is graphically expressed in Figure 5.1. In the Virtual World of Virtual Institutions the couple agent/principal is represented by an avatar. The autonomous agent is always active, and when the human is driving the avatar the autonomous part is observing the behavior of the human, learning from his/her behavior patterns. The avatars are fully controlled by either humans or autonomous agents through an *interface* (the interface is a sort of “glove puppet” that translates all decisions of its “puppeteer” into terms of the institution machine understandable language). Our view is that the agent and the human co-operate in the solution of the tasks the human has to deal with. We want to permit that either the human takes full control over the avatar or that the autonomous agent is in full charge of the decision-making process. Furthermore, we want to allow other types of interaction among them, such as the human giving guidelines to the agent, or the agent suggesting potential solutions to the human (via the interface), in a sort of “expanded intelligence” mechanism similar to the “expanded reality” that nowadays virtual reality tools offer.

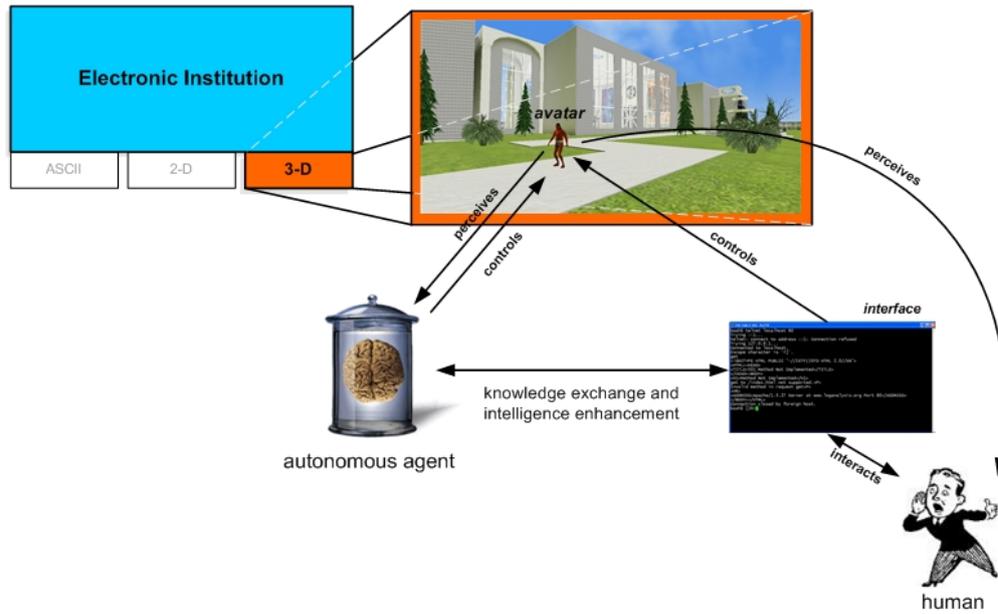


Figure 5.1: Interaction between Autonomous Agents and their Principals.

Having 3D visualization plays a very important role here. It is highly beneficial for humans to perceive the actions of the autonomous agents through avatars – in order to decide whether to intervene or not, and to implement decisions in an easy way. Many people find it useful to delegate some of their activities to autonomous agents but most of them would refuse to give the agents full control over critical operations and would rather like to make those critical decisions themselves. For example, a participant of an E-Commerce environment is not likely to blindly trust an autonomous agent with respect to bidding and payment. Even if an autonomous agent has received precise instructions, a human will still insist on having means to observe the behavior of the agent and to intervene in the decision making process. Observing the actions executed by the agent in the believable environment of the Virtual World and having a simple mechanism to control the agent and interrupt its execution, helps in providing humans with the feeling of security and building trust towards the system in general [189].

The agents themselves also benefit from the 3D visualization and immersive nature of Virtual Worlds. The behavior of humans in a 3D Virtual World is fully observable and much more information about the human can be extracted and learned from. 3D representation of the environment is almost as rich as the real world and provides even more possibilities to observe the behavior of the users than the real world itself. It assumes similar embodiment for all participants, including humans and autonomous agents who imitate the humans. So, every action that a human performs can be recorded and easily understood and reproduced without a need to overcome the embodiment dissimilarities.

In contrast to form-based interfaces, where the human is not fully immersed and only interacts with the system by submitting some data for processing, in the Virtual Worlds' environment the range of actions that can be observed is much wider. Every mouse or keyboard event can be associated with a variety of attributes that play a role in triggering this event, and those attributes can be easily extracted from the environment and analyzed. It is possible to know who is currently present in the system, what does the observed individual see at the moment when an event occurred and even which part of a particular object he/she is focused on. Every movement of the avatar can be precisely described by a set of transformation vectors and each of those vectors can be easily related to the surrounding objects, helping to build a mathematical model of the training data. Moreover, analyzing the movements of an avatar can reveal information about the cognitive state [16] of the human behind it, which is much harder to identify in form-based interfaces.

Additional features that can be observed and learned from the behavior of avatars and spatial relations between them in 3D Virtual Worlds include the following:

- Avatar's Trajectory, which in combination with landmark information may help to predict user behavior (i.e. now he will turn to the supermarket). It may help to identify deviant behavior (i.e. he usually turns to the supermarket after this). May also help to identify user's goal (i.e. bank + travel agent = he is going on holidays) and a degree of commitment to the goal (i.e. bank before the travel agent = strongly committed) [16].
- Avatar's position, eye-direction, head rotation (known vectors in 3D space) can offer even more information about the user, than eye-ball trackers and GPS coordinate readers offer in real world. For example, observing ones avatar in a Virtual World makes it possible to determine how a participant perceives advertisement and even how he/she reacts on it (this can be used for testing on advertisement before screening it on TV or billboards in real world) [84, 201].
- Correlated movements, proximity to others or the similar appearance of the avatars may give information on social relations between different participants or belonging to a particular group [201].
- The appearance of an avatars tells a lot about the personality of the participants [201].
- Invasion of personal space may either identify navigation problems or conflicts between participants [218].
- Nicknames in online communities may be treated as an indicator of trust, reputation or partnership [201].

One of the reasons for our interest in co-learning is the desire to develop human-like autonomous assistant agents capable of believable human-like behavior. Those assistants are particularly useful for commercial applications which require the employment of sales clerks. In Virtual Institutions full observation of the human actions in the Visual Interaction Layer helps in collecting all necessary data for teaching the autonomous agents to act in a similar way the human teacher was acting, while the actions of the Normative Control Layer help in structuring, formal explanation of this data, separating the actions of the human into meaningful states and providing context and background knowledge for learning, helping to explain the tactical behavior and goals of the humans.

Further, we emphasize the need for the training of the autonomous agents by humans, identify understudied areas in the associated research and propose the idea of implicit training. Teaching autonomous agents to act like humans is a growing research field. The main contribution of the thesis to this field is identifying the need for implicit training and justifying that Virtual Institutions is able to satisfy this need. An important point we want to make is that Virtual Institutions provide facilities to learn things that are very hard to learn in form-based environments. To demonstrate this, we will show how the trajectory of human movements can be utilized to reveal some details about the cognitive state of a participant and how knowing it and observing human movements can help to teach agents to execute learned tasks in a believable way.

5.1 Related Work

The demand for intelligent autonomous agents embodied as virtual characters is growing in different kinds of computer-operated environments. The need for tactically smart computer-driven opponents in video games, human-like shopping assistants in E-Commerce or smart guides and travel agents in tourism systems stimulates researchers to look for more and more complex software architectures for controlling the behavior of the autonomous agents.

Many scholars, whose work is focused on such virtual characters, face the problem of making these characters believable. The believability has many different characteristics: personality [157], social role awareness [164] etc. New aspects of believability are constantly discovered and introduced to the research community (e.g. [163], [131]). The complexity of implementation grows, but passing the Turing test is still far from being possible [127].

Instead of trying to discover and implement the different believability characteristics some researchers focus on the simulation theory. The main hypothesis of this theory can be best summarized by the cliché “to know a man is to walk a mile in his shoes” [29].

It is believed that simulation and imitation are the key technologies for achieving believability. Applying the simulation theory to the development of believable autonomous agents is known as *imitation learning*. This approach is not new, but is also not as popular within the learning community as the other types of learning and, most importantly, not very well developed. One of the early examples of this approach is “programming by example” [15], where agents are trained by humans to recognize relevant text on a web page. Similar techniques are also used in robotics, where humans (or other robots) are used to demonstrate some actions which a robot tries to imitate [65]. In [5] robots acquire their behaviors from the provided demonstration of how to solve a certain task, given the necessary initial knowledge to the system. The trainee (robot) then automatically interprets what is to be done from the observed task, thus eliminating the need for tedious textual or iconic programming. A successful approach for training robots by humans despite dissimilarities in the embodiments was proposed in [6]. Similar approach has been used in 3D Virtual Worlds for training virtual characters to recognize and imitate different types of movements [34], but the key emphasis of this research was also on overcoming the embodiment dissimilarities.

Some research in the area of Artificial Life is also highly interesting. The most relevant work originates in the area of computer games. The developers of “Black and white” used a combination of Artificial Life technology and reinforcement learning to train a virtual pet creature for doing menial tasks, like watering villagers’ crops or gathering materials to build structures. Reinforcement learning was also used in [105], where agents learned how to explore a game arena area, to discover as many fields with food as possible and to avoid fields containing poison. Agents evolve to achieve the best possible result, while learning (genetic algorithm based on neural networks) from reinforcement.

There are also some achievements in the area of training of chatter bots using natural language [176]. Transplantable Artificial Neurological Unit (TANU) chatter bots demonstrate the ability to participate in a conversation with a human in a very reasonable way, after some training by a human is conducted. Unfortunately, training is still fairly explicit in nature and doesn’t happen automatically. Training a TANU chatbot is very much like designing a state machine in UML. Authors claim that an average human only goes through about 70,000 important states in a 5 year span. So, creating 70,000 states properly interconnected with transitions would make a very smart chatbot. The TANU network runs language aware transitions, so a transition that supports an event “Can you teach?” will automatically be mapped to a transition from the source state when it receives “Can you educate?”, “Can you tutor?”, “Can you lecture?”, “Can you instruct?” and “Can you edify?”. The problem the developers of the TANU bots face is that there are no implicit training mechanisms used for training and creating 70,000

states requires a significant effort from the human operators.

The presented solutions have a great potential to address many aspects of believability ranging from text-based interactions and personality modeling to learning from the principals that have a different embodiment. While the research on those aspects is more or less established and its results can be successfully used, there are still some components of the believability that are understudied. One of those is the believable human-like movement of the virtual characters. The problem of teaching autonomous agents to express believable movements has started to be analyzed quite recently. The movement of the majority of virtual characters nowadays is still being established using Artificial Intelligence techniques developed a few decades ago (e.g [120], [70]). The imitation learning is an obvious choice here, but it is mostly used in robotics and only few researchers are concerned with applying it to virtual characters in 3D environments (i.e. [124], [81]). The research conducted by [81] is the most relevant work in this area, however it is only concerned with video games and, therefore, has a number of drawbacks, limitations and simplifications.

The algorithms described in [81] seem to be quite successful in teaching the agent reactive behaviors, where next state an agent should switch to is predicted on the basis of the previous state and the set of parameters observed in the environment. These algorithms also prove to be quite useful in learning strategic behavior inside a particular video game (Quake II). The main limitation we see in this approach is that the long term goals of the players are assumed to be quite simple, namely to collect as many items as possible and to defeat their opponents [204]. At this simplistic level the authors do not need to identify different logical states behind the agent's decision making but simply need to create a topological representation of all positions that the human visited in the environment and subdivide it into a number of position prototypes. Next, the transitions between the prototypes are made using the probabilistic approach. Provided the human only has simple goals (as described above) this method is quite sufficient. However, in many applications, including real world electronic business (which is of high interest to us), this is not the case. Not only are the goals more complex, but there is also a need to be able to instruct the agent on achieving a subgoal (i.e. to answer customer enquiries about the product or to participate in an auction on the user's behalf). Such tasks are impossible to achieve using the algorithms presented in [81].

As our research is mostly focused on virtual characters collaborating with humans and assisting them in Virtual Worlds, another problem we are faced with is letting the autonomous agents understand non-verbal cues expressed by humans. This need was identified by [44], where assistant agents in virtual museums were required to classify users based on their movement style in order to provide them with adequate exhibition

presentation. Such a classification is highly beneficial for any application with a high degree of human-agent interaction and the authors have provided enough motivation for its significance and had even developed a graphical tool to assist them in analyzing the movements of museum visitors and identifying 4 distinct classes of behavior. However, the problem of real time analysis and classification of human behavior based on the avatar movement still requires further investigation.

To complement the lack of research in this field we have considered research on mobile devices and wearable computing as being very useful. In [16] the authors propose that when dealing with mobile users (moving around with PDAs) the use of motion-based information can reveal some aspects of the cognitive state of the humans. The authors claim that the mood of humans, and a degree of commitment to a goal (as well as the goal itself) can be analyzed and predicted from non-verbal cues. In [10] it is shown how tracking GPS coordinates of users with PDA-devices helps to identify significant locations for the users on the basis of their trajectories.

Having completed the literature review we will now identify the most understudied areas in research on training virtual characters as *teaching believable human-like movements to autonomous agents* and *assessing the cognitive state of the humans based on non-verbal cues they express*. Virtual Institutions, in our opinion, have great potential in providing additional research evidence in these areas. We aim to lay the foundation for future research by showing how this technique should be applied to each of them.

Further in this chapter it is demonstrated how in the systems built based on the Virtual Institutions metaphor, imitation learning becomes an integral part of the development process, and how the architecture we propose helps to achieve what we call *implicit training*. Then we explain the concept of implicit training in more detail and present the experimental results that demonstrate that intelligent spatial behavior can be simulated by simply imitating humans, without any need to program this behavior or to explicitly teach it. We propose an algorithm for run-time assessment of the humans' cognitive state based on the trajectory of their movement and show how this information can be utilized in commercial applications.

5.2 Implicit Training

Implicit training in this thesis is defined as follows:

Definition: *Implicit training* is the training of an autonomous agent from interaction with a human when the human is not explicitly involved in training (and might not be even aware of it) but trains an agent by performing some routine tasks (i.e. answering customers' enquiries), which agent learns through imitating the human.

In contrast to implicit training, explicit training is the method where training of the agent requires explicit teaching effort from the human.

The implicit training aspect is particularly interesting in its application to the domain of E-Commerce, where one of the aims of the training is to produce intelligent and believable human-like sales assistants, which implicitly learn from human operators how to respond to customer enquires.

To show the capabilities of the proposed implicit training approach, outline the technological solution for achieving it and demonstrate the benefits of this method we consider the following scenario.

5.2.1 Scenario

The scenario is based on the Trading Institution from Section 3.2 and takes place in the Meeting Room of this institution. The artist in the Meeting Room is replaced with an Assistant, whose goal is to provide the visitors of the graffiti poster exhibition with adequate assistance. Instead of being programmed, this agent will be trained by human operators via implicit training to assist the buyers in a believable human-like way, copying the assistance style of the principals who conducted the training.

Initially, a human operator on the Assistant side answers customers' requests while the corresponding autonomous agent observes the behavior of the human and learns from it. Eventually, the human decides to pass control over the Assistant avatar to the autonomous agent, which replaces the human and starts communicating with customers itself. In case a customer puts an enquiry that an agent is not trained to deal with, the agent requests help from an available human operator. The human interchanges the software agent (with or without explicitly notifying the customer about this) and answers the non-standard enquiry. Unlike explicit training, the autonomous agent meanwhile observes the human operator and learns how to deal with similar inquires in the future (extends its intelligence).

Figure 5.2 presents a schematic representation of the updated Trading Institution and shows some aspects of the user behavior inside. We would like to quickly remind the reader that the institution consists of three functional rooms, where the Registration Room is used for identifying the users and assigning them with corresponding roles. The Meeting Room acts as a graffiti poster exhibition and supports communication between Buyers and the Assistant agent. The Trade Room is used for conducting different kinds of auctions.

The solid black figure on the picture represents the participant, which initially plays the external role "Guest" , but after the authorization in the Registration Room it will be playing the "Buyer" role. Other characters correspond to internal agents (employees

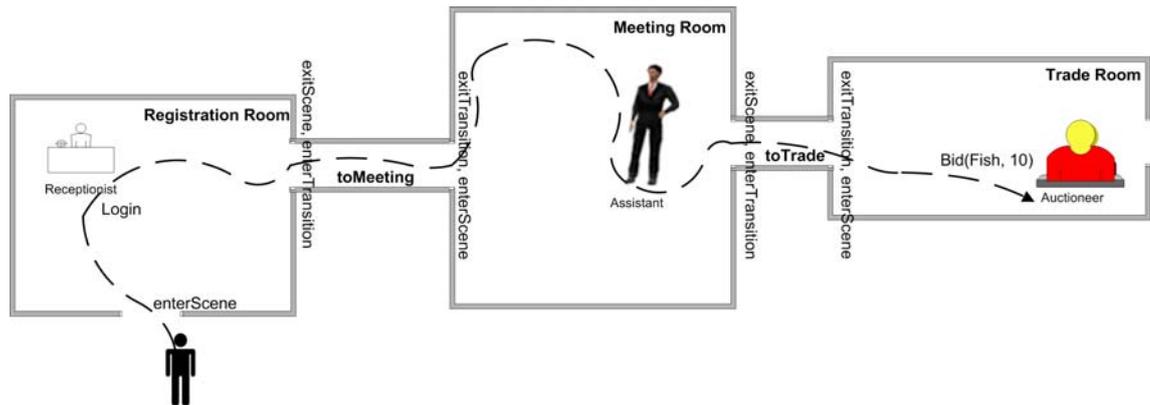


Figure 5.2: Visual and Normative Levels of Execution.

of the institution) Receptionist, Auctioneer and Assistant. The Receptionist welcomes the participant in the Registration Room, verifies the login and password and unlocks the doors to other scenes if the identity of the participant is proven. The Auctioneer sells different goods in the Trade Room. It announces the product to be auctioned, waits for incoming bids and sells it to the winner. The Assistant is helping the Buyers in the meeting room with information on the poster exhibition.

The implicit training approach addresses the behavior of two agents: Buyer and Assistant, involved in the business interaction. In the remainder of the section we shall focus on the following two aspects of this approach:

- *On the Buyer side:* how to train the Buyer agent so that, given an instruction, it will be able to execute it in a believable way, imitating the human teacher;
- *On the Assistant side:* how the trajectory of the Buyer can help in analyzing the cognitive state of the Buyer and how knowing some aspects of Buyer's cognitive state can be used in training the Assistant to select the right strategy in assisting the Buyer.

On the Buyer side implicit training is used in the following way. The human Buyer enters the institution a couple of times and executes a number of actions. The autonomous agent associated with the Buyer records these actions until the moment when the human exits the system. An example of the set of actions expressed by a Buyer during one of the visits to the institution is expressed in the Figure 5.2. There the Buyer enters the Registration Room and then, through the Meeting Room and the related transitions, enters the fish auction conducted in the Trade Room, where the human places a \$10 bid for a box of fish. One of the tasks we potentially want to achieve is that after a couple of training sessions the human may instruct the agent to buy fish or to perform some other action, which the agent will be able to execute in a similar way to the human.

On the Assistant side while the avatar of the Assistant is controlled by a human operator, the Assistant representative agent tries to identify the “shopping mood” of every approaching Buyer and present this information to the human. The shopping mood is detected on the basis of the approaching trajectory of a Buyer. Depending on the shopping mood the operator behind the Assistant avatar will select the right assistance strategy. While assisting the Buyer, the human operator will implicitly train the assistant representative agent how to behave in a given situation.

5.2.2 Visual and Normative Levels of Execution

In order to be able to present further details of the implicit training method, we shall first introduce the concept of different levels of execution of a Virtual Institution. As described in the Section 3.1 a Virtual Institution consists of 2 conceptual layers: the Visual Interaction Layer and the Normative Control Layer. This conceptual subdivision is also reflected on the runtime infrastructure, where we distinguish between visual and normative levels of execution.

The normative level ensures that the institutional rules are not violated. On this level a participant is restricted to sending a text message to the institution for verification, requesting to perform an action. If under the current circumstances the action is allowed to be performed without violating the rules, the agent receives a response and the action is performed and visualized. The way the actions are visualized can not be changed and their execution can not be terminated. For human participants sending text messages is transparent as it happens as a result of their actions in the Virtual World. The normative level actions presented in Figure 5.2 are marked as: (enterScene, exitScene, enterTransition, exitTransition and login). These labels correspond to textual messages that will be sent to the institutional infrastructure as the result of the corresponding actions of an avatar in the Virtual World. For example, the fact that a “Buyer” agent collides with a closed door will result in the Buyer representative agent sending the “enterScene” message to the institutional infrastructure. If entering the room is not prohibited by the institution, the institutional infrastructure will reply with “enteredScene” message, which will result in opening the door and moving the agent through the door.

Actions that are not controlled by the institution are performed at the visual level. They are executed directly by the participant without prior verification by the institution. For example, there is no need for the institution to specify how the participants should walk from one room to another. Requesting such an action does not trigger an institutional message and, therefore, the action is freely executed. An example of the sequence of visual level actions is presented in Figure 5.2 as a black dotted line which ends with a directed arrow. This line illustrates the trajectory of the participant’s movement through

Registration Room, Meeting Room, Trade Room and the transitions connecting them.

For effective building of user profiles and for making the implicit training of participants possible the actions from both levels have to be observed by autonomous agents. The actions of the normative level, on the one hand, help autonomous agent to understand when to start and when to stop recording the actions of the visual level and which context to assign to the recorded sequences. On the other hand, analyzing the sequence of normative level actions helps, in the long run, to understand how to reach different scenes and states in those scenes. The actions of the visual level are very important for teaching human characteristics to autonomous agents and modeling the user preferences.

At any time, a human may decide to let the agent control the avatar via ordering it to achieve some task. If the agent is trained to do so it will first find the right sequence of normative level actions that lead to completing the task and then will use the actions of the visual level to know how to make believable transitions between the actions of the normative level.

The underlying formal specification of a Virtual Institution contributes to learning the human-like behavior through helping autonomous agents to structure the actions of the human and providing more control over the data used for training. Every dimension of the institutional specification contributes to the quality of learning in the following way:

- *Dialogical Framework*: the roles of the agents help to separate the actions of the human into different logical patterns. The message types specified in the ontology help to create a connection between the objects present in the Virtual Worlds, their behaviors and the actions executed by the avatars.
- *Performative Structure*: Helps to group human behavior patterns into actions relevant for each room.
- *Scene Protocols*: Help to create logical landmarks within human action patterns in every room.

5.2.3 Learning to Imitate the Human

To better explain how the training on each level is conducted, we refer back to the scenario outlined in Figure 5.2. After completing the integration step of the Virtual Institutions methodology for this institution, behavior visualization of the autonomous agents in the visual level was preprogrammed in a simple way (e.g. entering a scene is visualized as making the avatar disappear in the previous room and appear in the middle of

the entered room). These actions are programmed as scripts and the scripts are assigned with the corresponding messages in the action/message table.

As an example, let's observe what happens if the human takes control over the avatar to complete the scenario expressed in Figure 5.2. After entering the institution the autonomous agent starts recording the actions at the visual level as it receives the message "enterScene" as the result of a human asking permission to enter the Registration Room. When the login message is sent by the human through the interface the agent realizes that the context at the visual level has changed (which means that all new actions have to be observed in a new context). In a similar way the agent records all other actions and assigns context to them when the human moves throughout scenes and transitions to the Trade Room. At the normative level the agent records the whole sequence of the institutional messages sent by the human (enterScene, exitScene, enterTransition, exitTransition, login, bid).

Now let's assume that as a result of participating in the fish auction the human has purchased a box of fish for the price of \$10 and has left the institution. The next time the human enters, he/she expects that the corresponding autonomous agent has already received enough training, and uses a special command "Do:bid(fish, 10)" to instruct the agent to buy the fish. As a response to this command the agent searches the prerecorded sequence of normative level actions for the appearances of bid(x,y) function. Once the function is found, the agent knows which sequence of normative level actions will lead to achievement of the goal. At the visual level the agent knows which actions it has to execute for a believable imitation of the human performance. The aforementioned reasoning will result in the following behavior: the avatar enters through the door into Registration Room, the avatar reproduces the trajectory of the human and approaches the reception desk, the request for login information is received and the agent sends the login details. In a similar way the agent continues its movement to the Trade Room, where it offers \$10 for the box of fish. If the agent wins the lot – the scenario is finished; if the price this time is higher – the agent will request human intervention.

The same approach is valid for both training the Buyer agent and training the Assistant agent. The only difference is that the Buyer in our scenario is given a precise instruction on what to do, while the assistant will remain idle until the state of the environment changes and the assistance is required.

An important aspect in training is to be able to execute the right sequence of actions in the right context. For the purpose of our scenario we limit the context to the following 5 parameters: the objects that are visible to the human during training, the distances to those objects, the avatars visible to the human, the distances to the avatars and the cognitive state of the human.

While the first 4 parameters are pretty much self explanatory the cognitive state and the way to asses it requires additional explanation.

5.2.4 Assessing the Cognitive State

Cognitive state is a very broad term. It is used in different disciplines and has various definitions. Below is the definition that most accurately reflects what is understood by the cognitive state in this thesis.

Definition: *Cognitive State* is the state of a person's cognitive processes¹. In DAI cognitive state is usually associated with intentions, beliefs and desires of an individual [113].

It is not the goal of this thesis to analyze different aspects of the cognitive state and provide a comprehensive study on how they can be learned. Instead, our goal is simply to show the potential that Virtual Institutions have in analyzing it. Therefore, the presentation here is limited to analyzing only one aspect of the cognitive state, namely the mood of the customer inside a Virtual World.

The mood aspect of the cognitive state has received particular attention in ethnography [44]. Based on the mood of the people visiting art expositions in museums researchers identified four distinct categories of visitors, briefly summarized in the following [44].

1. The *ant visitor* spends quite a long time to observe all exhibits, stops frequently and usually moves close to walls and exhibits, avoiding empty spaces.
2. The *fish visitor* moves preferably in the center of the room, walking through empty spaces. Fish visitors do not look at details of exhibits and make just a few or no stops; most of the exhibits are seen but for a short time.
3. The *grasshopper visitor* only sees the exhibits which comply with grasshopper's interests. These personal interests and pre-existing knowledge about the contents of the exhibition guide the grasshopper. The grasshopper quickly crosses empty spaces, but the time spent to observe single selected exhibits is quite long.
4. The *butterfly visitor* frequently changes the direction of visit, usually avoiding empty spaces. The butterfly sees almost all the exhibits, stopping frequently, but times vary for each exhibit.

The set up in the Meeting Room of the Trading Institution is pretty close to the art exhibition in a museum. So, for the purpose of the Trading Institution scenario it is also

¹<http://www.dictionary.com>

possible to distinguish between 4 different mood state values representing the cognitive state as applied to Buyer's behavior in the Meeting Scene: "Ant", "Grasshopper", "Butterfly" and "Fish". In contrast to a museum in our scenario only one room is used for the exhibition.

One of the goals of the scenario expressed in Section 5.2.1 is to use the trajectory of the Buyers to predict some aspects of their cognitive state. In order to do so we need to analyze the behavior of the visitors dynamically, meaning that it is not acceptable to wait until a participant exits the room to be able to classify the mood. The mood classification is required to be completed before the participant approaches the Assistant agent. Having these limitations requires a slight modification of the legends behind the behaviors presented in [44]. As adapted to our scenario, in the "Ant" state the main task of the user is considered to be walking around the room and collecting the visual information presented there. In the "Grasshopper" state a user is focused on particular items in the room (graffiti posters) and requires more information about them. In the "Butterfly" state the user is experiencing the problem with either navigating the Virtual World or with the presented information. While being in "Fish" state means that the user is mainly focused on some task outside the Meeting Room, wishes to pass through the room as quickly as possible and doesn't want to be distracted from the main activity.

The mood of the visitors of the graffiti poster exposition in the Meeting Room affects the way they perceive information. For the assistance purposes knowing it is very important to determine which strategy to select and to understand whether any assistance is required at all. Clearly the person in the "Fish" state would not be very excited about hearing the information about graffiti posters and the person in "Butterfly" state experiencing navigational problems would first like to know how to solve the problem and only then may express some interest in the poster exhibition.

Following the assumptions made in [44] and [16] the cognitive state (and the mood in particular) of an individual is tightly connected with the trajectory of his/her movement. Assessing the cognitive state in our scenario is supported by shallow reasoning. We record a classification list of trajectories generated by avatars driven by humans, the time the avatar spent on the trajectory and the utterances in which the avatar was engaged. The humans that were driving the avatars then manually associate each trajectory with a corresponding class label. The final classification is refined by an expert, which decides on the most characteristic trajectories that uniquely reflect the label.

5.2.5 Distributed User Modeling

In our scenario the Assistant agent has to be able to assess the cognitive state of the Buyer approaching the Assistant in the Meeting Room to select the right assistance strategy. As

shown above, the trajectory of the Buyer could reveal some of the aspects of the cognitive state. Analyzing a trajectory of another avatar, however, is a very challenging task associated with a number of problems. The Assistant agent has to be able to constantly observe every Buyer that enters the room and to translate the movements of the Buyers that it observes into arrays of landmarks. Without being able to directly acquire this information from the system, analyzing the cognitive state becomes nearly impossible.

Obtaining landmarks and their precise coordinates just from observation of the movements of one avatar through the field of view of another avatar is a very challenging task. This task could be simplified by letting the system provide every agent with detailed information about all the other agents (including the positions of the corresponding avatars at a given time). Such simplification, however, raises serious privacy concerns, as providing detailed information about participants can become a basis for abuse.

To be able to make the trajectory recognition achievable and, at the same time, ensure the privacy of participants we propose the following decentralized solution to the problem. Each autonomous agent only observes its principal and dynamically updates the user model of the principal (in our particular case the user model only constitutes the cognitive state). When some other agent (i.e. the Assistant) needs to obtain some information from the user profile of this agent (Buyer), instead of trying to observe the behavior of its avatar and use sophisticated modeling techniques, it simply sends a direct request to the agent “responsible” for the corresponding avatar. If the agent on the other end agrees to share the information it will reply with the relevant part of its user profile.

Such decentralized solution is feasible because the duality (agent/principal) is a general feature of Virtual Institutions and every participant is integrated into the system via such architecture. The proposed approach can significantly reduce the amount of computations and the size of the stored data. It also permits easier control over privacy (e.g. if a participant doesn’t want to be observed he/she just prohibits the agent to share personal information with others or may even select which aspects of the user profile can be shared and which aspects are private). The decentralized approach also helps to easily use the characteristics of the user profiles of the surrounding agents in the system as the attributes for learning. The developed architecture supports that an agent that observes the behavior of its principal may directly communicate with the agents attached to the avatars currently visible to it and ask them to share a particular part of their profile. The elements of the profile are then used as the input for classifier, whose task is to determine whether those influence the behavior of the human or not.

Figure 5.3 graphically presents the idea of distributed user modeling. It outlines the case of a human controlling the avatar marked as “Buyer” and an autonomous agent controls the avatar of the “Assistant”. In the beginning of the scenario outlined in the

figure the Assistant representative agent notices a Buyer approaching its avatar. To be able for the Assistant to detect that it was approached the notion of *audibility distance* is used. The audibility distance represents the radius of the imaginary circle surrounding the avatar (audibility region), which is used to determine the range in which everything that is said by any other avatar will be heard, while outside of this range nothing can be heard. Audibility distance is a very useful concept for facilitating social interactions and providing a natural way to filter the communications. Another purpose of the audibility region in our system is that the fact of entering the audibility region of one avatar by another avatar means that the first avatar was approached by the second one.

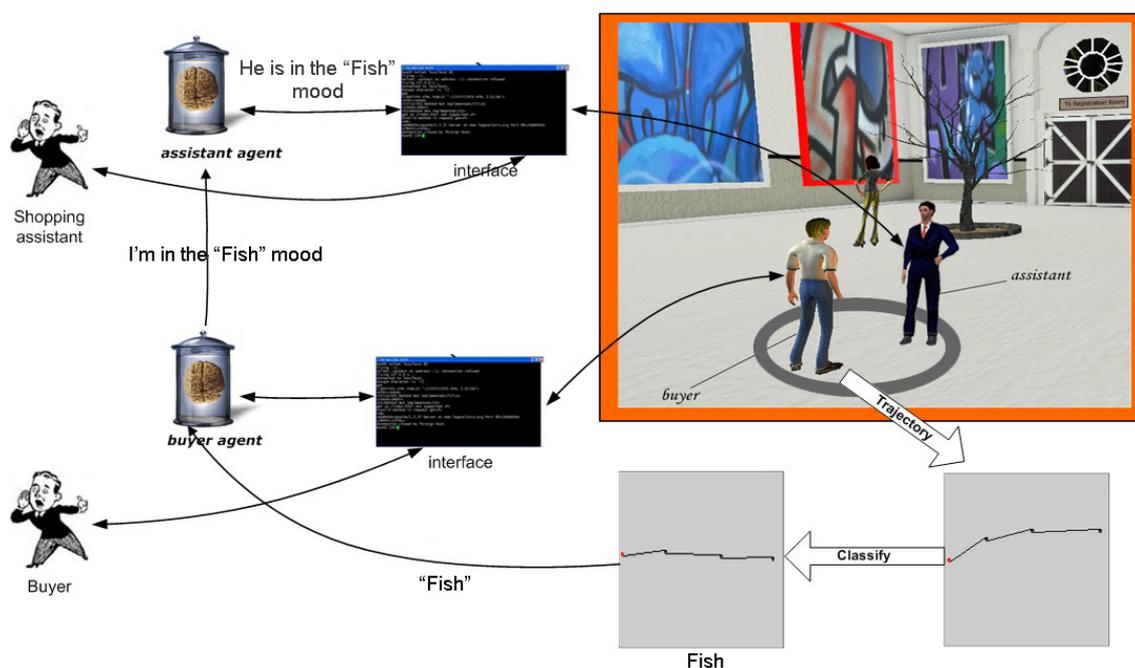


Figure 5.3: Distributed User Modeling

Once approached, before getting involved into a conversation the Assistant requests the description of the Buyer’s cognitive state from the Buyer representative agent. In its current form the state corresponds to the label describing the mood of the human. This label is acquired by the Buyer representative agent through comparing the current approaching trajectory of the Buyer with the set of predefined prototypes and extracting the label of the prototype that has the closest match. In the scenario outlined in the figure this label is “Fish”.

Once the label is assigned it is sent back to the Assistant representative agent. Before starting a conversation, the Assistant already knows that it shouldn’t bother the visitor with additional information about the posters and if asked directly, its responses have to be very short and precise.

Notice that it is also possible that the Assistant avatar is controlled by the human. In this case the corresponding autonomous agent will inform the human about the mood of the approaching participants and the human will be able to use this information for selecting the right strategy, and will train the agent accordingly.

5.3 Implementation Details

To verify the aforementioned concepts and ideas we modified the Trading Institution prototype so that it corresponds to the scenario outlined in Figure 5.2. From the institutional perspective the system consists of a Federation, called “TradingFederation”, and one institution inside it, called “TradingInstitution”. Although the notion of Federation is not present neither in the conceptual model of Virtual Institutions nor in its formalization it is introduced in the current version of AMELI software that we use. The reader should consider Federation being a unification of institutions serving a similar purpose. Entering a Federation in the Virtual World corresponds to an avatar entering the Garden surrounding the institutional buildings. In our scenario we have only one institutional building and entering the garden corresponds to starting the system.

The Trading Institution, represented as a building in the garden, contains 3 functional scenes (Registration, Meeting and Trading) and the transitions connecting them. The participants in the institution can initially play two different roles: Receptionist (internal role) and Guest (external role). Further on, a Guest can become either a Buyer or a Seller.

The system is visualized as a 3D Virtual World, which contains a garden (TradingFederation) and an institutional building inside the garden. Scenes and transitions are visualized as rooms inside the institutional building. Each participant starts as an avatar in the garden, from where it can enter the building (join the institution) and continue moving through the rooms (scenes and transitions) present in this building.

Next, we explain how the trajectory classification was conducted, as well as outline the methods used for recording of the human actions on both visual and normative levels of execution. The recorded information is used by the agent to believably imitate the movements of the human. Additionally, we describe how the recordings are classified and how the parameters used for classifying them are discovered.

5.3.1 Comparing the Trajectories

The goal of trajectory comparison is to show the capabilities of Virtual Institutions in respect to supplying an autonomous agent with details of the principal’s user profile that are nearly impossible to gain in form-based interfaces. An example of such profile

details is the “mood” aspect of the cognitive state, which can be estimated on the basis of the trajectory of the avatar movement. We do not intend to prove the connection between the movement of the avatars and the cognitive state of the humans generating these movements, but rely on the outcomes of the research presented in [44] to make this link. We also use the adapted versions of the four trajectory classes presented in [44] as the basis for the trajectory comparison.

Each of the four trajectory prototypes is stored in the classification list. In order to classify the approaching trajectory of an avatar we compare it with every trajectory in the classification list and identify it as the most similar one from the list. The label associated with the resulting trajectory is considered to be the result of the classification.

Technically, the trajectories are specified as arrays of landmarks. Each of the landmarks corresponds to a position of an avatar in a given moment. The position is permanently updated by the system every 50 Ms, so the information about avatar’s velocity is easily reconstructed from the distance between two neighboring landmarks. This simple representation allows efficient trajectory classification. To increase the performance of the classification on the first step of the algorithm the irrelevant landmarks (noise) are removed using the approach presented in [156].

After this, a combination of Levenshtein Distance and Euclidean Distance algorithms is applied to compare the analyzed trajectory with each trajectory stored in the classification list. As the result of the comparison, the trajectory from the classification list with the lowest distance value is selected and the corresponding text value is extracted to be used as a behavior label for the cognitive state of the human.

The Levenshtein Distance is the algorithm normally used to measure the distance between two strings. It determines the minimum number of operations needed to transform one string into another given string, where possible operations are insertion, deletion, or substitution of a single character [125].

The steps of the Levenshtein Distance Algorithm [125] are presented in Table 5.1. Here s and t are the two strings being compared, n – the length of string s and m – the length of string t .

In order to apply the Levenshtein Distance to the trajectory comparison we propose the following modifications to the original algorithm. Firstly, instead of comparing the Strings we compare arrays of landmarks. So each of the $s[i]$ and $t[j]$ will be a point in a 3-dimensional system of coordinates – (x_i, y_i, z_i) and (x_j, y_j, z_j) , respectively.

Second change is the replacement of the *cost* assessment model. In the original algorithm the cost can be seen as the actual distance between two characters. This cost model is very simple and is equal to “0” if the two characters are similar and is equal to “1” if the characters are different.

Table 5.1: The Steps of the Levenshtein Distance Algorithm.

Step	Description
1	Set n to be the length of s. Set m to be the length of t. If n = 0, return m and exit. If m = 0, return n and exit. Construct a matrix containing 0..m rows and 0..n columns.
2	Initialize the first row to 0..n. Initialize the first column to 0..m.
3	Examine each character of s (i from 1 to n).
4	Examine each character of t (j from 1 to m).
5	If s[i] equals t[j], the cost is 0. If s[i] doesn't equal t[j], the cost is 1.
6	Set cell d[i,j] of the matrix equal to the minimum of: a. The cell immediately above plus 1: d[i-1,j] + 1. b. The cell immediately to the left plus 1: d[i,j-1] + 1. c. The cell diagonally above and to the left plus the cost: d[i-1,j-1] + cost.
7	After the iteration steps (3, 4, 5, 6) are complete, the distance is found in cell d[n,m].

In our case we are dealing with arrays of landmarks instead of characters. Each landmark has a unique coordinate in the 3-dimensional space and, therefore, instead of just having “0” and “1” we can employ a more appropriate distance measurement technique, namely, Euclidean Distance.

The Euclidean Distance between two points in a 3D space is calculated as follows:

$$D_{euclid} = \sqrt{(p_1.x_1 - p_2.x_2)^2 + (p_1.y_1 - p_2.y_2)^2 + (p_1.z_1 - p_2.z_2)^2} \quad (5.1)$$

Here D_{euclid} is the Euclidean distance, p_1 and p_2 are the landmarks for which the distance is measured and (x_1, y_1, z_1) and (x_2, y_2, z_2) are the coordinates of p_1 and p_2 correspondingly.

In theory the values of the Euclidean Distance can vary between “0” and infinity, practically the distance is always limited by the dimensions of the space where the measurement is taking place. The cost value in the original Levenshtein Distance algorithm is required to be normalized (take values in the [0,1] interval). Therefore, instead of using pure distance value we use the following equation:

$$cost = \frac{D_{euclid}}{\sqrt{R.width^2 + R.height^2 + R.depth^2}} \quad (5.2)$$

Here $cost$ is the value which should be used instead of “1” on the step 5 of the Levenshtein Distance algorithm. And $R.width$, $R.height$, $R.depth$ are the dimensions of the

room where the trajectory comparison takes place. In case the trajectory comparison is required to be done outside any of the rooms, R should correspond to the space where the experiment is conducted with $width$, $height$, $depth$ being the dimensions of this space.

5.3.2 Recording

The learning-related information for each of the agents is stored in a separate learning graph (Figure 5.4). The nodes of this graph correspond to the institutional messages, executed in response to the actions of the human inside the institution. Each of the nodes is associated with two variables: the message name together with parameters and the probability ($P(Node)$) of this message to be executed. The probability is continuously updated, and in the current implementation it is calculated as follows:

$$P(Node) = \frac{n_a}{n_o} \quad (5.3)$$

Here n_o is the number of times a user had a chance to execute this particular message and n_a is the number of times when he actually did execute it.

The arcs connecting the nodes of the graph are associated with the prerecorded sequences of the visual level actions (s_1, \dots, s_n) and the attributes that influenced them (a_1, \dots, a_n). Each pair $\langle a_n, s_n \rangle$ is stored in a hashtable, where a_i is the key of the table and s_i is the value.

Each of the sequences (s_i) consists of the finite set of visual level actions:

$$s_i = \langle SA_1, SA_2, \dots, SA_q \rangle \quad (5.4)$$

Each of those actions defines a discrete state of the trajectory of avatar's movement. They are represented as the following vector:

$$SA_l = \langle p, r, h, b \rangle \quad (5.5)$$

Where p is the position of the agent, r is the rotation matrix, h is the head pitch matrix, b is the body yaw matrix. Those matrixes provide the most typical way to represent a movement of an avatar in a 3D Virtual World.

Each a_i is the vector of parameters:

$$a_i = \langle p_1, \dots, p_k \rangle \quad (5.6)$$

One of the simplifying assumptions behind training of the agents is that the behavior of the principle is only influenced by which objects are visible through the field of view of the avatar and the role the agent is currently playing in the institution. We limit

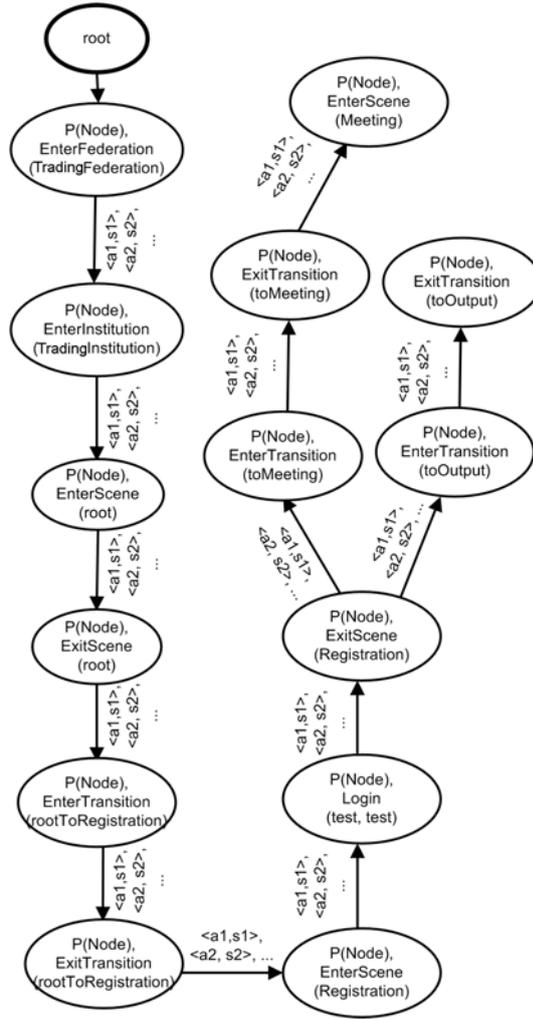


Figure 5.4: A Fragment of the Learning Graph.

the visible items to the objects located in the environments and other avatars. So, the parameters that can be used for learning are recorded in the following form:

$$p_i = \langle role, V_o, V_{av} \rangle \quad (5.7)$$

where *role* is the role currently played by the agent; V_o is the list of currently visible objects; V_{av} is the list of currently visible avatars. The list of the visible objects is represented by the following set:

$$V_o = \{ \langle O_1, D_1 \rangle, \dots, \langle O_m, D_m \rangle \} \quad (5.8)$$

O_j are the objects that the agent is able to see from its current position in the 3D Virtual World. D_j are the distances from the current location of the agent to the centers of mass of these objects.

The list of visible avatars is specified as follows:

$$V_{av} = \{\langle N_1, R_1, CS_1, DAv_1 \rangle, \dots, \langle N_p, R_k, CS_k, DAv_p \rangle\} \quad (5.9)$$

Here, N_k correspond to the names of the avatars visible to the user, R_k are the roles played by those avatars, CS_k are the text values of the cognitive state of these avatars (we decided to include the cognitive state of the visible avatar so that we can illustrate how it can be discovered and utilized) and DAv_k are the distances to them.

Each time an institutional message is executed, the autonomous agent records the attributes it is currently able to observe, creates a new visual level sequence and every 50 Ms adds a new visual level message into it. The recording is stopped once a new institutional message is executed.

5.3.3 Using the Learning Graph

Once a user from the above mentioned scenario completed the bidding task a couple of times and the learning graph was updated accordingly, the human may decide to give the instructions to the agent (e.g. bid for fish). In order to ask the agent to take control over the avatar and achieve some task, we specified a list of textual commands that have to be typed in the chat window. Each of the commands starts with the special keyword “Do:” and the rest of the command presents one of the valid institutional messages (the illocutionary particles are currently ignored). An example of such a command is “Do:Bid(Fish, 10)”. It instructs the agent to find the node marked with Bid(x, y) in the learning graph and backtrack through it to the current node, while executing the sequence of the most probable actions (with the highest $P(Node)$) that lead from the current node to the Bid(Fish,10) node.

The nodes of the learning graph should be seen as internal states of the agent, the arcs determine the mechanism of switching from one state to another and the probability $P(Node)$ determines how likely it is for the agent to change its current state to the state determined by the next node. Once the agent reaches a state $S(Node_i)$ it checks all the other nodes of the learning graph connected to the $Node_i$ and select the node ($Node_k$) with the highest probability. If $Node_k$ with $P(Node_k) > 0$ is found, the agent changes its current state to $S(Node_k)$ by executing the best matching sequence of the visual level actions recorded on the arc that connects $Node_i$ and $Node_k$. If there are no actions recorded on the arc this means that the agent simply has to send the message associated to $Node_k$ to the institutional infrastructure. If there are no nodes connected to $Node_i$ in the learning graph that have $P(Node_i) > 0$ the agent will remain in the state $S(Node_i)$ until it is changed from outside (an example of such a situation is presented in Section 5.5).

To explain how the particular learning graph from Figure 5.4 can be utilized let's imagine that the agent's "state" in the learning graph is " $S(EnterFederation(TradingFederation))$ ". In order for the agent to proceed, it first needs to reach `EnterInstitution(TradingInstitution)` node. To achieve this it has to select and execute one of the visual level action sequences stored on the arc between the current node and the desired node of the learning graph. The attributes of this sequence must match the current situation as close as possible. To do so the agent creates the list of attributes and passes them to the classifier (in the current version of the prototype we use a nearest neighbor classifier [92]). The classifier returns the sequence of the visual level actions and the agent executes each of the actions consecutively. The same procedure continues until the target node is reached.

One of the drawbacks of the approach presented so far is that the action sequences of the visual level are tightly connected to the position of the avatar in relation to the surrounding objects. The classifier will correctly classify the situation where the position of the avatar is slightly different, however, replaying the actions recorded during the training session will have a consequence that the avatar will have to be first moved to the initial position in the recording. If the actual position of the avatar is significantly different to the initial position in the matched example – this movement will not look natural and may brake the overall believability that we try to embrace.

To be able to generalize the learning and avoid the aforementioned drawback we use the following approach. Once the best matching example is detected, we move the agent from its actual position to the nearest point on the selected trajectory and change the body orientation vector accordingly. To achieve this we create an approximation of the average distance that the avatar travels between two landmarks and divide the segment between the current position and the nearest position in the recording into the subsegments with this length. Next, instead of the instant movement from the original position to the target position the avatar will travel through each of the subsegments and only then will start executing one of the recordings. Additionally, the body orientation vector of the avatar is slightly changed throughout the movement between these two landmarks so that it is parallel to the vector of the body orientation of the target landmark at the time the agent arrives there.

5.3.4 Nearest Neighbor Classifier

We previously explained that the classification of the current situation is made using a classifier. The task of the classifier is to cluster the parameters associated with each of the recorded sequences into different classes. So, when an agent is driving the avatar, the current parameters associated with the situation experienced by the agent can be matched

against one of the classes created by the classifier and the best matching recording can be extracted.

For our purposes the requirements we had for selecting an appropriate classifier are as follows. The classifier was required to be able to classify the given examples in real time. The selected method needed to provide reliable classification and at the same time was expected to have a simple representation and description.

The classification method that closely matched our requirements was Nearest Neighbor. The Nearest Neighbor method uses the “lazy learning” strategy where the modeling of training data is done only when it is needed. The simple strategy used in this approach is to find all the examples in the recorded data, which are relatively similar to the presented situation and then decide which one matches the best [202].

The classifier represents each example of the recorded data as a point in n-dimensional space, where n – is the number of attributes used for training. The presented example is also described as a point in n-dimensional space. Then the test example from the training data that matches the presented example the best is identified as having the smallest value of the Euclidean distance between those two points [202].

Below is the detailed description of the k-nearest neighbor algorithm [202]. This algorithm sorts the set of given examples into k clusters.

Algorithm: The k-nearest neighbor classification algorithm.

1. Let k be the number of nearest neighbors and D be the set of training examples.
 2. **for** each test example $z = (x', y')$ **do**
 3. Compute $d(x', x)$, the distance between z and every example, $(x, e) \in D$.
 4. Select $D_z \subset D$, the set of k closest training examples to z .
 5. $y' = \underset{v}{\operatorname{argmax}} \sum_{(x_i, y_i) \in D_z} I(v = y_i)$
 6. **end for**
-

The k in this case corresponds to the number of clusters that are created as the result of the classification. For our purposes, however, we treated each example as being a unique cluster. That’s why we use the 1-nearest neighbor approach (where k is always equal to one).

5.3.5 Detecting the Visibility of Objects

As it was explained before, the parameters that determine a particular case to be classified include visible objects and visible avatars. To be able to use those parameters we needed to determine which objects and avatars are actually visible to the avatar that a human or agent drives.

Most of the Virtual Worlds platforms (including Adobe Atmosphere, which was used for conducting the experiments) do not provide facilities to detect whether an object is

visible by a particular avatar or not. Therefore, we find it being useful to describe the approach taken in our experiments to detect the visibility of objects.

Virtual Worlds offer diverse sets of objects (i.e. trees, doors, desks etc.) and more complex structures that are build out of them. In order to detect the visibility of an object in a field of view of an avatar we approximate a complex object by it's center and the bounding box².

The visibility of every object available in the Virtual World is assumed to be determined by the visibility of its center in the field of view of the avatar. The value of the field of view of an avatar can usually be directly obtained.

Our approach for detecting visibility is illustrated in Figure 5.5.

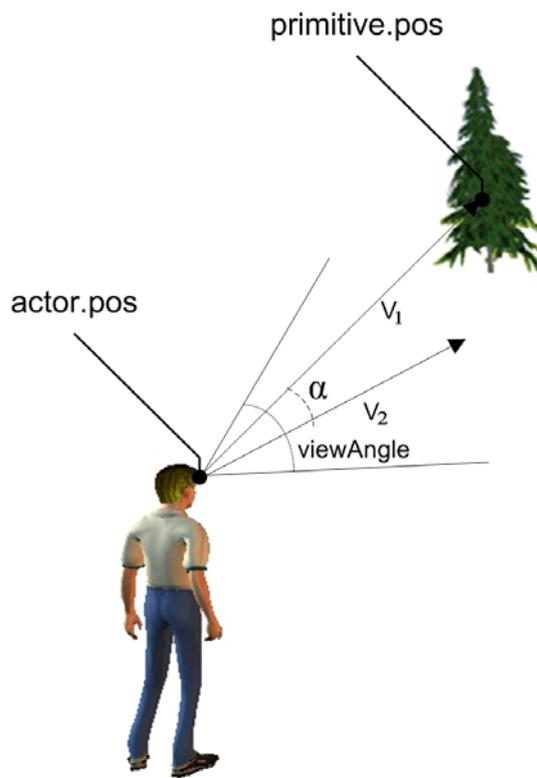


Figure 5.5: Detecting the Visibility.

Here \vec{v}_1 is the vector connecting the view point of the avatar and the center of the object. Given the information provided by the Adobe Atmosphere this vector can be calculated as follows:

$$\vec{v}_1 = actor.\vec{pos} - primitive.\vec{pos} \quad (5.10)$$

²Bounding box is a 3-dimensional box of minimal dimensions with its edges parallel to the coordinate axes that can fully enclose the object.

In this equation $actor.\overrightarrow{pos}$ represents the position of the avatar and $primitive.\overrightarrow{pos}$ corresponds to the position of the object.

Vector \vec{v}_2 corresponds to the direction which the avatar is facing and is expressed by the following equation.

$$\vec{v}_2 = actor.\overrightarrow{facing} \quad (5.11)$$

Having \vec{v}_1 and \vec{v}_2 we can calculate the angle between them using the dot product of two vectors.

$$\alpha = \arccos\left(\frac{\vec{v}_1 \bullet \vec{v}_2}{|\vec{v}_1| \times |\vec{v}_2|}\right) \quad (5.12)$$

In order to determine whether the center of the object is visible or not we simply compare the angle (α) received on the previous step with the viewing angle of the avatar. If $\alpha < viewAngle$ then we consider the object being visible.

In this approach we do not take into account whether the object in question is occluded by another object or not. To detect this within the framework of our assumptions for every object that was marked as visible on the previous step we determine whether it's center is occluded by some other object or not. This is done by applying the algorithm [219] for testing the intersection of the vector \vec{v}_1 with the bounding boxes of each of the objects present in the Virtual World. Once the intersection is positive the object marked visible before is removed from the list and is considered being invisible.

5.4 Experiments

In the series of experiments we trained an autonomous agent to believably act in the Trading Institution from Figure 5.2. This agent was playing the external “Guest” role. It was supposed to learn from a human operator how to execute a given command. The agent was required to recognize the surrounding objects and avatars and to assess the cognitive state of the humans behind the avatars.

For the cognitive state assessment we recorded four different trajectories which then were assigned with the following labels: “Ant”, “Butterfly”, “Grasshopper” and “Fish”. Those prototypical trajectories were used as the basis for comparison. During the recording process, as well as during the actual tests the trajectory of each avatar was compared with each of those four and the label of the one that matched the best was used as the result of comparison.

There many aspects of believable and intelligent agent behavior in Virtual Worlds that desire attention like obstacle avoidance, participating in conversations with humans, crowd simulation, etc. Most of these topics, however, are well studied by other re-

searchers (as shown in Section 5.1) and, therefore, are not investigated in this thesis. Here, for clarity of presentation, we only focused on movements and didn't take into account any text that the humans exchanged during recording and didn't employ any obstacle avoidance strategies.

5.4.1 Experiment 1: Trajectory Recognition

The goal behind this experiment was to verify the proposed method for trajectory recognition and classification of the cognitive state.

The challenge behind the assessment of the cognitive state was not to predict the actual cognitive state of the user, e.g. whether a user really was expressing fish browsing style or was rather a grasshopper. Instead, we wanted to prove that the trajectory recognition method based on the combination of Levenshtein Distance and Euclidean Distance was working properly.

Design of Experiment

The experiment was conducted in the Meeting Room of the Trading Institution, where the buyer agent was demonstrating the trajectory and when it was approaching the assistant agent the current trajectory of the buyer was classified and passed on to the assistant agent. Each buyer was assigned with the corresponding agent that was constantly observing the trajectory of its principal. In order for this agent to assess the cognitive state of the human we recorded 4 prototypical trajectories as displayed in Figure 5.6.

Here each trajectory is shown as a set of landmarks connected by lines. Each of the landmarks corresponds to the position of the avatar at the moment of measurement. The measurement is conducted every 50 Ms. The schematic representation of the Meeting Room is added to show the context under which the trajectories were obtained. Each of the landmarks is projected onto the corresponding position in the meeting room and the label describing the trajectory is shown in the bottom right corner of the room. The figure consists of 4 duplicates of the Meeting Room marked as "a)", "b)", "c)" and "d)". The trajectories on each of those duplicates corresponds to one class of the cognitive state as marked in the picture. The black solid figure present in each copy of the Meeting Room represents the autonomous agent associated with the assistant and the circle around it represents the audibility distance.

The trajectory in Figure 5.6 a) corresponds to the case when a human enters the meeting room already aware of the exhibition and once in the meeting room moves along the wall with a moderate speed checking out the exposition. Here we wanted to present the case of a curious browser, who has no specific interests but simply wants to

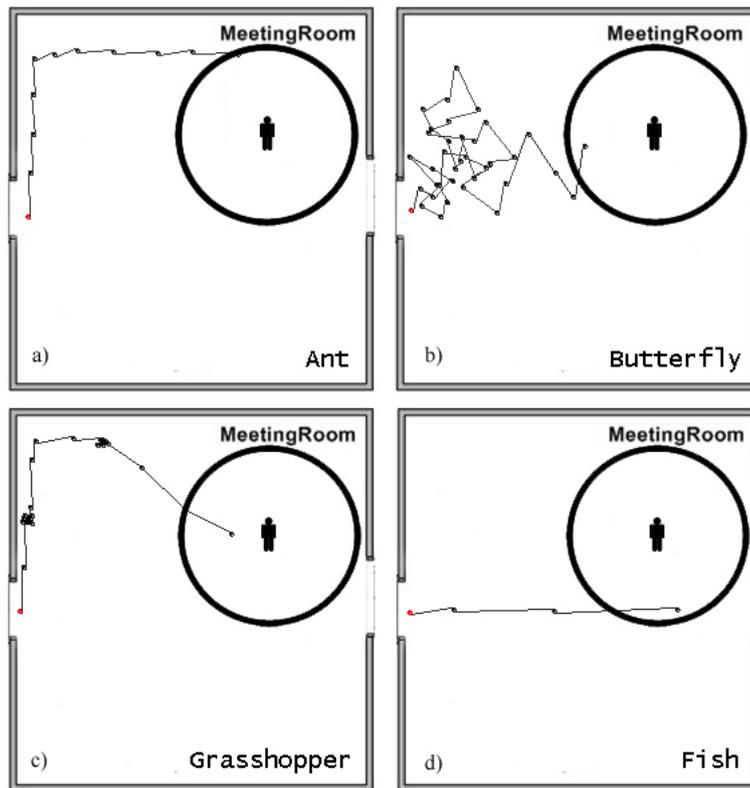


Figure 5.6: Trajectories Used for Training.

get an impression of the exposition in general. This trajectory was associated with the label “Ant”. The key characteristic of this trajectory is that the participant who generated it was moving along the walls with relatively constant speed.

In Figure 5.6 b) we present the case of the participant randomly walking around and demonstrating a high degree of confusion. Such a trajectory is typically generated by novice participants who are not yet quite familiar with controlling their avatars. Label “Butterfly” is assigned to this trajectory. The key characteristic of this movement class is that a participant frequently changes the movement direction and returns to the location close to initial point a number of times.

Figure 5.6 c) illustrates the case of the visitor of the room who has particular interest in some posters and no interest in the others. From the distance between landmarks it is clear that the browsing speed is not constant. In front of two pictures the human stopped for a while and then headed very fast straight towards the exit. This trajectory is labeled as “Grasshopper”. On the picture the groups of landmarks placed closely together do not represent very short movements but are due to the fact that during recording a landmark is added after a constant interval of time even if there was no movement produced.

Finally, Figure 5.6 d) illustrates the case when a participant has no interest in the

exhibition and shortly after entering the Meeting Room quickly walks towards the exit (using the Meeting Room as a corridor) in order to continue with some activities in the Trade Room. The label we use for this behavior is “Fish”. The key characteristics here: the high speed of movement, which can be recognized from the distance between the landmarks and the direction of the trajectory.

As it is clearly seen on the pictures, each of the trajectories was recorded from the moment a human entered the room until the moment the corresponding avatar approached the assistant within the audibility distance. In our experiments once the avatar entered the audibility zone of the assistant agent the recording of the trajectory was finished, the currently recorded sequence of landmarks was compared with each of the prototypical 4 sequences and the trajectory with the lowest Levenshtein Distance was selected as the class describing the cognitive state of the human. Then, the corresponding label associated with this trajectory was sent by the autonomous agent of the buyer to the autonomous agent of the Assistant to inform it about the mood of the buyer. This information was further used by the assistant to decide whether to offer help (the help should be offered for every case except for “Fish”) and what kind of help a participant may require (i.e. the “Butterfly” participant clearly needs a different kind of help than the participant classified as “Ant” or “Grasshopper”).

To validate the trajectory recognition method we conducted a series of experiments with the set of 40 different movement patterns executed in the Meeting Room. The human operator playing the “Buyer” role was told about the distinct characteristics of each of the 4 classes presented earlier and then was asked to produce 10 different movement patterns for each of the classes so that those patterns would match the given descriptions and at the same time would be distinct. Each of the 10 patterns would end with buyer approaching the assistant. The result of the classification of buyer’s trajectory by the assistant agent would be printed in the chat window and will also be visible from the agent’s behavior.

To give an impression about the movement patterns expressed by the operator Figure 5.7 outlines the results of the 16 out of first 40 experiments we conducted. We show only 16 (4 per each of the classes) to avoid overcrowding the picture with unnecessary data. The recorded trajectories exemplify the series of movements which begin when the operator driving the avatar of the “Buyer” agent entered the Meeting Room and end at the moment this avatar entered the audibility zone of the Assistant agent. For presentation purposes the trajectories are projected onto the schematic representation of the meeting room. Each of the recordings is classified into one of the four classes as in Figure 5.6. For simplicity of understanding we placed all of the trajectories having the same class into the same part of the figure, which allows for a better comparison.

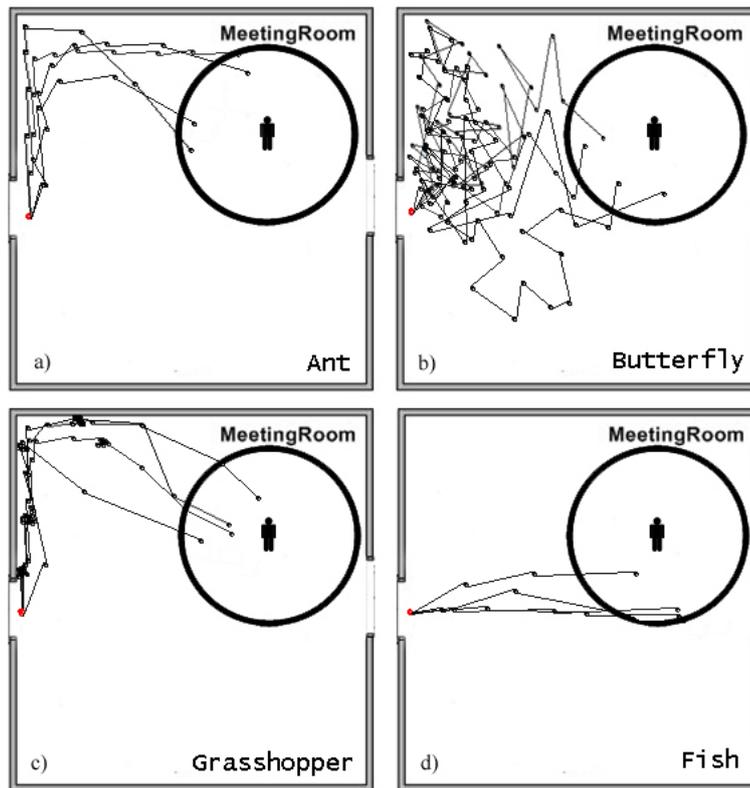


Figure 5.7: Comparing the Trajectories: Experiments.

Discussion of Results

Two of the presented four classes were correctly identified by the system in all the conducted tests. These classes are “Ant” and “Fish”. Out of 10 experiments per each of those classes 10 were recognized correctly.

The “Butterfly” trajectory was also detected very accurately with the precision of 80%, where 8 out of 10 generated examples were recognized correctly. The motion in vertical direction in one of the misclassified trajectories was very low, which became the main reason why this pattern was classified as “Fish”. In another misclassified example the operator approached the initial position only twice (while in the prototypical trajectory it happened 4 times) and, therefore, this pattern was also classified as “Fish”.

The recognition of the “Grasshopper” trajectory showed the worst precision of only 60%. It proved to be too similar to the “Ant” behavior with all of the 4 misclassified examples being recognized as “Ant”.

To understand the reasons for misclassification we conducted 50 more experiments with the “Grasshopper” and “Butterfly” classes. These experiments showed that for “Grasshopper” class the irregularities in speed (expressed by different distance between landmarks in “Grasshopper” class) had higher importance for the classification than the

fact of a participant stopping. The trajectories with similar number of stops as in the “Grasshopper” class but with relatively constant movement speed in between stops had a very high risk of being classified as “Ant”, while the movement patterns that expressed clear speed irregularities were more likely to be classified as “Grasshopper”.

For the “Butterfly” class our initial hypothesis that the amplitude of the vertical movement as well as the number of returns to the initial position are the key factors in misclassification proved to be correct. The misclassified trajectories had the lowest value of the Levenshtein distance with “Fish” class as the generated examples were too distinct and, therefore, the example with the fewest number of landmarks had the lowest distance value.

5.4.2 Experiment 2: Training the External Agent

The goal of this experiment was to train an external agent by a human operator to execute a delegated task while expressing believable movements. This goal consists of 2 subgoals:

1. to prove that it is possible to implicitly train the agent to execute a desired task
2. to prove that the agent is capable of assessing the parameters present in the environment and selecting the learned pattern that matches those parameters in the best possible way.

Design of Experiment

To train the external agent we let a human operator control the avatar playing the “Guest” role and completed a number of recording sessions, staging different scenarios. Without the loss of generality these scenarios are demonstrated by 10 recordings. The training started in the garden, where one human operator was facing the avatar towards different objects and another operator was controlling the Receptionist, moving it to different positions and changing its cognitive state. As the result of the training we generated 10 trajectories, which we tried to make easily distinguishable given the observed attributes (i.e if the avatar was facing a pine tree in the moment of recording, the trajectory included walking around this tree before entering the Registration Room).

Figure 5.8 gives an impression of how the training was conducted. It shows fragments of the 4 different trajectories generated by the “Guest” avatar ($S_1 \dots S_4$). The arrows marked with $S_1 \dots S_4$ correspond to the direction of view of the avatar at the moment when the recording was initiated. The dotted colored lines represent the actual trajectories (where color matches the color of the corresponding arrow $S_1 \dots S_4$). Trajectories S_1 and

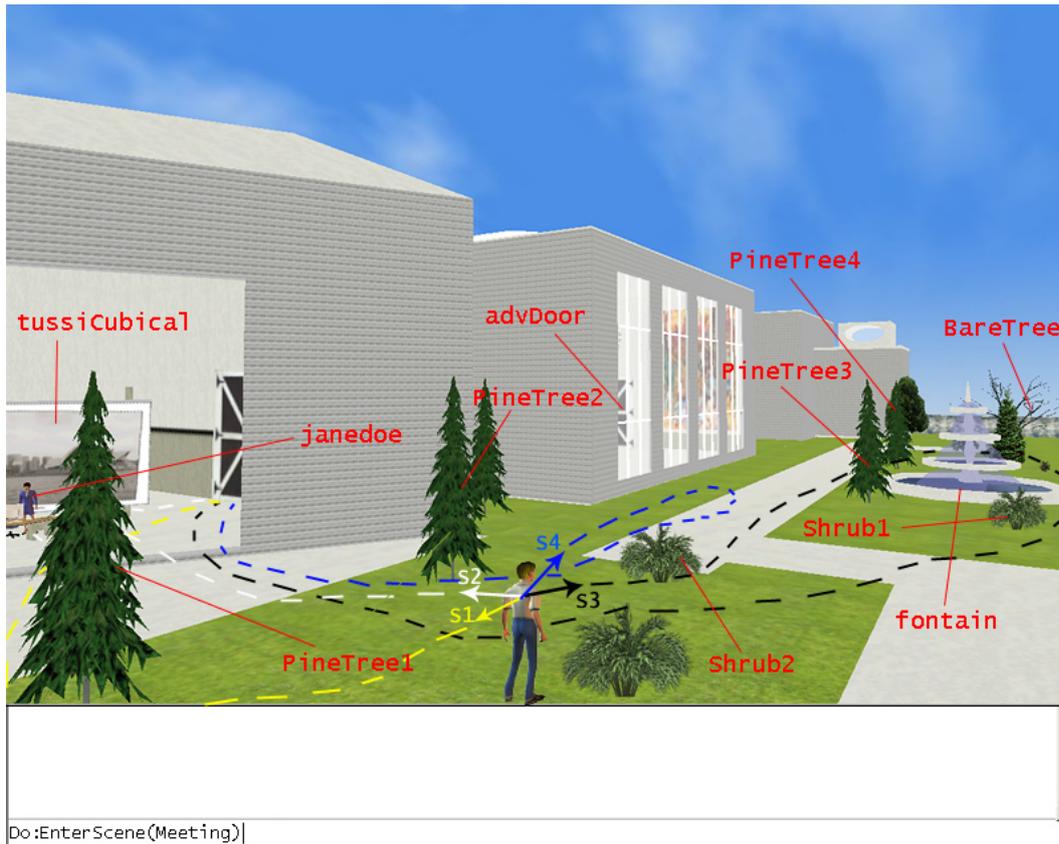


Figure 5.8: Training the Guest Agent in the Garden.

S_3 represent the “walkabout” case, where the avatar was first walking around the object it was directly facing and only then entered the institutional building. The S_2 trajectory is the “direct walk”, when the avatar was entering the institution and walking straight to the reception desk. The S_4 case is the “philosopher’s walk”, showing a participant that for a short period of time walks towards the direction it is facing, but then recalls the original goal, sharply turns around and continues to pursue the goal.

The recording of the learning graph was started by the operator through pressing the “F9” key on the keyboard. The role of the trained agent (which in the garden was always Guest), the location of the Receptionist and its cognitive state together with the objects located in the field of view of the avatar at the moment of recording played the role of the attributes used for learning. In Figure 5.8 the possible objects are marked red and include “Pinetree”, “regDoor”, “tussiCubical”, “advDoor” and “Fontain”). The avatar of the Receptionist is showed in the figure as “janedoe”.

For the purpose of our experiments there was no need to train the agent to bid in the Trade Room as the main focus was on verifying whether the agent’s movements in the garden will appear believable and human-like. The bidding behavior itself was not

Table 5.2: Attributes Used during the Training Session.

parameterID	parameterName	possibleValues
p_1	janedoeRole	{recep,null,guest}
p_2	DISTadvDoor	numeric
p_3	DISTPineTree4	numeric
p_4	DISTBareTree	numeric
p_5	SEEfontain	{y,n}
p_6	DISTtussiCubical	numeric
p_7	DISTPineTree3	numeric
p_8	DISTPineTree2	numeric
p_9	DISTPineTree1	numeric
p_{10}	SEEtussiCubical	{y,n}
p_{11}	SEEBareTree	{y,n}
p_{12}	SEEadvDoor	{y,n}
p_{13}	SEEshrub3	{y,n}
p_{14}	SEEshrub2	{y,n}
p_{15}	DISTfontain	numeric
p_{16}	SEEshrub1	{y,n}
p_{17}	DISTshrub3	numeric
p_{18}	SEEPineTree4	{y,n}
p_{19}	DISTshrub2	numeric
p_{20}	SEEPineTree3	{y,n}
p_{21}	DISTshrub1	numeric
p_{22}	SEEPineTree2	{y,n}
p_{23}	SEEPineTree1	{y,n}
p_{24}	SEEjanedoe	{y,n}
p_{25}	DISTjanedoe	numeric
p_{26}	janedoeCS	{0,1,2,3,4}

important at this stage. Therefore, in all of the 10 recording sessions instead of bidding the agent was trained to enter the Meeting Room.

The learning graph that was created as the result of training looked similar to the one presented in Figure 5.4. Table 5.2 presents the list of all the attributes stored in the learning graph on the arc between EnterFederation(TradingFederation) and EnterInstitution(TradingInstitution) nodes.

Here the attributes, having names beginning with “SEE” correspond to the objects or avatars that were appearing in the field of view of the avatar at the moment of recording. The attributes with the name starting with DIST correspond to the distance measure between the avatar of the user and the center of mass of the object. The distance to the objects that are not currently visible were considered to be equal to zero.

The attributes “janedoeRole” and “janedoeCS” define the values for the role and the cognitive state of the receptionist agent “janedoe” present in the institution. The cognitive state is represented by the following values: “1”, “2”, “3” and “4”, which correspond to “Ant”, “Grasshopper”, “Butterfly” and “Fish”. In cases when the Receptionist is not visible in the field of view of the Guest avatar, the values of janedoeRole and janedoeCS were null and 0 accordingly.

Table 5.3 presents the training data, which was collected during the ten training sessions. The attribute IDs in this table correspond to the IDs in the Table 5.2, the “S” column stores the acronyms of the sequences of actions of the visual level of execution.

The first 4 of those sequences correspond to the trajectories $S_1 \dots S_4$ outlined in Figure 5.8.

Table 5.3: A Fragment of Data Used in Recording.

Nr	p_1	p_2	p_3	p_4	p_5	p_6	p_7	p_8	p_9	p_{10}	p_{11}	p_{12}	p_{13}	p_{14}	p_{15}	p_{16}	p_{17}	p_{18}	p_{19}	p_{20}	p_{21}	p_{22}	p_{23}	p_{24}	p_{25}	p_{26}	S		
1	null	0	0	0	n	0	0	0	31	n	n	n	n	n	0	n	0	n	0	n	0	n	y	n	0	0	S_1		
2	recep	0	0	0	n	55	0	0	39	y	n	n	n	n	0	n	0	n	0	n	0	n	y	y	67	1	S_2		
3	null	0	0	0	y	0	61	0	0	n	n	n	n	n	96	y	0	n	0	y	70	n	n	n	0	0	S_3		
4	null	74	0	0	y	0	61	0	0	n	n	y	n	n	95	y	0	n	0	y	67	n	n	n	0	0	S_4		
5	recep	0	0	0	n	56	0	0	0	y	n	n	n	n	0	n	0	n	0	n	0	n	n	y	68	2	S_5		
6	null	0	0	0	n	0	0	43	77	n	n	n	n	y	0	n	0	n	48	n	0	y	y	n	0	0	S_6		
7	guest	0	0	0	y	0	0	0	0	n	n	n	n	n	96	y	0	n	0	n	0	n	69	n	n	y	24	3	S_7
8	null	0	0	24	n	0	0	0	0	n	y	n	y	n	0	n	42	n	0	n	0	n	n	n	0	0	S_8		
9	null	0	0	0	n	0	0	0	0	n	n	n	n	n	0	n	0	n	0	n	0	n	n	n	0	0	S_9		
10	guest	0	41	0	y	0	96	0	0	n	n	n	n	n	69	n	0	y	0	y	0	n	n	y	38	4	S_{10}		

After the recording was completed, we conducted a series of 20 tests. The aim of the tests was to see whether the trajectory representing the cognitive state of the human can be properly recognized, to check whether the agent will act believable and reasonable in accordance with the observed attributes and to determine the limitations of the taken approach.

In order to verify the learning capabilities of the agents each of the 20 tests was conducted as follows. Two operators entered the Virtual World as two different avatars: “janedoe”, playing the “Receptionist” role and “agent0”, playing the “Guest” role. The “janedoe” avatar was first driven around the garden by the operator to let the system recognize the cognitive state of the user. After producing the trajectory that had a potential to be recognized as one of the four values of the cognitive state, we staged a situation where the Receptionist was located at some position and the Guest avatar was facing a selected direction. Next step was to instruct the Guest agent to leave the garden, enter the institution, walk into the Registration Room, exit it and then walk through the next transition to the Meeting Room. This instruction was given in a form of a text command typed into the chat window in the following way “Do:EnterScene(MeetingRoom)”.

After receiving this command the Guest agent was searching for the right sequence of the normative level actions, which in the given case were: EnterInstitution(TradingInstitution), EnterScene(root), ExitScene(root), EnterTransition(rootToRegistration), ExitTransition(rootToRegistration), EnterScene(Registration), Login(test, test) ExitScene(Registration), EnterTransition(toMeeting), ExitTransition(toMeeting), EnterScene(Meeting). To make transitions between those actions the agent needed to launch the appropriate sequence of the visual level actions, stored on the arcs of the learning graph. The appropriate sequence of visual level actions was selected from the recorded data by the classifier. The classifier was given the list of currently observed attributes as the input and as the output it returned the sequence that was supposed to fit those attributes the best.

The 20 conducted experiments consisted of different approximations of the situations recorded in the garden. During the experiments the Guest avatar was moving to differ-

ent positions around the garden. The Receptionist avatar was also randomly moving in the garden, so that it was either visible or invisible in the field of view of the Guest. Every test was finishing by the human operator behind the Guest avatar typing in the “Do:EnterScene(MeetingRoom)” command in the chat field. The visible objects were recorded and the best matching case in the Table 5.3 was identified. The executed movement pattern was then observed to determine whether it will correspond to the trajectory recorded for the identified case.

To assess the believability of the agent’s movement another human operator (test subject) was asked to enter the Virtual World as an external observer and express his opinion regarding the believability of the avatar movement of the Guest agent in each of the experiments. The test subject was also explained how the agent was supposed to react to different parameters and was asked to assess whether the movement of the agent would go inline with these descriptions. During the assessment the test subject marked the number of the experiment and assigned the believability value (believable/not believable) and the movement expectation value (as expected/not as expected) to each of them.

To have a precise understanding of the process of the classification, the attributes that the agent was able to observe and the textual value assigned to the resulting sequence that was selected as the classification result were stored in a text (log) file.

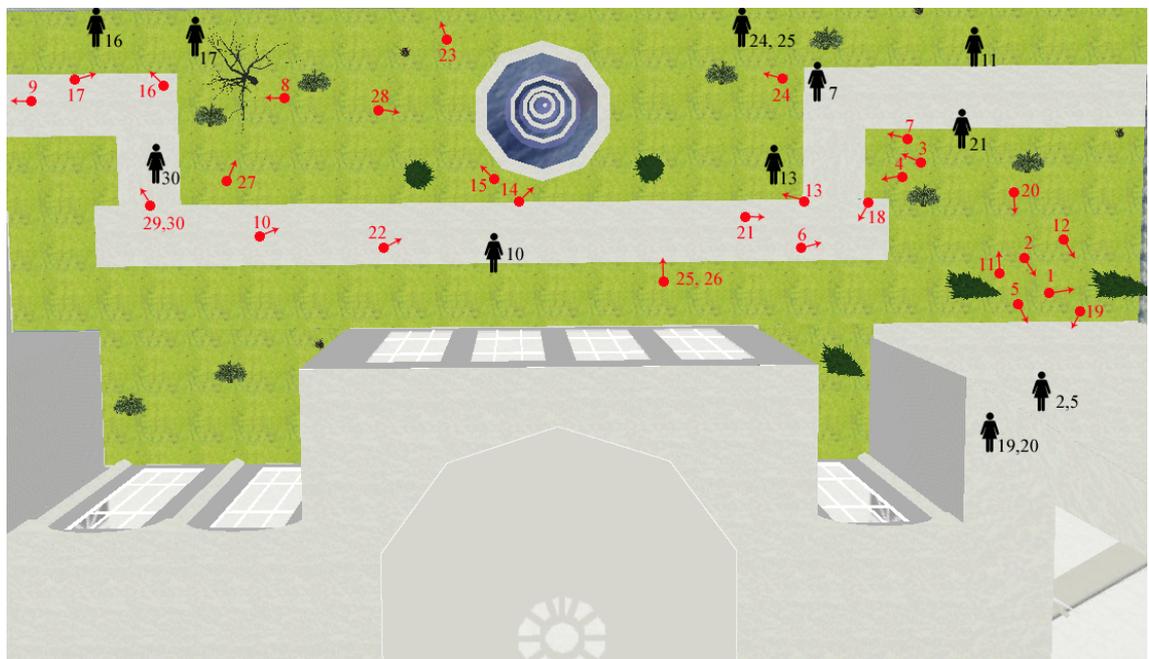


Figure 5.9: Experiments: Avatar Positions and Eye Direction

Discussion of Results

Table 5.4 presents the data reconstructed from the recorded log file after 20 experiments and Figure 5.9 shows the eye direction of the Guest and the positions of both avatars that participated in them. Solid red dots marked with the number of the experiment in the figure correspond to the positions of the Guest and red arrows represent guest’s eye direction. Where it was visible to the Guest, the black female figure (also marked with the experiment number) shows the positions of the Receptionist avatar. The experiment numbers correspond to the ones specified in the “Nr” columns in Table 5.3 and Table 5.4. The numbers 1–10 are the initial recordings and 11–30 represent the conducted experiments. The “S” column in Table 5.4 outlines the acronyms of the action sequences (same as used in Table 5.3) executed by the agent as the result of classification.

In all of the 20 experiments the agent behavior was considered believable and human-like by the test subject. The sequence selected by the classifier was fitting the situation very well and has also received 20 out of 20 score from the test subject. Our initial assumption that the visibility of the Receptionist is valued much higher than the visibility of an additional object was confirmed by the experiments. This is due to the fact that the visibility of the agent immediately changes 4 attributes: SEEjanedoe, DISTjanedoe, janedoeRole and janedoeCS, while the visibility of an object would only change two attributes.

Table 5.4: A Fragment of Data Used in the Experiments.

Nr	p1	p2	p3	p4	p5	p6	p7	p8	p9	p10	p11	p12	p13	p14	p15	p16	p17	p18	p19	p20	p21	p22	p23	p24	p25	p26	S	
11	guest	0	0	0	n	0	0	0	0	n	n	n	n	y	0	y	0	n	40	n	91	n	n	y	36	1	S ₇	
12	null	0	0	0	n	59	0	0	36	y	n	n	n	n	0	n	0	n	0	n	0	n	y	n	0	0	S ₁	
13	guest	0	36	0	y	0	91	0	0	n	n	n	n	n	75	y	0	y	0	y	63	n	n	y	17	2	S ₁₀	
14	null	0	0	0	y	0	0	0	0	n	n	n	n	n	36	y	0	n	0	n	82	n	n	n	0	0	S ₃	
15	null	0	0	77	y	0	0	0	0	n	y	n	y	n	30	n	94	n	0	n	0	n	n	n	0	0	S ₈	
16	guest	0	0	0	n	0	0	0	0	n	n	n	n	n	0	n	0	n	0	n	0	n	n	y	25	4	S ₉	
17	guest	0	0	86	n	0	0	0	0	n	y	n	y	n	0	n	77	n	0	n	0	n	n	y	34	1	S ₈	
18	null	65	0	0	y	0	51	0	0	n	n	y	n	n	87	n	0	n	0	y	0	n	n	n	0	0	S ₄	
19	recep	0	0	0	n	41	0	0	0	y	n	n	n	n	0	n	0	n	0	n	0	n	n	y	51	2	S ₅	
20	recep	0	0	0	n	72	0	41	40	y	n	n	n	n	0	n	0	n	0	n	0	y	y	y	78	3	S ₂	
21	guest	0	0	0	n	0	0	63	95	n	n	n	n	y	0	n	0	n	42	n	0	y	y	y	18	0	S ₆	
22	null	0	0	0	y	0	18	0	0	n	n	n	n	n	50	y	0	n	0	y	99	n	n	n	0	0	S ₃	
23	null	0	0	0	n	0	0	0	0	n	n	n	n	n	0	n	0	n	0	n	0	n	n	n	0	0	S ₉	
24	guest	0	95	0	y	0	0	0	0	n	n	n	n	n	71	y	0	y	0	n	41	n	n	y	14	3	S ₇	
25	guest	0	0	0	y	0	26	0	0	n	n	n	n	y	59	y	0	n	82	y	84	n	n	n	y	71	1	S ₇
26	null	0	0	0	y	0	26	0	0	n	n	n	n	y	59	y	0	n	82	y	84	n	n	n	0	0	S ₃	
27	null	0	0	43	n	0	0	0	0	n	y	n	y	n	0	n	60	n	0	n	0	n	n	n	0	0	S ₈	
28	null	0	45	0	y	0	96	0	0	n	n	n	n	n	58	n	0	y	0	y	0	n	n	n	0	0	S ₁₀	
29	null	0	0	0	n	0	0	0	0	n	n	n	y	n	0	n	63	n	0	n	0	n	n	n	0	0	S ₉	
30	guest	0	0	0	n	0	0	0	0	n	n	n	y	n	0	n	63	n	0	n	0	n	n	y	14	4	S ₉	

5.5 Training the Internal Agent

We showed how the learning graph can be used for teaching an external agent to express pure reactive behavior and conducted the experiments to demonstrate the selected implementation method. In this section we explain how the internal agent can be trained using this approach.

The goal behind creating a conceptual separation between the training of internal and external agents is to demonstrate that the agents can be trained to complete a task which is beyond immediate reaction on an instruction. The training of the external agent was so far only focused on executing a particular command, while here we show how the agent can be left alone to act in the environment without any commands requested by the principal. Its actions in this case will be triggered as the result of the agent sensing the changes in the environment. We illustrate this approach on the scenario, where the assistant agent in the Meeting Room can be trained to assist the buyers approaching it. We will demonstrate how the assistant agent can learn to select an appropriate assistance strategy depending on the mood of the buyers.

To be able to replace the artist by the assistant agent and to facilitate the training, the specification of the Trading Institution had to be modified. Figure 5.10 shows the change of the protocol in the Meeting Room that had to be made. The assistant agent plays the “RoomManager” role in this institution.

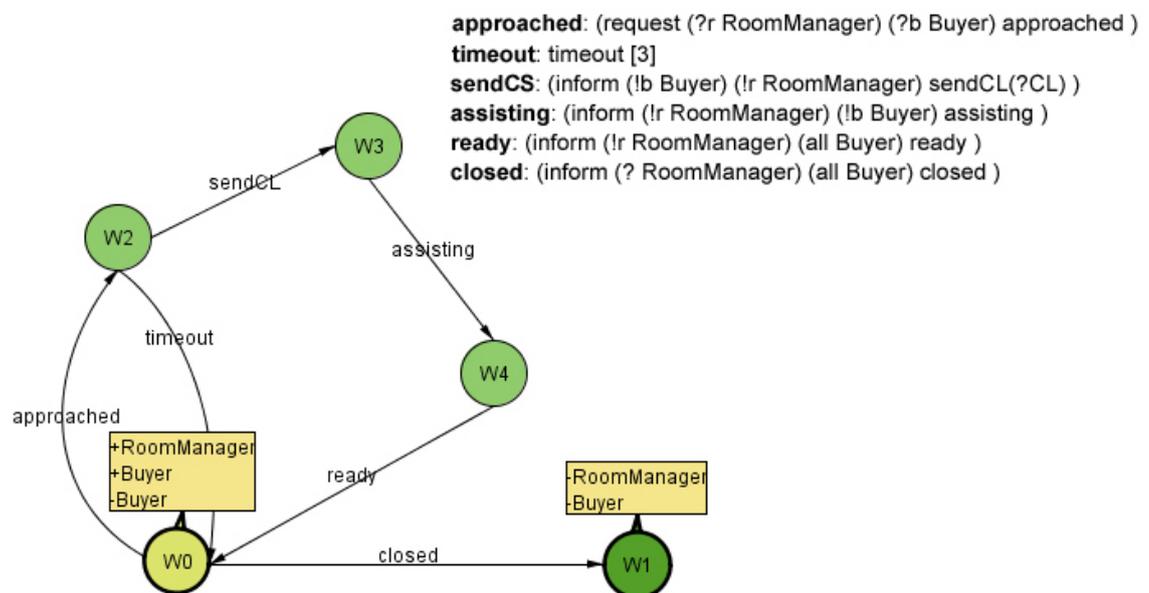


Figure 5.10: The Scene Protocol for Meeting Room.

Now the protocol consists of 5 states W0, W1, W2, W3, W4. The right side of the figure shows the illocutions that trigger the transitions between those states. The state

W0 is reached when the room is created. While the corresponding scene is in this state the participants with roles RoomManager and Buyer can enter the room and only the Buyers are allowed to leave the room.

When the RoomManager decides to close the room it uses the “closed” illocution to inform all the participants that the room is closed and changes the state of the corresponding scene to the final state W1, where all the Buyers will be forced to leave the room.

Another possibility to change the state of the scene from W0 is sending by a RoomManager an “approached” illocution to a Buyer. This should happen as the result of the RoomManager sensing that it has been approached by an avatar playing the “Buyer” role. This illocution is marked as “request”, meaning that the Buyer is expected to send something in return. Sending the “approached” illocution will change the state of the scene to W2.

In state W2 the Buyer may decide to disclose its cognitive state to the RoomManager. This can be done using the “sendCS” illocution. Executing this illocution will change the state of the scene to W3.

The “timeout” illocution is fired after 3 seconds if there is no response from the Buyer and will return the scene into the initial state, where the RoomManager will wait for another Buyer to approach it.

While in state W3, the RoomManager will inform the Buyer that it started the assisting by executing the “assisting” illocution and changing the state of the scene to W4.

Once the RoomManager finished with assisting the Buyer it should send the “ready” illocution informing all the Buyers that it is ready to deal with new assistance requests.

There are two possible strategies that can be employed depending on the state of the Action/Message table associated with the corresponding Virtual Institution. The choice of the strategy is driven by the following two conditions:

1. There is a record in the local Action/Message Table and a corresponding script for the action.
2. There is no record in the table, which means that the agent can be trained for doing the action.

All the messages that have corresponding actions assigned to them in the Action/Message Table will result in the action (a script with the given name) being automatically executed once the corresponding message is received by the agent. All the other messages will be initially ignored, but the agent will use them as the basis for learning.

In the example presented above the scripts for all the messages apart from “assisting” are present in the action/message table. So, the training of the agent only starts when the

scene is in the state W3 and finishes in the state W4.

Figure 5.11 presents the fragment of the learning graph of the Assistant agent. As in the case of the external agent, every normative level message that is executed by an agent as a result of a visual level action of the corresponding avatar will result in adding a node in the learning graph. The number of times the message was executed will influence the associated probability ($P(Node)$).

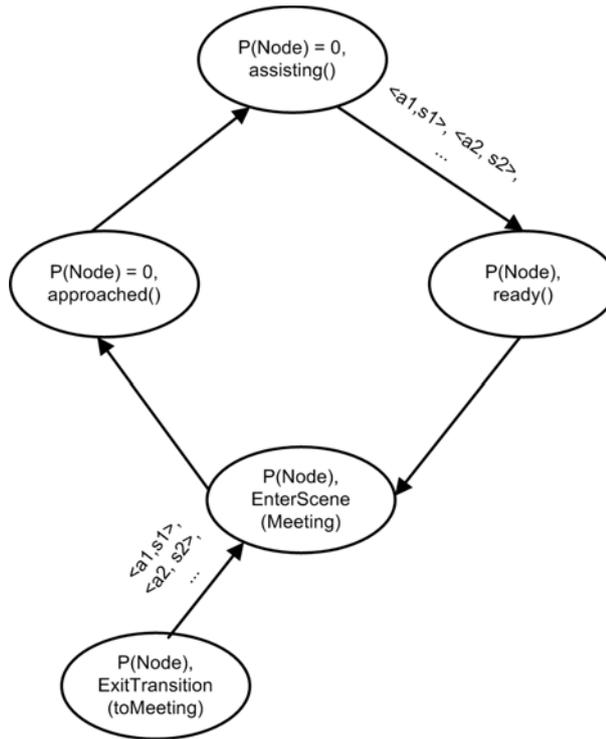


Figure 5.11: The Fragment of Assistant's Learning Graph.

The main difference between this graph and the previously presented learning graph of the Guest agent is that there are some nodes that have $P(Node)$ set to “0” present in Figure 5.11. This value is assigned to all messages that have a corresponding record in the action/message table. It is done to prevent the agent from immediately “moving” through such nodes. The agent will not record the visual level actions on the incoming arcs connected to such nodes and will be waiting for the corresponding message to be sent on its behalf (as the result of the avatar action). Only then can the agent proceed with the next graph node.

Another characteristic feature of this graph is the loop created by connecting the “ready()” and “EnterScene(Meeting)” nodes. To be able to make this loop the agent had to access the .xml file with the specification of the institution and recognize it in the state protocol of the Meeting scene.

Having such a learning graph the agent will act as follows. After entering the Meeting Room it will be waiting for the action corresponding to “approached()” message to be executed. This action will be fired when a Buyer enters the audibility region of the assistant. The “approached()” message will be sent to the Buyer representative agent as the result of this action, informing the Buyer that it has approached the Assistant. If not prohibited by the principle, the Buyer representative agent will send the label of its cognitive state to the Assistant, which will trigger the “assisting()” message.

There is no corresponding action present in the action/message table for the “assisting()” message, so to “move” to the next graph node the agent will have to select one of the prerecorded visual level sequences. The classifier on this stage will receive the label of the cognitive state as one of the attributes.

Once the assistance is finished the Assistant agent will execute the “ready()” message, which will return it to the state where it can start assisting other Buyers.

5.6 Summary

The deployment architecture proposed in the previous chapter suggests that behind each avatar there are always two entities: a human and an autonomous agent. Due to constant presence of autonomous agents and similar embodiment with humans the agents can employ implicit training mechanisms to learn human-like behavior from the humans.

Implicit training is the training of an autonomous agent from interaction with a human when the human is performing a routine job (i.e. answering customer enquiries). In contrast to explicit training, no explicit teaching efforts are required from the human.

Virtual Institutions facilitate implicit training providing context and background knowledge to agents through institutional specification, while 3D visualization gives them full observation of the human actions.

Our view is that the agent and the human co-operate in the solution of the tasks the human has to deal with. We want to permit that either the human takes full control over the avatar or that the autonomous agent is in full charge of the decision making process. While the human is controlling the avatar the agent constantly observes the actions of the principal and stores them inside the learning graph. The nodes of this graph represent the actions that the human executed in the Normative Control Layer. The arcs connecting any two nodes ($Node_k$, $Node_{k+1}$) in the graph store the sequences of the Visual Interaction Layer actions that the human executed in order two change the state of the Normative Control layer from W_i – the state resulted by the action stored in $Node_k$, to W_{i+1} – the state resulted by the action stored in $Node_{k+1}$.

It has been showed how an autonomous agent can utilize the information collected in

the learning graph for dealing with instructions received from a human and how proactive behavior can be emulated through the employment of this graph.

We have also illustrated the benefits of 3D visualization in terms of analyzing some aspects (mood) of the cognitive state of the user on the basis of the trajectory of the avatar movement. The proposed solution for trajectory analysis is based on the Levenshtein Distance algorithm.

Another contribution of this chapter is the concept of Distributed User Modeling. It is suggested that each of the agents should constantly collect the information relevant to the user profile of the corresponding principle and may wish to share some parts of the collected profile with other agents in order to facilitate the cooperation with them. This idea has been illustrated on the case of a Buyer representative agent supplying the Assistant agent with its “shopping mood”, so that an appropriate assisting strategy can be selected.

Various aspects of implicit training and user modeling are tested through a set of experiments conducted within a framework of the developed Trading Institution prototype.

The results of the experiments show that it is indeed possible to train an autonomous agent to produce believable human-like movements inside a Virtual Institution. The agent was also capable of recognizing the cognitive state of the agents surrounding it. We deliberately had no intention to conduct a statistically significant number of experiments so that the presented method is fully verified. In its present form our implementation suffers from generalization problems and it can not be treated as a fully-fledged solution. The goal of our work presented in this section was to provide the proof of concept for implicit training rather than to offer a complete software solution utilizing this training method. In Chapter 7, Section 7.1.7 we discuss how the implementation can be improved and also give details of how the evaluation of believability should be conducted once the implementation is finished.

Chapter 6

Virtual Institutions in E-Commerce

In the previous chapters we presented the metaphor of Virtual Institutions and outlined the technological aspects behind it. This new metaphor can be applicable to a wide range of possible domains and not only to the domain of E-Commerce as it was initially intended. In this chapter, however, we return to the problem of building believable immersive environments for commercial activities in the Internet. We demonstrate how this problem can be addressed with Virtual Institutions by illustrating the application of the new technology to the domain of E-Commerce.

Some shortcomings of the existing E-Commerce solutions were briefly outlined in Chapter 1. Here, in Section 6.1.1 we present a detailed overview of all the drawbacks we have identified. In Section 6.1.2 we discuss the benefits of using Virtual Institutions to address these drawbacks. Further, in Section 6.2 we illustrate the application of Virtual Institutions to E-Tourism (a subdomain of E-Commerce where the identified drawbacks of E-Commerce are the most evident).

6.1 Issues in Existing E-Commerce Solutions and the Potential of Virtual Institutions in Addressing Them

Electronic Commerce has radically changed the way business is conducted around the world. It made possible to vastly increase the interactivity in the economy, helping small businesses and households reaching out to the world at large. It has provided people with the means to do their business in a fast and convenient way, abolishing distances and altering the concept of community. Despite all the benefits there is a number of drawbacks that E-Commerce in its current form is unable to solve. In this section we identify some of these drawbacks and illustrate how the application of Virtual Institutions can help in addressing them.

6.1.1 Drawbacks of Existing E-Commerce Solutions

Through the process of putting the shopping environments on the World Wide Web some of the useful and convenient aspects of the brick and mortar shops have disappeared. An extensive research on commercial activities in brick and mortar stores was conducted by [206]. Based on this work and some other sources we identified the list of important features of brick and mortar stores that present E-Commerce solutions do not include.

Insufficient Product Presentation

One of the most significant drawbacks of E-Commerce is the product presentation. The most popular product presentation method that existing E-Commerce solutions provide is to experience products through 2-dimensional images and textual descriptions.

In contrast, in brick and mortar stores product presentation is 3-dimensional. A product can be experienced from different angles, in close proximity, with a focus on small details or from a distance (giving a more global understanding of its features). A product can be seen under different lighting conditions to reduce the uncertainty about its colors. Additionally, products can be touched, smelled and interacted with.

The interaction with a product is very important. It helps to increase the customer learning about the product or to build customers trust and confidence towards the purchase [56]. While for some products it is not particularly important to be able to interact with them, for other products the interaction is absolutely necessary. For example, touching a fabric will tell more about bed linen than pictures on the web. Pressing buttons on the keyboard will give better understanding about its quality than textual descriptions.

Another presentational drawback of E-Commerce is its inability to have personalized experience with the product [66]. In brick and mortar stores it is possible to try the clothes on and see whether they fit. A customer may hold a digital camera and experience its features through direct interaction, which can be more useful than reading about its technical characteristics. It is possible to see whether a bag a woman is about to purchase is a good match with her shoes or not.

Poor Support of Social Interactions

Individuals today are the product of a particularly mobile and entrepreneurial society. As a result, in everyday business activities, including E-Commerce, an individual is socially constituted and socially situated [194]. Although, researchers have pointed out that social needs of customers play a crucial role and are of great importance in E-Commerce applications [162], these needs are mostly neglected in nowadays systems, regardless of the business model that they implement (whether it is B2C, C2B or C2C). Most system

analysts perceive E-Commerce from a purely technical viewpoint without trying to establish the social and business norms that companies and consumers abide by. A truly functional E-Commerce system that supports business activities has to take care of social issues behind these activities [93].

The social side of commerce is quite an important topic. Socializing with fellow shoppers, clerks and store owners not only proves to be an important purchase facilitator but sometimes is also one of the key goals of the shopping itself [206]. Some retailers even believe that social interactions serve the same function as the “sweet smell of baking cakes” does in a pastry shop. Both evoke images of comfort, warmth, happiness and maybe even trust. Other researchers are convinced that the absence of social cues in online auctions may limit bidders’ ability to analyze behavior of other bidders and make improved decisions [168]. Unfortunately, social needs of the customers are mostly neglected in contemporary E-Commerce systems.

Some attempts to bring the notion of communities into E-Commerce systems were made. There are commercial solutions (LinkedIn.com, Ryze.com, Itsnotwhatyouknow.com) that provide business equivalents of the popular web sites facebook.com and friendster.com, where people try to find new contacts with the help of other people they know and trust. The aforementioned web sites, however, are focused only on social aspects while the participating in business interactions inside the community is not directly supported. Despite the fact that the question “whom do you know?” appears to be a key characteristic of a businessman’s success in the real world [50], the functionality of finding reliable business partners is not yet integrated into E-Commerce sites.

Another attempt to design a form-based environment for fostering social interactions in electronic market places was done by [78]. Their environment incorporates a novel, spatially-organized and interactive site map which provides visibility of people, activities, and incorporates mechanisms for social interactions. However, this interface is too overloaded, does not provide an immersive experience and is difficult to navigate.

Poor Support of Impulsive Decisions

The Point-of-Purchase Advertising Institute’s research shows that in traditional shopping environments 70% of buying decisions are made directly in the store [9]. Hence a good E-Commerce system should also be able to trigger such impulsive purchase decisions. With the increasing consumption around the world the variety of products being offered to customers grows exponentially. Moreover, every day it becomes harder and harder to successfully utilize marketing techniques for selling these products. There are so many information sources available to customers nowadays that it gets harder and harder to reach them [206]. Because of that, the importance of in-store information and expe-

rience about the products becomes much higher than the importance of brand loyalty or advertisement. The stores become big 3-dimensional advertisements for themselves. Signage, shelf position, display space and special fixtures all make it either likely or less likely that a shopper will buy a particular item (or any of them) [206].

Despite this, most E-Commerce environments are designed with the assumption that customers' buying style (or behavior) is rational, without much consideration of the needs of customers showing an impulsive style of buying [31, 45]. The product presentation in traditional E-commerce is done through form-based web sites, where the possibilities to market the product are quite limited and presentation facilities are rather poor in regards to browsing through a large number of products [31].

One of the brightest examples of inability of form-based E-Commerce solutions to efficiently support impulsive decisions are online supermarkets (i.e. Coles Online¹). With thousands of different products available in stock the user is faced with an option to browse through long lists and select the products from them. Such an approach clearly limits the possibilities to select an unknown brand or pick an item that was not present on the shopping list [169]. In contrast, the product presentation in brick and mortar stores is much richer. Locating items on the shelves of a supermarket makes it possible to present hundreds of different options to a user at any given time.

Another example of inability of present E-Commerce solutions to satisfy the needs of impulsive buyers is a situation when the product requirements are unclear. Imagine a situation which very often happens to female shoppers, when a lady requires something red to match her new shoes. It may be either a bag or a hat or something else. Going online with such vague requirements will most probably bring such a shopper nowhere, while browsing through a brick and mortar store would most definitely result in a success.

Limited Advertisement Possibilities

Each E-Commerce activity has two parties: buyers and sellers. To make a successful E-Commerce application both parties have to be satisfied! This implies that e.g. advertisement has to be an integral part of an E-Commerce experience.

In its current form the advertisement in E-Commerce is usually limited to promotional emails and banners, which many people find rather annoying and ineffective [141]. It is quite hard to carefully select appropriate target audience for email advertisement and most of them are sent to random computer users around the world. Because of this email advertisement is mainly ignored by recipients. Moreover, using this advertisement channel may even cause negative impact on the retailer reputation [85]. Banner advertisement is much less annoying, but is also very limited in presentation. One banner can usually

¹<http://www.colesonline.com.au>

promote only one product, unlike offline promotion facilities in brick and mortar stores, which are able to present the whole variety of available products.

Some E-Commerce web sites (usually the ones that are associated with brick and mortar stores) also employ online catalogs for improving the advertisement impact of their sites, but the majority of the E-Commerce providers don't use this technique and purely rely on banners and promotional email.

The advertisement possibilities in brick and mortar stores are richer. Apart from brochures and catalogues product demonstrations or trial techniques are quite widely used. Supermarkets, for example, often employ staff to demonstrate their new products or organize free food tasting stands inside the shops. Make up promoters in big retail stores often offer customers free make up trials and employ professional consultants that can demonstrate the effects of a particular product.

Another important advertisement factor are so-called "specials", heavily discounted products, which find their ways in almost any area of traditional commerce. Specials vary through promotions of new products in supermarket, seasonal discounts in clothing stores to cheap airfares in travel agencies.

Despite being heavily used in brick and mortar stores specials are not widely present in E-Commerce. Behind E-Commerce there is usually an assumption that the prices offered online are already the lowest possible ones. Therefore, most of the bargain hunting tools (i.e. <http://www.cdrom-guide.com/bargainfinder.htm>) present online are focused on finding the best possible price for a specified product within a limited number of the available online stores. The search for special offers within a desired store a customer is loyal to is usually not supported.

Poor Support of Collaborative Shopping

Another downside of form-based nature of E-Commerce is its poor support of collaborative experience. There is a strong research evidence that suggests that when in the company of others people engage in shopping more often and enjoy it more [167]. There is also evidence suggesting that when people shop together they spend more time and money on this activity [206].

In brick and mortar stores people highly value the possibility to ask friend's opinion on a particular clothing piece, which may give them extra confidence in making a purchase decision. They enjoy the ability to be together in a grocery store and decide what to buy based on the discussion of possible dinner options. Being together at travel agent's place could help in satisfying the requirements of all travellers participating in a group trip and help in avoiding future disappointments [206].

The support of collaborative experience in E-Commerce is usually not provided. One

of the closest attempts to implement the collaborative shopping is Stylehive system², which allows users to annotate different web sites and make this annotation accessible for others. In this way shoppers may communicate their previous experience to their relative, friends or other unknown people. While this approach is quite useful for following someone's taste and discovering the products that a familiar person recommends, it is still far away from direct communication as in the above mentioned examples.

Inconsistency throughout Different Interfaces

In brick and mortar stores a customer may be faced with different product placement techniques or unpleasant sales assistants, but the process of shopping is always the same and doesn't differ from one shop to another. Once a customer learns how to pick up the products and pay for them at the check out there is no need to learn these tasks again.

In E-Commerce the situation is different. Every new store means new interface, new design and new problems for the users. For every new online store it will take customers a while to get familiar with the interface and learn how to perform necessary operations. Expressing decisions through mouse clicks is not natural for humans and perceiving the world through a 2-dimensional screen contributes to the overall confusion.

A research study that had a goal to analyze usability of different E-Commerce web sites has shown that throughout the whole range of the tested online stores on average a 40% of the buying attempts failed because customers experienced problems in performing simple operations (navigation, selection, payment etc) [103]. So, almost half of the users in this study gave up on making a purchase just because they were unable to do things that are natural in brick and mortar stores and never cause any problems there.

Limited Information Presentation Facilities

Form-based information presentation is not natural for humans and vast amounts of textual information are hard to process. Most of the operations a customer of an online store has to perform involve dealing with information expressed in the form of text: navigating through different sections of the web site, searching for a desired product, getting familiar with product characteristic; all these tasks normally require text processing. Many web site developers seem to be aware of this problem and are keen on utilizing the emerging technological possibilities in order to enrich the web sites with graphical elements. However, the amount of text processing required to be performed by the visitors of form-based web sites is still much larger than in the brick and mortar stores.

In brick and mortar stores we use all our sensors to perceive information. Most of

²<http://www.stylehive.com/>

this information comes to us in a graphical form, the rest is smells sounds etc, and only a small percentage of it is represented in textual form. What in brick and mortar stores can be perceived in milliseconds (smells, non verbal cues, noises, movements) requires detailed textual documentation on the web sites and often (like in the case of smells) can hardly be documented at all.

A famous cliché that a picture tells a 100 words precisely describes the problems that online shoppers are faced with. It would take us less than a second after entering a poster exhibition in a brick and mortar gallery to determine which are the most popular paintings, what is the appropriate dress code, are there artists available, what is the genre of the presented art works etc. Presenting such information on a form-based web site would usually require hundreds or even thousands of words. About 200 words would be required to produce a perfume fragrance description³, which can hardly be comparable by its effect to the experience of actually smelling the perfumes.

An average human can only read about 200 words per minute with 60% comprehension⁴. This means that almost every text block present on the web site can potentially take 1000 times longer to be processed than the corresponding image, sound or smell. Form-based information presentation, on the one hand, reduces the information richness of the real world and, on the other hand, increases the information overload due to the increase of the processing time required to comprehend what is presented.

One of the existing solutions for dealing with information overload in E-Commerce is employing the search facility. Search, however, causes another set of problems. It involves the articulation of an information need, often ambiguous, into precise words and relationships that match the structure of the system being searched (either manual or automated) [25]. In an automated environment, the user must apply two types of knowledge: knowledge of the mechanical aspects of searching (syntax and semantics of entering search terms, structuring a search, and negotiating through the system) and knowledge of the conceptual aspects (the how and why of searching – when to use which access point, ways to narrow and broaden search results, alternative search paths, distinguishing between no matches due to a search error and no matches because the item is not in the database, and so on) [25]. All this makes search a rather inefficient solution to the information overload problem.

Lack of Shopping Assistants

“Shopping assistant” is one of the most persuasive sales tools in traditional commerce. Shopping assistants offer help in a store, provide detailed information on products and

³http://www.perfumes.com/eng/women_product.htm?prod_id=7964

⁴<http://www.readingsoft.com/>

simplify decision making process helping to find a good that satisfies customer's requirements and various constraints. One of the major drawbacks that E-Commerce is facing today is the lack of such sales clerks. There is a strong evidence that in brick and mortar stores customers find interaction with a sales person very beneficial. People value and are willing to pay for the reduction of perceived risk, the optimal configuration of the transaction for their specific usage context, and the enhancement of the in-use experience, which shopping assistants can provide them with. There is scientific evidence suggesting that employment of shopping assistants can cause a significant increase in sales. A study conducted by "Celio" showed that introducing shopping assistant in a store resulted an instant 18% increase in purchases [47].

The demand for customer service and assistance in E-Commerce is going to be growing. Researchers [139] suggest that we are now in the ten year transition in the way consumers shop and save. The majority of the Internet purchases are still done by technology optimists, which are ready to deal with the lack of support for the sake of new and convenient technology, while mainstream customers are just starting to shop online. When the majority of mainstream shoppers arrive, either sites are going to have to offer more customer service and sales support on their sites, or they are going to see skyrocketing customer service costs.

If there is a problem to be clarified, online shoppers of today have to use expensive telephone service or accept long waiting times from e-mail communication. In case of the telephone service, either consumers who have to pay high per minute rates or the vendors that provide free customer service numbers have to deal with significant telephone service costs. Moreover, the majority of customers do not normally feel enthusiastic about switching from Web browsing to the telephone service [139].

The situation with e-mail as the way of customer support is not much better. Many e-mail enquiries take multiple correspondences to be satisfied, e-mail isn't real time while customers expect quick responses, quality assurance is difficult. According to [88] the only customer service option that seems to fit the speed of the Internet is real time chat.

Commercial success of the chat technology was documented by Earthlink [61]. Tracking customer actions at a web site offering ISP services showed that 70% of the customers were leaving without purchasing even when they saw broadband was available in their area for a "best offer" price. Following this observation the shopping team decided to integrate a chat feature into the web site to provide confused customers with real time online support, and conducted a case study for better understanding of the attitude of the shoppers towards chat service. The results showed that 15% of people who chatted with sales assistants converted and 80% of consumers gave the chat a good or excellent rating. Moreover, 61% of chat users preferred chat to other customer support options.

One of the great advantages of chat from the site owners' point of view is that it is also much less expensive than the telephone service. A phone call costs in average around US\$6.00, but the average chat is US\$2.40 [132].

Even further cost reduction can potentially be achieved with the help of automated chat tools where customers' questions are answered by a computer program (autonomous agent) rather than a human being [142]. These automations combine natural language processing algorithms with very large databases to give consumers answers to their queries. The drawback of this approach is that the intelligence of such autonomous agents is usually very limited. In order for them to be able to act believable and provide customers with reasonable feedback without switching to human-operators straight away enormous resources have to be used for explicit training (i.e. see [176]). This training usually means that human operators have to type in huge list of possible customers' questions and for each of the questions provide an enormous variety of answers, which is very inefficient and resource demanding.

Many researchers identified trust as a very important issue in E-Commerce. According to [1] the availability of shopping assistants is the key factor for developing trust in an E-Commerce site. If those assistants are not humans, their believability and possibility to answer all customers' requests is at stake. None of the existing solutions today was able to pass the Turing test [127], so it is impossible for an agent to act as a human and give reasonable answers to all possible customers' questions. The majority of E-Commerce solutions tend to ignore this issue and offer on their web sites totally autonomous assistants with limited abilities to answer customers' requests. When dealing with a complicated issue these agents usually end up notifying the customer about their limited knowledge base and inability to answer the questions. Such situations may dramatically reduce the overall satisfaction and trust of the customers in the shopping assistant as well as in the E-Commerce web site in general. To address this problem [165] proposes to connect a customer with a human if the agent is unable to answer a question after several attempts. The drawback of the approach suggested in [165] is that switching to a human operator for answering customer enquiry doesn't usually extend the intelligence of the corresponding autonomous agent. If similar enquiry will appear next time, the agent will again have to search for human help.

Poor Shopping Experience

Providing a good customer experience becomes extremely important for retailers [159]. The commerce world became so competitive that many retailers are no longer opening stores to offer new products and serve new markets; they open them to "steal someone else's customers" [206]. The price still plays an important role for the customer in

making a decision which store to choose. In many cases, however, providing a better customer experience together with the product is rated higher than a lower price [159].

Brick and mortar stores invest significant amount of money into making customers feel comfortable in their stores and stage memorable experiences [159]. The retailers try to “attack” all our sensors to get through to us. The experiences we are faced with every day range through smells of the freshly baked bread coming out of the bakery, delightful sounds of piano music produced by professional musicians invited to entertain the customers in a shopping malls to entertainment theme parks in fast food restaurants or furniture stores. All these experiences attract people to enter particular stores and buy from them.

Once a customer is already convinced to enter the store it is also quite important for shop owners to keep the shopper inside as long as possible and stimulate more purchases. Research on customer behavior [206] suggests that the more a shopper remains in the store, the more money he/she will eventually spend. And the amount of time spent in the shop depends on the quality of experience provided to the customer [206].

E-Commerce in its present form still seems to find itself in the product economy or service economy phase. Therefore, the choice of the E-Commerce provider is usually made by the customer solely on the basis of minimal price and trust. The notion of experience do not seem to be an important factor there, while in brick and mortar stores the phenomenon of experience economy [160] is well understood and is being successfully exploited.

No Support for Non-verbal Clues

Non-verbal clues are a set of means by which humans communicate except for linguistic system and its derivatives (e.g. writing, sign language etc.). They play an extremely important role in human interactions. Psychological studies have concluded that more than 65% of the information exchanged in face-to-face interactions is expressed through non-verbal clues. Without them human communication loses its colorfulness and expressiveness [8].

There are three main channels through which non-verbal communication is taking place [51]:

- the body and its movement
- the artifacts linked to the body or to the environment
- the distribution of the individuals in space

On the form-based web sites none of the aforementioned channels can be successfully utilized to fully support non-verbal clues, but it doesn't mean that there are no substitutes for non-verbal communication possible there. In form-based interfaces some of the non-verbal information can be explicitly presented in textual form. The most popular way of doing it is by utilizing so-called "smilies"⁵. The problem with this approach, however, is that emotional communication channel in text communication is very limited and emotion transmitters like smilies can hardly mirror the shifting situations in which the communication take place [196]. Moreover, as the true emotional state of the user can not be recognized from non-verbal clues there is no guarantee that the received smile reflects the actual state. One of the consequences of this is that trust is much harder to gain in textual communication [91].

Non-verbal clues may significantly help in developing trust between individuals [91]. Furthermore, they remain by far the best way to convey information about the affective state of a person [89]. In particular, postures and facial expressions are the most successful mechanisms for communicating emotions and states-of-mind [89, 8]. Emotions prove to be quite important to ground social interactions [36]. Expressed through non-verbal clues emotions often help in starting a conversation and are useful in monitoring the responses of the participants.

Additionally, non-verbal clues are quite useful tools of persuasion. As a proof of this a study [79] conducted with a large group of university students confirmed that non-verbal clues may have a significant impact on subjects' decision making. The students in this study were told that they participate in market research requested by a company making headphones and the company wants to test how their products operate if the listener is in motion. The students were divided into three equally sized groups and each of the students was given a headset. Every group was listening to a radio editorial arguing that the tuition fees at their university should be increased. One of the groups was instructed to nod their heads vigorously up and down while listening to the recording. Second group was told to shake their heads from side to side. The final group was instructed to keep their heads still. At the end of the recording each of the students in each of the groups was asked the same question: "What do you feel would be an appropriate dollar amount for undergraduate tuition per year?" The answers are quite astonishing: the majority of students who kept their heads still said that the amount they currently pay should be kept. Most of the students who shook their heads from side to side strongly disagreed with the proposed increase and wanted the tuition fees to fall. A large number of students who were told to nod their heads agreed with the arguments

⁵Presentation of different kinds of emotions through various combinations of ASCII characters. The most popular smilies are ":" for happy and ":(for sad.

presented in the editorial [79].

The above examples show that non-verbal clues are an important part of our every day life and are quite useful in commercial activities. Excluding them from E-Commerce significantly limits the possibilities of the sellers to promote their products as well as it limits buyers' ability to gain necessary information from the environment and surrounding people. With no support of non-verbal clues it becomes much harder to develop trust between humans and to integrate social interactions into E-Commerce portals.

6.1.2 Beneficial Aspects of Virtual Institutions

The drawbacks of E-Commerce presented above can be successfully addressed with employment of Virtual Institutions. Next, we demonstrate how Virtual Institutions can be helpful in this respect.

Product Presentation

In contrast to form-based interfaces, Virtual Institutions offer a consistent and immersive experience. Product presentation is 3-dimensional and is quite similar to product presentation in real world, and may be comparable by its effect on the consumers [56]. Looking at objects from different angles, changing of viewing direction and the distance to the product is possible to even higher degree than it is in the real world. As no physical restrictions exist in Virtual Worlds, product presentation can happen even at microscopic levels. A bright example of the power of 3D visualization is a demonstration of how a microwave oven works conducted by the developers of the online World Book encyclopedia⁶. In this presentation 3D visualization is used to illustrate how molecules of water present in food interact with the waves emitted by the oven. Such a presentation is impossible in real world, while in a Virtual World it is easily achievable.

Although producing 3D models is still quite expensive there are solutions available that offer very efficient model building techniques using laser scanning [72]. It is claimed that an average city can be scanned within minutes by applying this method. The possibilities to reproduce tactile characteristics of different objects [151] as well as transmitting smells over the Internet [118] also start to appear.

Social Interactions and Non-verbal Clues

Virtual Institutions implicitly incorporate location awareness of other users and offer mechanisms for social interaction. They support to a certain extent the way humans operate and communicate in the real world. The participants of Virtual Institutions are

⁶<http://www.worldbook.com>

embedded as avatars and can freely interact and communicate in a shared visual space using verbal and non-verbal ways of interaction [54].

3D visualization offers additional possibilities to get people involved into conversations. Observing the position of participants and the direction of their eyesight provides information about the *environmental context* of each participant and helps to find conversation topics based on this context [201]. The interaction can happen using verbal, written and non-verbal clues. Written form of interaction currently dominates, but voice communication through headphones and microphones is also supported by the majority of 3D visualization platforms. Movements of the bodies and attached artifacts, distribution of avatars in space and appearance of others helps to enrich social interactions with non-verbal clues. Gestures and facial expressions can also be used for communicating non-verbal information [54].

Furthermore, the support of emotion expression facilities (provided in 3D Virtual Worlds through gestures) is also an important facilitator of social interactions [36].

Support of Impulsive Decisions

Virtual Institutions also have a great potential in satisfying the needs of impulsive customers. As it was noticed by [46] the visualization of a shopping environment as a 3D Virtual World is quite close to the shopping environments in real world. It is possible for the customers to observe a large number of products and make selection without a need to spend time browsing through catalogues and reading textual descriptions [42]. The same product placement techniques as applied in brick and mortar shops can be successfully utilized in virtual shops [42]. Personalized product placement (when the location of a product in space depends on the profile of the user) may even make this process more efficient than it currently happens in brick and mortar stores [42].

Interactive and Less Conspicuous Advertisement

Similar advertisement channels available in brick and mortar stores are also present in Virtual Institutions. Moreover, Virtual Institutions allow for more engaging and less conspicuous advertisement techniques than both form-based interfaces and brick and mortar stores. The idea of animated products as a navigation aid for E-Commerce (where 3D models of different products walking around the store helping users to find appropriate section in a supermarket) is a bright example of an innovative way of advertisement presentation [46]. On the one hand, advertisements in this case help buyers to improve navigation and bring entertainment. On the other hand, they satisfy the demand of the sellers for advertisement. Another possibility that appears with the employment of Virtual Institutions is conducting engaging competitions, where an instance of a product

can be won in a game-like tournament. For example, a skateboard competition may be conducted inside a virtual store and the winner in the Virtual World will be awarded with a free skateboard in the real world.

Collaborative Shopping

With the use of avatars collaborative shopping becomes possible and can be even more convenient than in brick and mortar stores [153]. With Virtual Institutions there is no need for the participants to be physically located in a store, but they still can participate in a joint virtual shopping activity. People can be physically located in different countries and still be able to have a joint meeting at a travel agent's place or walk together through a shopping mall.

Consistency of the Interface and Reduction of the Information Overload

The provided 3D interface is quite natural for humans as it reflects the real world [46]. Once a participant learns how to navigate inside a Virtual Institution he/she will naturally be able to act inside any store supported by this technology. The amount of text can be kept to a minimum, so that no information overload occurs. The fact that Virtual Worlds help reducing the information overload made Cisco choose this media for conducting many team meeting in Virtual Worlds [82]. Similar to the real world, the use of 3D visualization also causes the level of detail to decrease with the distance, which helps to structure information into different levels of abstraction. For example, in a virtual supermarket a customer will only be able to see basic appearance details of the products (shape, colors) that are located further away, while at the same time it will be possible to read textual labels or detailed descriptions for products in close proximity.

The use of intelligent sales assistants has also proved to reduce the information overload of the consumers [190]. They can act as information filters and only present the information relevant for the consumer, instead of presenting it all and make the consumer search through it.

Shopping Assistants

Virtual Assistants controlled by autonomous agents have a high potential to satisfy the demand of the customers for sales clerks. Implicit training method proposed in Chapter 5 creates a possibility to make these clerks believable and intelligent. Wide use of autonomous agents that learn from multiple sources can increase the quality of assistance and at the same time reduce its cost.

The existing scientific evidence suggests that the use of shopping assistants embodied as 3D avatars may have a positive influence over trust and online purchasing intention of the consumers [98, 171].

Rich and Personalized Experience

Shopping experience in Virtual Institutions can be as rich and engaging as in the real world [42, 41]. The ambience and unique atmosphere of a brick and mortar environment can be easily replicated in a Virtual World. Moreover, virtual environments can be personalized according to the preferences of the participants and enhanced with entertaining activities that are not possible or too expensive to conduct in a brick and mortar store. Quests, competitions and other forms of entertainment may help to attract customers and keep them longer inside the shops.

Benefits for the Sellers

Sellers also receive additional benefits with the use of Virtual Institutions. In particular, it is widely believed that Virtual Worlds is a perfect new channel for marketing and advertising [13, 95]. Sellers in Virtual Institutions can implicitly attract attention to their products using advanced interior decoration techniques, lighting and even music. Product placement techniques, which are widely used in the real world, can be employed to the same degree as in brick and mortar stores. Sellers may attract attention to themselves selecting unusual appearance or performing unexpected actions. There is also a lot of space for statistical analysis, as each buyer is fully observable and autonomous agents associated to them may collect statistical information and pass it on to the sellers (if allowed by the buyer).

Novelty

Finally, one of the factors that can make virtual experience provided by Virtual Institutions more successful than direct experience of the real world or form-based experience of E-Commerce web portals is the novelty of product presentation. There is a strong evidence [64] that people that have not been exposed to 3D presentations of products online may simply be more curious than if the information was presented in another medium or format. Driven by curiosity they will enter virtual stores and depending on the provided experience will decide whether to switch to this media or not.

6.2 Developing a Virtual Institution for E-Tourism

Now that the key drawbacks of existing E-Commerce solutions are identified and the advantages of applying Virtual Institutions to this domain are detected, we find it necessary to present an example within this domain, which would demonstrate the capabilities of Virtual Institutions. In order to select such an example the domain of E-Commerce had to be narrowed down to the area where the drawbacks are the most evident and the benefit of using Virtual Institutions is significant. As a result of a detailed research analysis the area of E-Tourism was identified as the application domain within E-Commerce that satisfies the aforementioned requirements.

Further in Section 6.2.1 we explain the reasons for selecting E-Tourism. Then, by developing the World Trotter Travel Agency prototype in the domain of E-Tourism we provide a proof of concept for the Virtual Institutions Methodology outlined in Chapter 4. The key focus of Section 6.2.2 is on Step 1 of the methodology: Eliciting the Specification Requirements. The details of the application of the rest of the methodology steps have already been explained and only some specific aspects relevant to the development of the World Trotter Institution are described in Section 6.2.3 and Section 6.2.4.

6.2.1 Selecting the Application Domain

In search for a convincing illustration of the capabilities of Virtual Institutions we took into account the identified benefits of this technology as well as the weak points of existing E-Commerce solutions. We looked through a range of different problems that E-Commerce is trying to solve with an intention to identify the ones where the drawbacks are the most evident and the benefits of applying Virtual Institutions are significant.

As a result of the detailed literature review we selected E-Tourism as a subdomain of E-Commerce that satisfies the above requirements. In order to justify our choice we further present the analysis of the relevant aspects of E-Tourism from the point of view of the product and look at these features through the prism of the experience economy.

Once the application domain was identified we conducted a deep analysis of this domain in order to understand the shortcomings of existing solutions, needs of the users and activities that users desire to perform in E-Tourism. The acquired knowledge showed that the employment of Virtual Institutions can address the identified drawbacks of E-Tourism and helped to elicit the specification requirements for the Normative Control Layer of the target Virtual Institution.

In Chapter 4 it was shown how to develop a Virtual Institution on the basis of the specification requirements. In this section we demonstrate how these requirements can be obtained. For the case of E-Tourism the following strategy was employed to elicit

specification requirements. The analysis of the domain started with identifying the features of the tourism as a product with the focus on those features that are not properly addressed by the existing solutions. Next we analyzed what actually motivates people to travel, what motivates them to select a particular destination and which of these motivations haven't received proper attention from the E-Tourism providers.

Once the aforementioned features and motivations were recognized, the next goal was to determine how these can be addressed with the help of Virtual Institutions. To do so we learned from the strategies that are being employed in addressing the drawbacks of E-Tourism by existing travel providers.

The class of travel providers that are quite successful in this respect are brick and mortar travel agencies. Many researchers [23, 121] have predicted that brick and mortar travel agents will be soon fully replaced by online booking. Although, the number of travellers using them is declining, but not as sharply as it was predicted. Brick and mortar travel agencies seem to have found their market niche. They are now used for planning more complex journeys, as well as booking group hotel and flight packages [145].

To verify the connection between the drawbacks of E-Tourism and the benefits provided by brick and mortar travel agents and to gain more insight into their specific features that attract the travellers, we conducted a user study. One of the goals of this study was to discover the features and activities present in travel agencies that could be integrated into online booking with the help of Virtual Institutions. Another goal was to determine important features and activities associated with online booking, so that they are not accidentally eliminated in the target Virtual Institution. The results of the study were documented and formalized in the specification of the Virtual Institution.

Next, we describe how each of the stages of eliciting specification requirements was conducted. The presentation starts with analyzing tourism as a product.

Tourism Product in the Experience Economy

What sets E-Tourism apart from other areas of E-Commerce is the nature of the product that is being sold. Tourism product is intangible in nature [106]. Unlike other fields of E-Commerce, what is being sold in tourism are mostly experiences and services [106]. Due to this fact the influence of the experience economy [160] over tourism is much higher than in any other industry.

The experience economy phenomenon has been discovered quite recently and its role in E-Tourism is as significant as in tourism overall. Despite this, existing travel providers offering their services online often find themselves in the service economy era. The majority of sales strategies employed in E-Tourism are mostly product or service oriented, not experience oriented. Instead of highlighting and promoting experiences

the majority of advertisement efforts in E-Tourism are put into promoting services (i.e. flights and accommodations). Most of the travel providers are not usually concerned with enhancing the experience provided during the purchase or staging experiences offered at the destination. They tend to ignore the fact that in today's reality the experience value of a product or service is a dominant factor influencing consumers decisions to purchase a product [209] and discovering the experience associated with a product may stimulate the customers for impulsive fast purchases [159, 166].

To keep up with the changing economical circumstances existing strategies of online travel providers may require rethinking, as today's situation in the travel market is totally different from both product and service economy times.

In 1950s and 1960s (product economy still dominates) tourism was seen as the time of family togetherness. An average person only travelled to a domestic or nearby destination to spend some time with the partner and kids. Back then, there were no scheduling problems and international travel was the preserve of business people and the rich [208].

In 1960s and 1970s (service economy takes over) consumers started to move away from the traditional commitment to materialism and obligation to others and began to seek personalized products. They selected more remote destinations and demanded a better service. A great number of people viewed vacations as a leisure time, when they do absolutely, positively nothing. The tourists may not have gotten out of a swimsuit the entire time they are on a vacation, but they wanted someone else to bring the frozen daiquiri to the pool side – and it should better be well made [208].

Today, when the experience economy matures, there is a shift of customers' preferences from goods and services to experiences. For many travellers a service became secondary, and it can even be sacrificed for a better experience. It is time when providers have to stage the customer experience beyond basics; just a good service is not enough anymore. Moreover, the travellers of today decide to buy or not to buy based upon how real they perceive the product/service offering to be and the experience they receive during the trip selection is very important and determines the success of the purchase [221].

In order to satisfy the growing customer demands in the experience economy it is important to gain a clear understanding of the features of the tourism product as well as motivations and desires of the customers. As the travel product becomes more and more diverse, it becomes harder and harder for consumers to choose the right packages, as well as for the travel providers to create these packages. We argue that experience economy increases the demand for travel intermediaries, which can provide information, reduce the diversity, offer professional advice and stage the ultimate customer experience.

Online booking starts invading the travel market, because with the Internet travellers have received the possibility to access the information about prices, deals and sched-

ules in the comfort environments of their homes and create travel packages themselves [122]. However, the amount of this information is so huge, that the majority of travellers to previously unvisited destinations would use traditional travel agents for filtering it [122]. The amount of time a traveller spends for creating a trip package online is significantly higher than the time an experienced travel agent would spend for this task [122]. Moreover, the online booking in its current form is not concerned with providing a good customer experience and is unable to cope with the uncertainty in the requirements of the travellers [122]. Many travellers, even technology optimists who feel quite comfortable with online booking would still book many trips with travel agents [122]. This shows that existing online portals are not addressing all the requirements of the travellers.

Based on the above presented evidence we emphasize the importance of online travel agents as the subclass of travel intermediaries that have the highest potential to take over a significant amount of the travel market. With the help of Virtual Institutions they will be able to reduce customers' exposure to complexity in combining the travel packages, addressing their complex requirements and exploit the sophisticated nature of the travel product. Virtual Institutions have a high potential to help in coping with the challenges of the experience economy through employment of rich visualization mechanisms. 3D visualization can provide customers with pleasant purchase experience as well as offer means to discover experiences associated with the products being sold.

Besides visualization, tourists also require a much higher degree of support during the decision making process than the customers of the majority of other fields of E-Commerce. The product in tourism is quite complex and consists of a large number of components. Making a decision on identifying the necessary components and combining them into a package without decision support mechanisms is not an easy task. Such support can be provided by the wide employment of sales assistants who may be autonomous agents that are capable of extending their intelligence by learning from the humans through implicit training mechanisms (described in Chapter 5).

Apart from better product visualization and reduction of uncertainty in package selection most of the benefits of using Virtual Institutions presented in the previous section are also applicable to tourism, as the majority of identified weak points of E-Commerce drawbacks are also present here. To gain more insight into how Virtual Institutions can address these drawbacks it is important to understand the unique characteristics of tourism. Through a detailed literature review we identified the features of the tourism product and motivations of the travellers, which are not properly addressed in existing form-based solutions. For each identified feature and motivation we present the benefits that can be gained by using Virtual Institutions. To confirm the benefits of Virtual Institutions in E-Tourism and gather evidence in favor of online travel agents we also conducted

a study that had a goal to identify the features of brick and mortar travel agents that are highly desirable in E-Tourism. The identified features help to understand which of these can be used to improve existing online tourism portals.

Tourism Product Features

Tourism as a product has been defined as “an amalgam of three main components: the attractions of the destination, the facilities at the destination and the accessibility of it” [76]. This definition, however, is too broad and doesn’t completely answer the question what sets tourism apart from other businesses in context of the experience economy. Based on the literature review we selected five unique characteristics of tourism that make it different from other products sold in E-Commerce. For each of the characteristics we showed how the use of Virtual Institutions can benefit potential travellers.

1. *Tourism product is intangible in nature.* Souvenirs and photographs are the only tangible reminders, purchased at a much lower cost than the experiences provided with the trip [106]. More specifically, none of the components of a tourism product can be seen or touched before purchase by neither a buyer nor a seller [106]. For this reason, tourists need to get as much correct information as they can feel confident that their desires and expectations will be fulfilled. That is the reason why information, along with price and customer service, is one of the key competitive factors and an element that affects the tourist-receiving society and the quality of the travel experience [30].

3D visualization supported by Virtual Institutions provides an efficient way to present destination related information. The evidence provided by [56] suggests that in general virtual experience (3D visualization) can be much richer than indirect experience (form-based presentation) and sometimes as rich as direct experience (direct contact with the product). Product knowledge and decision quality are both significantly higher when exposed to interactive 3D products than to static form-based product presentations. As direct experience prior to purchase is impossible in tourism (unless a traveller visits the same destination again), virtual experience seems to be one of the most efficient way of information presentation. Such virtual experience may include 3D interactive tours through main attractions of the destination or accommodation, which may give a much more clear impression about the package a customer is about to purchase than photographs and textual descriptions. By careful selection of what to present, the most important aspects of the destination can be highlighted. By “walking” through the list of possible resorts at the destination the customer will be able to gain an understanding about

the location, facilities and even about the kind of people that normally stay there. For example, by showing children playing in front of a hotel the customers can be notified that this is a family friendly resort.

2. *Tourism is remote in nature* with the person selling a product possibly never having seen the destination sold and yet being the prime source of advice [106]. This has a consequence that the collective intelligence is required to provide customers with correct information about the trip [106].

Such collective intelligence can be supported with implicit training mechanisms. The task of answering customer enquiries will be done by a trained autonomous agent. When the agent associated with the avatar of the travel agent is unable to answer a customer enquiry it will contact a human operator. The human operator will be presented with the context of the conversation, so that it is possible to select an operator who possesses expertise in the topic being discussed. The autonomous agent associated with the avatar will collect the information from each of the operators as they answer customers enquiries and in case of similar future requests will be able to share the gained expertise more rapidly. At present the only access point for collective intelligence in tourism are moderated travel forums⁷. The information present there, however, is mostly text based and finding something on the topic of interest takes a lot of time. With the help of Virtual Institutions some of this information can be represented in visual form by letting the users enrich the 3D environment at a given destination with meaningful objects. Autonomous agents can also be employed as a human friendly interface to the forums.

3. *The description of a product is always subjective* (somebody's perspective) and background related [76]. This suggests that the more real the travel product presentation is the less uncertain is the customer about its characteristics.

3D visualization has a potential to be the most objective way of information presentation. In contrast to photographs it is not showing the part of the object that fits inside the viewfinder, but provides the possibility to fully inspect the object from every side. Reputation mechanisms can help to ensure that the 3D objects presented to the users correspond to the original. Using digital signatures for verifying the validity of the objects present on the web sites may act as an important argument in dispute resolutions.

4. *The product is delivered by many different firms, which are typically different in terms of their functions and capabilities* [152]. For the customer this means that

⁷i.e. <http://www.tripadvisor.com>

the more components are present in the desired package, the harder it is to order the package without help from travel intermediaries.

Virtual Institutions with support of implicit training will be able to provide such support and at the same time minimize the number of actual humans being involved by employing autonomous agents.

5. *Few other industries link so many diverse and different kinds of products and services* as the tourism industry [62]. The list of all travel-oriented “commodities” is too large, so a catalogue is not feasible to produce [175].

As it was shown in the previous section Virtual Institutions have richer product presentation facilities than form-based interfaces. The 3D Virtual World can be enriched with localized visualizations, which are much more appropriate than endless catalogues. The employment of travel assistants (which can be either humans or autonomous agents) can help customers in filtering the available information.

The above-mentioned characteristics explain what is sold in the tourism industry and how it is sold. In order to complete the picture it is also necessary to understand which motivations create the need for a person to travel.

Consumers’ (Travellers’) Motivations

According to [76] the factors that motivate tourists to go on a trip include: sense of power (visitors demanding service and attention), search for romance, desire for cultural exchange, need for leisure/escape, desire of social contact, following social trends, change from routine to new experiences, satisfying curiosity and personal values (i.e. trips to places of religious significance). The choice of a particular destination is usually motivated by: *scenic beauty of a place, pleasant attributes of the local people, suitable accommodation, rest, relaxation* and, finally, the airfare [76, 100].

Existing online travel providers are very successful in presenting the fare-related information and are moderately successful in regards to the destination presentation, which is normally limited to 2-dimensional images and textual description. However, the rest of the motivations (which are mostly social and experiential in nature) are often not addressed, as those factors are hard to present in terms of catalogues. Travel agents are capable of presenting some of the attributes in a narrative form, but many of them haven’t been to the destination and can not really provide any kind of advice.

Modeling a 3D representation of a destination makes it possible to objectively assess the scenic beauty. The accommodation presentation in 3D Virtual Worlds may also help customers to assess all the characteristics and make a sound decision. Rest and

relaxation factors can be expressed in Virtual Institutions through non-verbal clues. The pleasant attributes of the local people are harder to represent, however the most typical locals can be shown as avatars and their behavior may be collaboratively modified by the travellers to reflect their experiences in this respect.

Further travellers' motivations are outlined in [193]. According to this research, when searching for a place to go, tourists create a stereoscopic image of the kind of people who typically visit a given destination (*destination visitor image*), which is then compared with the actual *self-image* (how travellers see themselves) and an *ideal self-image* (how they would like to see themselves). This comparison is a crucial factor in the destination selection, but in order to make it accurately, a certain level of the customer involvement needs to be reached. One possible way of estimating the self-image is creating the user profile of the traveller, which helps to classify different customers into most typical categories and further facilitates the reduction of diversity in the proposed package. The PRIZM system proposed by [76] classifies consumers by 12 social groups and 40 lifestyle clusters. Some of the classification segments are determined by: travel habits and preferences, group or individual travellers, purpose of travel, demographics, psychographics and frequency of travel. Such segmentation helps in understanding of what a particular class of customers value and what trade offs can be made.

For helping travellers to create the stereoscopic destination visitor image the same technique as for representing the pleasant attributes of the local people can be applied. The ideal self image of the travellers can also be easily discovered, as an avatar which a user selects reflects this image to a very high degree [201]. Some of user preferences can be recognized by the human assistants just by looking at customers' avatars (while autonomous agents would require mechanisms for classification of the avatar appearance into predefined social types). Some other preferences may be extracted from previous transactions.

The analysis of the characteristics of the travel product and motivations of the travellers through the prism of the experience economy provides an overview of the important features of the travel industry and helps to identify the needs of the travellers that are not adequately addressed with existing online solutions. The acquired knowledge presents a motivation for selecting tourism as an appropriate application domain for Virtual Institutions. Further we show how this knowledge is used to generate precise requirements for the implementation of the Virtual Institution prototype.

6.2.2 Eliciting Specification Requirements

To confirm that the discussed benefits of using Virtual Institutions in E-Tourism are actually demanded by the customers and to generate the precise specification requirements,

we conducted a user study. The main purpose of this study was to understand what are the reasons why some people do not use online booking facilities and prefer to go to travel agents instead. As some of the travellers (especially elderly people) clearly do not go online simply because they are not well familiar with computers we carefully selected our sample to exclude these participants. The findings of this study and the details of its design are discussed below.

Travel Agents vs Online Booking: User Study

In order to elicit the specification requirements for the Virtual Institution in E-Tourism we look at the relationship between online booking systems and traditional travel agents. We identify the “best of both sides” and highlight important features of traditional travel agencies such as simplification of decision making, support of impulse travellers, collaborative booking possibility, etc., that will be valuable to incorporate into the Virtual Institution solution.

Requirements Elicitation Methodology

To elicit the specification requirements we follow the qualitative research inquiry. Chowdhury’s [47] statement “Customers value, and are willing to pay for the simplification of decision making, the reduction of perceived risk, the optimal configuration of the transaction for their specific usage context, and the enhancement of the in-use experience” summarises the initial position of our research. We also rely on the results of the research study conducted by [122]. This study compared the pros and cons of online booking by using a survey method on the broad sample of North American Travellers in Seattle Airport.

The study presented in [122] was conducted in 2000, when online booking wasn’t as widespread as it is today. Therefore, it can not encounter for the progress made lately by online booking portals and for the changes in the consumer booking patterns. Furthermore, this study was focused on investigating the travel purchasing behavior of consumers, rather than on eliciting the requirements for Virtual Institutions. It used a very general sample, while for eliciting requirements we find it necessary to use a much more limited group of respondents. Another drawback of this study was that due to the selected study method (survey) it could only analyze the available features of travel booking, while we also need to analyze the possible future features. Therefore, we had to reuse the results acquired by [122] and conduct a more specific user study. This study was conducted on the following 3 stages:

Stage 1: Hypothesis formulation. The set of hypotheses has been formulated based on the literature review, the results of the previously conducted study [122] and the in-

formation obtained through semi-structured interviewing of a carefully selected sample – active travellers possessing ample expertise in online booking. Moreover, we also considered travel agents’ point of view. The results of the publicly available interview [199] were used for refining the hypotheses.

Stage 2: Hypothesis validation. The set of hypotheses provided the dimensions for constructing a questionnaire. After testing and refinement, the questionnaire has been used to collect data that provides evidence in favor or against these hypotheses.

Stage 3: Requirements formalization. The formulated hypotheses that were confirmed by the study form a basis for the formalization of the requirements for the development of the Virtual Institution in E-Tourism.

Stage 1. Hypothesis Formulation

Below we present each hypothesis, the supporting reference in the literature and relevant quotes from the conducted interviews (indicating gender and age of the respondent) to each hypothesis.

- **H0:** The majority of people prefer booking their international trips from a travel agent. Domestic trips are usually booked online. According to [122] and the interview results.

- **H1:** Human expertise is an important convenience factor that is missing in online booking [199, 122]. [travel agents know dates when flights are cheaper, may give a good advice (visa, insurance, dangers, etc). So there is no need to spend time on searching the web; travel agent will do everything for the client.]

“It’s more a lazy decision. Going to a travel agent is like to study with a teacher. It will save my time!” [Male, 25] “Online booking can be real pain if your travel plan is not simple” [Male, 30] “Travel agents have much more knowledge in travel issues than I do. I think the experience comes both from a lot of travelling they do and just from the fact that holiday booking is their profession. They can provide me with a valuable suggestion regarding where to go and with some details about the destination.”[Male, 45]

- **H2:** Social interaction with a travel agent is the key to a good customer experience [161, 199, 122].

“I just need face-to-face communication to understand all the details! This is very important to me. It is easier to understand each other via face-to-face communication! I work in distance learning... there is clear evidence that distant students perform worse than those who attend the classes.” [Male, 44]

- **H3:** Travel agents satisfy impulsive buyers better [9]. [i.e. decision making during planning and purchasing a travel arrangement is not always rational.]
 “Sometimes I just do not know where I want to go. I may go to a travel agent and say that I have 500\$ and want to go to a warm place where I can swim in the ocean and enjoy palm trees on the beach. It will take ages if I will go searching the web with such requirements.”[Male, 27]
- **H4:** Collaborative booking experience is important [33].
 “We usually go to a travel agent with my partner. But it may be very hard to get together as we work pretty far away from each other. It would be great to be able to meet online with her and the travel agent to discuss everything.”[Male, 27]
- **H5:** Security and trust towards humans is higher [122, 199]. [i.e. people feel more secure interacting with people and have higher trust to them than to web sites, interfaced with forms.]
 “Booking in the Internet is insecure. I will not book from a web site that I know nothing about”. [Male, 35]
- **H6:** Loyalty is rewarded and appreciated ([114, 199, 122]). [Customers believe that for their loyalty they will be rewarded with personal care and member discounts]
 “I always use the same travel agent. I shop around first to find which one is cheaper. But once I found the travel agent I like, I always book from them, because my loyalty will be rewarded. They know all my preferences, know that I’m an old customer. They will make a discount for me or at least offer me a better service.” [Male, 25]
- **H7:** Detailed information about a trip is important [48, 122]. [travellers use as many sources to gather the information about the travel destination as possible: Internet, opinions of other travellers and travel agents etc.]
 “If I have never been to a place before I need heaps of information. In this case I’m more likely to go to a travel agent”. “I use many sources. I take brochures from travel agent, ask my friends and search the web”. [Female, 27]

Stage 2. Hypothesis Validation

To gather the evidence in favor or against these hypotheses we conducted an online survey. The questionnaire of the survey comprised 40 questions. Most of the questions

are formulated as multiple-choice statements (e.g. “I believe that loyalty is always rewarded.”) with answers on a 5-point Likert-type scale (Strongly Agree, Agree, Neutral, Disagree, and Strongly Disagree). The questionnaire and the responses of the participants is publicly available on the Web⁸. The call for participation was sent to travel enthusiasts via e-mail and travel-related forums. No rewards were offered, as we are convinced that voluntary participants will answer the questions more accurately. We focused primarily on people that are experienced computer users, familiar with navigating the World Wide Web and booking online. Such a sample represents technology optimists, who are able to identify the conceptual shortcomings, rather than provide us with a straightforward answer that they do not use online booking simply because they do not know how to do it. More precisely, we are interested in the reasons why such people still prefer to rely on travel agents instead of booking their trips online.

Sample

One of the key assumptions behind our study is that the number of active computer users will continue to grow and the number of travellers who are not familiar with online booking will soon become insignificant. We make this assumption based on the statistical data [148], which suggests that the number of travellers who plan and book trips or vacations online continues to climb rapidly (with an approximate 10% annual increase). Therefore, we targeted participants who are confident with computers and Internet and who are likely to possess online booking experience.

For the purpose of the interview part of our study not only we required the participants to be familiar with online booking, but also to be active travellers capable of innovative thinking. This requirement is due to the fact that only such people would be able to represent the future travellers confident in online booking, as well as help us with accurate prediction of the future features of online booking portals and with the analysis of the drawbacks of the existing solutions. The desired sample was found amongst Australian researchers and PhD students. 10 People agreed to participate in extensive interviews aiming at producing the research hypotheses for the study.

To target survey participants we posted the call for participation on online travel forums and also sent invitations to the people who we knew would satisfy our requirements of being confident computer and Internet users. Subsequent to posting the call for participation, 132 people from 25 different countries filled in the questionnaires (61% male, 39% female). Figure 6.1 depicts the distribution of age and location amongst the respondents. As expected, almost all interviewees are heavy computer users (except for one person). The respondents spent an average of 12 minutes on completing the question-

⁸<http://www.my3q.com/view/viewSummary.phtml?questid=76535>

naire.

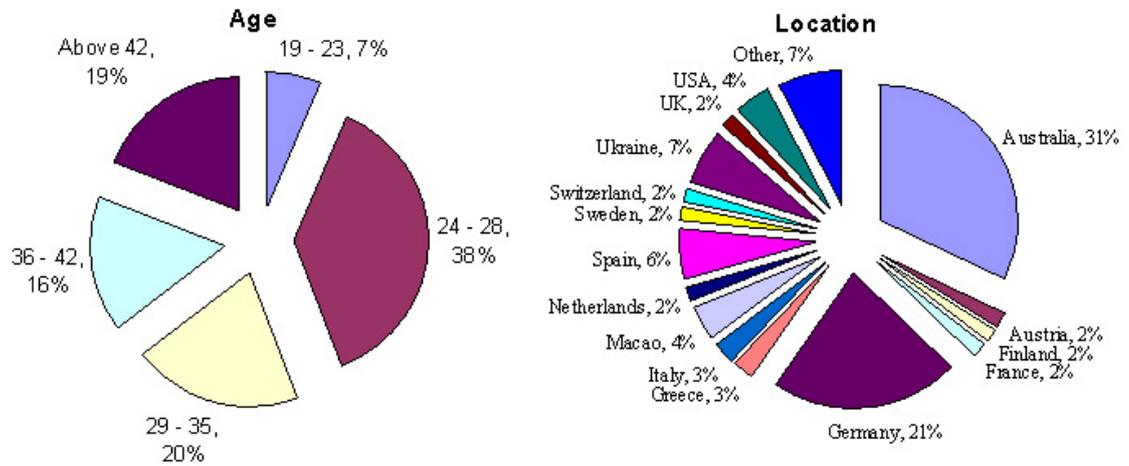


Figure 6.1: Age and Location of Participants.

Data Analysis

Further in this section we present the results of the analysis of respondent answers. The presentation structure follows the hypotheses formulated in Section 6.2.2.

Preferable Way of Booking [validating H0]

For the reasons explained above, our research is focused on heavy computer users only. That's why we could not rely on general official statistics for validation of H0. The results of the survey show that 65% prefer to book international trips from travel agents, despite the fact that 50% of the respondents believe it is cheaper to book online and only 3% think that online booking is usually more expensive. Additionally, 30% of the people never book international trips online in contrast to 13% that always book their trips online. In case of domestic trips the picture is slightly different, 54% of respondents tend to book online, which is 12% more than the number of people who book domestic trips from a travel agent. So, hypothesis H0 appears to be correct. The data analysis also revealed some gender specific patterns. In particular, females tend to visit travel agents more frequently than males. In case of domestic trips 53% of female respondents prefer to go to a travel agent.

Convenience and Expertise [validating H1]

On the one hand, about 60% of respondents think that booking from a travel agent is convenient and 13% disagree. On the other hand, 87% of respondents think that booking

online is convenient and only less than 10% disagree. This contradiction suggests that there is a strong need for incorporating important aspects from the travel agent experience into online booking systems. Our initial assumption that expertise of the travel agents is an important factor that is missing in nowadays tourism portals is supported by the following statistics: 54% of respondents agreed that talking to a travel agent helps to get a much clearer picture about the destination. 65% of respondents agreed that talking to a travel agent clarifies all details of the offer. 43% travellers agreed that travel agents provide them with important tips that are almost impossible to find online, e.g. insurance, visas, dangers, etc.

Social Interaction [validating H2]

78% of respondents answered that social interaction with a travel agent is an important part of the booking experience, indicating that hypothesis H2 is valid. However, 20% do not see the act of talking to travel agents as being beneficial for them. As we expected, females value social interactions with travel agents much more than males do. 45% of females agree or strongly agree that social interaction with a travel agent is important while only 9% disagree. Males are less keen on social interaction, 36% of them agree and 29% disagree. As part of investigating the social interaction element, we looked at preferable ways of communication. Only 46% of respondents would prefer multiple ways of communication with a travel agent. Among those, face-to-face communication is the most popular one (76%), followed by telephone (43%), E-Mail (34%) and real time chat (11%). The high percentage of people who prefers to communicate with a travel agent face-to-face highlights the importance of social interaction in trip booking.

Impulsive Decisions [validating H3]

Our survey showed that for 56% of responders a very cheap offer can change their travel plans, while only 25% disagreed on that issue. Another interesting aspect is that only 10% of responders are always sure about their destinations. These numbers support hypothesis H3 and show a major demand in an intermediary that assists travellers in selecting travel destination.

Collaborative Booking [validating H4]

In case of group trips, about 90% of participants consult with other travellers prior to booking. This shows that introducing the possibility of collaborative online booking may be very much appreciated amongst travellers and provides important evidence in favor of hypothesis H4.

Trust and Security [validating H5]

Security is regarded as a very important issue in any type of business. Despite the fact that all the respondents (except for one) use computers almost every day, 14% believe that booking online is not secure in general. When it comes to unknown online vendors 53% feel not secure. Interestingly, 39% of respondents regard booking at an unknown travel agent as not secure. An immediate interpretation is that these figures are caused by some regional specifics. However, the response distribution does not show any significant outliers with respect to different countries. Concerning trust issues, 8% of travellers answered that they feel uncomfortable (or do not trust) booking with computers and less than 3% said that they do not trust travel agents, mostly because travel agents tend to fool customers pushing them towards more expensive deals. The above results show that hypothesis H5 tends to be correct. Based on the results from [122], we expected that trust towards humans is higher than towards computers. Despite the fact that the results confirmed this initial assumption, the difference appeared to be insignificant, which gives us the reason to suggest that trust and security in computer systems are increasing and in the future they may be even higher than in humans.

Loyalty [validating H6]

We expected that convenience is the most important factor for the interviewees and the choice of a travel agent is defined by current location. Surprisingly enough, 43% travellers responded that they book their trips from the same travel agent, while only 29% would go to the closest one. 62% of respondents find it important that the travel agent knows their preferences and makes the decisions on the basis of that information. 38% of respondents believe that their loyalty is not rewarded while 33% believe it is. These results demonstrate strong support in favor of hypothesis H6.

Learning about the Destination [validating H7]

To validate hypothesis H7 we tried to understand whether people need help with choosing a destination or not. Only 39% of respondents agreed or strongly agreed that they do not need any help, while the majority either agreed or felt neutral about it, showing that it is important for people to gather the information prior to travelling. If looking at how people learn about their future travel destinations, the results of the survey suggest that only 9% of respondents would use a single source of information. For those travellers this source would always be either a web page or personal experience. 91% of the respondents would use more than one source of information. 79% of travellers believe that personal experience is the best way to learn about the destination. 73% of respondents

would rely on opinions of other travellers, while only 38% would rely on an opinion of a travel agent. The majority (84%) identified a web page as the preferable way for presentation of information about a future travel destination. 54% of interviewees answered that detailed photos are important in helping to make a decision about the destination. Surprisingly, both video and interactive 3D attracted 12% of respondents. We think that such a low response rate is due to the fact that these ways of product presentation are currently unavailable – not only online but also from a travel agent, and it was hard for respondents to appreciate their usefulness.

Stage 3. Requirements Formalization

The study confirms that many people prefer going to travel agents instead of booking their trips online. In case of international trips the majority of travellers would do so.

As it was expected the survey also confirmed our assumption that the number of impulse travellers and people who enjoy collaborative booking is very high.

However, respondents in general trust travel agents more than web sites and feel more secure about booking with people, the difference is rather insignificant. This may be due to the sample we selected, but could also mean that trust and security towards computers are increasing.

The study identified a number of features that respondents appreciate the most in on-line booking portals and brick and mortar travel agencies. Below we have summarized these features in terms of “the best of both sides”:

Travel agents pros:

- Expertise of travel agents.
- Convenience of using this expertise.
- Specific destination tips that are hard to find online.
- The possibility to receive help with impulsive decisions.
- Can satisfy a demand for social interaction before making an important decision.
- The majority of respondents remain loyal to a single travel agent and appreciate that travel agent knows their preferences.
- Collaborative booking.

Online booking pros:

- Convenience of booking online, where they can enjoy the comfort of their familiar environments.
- Fast responses on travel-related requests.
- More information on the destination is available online.
- Cheaper airfares.

Both classes of travel providers are valued by the respondents. The convenience of online booking as well as convenience of the human expertise is similarly appreciated by the travellers. The outcomes of this study suggest that online travel agency is a model capable of incorporating all the identified pros.

Requirements for Virtual Institutions Supporting Tourism

Virtual Institutions have a potential to eliminate the weak points of online booking by incorporating the pros of brick and mortar travel agencies into online portals. Based on the literature review, the results of conducted interviews and the user study the following set of requirements for Virtual Institutions in the E-Tourism domain was identified:

- **Requirement 1:** Virtual Institutions should provide both entertaining and informative immersive experience for potential tourists;
- **Requirement 2:** Virtual Institutions should include detailed visualizations of destinations, accommodations and activities at a destination;
- **Requirement 3:** Virtual Institutions should provide rich communication environment in which the context of a conversation is visualized;
- **Requirement 4:** Virtual Institutions should facilitate social interactions between travellers and travel agents;
- **Requirement 5:** Virtual Institutions should provide the possibility of collaborative booking;
- **Requirement 6:** Virtual Institutions should facilitate impulsive decisions;
- **Requirement 7:** Virtual Institutions should create automated facilities to discover customer preferences and offer them a personalized experience;
- **Requirement 8:** Virtual Institutions should provide low fares and last minute deals;

- **Requirement 9:** Virtual Institutions should offer facilities to collect the human expertise and offer customers human friendly interface to this information.

Further we describe how based on this requirements the specification of the Normative Control Layer of a Virtual Institution can be produced.

6.2.3 Developing the Normative Control Layer

The list of the requirements presented in the previous section was initially translated into a textual description of two prototypical scenarios: “Honeymoon trip” and “Weekend adventure”, which reflect the needs of the customers identified by the user study.

The scenarios form a basis for eliciting the specification requirements of the Normative Control Layer and further implementation of the Virtual Institution prototype (the “World Trotter” travel agency).

The “World Trotter” Travel Agency: Scenarios Description

Scenario 1. Honeymoon Trip: *Michael and his wife are planning their honeymoon trip. They want to escape from a cold winter and go to some exotic destination, but are not sure which place to select. Their requirements are not very precise: they need a small but romantic room somewhere in a tropical paradise with palm trees on the beach, where they can enjoy warm temperatures and swim in the ocean. Searching the web with such vague requirements requires sophisticated search skills and is time consuming. Unfortunately, they currently work in different cities and it is hard to physically attend a travel agent’s office together and make use of the human expertise. Therefore, they decide to visit the “World Trotter” travel agency. After registering they both proceed to the booking room and start searching for the destination there. Michael has already been to the agency a couple of times, so the travel agent in the booking room knows his preferences. The preferences of his wife are yet to be discovered. After a couple of virtual tours to different places and local hotels they finally reach an agreement about a particular hotel on Tahiti. Newlyweds are satisfied; both his and her requirements are fulfilled and the price is very competitive. Michael exits and continues with his work, while his wife decides to spend some time in the virtual shop. As suggested by the travel agent, she participates in the free presentation about the bargaining techniques on Tahiti. She also walks through the shop to check the prices and see what can be bought or rented there, so that some space in their bags can be cleared.*

Scenario 2. Weekend Adventure: *David is up to a short adventure on the coming weekend. On Thursday he enters the “World Trotter” to see if he can acquire a cheap*

ticket to some unvisited destination in Australia. He registers and moves to the auction room. There he quickly finds a cheap last minute ticket to Darwin. In search for an accommodation he enters the booking room. There he describes his requirements and, together with the travel agent, visits a couple of hotels using the visualization service of the institution (see Figure 6.3). From the proposed choices, pretty much similarly priced, he selects a small but comfortable room close to the airport. The travel agent also makes a very competitive offer on the 4-wheel drive rental, which David accepts. To check that he didn't forget anything, David enters a virtual shop, which is a 3D visualization of one of the local Darwin stores. He starts a conversation with the shop assistant there. As he wants to visit the Kakadu National park, the shop owner recommends him to buy a particularly strong mosquito repellent. David puts the mosquito repellent into the shopping basket and continues walking around the store. He finds himself a nice hat, sunscreen, water and some snacks for the long driving trip. On arrival all the purchased products will be delivered to David's hotel room.

These scenarios reflect the key activities that the prototype should support. In order to use the Virtual Institutions Methodology in the development of the “World Trotter” travel agency the above scenarios have to be decomposed into the main elements of the Electronic Institutions specification (scenes, roles etc.). Next, we outline how this decomposition was done and explain the reasoning behind it.

Roles

Once the specification requirements are produced the Step 2 of the Virtual Institutions Methodology (see Figure 4.1) is executed to conduct the specification of the Normative Control Layer.

The first step in specifying the Normative Control Layer of a Virtual Institution is determining the set of roles that users are allowed to play. In this context a role defines a pattern of behavior establishing which actions participants playing the role will be allowed to do. The following *roles* can be identified from the aforementioned scenarios:

- *Customer (C)*: participants entering the institution to acquire travel products.

Participants playing the “Customer” role are always present in both online booking portals and brick and mortar travel agencies. Although, this role is not clearly stated there, such participants are associated with a clearly defined set of activities that they are allowed to take part in, as well as there are activities that customers can not be involved in. Introducing a “Customer” as a separate role into a Virtual Institution is an efficient way to provide such functionality.

- *Travel Agent (TA)*: devoted to assist customers in looking for destinations and products and booking them.

As the key intention behind the “World Trotter” prototype is to introduce human expertise into online portals in a similar way as it is done in brick and mortar – the choice of the “Travel Agent” role is also obvious.

- *Receptionist (R)*: in charge of controlling which customers enter the institution.

It’s a common practice in brick and mortar institutions to have a receptionist, who is responsible for providing information about the functions of the institution to the customers and validating the access of the customers to different activities of the institution. The metaphor of Virtual Institutions also suggests that receptionist should be a common practice. The functions of this role are very similar to the functionality of the receptionist agent in the Trading Institution (presented in Chapter 3).

- *Auctioneer (A)*: in charge of governing the auction.

As our scenarios include the possibility to auction flight tickets and travel packages as an activity separate from the general booking, a separate institutional role is required to organize and monitor the auctions. This role is “Auctioneer”.

- *Shop Assistant (SA)*: in charge of managing the shop.

In every brick and mortar shop there is usually a shop assistant. That’s why we selected it to be present in the Virtual Institution as well. As this activity is not directly related to the activities conducted by other roles, we find it necessary to provide a separate “Shop Assistant” role for this task.

Once the roles are defined we have to distinguish between internal and external roles. An institution delegates their services, duties and tasks to internal agents. Only tasks which can be clearly associated with representing the institution can be assigned to them. The only role, whose activities can not be associated with representing the institution is the “Customer” role. Hence, in the “World Trotter” travel agency there is only one external role, the “Customer”, while the rest are internal roles.

Scenes

Next important step is defining activities that can happen in a Virtual Institution and can be visualized as different rooms. The activities in an institution are the composition of multiple, distinct, and possibly concurrent, dialogical activities, each one involving different groups of participants playing different roles. For each activity, interactions

between participants are articulated through scenes that follow well-defined communication protocols. The World Trotter Institution has the following *scenes*.

Reception Scene: where customers are asked to identify themselves by their login and password. This scene should be visualized as a separate room, which is controlled by an agent playing the “Receptionist” role. In this scene customers can also ask the receptionist about the different scenes in the institution and their location. The Reception room inside the institutional building, that will correspond to this scene will be the entrance point into the institution. As it was said before, we suggest having such a room in every Virtual Institution as a common practice.

Booking and Destination Search Scene: where customers can search and book travel products through interaction with a travel agent. In this scene customers may use travel agent’s expertise to select their trip. They can enquiry about possible destinations and activities available there. They can also move to a virtual representation of a destination (such as Virtual Amsterdam presented in Figure 6.2). Moreover, the travel agent can also make suggestions or inform them about current special deals taking into account customers’ preferences. This scene covers all the issues related to the searching and booking of travel products except for the hotel selection, which is carried out in the Hotel Room scene.



Figure 6.2: Virtual Amsterdam Tour as Implemented in Second Life Technology.

As in brick and mortar travel agencies customers usually have personalized experience with a travel agent – the Virtual Institution should support multiple executions of the Booking and Destination Search scene. Each scene execution (instance) will be visualized as a separate room. Creating a new scene instance corresponds to adding another floor in the institutional building.

Each instance of the Booking and Destination Search is controlled by a single agent playing “Travel Agent” role.

This scene targets the key demands of the customers identified by the user study: human expertise, collaborative booking, detailed visualization, social interaction, help with impulsive decisions etc.

Hotel Room Scene: helps customers to book the accommodation. As a starting point, a customer is presented with an opportunity to explore a 3D reproduction of a suggested hotel room. If the user is satisfied with it, the travel agent can book it straight away. If not, another hotel room will be presented to the customer. The process continues until the customer finds an appropriate accommodation or decides to leave the scene without booking a hotel. While searching through hotels customers can inform the travel agent about their preferences and explain why they do not like a particular offer. Thus, the travel agent can use this information when selecting the next hotel room to display (the 3D representation of the Hotel Room Scene is presented in Figure 6.3).



Figure 6.3: Hotel Visualization Service.

This scene also should support multiple execution. It is created and destroyed on demand and is also controlled by a single agent playing the “Travel Agent” role. The Travel Agent and the Customer will physically move from the Booking and Destination Search room into the Hotel Room and the Travel Agent will conduct a guided tour there.

The presence of this scene is stimulated by a strong need of the customers for precise information about the accommodation before booking it (expressed in the literature [30, 106] and in the responses of the user study participants).

Auction Room Scene: where last minute products (flights, hotels etc.) are sold by the auctioneer following the downward bidding protocol.

This scene is controlled by the Auctioneer. We identified the need for introducing such a scene by observing (in the literature [23, 112] and through the study) a strong desire of the customers for cheap fares and bargains.

Shop and Renting Scene: where users can buy or rent useful items for their trip. The decoration and the goods shown in the shop depend on the destination. Customers choose between getting acquired goods delivered to their home or to specific points of their destination (e.g. to their hotel room).

This scene also requires to support multiple execution, so that different destinations are represented by at least one scene. An agent playing the “Shop Assistant” role is in charge of this scene.

The demand for this scene is identified in the literature. Shopping is an increasingly important leisure and tourist activity. Researchers report that about two-third of visitor expenditure in case of domestic travel is on shopping [83]. In the case of overseas travel Australian Government report [186] suggests that shopping accounts for approximately one-fifth of overseas visitor expenditure.

As the user study showed that most of the customers are convenience-oriented, we supplied this scene with the convenience element, which is allowing customers to have purchased products delivered to a desired location.

Performative Structure

Figure 6.4 outlines one of the results of eliciting the specification requirements. It depicts the Performative Structure of the World Trotter Institution, where rectangles represent scenes and semicircles (synchronization and parallelization points) and triangular shapes (choice points) are transitions.

Labels on the arcs determine which types of participants can move to which scenes. A label “x:TA” above an arc connecting a scene and a transition means that a participant playing the Travel Agent role can progress through that arc and enter the scene or transition it leads to.

Besides the scenes presented above, Figure 6.4 also includes the root and exit scenes, which represent the institution entrance and exit.

Notice that customers (C) are required to start in the Reception scene, where they have to identify themselves. After the registration, they can progress to the Auction Room, the Booking and Destination Search scenes or exit the institution. Once in any of them customers can freely move among these two scenes and can also visit Hotel Room and Shop scenes.

In the case of internal roles, receptionists (R) can go to the Reception, auctioneers (A) to the Auction Room, shop assistants (SA) to the Shop, while travel agents (TA) can

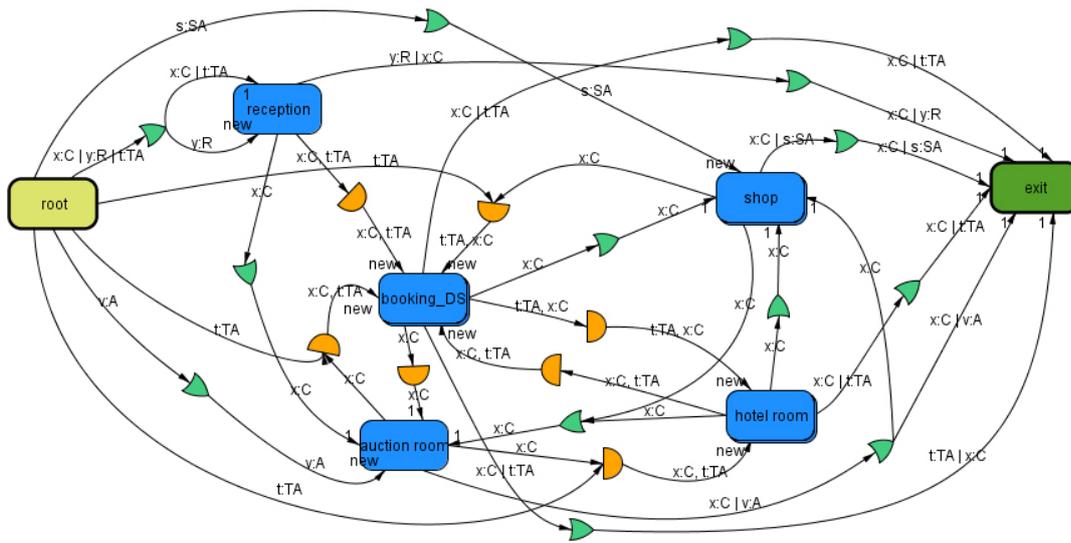


Figure 6.4: Performative Structure of the World Trotter Institution.

access the Reception, Booking and Destination Search and Hotel Room scenes.

As the key focus of this section was on showing how to elicit specification requirements, presenting the complete details of the specification of the World Trotter Institution is beyond the scope of our presentation. Here we only focused on defining the main specification elements (roles and scenes). The remaining specification steps can be easily conducted once these key elements are identified. The detailed explanation of all the aspects of the specification of the Normative Control Layer of a Virtual Institution is given in Chapter 2 (Section 2.5.4).

On Step 3 of the methodology the resulting specification should be verified for correctness by ISLANDER before further steps can be executed.

Next, we show how the Visual Interaction Layer is created based on the resulting verified specification.

6.2.4 Developing the Visual Interaction Layer

Once the specification of the Normative Control Layer is completed and verified, Step 4 of the Virtual Institutions Methodology is applied to automatically transform the Performative Structure into a map of the institutional building. Figure 6.5 outlines the result of this transformation. The automatic generation was conducted in an Euclidean mode and transitions were selected to be omitted from the resulting map.

With the help of the World Generator tool (see Chapter 4, Section 4.2.1) the map from Figure 6.5 is automatically transformed into the corresponding Virtual World.

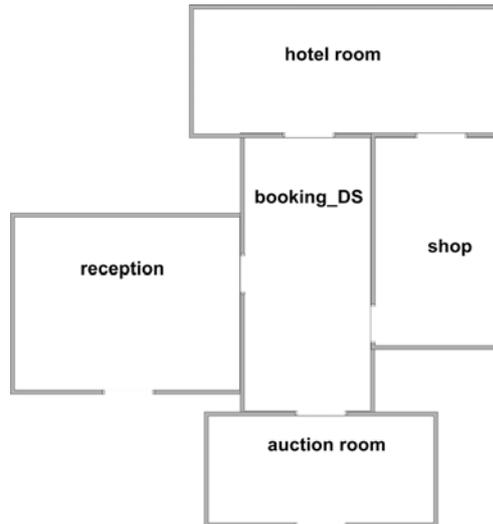


Figure 6.5: The Map of the World Trotter Institution.

To enrich the generated rooms with appealing visualization on the Step 5 of the Virtual Institutions methodology the rooms are annotated with additional visual content. This content may address the requirements for the “interior design” of the spaces, coming from the clients (i.e. travel agents). The result of such annotation for one of the rooms (Booking Room) is shown in Figure 6.6.



Figure 6.6: Booking Room.

When the annotation is completed the Step 6 (Integration) is executed to connect the Normative Control Layer and Visual Interaction layer. Then, to enable the implicit

training of the shopping assistant Step 7 (Enabling Implicit Training) is also performed.

The details of these steps are given in Chapter 4 and Chapter 5 correspondingly.

6.3 Summary

In this chapter we have explored using Virtual Institution in E-Commerce. The shortcomings of existing E-Commerce systems include: insufficient product presentation, poor support of social interactions, lack of facilities to support impulsive decisions, limited advertisement possibilities, poor support of collaboration shopping, inconsistency throughout different interfaces, limited information presentation facilities, lack of shopping assistants, poor shopping experience and no support for non-verbal clues.

It has been showed how these drawbacks can be addressed with Virtual Institutions and outlined additional benefits that this technology can bring into E-Commerce.

Through detailed literature review we have narrowed the E-Commerce domain down to E-Tourism. E-Tourism, in our opinion, is the subdomain of E-Commerce where the shortcomings are the most evident and the benefits of using Virtual Institutions are significant.

To provide a proof of concept for the Virtual Institutions Methodology this methodology was applied to the development of a prototype of a Virtual Institution applicable to the domain of E-Tourism. Through a detailed literature review and the outcomes of the conducted user study we have collected a set of requirements for implementing an online travel agency as a Virtual Institution. Based on these requirements a scenario for the prototypical World Trotter Travel Agency was produced.

It has also been showed how the institutional specification for the World Trotter Travel Agency can be derived from the resulting scenarios. The relevant details of each of the Virtual Institutions Methodology steps as applied to completing the implementation of the World Trotter prototype were outlined.

Chapter 7

Conclusion and Future Work

In this thesis we have introduced the concept of Virtual Institutions, which are 3D Virtual Worlds with normative regulation of participants' interactions. The concept of Virtual Institutions consists of two independent logical levels: Visual Interaction Layer (responsible for the visualization of participants and for providing them with interaction facilities inside the space of the Virtual Institution) and Normative Control Layer (responsible for establishing and controlling the enforcement of the interaction rules within the Virtual Institution).

The design of these two logical levels is covered by the Virtual Institutions Methodology. To support this methodology the thesis has developed a feasible and economically efficient technological solution, which utilizes two existing technologies - Electronic Institutions technology and Virtual Worlds technology.

Virtual worlds have been developed on an ad-hoc basis. Virtual Institutions is the first formal methodology that addresses in a more systematic way the development of Virtual Worlds. As a formal methodology, it has a number of advantages. Firstly, employing formal methods forces the system designer to analyze the system in details before implementing it. Secondly, it permits to detect the critical points and errors at an early stage. Thirdly, the methodology clearly distinguishes between the design of institutional rules and the design of its visualization in Virtual Worlds, so that these two processes can be run in parallel. Another advantage of using this methodology is that the supplied tools make the development faster, helping to achieve some tasks automatically. Moreover, due to the distributed architecture possible updates of the system can be accommodated in an easy way. This in combination with the execution infrastructure permits a quick and easy portability of the system to new visualization platforms.

One of the novel ideas of the deployment architecture proposed in this thesis is that every avatar is treated as a combination of two entities: human and autonomous agent. When the agent controls the avatar, the human can conveniently observe its actions and intervene when necessary. In this way the behavior of the agent acting on user's behalf becomes more transparent, increasing the trust and confidence of the humans in the agent. Such approach permits introducing the implicit training of autonomous agents as a method for extending agent's behavioral intelligence. By implicit training we mean that

agents are trained to act like humans by observing humans, without any explicit training efforts required from the humans. Being the integral part of the overall methodological approach, implicit training helps to further shift the focus of the developers away from the agent-centred view on the development of MAS. It is suggested that programming of the agent's behavior is complemented with implicit training, and in some cases completely replaced by implicit training. The immersive nature of 3D Virtual Worlds creates better possibilities for agents to observe human behavior without a need to overcome the embodiment dissimilarities, while institution control of the interactions helps the agent to reduce the number of possible behaviors and learn faster.

The validity of the Virtual Institutions concept has been tested on applying it to the area of E-Commerce, in particular to the domain of online travel agencies. In this domain advanced visualization of Virtual Institutions helps to achieve better observation of the purchased goods and services; the support of social interactions has a potential to create more comfortable environments, while strict institutional moderation of security-related issues will make those environments a safe place for trading and leisure. The similarities with real world, in this case, will facilitate the participation of new users through merging Virtual Institutions with existing 3D Virtual Worlds communities.

3D Virtual Worlds is a well established phenomenon with thousands of existing environments and millions of inhabitants. Institutionalizing 3D Virtual Worlds and bringing some of existing institutions there, for example, E-Commerce shops and travel agent facilities, would mean moving the MAS-based technology to where the users already are, rather than waiting for the users to join.

7.1 Future Work

The introduced concept of Virtual Institutions requires further validation. The Trading Institution prototype demonstrated a proof of concept. Collecting evidence regarding the benefits of using Virtual Institutions for commercial applications requires the development of an industrial strength system and evaluating its application in one of the domains, for example, virtual travel agency. To gain such evidence we initiated a project that aims at developing the World Trotter travel agency presented in Chapter 6. We plan to start our research in this direction from the development of a comprehensive prototype. Once the prototype is ready, we plan to conduct an extensive usability study that will evaluate the acceptance and feasibility of the Virtual Institutions technology as applied to E-Tourism.

7.1.1 Design Grammars

The development of Virtual Institutions architecture requires further refinement. The Causal Connection Server is technology independent and functional. Some components in the prototype were developed with a strong focus on the employed rendering engine - in this prototype, Adobe Atmosphere. The choice of Adobe Atmosphere at the time was based on the readiness of the technology and the range of customization that it offered. Unfortunately, its development has been discontinued. This prompted the exploration of further refinement of the level of independency, looking at the possibility to plug-unplug rendering engines without massive change of the underlying code supporting the Visual Interaction Layer of a Virtual Institution. After a detailed investigation we discovered an approach that can help to avoid technological dependency. With the use of design grammars it is possible to specify appearances and behavior of the objects in the Visual Interaction layer. Designs specified through design grammars are fully independent from rendering and would only require a small programming effort to be transferred to any desired rendering platform. We initiated collaboration with the group of Mary Lou Maher from the University of Sydney with the aim to conduct the automatic generation of Virtual Institutions via design grammars.

7.1.2 Gestures

We also plan to further explore the utilization of gestures and other social cues in Virtual Institutions. Currently, avatars in Virtual Worlds are equipped with some set of general gestures or ways of expressing some social cues (for example, waving, jumping and spinning to express joy, dancing movements, etc.). The concept of Virtual Institutions offers the opportunity to utilize a set of professionally specialized gestures as part of the Visual Interaction layer. The learning mechanism associated with the dual human/agent control of the avatar allows to associate the use of specific gestures to a particular person, scenes, or, class of scenes and type of people. This can make possible the anticipation and dynamic customization of the gesture sets associated with different individuals.

7.1.3 Institutional Rules and Natural Language

In the thesis we presented our vision that a governor agent could be visualized as an avatar and may be employed to explain the institutional rules to the human participants. In order for this to be possible, it is necessary to have a mechanism of translation of the institutional specification into natural language. This task is quite difficult and requires significant research effort. Ultimately, it would also be useful to be able to translate rules expressed in the natural language into the formal form acceptable by the electronic

institution software, supporting the normative layer. This would be quite helpful for supporting the evolution of Electronic Institutions.

7.1.4 Introducing Implicit Social Conventions

The employment of Electronic Institutions technology provides mechanism for establishing and enforcing explicit social conventions. However, there is no clear mechanism available in Electronic Institutions for establishing implicit social conventions. Further investigation is required to understand whether implicit social conventions should be strictly enforced, and if yes – what are the mechanisms for expressing and enforcing them.

7.1.5 Evolution of Virtual Institutions

As human societies evolve, the rules regulating the interactions in these societies also change. This process is continuous. There are even institutions (like parliament) which employ people whose task is to improve the existing rules and create the new ones.

Although, Virtual Worlds to a very high degree reflect human societies, rule evolution is almost not present there. The rules of Virtual Worlds are introduced by the developers and remain almost unchanged throughout the life cycle of the system.

The employment of Virtual Institutions has a potential to change the situation. With the help of this technology the rules can be formally expressed and strictly enforced. The changes of the rules can also be introduced in a quite efficient way (as it is only required to change the ISLANDER specification). The research on the narrative even sequence analysis [35] is a good starting point in exploring the evolution of the institutional rules.

The evolution of the institutions also brings some “deep” issues. Namely, who is responsible for introducing the rules of virtual societies? There is an ongoing debate regarding this topic and most of the researchers are convinced that the participants themselves should choose these rules through democratic procedures. Implementing democratic procedures in Virtual Worlds is quite a challenge. However, we are convinced that the fact that with Virtual Institutions the rules are expressed in a formal way makes this challenge feasible.

Automatic evolution of Virtual Institutions is one of the most interesting challenges. How the visualization of the Virtual World could automatically adapt to the needs of participants inside was studied by [86, 44]. The availability of Virtual Institutions also raises the issue of automatic adaptation of the institutional rules to the behavior of participants.

7.1.6 Normative Virtual Environments

One of the significant technological contributions of this thesis is the development of the Causal Connection Server. This component acts as the middle layer, which glues together two concepts: 3D Virtual Worlds and Electronic Institutions. Apart from technological advantages this technology has a high potential to also make a significant contribution to research.

We plan to develop the combination of Electronic Institutions + Causal Connection Server into a new paradigm that we call Normative Virtual Environments. Normative Virtual Environments should be seen as an infrastructure capable of establishing interaction rules in any domain, whether it is 3D Virtual Worlds, form-based systems or even the real world. Moreover, through the Causal Connection Server it is possible to use the Electronic Institutions infrastructure on several domains simultaneously. We plan to build on top of the work of [85], who suggest using Electronic Institutions in ubiquitous environments. We see the Causal Connection Server being a technology that could facilitate this process and make Normative Virtual Environments a reality.

As an example of the possibilities that arise with Normative Virtual Environments consider a Fish Market. Electronic Institutions technology is currently used to control the interactions of fish buyers and sellers in a real fish market in Barcelona [43]. The participants enter the Electronic Institution as agents, which are controlled in real world by humans. Humans interact with the corresponding agents through handheld devices similar to remote control facilities of a television set. There is also a fully computer-based version of a fish market (also based on Electronic Institutions) where humans conduct their trading through form-based interfaces. In this thesis we presented the prototype of the Trading Institution, one of the rooms in which acts as a fish market auction with similar protocol as in the previous two examples.

Through the employment of Normative Virtual Environments it could be possible to create a joint environment, the participants of which could participate in it through special devices in real world, through avatars in a 3D Virtual World and through a form-based interface. The Causal Connection Server would reflect the institutional actions onto all of the connected interfaces in a consistent manner.

7.1.7 Learning

Another research direction spanning from the thesis is the implicit training approach (see Chapter 5). The concept of implicit training is used for teaching human behavioral characteristics to autonomous agents in Virtual Institutions. The developed prototype confirmed the workability of the selected method for the trajectory classification and

supported the validity of the implicit training concept.

In its current version the presented learning method was not able to generalize and handle the situations with dynamically moving agents. However, we insist that the prototype can be easily extended to address this issue.

In order to be able to handle the situations with dynamically moving avatars we propose to add two more levels of abstractions of the training data and conduct the training on each of those levels. On the first level the initial trajectory will be discretized by a sequence of textual actions that represent simplest moves, turns, jumps etc. The next level of abstraction will include the actions of the higher level like approaching, leaving, etc. In this way not only dynamic objects will be handled properly, but also the learned data from the higher levels of abstraction can be successfully transferred and utilized in other institutions. We plan to investigate the adaptation of the information pyramid approach presented in [22] to introduce different levels of abstraction.

We also consider replacing the simple probability-oriented approach for selecting the sequence of the institutional messages by a bayesian approach in a similar way as it is planned for the social level actions.

The research on prediction of the cognitive state of humans presented in the thesis is in its preliminary stage. In particular, at present the comparison models for the trajectories that represent a particular cognitive state are manually created and all the labels are manually assigned. Further investigation is required in this direction. Moreover, we analyzed only one aspect of the cognitive state, while there are many more aspects that should be studied.

For the recognition of trajectories representing the “mood” of the buyers we only analyzed geometrical features, while it might be also useful to experiment with non-geometrical features. Such features for a particular scenario presented in the thesis may include: a number of stops before approaching the assistant, an average length of the trajectory segment, a number of direction changes etc. Such information can be easily obtained and analyzed using standard machine learning algorithms.

The implicit training method in the present stage only allows for executing an action by an agent when the principal conducted this action at least once. We plan to improve this situation by supplying agents with a simple navigation possibilities, so that they can perform simple transitions between learning graph nodes by themselves.

The simplistic method for the evaluation of agent’s movement believability employed in Section 5.4 was suitable for the purpose of the section, but can not be used as an actual measure of believability. In the future we plan to develop a Turing test variant to evaluate whether the behavior of computer-controlled avatars can pass as believable to human observers. The analysis will be based on a rich data set that includes a full

recording of the activities of every participant in the given environment, the states of the environment and its responses. The experimental designs will include interaction between a tested human from the sample and some amount of participants among which all are computer-controlled (that may operate at different levels of believability) except one, which is a human-controlled participant. The believability score is then formed by asking the test human to evaluate her/his perception of each of the other participants as human-controlled on confidence scale. Then the evaluation score for each participant is subtracted from the evaluation score assigned to the human participant to produce behavioral believability rating for every computer-controlled participant. These scores will be integrated with the data set and analyzed further. We also plan to use qualitative evaluation of the experience in the experiments.

Another direction we would like to investigate is the agent-human side of co-learning. This aspect was only briefly mentioned in the thesis, while only the problem of teaching autonomous agents to believably imitate humans was properly examined. One of the future directions of our research is to use the autonomous agents, which have learned a particular behavior from humans who are experts in a given area, to teach similar behavior to the humans who have no (or limited) expertise in this area. For example, Tiger Woods could teach an agent to play golf and this agent can be then further used to teach other people to play Tiger Wood's way. In a similar way, a junior combat team could be trained by a more experienced combat team through a team of agents.

7.1.8 Agent Programming

We consider implicit training being quite helpful for teaching simple behaviors to internal institutional agents. However, we are fully aware that in many situations agent programming is required. There are currently no tools available, which could facilitate programming of the agents that participate in Virtual Institutions. Therefore, we are considering using the functionality of the ABuilder tool (see Section 2.5.3) for this purpose. To be used in Virtual Worlds this tool requires significant redevelopment.

7.1.9 World Trotter Institution Prototype

One of the goals of our future research is finalizing the World Trotter Institution prototype. While Chapter 6 illustrated the application of the Virtual Institutions methodology on the example of the "World Trotter" travel agency, the prototype itself is still in the preliminary stage of development. Some of its aspects require significant research and development efforts. In particular, we are investigating the problem of quick capturing of the look and feel of hotel rooms, and search for mechanisms of their dynamic visu-

alization. At this stage we are experimenting with the shape grammar based method of dynamic visualization proposed by [86]. For the destinations search task we are currently working on utilizing the existing virtual replications of the famous heritage sites created by the inhabitants of the Second Life Virtual World. As these sites are created by the third party, they are hard to institutionalize. Further research is required to find an efficient solution for conducting user observation and guided tours there.

7.1.10 Other Application Domains

In this thesis we have only focused on applying Virtual Institutions to the domain of E-Commerce (and its subdomain E-Tourism). Virtual Institutions is a very broad concept and is applicable to a much wider range of problems. In the future we plan to identify other application domains where the use of Virtual Institutions would be valuable.

Appendix A

Z Specification of Electronic Institutions

This appendix is based on [128]. Here we present Z specification of Electronic Institutions to provide a complete account of the essential data structures, state and operations. It can be used as a blueprint for implementation or the basis for future investigation into how EIs might be extended or deployed.

A.1 Electronic Institution Static Data Structures

A.1.1 Basics: Variables, Constants, Terms, Ontology, Language

First we define how terms are constructed.

Variables

We use different sets of symbols for different classes of object we wish to represent.

$[AgentSymbol, RoleSymbol, TimeSymbol, OntologySymbol]$

Each variable is either prefixed with bound (meaning it is equal to some constant) or free.

$Prefix ::= ! | ?$

$AgentVar == Prefix \times AgentSymbol$

$RoleVar == Prefix \times RoleSymbol$

$TimeVar == Prefix \times TimeSymbol$

$OntologyVar == Prefix \times OntologySymbol$

We can define the set of bound variables.

$$\begin{aligned}
BoundAgentVar &== \{a : AgentVar \mid first\ a = !\} \\
BoundRoleVar &== \{a : RoleVar \mid first\ a = !\} \\
BoundTimeVar &== \{a : TimeVar \mid first\ a = !\} \\
BoundOntologyVar &== \{a : OntologyVar \mid first\ a = !\}
\end{aligned}$$

And the set of free variables.

$$\begin{aligned}
FreeAgentVar &== \{a : AgentVar \mid first\ a = ?\} \\
FreeRoleVar &== \{a : RoleVar \mid first\ a = ?\} \\
FreeTimeVar &== \{a : TimeVar \mid first\ a = ?\} \\
FreeOntologyVar &== \{a : OntologyVar \mid first\ a = ?\}
\end{aligned}$$

The set of all bound variables is as follows.

$$\begin{aligned}
BoundVar ::= & boundavar \langle\langle BoundAgentVar \rangle\rangle \\
& \mid boundrvar \langle\langle BoundRoleVar \rangle\rangle \\
& \mid boundtvar \langle\langle BoundTimeVar \rangle\rangle \\
& \mid boundovar \langle\langle BoundOntologyVar \rangle\rangle
\end{aligned}$$

The set of all Free variables is as follows.

$$\begin{aligned}
FreeVar ::= & freeavar \langle\langle FreeAgentVar \rangle\rangle \\
& \mid freervar \langle\langle FreeRoleVar \rangle\rangle \\
& \mid freetvar \langle\langle FreeTimeVar \rangle\rangle \\
& \mid freeovar \langle\langle FreeOntologyVar \rangle\rangle
\end{aligned}$$

Last, we define the set of all variables.

$$\begin{aligned}
Var ::= & avar \langle\langle AgentVar \rangle\rangle \\
& \mid rvar \langle\langle RoleVar \rangle\rangle \\
& \mid tvar \langle\langle TimeVar \rangle\rangle \\
& \mid dvar \langle\langle OntologyVar \rangle\rangle
\end{aligned}$$

Constants

The constants are the agents, roles, constants of the ontology and natural numbers relating to time. These are the values that bound variables have.

$$\begin{aligned}
Time &== \mathbb{N} \\
& [AgentId, RoleId, OntologyConst]
\end{aligned}$$

Next the set of all constants.

$$\begin{aligned}
Const ::= & \text{aconst}\langle\langle AgentId \rangle\rangle \\
& | \text{rconst}\langle\langle RoleId \rangle\rangle \\
& | \text{tconst}\langle\langle Time \rangle\rangle \\
& | \text{cconst}\langle\langle OntologyConst \rangle\rangle
\end{aligned}$$

There are two special symbols, signifying the set of all agents and all roles.

$$\begin{array}{|l}
agentall : AgentId \\
roleall : RoleId
\end{array}$$

This allows for expressions that are essentially a Role Id prefixed by the constant *agentall*. For example, (*all, allroles*) refers to the set of all agents playing any role, and expressions such as (*all, buyer*) which is the set of all agents that are currently playing the buyer role.

Terms

Terms can be either variables or constants (identifiers).

- *AgentTerm* is the set of agent variables and agent identifiers.
- *RoleTerm* is the set of role variables and role identifiers.
- *TimeTerm* is the set of time variables and time values.
- *OntologyTerm* is the set of (ontology - content language) variables and values.

$$\begin{aligned}
AgentTerm ::= & \text{agentconst}\langle\langle AgentId \rangle\rangle | \text{agentvar}\langle\langle AgentVar \rangle\rangle \\
RoleTerm ::= & \text{roleconst}\langle\langle RoleId \rangle\rangle | \text{rolevar}\langle\langle RoleVar \rangle\rangle \\
TimeTerm ::= & \text{timeconst}\langle\langle Time \rangle\rangle | \text{timevar}\langle\langle TimeVar \rangle\rangle \\
OntologyTerm ::= & \text{domainconst}\langle\langle OntologyConst \rangle\rangle | \text{domainvar}\langle\langle OntologyVar \rangle\rangle
\end{aligned}$$

Ontology

The set of domain variables and constants is part of what defines the *ontology* of the electronic institution. An ontology language is a compound expression of ontology terms, which we specified above.

However, we do not specify a specific ontology here in order to maintain generality of this model.

The set of constructors enables us to define an Ontology Language which is made of a syntax based on constructors and ontology terms.

$$[Constructor]$$

An ontology language is made from a set of expressions, each of these expressions is an ordered pair which consists of a sequence of ontology terms and a sequence of constructors.

$$OntologyLanguage == \mathbb{P}(\text{seq } OntologyTerm \times \text{seq } Constructor)$$

An ontology theory is a set of elements of ontology languages.

$$OntologyTheory == \mathbb{P} OntologyLanguage$$

Typically, an EI has one pre-determined ontology which remains constant though the lifetime of the EI.

$$| \quad EIontology : OntologyTheory$$

A subset of an ontology language is ground.

$$| \quad GroundOntology : \mathbb{P} OntologyLanguage$$

Action Language

There is an action language which determine when an illocution happens how we update the *information model*, which we will define subsequently.

In fact *ALFormula* consists of constructors on the four kinds of terms specified earlier.

$$[ALFormula] \\ ActionLanguage == \mathbb{P} ALFormula$$

An EI has a chosen action language.

$$| \quad al : ActionLanguage$$

Some subset of action language formula are ground.

$$| \quad GroundALFormula : \mathbb{P} ALFormula$$

Constraint Language

There is a constraint language.

$$[CLFormula] \\ ConstraintLanguage == \mathbb{P} CLFormula$$

An EI has a chosen constraint language.

$$\begin{array}{|l} cl : \textit{ConstraintLanguage} \\ \\ \textit{GroundCLFormula} : \mathbb{P} \textit{CLFormula} \end{array}$$

A.1.2 The Social Model

This consists of a set of roles. Moreover, at design time, we specify some *fixed* relationships between the roles. The social model consists of these roles and a set of relationships between them. In the schema below we state that for any role related in a social relationship must be one of the previously identified roles of the system. We also state that every role must belong to at least one social relationship.

$$\begin{array}{|l} \textit{SocialModel} \\ \hline roles : \mathbb{P} \textit{RoleId} \\ \textit{socialrelationships} : \mathbb{P}(\textit{RoleId} \leftrightarrow \textit{RoleId}) \\ \hline \forall sr : \textit{socialrelationships} \bullet ((\textit{dom } sr) \cup (\textit{ran } sr)) = roles \end{array}$$

The Standard Social Model

In theory the designer can specify as many as they wish. In practice we have found the following to be useful.

1. The *subsumes* relation holds between a first role and a second role if all the illocutionary actions allowed in the second role are also allowed in the first.
2. The *ssd* exists between two roles that cannot be played by an agent simultaneously in an institution.

(In addition the *dsd* relation exists between two roles that cannot be played within a scene. We specify this later when we have define a scene in the schema *SocialModelState* later.)

We can specify some important properties of these relations.

1. The subsumes relation is a partial order. in the schemas below we assume the existance of a predicate *isPartialOrder* which is so standard that if we did use it we should put it in an appendix.
2. The exclusive (*ssd*) relation is symmetric.

3. If one role subsumes a second role, and the second is exclusive of the third, then the first is also exclusive of the third.
4. If one role subsumes another, then they cannot be exclusive of each other.
5. The *ssd* relation is irreflexive.
6. If two roles that are exclusive are respectively subsumed by two other roles, then these other roles are also exclusive.
7. If one role subsumes two other roles, then those other roles cannot themselves be exclusive of each other.
8. The intersubsection of *dsd* and *ssd* is always empty.

These are specified (in the order given above) below.

<p><i>EISocialModel</i></p> <hr/> <p><i>SocialModel</i></p> <p>$ssd, subsumes : RoleId \leftrightarrow RoleId$</p> <hr/> <p>$\{ssd, subsumes\} \subseteq socialrelationships$</p> <p><i>isPartialOrdersubsumes</i></p> <p>$ssd = ssd^{\sim}$</p> <p>$\forall r_1, r_2, r_3, r_4 : RoleId \bullet$</p> <p style="padding-left: 20px;">$(r_1, r_2) \in subsumes \wedge (r_2, r_3) \in ssd \Rightarrow (r_1, r_3) \in ssd \wedge$</p> <p style="padding-left: 20px;">$(r_1, r_2) \in subsumes \Rightarrow (r_1, r_2) \notin ssd \wedge$</p> <p style="padding-left: 20px;">$(r_1, r_1) \notin subsumes \wedge$</p> <p style="padding-left: 20px;">$(r_1, r_2) \in ssd \wedge (r_3, r_1) \in subsumes \wedge (r_4, r_2) \in subsumes \Rightarrow$</p> <p style="padding-left: 20px;">$(r_3, r_4) \in ssd \wedge (r_1, r_2) \in subsumes \wedge (r_1, r_3) \in subsumes \Rightarrow$</p> <p style="padding-left: 20px;">$(r_2, r_3) \notin ssd$</p>
--

A.1.3 The Communication Model

The communication model specifies the mechanisms for communication between EIs.

The basic template is the *IllocutionType*, which can be specified as an illocutionary particle (the basic illocution force of the communication), a formula from the ontology (content language), a time term, a sender agent and a receiver agent both with associated roles. The predicates state that the sender cannot be set equal to all agents, and that the sender and receiver agents must be distinct. The final predicate states that the formula must belong to the EIs ontology specified.

[*IllocutionParticle*]

IllocutionType

$illocutionparticle : IllocutionParticle$
 $formula : OntologyLanguage$
 $time : TimeTerm$
 $sender, receiver : AgentTerm$
 $sendrole, receiverole : RoleTerm$

$sender \neq (agentconst\ agentall)$
 $sender \neq receiver$
 $formula \in EIontology$

An *illocution scheme* contains variables. More specifically, the time term, the sender (agent) term and the receiver (agent) term must all be variables.

IllocutionScheme

IllocutionType

$\{sender, receiver\} \subseteq (\mathbf{ran}\ agentvar)$
 $time \in (\mathbf{ran}\ timevar)$

If all the terms in an illocution type are *constants* then the illocution type is referred to as an *illocution*.

Illocution

IllocutionType

$\{sender, receiver\} \subseteq (\mathbf{ran}\ agentconst)$
 $\{sendrole, receiverole\} \subseteq (\mathbf{ran}\ roleconst)$
 $time \in (\mathbf{ran}\ timeconst)$
 $formula \in GroundOntology$

Finally we define an *free illocution* as an illocution type that contains no bound variables. In other words all variables are either constants, or if they are variables then they must be *free* variables.

FreeIllocution

IllocutionType

$(sender \in (\mathbf{ran}\ agentvar)) \Rightarrow (agentvar \sim sender) \in FreeAgentVar$
 $(receiver \in (\mathbf{ran}\ agentvar)) \Rightarrow (agentvar \sim receiver) \in FreeAgentVar$
 $(sendrole \in (\mathbf{ran}\ rolevar)) \Rightarrow (rolevar \sim sendrole) \in FreeRoleVar$
 $(receiverole \in (\mathbf{ran}\ rolevar)) \Rightarrow (rolevar \sim receiverole) \in FreeRoleVar$
 $(time \in (\mathbf{ran}\ timevar)) \Rightarrow (timevar \sim time) \in FreeTimeVar$
 $formula \in GroundOntology$

The Communication Model defines the set of illocution schemes. Naturally, any formula in any of these schemes must belong to the pre-defined EI ontology. We also define the dummy variable *iparticles* which will give you the set of all basic illocution particles in the EI.

<p><i>CommunicationModel</i></p> <p><i>communicationlanguage</i> : \mathbb{P} <i>IllocutionScheme</i></p> <p><i>iparticles</i> : \mathbb{P} <i>IllocutionParticle</i></p> <p>$\forall i : \text{communicationlanguage} \bullet i.\text{formula} \in \text{EIOntology}$</p> <p><i>iparticles</i> = $\{i : \text{IllocutionScheme} \bullet i.\text{illocutionparticle}\}$</p>

A.1.4 The Normative Model

With each electronic institution the designer may specify a model of norms at the declarative level. (Equally this might be specified at an operational level which we would then specify with the operational semantics.)

We provide a model at the highest possible level of abstraction.

[*Norm*]

<p><i>NormativeModel</i></p> <p><i>norms</i> : \mathbb{P} <i>Norm</i></p>
--

A.1.5 The Performative Model

Scenes

An electronic institution comprises several scenes through which agents pass whilst involved in an EI. An individual scene is made up of *conversation states*. At any time during the life of a scene, the scene will be in one of these conversation states. Transitions between these states will be achieved by agents uttering illocutions as we defined in the communication model. A scene is essentially a directed graph of *conversation states*, where the arcs are referred to as *dialogue moves*.

[*ConvState*]

Scene Types

A scene type contains the following data structures:

1. a set of role identifiers, called the *scene roles*
2. a set of *illocution particles* (which is a subset of those defined in the communication model)
3. the set of *all* scene conversation states, that we sometimes refer to as scene states;
4. one *initial* conversation state; which defines where the agents who create a scene (instance) must join
5. a set of *closing* conversation states; these conversational states are those from which a set of agents may close down a scene;
6. for each scene role, a set of *access* conversation states, which states where an agent with a specific role may join once the scene is up and running;
7. for each role in the scene roles, a set of *leaving* conversation states, where an agent with that role may leave the scene;
8. a set of directed edges called *dialogue moves* between conversation states;

$$DialogueMove == ConvState \times ConvState$$

9. a function that labels each of the scenes dialogue moves with either an illocution scheme (not grounded), set of constraints (from our constraint language) and a set of actions (from our action language). We call these *DMLabels*.

$$DMLabel ::= label\langle\langle Label \rangle\rangle \\ | \quad timeout\langle\langle TimeOut \rangle\rangle$$

10. for each scene role, the minimum and maximum number of agents that can play that role, *agentnumbers*. (Note that as this necessarily returns a non-empty set of natural numbers as there must be at least one legitimate value for the number of agents in a scene.)

Timeouts are natural numbers.

$$TimeOut == \mathbb{N}$$

We introduce labels for dialogue moves in a scene, as described above. Each label that is associated with a dialogue move consists of an illocution scheme, a set of constraints (which should be interpreted as conjunction), and a sequence of actions (that

should be performed individually and in the order as described by the sequence when the dialogue move is successfully traversed).

<i>Label</i>
<i>ischeme</i> : <i>IllocutionScheme</i>
<i>constraints</i> : \mathbb{P} <i>CLFormula</i>
<i>actions</i> : <i>seq ALFormula</i>

Scenes can then be specified in the schema below, with the following predicates:

1. the initial and closing conversation states, are all contained in the set representing all the scene's states;
2. the access, leaving and stay-and-go conversation states, for all roles, are all contained in the set representing all the scene's states;
3. any role associated with an access, leave or stay-and-go state must be contained in the scene's roles;
4. dialogue moves must only connect conversation states belonging to the scene;
5. all conversation states must be reachable from the initial conversational state;
6. for each role there must be a path from any access conversation state to at least one closing conversation state;
7. there are no dialogue moves into the initial conversation state;
8. there are no dialogue moves out from any closing conversation state;
9. the directed graph of conversation states and dialogues moves is connected;
10. each dialogue move is labeled;
11. the *agentnumbers* function (the range of the number of agents that can play a role) is defined for all roles in the scene roles.

SceneType

$sceneroles : \mathbb{P} \text{ RoleId}$
 $illocutions : \mathbb{P} \text{ IllocutionParticle}$
 $allconvstates : \mathbb{P} \text{ ConvState}$
 $initconvstate : \text{ConvState}$
 $closingconvstates : \mathbb{P} \text{ ConvState}$
 $accessconvstates, leavingconvstates, stgoconvstates : \text{RoleId} \mapsto (\mathbb{P} \text{ ConvState})$
 $dmoves : \text{ConvState} \leftrightarrow \text{ConvState}$
 $label : \text{DialogueMove} \mapsto \text{DMLabel}$
 $agentnumbers : \text{RoleId} \mapsto \mathbb{P}_1(\mathbb{N})$

$(\{initconvstate\} \cup closingconvstates) \subseteq allconvstates$
 $\bigcup(\text{ran } accessconvstates \cup \text{ran } leavingconvstates \cup \text{ran } stgoconvstates) \subseteq allconvstates$
 $(\text{dom } accessconvstates \cup \text{dom } leavingconvstates \cup \text{dom } stgoconvstates) \subseteq sceneroles$
 $(\text{dom } dmoves \cup \text{ran } dmoves) = allconvstates$
 $\forall cs : \text{ConvState} \mid cs \neq initconvstate \bullet$
 $\quad cs \in (\text{ran}(\{initconvstate\} \triangleleft dmoves)\star)$
 $\forall r : \text{RoleId}; cs_1 : \text{ConvState} \mid cs_1 \in (accessconvstates \ r) \bullet$
 $\quad \exists cs_2 : closingconvstates \bullet (cs_1, cs_2) \in dmoves\star$
 $initconvstate \notin (\text{ran } dmoves)$
 $closingconvstates \cap (\text{dom } dmoves) = \{\}$
 $\forall s_1, s_2 : allconvstates \bullet (s_1, s_2) \in (dmoves \cup dmoves\sim)\star$
 $\text{dom } label = dmoves$
 $\text{dom } agentnumbers = sceneroles$

Scenes

We can now define a scene, which consists of a name, the type, and a flag which determines whether multiple instances can be instantiated as a scene instance.

[*SceneName*]

Scene

$name : \text{SceneName}$
 $type : \text{SceneType}$
 $multiple : \text{Bool}$

Transitions

A transition is defined solely by its type.

$\text{TransitionType} ::= \text{and} \mid \text{or} \mid \text{xor}$

<i>Transition</i> <i>type : TransitionType</i>

Arcs connect scenes to transitions (which we call outarcs) and transitions to scenes (inarcs). These arcs may have (optionally) a type.

ArcType ::= one | some | all | new

<i>Arc</i> <i>transition : Transition</i> <i>scene : Scene</i> <i>type : optional [ArcType]</i>
--

In fact, inarcs have a type, and outarcs do not. This allows us to distinguish them.

<i>InArc</i> <i>Arc</i>
defined <i>type</i>

<i>OutArc</i> <i>Arc</i>
undefined <i>type</i>

Unfolding the Performative Model

The (base) Performative Model (PM) captures causal dependencies between scenes indicating order, synchronization of scenes, parallelism, choice points (between joining scenes on leaving another), flow, or change of roles between scenes. A PM contains not just scenes but transitions (the way an agent in a role can move between scenes). Each scene may be connected to multiple transitions, and each transition to multiple scenes. However, there are no direct connections between scenes themselves, or between transitions. A Performative Model has the following data components:

1. a finite non-empty set of scenes;
2. a finite non-empty set of transitions;
3. one entryscene and one exitscene

4. a *non-empty* set of arcs that are links from either:
 - scenes to transitions (outarcs) or
 - transitions to scenes (inarcs);
5. a first labelling function (disjnorm), mapping each arc to a disjunctive normal form of agent variables and role identifiers, and defining the possible set of agents and associated roles that can simultaneously traverse the arc. (Note, that this set must contain at least one set, and that set must contain at least one ordered pair which means that we have a non empty set, and that any element in that set must also be non-empty as specified.)
6. a second labelling function (constraints), that maps arcs to constrains (in our constraint language), which individual agents must fulfil in order to traverse the arc.

Defining meta-language constraints and sets of booleans as given sets, each of these components is included, in order, in the schema below. The predicates describe the following constraints of the state variables:

1. the entry scene and the exit scene are distinct;
2. the entryscene and exitscenes are in the set of all the PM scenes;
3. the inarcs are those arcs with a defined type;
4. the outarcs are those arcs with an undefined type;
5. every PM arc is labeled with a (non-empty) disjunctive normal form of agent variables and role Ids;
6. every arc are labeled with a set of (possibly empty) constraints;
7. it is not possible to return to the entryscene; and
8. it is not possible to leave the exitscene (and return to another scene within that PM).

PerformativeModel

$allscenes : \mathbb{P}_1 \text{ Scene}$
 $transitions : \mathbb{P}_1 \text{ Transition}$
 $entryscene, exitscene : \text{Scene}$
 $arcs, inarcs, outarcs : \mathbb{P}_1 \text{ Arc}$
 $disjnorm : \text{Arc} \leftrightarrow \mathbb{P}_1(\mathbb{P}_1(\text{AgentVar} \times \text{RoleId}))$
 $constraints : \text{Arc} \leftrightarrow (\mathbb{P} \text{ CLFormula})$

$entryscene \neq exitscene$
 $\{entryscene, exitscene\} \subseteq allscenes$
 $inarcs = \{a : arcs \mid \text{undefined } a.type\}$
 $outarcs = \{a : arcs \mid \text{defined } a.type\}$
 $\text{dom } disjnorm = arcs$
 $\text{dom } constraints = arcs$
 $\neg (\exists a : \text{InArc} \bullet a.scene = entryscene)$
 $\neg (\exists a : \text{OutArc} \bullet a.scene = exitscene)$

A.1.6 Standard Constraints on Performative Models

There are several constraints on a PM that we need to define. In order to do so, we employ *six auxiliary functions*: the first two return the set of agent variables and role identifiers that label the arc; the next two return the inarcs and outarcs (respectively) of a scene within a PM; the final two return the inarcs and outarcs (respectively) of a transition.

PerformativeModelAux

$arcagentvars : \text{Arc} \leftrightarrow (\mathbb{P} \text{ AgentVar})$
 $arcroles : \text{Arc} \leftrightarrow (\mathbb{P} \text{ RoleId})$
 $sceneinarcs, sceneoutarcs : \text{Scene} \leftrightarrow (\mathbb{P} \text{ Arc})$
 $transitioninarcs, transitionoutarcs : \text{Transition} \leftrightarrow (\mathbb{P} \text{ Arc})$

$\forall arc : arcs \bullet$
 $arcagentvars(arc) = \{a : \text{AgentVar}; r : \text{RoleId} \mid (a, r) \in (\bigcup(disjnorm(arc))) \bullet a\} \wedge$
 $arcroles(arc) = \{a : \text{AgentVar}; r : \text{RoleId} \mid (a, r) \in (\bigcup(disjnorm(arc))) \bullet r\}$
 $\forall s : allscenes \bullet$
 $sceneinarcs(s) = \{i : \text{InArc} \mid i.scene = s\} \wedge$
 $sceneoutarcs(s) = \{o : \text{OutArc} \mid o.scene = s\}$
 $\forall t : transitions \bullet$
 $transitioninarcs(t) = \{i : \text{InArc} \mid i.transition = t\} \wedge$
 $transitionoutarcs(t) = \{o : \text{OutArc} \mid o.transition = t\}$

We can then define the constraints on a PM as follows.

1. On any outarc from a scene, all the agent variables are free. On any inarc to a scene, all the agent variables are bound.

$\frac{\textit{Constraint1}}{\textit{PerformativeModelAux}}$
$\forall in : \textit{in arcs} \bullet \text{dom}(\textit{arcagentvars}(in)) = \{?\}$ $\forall out : \textit{out arcs} \bullet \text{dom}(\textit{arcagentvars}(out)) = \{!\}$

2. For any transition, the union of agent variables labelling outarcs of that transition *equals* the union of all agent variables labelling inarcs from that transition. (We cannot lose agents at a transition. In addition, we cannot suddenly introduce new agents.)

Note, in the schema below we take the *range* of the agent variables on the arcs as we wish to strip away the prefix before we test for equality.

$\frac{\textit{Constraint2}}{\textit{PerformativeModelAux}}$
$\forall t : \textit{Transition} \bullet$ $\bigcup \{out : \textit{transitionoutarcs}(t) \bullet \text{ran}(\textit{arcagentvars}(out))\} =$ $\bigcup \{in : \textit{transitioninarcs}(t) \bullet \text{ran}(\textit{arcagentvars}(in))\}$

3. For every scene, there is a path from the entryscene to the exitscene, which passes through that scene. (Checks that a PM has been well-defined and that all scenes can potentially be used.)

If we compose the inarcs relation with the outarcs relation the reader should see that we define a homogeneous relation that holds for two scenes if the second can be reached from the first via a transition. If we then take the reflexive transitive closure, we define a relation that will contain pairs of scenes if and only if it is possible to move from the first scene to the second scene by moving along any number of arcs. Then, for any scene, the ordered pair of the entryscene and that scene, and the ordered pair of the scene and an exit scene, are contained in the reflexive transitive closure of the relation as described above. This means that there is a route from the entryscene to any other scene, and there is a route to the exitscene from any other scene.

Constraint3

PerformativeModelAux

$$\forall s : \text{allscenes} \bullet \\ \{(entryscene, s), (s, exitscene)\} \subseteq \\ \{out : OutArc; in : InArc \mid out.transition = in.transition \bullet \\ (out.scene, in.scene)\}^*$$

4. For every scene, the union of the roles labelling the outarcs of that scene must equal the scene's roles. (So any agent with any role can get out of a scene.)

Constraint4

PerformativeModelAux

$$\forall s : \text{allscenes} \bullet \\ \bigcup \{out : sceneoutarcs(s) \bullet arcroles(out)\} = s.type.sceneroles$$

5. For every scene, the union of all the roles on all the inarcs of that scene, must equal the scene's roles. (Ensures that all there is no scene role that cannot join that scene.)

Constraint5

PerformativeModelAux

$$\forall s : \text{allscenes} \bullet \\ \bigcup \{in : sceneinarcs(s) \bullet arcroles(in)\} = s.type.sceneroles$$

6. For every role labelling an inarc of a scene, there must be at least one access state in that scene *for that role*.

Constraint6

PerformativeModelAux

$$\forall r : RoleId; in : inarcs; s : \text{allscenes} \mid \\ (r \in arcroles(in)) \wedge in.scene = s \bullet \\ r \in (\mathbf{dom} \ s.type.accessconvstates)$$

7. For every conjunction in the disjunction of agent roles labelling an inarc, there must be an access state in the scene that contains all the roles that are contained in the conjunction. Intuitively this means that all agents with their associated roles can join together.

<i>Constraint7</i> <i>PerformativeModelAux</i>
$\forall conjunction : \mathbb{P}(AgentVar \times RoleId); in : inarcs \mid$ $(conjunction \in disjnorm(in)) \bullet$ $(ran\ conjunction) \subseteq (dom\ in.scene.type.accessconvstates)$

8. Every role labelling an inarc of type *new*, the access states for that role must include the initial state.

<i>Constraint8</i> <i>PerformativeModelAux</i>
$\forall r : RoleId; in : inarcs \mid$ $in.type = \{new\} \wedge$ $r \in arcroles(in) \bullet$ $(in.scene).type.initconvstate \in$ $(in.scene).type.accessconvstates(r)$

A.1.7 The Electronic Institution

We can now define the Electronic Institution data structure which consists of the following 4 components.

1. The Social Model
2. The Communication Model
3. The Performative Model
4. The Normative Model

<i>ElectronicInstitution</i> <i>SocialModel</i> <i>NormativeModel</i> <i>CommunicationModel</i> <i>PerformativeModel</i>
--

Next, we describe the state of the system at runtime.

A.2 The State of an Electronic Institution at Run Time

A.2.1 Properties and Environments

Properties

Properties are associated with various data structures at runtime. They are typically updated when a successful communication occurs within an EI.

A property consists of an *property name*, the *range* of values that the property name could be mapped to, an optional *current value* (it is optional as there may not be a current value), and an optional *default value*.

$$[PropertyName, PropertyValue]$$

$\begin{array}{l} \textit{Property} \\ \textit{label} : \textit{PropertyName} \\ \textit{range} : \mathbb{P} \textit{PropertyValue} \\ \textit{value} : \textit{optional} [\textit{PropertyValue}] \\ \textit{default} : \textit{optional} [\textit{PropertyValue}] \end{array}$
--

A subset of properties called *required properties* must have a defined value at all times.

$$\textit{RequiredProperty} == \{a : \textit{Property} \mid \textit{defined } a.\textit{value}\}$$

Those required properties that do not have a default, clearly require a value to be given by the user before the EI can be initiated. We call these *user defined properties*.

$$\textit{UserDefinedProperty} == \{a : \textit{RequiredProperty} \mid \textit{undefined } a.\textit{default}\}$$

Binding Sets, Frames and Environments

We introduce a binding set, a frame (which records the bindings that occur when a dialogue move occurs), and an environment (which is a history of the frames).

A *binding set* is a function from variables to constants that contains exactly one time variable.

$$\begin{array}{l} \textit{BindingSet} == \{f : \textit{Var} \rightarrow \textit{Const} \mid \exists_1 t : \textit{TimeVar} \bullet \textit{tvar } t \in (\textit{dom } f) \bullet f\} \\ \textit{Frame} == \textit{ConvState} \times \textit{BindingSet} \times \textit{ConvState} \\ \textit{Environment} == \textit{seq } \textit{Frame} \end{array}$$

We introduce the set of all scene and transition instance identifiers to uniquely identify each (instance of a) scene and transition. (Instances are defined later.)

$[STId]$

The schema bindings keeps a record of the scene instance, the time, and the frame recorded at that time. Note, that we assume the existence of a common global clock in this model.

$Bindings$ $history : \mathbb{P}(STId \times Environment)$

A.2.2 Social Model State

There social model state includes a set of properties for each role. In addition, we define a relation called dynamic separation of duties dsd . If two roles are related in this predicate then it means that the roles cannot be played by the agent at the same time (in a scene?).

The other predicates are as follows,

1. The dynamic separation of duties and static separation of duties relations are disjoint.
2. There is no role in dynamic separation of duties with itself.
3. The dynamic separation of duties relation is symmetric.
4. Any two roles in dynamic separation of duties do not inherit (directly or indirectly) one another.
5. There is no role inheriting (directly or indirectly) two roles in dynamic separation of duties.
6. If a role inherits (directly or indirectly) another role and that role is in dynamic separation of duties with a third role, then the inheriting role is in dynamic separation of duties with the third one.

Presume we need a note here to say that subsumes is the same as inherits? if its the opposite then we need to change the maths accordingly.

$SocialModelState$
$SocialModel$ $roleproperties : RoleId \leftrightarrow \mathbb{P} Property$ $dsd : \mathbb{P}(RoleId \times RoleId)$
$\forall r_1, r_2, r_3 : RoleId \bullet$ $ssd \cap dsd = \{\}$ \wedge $\neg \exists r : RoleId \bullet (r, r) \in dsd \wedge$ $dsd = dsd^\sim \wedge$ $(r_1, r_2) \in dsd \Rightarrow ((r_1, r_2) \notin subsumes \wedge (r_2, r_1) \notin subsumes) \wedge$ $(r_1, r_2) \in dsd \Rightarrow (\neg existsr_3 : RoleID \bullet ((r_3, r_1) \in subsumes \wedge$ $(r_3, r_1) \in subsumes)) \wedge$ $(r_1, r_2) \in subsumes \wedge (r_2, r_3) \in dsd \Rightarrow (r_1, r_3) \in dsd$ $dom\ roleproperties = roles$

A.2.3 Normative Model State

For now we include a set of properties for each norm.

$NormativeModelState$
$NormativeModel$ $normproperties : Norm \leftrightarrow \mathbb{P} Property$ $dom\ normproperties = norms$

A.2.4 Performative Model State

Scene and Transition Instances

A *scene instance* represents an running instance of a scene during the life of an EI. A scene instance consists of a copy of the scene data structure from which it was instantiated, its environment, the current conversation state, the set of agents and their roles (which is *functional* since within a scene instance as an agent can only have one role) along with a unique scene identifier. Also we introduce a counter that records how much time has elapsed since the last successful illocution action.

The final predicate states that the number of agents in the scene instance is within the allowable range.

SceneInstance

sorig : *Scene*

currentconvstate : *ConvState*

currentroles : *AgentId* \leftrightarrow *RoleId*

id : *STId*

environment : *Environment*

agents : \mathbb{P} *AgentId*

timesincelastchange : \mathbb{N}

$currentconvstate \in sorig.type.allconvstates$

$agents = (\text{dom } currentroles)$

$\forall r : RoleId; ags : \mathbb{P} AgentId \bullet \#(currentroles \triangleright \{r\}) \in$
 $sorig.type.agentnumbers \ r$

The initial instance of any scene has a conversational state is equal to the initial state.

InitSceneInstance

SceneInstance

$currentconvstate = sorig.type.initconvstate$

A transition instance contains the transition data structure and the set of agents waiting at that instance. Each waiting agent has a set of associated scene instances that are potentially reachable. We include a variable to keep track of the roles that agents had from the scene instance from which they have just come. We identify those reachable scene instances that currently exist as *available*, those the agent can create as *creatable*, the entire set of reachable scene instances as *possible*, and those that the agent has selected as the ones it wishes to follow as *selected*. The predicates are straightforward.

We also introduce the notion of a *inarcinstance* which is an inarc coupled with the scene instance to which it points.

InArcInstance

inarc : *InArc*

sinstance : *SceneInstance*

<p><i>TransitionInstance</i></p> <p>$torig : Transition$ $currentroles : AgentId \leftrightarrow RoleId$ $possible, available, creatable, selected : AgentId \leftrightarrow \mathbb{P} InArcInstance$ $id : STId$ $waitingagents : \mathbb{P} AgentId$</p> <hr/> <p>$dom\ available = waitingagents$ $dom\ possible = waitingagents$ $dom\ creatable = waitingagents$ $dom\ selected = waitingagents$ $possible = available \cup creatable$ $available \cap creatable = \{\}$ $selected \subseteq possible$</p>
--

Agents and their Alteroids

The *Alteroids* schema represents the collection of all instances of an individual agent. We also include a mapping from agents to all the roles in which they are currently engaged through their alteroids called *agentsroles*. Finally, we have a redundant variable that simply records the set of agents which are in an EI at any one time, *allagents*.

<p><i>Alteroids</i></p> <p>$alteroids : AgentId \leftrightarrow (\mathbb{P} STId)$ $agentsroles : AgentId \leftrightarrow RoleId$ $allagents : \mathbb{P} AgentId$</p> <hr/> <p>$allagents = dom\ agentsroles$</p>
--

Performance Model Properties

For each performative model state we record properties, as we do for each of the scene types which make up that PM. Note, that the second mapping from scene types rather than from scene instances.

<p><i>PerformativeModelProperties</i></p> <p>$pmpproperties : \mathbb{P} Property$ $scenetypeproperties : SceneType \leftrightarrow \mathbb{P} Property$</p>

Performative Model State

We can now bring all this information together. The Performance Model State then consists of the set of scene instances *sinstances*, transition instances *tinstances*, alteroids

and associated properties. In addition we include the following redundant variables: the *entry scene instance* and *exit scene instance*. The predicates in the schema below are written in English as follows:

1. all scene instances are instantiations of scenes included in the Performative Model;
2. there is one and only one instance of a entryscene; and we identify the dummy variable *entrysceneinstance* with this;
3. there is one and only one instance of an exitscene; and we identify the dummy variable *exitsceneinstance* with this;
4. the alteroids function is a mapping to all instances (scenes and transitions) that include that agent;
5. the agent to roles mapping is derived from the union of all such mappings for all the scene and transition instances;

<p><i>PerformativeModelState</i></p> <hr/> <p><i>PerformativeModel</i> <i>Alteroids</i> <i>PerformativeModelProperties</i> <i>sinstances</i> : \mathbb{P} <i>SceneInstance</i> <i>tinstances</i> : \mathbb{P} <i>TransitionInstance</i> <i>entryinstance, exitinstance</i> : <i>SceneInstance</i></p> <hr/> <p>$\{s : sinstances \bullet s.sorig\} \subseteq allscenes$ <i>entryinstance</i> = $(\mu s : sinstances \mid s.sorig = entryscene)$ <i>exitinstance</i> = $(\mu s : sinstances \mid s.sorig = exitscene)$ $\forall a : AgentId \bullet alteroids(a) =$ $\{s : sinstances \mid a \in s.agents \bullet s.id\} \cup$ $\{t : tinstances \mid a \in t.waitingagents \bullet t.id\}$ <i>agentsroles</i> = $\cup\{s : sinstances \bullet s.currentroles\} \cup \cup\{t : tinstances \bullet$ <i>t.currentroles\}</i></p>
--

The initial state of a performative model state is given below.

<p><i>InitPerformativeModelState</i></p> <hr/> <p><i>PerformativeModelState</i></p> <hr/> <p><i>sinstances</i> = $\{entryinstance, exitinstance\}$ <i>agentsroles</i> = $\{\}$ <i>alteroids</i> = $\{\}$</p>

The predicate below states that all possible and creatable scene instances for an agent at a transition instance are well-defined.

$ \begin{array}{l} \textit{PMSConstraint} \\ \textit{PerformativeModelState} \\ \hline \forall t : \textit{tinstances}; s : \textit{sinstances}; in : \textit{InArcInstance}; a : \textit{AgentId} \mid \\ (a \in t.\textit{waitingagents}) \wedge \\ (in.\textit{inarc.transition} = t.\textit{torig}) \wedge \\ (in.\textit{inarc.scene} = s.\textit{sorig}) \bullet \\ ((in \in (t.\textit{available}(a))) \Rightarrow in.\textit{inarc.type} \neq \{\textit{new}\}) \wedge \\ ((in \in (t.\textit{creatable}(a))) \Rightarrow in.\textit{inarc.type} = \{\textit{new}\}) \end{array} $
--

A.2.5 The Electronic Institution State

$ \begin{array}{l} \textit{ElectronicInstitutionState} \\ \textit{SocialModelState} \\ \textit{NormativeModelState} \\ \textit{PerformativeModelState} \end{array} $

A.3 Electronic Institution Operations

In this chapter we specify the operations. There are a number of issues worth noting here.

- Anything specified in the first subsection (data structures) will not change at run time; they are *static*.
- Any infrastructure that supports the execution of EIs should support the implementation of these operations.
- We need a note to say how Z schemas are structured. Inputs, state change, conditions and trigger.
- There are two types of operation. The first we called *voluntary* and is the result of a decision making process. In a sense this is a call to the infrastructure to move things along. The second is an automatic operation and is performed by the system which essentially “cleans up” after a voluntary action or otherwise in order to maintain the integrity of the system.
- For ease on specification we will specify operations at the level of a scene and transition instances. Other operations we will specify at the level of the performative

model state. In an appendix we show how these operations can, in general, be promoted so that they are effectively working at the level of the electronic institution itself.

A.3.1 Operations on Scene Instances

Joining a Scene Instance

An agent (with a role) may *join* a scene if it is not in the scene already, the scene's current conversation state is in the access states of the agent's role and we have not reached the maximum allowed number of agents for the role.

$\begin{array}{l} \textit{JoinScene} \\ \hline \Delta \textit{SceneInstance} \\ ag? : \textit{AgentId} \\ role? : \textit{RoleId} \\ \hline ag? \notin \textit{agents} \\ \textit{currentconvstate} \in \textit{sorig.type.accessconvstates}(role?) \\ \#agents < \textit{mymax}(\textit{sorig.type.agentnumbers}(role?)) \\ \textit{currentroles}' = \textit{currentroles} \cup \{(ag?, role?)\} \end{array}$
--

Those aspects that are not specified in an operation schema, like this, are assumed to remain unchanged.

Leaving a Scene Instance

An agent can *leave* a scene if the conversation state is a leaving state for that agent's role. The second schema uses the Z construct known as anti-domain restriction that removes the leaving agent from the domain of the function.

$\begin{array}{l} \textit{LeaveSceneInstance} \\ \hline \Delta \textit{SceneInstance} \\ ag? : \textit{AgentId} \\ \hline ag? \in \textit{agents} \\ \textit{currentroles}' = \{ag?\} \triangleleft \textit{currentroles} \\ \textit{currentconvstate} \in \textit{sorig.type.leavingconvstates}(\textit{currentroles} ag?) \end{array}$
--

Closing a Scene Instance

If the conversation state is an closing state then all agents may leave. (Note that the architecture will then need to remove this scene automatically.)

<i>CloseSceneInstance</i>
$\Delta SceneInstance$
$currentconvstate \in sorig.type.closingconvstates$ $currentroles' = \{\}$

A.3.2 Operations on Transition Instances

Joining a Transition Instance

We include a mapping from the new agent to its role in the *currentroles* function.

<i>JoinTransitionInstance</i>
$ag? : AgentId$ $role? : RoleId$ $\Delta TransitionInstance$
$ag? \notin (\text{dom } currentroles)$ $currentroles' = currentroles \cup \{(ag?, role?)\}$

Leaving a Transition Instance

This is defined simply by removing the leaving agent from the domain of the *currentroles* function.

<i>LeaveTransitionInstance</i>
$ag? : AgentId$ $\Delta TransitionInstance$
$currentroles' = \{ag?\} \triangleleft currentroles$

A.3.3 Dialogue Moves

Timeout Dialogue Move

There exists a dialogue move from the current conversation state which has a timeout that is greater than the counter of the scene instance.

<i>Timeout</i>
$\Delta SceneInstance$
$\exists dm : DialogueMove \mid$ $first(dm) = currentconvstate \wedge$ $timesincelastchange > timeout^{\sim}(sorig.type.label(dm)) \bullet$ $currentconvstate' = second(dm)$

A Successful Dialogue Move

In order to make sure we have the right specification for this operation we take it slowly at first.

We are at a particular conversation state given by:

currentconvstate

We have a current environment, that is a sequence of frames binding variables to values:

environment

From the state there are a number of outgoing arcs, let us call these *nextmoves*

$nextmoves = \{currentconvstate\} \triangleleft sorig.type.dmoves$

Recall that these are labeled with illocution schemes and/or timeouts. Let us just consider the non-time out arcs and get all the resulting illocution schemes. This can be done by applying the inverse of the *label* function used to define the *DMLLabel* type. Let us call the set of all illocution schemes as follows:

$availableillocutionschemes = \{m : nextmoves \mid is = (label^{-1}(sorig.type.label(m))).ischeme \bullet is\}$

A dialogue move is successful if we complete the following steps without failure:

Substitute all bound variables in the illocution schemes on the arc(s) outgoing from the current state (prefixed by !) by their last binding value in the environment.

We define the signature for a function that, given an environment, takes a bound variable and returns the most recent constant.

| $findlastbinding : Environment \rightarrow BoundVar \rightarrow Const$

The next function takes an illocution scheme and replaces all bound variables with constants that have been discovered by the *findlastbinding* function. This function results in a Free Illocution - that is it does not contain bound variables.

| $replaceboundvariables : IllocutionScheme \rightarrow FreeIllocution$

We apply this function to each illocution scheme in the set *availableillocutionschemes* using the generic function map. This now gives us a set of free illocutions on the arcs

out from the current conversation state.

$$availablefreeillocutions = map\ replaceboundvariables\ availableillocutionschemes$$

Next, an agent speaks with an illocution.

$$\begin{aligned} agent? &: AgentId \\ i? &: Illocution \end{aligned}$$

We define a (partial) function which applies a binding set to an free illocution and returns a free illocution.

$$| \quad applybindingset : BindingSet \rightarrow FreeIllocution \rightarrow FreeIllocution$$

If it is possible to find a binding set that when applied to a Free Illocution returns an Illocution, then we say that the Free Illocution *matches* the illocution.

We define a predicate *matches*, which holds between a free illocution and an illocution when a binding set exists that maps the free illocution to the illocution.

$$\left| \begin{array}{l} matches : \mathbb{P}(FreeIllocution \times Illocution) \\ \hline \forall f : FreeIllocution; i : Illocution \bullet \\ ((f, i) \in matches) \Leftrightarrow (\exists bs : BindingSet \bullet applybindingset\ bs\ f = i) \end{array} \right.$$

If the EI system has been correctly defined there should always be one and only one FreeIllocution that matches from those that are available.

$$\exists_1 fi : availablefreeillocutions \bullet matches(fi, i?)$$

We compute the new frame from the related binding set and add it to the current environment to form a new environment.

$$environment' = environment \frown \langle (first(move), bs, second(move)) \rangle$$

Next, we substitute all variables in the constraint of the arc using this new environment. This must give us a ground CL Formula.

$$| \quad ApplyEnvironmentCL : Environment \leftrightarrow CLFormula \leftrightarrow GroundCLFormula$$

We next ensure that each of the constraints evaluates to true.

$$\forall cs : CLFormula \mid cs \in (label \sim (sorig.type.label(move)).constraints) \bullet isTrue(ApplyEnvironmentCL\ environment'\ cs)$$

Next, substitute all variables in the actions of the arc (we must get a Ground AL formula).

$$\left| \text{ApplyEnvironmentAL} : \text{Environment} \leftrightarrow \text{ALFormula} \leftrightarrow \text{GroundALFormula} \right.$$

Next we define a generic function which executes actions, in order, to change the state of the electronic institution.

$$\left| \begin{array}{l} \text{ExecuteActions} : \text{ElectronicInstitutionState} \leftrightarrow \\ \text{seq ALFormula} \leftrightarrow \text{ElectronicInstitutionState} \end{array} \right.$$

The current conversation state is updated.

$$\text{currentconvstate}' = \text{second}(\text{move})$$

A.3.4 Operations on the Performative Model State

Let us use the following abbreviation for now.

$$\text{PMS} == \text{PerformativeModelState}$$

Defining Operation Promotion

Any operation at the scene level can be promoted to an operation at the PMS level using the following schema.

$\begin{array}{l} \text{OpPMSScene} \\ \hline \Delta \text{PMS} \\ \Delta \text{SceneInstance} \\ \text{sceneinstance} : \text{SceneInstance} \\ \hline \text{sceneinstance} = \theta \text{SceneInstance} \\ \text{instances}' = \text{instances} \setminus \{\text{sceneinstance}\} \cup \{\theta \text{SceneInstance}'\} \end{array}$
--

Similarly we can define a similar promotion operation for transitions.

$\begin{array}{l} \text{OpPMSTransition} \\ \hline \Delta \text{PMS} \\ \Delta \text{TransitionInstance} \\ \text{transitioninstance} : \text{TransitionInstance} \\ \hline \text{transitioninstance} = \theta \text{TransitionInstance} \\ \text{tinstances}' = \text{tinstances} \setminus \{\text{transitioninstance}\} \cup \{\theta \text{TransitionInstance}'\} \end{array}$
--

Starting a PMS

This operation will create the initial state for PMS.

$StartPMS$
ΔPMS
$InitPerformativeModelState'$

Joining a PMS

If the agent is not currently in the PMS, then it must join the entry scene instance.

$Join$
$OpPMSScene$
$JoinScene$
$ag? \notin allagents$

Removing an agent from the PMS

If an agent does not have any alteroids we remove the agent. *As stated before this is a clean up operation that the infrastructure must perform.*

$RemoveAgent$
ΔPMS
$allagents' = allagents \setminus \{ag : allagents \mid alteroids(ag) = \{\}\}$

Removing a scene from the PMS

Any scene that contains no agents, and is not in its initial state, is removed from the system. *This is a clean up operation that the infrastructure must perform.*

$RemoveScene$
ΔPMS
$sinstances' = sinstances \setminus$ $\{s : sinstances \mid s.currentconvstate \neq s.sorig.type.initconvstate \wedge$ $s.agents = \{\}\}$

Appendix B

ISLANDER Specification Example: Trading Institution

```
<?xml version="1.0" encoding="ISO-8859-1" ?>
- <ElectronicInstitution id="eInstitution"
performativestructureid="PS"> - <Ontology id="ontology"> - <DataType
name="Good" constructor="Good"> - <DataTypeElement key="id"
required="true" list="false" type="Identifier"> - <DefaultValue>
  <NullInstance />
  </DefaultValue>
  </DataTypeElement>
- <DataTypeElement key="name" required="true" list="false"
type="String"> - <DefaultValue>
  <NullInstance />
  </DefaultValue>
  </DataTypeElement>
  </DataType>
  <FunctionType name="accept" />
  <FunctionType name="approached" />
  <FunctionType name="closed" />
- <FunctionType name="fail"> - <FunctionTypeArgument required="true"
list="false" type="String"> - <DefaultValue>
  <NullInstance />
  </DefaultValue>
  </FunctionTypeArgument>
  </FunctionType>
  <FunctionType name="hello" />
- <FunctionType name="login"> - <FunctionTypeArgument
required="true" list="false" type="String"> - <DefaultValue>
  <NullInstance />
  </DefaultValue>
  </FunctionTypeArgument>
- <FunctionTypeArgument required="true" list="false" type="String">
- <DefaultValue>
  <NullInstance />
  </DefaultValue>
  </FunctionTypeArgument>
  </FunctionType>
- <FunctionType name="sold"> - <FunctionTypeArgument required="true"
list="false" type="Identifier"> - <DefaultValue>
  <NullInstance />
  </DefaultValue>
  </FunctionTypeArgument>
- <FunctionTypeArgument required="true" list="false" type="Agent"> -
<DefaultValue>
  <NullInstance />
```

```

    </DefaultValue>
  </FunctionTypeArgument>
- <FunctionTypeArgument required="true" list="false" type="Float"> -
<DefaultValue>
  <NullInstance />
</DefaultValue>
</FunctionTypeArgument>
</FunctionType>
- <FunctionType name="startRound"> - <FunctionTypeArgument
required="true" list="false" type="DataType" name="Good"> -
<DefaultValue>
  <NullInstance />
</DefaultValue>
</FunctionTypeArgument>
- <FunctionTypeArgument required="true" list="false" type="Float"> -
<DefaultValue>
  <NullInstance />
</DefaultValue>
</FunctionTypeArgument>
- <FunctionTypeArgument required="true" list="false" type="Integer">
- <DefaultValue>
  <NullInstance />
</DefaultValue>
</FunctionTypeArgument>
</FunctionType>
  <FunctionType name="success" />
- <FunctionType name="updatePrice"> - <FunctionTypeArgument
required="true" list="false" type="Float"> - <DefaultValue>
  <NullInstance />
</DefaultValue>
</FunctionTypeArgument>
</FunctionType>
- <FunctionType name="withdrawn"> - <FunctionTypeArgument
required="true" list="false" type="Identifier"> - <DefaultValue>
  <NullInstance />
</DefaultValue>
</FunctionTypeArgument>
</FunctionType>
  <FunctionType name="endRound" />
</Ontology>
- <DialogicalFramework id="DF" language="Prolog"
ontologyref="ontology">
  <Particle id="inform" />
  <Particle id="request" />
- <Role id="Buyer" type="External">
  <Parent idref="Guest" />
</Role>
  <Role id="Guest" type="External" />
  <Role id="Receptionist" type="Internal" />
  <Role id="RoomManager" type="Internal" />
- <Role id="Seller" type="External">
  <Parent idref="Guest" />
</Role>
  <RoleRelation id="Buyer - ssd - Seller" sourceref="Buyer"
type="ssd" targetref="Seller" />
- <graphdata id="3">
  <graphnode id="6" idref="Guest" x="387.0" y="69.0" rotation="0.0" />

```

```

<graphnode id="5" idref="Buyer" x="467.0" y="213.0" rotation="0.0" />
<graphnode id="4" idref="Seller" x="314.0" y="214.0" rotation="0.0" />
<graphnode id="11" idref="RoomManager" x="175.0" y="318.0" rotation="0.0" />
<graphnode id="1" idref="Receptionist" x="133.0" y="180.0" rotation="0.0" />
<graphedge id="7" idref="Buyer - ssd - Seller" isbeziercurve="false"
  sourcenodeid="5" targetnodeid="4" sourceportindex="4" targetportindex="0"
  ctrlx1="390.5" ctrly1="213.5" ctrlx2="390.5" ctrly2="213.5"
  sourcecx="442.0" sourcecy="213.0" targetx="339.0" targety="214.0" />
</graphdata>
</DialogicalFramework>
- <SceneType id="MeetingScene" dialogicalframeworkref="DF"> -
<SceneState id="W0" initial="true" final="false"> - <SceneIOAction
type="EnterScene">
  <AgentVariable roleref="RoomManager" />
  </SceneIOAction>
- <SceneIOAction type="EnterScene">
  <AgentVariable roleref="Buyer" />
  </SceneIOAction>
- <SceneIOAction type="ExitScene">
  <AgentVariable roleref="Buyer" />
  </SceneIOAction>
</SceneState>
- <SceneState id="W1" initial="false" final="true"> - <SceneIOAction
type="ExitScene">
  <AgentVariable roleref="RoomManager" />
  </SceneIOAction>
- <SceneIOAction type="ExitScene">
  <AgentVariable roleref="Buyer" />
  </SceneIOAction>
</SceneState>
  <SceneState id="W2" initial="false" final="false" />
  <SceneState id="W3" initial="false" final="false" />
  <SceneState id="W4" initial="false" final="false" />
- <SceneArc id="W0-W1" sourceref="W0" targetref="W1"> -
<SceneMessageAction id="closed"> - <IllocutionPattern
particle="inform"> - <Sender> - <AgentNamePattern>
  <VariablePattern />
  </AgentNamePattern>
- <AgentRolePattern> - <InstancePattern>
  <Instance class="string">"RoomManager"</Instance>
  </InstancePattern>
  </AgentRolePattern>
  </Sender>
- <Receiver> - <AgentNamePattern> - <InstancePattern>
  <Instance class="string">"all"</Instance>
  </InstancePattern>
  </AgentNamePattern>
- <AgentRolePattern> - <InstancePattern>
  <Instance class="string">"Buyer"</Instance>
  </InstancePattern>
  </AgentRolePattern>
  </Receiver>
- <Content> - <InstancePattern>
  <Instance class="string">"closed"</Instance>
  </InstancePattern>
  </Content>
  </IllocutionPattern>

```

```

</SceneMessageAction>
</SceneArc>
- <SceneArc id="W0-W2" sourceref="W0" targetref="W2"> -
<SceneMessageAction id="approached"> - <IllocutionPattern
particle="request"> - <Sender> - <AgentNamePattern>
  <VariablePattern name="r" />
</AgentNamePattern>
- <AgentRolePattern> - <InstancePattern>
  <Instance class="string">"RoomManager"</Instance>
</InstancePattern>
</AgentRolePattern>
</Sender>
- <Receiver> - <AgentNamePattern>
  <VariablePattern name="b" />
</AgentNamePattern>
- <AgentRolePattern> - <InstancePattern>
  <Instance class="string">"Buyer"</Instance>
</InstancePattern>
</AgentRolePattern>
</Receiver>
- <Content> - <InstancePattern>
  <Instance class="string">"approached"</Instance>
</InstancePattern>
</Content>
</IllocutionPattern>
</SceneMessageAction>
</SceneArc>
- <SceneArc id="W2-W3" sourceref="W2" targetref="W3"> -
<SceneMessageAction id="sendCL"> - <IllocutionPattern
particle="inform"> - <Sender> - <AgentNamePattern>
  <BoundPattern variableName="b" />
</AgentNamePattern>
- <AgentRolePattern> - <InstancePattern>
  <Instance class="string">"Buyer"</Instance>
</InstancePattern>
</AgentRolePattern>
</Sender>
- <Receiver> - <AgentNamePattern>
  <BoundPattern variableName="r" />
</AgentNamePattern>
- <AgentRolePattern> - <InstancePattern>
  <Instance class="string">"RoomManager"</Instance>
</InstancePattern>
</AgentRolePattern>
</Receiver>
- <Content> - <InstancePattern>
  <Instance class="string">"sendCL"</Instance>
</InstancePattern>
</Content>
</IllocutionPattern>
</SceneMessageAction>
</SceneArc>
- <SceneArc id="W3-W4" sourceref="W3" targetref="W4"> -
<SceneMessageAction id="assisting"> - <IllocutionPattern
particle="inform"> - <Sender> - <AgentNamePattern>
  <BoundPattern variableName="r" />
</AgentNamePattern>

```

```

- <AgentRolePattern> - <InstancePattern>
  <Instance class="string">"RoomManager"</Instance>
</InstancePattern>
</AgentRolePattern>
</Sender>
- <Receiver> - <AgentNamePattern>
  <BoundPattern variableName="b" />
</AgentNamePattern>
- <AgentRolePattern> - <InstancePattern>
  <Instance class="string">"Buyer"</Instance>
</InstancePattern>
</AgentRolePattern>
</Receiver>
- <Content> - <InstancePattern>
  <Instance class="string">"assisting"</Instance>
</InstancePattern>
</Content>
</IllocutionPattern>
</SceneMessageAction>
</SceneArc>
- <SceneArc id="W4-W0" sourceref="W4" targetref="W0"> -
  <SceneMessageAction id="ready"> - <IllocutionPattern
particle="inform"> - <Sender> - <AgentNamePattern>
  <BoundPattern variableName="r" />
  </AgentNamePattern>
- <AgentRolePattern> - <InstancePattern>
  <Instance class="string">"RoomManager"</Instance>
</InstancePattern>
</AgentRolePattern>
</Sender>
- <Receiver> - <AgentNamePattern> - <InstancePattern>
  <Instance class="string">"all"</Instance>
</InstancePattern>
</AgentNamePattern>
- <AgentRolePattern> - <InstancePattern>
  <Instance class="string">"Buyer"</Instance>
</InstancePattern>
</AgentRolePattern>
</Receiver>
- <Content> - <InstancePattern>
  <Instance class="string">"ready"</Instance>
</InstancePattern>
</Content>
</IllocutionPattern>
</SceneMessageAction>
</SceneArc>
  <RoleConstraint roleref="RoomManager" minAgents="0" maxAgents="1" />
  <RoleConstraint roleref="Buyer" minAgents="0" maxAgents="100" />
- <graphdata id="5">
  <graphnode id="15" idref="W1" x="403.0" y="428.0" rotation="0.0" />
  <graphnode id="14" idref="W0" x="124.0" y="426.0" rotation="6.400006" />
  <graphnode id="4" idref="W4" x="316.0" y="311.0" rotation="0.0" />
  <graphnode id="2" idref="W3" x="240.0" y="172.0" rotation="0.0" />
  <graphnode id="1" idref="W2" x="88.0" y="228.0" rotation="0.0" />
  <graphedge id="6" idref="W2-W3" isbeziercurve="false" sourcenodeid="1" targetnodeid="2"
    sourceportindex="0" targetportindex="2" ctrlx1="209.0" ctrlx2="209.0"
    ctrly1="183.0" ctrly2="183.0" sourcecx="108.0" sourcecy="228.0" targetcx="220.0" targetcy="172.0" />

```

```

<graphedge id="5" idref="W0-W2" isbeziercurve="false" sourcenodeid="14" targetnodeid="1"
  sourceportindex="3" targetportindex="1" ctrlx1="139.0" ctrly1="279.0" ctrlx2="139.0"
  ctrly2="279.0" sourcecx="126.3311" sourcecy="406.13632" targetx="88.0" targety="248.0" />
<graphedge id="8" idref="W4-W0" isbeziercurve="false" sourcenodeid="4" targetnodeid="14"
  sourceportindex="2" targetportindex="0" ctrlx1="235.0" ctrly1="332.0" ctrlx2="235.0"
  ctrly2="332.0" sourcecx="296.0" sourcecy="311.0" targetx="143.86368" targety="428.3311" />
<graphedge id="16" idref="W0-W1" isbeziercurve="false" sourcenodeid="14" targetnodeid="15"
  sourceportindex="0" targetportindex="2" ctrlx1="272.0" ctrly1="92.0" ctrlx2="272.0"
  ctrly2="92.0" sourcecx="143.86368" sourcecy="428.3311" targetx="383.0" targety="428.0" />
<graphedge id="7" idref="W3-W4" isbeziercurve="false" sourcenodeid="2" targetnodeid="4"
  sourceportindex="1" targetportindex="3" ctrlx1="300.0" ctrly1="170.0" ctrlx2="300.0"
  ctrly2="170.0" sourcecx="240.0" sourcecy="192.0" targetx="316.0" targety="291.0" />
</graphdata>
</SceneType>
- <SceneType id="RegistrationScene" dialogicalframeworkref="DF"> -
<SceneState id="W0" initial="true" final="false"> - <SceneIOAction
type="EnterScene">
  <AgentVariable roleref="Receptionist" />
</SceneIOAction>
</SceneState>
- <SceneState id="W1" initial="false" final="false"> -
<SceneIOAction type="EnterScene">
  <AgentVariable roleref="Guest" />
</SceneIOAction>
- <SceneIOAction type="ExitScene">
  <AgentVariable roleref="Guest" />
</SceneIOAction>
</SceneState>
- <SceneState id="W2" initial="false" final="true"> - <SceneIOAction
type="ExitScene">
  <AgentVariable roleref="Receptionist" />
</SceneIOAction>
- <SceneIOAction type="ExitScene">
  <AgentVariable roleref="Guest" />
</SceneIOAction>
</SceneState>
<SceneState id="W3" initial="false" final="false" />
<SceneState id="W4" initial="false" final="false" />
- <SceneArc id="W0-W1" sourceref="W0" targetref="W1"> -
<SceneTimeOutAction id="start">
  <TimeExpression>0</TimeExpression>
</SceneTimeOutAction>
</SceneArc>
- <SceneArc id="W1-W2" sourceref="W1" targetref="W2"> -
<SceneMessageAction id="closed"> - <IllocutionPattern
particle="inform"> - <Sender> - <AgentNamePattern>
  <VariablePattern name="x" />
</AgentNamePattern>
- <AgentRolePattern> - <InstancePattern>
  <Instance class="string">"Receptionist"</Instance>
</InstancePattern>
</AgentRolePattern>
</Sender>
- <Receiver> - <AgentNamePattern> - <InstancePattern>
  <Instance class="string">"all"</Instance>
</InstancePattern>
</AgentNamePattern>

```

```

- <AgentRolePattern> - <InstancePattern>
  <Instance class="string">"Guest"</Instance>
</InstancePattern>
</AgentRolePattern>
</Receiver>
- <Content> - <InstancePattern>
  <Instance class="string">"closed"</Instance>
</InstancePattern>
</Content>
</IllocutionPattern>
</SceneMessageAction>
</SceneArc>
- <SceneArc id="W1-W3" sourceref="W1" targetref="W3"> -
<SceneMessageAction id="hello"> - <IllocutionPattern
particle="request"> - <Sender> - <AgentNamePattern>
  <VariablePattern name="x" />
</AgentNamePattern>
- <AgentRolePattern> - <InstancePattern>
  <Instance class="string">"Receptionist"</Instance>
</InstancePattern>
</AgentRolePattern>
</Sender>
- <Receiver> - <AgentNamePattern>
  <VariablePattern name="y" />
</AgentNamePattern>
- <AgentRolePattern> - <InstancePattern>
  <Instance class="string">"Guest"</Instance>
</InstancePattern>
</AgentRolePattern>
</Receiver>
- <Content> - <InstancePattern>
  <Instance class="string">"hello"</Instance>
</InstancePattern>
</Content>
</IllocutionPattern>
</SceneMessageAction>
</SceneArc>
- <SceneArc id="W3-W4" sourceref="W3" targetref="W4"> -
<SceneMessageAction id="login"> - <IllocutionPattern
particle="inform"> - <Sender> - <AgentNamePattern>
  <BoundPattern variableName="y" />
</AgentNamePattern>
- <AgentRolePattern> - <InstancePattern>
  <Instance class="string">"Guest"</Instance>
</InstancePattern>
</AgentRolePattern>
</Sender>
- <Receiver> - <AgentNamePattern>
  <BoundPattern variableName="x" />
</AgentNamePattern>
- <AgentRolePattern> - <InstancePattern>
  <Instance class="string">"Receptionist"</Instance>
</InstancePattern>
</AgentRolePattern>
</Receiver>
- <Content> - <FunctionTypePattern name="login">
  <VariablePattern name="username" />

```

```

<VariablePattern name="password" />
</FunctionTypePattern>
</Content>
</IllocutionPattern>
</SceneMessageAction>
</SceneArc>
- <SceneArc id="W4-W1" sourceref="W4" targetref="W1"> -
<SceneMessageAction id="fail"> - <IllocutionPattern
particle="inform"> - <Sender> - <AgentNamePattern>
  <BoundPattern variableName="x" />
  </AgentNamePattern>
- <AgentRolePattern> - <InstancePattern>
  <Instance class="string">"Receptionist"</Instance>
  </InstancePattern>
  </AgentRolePattern>
  </Sender>
- <Receiver> - <AgentNamePattern>
  <BoundPattern variableName="y" />
  </AgentNamePattern>
- <AgentRolePattern> - <InstancePattern>
  <Instance class="string">"Guest"</Instance>
  </InstancePattern>
  </AgentRolePattern>
  </Receiver>
- <Content> - <FunctionTypePattern name="fail">
  <VariablePattern name="reason" />
  </FunctionTypePattern>
  </Content>
  </IllocutionPattern>
  </SceneMessageAction>
- <SceneMessageAction id="success"> - <IllocutionPattern
particle="inform"> - <Sender> - <AgentNamePattern>
  <BoundPattern variableName="x" />
  </AgentNamePattern>
- <AgentRolePattern> - <InstancePattern>
  <Instance class="string">"Receptionist"</Instance>
  </InstancePattern>
  </AgentRolePattern>
  </Sender>
- <Receiver> - <AgentNamePattern>
  <BoundPattern variableName="y" />
  </AgentNamePattern>
- <AgentRolePattern> - <InstancePattern>
  <Instance class="string">"Guest"</Instance>
  </InstancePattern>
  </AgentRolePattern>
  </Receiver>
- <Content> - <InstancePattern>
  <Instance class="string">"success"</Instance>
  </InstancePattern>
  </Content>
  </IllocutionPattern>
  </SceneMessageAction>
  </SceneArc>
  <RoleConstraint roleref="Receptionist" minAgents="1" maxAgents="1" />
  <RoleConstraint roleref="Guest" minAgents="0" maxAgents="10" />
- <graphdata id="19">

```

```

<graphnode id="6" idref="W1" x="254.0" y="109.0" rotation="0.0" />
<graphnode id="4" idref="W2" x="415.0" y="109.0" rotation="0.0" />
<graphnode id="11" idref="W4" x="371.0" y="257.0" rotation="0.0" />
<graphnode id="1" idref="W0" x="108.0" y="109.0" rotation="0.0" />
<graphnode id="8" idref="W3" x="141.0" y="257.0" rotation="0.0" />
<graphedge id="15" idref="W1-W2" isbeziercurve="false" sourcenodeid="6" targetnodeid="4"
  sourceportindex="0" targetportindex="2" ctrlx1="385.0" ctrly1="117.0" ctrlx2="385.0"
  ctrly2="117.0" sourcecx="274.0" sourcecy="109.0" targetx="395.0" targety="109.0" />
<graphedge id="14" idref="W4-W1" isbeziercurve="false" sourcenodeid="11" targetnodeid="6"
  sourceportindex="3" targetportindex="1" ctrlx1="283.0" ctrly1="235.0" ctrlx2="283.0"
  ctrly2="235.0" sourcecx="371.0" sourcecy="237.0" targetx="254.0" targety="129.0" />
<graphedge id="12" idref="W3-W4" isbeziercurve="false" sourcenodeid="8" targetnodeid="11"
  sourceportindex="0" targetportindex="2" ctrlx1="280.0" ctrly1="314.0" ctrlx2="280.0"
  ctrly2="314.0" sourcecx="161.0" sourcecy="257.0" targetx="351.0" targety="257.0" />
<graphedge id="9" idref="W1-W3" isbeziercurve="false" sourcenodeid="6" targetnodeid="8"
  sourceportindex="1" targetportindex="3" ctrlx1="278.0" ctrly1="174.0" ctrlx2="278.0"
  ctrly2="174.0" sourcecx="254.0" sourcecy="129.0" targetx="141.0" targety="237.0" />
<graphedge id="7" idref="W0-W1" isbeziercurve="false" sourcenodeid="1" targetnodeid="6"
  sourceportindex="0" targetportindex="2" ctrlx1="197.0" ctrly1="111.0" ctrlx2="197.0"
  ctrly2="111.0" sourcecx="128.0" sourcecy="109.0" targetx="234.0" targety="109.0" />
</graphdata>
</SceneType>
- <SceneType id="TradingScene" dialogicalframeworkref="DF"> -
<SceneState id="W0" initial="true" final="false"> - <SceneIOAction
type="EnterScene">
  <AgentVariable roleref="Seller" />
</SceneIOAction>
- <SceneIOAction type="EnterScene">
  <AgentVariable roleref="Buyer" />
</SceneIOAction>
</SceneState>
- <SceneState id="W1" initial="false" final="false"> -
<SceneIOAction type="ExitScene">
  <AgentVariable roleref="Buyer" />
</SceneIOAction>
- <SceneIOAction type="StayAndGo">
  <AgentVariable roleref="Buyer" />
</SceneIOAction>
</SceneState>
- <SceneState id="W2" initial="false" final="true"> - <SceneIOAction
type="ExitScene">
  <AgentVariable roleref="Buyer" />
</SceneIOAction>
- <SceneIOAction type="ExitScene">
  <AgentVariable roleref="Seller" />
</SceneIOAction>
</SceneState>
- <SceneState id="W3" initial="false" final="false"> -
<SceneIOAction type="EnterScene">
  <AgentVariable roleref="Buyer" />
</SceneIOAction>
- <SceneIOAction type="StayAndGo">
  <AgentVariable roleref="Buyer" />
</SceneIOAction>
- <SceneIOAction type="ExitScene">
  <AgentVariable roleref="Buyer" />
</SceneIOAction>

```

```

</SceneState>
<SceneState id="W4" initial="false" final="false" />
<SceneState id="W5" initial="false" final="false" />
<SceneState id="W6" initial="false" final="false" />
- <SceneArc id="W0-W1" sourceref="W0" targetref="W1"> -
<SceneMessageAction id="start"> - <IllocutionPattern
particle="inform"> - <Sender> - <AgentNamePattern>
  <VariablePattern name="x" />
  </AgentNamePattern>
- <AgentRolePattern> - <InstancePattern>
  <Instance class="string">"Seller"</Instance>
  </InstancePattern>
  </AgentRolePattern>
  </Sender>
- <Receiver> - <AgentNamePattern> - <InstancePattern>
  <Instance class="string">"all"</Instance>
  </InstancePattern>
  </AgentNamePattern>
- <AgentRolePattern> - <InstancePattern>
  <Instance class="string">"Buyer"</Instance>
  </InstancePattern>
  </AgentRolePattern>
  </Receiver>
- <Content> - <InstancePattern>
  <Instance class="string">"start"</Instance>
  </InstancePattern>
  </Content>
  </IllocutionPattern>
  </SceneMessageAction>
  </SceneArc>
- <SceneArc id="W1 - W2" sourceref="W1" targetref="W2"> -
<SceneMessageAction id="closed"> - <IllocutionPattern
particle="inform"> - <Sender> - <AgentNamePattern>
  <BoundPattern variableName="x" />
  </AgentNamePattern>
- <AgentRolePattern> - <InstancePattern>
  <Instance class="string">"Seller"</Instance>
  </InstancePattern>
  </AgentRolePattern>
  </Sender>
- <Receiver> - <AgentNamePattern> - <InstancePattern>
  <Instance class="string">"all"</Instance>
  </InstancePattern>
  </AgentNamePattern>
- <AgentRolePattern> - <InstancePattern>
  <Instance class="string">"Buyer"</Instance>
  </InstancePattern>
  </AgentRolePattern>
  </Receiver>
- <Content> - <InstancePattern>
  <Instance class="string">"closed"</Instance>
  </InstancePattern>
  </Content>
  </IllocutionPattern>
  </SceneMessageAction>
  </SceneArc>
- <SceneArc id="W1 - W3" sourceref="W1" targetref="W3"> -

```

```

<SceneMessageAction id="startRound"> - <IllocutionPattern
particle="inform"> - <Sender> - <AgentNamePattern>
  <BoundPattern variableName="x" />
  </AgentNamePattern>
- <AgentRolePattern> - <InstancePattern>
  <Instance class="string">"Seller"</Instance>
  </InstancePattern>
  </AgentRolePattern>
  </Sender>
- <Receiver> - <AgentNamePattern> - <InstancePattern>
  <Instance class="string">"all"</Instance>
  </InstancePattern>
  </AgentNamePattern>
- <AgentRolePattern> - <InstancePattern>
  <Instance class="string">"Buyer"</Instance>
  </InstancePattern>
  </AgentRolePattern>
  </Receiver>
- <Content> - <InstancePattern>
  <Instance class="string">(startRound ?good ?price ?bidding_time)</Instance>
  </InstancePattern>
  </Content>
  </IllocutionPattern>
  </SceneMessageAction>
  </SceneArc>
- <SceneArc id="W1 - W6" sourceref="W1" targetref="W6"> -
<SceneMessageAction id="payment"> - <IllocutionPattern
particle="inform"> - <Sender> - <AgentNamePattern>
  <BoundPattern variableName="y" />
  </AgentNamePattern>
- <AgentRolePattern> - <InstancePattern>
  <Instance class="string">"Buyer"</Instance>
  </InstancePattern>
  </AgentRolePattern>
  </Sender>
- <Receiver> - <AgentNamePattern>
  <BoundPattern variableName="x" />
  </AgentNamePattern>
- <AgentRolePattern> - <InstancePattern>
  <Instance class="string">"Seller"</Instance>
  </InstancePattern>
  </AgentRolePattern>
  </Receiver>
- <Content> - <InstancePattern>
  <Instance class="string">(payment !price)</Instance>
  </InstancePattern>
  </Content>
  </IllocutionPattern>
  </SceneMessageAction>
  </SceneArc>
- <SceneArc id="W3 - W5" sourceref="W3" targetref="W5"> -
<SceneMessageAction id="endRound"> - <IllocutionPattern
particle="inform"> - <Sender> - <AgentNamePattern>
  <BoundPattern variableName="x" />
  </AgentNamePattern>
- <AgentRolePattern> - <InstancePattern>
  <Instance class="string">"Seller"</Instance>

```

```

</InstancePattern>
</AgentRolePattern>
</Sender>
- <Receiver> - <AgentNamePattern> - <InstancePattern>
  <Instance class="string">"all"</Instance>
  </InstancePattern>
  </AgentNamePattern>
- <AgentRolePattern> - <InstancePattern>
  <Instance class="string">"Buyer"</Instance>
  </InstancePattern>
  </AgentRolePattern>
</Receiver>
- <Content> - <InstancePattern>
  <Instance class="string">"endRound"</Instance>
  </InstancePattern>
</Content>
</IllocutionPattern>
</SceneMessageAction>
</SceneArc>
- <SceneArc id="W3 - W5_0" sourceref="W3" targetref="W5"> -
<SceneMessageAction id="accept"> - <IllocutionPattern
particle="request"> - <Sender> - <AgentNamePattern>
  <VariablePattern name="y" />
  </AgentNamePattern>
- <AgentRolePattern> - <InstancePattern>
  <Instance class="string">"Buyer"</Instance>
  </InstancePattern>
  </AgentRolePattern>
</Sender>
- <Receiver> - <AgentNamePattern>
  <BoundPattern variableName="x" />
  </AgentNamePattern>
- <AgentRolePattern> - <InstancePattern>
  <Instance class="string">"Seller"</Instance>
  </InstancePattern>
  </AgentRolePattern>
</Receiver>
- <Content> - <InstancePattern>
  <Instance class="string">"accept"</Instance>
  </InstancePattern>
</Content>
</IllocutionPattern>
</SceneMessageAction>
</SceneArc>
- <SceneArc id="W4 - W3" sourceref="W4" targetref="W3"> -
<SceneMessageAction id="decreasePrice"> - <IllocutionPattern
particle="inform"> - <Sender> - <AgentNamePattern>
  <BoundPattern variableName="x" />
  </AgentNamePattern>
- <AgentRolePattern> - <InstancePattern>
  <Instance class="string">"Seller"</Instance>
  </InstancePattern>
  </AgentRolePattern>
</Sender>
- <Receiver> - <AgentNamePattern> - <InstancePattern>
  <Instance class="string">"all"</Instance>
  </InstancePattern>

```

```

</AgentNamePattern>
- <AgentRolePattern> - <InstancePattern>
  <Instance class="string">"Buyer"</Instance>
</InstancePattern>
</AgentRolePattern>
</Receiver>
- <Content> - <InstancePattern>
  <Instance class="string">(updatePrice ?price)</Instance>
</InstancePattern>
</Content>
</IllocutionPattern>
</SceneMessageAction>
</SceneArc>
- <SceneArc id="W5 - W1" sourcerref="W5" targetref="W1"> -
<SceneMessageAction id="sold"> - <IllocutionPattern
particle="inform"> - <Sender> - <AgentNamePattern>
  <BoundPattern variableName="x" />
</AgentNamePattern>
- <AgentRolePattern> - <InstancePattern>
  <Instance class="string">"Seller"</Instance>
</InstancePattern>
</AgentRolePattern>
</Sender>
- <Receiver> - <AgentNamePattern> - <InstancePattern>
  <Instance class="string">"all"</Instance>
</InstancePattern>
</AgentNamePattern>
- <AgentRolePattern> - <InstancePattern>
  <Instance class="string">"Buyer"</Instance>
</InstancePattern>
</AgentRolePattern>
</Receiver>
- <Content> - <InstancePattern>
  <Instance class="string">(sold !good !y !price)</Instance>
</InstancePattern>
</Content>
</IllocutionPattern>
</SceneMessageAction>
- <SceneMessageAction id="withdrawn"> - <IllocutionPattern
particle="inform"> - <Sender> - <AgentNamePattern>
  <BoundPattern variableName="x" />
</AgentNamePattern>
- <AgentRolePattern> - <InstancePattern>
  <Instance class="string">"Seller"</Instance>
</InstancePattern>
</AgentRolePattern>
</Sender>
- <Receiver> - <AgentNamePattern> - <InstancePattern>
  <Instance class="string">"all"</Instance>
</InstancePattern>
</AgentNamePattern>
- <AgentRolePattern> - <InstancePattern>
  <Instance class="string">"Buyer"</Instance>
</InstancePattern>
</AgentRolePattern>
</Receiver>
- <Content> - <InstancePattern>

```

```

<Instance class="string">(withdrawn !good)</Instance>
</InstancePattern>
</Content>
</IllocutionPattern>
</SceneMessageAction>
</SceneArc>
- <SceneArc id="W6 - W1" source="W6" target="W1"> -
<SceneMessageAction id="success"> - <IllocutionPattern
particle="inform"> - <Sender> - <AgentNamePattern>
  <BoundPattern variableName="x" />
  </AgentNamePattern>
- <AgentRolePattern> - <InstancePattern>
  <Instance class="string">"Seller"</Instance>
  </InstancePattern>
  </AgentRolePattern>
</Sender>
- <Receiver> - <AgentNamePattern>
  <BoundPattern variableName="y" />
  </AgentNamePattern>
- <AgentRolePattern> - <InstancePattern>
  <Instance class="string">"Buyer"</Instance>
  </InstancePattern>
  </AgentRolePattern>
</Receiver>
- <Content> - <InstancePattern>
  <Instance class="string">"success"</Instance>
  </InstancePattern>
</Content>
</IllocutionPattern>
</SceneMessageAction>
- <SceneMessageAction id="failure"> - <IllocutionPattern
particle="inform"> - <Sender> - <AgentNamePattern>
  <BoundPattern variableName="x" />
  </AgentNamePattern>
- <AgentRolePattern> - <InstancePattern>
  <Instance class="string">"Seller"</Instance>
  </InstancePattern>
  </AgentRolePattern>
</Sender>
- <Receiver> - <AgentNamePattern>
  <BoundPattern variableName="y" />
  </AgentNamePattern>
- <AgentRolePattern> - <InstancePattern>
  <Instance class="string">"Buyer"</Instance>
  </InstancePattern>
  </AgentRolePattern>
</Receiver>
- <Content> - <InstancePattern>
  <Instance class="string">(fail ?reason)</Instance>
  </InstancePattern>
</Content>
</IllocutionPattern>
</SceneMessageAction>
</SceneArc>
- <SceneArc id="offerDec" source="W3" target="W4"> -
<SceneTimeOutAction id="newPrice">
  <TimeExpression>!biddingTime/100</TimeExpression>

```

```

</SceneTimeOutAction>
</SceneArc>
<RoleConstraint roleref="Seller" minAgents="0" maxAgents="50" />
<RoleConstraint roleref="Buyer" minAgents="0" maxAgents="50" />
<RoleConstraint roleref="RoomManager" minAgents="0" maxAgents="1" />
- <graphdata id="61">
<graphnode id="6" idref="W2" x="153.0" y="263.0" rotation="0.0" />
<graphnode id="5" idref="W6" x="243.0" y="344.0" rotation="0.0" />
<graphnode id="4" idref="W3" x="419.0" y="151.0" rotation="-7.4505806E-9" />
<graphnode id="2" idref="W1" x="232.0" y="151.0" rotation="0.0" />
<graphnode id="1" idref="W0" x="92.0" y="151.0" rotation="0.0" />
<graphnode id="21" idref="W4" x="580.0" y="151.0" rotation="0.0" />
<graphnode id="7" idref="W5" x="419.0" y="290.0" rotation="0.0" />
<graphedge id="23" idref="W4 - W3" isbeziercurve="true" sourcenodeid="21"
targetnodeid="4" sourceportindex="2" targetportindex="0" ctrlx1="509.0"
ctrly1="61.0" ctrlx2="467.0" ctrly2="85.0" sourcecx="560.0" sourcecy="151.0"
targetx="439.0" targety="151.0" />
<graphedge id="22" idref="offerDec" isbeziercurve="false" sourcenodeid="4"
targetnodeid="21" sourceportindex="0" targetportindex="2" ctrlx1="495.0"
ctrly1="184.0" ctrlx2="486.0" ctrly2="150.0" sourcecx="439.0" sourcecy="151.0"
targetx="560.0" targety="151.0" />
<graphedge id="18" idref="W3 - W5_0" isbeziercurve="true" sourcenodeid="4"
targetnodeid="7" sourceportindex="0" targetportindex="0" ctrlx1="542.0"
ctrly1="197.0" ctrlx2="501.0" ctrly2="275.0" sourcecx="439.0" sourcecy="151.0"
targetx="439.0" targety="290.0" />
<graphedge id="9" idref="W1 - W3" isbeziercurve="false" sourcenodeid="2"
targetnodeid="4" sourceportindex="0" targetportindex="2" ctrlx1="297.0"
ctrly1="148.0" ctrlx2="297.0" ctrly2="148.0" sourcecx="252.0" sourcecy="151.0"
targetx="399.0" targety="151.0" />
<graphedge id="17" idref="W3 - W5" isbeziercurve="false" sourcenodeid="4" targetnodeid="7"
sourceportindex="1" targetportindex="3" ctrlx1="497.0" ctrlly1="170.0" ctrlx2="508.0"
ctrly2="257.0" sourcecx="419.0" sourcecy="171.0" targetx="419.0" targety="270.0" />
<graphedge id="8" idref="W1 - W2" isbeziercurve="false" sourcenodeid="2" targetnodeid="6"
sourceportindex="1" targetportindex="3" ctrlx1="198.0" ctrlly1="204.0" ctrlx2="198.0"
ctrly2="204.0" sourcecx="232.0" sourcecy="171.0" targetx="153.0" targety="243.0" />
<graphedge id="13" idref="W5 - W1" isbeziercurve="true" sourcenodeid="7" targetnodeid="2"
sourceportindex="2" targetportindex="0" ctrlx1="303.0" ctrlly1="240.0" ctrlx2="355.0"
ctrly2="206.0" sourcecx="399.0" sourcecy="290.0" targetx="252.0" targety="151.0" />
<graphedge id="3" idref="W0-W1" isbeziercurve="false" sourcenodeid="1" targetnodeid="2"
sourceportindex="0" targetportindex="2" ctrlx1="180.0" ctrlly1="103.0" ctrlx2="180.0"
ctrly2="103.0" sourcecx="112.0" sourcecy="151.0" targetx="212.0" targety="151.0" />
<graphedge id="11" idref="W6 - W1" isbeziercurve="true" sourcenodeid="5" targetnodeid="2"
sourceportindex="3" targetportindex="1" ctrlx1="298.0" ctrlly1="254.0" ctrlx2="259.0"
ctrly2="228.0" sourcecx="243.0" sourcecy="324.0" targetx="232.0" targety="171.0" />
<graphedge id="10" idref="W1 - W6" isbeziercurve="true" sourcenodeid="2" targetnodeid="5"
sourceportindex="1" targetportindex="3" ctrlx1="196.0" ctrlly1="265.0" ctrlx2="208.0"
ctrly2="281.0" sourcecx="232.0" sourcecy="171.0" targetx="243.0" targety="324.0" />
</graphdata>
</SceneType>
- <PerformativeStructure id="PS" dialogicalframeworkref="DF"> -
<MultipleProtocolNode id="MeetingRoom" list="false"
staticref="MeetingScene">
<SceneType idRef="MeetingScene" />
</MultipleProtocolNode>
- <MultipleProtocolNode id="RegistrationRoom" list="false"
staticref="RegistrationScene">
<SceneType idRef="RegistrationScene" />

```

```

    </MultipleProtocolNode>
- <MultipleProtocolNode id="TradeRoom" list="true"
staticref="TradingScene">
  <SceneType idRef="TradingScene" />
</MultipleProtocolNode>
  <FinalNode id="exit" />
  <TransitionNode id="gardenToExit" type="OrX" />
  <InitialNode id="root" />
  <TransitionNode id="toExit" type="OrX" />
  <TransitionNode id="toMeetingORTrade" type="OrX" />
  <TransitionNode id="toRegistration" type="OrX" />
  <TransitionNode id="toTrade" type="Or" />
- <ProtocolToTransitionArc id="MeetingRoom-toExit"
sourceref="MeetingRoom" sourcetype="MultipleProtocolNode"
targetref="toExit"> - <AgentConjunction>
  <AgentVariable id="rm" roleref="RoomManager" />
</AgentConjunction>
</ProtocolToTransitionArc>
- <ProtocolToTransitionArc id="TradeRoom-toTrade_0"
sourceref="TradeRoom" sourcetype="MultipleProtocolNode"
targetref="toTrade"> - <AgentConjunction>
  <AgentVariable id="b" roleref="Buyer" />
</AgentConjunction>
</ProtocolToTransitionArc>
- <ProtocolToTransitionArc id="TradeRoom-transition3_0"
sourceref="TradeRoom" sourcetype="MultipleProtocolNode"
targetref="toExit"> - <AgentConjunction>
  <AgentVariable id="b" roleref="Buyer" />
</AgentConjunction>
- <AgentConjunction>
  <AgentVariable id="s" roleref="Seller" />
</AgentConjunction>
</ProtocolToTransitionArc>
- <ProtocolToTransitionArc id="garden-transition0_0"
sourceref="root" sourcetype="InitialNode"
targetref="toRegistration"> - <AgentConjunction>
  <AgentVariable id="r" roleref="Receptionist" />
</AgentConjunction>
- <AgentConjunction>
  <AgentVariable id="g" roleref="Guest" />
</AgentConjunction>
</ProtocolToTransitionArc>
- <ProtocolToTransitionArc id="meetingRoom-transition2_0"
sourceref="MeetingRoom" sourcetype="MultipleProtocolNode"
targetref="toTrade"> - <AgentConjunction>
  <AgentVariable id="b" roleref="Buyer" />
</AgentConjunction>
</ProtocolToTransitionArc>
- <ProtocolToTransitionArc id="registrationRoom-gardenToExit_0"
sourceref="RegistrationRoom" sourcetype="MultipleProtocolNode"
targetref="gardenToExit"> - <AgentConjunction>
  <AgentVariable id="r" roleref="Receptionist" />
</AgentConjunction>
- <AgentConjunction>
  <AgentVariable id="g" roleref="Guest" />
</AgentConjunction>
</ProtocolToTransitionArc>

```

```

- <ProtocolToTransitionArc id="registrationRoom-transition1_0"
sourcerref="RegistrationRoom" sourcetype="MultipleProtocolNode"
targetref="toMeetingORTrade"> - <AgentConjunction>
  <AgentVariable id="g" roleref="Guest" />
</AgentConjunction>
</ProtocolToTransitionArc>
- <ProtocolToTransitionArc id="root-toMeetingORTrade"
sourcerref="root" sourcetype="InitialNode"
targetref="toMeetingORTrade"> - <AgentConjunction>
  <AgentVariable id="rm" roleref="RoomManager" />
</AgentConjunction>
</ProtocolToTransitionArc>
- <TransitionToProtocolArc id="toMeetingORTrade-MeetingRoom"
destination="New" sourcerref="toMeetingORTrade"
targetref="MeetingRoom" targettype="MultipleProtocolNode"> -
<AgentConjunction>
  <AgentVariable id="rm" roleref="RoomManager" />
</AgentConjunction>
</TransitionToProtocolArc>
- <TransitionToProtocolArc id="toMeetingORTrade-TradeRoom_0"
destination="New" sourcerref="toMeetingORTrade" targetref="TradeRoom"
targettype="MultipleProtocolNode"> - <AgentConjunction>
  <AgentVariable id="g" roleref="Seller" />
</AgentConjunction>
</TransitionToProtocolArc>
- <TransitionToProtocolArc id="toRegistration-RegistrationRoom"
destination="One" sourcerref="toRegistration"
targetref="RegistrationRoom" targettype="MultipleProtocolNode"> -
<AgentConjunction id="Guest">
  <AgentVariable id="g" roleref="Guest" />
</AgentConjunction>
</TransitionToProtocolArc>
- <TransitionToProtocolArc id="toTrade-meetingRoom_0"
destination="One" sourcerref="toTrade" targetref="MeetingRoom"
targettype="MultipleProtocolNode"> - <AgentConjunction>
  <AgentVariable id="b" roleref="Buyer" />
</AgentConjunction>
</TransitionToProtocolArc>
- <TransitionToProtocolArc id="transition0-registrationRoom_0"
destination="New" sourcerref="toRegistration"
targetref="RegistrationRoom" targettype="MultipleProtocolNode"> -
<AgentConjunction>
  <AgentVariable id="r" roleref="Receptionist" />
</AgentConjunction>
</TransitionToProtocolArc>
- <TransitionToProtocolArc id="transition1-meetingRoom_0"
destination="One" sourcerref="toMeetingORTrade"
targetref="MeetingRoom" targettype="MultipleProtocolNode"> -
<AgentConjunction>
  <AgentVariable id="g" roleref="Buyer" />
</AgentConjunction>
</TransitionToProtocolArc>
- <TransitionToProtocolArc id="transition2-TradeRoom_0"
destination="Some" sourcerref="toTrade" targetref="TradeRoom"
targettype="MultipleProtocolNode"> - <AgentConjunction>
  <AgentVariable id="b" roleref="Buyer" />
</AgentConjunction>

```

```

</TransitionToProtocolArc>
- <TransitionToProtocolArc id="transition3-exit_0" destination="One"
sourceref="toExit" targetref="exit" targettype="FinalNode"> -
<AgentConjunction>
  <AgentVariable id="b" roleref="Buyer" />
</AgentConjunction>
- <AgentConjunction>
  <AgentVariable id="s" roleref="Seller" />
</AgentConjunction>
- <AgentConjunction>
  <AgentVariable id="rm" roleref="RoomManager" />
</AgentConjunction>
</TransitionToProtocolArc>
- <TransitionToProtocolArc id="transition4-exit_0" destination="One"
sourceref="gardenToExit" targetref="exit" targettype="FinalNode"> -
<AgentConjunction>
  <AgentVariable id="r" roleref="Receptionist" />
</AgentConjunction>
- <AgentConjunction>
  <AgentVariable id="g" roleref="Guest" />
</AgentConjunction>
</TransitionToProtocolArc>
- <graphdata id="4">
<graphnode id="22" idref="gardenToExit" x="394.0" y="267.0" rotation="6.6500077" />
<graphnode id="5" idref="exit" x="406.0" y="384.0" rotation="0.0" />
<graphnode id="13" idref="toExit" x="741.0" y="374.0" rotation="1.2499998" />
<graphnode id="4" idref="MeetingRoom" x="380.0" y="143.0" rotation="0.0" />
<graphnode id="3" idref="TradeRoom" x="724.0" y="154.0" rotation="0.0" />
<graphnode id="12" idref="toTrade" x="558.0" y="145.0" rotation="11.900025" />
<graphnode id="2" idref="RegistrationRoom" x="54.0" y="141.0" rotation="0.0" />
<graphnode id="11" idref="toMeetingORTrade" x="223.0" y="93.0" rotation="5.850004" />
<graphnode id="1" idref="root" x="126.0" y="380.0" rotation="0.0" />
<graphnode id="10" idref="toRegistration" x="82.0" y="280.0" rotation="-1.5999997" />
<graphhedge id="33" idref="toTrade-meetingRoom_0" isbeziercurve="true" sourcenodeid="12"
  targetnodeid="4" sourceportindex="0" targetportindex="1" ctrlx1="480.0" ctrly1="215.0"
  ctrlx2="459.0" ctrly2="206.0" sourcecx="567.82605" sourcecy="137.27353" targetx="395.0"
  targety="163.0" />
<graphhedge id="32" idref="TradeRoom-toTrade_0" isbeziercurve="true"
  sourcenodeid="3" targetnodeid="12" sourceportindex="3" targetportindex="0"
  ctrlx1="667.0" ctrly1="221.0" ctrlx2="557.0" ctrly2="223.0" sourcecx="709.0"
  sourcecy="174.0" targetx="554.0696" targety="148.09059" />
<graphhedge id="25" idref="root-toMeetingORTrade" isbeziercurve="true"
  sourcenodeid="1" targetnodeid="11" sourceportindex="7" targetportindex="0"
  ctrlx1="236.0" ctrly1="318.0" ctrlx2="178.0" ctrly2="252.0" sourcecx="141.0"
  sourcecy="360.0" targetx="218.46182" targety="95.0988" />
<graphhedge id="24" idref="transition4-exit_0" isbeziercurve="false"
  sourcenodeid="22" targetnodeid="5" sourceportindex="0" targetportindex="6"
  ctrlx1="158.0" ctrly1="284.0" ctrlx2="158.0" ctrly2="284.0" sourcecx="405.6684"
  sourcecy="271.48312" targetx="406.0" targety="364.0" />
<graphhedge id="19" idref="transition2-TradeRoom_0" isbeziercurve="false"
  sourcenodeid="12" targetnodeid="3" sourceportindex="0" targetportindex="4"
  ctrlx1="612.0" ctrly1="91.0" ctrlx2="668.0" ctrly2="139.0" sourcecx="567.82605"
  sourcecy="137.27353" targetx="689.0" targety="154.0" />
<graphhedge id="21" idref="transition3-exit_0" isbeziercurve="false"
  sourcenodeid="13" targetnodeid="5" sourceportindex="0" targetportindex="0"
  ctrlx1="644.0" ctrly1="278.0" ctrlx2="644.0" ctrly2="278.0" sourcecx="744.9415"
  sourcecy="385.8623" targetx="441.0" targety="384.0" />

```

```

<graphedge id="18" idref="meetingRoom-transition2_0" isbeziercurve="false"
  sourcenodeid="4" targetnodeid="12" sourceportindex="0" targetportindex="0"
  ctrlx1="591.0" ctrly1="134.0" ctrlx2="419.0" ctrly2="253.0" sourcecx="415.0"
  sourcecy="143.0" targetx="554.0696" targety="148.09059" />
<graphedge id="20" idref="TradeRoom-transition3_0" isbeziercurve="false"
  sourcenodeid="3" targetnodeid="13" sourceportindex="1" targetportindex="0"
  ctrlx1="592.0" ctrly1="233.0" ctrlx2="592.0" ctrly2="233.0" sourcecx="739.0"
  sourcecy="174.0" targetx="739.4234" targety="369.25507" />
<graphedge id="8" idref="toMeetingORTrade-MeetingRoom" isbeziercurve="false"
  sourcenodeid="11" targetnodeid="4" sourceportindex="0" targetportindex="4"
  ctrlx1="258.0" ctrly1="123.0" ctrlx2="317.0" ctrly2="105.0" sourcecx="234.34544"
  sourcecy="87.753" targetx="345.0" targety="143.0" />
<graphedge id="17" idref="transition1-meetingRoom_0" isbeziercurve="true"
  sourcenodeid="11" targetnodeid="4" sourceportindex="0" targetportindex="3"
  ctrlx1="223.0" ctrly1="159.0" ctrlx2="284.0" ctrly2="219.0" sourcecx="234.34544"
  sourcecy="87.753" targetx="365.0" targety="163.0" />
<graphedge id="7" idref="toRegistration-RegistrationRoom" isbeziercurve="true"
  sourcenodeid="10" targetnodeid="2" sourceportindex="0" targetportindex="2"
  ctrlx1="17.0" ctrly1="254.0" ctrlx2="29.0" ctrly2="219.0" sourcecx="81.63501"
  sourcecy="267.50534" targetx="54.0" targety="161.0" />
<graphedge id="16" idref="registrationRoom-transition1_0" isbeziercurve="false"
  sourcenodeid="2" targetnodeid="11" sourceportindex="0" targetportindex="0"
  ctrlx1="260.0" ctrly1="206.0" ctrlx2="260.0" ctrly2="206.0" sourcecx="89.0"
  sourcecy="141.0" targetx="218.46182" targety="95.0988" />
<graphedge id="15" idref="transition0-registrationRoom_0" isbeziercurve="true"
  sourcenodeid="10" targetnodeid="2" sourceportindex="0" targetportindex="1"
  ctrlx1="117.0" ctrly1="261.0" ctrlx2="148.0" ctrly2="228.0" sourcecx="81.63501"
  sourcecy="267.50534" targetx="69.0" targety="161.0" />
<graphedge id="6" idref="MeetingRoom-toExit" isbeziercurve="true" sourcenodeid="4"
  targetnodeid="13" sourceportindex="2" targetportindex="0" ctrlx1="463.0"
  ctrly1="337.0" ctrlx2="584.0" ctrly2="240.0" sourcecx="380.0" sourcecy="163.0"
  targetx="739.4234" targety="369.25507" />
<graphedge id="14" idref="garden-transition0_0" isbeziercurve="false"
  sourcenodeid="1" targetnodeid="10" sourceportindex="5" targetportindex="0"
  ctrlx1="-22.0" ctrly1="17.0" ctrlx2="158.0" ctrly2="46.0"
  sourcecx="111.0" sourcecy="360.0" targetx="82.145996" targety="284.99786" />
<graphedge id="42" idref="toMeetingORTrade-TradeRoom_0" isbeziercurve="true"
  sourcenodeid="11" targetnodeid="3" sourceportindex="0" targetportindex="5"
  ctrlx1="186.0" ctrly1="82.0" ctrlx2="699.0"
  ctrly2="55.0" sourcecx="234.34544" sourcecy="87.753"
  targetx="709.0" targety="134.0" />
<graphedge id="36" idref="registrationRoom-gardenToExit_0" isbeziercurve="true"
  sourcenodeid="2" targetnodeid="22" sourceportindex="0" targetportindex="0"
  ctrlx1="361.0" ctrly1="250.0" ctrlx2="227.0"
  ctrly2="242.0" sourcecx="89.0" sourcecy="141.0"
  targetx="389.33264" targety="265.20676" />
</graphdata>
</PerformativeStructure>
- <norm id="paymentNorm">
  <antecedent>(((tradeRoom (inform(?x Seller) (?y Buyer)
    (sold ?good !y ?price))))))
  </antecedent>
  <defeasibleantecedent>(((tradeRoom (!x Seller)
    (!y Buyer) success))))
  </defeasibleantecedent>
  <consequent>(((obl !o (inform (!x Seller)
    (!y Buyer) success) tradeRoom)))

```

</consequent>
</norm>
</ElectronicInstitution>

Bibliography

- [1] Lisa Abrams and Shervin Hawley. Six Signs of Trustworthiness. *ABInsight*, June, 12/10 2002.
- [2] Jean-Raymond Abrial. Formal methods in industry: achievements, problems, future. In *ICSE '06: Proceeding of the 28th International Conference on Software Engineering*, pages 761–768, New York, NY, USA, 2006. ACM.
- [3] Anna Accornero, Massimo Ancona, and Sonia Varini. All Separating Triangles in a Plane Graph Can Be Optimally “Broken” in Polynomial Time. *International Journal of Foundations of Computer Science*, 11(3):405–421, 2000.
- [4] Adobe Atmosphere. Web site. <http://www.adobe.com/products/atmosphere>, 2004.
- [5] Jacopo Aleotti, Stefano Caselli, and Monica Reggiani. Toward Programming of Assembly Tasks by Demonstration in Virtual Environments. In *12th IEEE Workshop Robot and Human Interactive Communication.*, pages 309 – 314, October 31 - November 2 2003.
- [6] Aris Alissandrakis, Chrystopher L. Nehaniv, and Kerstin Dautenhahn. Through the Looking-Glass with ALICE: Trying to Imitate using Correspondences. In *Proceedings of the First International Workshop on Epigenetic Robotics: Modeling Cognitive Development in Robotic Systems*, pages 115–122, Lurid, Sweden, 2001.
- [7] Josep Llus Arcos, Marc Esteva, Pablo Noriega, Juan Antonio Rodriguez-Aguilar, and Carles Sierra. An Integrated Developing Environment for Electronic Institutions. In Rainer Unland, Matthias Klusch, and Monique Calisti, editors, *Agent Related Platforms, Frameworks, Systems, Applications, and Tools*, Whitestein Book Series, pages 121–142. Birkhaeuser Basel, 2005.
- [8] Michael Argyle. *Bodily Communication*. Methuen, London, UK, 1988.
- [9] Kevin Armata. Signs that Sell. *Progressive Grocer*, 17(21), 1996.
- [10] Daniel Ashbrook and Thad Starner. Using GPS to learn significant locations and predict movement across multiple users. *Personal Ubiquitous Computing*, 7(5):275–286, 2003.

- [11] John L. Austin. *How to Do Things with Words: Second Edition*. Harvard University Press, 1975.
- [12] Yannis J. Bakos. A strategic analysis of electronic marketplaces. *MIS Quarterly*, 15(3):295–310, 1991.
- [13] Stuart Barnes. Virtual worlds as a medium for advertising. *SIGMIS Database*, 38(4):45–55, 2007.
- [14] Richard Bartle. *Designing Virtual Worlds*. New Riders Games, 2003.
- [15] Mathias Bauer, Dietmar Dengler, Gabriele Paul, and Markus Meyer. Programming by example: programming by demonstration for information agents. *Communications of the ACM*, 43(3n):98–103, March 2000.
- [16] Mathias Bauer and Matthieu Deru. Motion-Based Adaptation of Information Services for Mobile Users. In *Proceedings of the 10th International Conference on User Modeling (UM05)*, pages 271–276, Edinburgh, Scotland, 2005.
- [17] Barbara Becker and Gloria Mark. Social conventions in collaborative virtual environments. In E. Churchill and D. Snowdon, editors, *Proceedings of the CVE (Collaborative Virtual Environments) conference.*, Manchester, 1998.
- [18] Helmut Berger, Michael Dittenbach, Dieter Merkl, Anton Bogdanovych, Simeon Simoff, and Carles Sierra. Opening new dimensions for e-tourism. *Virtual Real.*, 11(2):75–87, 2007.
- [19] Timothy W. Bickmore and Rosalind W. Picard. Establishing and maintaining long-term human-computer relationships. *ACM Transactions on Computer-Human Interactions*, 12(2):293–327, June 2005.
- [20] Therese C. Biedl, Prosenjit Bose, Erik D. Erik D. Demaine, and Anna Lubiw. Efficient Algorithms for Petersen’s Matching Theorem. In *Proceedings of the Tenth Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 130–139, New York, USA, January 1999.
- [21] Therese C. Biedl, Goos Kant, and Michael Kaufmann. On Triangulating Planar Graphs Under the Four-Connectivity Constraint. *Algoritmica*, 19(4):427–446, December 1997.
- [22] Robert P. Biuk-Aghai. *Patterns of Virtual Collaboration*. PhD thesis, University of Technology Sydney, Australia, 2003.

- [23] Michael Bloch and Arie Segev. The impact of electronic commerce on the travel industry. *Proceedings of the Thirtieth Hawaii International Conference on System Sciences*, 04:48–58, 1997.
- [24] Guido Boella, Leendert van der Torre, and Harko Verhagen. Introduction to normative multiagent systems. In Guido Boella, Leon van der Torre, and Harko Verhagen, editors, *Normative Multi-agent Systems*, number 07122 in Dagstuhl Seminar Proceedings, pages 71–79. Internationales Begegnungs- und Forschungszentrum fuer Informatik (IBFI), Schloss Dagstuhl, Germany, 2007.
- [25] Christine L. Borgman. Why are online catalogs hard to use? lessons learned from information-retrieval studies. *Journal of the American society for information science*, 37:387–400, 1986.
- [26] Doug A. Bowman, David Koller, and Larry F. Hodges. Travel in Immersive Virtual Environments: An Evaluation of Viewpoint Motion Control Techniques. *VRAIS*, 00:45–52, 1997.
- [27] Doug A. Bowman, Ernst Kruijff, Joseph J. LaViola, and Ivan Poupyrev. An Introduction to 3-D User Interface Design. *Presence: Teleoperators and Virtual Environments*, 10(1):75–95, 2001.
- [28] David A. Bray and Benn Konsynski. Virtual Worlds, Virtual Economies, Virtual Institutions. *SSRN eLibrary*, 2006.
- [29] Cynthia Breazeal. Imitation as social exchange between humans and robots. In *Proceedings of the AISB Symposium on Imitation in Animals and Artifacts*, pages 96–104, 1999.
- [30] Robert Britton. Image of the Third World in Tourism Marketing. *Annals of Tourism Research*, 6:318–329, March 1978.
- [31] Mark Brohan. Gotta have it. online retailers will try anything and everything to get customers to impulse buy. are shoppers succumbing to their spell? *Internet Retailer*, September, 1999.
- [32] Frederick P. Brooks. *The Mythical Man-Month: Essays on Software Engineering, 20th Anniversary Edition*. Addison-Wesley Professional, August 1995.
- [33] Barry Brown and Matthew Chalmers. Tourism and mobile technology. In K. Kutti, E.H. Karsten, G. Fitzpatrick, P. Dourish, and K. Schmidt, editors, *Proceedings of the ECSCW conference*, pages 335–355, Helsinki, Finland, 2003. Kluwer Academic Press.

- [34] Adam L. Buchsbaum, Emden R. Gansner, Cecilia M. Procopiuc, and Suresh Venkatasubramanian. Rectangular Layouts and Contact Graphs. In *AT & T TD-59JQx5*.
- [35] Gary Buttriss and Ian Wilkinson. From “snap-shots” to “moving pictures”: Tracing processes using narrative sequence analysis in the evolution of an e-business. In *Proceedings of the 20th IMP-conference*, 2004.
- [36] Dolores Cariamero and Walter Van de Vetde. Socially emotional: using emotions to ground social interaction. In *Papers from AAAI Fall Symposium*, pages 10–15, 1997.
- [37] Jamais Cascio, Jerry Paffendorf, John Smart, Corey Bridges, Jochen Hummel, James Hurtsthouse, and Randal Moss. Metaverse Roadmap: Pathways to the 3D Web. *Report on the cross-industry public foresight project*, (July 4), 2007.
- [38] Edward Castronova. Virtual Worlds: A First-Hand Account of Market and Society on the Cyberian Frontier. *CESifo Working Paper Series*, 618, December 2001.
- [39] A. Celentano and F. Pittarello. Observing and adapting user behavior in navigational 3D interfaces. In *Proceedings of the Working Conference on Advanced Visual Interfaces, Gallipoli, Italy*, pages 275–282, May 2004.
- [40] Norishige Chiba and Takao Nishizeki. Arboricity and Subgraph Listing Algorithms. *SIAM Journal on Computing*, 14(1):210–223, February 1985.
- [41] Carolyn Chin and Paula Swatman. The Virtual Shopping Experience: using virtual presence to motivate online shopping. *Australasian Journal of Information Systems*, 13(1):239–253, 2005.
- [42] Luca Chittaro and Roberto Ranon. New Directions for the Design of Virtual Reality Interfaces to E-Commerce Sites. In *Proceedings of AVI 2002: 5th International Conference on Advanced Visual Interfaces*. ACM Press, 2002.
- [43] Luca Chittaro and Paolo Coppola. Animated Products as a Navigation Aid for e-Commerce. In *CHI '00 Extended Abstracts on Human Factors in Computing Systems*, pages 107–108. ACM Press, 2000.
- [44] Luca Chittaro and Lucio Ieronutti. A visual tool for tracing users’ behavior in virtual environments. In *AVI '04: Proceedings of the working conference on Advanced visual interfaces*, pages 40–47, New York, NY, USA, 2004. ACM Press.

- [45] Luca Chittaro and Roberto Ranon. Adding Adaptive Features to Virtual Reality Interfaces for E-Commerce. In *AH '00: Proceedings of the International Conference on Adaptive Hypermedia and Adaptive Web-Based Systems*, pages 86–97, London, UK, 2000. Springer-Verlag.
- [46] Luca Chittaro, Roberto Ranon, and Lucio Ieronutti. Navigating 3D Virtual Environments by Following Embodied Agents: a Proposal and its Informal Evaluation on a Virtual Museum Application. In *PsychNology Journal (Special issue on Human-Computer Interaction)*, pages 24–42, 2004.
- [47] Subir Chowdhury. *Next Generation Business Handbook. New Strategies from Tommorrow's Thought Leaders*. John Wiley & Sons., 2004.
- [48] Ray Chu. What online Hong Kong travelers look for on airline/ travel websites? *International Journal of Hospitality Management*, 20:95–100, 1 2001.
- [49] Edmund M. Clarke and Jeannette M. Wing. Formal methods: state of the art and future directions. *ACM Computing Surveys*, 28(4):626–643, 1996.
- [50] Pierre-Philippe Combes, Miren Lafourcade, and Thierry Mayer. The trade-creating effects of business and social networks: evidence from france. *Journal of International Economics*, 66(1):1–29, May 2005. available at <http://ideas.repec.org/a/eee/inecon/v66y2005i1p1-29.html>.
- [51] Jacques Corraze. *Les communications non-verbales, 4e d. rev. et corr edition*. Presses universitaires de France, Paris, 1988.
- [52] Mihaly Csikszentmihalyi. *Flow: the classic work on how to achieve happiness*. The Random House Group Ltd, London, UK, 2 edition, 2002.
- [53] Guifré Cuní, Marc Esteva, Pere Garcia, Eloi Puertas, Carles Sierra, and Teresa Solchaga. MASFIT: Multi-Agent Systems for Fish Trading. In *Proceedings of the 16th European Conference on Artificial Intelligence, (ECAI 2004)*, pages 710–714, 2004.
- [54] Bruce Damer. *Avatars! Exploring and Building Virtual Worlds on the Internet*. Peachpit Press, 1998.
- [55] Rudolph P. Darken and John L. Sibert. Wayfinding strategies and behaviors in large virtual worlds. In *CHI '96: Proceedings of the SIGCHI conference on Human factors in computing systems*, pages 142–149, New York, NY, USA, 1996. ACM Press.

- [56] Terry Daugherty, Hairong Li, and Biocca Frank. *Online Consumer Psychology: Understanding and Influencing Consumer Behavior in the Virtual World*, chapter Experiential Ecommerce: A Summary of Research Investigating the Impact of Virtual Experience on Consumer Learning. Lawrence Erlbaum Associates, 2005.
- [57] Antonella De Angeli, Sheryl Brahnham, Peter Wallis, and Alan Dix. Misuse and abuse of interactive technologies. In *CHI '06: CHI '06 extended abstracts on Human factors in computing systems*, pages 1647–1650, New York, NY, USA, 2006. ACM Press.
- [58] Edsger W. Dijkstra. *A Discipline of Programming*. Prentice-Hall, 1976.
- [59] Alan Dix. *Perspectives in HCI: Diverse Approaches*, chapter 2. Formal Methods. Academic Press Inc., 1995.
- [60] Sara Drago, Anton Bogdanovych, Massimo Ancona, Simeon Simoff, and Carles Sierra. From Graphs to Euclidean Virtual Worlds: Visualization of 3D Electronic Institutions. In Gillian Dobbie, editor, *Australasian Computer Science Conference (ACSC2007)*, volume 62 of *CRPIT*, pages 25–33, Ballarat Australia, 2007. ACS.
- [61] Earthlink. How to convert more online shoppers with live chat invites: Test data from earthlink. <http://www.liveperson.com>, 02, 2005. Case study.
- [62] David L. Edgell. *International Tourism Policy*. Van Nostrand Reinhold, New York, USA, 1990.
- [63] Jack Edmonds and Ellis L. Johnson. Matching: A Well-Solved Class of Integer Linear Programs. In Michael Jünger, Gerhard Reinelt, and Giovanni Rinaldi, editors, *Combinatorial Optimization: Eureka, You Shrink!, Papers Dedicated to Jack Edmonds*, LNCS 2570, pages 27–30. Springer, July 2003.
- [64] Steven Edwards and Harshavardhan Gangadharbatla. The Novelty of 3D Product Presentation. *Journal of Interactive Advertising [Online]*. Available at <http://jiad.org>, 2(1), 2001.
- [65] Staffan Ekvall and Danica Kragic. Grasp recognition for programming by demonstration. pages 748–753, 2005.
- [66] L. Encarnao, A. Divivier, R. Trieb, A. Ebert, H. Hagen, C. Gross, A. Fuhrmann, V. Luckas, E. Kirchdoerfer, M. Rupp, S. Vieth, S. Kimmerle, M. Keckeisen, M. Wacker, W. Strasser, M. Sattler, R. Sarlette, and R. Klein. Virtual try-on: Topics in realistic, individualized dressing in virtual reality. In *Proceedings of the Virtual and Augmented Reality Status Conference*, 2004.

- [67] Marc Esteva. *Electronic Institutions: From Specification to Development*. PhD thesis, Institut d'Investigació en Intel·ligència Artificial (IIIA), Spain, 2003.
- [68] Marc Esteva, David de la Cruz, and Carles Sierra. ISLANDER: an Electronic Institutions editor. In *First International Conference on Autonomous Agents and Multiagent systems*, pages 1045–1052, Bologna, July 2002. ACM Press.
- [69] Marc Esteva, Bruno Rosell, Juan A. Rodríguez-Aguilar, and Josep Ll. Arcos. AMELI: An Agent-Based Middleware for Electronic Institutions. In *Proceedings of the Third International Joint Conference on Autonomous Agents and Multiagent Systems (AAMAS'04)*, volume 01, pages 236–243, Los Alamitos, CA, USA, 2004. IEEE Computer Society.
- [70] Chris Fairclough, Michael Fagan, Brian Mac Namee, and Pdraig Cunningham. Research Directions for AI in Computer Games. Technical report. Trinity College Dublin, 2001.
- [71] Ernst Fehr and Urs Fischbacher. Social norms and human cooperation. *Trends in Cognitive Sciences*, 8(4):185–190, April 2004.
- [72] Christian Frueh, Siddharth Jain, and Avidesh Zakhor. Data Processing Algorithms for Generating Textured 3D Building Faade Meshes From Laser Scans and Camera Images. *International Journal on Computer Vision*, 61(2):159–184, 2005.
- [73] Zvi Galil. Efficient Algorithms for Finding Maximum Matching in Graphs. *ACM Computing Surveys*, 18(1):23–38, March 1986.
- [74] Bryan Gardiner. Bank Failure in Second Life Leads to Calls for Regulation. http://www.wired.com/gaming/virtualworlds/news/2007/08/virtual_bank, 15 August 2007.
- [75] Gartner, Inc. Gartner Says 80 Percent of Active Internet Users Will Have A "Second Life" in the Virtual World by the End of 2011, visited 23.07.2007. <http://www.gartner.com/it/page.jsp?id=503861>, 2007.
- [76] Chuck Y. Gee, James C. Makens, and Dexter J. L. Choy. *The Travel Industry, 3rd Edition*. Wiley, 1997.
- [77] William Gibson. *Neuromancer*. Ace Books, New York, NY, USA, 1984.
- [78] Andreas Girgensohn and Alison Lee. Making web sites be places for social interaction. In *Proceedings of the 2002 ACM Conference on Computer Supported Cooperative Work*, pages 136–145. ACM Press, 2002.

- [79] Malcolm Gladwell. *The Tipping Point: How Little Things Can Make a Big Difference*. Little, Brown and Company, Boston, New York, London, 2002.
- [80] Cindy Gordon. The experience economy expands. *KMWorld Magazine*, 1 September 2007. <http://www.kmworld.com/Articles/ReadArticle.aspx?ArticleID=37315>.
- [81] Bernard Gorman, Christian Thureau, Christian Bauckhage, and Mark Humphrys. Believability Testing and Bayesian Imitation in Interactive Computer Games. In *Proceedings of the 9th International Conference on the Simulation of Adaptive Behavior (SAB'06)*, volume LNAI 4095, pages 655–666. Springer, 2006.
- [82] Brian Gracely. Why play in the virtual world schoolyard? http://blogs.cisco.com/virtualworlds/2007/08/why_play_in_the_virtual_world.html, 28 August 2007.
- [83] Chris Gratton and Peter Taylor. Leisure and shopping: The domesday experience. *Leisure Management*, 7:29–30, 1987.
- [84] Dan Grigorovici. *Being There: Concepts, Effects and Measurement of User Presence in Synthetic Environments*, chapter 13. Persuasive Effects of Presence in Immersive Virtual Environments, pages 191–207. Lost Press, Amsterdam, The Netherlands, 2003.
- [85] Ad-Hoc Working Group. Report to the federal trade commission on unsolicited commercial email. Published Online. www.cdt.org/spam/, 1997.
- [86] Ning Gu and Mary Lou Maher. Dynamic Designs of 3D Virtual Worlds Using Generative Design Agents. In *Proceedings of Computer Aided Architectural Design Futures*, pages 239–248. Springer, 2005.
- [87] Ning Gu and Mary Lou Maher. New place designs with emerging technologies. *UTSePress Institutional Repository*, 2007.
- [88] Alexis Gutzman. Whitepaper: Online customer service: Do or die. *E-Commerceguide.com, Technology Solutions*, February 15 2000.
- [89] Anthony Guye-VuiliEme, Tolga K. Capin, Igor Sunday Pandzic, Nadia Magnenat Thalmann, and Daniel Thalmann. Nonverbal communication interface for collaborative virtual environments. *Virtual Reality*, 4:49–59, 1999.

- [90] E. Haik, T. Barker, J. Sapsford, and S. Trainis. Investigation into effective navigation in desktop virtual interfaces. In *In Proceedings of the 7th International Conference on 3D Web Technology, Tempe, Arizona*, pages 59–66, 2002.
- [91] Bill Harris. Online facilitation for inperson facilitators. *The Facilitator. Published Online: <http://facilitatedsystems.com/onlinefac.pdf>*, March, 2000.
- [92] Trevor Hastie and Robert Tibshirani. Discriminant adaptive nearest neighbor classification and regression. In David S. Touretzky, Michael C. Mozer, and Michael E. Hasselmo, editors, *Advances in Neural Information Processing Systems*, volume 8, pages 409–415. The MIT Press, 1996.
- [93] Richard Hawkins, Robin Mansell, and Philip Swan. *The Economic and Social Impact of Electronic Commerce: Preliminary Findings and Research Agenda*. PhD thesis, OECD, Paris, 1999.
- [94] Xin He. On Floorplans of Planar Graphs. In *Proceedings of the 29th Annual ACM Symposium on the Theory of Computing (STOC '97)*, pages 426–435. ACM Press, 1997.
- [95] Paul Hemp. Avatar-based marketing. *Harvard Business Review*, 84(6):48–57, June 2006.
- [96] Carl Hewitt. Offices are open systems. *ACM Transactions on Office Information Systems*, 4(3):271–287, 1986.
- [97] Donna Hoffman and Thomas P. Novak. Marketing in hypermedia computer mediated environments: Conceptual foundations. *Journal of Marketing*, 60(3):50–68, 1996.
- [98] Martin Holzwarth, Chris Janiszewski, and Marcus M. Neumann. The influence of avatars on online consumer shopping behavior. *Journal of Marketing*, 70(64):19–36, October 2006.
- [99] Tohru Hoshi, Koji Tsukada, Kazuma Yumoto, Keiko Tanigawa, and Yoshiyuki Nakayama. Voice over ip enabling telephony and ip network convergence (special issue on high-speed internet technology and its applications). *IEICE transactions on information and systems*, 84(5):548–559, 2001.
- [100] Seddighi H.R. A model of tourism destination choice: a theoretical and empirical analysis. *Tourism Management*, 23:475–487(13), October 2002.

- [101] Charles E. Hughes and J. Michael Moshell. Shared virtual worlds for education: the explorenet experiment. *Multimedia Systems*, 5(2):145–154, 1997.
- [102] Dan Hunter and F. Gregory Lastowka. To kill an avatar. *Legal Affairs [Published Online]*. Available at http://www.legalaffairs.org/issues/July-August-2003/feature_hunter_julaug03.msp, July 2003.
- [103] Mark Hurst. Guest Viewpoint: Why 3D Shopping Makes No Sense. <http://www.atnewyork.com/news/print.php/490091>, 2000. visited 03 September 2004.
- [104] Stephen Hutcheon. Grand slam tennis that’s out of this world. *The Sydney Morning Herald*, (January 12, 2007), 2007.
- [105] Velibor Ilic. Evolutionary neuro autonomous agents. In *Proceedings of 6th Seminar on Neural Network Applications in Electrical Engineering (NEUREL '02)*, pages 37–40, 2002.
- [106] Ian Jackson. *The Travel Industry, 3rd Edition*. Hospitality Press Pty. Ltd, 1997.
- [107] Marc Jeannerod. Visual and action cues contribute to the self-other distinction. *Nature (Neuroscience)*, (7):542–548, 2004.
- [108] Nicholas R. Jennings, Katia Sycara, and Michael Wooldridge. A roadmap of agent research and development. In *Autonomous agents and Multiagent Systems*, pages 275–306, 1998.
- [109] Shinji Kakei, Donna S. Hoffman, and Peter L. Strick. Direction of action is represented in the ventral premotor cortex. *Nature (Neuroscience)*, (4):1020–1025, 2001.
- [110] Goos Kant and Xin He. Two algorithms for finding rectangular duals of planar graphs. In Jan van Leeuwen, editor, *Computer Science (WG'93)*, LNCS 790, pages 396–410, Utrecht, The Netherlands, June 1993. Springer.
- [111] Gerard Jounghyun Kim, Kyo Chul Kang, Hyejung Kim, and Jiyoun Lee. Software Engineering of Virtual Worlds. In *VRST '98: Proceedings of the ACM symposium on Virtual reality software and technology*, pages 131–138, New York, NY, USA, 1998. ACM Press.
- [112] Stefan Klein and Claudia Loebbecke. Signaling and Segmentation on Electronic Markets: Innovative Pricing Strategies for Improved Resource Allocation. In *Pro-*

ceedings of 6-th Symposium on Emerging Electronic Markets, pages 127–142, 1999.

- [113] Kurt Konolige and Martha E. Pollack. A representationalist theory of intention. In Ruzena Bajcsy, editor, *Proceedings of the Thirteenth International Joint Conference on Artificial Intelligence (IJCAI-93)*, pages 390–395, Chambéry, France, 1993. Morgan Kaufmann publishers Inc.: San Mateo, CA, USA.
- [114] Otto R. Koppius, Wouter Speelman, Oliver Stulp, Bart Verhoef, and Eric van Heck. Why are customers coming back to buy their airline tickets online? theoretical explanations and empirical evidence. In *ICEC '05: Proceedings of the 7th international conference on Electronic Commerce*, pages 319–326, Xi'an, China, 2005.
- [115] Krzysztof Kozminski and Edwin Kinnen. An Algorithm for Finding a Rectangular Dual of a Planar Graph for Use in Area Planning for VLSI Integrated Circuits. In *ACM IEEE 21st Design Automation Conference (DAC '84)*, LNCS 3590, pages 655–656, Los Angeles, Ca., USA, 1984. IEEE Computer Society Press.
- [116] Lori Kremen. A Second Life for Big Business. *Technology Review [Published Online]*. <http://www.technologyreview.com/Biztech/18016>, (January 5, 2007), 2007.
- [117] Linden Lab. Second Life Terms of Service. available online at <http://secondlife.com/corporate/tos.php>.
- [118] Theresa Lai and Simon So. Could we smell the cooking over the internet? an interactive e-learning solution for home economics using synchronized multimedia. In *Proceedings of World Conference on E-Learning in Corporate, Government, Healthcare, and Higher Education*, pages 2195–2198. AACE, 2002.
- [119] Yen-Tai Lai and Sany M. Leinwand. Algorithms for Floorplan Design Via Rectangular Dualization. *IEEE Transaction on Computer-Aided Design*, 7:1278–1289, 1988.
- [120] John E. Laird and Michael Lent. Interactive Computer Games: Human-Level AI's Killer Application. In *Proceedings of AAAI 2000 conference*, pages 1171–1178, 2000.
- [121] Tania C. Lang. The effect of the internet on travel consumer purchasing behavior and implications for travel agencies. *Journal of Vacation Marketing*, 6(4):368–385, 2000.

- [122] Tania C. Lang. The effect of the internet on travel consumer purchasing behavior and implications for travel agencies. *Journal of Vacation Marketing*, 6(4):368–385, 2000.
- [123] Greg Lastowka and Dan Hunter. The laws of the virtual worlds. *California Law Review*, 92(1):1–73, 2004.
- [124] Ronan Le Hy, Anthony Arrigony, Pierre Bessiere, and Olivier Lebeltel. Teaching bayesian behaviors to video game characters. *Robotics and Autonomous Systems*, 47:177–185, 2004.
- [125] Vladimir Levenshtein. Binary codes capable of correcting deletions, insertions, and reversals. *Soviet Physics Doklady*, 10:707–710, 1966.
- [126] David Lightfoot. *Formal Specification Using Z. Second Edition*. PALGRAVE, NY, USA, 2001.
- [127] Daniel Livingstone. Turing’s test and believable AI in games. *Computers in Entertainment*, 4(1):6–18, 2006.
- [128] Mike Luck, Mark d’Inverno, Juan Antonio Rodriguez, and Carles Sierra. Z Specification of Electronic Institutions. Personal Communication, 2006.
- [129] Kim Halskov Madsen. A guide to metaphorical design. *Communication of the ACM*, 37(12):57–62, 1994.
- [130] Pattie Maes and Daniele Nardi. *Meta-Level Architectures and Reflection*. Elsevier Science Inc., NY, USA, 1988.
- [131] Nadia Magnenat-Thalmann, HyungSeok Kim, Arjan Egges, and Stephane Garchery. Believability and Interaction in Virtual Worlds. In *Proceedings of the 11th International Multimedia Modelling Conference (MMM’05)*, pages 2–9, Washington, DC, USA, 2005. IEEE Computer Society.
- [132] James Maguire. Case study: Liveperson: Searching for the personal in e-commerce [published online]. available at <http://www.instantmessagingplanet.com/enterprise/article.php/1484371>, 2002.
- [133] Mary Lou Maher, Simeon Simoff, and John Mitchell. Formalizing Building Requirements using an Activity/Space Model. *Automation in Construction*, 6:77–95, 1997.

- [134] Anna Mancini. *Internet Justice, Philosophy of Law for the Virtual World*. Buenos Books America LLC, 2005.
- [135] Roger Marcha and Arch G. Woodside. Testing theory of planned versus realized tourism behavior. *Annals of Tourism Research*, 32(2):905—924, 2005.
- [136] Viktor Mayer-Schoenberger and John Crowley. Napster’s Second Life? The Regulatory Challenges of Virtual Worlds. *Northwestern Law Review*, 1775:1775–1826, 2006.
- [137] John McLean. Twenty years of formal methods. In *IEEE Symposium on Security and Privacy*, pages 115–116, 1999.
- [138] Michael Meehan. Virtual property: Protecting bits in context. *Richmond Journal of Law and Technology*, XIII(2), 2006.
- [139] Mary Modahl. *Now or never : how companies must change today to win the battle for internet consumers*. HarperBusiness, 2000.
- [140] Chip Morningstar and F. Randall Farmer. The lessons of lucasfilm’s habitat. *Cyberspace: first steps*, pages 273–302, 1991.
- [141] Charlie Morris. Sell ads on my site? can i? should i? how do i? http://www.webdevelopersjournal.com/articles/sell_ads_on_my_site.html, 1999.
- [142] Ian Mount. Cranky consumer. *The Wall Street Journal*, February 2005.
- [143] Janet Murray. *Hamlet on the Holodeck*. The Free Press, 1997.
- [144] Hideyuki Nakanishi, Satoshi Nakazawa, Toru Ishida, Katsuya Takanashi, and Katherine Isbister. Can software agents influence human relations?: balance theory in agent-mediated communities. In *AAMAS '03: Proceedings of the second international joint conference on Autonomous agents and multiagent systems*, pages 717–724, New York, NY, USA, 2003. ACM.
- [145] Nick Easen for CNN. The rise of self-service travel, visited 27.07.2007. <http://www.cnn.com/2004/TRAVEL/06/10/bt.self.service/index.html>, 2004.
- [146] Jakob Nielsen. 2D is Better Than 3D. *Jakob Nielsen’s Alertbox*, November 15 1998. <http://www.useit.com/alertbox/981115.html>, visited 22 June 2004.
- [147] Douglass C. North. *Institutions, Institutional Change and Economic Performance*. Cambridge University Press, Cambridge, U.K., 1990.

- [148] Travel Industry Association of America. Travelers use of the internet, 2005 edition. <http://www.tia.org/pressmedia/pressrec.asp?Item=689>, 2005.
- [149] Paul O'Mahony. Sweden trumped by Maldives in Second Life. *The Local: Sweden News in English*, (22nd May 2007), 2007. Available at <http://www.thelocal.se/7379/20070522/>.
- [150] Jeremy Page. Tiny island nation opens the first real embassy in virtual world. *Times Online*, (May 24, 2007), 2007. Available at http://technology.timesonline.co.uk/tol/news/tech_and_web/article1832158.ece.
- [151] Dinesh Pai, Kees van den Doel, Doug James, Jochen Lang, John E. Lloyd, Joshua L. Richmond, and Som H. Yau. Scanning Physical Interaction Behavior of 3D Objects. In *Proceedings of ACM SIGGRAPH 2001*, pages 87–96, August 2001. MPEG Video available at: <http://www.cs.ubc.ca/nest/lci/acme/movies/acmesiggraph2001.mpg>.
- [152] Adrian Palmer and David Bejou. Tourism destination marketing alliances. *Annals of Tourism Research*, 22:616–629, 3 1995.
- [153] Zhigeng Pan, Bing Xu, Mingming Zhang, and Hongwei Yang. Collaborative shopping based on multi-agent in virtual environments. In *Proceedings of the 8-th International Conference on Computer Supported Cooperative Work in Design*, pages 386–391. IEEE, 2004.
- [154] Ojas Parekh. Edge Dominating and Hypomatchable Sets. In *Proceedings of 13th Annual ACM-SIAM Symposium on Discrete Algorithms (SODA'02)*, pages 287–291, San Francisco, CA, USA, January 2002. ACM/SIAM.
- [155] Adriano M. Pereira, Mark Song, Gustavo Gorgulho, Wagner Meira Jr., and Sérgio Vale Aguiar Campos. A Formal Methodology to Specify E-commerce Systems. *Proceedings of the 4th International Conference on Formal Engineering Methods: Formal Methods and Software Engineering. Lecture Notes for Computer Science*, 2495:180–191, 2002.
- [156] Chang-Shing Perng, Haixun Wang, Sylvia R. Zhang, and D. Stott Parker. Landmarks: A new model for similarity-based pattern querying in time series databases. In *Proceedings of the 16th International Conference on Data Engineering*, pages 33–42, 2000.
- [157] Paolo Petta and Robert Trappl. Emotions and agents. *Multi-agents systems and applications*, pages 301–316, 2001.

- [158] Graeme Philipson. Get a Second Life, why don't you? *The Sydney Morning Herald*, (March 27, 2007), 2007.
- [159] B. Joseph Pine and James H. Gilmore. *The Experience Economy: work is a theatre and every business is a stage: goods & services are no longer enough*. Harvard Business School Press, Boston, Massachusetts, USA, 1999.
- [160] B. Joseph Pine II and James H. Gilmore. Welcome to the experience economy. *Harvard Business Review*, pages 97–105, July-August 1998.
- [161] Pattarawan Prasarnphanich and Mark L. Gillenson. The Hybrid Clicks and Bricks Business Model. *Communications of the ACM*, 46:178–185, 12 2003.
- [162] Jenny Preece and Diane Maloney-Krichmar. The Human-Computer Interaction Handbook, chapter Online Communities: Sociability and Usability. Lawrence Erlbaum Associates Inc., 2003. Mahwah, USA, pp. 596-620.
- [163] Helmut Prendinger and Mitsuru Ishizuka. Let's talk! socially intelligent agents for language conversation training. *IEEE Transactions on Systems, Man, and Cybernetics, Part A*, 31(5):465–471, 2001.
- [164] Helmut Prendinger and Mitsuru Ishizuka. Social role awareness in animated agents. In *AGENTS '01: Proceedings of the fifth international conference on Autonomous agents*, pages 270–277, New York, NY, USA, 2001. ACM Press.
- [165] Marcelo Prince. E-commerce: Help (may be) on the way. *The Wall Street Journal*, 15(2):158–175, March 21 2005.
- [166] Petra Probst. Freizeit- und erlebniswelten: Entwicklung, trends und perspektiven. *Erlebnis- und Konsumwelten*, pages 104–118, 2000.
- [167] Stefano Puglia, Robert Carter, and Ravi Jain. Multecommerce: a distributed architecture for collaborative shopping on the www. In *EC '00: Proceedings of the 2nd ACM conference on Electronic commerce*, pages 215–224, New York, NY, USA, 2000. ACM Press.
- [168] Sheizaf Rafaeli and Avi Noy. Social presence: Influence on bidders in internet auctions. *EM-Electronic Markets*, 15(2):158–175, 2005.
- [169] Kim Ramus and Niels Asger Nielsen. Online grocery retailing: what do consumers think? *Journal of Internet Research*, 15(3):335–352, 2005.

- [170] Ronald C. Read. A New Method for Drawing a Graph given the Cyclic Order of the Edges at Each Vertex. *Congr. Numer.*, 56:31–44, 1987.
- [171] Byron Reeves. The benefits of interactive online characters. *Center for the Study of Language and Information, Stanford University*, 2004.
- [172] Juan Antonio Rodriguez. *On the Design and Construction of Agent-Mediated Electronic Institutions*. PhD thesis, Artificial Intelligence Research Institute (IIIA-CSIC), Spain, 2003.
- [173] Michael Rosemann and Peter Green. Integrating multi-perspective views into ontological analysis. In *ICIS '00: Proceedings of the twenty first international conference on Information systems*, pages 618–627, Atlanta, GA, USA, 2000. Association for Information Systems.
- [174] Roy A. Ruddle. Navigating Overlapping Virtual Worlds: Arriving in One Place and Finding that You're Somewhere Else. In *Spatial Cognition II, Integrating Abstract Theories, Empirical Studies, Formal Methods, and Practical Applications*, pages 333–347, London, UK, 2000. Springer-Verlag.
- [175] Donald Rugg. The Choice of Journey Destination: A Theoretical and Empirical Analysis. *The Review of Economics and Statistics*, 55:64–72, 1 1973.
- [176] Daniel Stephen Rule. Artificial Intelligence (AI) Chatbot, December 2003. Available at http://www.codeproject.com/useritems/AI_Chatbot.asp.
- [177] James Rumbaugh, Ivar Jacobson, and Grady Booch. *Unified Modeling Language Reference Manual, The (2nd Edition) (Addison-Wesley Object Technology Series)*. Addison-Wesley Professional, July 2004.
- [178] W. G. Runciman and Robert K. Merton. *The Social Animal*. University of Michigan Press, 2000.
- [179] John Rushby. Formal methods and their role in the certification of critical systems. Technical Report CSL-95-01, CSL-95-1, Menlo Park CA 94025 USA, March 1995.
- [180] Stuart Russel and Peter Norvig. *Artificial Intelligence a Modern Approach*. Prentice Hall. Pearson Education International, Upper Saddle River, New Jersey, USA, 2003.
- [181] Cristina Russo Dos Santos, Pascal Gros, Pierre Abel, Didier Loisel, Nicolas Trichaud, and Jean-Pierre Paris. Mapping Information onto 3D Virtual Worlds. In

- Proceedings of the International Conference on Information Visualization*, pages 379–386, Washington, DC, USA, 2000. IEEE Computer Society.
- [182] Laila Refiana Said. *The Influences of Cognitive, Experiential and Habitual Factors in Online Games Playing*. PhD thesis, University of Western Australia, Australia, 2005.
- [183] Andrea Sanna and Bartolomeo Montrucchio. 3D technologies and products for e-commerce on the Web. *Software Focus*, 2(4):157–163, 2001.
- [184] Andrew Schotter. *The Economic Theory of Social Institutions*. Cambridge University Press, Cambridge, USA, 1981.
- [185] John R. Searle. *Expression and Meaning: Studies in the Theory of Speech Acts*. Cambridge University Press, Cambridge, UK, 1 edition, 1979.
- [186] Australian Government Publishing Service. Report of the tourism shopping implementation committee. 1990.
- [187] Willam R. Sherman and Alan B. Craig. *Understanding Virtual Reality: Interface, Application and Design*. Morgan Kaufmann Publishers, San Francisco, USA, 2003.
- [188] Jeffrey Sam Siekpe. An examination of the multidimensionality of flow construct in a computer-mediated environment. *Journal of Electronic Commerce Research*, 6(1):31–43, 2005.
- [189] Goh Ong Sing, Kok Wai Wong, Chun Che Fung, and Arnold Depickere. Towards a more natural and intelligent interface with embodied conversation agent. In *CyberGames '06: Proceedings of the 2006 international conference on Game research and development*, pages 177–183, Murdoch University, Australia, 2006. Murdoch University.
- [190] Ong SING GOH and Chun CHE FUNG. Intelligent Agent Technology in E-commerce. *Lecture Notes in Computer Science. Intelligent Data Engineering and Automated Learning*, 2690/2003:10–17, 2003.
- [191] Sandeep Singhal and Michael Zyda. *Networked virtual environments: design and implementation*. ACM Press/Addison-Wesley Publishing Co., New York, NY, USA, 1999.
- [192] Alan Sipress. Where real money meets virtual reality, the jury is still out. *The Washington Post*, page A01, December 26 2006.

- [193] M. Joseph Sirgy and Chenting Su. Destination Image, Self-Congruity, and Travel Behavior: Toward an Integrative Model. *Journal of Travel Research*, 38:340–352, May 2000.
- [194] Robert C. Solomon. Aristotle, ethics and business organizations. *Organization Studies*, 25(6):1021–1043, 2004.
- [195] J. Mike Spivey. *The Z notation: a reference manual*. Prentice-Hall, Inc., Upper Saddle River, NJ, USA, 1989.
- [196] Anna Stahl, Petra Sundstroem, and Kristina Hoeoek. A foundation for emotional expressivity. In *DUX '05: Proceedings of the 2005 conference on Designing for User eXperience*, pages 33–53, New York, NY, USA, 2005. AIGA: American Institute of Graphic Arts.
- [197] Arseni Starodoumov. Real Money Trade Model in Virtual Economies. Master's thesis, Stockholm School of Economics, Institute of International Business (IIB), 2005.
- [198] Neal Stephenson. *Snow Crash*. Spectra; Reissue edition (April 1, 1993), 1992.
- [199] Jon Stewart. Travel Agencies Say they are a Better Choice than Internet. The Frederick News-Post, Frederick, Md. Aug. 22, <http://www.fredericknewspost.com/>.
- [200] Georgy Stiny and James Gips. Shape grammars and the generative specification of painting and sculpture. In *Information Processing (IFIP) Congress*, pages 1460–1465. North Holland Publishing Co, 1971.
- [201] John Suler. *The Psychology of Cyberspace*. Published Online., <http://www.rider.edu/suler/psyber/psyber.html>, 2005.
- [202] Pang-Ning Tan, Michael Steinbach, and Vipin Kumar. *Introduction to Data Mining*. Addison-Wesley, 2006.
- [203] Daniel Terdiman. Who Governs Virtual Worlds. *CNET News*, December, 2006.
- [204] Christian Thureau, Christian Bauckhage, and Gerhard Sagerer. Learning Human-Like Movement Behavior for Computer Games. In *Proceedings of the 8-th Simulation of Adaptive Behavior Conference (SAB'04)*, pages 315–323.
- [205] Aron Trauring. Software methodologies: Battle of the gurus. *White Paper*, 2002. Available at <http://www.fourm.info/MyArticles/SoftMeth.pdf>.

- [206] Paco Underhill. *Why We Buy: The Science of Shopping*. Touchstone, Rockefeller Center, 1230 Avenue of the Americas New York, NY 10020, 1999.
- [207] N. Vinson. Design Guidelines for Landmarks to Support Navigation in Virtual Environments. In *In Proceedings of CHI99, ACM Press*, pages 278–285, 1999.
- [208] Andrew Vladimir. *The Complete Travel Marketing Handbook*. NTC Business Books, USA, 1989.
- [209] Klaus Weiermair. Von der dienstleistungskonomie zur erlebniskonomie. *Industrie Erlebniswelten - Vom Standort zur Destination*, pages 35–48, 2001.
- [210] Jerry Weinstein and Jack Myers. Same principles apply to virtual world expansion as to china and other new markets. *Media Village*, (November 28th), 2006. Available at <http://www.mediavillage.com/jmr/2006/11/28/jmr-11-28-06/>.
- [211] Gerhard Weiss. *Multiagent Systems: A Modern Approach to Distributed Artificial Intelligence*. MIT Press, Cambridge, MA, USA, 2000.
- [212] Don Wells. Extreme programming: a gentle introduction. <http://www.extreme-programming.org>.
- [213] Mary C. Whitton. Making virtual environments compelling. *Communications of the ACM*, 46(7):40–47, 2003.
- [214] Wikipedia, the Free Online Encyclopedia with Open Content. Adobe atmosphere, visited 20.06.2007. http://en.wikipedia.org/wiki/Adobe_Atmosphere, 2007.
- [215] Wikipedia, the Free Online Encyclopedia with Open Content. Moo, visited 19.06.2007. <http://en.wikipedia.org/wiki/MOO>, 2007.
- [216] Wikipedia, the Free Online Encyclopedia with Open Content. Second Life, visited 20.06.2007. http://en.wikipedia.org/wiki/Second_life, 2007.
- [217] Wikipedia, the Free Online Encyclopedia with Open Content. Vrml, visited 20.06.2007. <http://en.wikipedia.org/wiki/VRML>, 2007.
- [218] Laurie M. Wilcox, Robert S. Allison, Samuel Elfassy, and Cynthia Grelik. Personal space in virtual reality. *ACM Transactions on Applied Perception (TAP)*, 3(4):412–428, 2006.
- [219] Amy Williams, Steve Barrus, R. Keith Morley, and Peter Shirley. An efficient and robust ray-box intersection algorithm. *Journal of Graphics Tools*, 10(1):49–54, 2005.

- [220] Michael Wooldridge, Nicholas R. Jennings, and David Kinny. The Gaia Methodology for Agent-Oriented Analysis and Design. *Autonomous Agents and Multi-Agent Systems*, 3(3):285–312, 2000.
- [221] Ian Yeoman, Alastair Durie, Una McMahon-Beattie, and Adrian Palmer. Capturing the essence of a brand from its history: The case of scottish tourism marketing. *Journal of Brand Management*, 13(2):34–147, 2005.