Euclidean Representation of 3D Electronic Institutions: Automatic Generation

Anton Bogdanovych Faculty of Information Technology
University of Technology Sydney
PO Box 123, Broadway, NSW 2007, Australia
anton@it.uts.edu.au

Sara Drago Department of Computer Science University of Genoa
via Dodecaneso 35, 16146 Genova, Italy
drago@disi.unige.it

ABSTRACT

In this paper we present the 3D Electronic Institutions metaphor and show how it can be used for the specification of highly secure Virtual Worlds and how 3D Virtual Worlds can be automatically generated from this specification. To achieve the generation task we propose an algorithm for automatic transformation of the Performative Structure graph into a 3D Virtual World, using the rectangular dualization technique. The nodes of the initial graph are transformed into rooms, the connecting arcs between nodes determine which rooms have to be placed next to each other and define the positions of the doors connecting those rooms. The proposed algorithm is sufficiently general to be used for transforming any planar graph into a 3D Virtual World.

Categories and Subject Descriptors

D.1.7 [**PROGRAMMING TECHNIQUES**]: [Visual Programming]

General Terms

Algorithms, Design.

Keywords

Rectangular Dualization, 3D Electronic Institutions.

1. INTRODUCTION

The advent of communication networks and increasing computational power of hardware have changed the way software applications are thought and designed. One of the quite recent technologies, 3D Virtual Worlds, provide the exciting possibility to create systems that look similar to our everyday life visual experiences. With 3D modeling we can imitate real objects and profit from users' familiarity with their affordances. Having avatars representing the participants makes it possible to include social and even psychological aspects into computer-operated environments. The use of avatars implicitly incorporates location awareness and offers mechanisms for social interaction, supporting to a certain extent the way humans operate and interact in the real world. Thus social interaction between participants becomes an important integral feature of software systems.

One of the drawbacks of the Virtual Worlds technology is that its design and development has emerged as a phenomenon shaped by a home computer user, rather than by the research and development in universities or companies. As a result, Virtual Worlds are somewhat unregulated environments, which do not have the means to enforce technological norms and rules on their inhabitants. Moreover, up until now the methodological support for design and implementation of 3D Virtual Worlds was not present on the market. Classical methodologies do not seem any longer well suited to deal with a new situation in which software has to be composed of hundreds or thousands of distributed components that may show a significant level of autonomy.

3D Electronic Institutions are a step towards new methods of software design of *open* systems based on the metaphor of 3D Virtual Worlds. It is a powerful methodology, which helps to structure the interactions between different participants by imposing, and enforcing, well established conventions based on organizational principles. As co-operation cannot necessarily be guaranteed due to the autonomy of inhabitants, such a restrictive framework permits a certain level of predictability in the overall software behavior, which is very important in software design.

The enforcement of organizational conventions in 3D Electronic Institutions methodology is achieved by separating different patterns of conversational activities into separate methodological entities (scenes), assigning different roles to different types of participants, specifying the rules (protocols) for inter-participant interactions and defining the role flow of participants between different scenes. The specification of scenes and the role flow are done in a form of a directed graph, where nodes represent scenes and arcs and their labels define the role flow. This graph, which we call *Performative Structure*, forms a basis for the visualization of the system. The nodes are visualized as rooms and arcs are transformed into doors connecting the rooms.

To make the visualization of 3D Electronic 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 non-Euclidean visualization: in this case all rooms are independent objects which are loaded on demand when a user needs to enter them. The second approach is to make the visualization fully Euclidean, meaning that the whole institution is located within some virtual space. The connected rooms in this case are physically located next to each other and are separated by doors inside this space.

The long term focus of our research is to conduct a user study to compare the acceptance by humans of both aforementioned approaches. As we believe that the Euclidean representation helps to avoid navigation problems we started with the second approach. Here we present how the Euclidean representation of a 3D Electronic Institution can be automatically generated from the Performative Structure graph using the rectangular dualization technique [5].

The remainder of the paper is structured as follows. Section 2 describes the 3D Electronic Institutions methodology. In Section 3 we give a generic description of the rectangular dualization method. Section 4 explains how the rectangular dualization technique can be adapted to the generation of 3D Electronic Institutions. Concluding remarks and details of future work are given in Section 5.

2. 3D ELECTRONIC INSTITUTIONS

3D Electronic Institutions is a methodology, supported by a number of tools, for designing highly secure and reliable immersive 3D solutions. Applying this methodology requires 3 important steps to be accomplished:

- *Specification* of a 3D Electronic Institution;
- *Annotation* of the specification with components of a 3D Virtual World;
- *Generation* of the corresponding 3D environment.

On the specification stage the institutional regulations are defined. An institution is seen as an infrastructure for regulating the interactions of autonomous agents, which can be either humans or autonomous software modules. The notion of institution introduces a dramatic difference to the development of open systems compared to the majority of present solutions. Instead of focusing on the implementation details of each particular agent, a system-oriented view is taken. We assume that participating agents 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 have to comply with. This assumption permits that agents behave autonomously and make their decisions freely up to the limits imposed by the institution. The limitations on user behavior in an institution are introduced on the specification phase and are determined by three types of conventions:

Conventions on language, the *Dialogical Framework*. This dimension determines what language ontology and illocutionary particles agents should use. It also fixes the organizational structure of the society of agents, that is, which roles agents can play, and what the incompatibilities and relationships among the roles are.

Conventions on activities, the *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 Performative Structure fixes which protocol 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. 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 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 special scenes called *transitions*. Transitions re-route agents and are where synchronization with other agents (if needed) takes place.

Conventions on behavior, the *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 talking 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.

While the specification strictly defines the limitations, it also helps to understand what participants need in order 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 automatically generate a 3D representation of the specification. The scenes and transitions, for example, are transformed into 3D rooms, connections correspond to doors, and the number of participants allowed in a scene determines the size of a room (for more details see [4]). The created 3D Electronic Institution is ready to be executed. The generated 3D Virtual World can be visualized and the 3D Electronic Institutions infrastructure will take care of the validity of interactions between participants, verify the permissions of participants to access different scenes and will make sure that all the institutional norms and obligations are imposed.

After an institution is successfully specified to make the resulting 3D Virtual World more appealing, the specification may be annotated with additional 3D related components (objects, textures, animations etc). In this case the resulting 3D Virtual World has to be regenerated again.

3. RECTANGULAR DUALIZATION

To produce a map of the institution and the corresponding 3D Virtual World we use the rectangular dualization technique. Rectangular dualization was originally used to generate rectangular topologies for floor planning of integrated circuits [5]. 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, such as the Performative Structure.

A *rectangular dual* of a planar graph $G = (V, E)$ is a rectangle R with a partition of R into a set $\Gamma = R_1, ..., R_n$ of non overlapping rectangles such that no four rectangles meet at the same point and there is a one-to-one correspondence $\boldsymbol{f} : V \longrightarrow \Gamma$ such that two vertices u and v are adjacent in G if and only if their corresponding rectangles $\boldsymbol{f}(u)$ and $\boldsymbol{f}(v)$

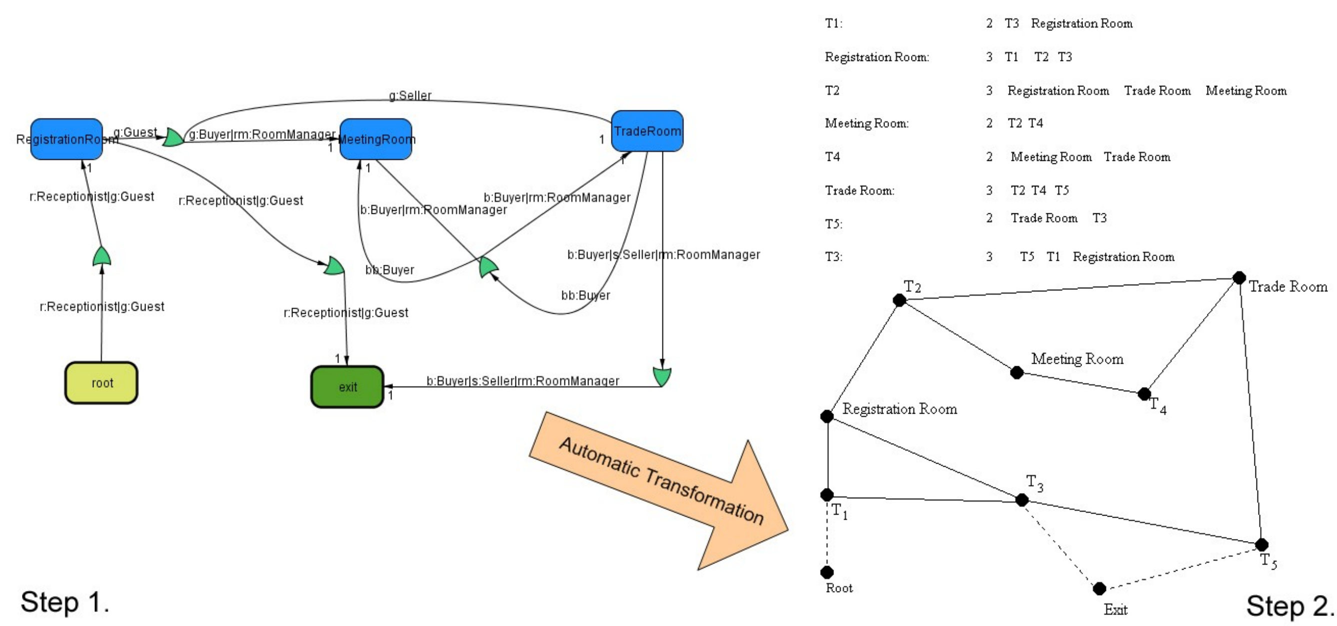

Figure 1: Transforming a performative structure into an OCoRD graph

share a common boundary.

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 [7]: 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.

4. AUTOMATIC GENERATION

The rectangular dualization technique is used for creating a 2-dimensional map of the institution. Further, on the basis of the map the 3D representation of the institution is created. This process goes through the following four steps.

On *Step 1* the redundant information contained in the Performative Structure is filtered out. If some nodes of the graph, for example, are connected with more than one arc only one randomly selected arc is left and all the others are deleted. Next, the performative structure graph is transformed into a form accessible by the OCoRD software. The OCoRD (Optimal Constructor of a Rectangular Dual) software is a tool aiming at the solution of floorplanning problems and at the orthogonal drawing of planar networks. Given a planar embedding of the graph, it accepts a numeric adjacency lists and, if the graph admits a rectangular dual, it returns its coordinates in the plane. 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. The implemented breaking method is optimal as it only adds a minimum number of such crossover vertices and inserts them in strategical positions, such that a single vertex may break two triangles or more [1].

Figure 1 presents an example of such graph transformation. The Performative Structure graph is shown on the left hand side. It corresponds to a very simple institution, which controls registration process, social interactions and trading activities of its inhabitants. Rectangular shapes on the picture represent scenes, triangular shapes are transitions, directed arrows (arcs connecting nodes) are connections. The labels above those arcs determine the role flow within the institution, defining which participants can access which scenes and transitions. In this example the 3D Electronic Institution consists of three functional scenes: RegistrationRoom, MeetingRoom and TradeRoom.

On *Step 2* of the algorithm the OCoRD software reads the transformed graph and creates its rectangular dual. In the generated rectangular dual the rectangles which were introduced because of the breaking points are removed and the adjacent rooms are reshaped to the size of the removed rectangles. Scenes and transitions are transformed into rooms and connections are visualized as doors. In the 3D Virtual World these rooms will be placed inside a building and the building itself is placed into a garden. Root and Exit scenes are not functional, they only determine entrance and exit points of the institution, so from the 3D Virtual Worlds prospective entering these rooms will force exiting the institution building and moving into the garden.

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 is automatically created as the rectangle surrounding the graphs 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 on the left side of Figure 2.

Step 3, transforming a 2D map of the institution into a 3D Virtual World, is pretty much trivial. 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 into the 3D Virtual World.

On *Step 4* the generated 3D Virtual World is visualized

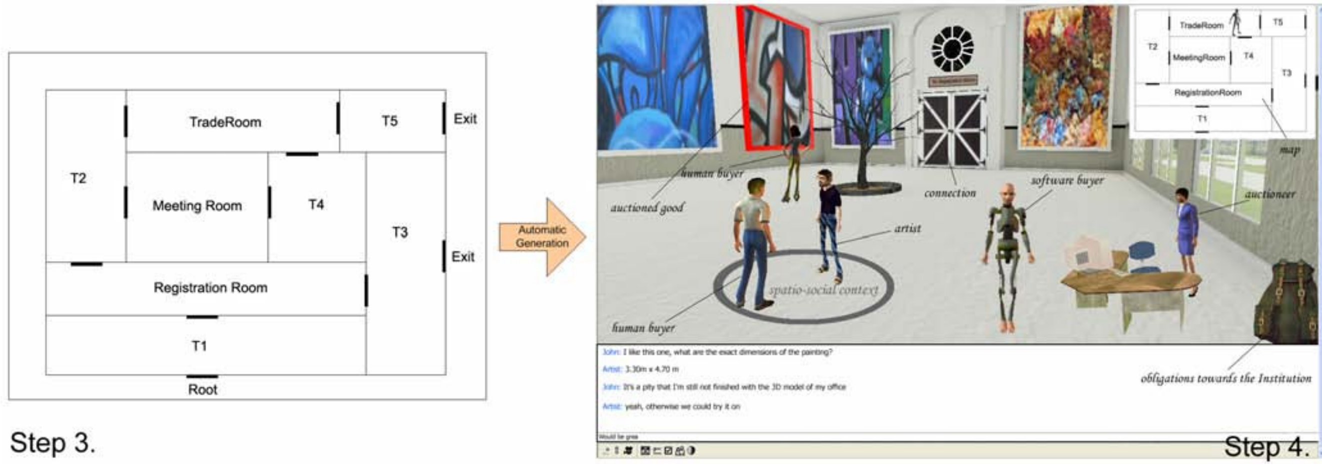

Figure 2: Generating the 3D representation of a Performative Structure graph.

and connected to the infrastructure, which controls the correct behavior of the participants. The result is similar to the right part of Figure 2. On this picture the map of the institution corresponds to the rectangular dual. A small human figure displays the current position of the user. The 3D View outlines the 3D representation of the scene currently observed by a user and displays the participants present in this room. The backpack on the right helps users to check their obligations towards the institution.

Below is a formal description of the algorithm for generation of a 3D representation of a Performative Structure.

Algorithm: *3D Virtual World Generation*

Input Performative Structure graph
Output 3D Virtual World

Transform the input into OCoRD graph
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 crossover vertices in G
Triangulate G [3]
Compute a Regular Edge Labeling [6]
Compute the rectangular dual coordinates
Delete external vertices and draw the map
Transform the map into a 3D Virtual World

The cost of this algorithm is logarithmic due to the fact that the matching algorithm we use [1] 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 [2]. 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 [3] (which triangulates without adding new separating triangles) before the computation of the geometrical dual.

5. CONCLUSIONS

We presented the algorithm for automatic transformation of a graph into a 3D Virtual World and particular application of the algorithm for automatic generation of the Euclidean representation of 3D Electronic Institutions from a Performative Structure graph. The rectangular dualization proved to be a reasonable technique for performing this task. Future work includes the development of the technology for automatic generation of non-Euclidean Virtual Worlds and conducting a user study for comparing the two approaches to understand which one is more appreciated by the users.

6. REFERENCES

[1] A. Accornero, M. Ancona, and S. 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.

[2] T. C. Biedl, P. Bose, E. D. Erik D. Demaine, and A. Lubiw. Efficient Algorithms for Petersen's Matching Theorem. In *Proceedings of the Tenth Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 130–139, N.Y., Jan. 1999. ACM-SIAM.

[3] T. C. Biedl, G. Kant, and M. Kaufmann. On Triangulating Planar Graphs Under the Four-Connectivity Constraint. *Algoritmica*, 19(4):427–446, Dec. 1997.

[4] A. Bogdanovych, H. Berger, C. Sierra, and S. Simoff. Narrowing the Gap between Humans and Agents in E-commerce: 3D Electronic Institutions. In *Proceedings of the 6th International Conference on Electronic Commerce and Web Technologies (EC-Web'05)*, LNCS 3590, pages 128–137. Springer-Verlag, 2005.

[5] X. He. On Floorplans of Planar Graphs. In *Proceedings of the 29th Annual ACM Symposium on the Theory of Computing*, pages 426–435. ACM Press, 1997.

[6] G. Kant and X. He. Two Algorithms for Finding Rectangular Duals of Planar Graphs. In J. van Leeuwen, editor, *Proceedings of the 19th International Workshop on Graph-Theoretic Concepts in Computer Science (WG'93)*, LNCS 790, pages 396–410, Utrecht, The Netherlands, June 1993. Springer.

[7] K. Kozminski and E. 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.