Faculty of Engineering and Information Technology
University of Technology, Sydney

An Agent-Based Service Oriented Architecture for Risk Mining

A thesis submitted in partial fulfilment of

the requirements for the degree of

Master of Science in Computer Science (research)

By

Jiahang Chen

July 2012

Certificate

I certify that the work in this thesis has not previously been submitted for a degree nor has it been submitted as 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

Production Note:
Signature removed prior to publication.

Acknowledgment

Foremost, I would like to express the deepest appreciation to my supervisor, Professor Longbing Cao, for his professional guidance, persistent help and continuous support throughout my Master study and research.

I would like to thank Dr. Xinhua Zhu for his patient guidance, and scientific advice. Without his generous support this dissertation would not have been possible.

Besides, I offer my regards and blessings to all of my co-workers at lab, and thank them for their support in my research and during the completion of this dissertation.

Last but not the least, I would like to thank my family: my parents, Without their encouragement, finishing this dissertation would be impossible; without them, nothing would have any value.

Jiahang chen

July 2012 @ UTS

Table of Contents

Abstract

Risk Mining (RM) is the process of analyzing data including risk information by data mining methods, with the mining results for risk prevention. In the last few years, some researchers have proposed the combination of data mining and agent technology (agent mining) to improve the performance of data mining methodology in the heterogeneous business environments. However, problems exist for further research with the application of risk mining systems in real industry environments to enhance the robustness of system architect, dynamic business process and model accuracy etc.

Therefore, in this thesis we present an Agent-based Service-oriented Risk Mining Architecture (ABSORM), which has been designed to facilitate the development of agent mining systems to address the above issues. This thesis focuses on developing the following strategies:

- **The integration of agent technology with web service.**
  In this framework, we propose a new and easier method, by which the system functions are not integrated into the structure of the agents, rather modeled as distributed services and applications which are invoked by the agents acting as controllers and coordinators. Therefore, techniques developed in this framework can improve the interoperability between different modules, distribution of resources, and the lack of dependency of programming languages.
- **The integration of agent technology with business process management.**
  In this work, we develop the autonomous agents that can collaborate in a business flow, which not only increases the reusability of the system, but also eases the system development in terms of re-usability of the computational resources. A group of agents solves problems in the following way: each individual agent solves

the problem individually, and then interacts with each other to finalize a business process.

- **The integration of agent technology with ensemble learning methods.**
  In this thesis, we are interested in developing agent-based ensemble learning strategies for risk mining: each ensemble agent individually gathers the evidence about model evaluation, and then ensembles learning methods like bagging and boosting is used to obtain prediction from the individually gathered evidence. Agent based ensemble learning can provide a critical boost to risk mining where predictive accuracy is more vital than model interpretability.

The proposed architecture has been evaluated for building an online banking fraud detection system and a student risk management system. These two applications have been proved to be a sophisticated, yet user friendly, risk analysis and management tool. They are modular, interactive, dynamic and globally oriented.

Keywords: risk analysis and management, risk mining, service oriented architecture, multi-agent system, BPEL, ensemble learning

Chapter 1

Introduction

This thesis presents an infrastructure for detecting, evaluating and managing risks in complicate and heterogeneous data scenarios. We have based our framework in Service Oriented Architecture (SOA) in order to have modularization, reusability, extendibility and maintainability. Moreover, we utilize autonomous agents that collaboratively work as workflow. A group of agents solves problems in the following way: each individual agent solves the problem at hand individually, and then interacts with each other to finalize a business process. Furthermore, ensemble learning can provide a critical boost to risk mining where predictive accuracy is more vital than model interpretability. Therefore, in this work we are interested in developing the agent-based ensemble learning strategies to improve the model accuracy and robustness. In this chapter we are going to motivate our framework and state our research goals. Then we will present a road-map of the contents of this thesis.

1.1 Background

Risk analysis is the process of defining and analyzing the dangers to individuals, businesses and government agencies posed by potential natural and human-caused adverse events. Risks can come from uncertainty in financial markets, project failure, legal liabilities, credit risk, accidents, natural causes and disasters as well as deliberate attack from an adversary, or events of uncertain or unpredictable root-cause.

Recently, risk analysis becomes a more and more serious problem in industrial fields which not only leads to security issues, but also causes huge economic losses. For example, with the rise and rapid growth of E-commerce, the convenience of banking from home or office is just one of many benefits that have led to the proliferation of online banking. But in the meantime, the attacks on online b also increased exponentially. A 2011 U.S. CyberSource Corporation Technology incident Report, compiled from suspicious activity reports of 300 merchants in North America, lists the revenue lost to online fraud in the last ten years, with an estimated loss in 2011 up to $3.4 billion (Fig 1.2). CyberSource Corporation is one of leading provider of Credit Card processing for Business, Electronic Payment & Risk Management Solutions in the world. In 2011, their customer reported losing an average of 1.0% of total online revenue to fraud (Fig 1.1). Although 2011 showed a decrease in the percent of orders lost to fraud versus the prior years, the estimated amount of revenue loss is up to approximately $3.4 billion, a 700 million increase over 2010 results. Also in 2011 an average of 2.8% of orders was rejected due to suspicion of payment fraud.

In the traditional management, corporations rely heavily on manual review to reduce risk. According to the CyberSource Online Fraud Report, in the last two years their customers still have around 27% orders routed to manual review. But as eCommerce sales continue to grow, huge amounts of digital data are collected and stored. According to our study on one Australian bank's online banking data, online banking fraud detection usually involves hundreds of millions of transactions per year (about 300,000 transactions per day). The

merchants continue to face the challenge of screening more online orders while keeping manual review staffing and fraud rates as low as possible. With more volume and limited resources, emphasizing and improving automated risk detection capabilities is a top priority for most of those companies. To be successful, risk analysis and management will need to adopt tools and practices to reduce the number which will be routed to manual review to an acceptable level, as well as enable their review team to assess the suspicious transactions more efficiently.

Fig. 1.1

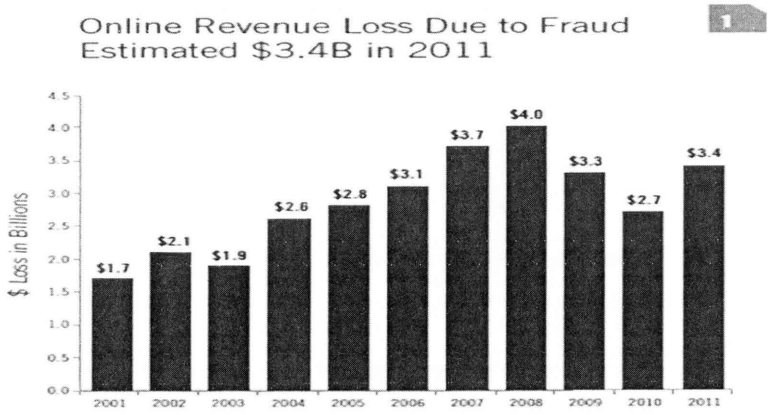

Fig.1.2

1.2 Motivation

For this purpose, *Risk Mining* (RM) has been proposed to apply the data mining as a tool for risk analysis. Data Mining (DM) is one of most popular and interdisciplinary field of computer science, which defines an analytic process that results in the discovery of new patterns in large data sets. A major goal of data mining is prediction, where a model or classifier is constructed to predict categorical labels. It discoveries the consistent pattern and/or systematic relationships between features, and then applies the detected patterns to new subsets of data.

Risk Mining is the process of analyzing data including risk information by data mining methods, and then uses the mining results for risk prevention. It consists of three major processes: risk detection, risk clarification and risk utilization, as shown in Fig. 3.

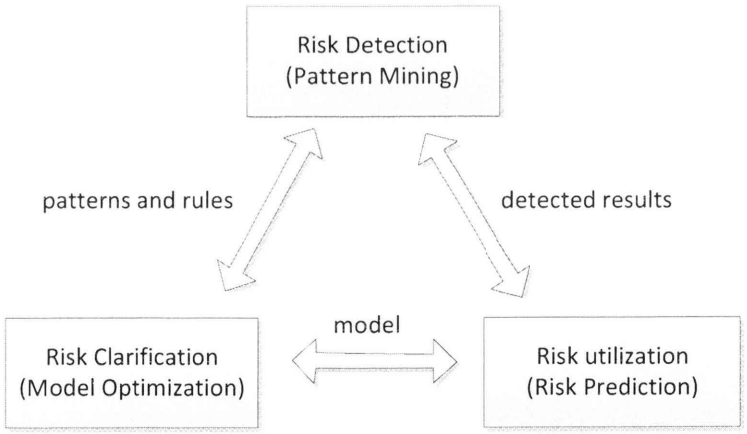

Figure .1.3

- **Risk detection**: mining patterns or other types of information which are unexpected to domain experts by data mining classification algorithms. Therefore, we also call this process *pattern mining*.

- **Risk clarification**: After pattern mining, domain experts and data miners can work together to clarify the model including the hidden mechanism of risk. This process is also referred as *model optimization*. If domain experts need more pattern or rule, the risk detection process will be repeated to get more information with finer granularity by collecting more data with detailed information or modifying the algorithm parameters.
- **Risk utilization**: The final model will be applied in a real world environment to prevent risk events. Also, the model will not always be high performance, and then regular analysis is requited. Then more data will be collect to repeat the risk detection and risk clarification. We call this process *risk prediction*.

A risk mining system is to be understood as the complete system: the database or data-warehouse, software for mining and analyses, the knowledge derived from it and the part of the system supporting final decision making in a business setting. Apart from the well-known limitations concerned with data quality, one often encounters major problems with the application of a risk mining system in real industry environments. Frequently occurring causes are:

- **Highly imbalanced large data set**

  According to our study, risks are always represented by a significantly less number of instances than the genuine transactions. For example, on online banking data there were only 5 frauds among more than 300,000 transactions on one day; the number of patients is much smaller than the number of healthy people in medical diagnosis data for normal populations. Obviously, the problem of learning from highly imbalance data is very common in risk mining.
- **Distributed data source**

  In many large enterprise data is normally stored at different database servers which locate in distributed sites, but some of them are relevant to each other (baik, Bala & Cho 2004). The traditional method to deal with such data distribution is to collect

data from different data sources, and store it in a centralized repository, where data mining methods can be used (baik, Bala & Cho 2004). Although, this approach can achieve accurate result of data analysis, it create enormous stress to computation and communication cost.

- **Data security and privacy**

  In many cases the fact that data is distributed among several sources is not the only problem. The problem is that the different data sources may correspond to different system or department, even different organizations, and that those owners may not be willing to allow other system to have access to their data either because of security mechanism or privacy concerns.

- **Model accuracy and robustness**

  In the traditional approach to data mining methods, there is only one model of all the data is going to be constructed. But the experiments prove that combining multiple models can improve accuracy and robustness in order to obtain better predictive performance. How to satisfy the precondition of combination of different models and achieve better prediction result is a critical issue.

- **Brittleness of system architecture**

  As scale, demand, volume, and rate of business change increase, the brittleness of traditional system architectures become exposed. Any significant change in any one of module will cause the brittleness of the systems to become a crisis: outdated or inefficient algorithm, lack of speed to involve new feathers and parameters, inability to rapidly shift to new competitive risks.

In the last few years, some researchers have proposed the framework which combine data mining and agent technology (agent mining) to improve the performance of data mining in the heterogeneous data environments. The characteristics of the agents make them appropriate for developing dynamic and distributed system as they possess the capability of adapting themselves to the users and environmental characteristics. Recently, agent mining

is a relevant area of research. The interaction and integration between agent and mining can greatly complement and strengthen each side. However, both of them should face some issues of the integration and interaction of two technologies (Cao, Gorodetsky & Mitkas 2009, p. 64). The major objective of this thesis is to design a framework to facilitate the development of agent mining integration and interaction especially in risk analysis and management scenarios.

1.3 The Challenge

Risk analysis and management is a sophisticated, yet user friendly, business decision support tool, which can release the burden of manual review as much as possible. It should be modular, interactive, dynamic and globally oriented.

Therefore, the challenges to define such an infrastructure may include:

- How to improve the efficiency of analysis from raw data with large scale and distributed sources. Recently, huge amount of data are collected and stored in the digital form for a variety of purposes. Also most of these data are stored in different database servers which locate in geographically distributed sites. It is a quit critical challenge to advance the risk identification and assessment.
- How to improve the performance of data mining classifiers in imbalance learning by combining ensemble methods, cost-sensitive techniques and Multi-agent system. For example, how to improve individual performance maintaining ensemble performance, determining how to redistribute case among the agents to achieve better distribution.
- How to customize workflow for different risk scenario, and how to parse those workflow to executable notation. Moreover, since the construction of a workflow for a specific task can become quite complex, how to increase the reuse of workflows through the implementation of specialized workflow repositories.

- How to organize agents into ensembles, and how they can collaborate to achieve the ensemble effect. Therefore, we need individually organize an ensemble to determine when an agent should solve a problem individually or when some agents should cooperate.
- How to explain the analysis results to the users from other areas of expertise. Data visualization is always one of the most popular research topics in computer science field. In risk analysis and management system, the results must be reasonable and explainable.

1.4 The Goals

The main goal of the thesis is to design a flexible and high performance risk mining architecture, specifically in heterogeneous date scenario where complicated analysis processes can be defined dynamically.

Moreover, in this thesis we have also several goals related with business process management, multi-agent system and ensemble learning:

- The first goal is the integration of the three related areas (service-oriented architecture, multi agent systems and ensemble learning) and formally defines the agent-based service-oriented risk mining architecture (ABSORM).
- To achieve the modularity and extendibility in ABSORM by importing web service mechanism. Thus, allowing users to add new resources (data sets, servers and algorithms) by simply advertising them to the application servers.
- Develop techniques to perform the reuse of risk analysis process in a dynamic way. Business process management would be preferable to perform the pattern mining under different conditions, since in the different scenario risk identification may include different operations. Business Process management should be able to

support users in the process of building the workflows, composing external web series and monitoring their execution

- Extend distributed service bus (DSB) to administrator the agents which connect distributed services and resources. In such a way agent's autonomy, data privacy, and individual service access are preserved in the autonomous agents. The four processes of DSB (discover, match, plan and compose) have to be rethought. Specifically, in this thesis we focus on how ontology support can be adapted to work in multi-agent systems.
- Analyze the ensemble effect and its preconditions in a wide range of situations so that measures can be defined to characterize ensembles, and thus predicted their performance. These measures are required so that agents try to behave as an ensemble can measure how well will they perform as an ensemble and decide which actions should be taken to improve their ensemble performance.
- Develop visual metaphors to help users from other areas of expertise understand the output of data mining algorithms and analysis result. The result should be converted to a business language which people without technique background can easy to understand and tackle the risks.

1.5 The Contribution

In this thesis, we will present an Agent-Based Service Oriented Risk Mining Architecture (ABSORM) to deal with risk analysis in heterogeneous business requirement and data situation. This architecture is aimed at the interoperability between different modules, distribution of resources, and the lack of dependency of programming languages. Our contributions in this work are to develop the following strategies:

- In this framework, we propose a new and easier method, where the functionalities of system are not integrated into the structure of the agents; rather they are modeled as distributed services and applications which are invoked by the agents acting as

controllers and coordinators. Therefore, techniques developed in this framework can improve the interoperability between different modules, distribution of resources, and independency of programming languages.

- In this work, we are interested in developing the autonomous agents that can collaboratively work as business flow, which not only increase the usability of the system, but also ease system development in terms of re-usability from the computational resources. A group of agents solves problems in the following way: each individual agent solvers the problem at hand individually, and then interacts with each other to finalize a business process.
- In this thesis we are interested in developing agent-based ensemble learning strategies for risk mining: each ensemble agent individually gathers the evidence about model evaluation, and then ensemble learning methods like bagging and boosting are used to obtain prediction from the individually gathered evidence. Agent based ensemble learning can provide a critical boost to risk mining where predictive accuracy is more vital than model interpretability.
- One of the first significant advantages of this infrastructure could be flexibility and efficiency. In the current data mining research there is a focus on efficiency improvement of algorithms for knowledge discovery. However, improving the algorithms is often not efficient in big data. Distributed computing can divide a large computing task into many small tasks, each of which is solved by one or more computers. It may improve the process velocity noticeable. Multi agent techniques can support data mining through distributed mechanisms to coordinate multiple agents so that they can jointly accomplish a give data mining task (Chaimontree, Atkinson and Coenen 2011). The distributed data can be mined effectively without the need to first move the data into a data warehouse. Also each site can be modeled as agent that has control over its private data without security concern. Both of them only collaborate when necessary.
-

- The cooperation of ensemble learning and multi-agent system can enhance the expressive power of the classifiers and reduce the impact of having a small or highly imbalanced training set.
- Web Services offer an evolution of the Internet-standards-based distributed computing model, foster a change from tightly coupled, rigid, and static solutions that focus on implementation technologies, to loosely coupled, flexible, and dynamic solutions focusing on dynamic business models.

1.6 The Thesis Organization

In this section, we will present a road map of the thesis, shortly summarizing the contents of the rest of the chapters and appendices. Figure 1.4 shows a condensed view of the contents of the thesis.

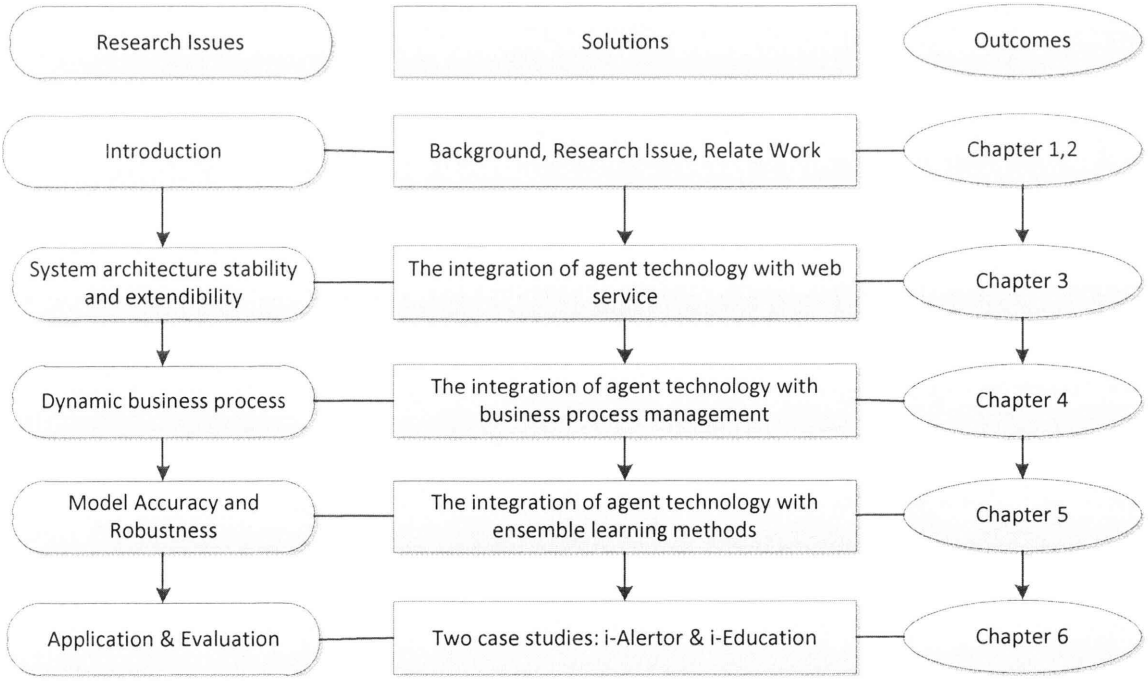

Figure.1.4

- **Chapter 2** presents an overview of the state of the art in the areas related to the research presented in this thesis. Firstly, related work in agent mining system is presented, emphasizing in the integration and interaction between multi-agent system and data mining. Then, the principles and methodologies of service-oriented architecture are considered, specifically the techniques and protocols we will use in our framework. Finally, related work in Ensemble Learning is presented, emphasizing in the work related to the ensemble effect and on different methods for creating ensembles.
- **Chapter 3** introduces the agent-based service-oriented architecture used integrates the multi-agent system and service-oriented computing. Then, we describe the major components in this architecture. Basically, Agent-Based Service Oriented Architecture is service-oriented architecture with distributed service bus (DSB) uses multi-agent technique to discover, match, plan and composing the services dynamically. We will explain the formalization used to specify interaction between agents and services and which is one of our research goals inside this framework. Finally, we present some capabilities that the agents inside the agent-based service-oriented architecture need in order to be able to apply the techniques that we will present in the following chapters of this monograph.
- **Chapter 4** presents the agent-based BPEL engine aimed at building workflows, composing the external Web services and monitor their execution. Specifically, we are going to present two alternative automatic procedures of risk mining, namely predefined workflow and dynamic workflow, and empirically evaluate them.
- **Chapter 5** introduces the collaboration strategies of a group of agents that joins together in order to improve model accuracy and robustness by using ensemble learning methods. Each ensemble agent individually gathers the evidence about model evaluation, and then ensemble learning method like bagging or boosting is

used to obtain a global prediction from the individually gathered evidence. Moreover, we will present the ensemble learning methods to deal with imbalanced data and empirically evaluate its performance.

- **Chapter 6** presents the case studies which implements the framework we proposed in this thesis in two real industry projects. The first one is an online banking risk management system (i-Alertor) which we develop for one major Australian bank. i-Alertor integrates various features and data mining models, and aims to consolidate different sources of resources for systematic problem-solving. The second case we will demonstrate is a student performance risk management system (i-Education) for university. i-Education builds various business processes for different subjects, and aim to identify students who are potentially at risk of failing those subjects, also the key risk factors.
- **Chapter 7** first summarizes the work presented in this thesis. Then the contributions with respect to risk mining, multi-agent systems and ensemble learning are presented. The chapter closes with a discussion of future lines of research.
- Appendix A lists my publications in the last two years.

Chapter 2

State of the Art

In this chapter, we will present an overview of the state of the art in the areas related to our work. We have divided it into three areas:

- Agent-based Distributed Data Mining
- Service Oriented Architecture
- Ensemble Learning

The following sections describe each the work done in each one of the previous areas in detail.

2.1 Agent-based distributed data mining

Data mining researches the process that results in the discovery of new patterns in large data sets. Agents comprise a powerful technology for the analysis, design and implementation of autonomous intelligent systems that can support and enhance the knowledge discovery process in many ways. For instance, agents can contribute to data selection, extraction, preprocessing, and integration. The interaction and integration between agent and mining can greatly complement and strengthen each side. Some complicated challenges may be effectively and efficiently tackled through agent-mining interaction.

It's obvious that there are two important components in multi-agent distributed data mining system: agent-driven distributed data mining and data-mining driven agents. The critical challenges they met can be alleviated by the other one. But both of them should face some issues of the integration and interaction of two technologies (Cao, Gorodetsky & Mitkas 2009, p. 64).

2.1.1. The requirement of agent-driven distributed data mining

Agent-driven distributed data mining is a branch research area of data mining which focuses on deal with analysis the distributed data and computing resources in distributed scenario (Silva et al. 2005). In distributed scenario data sources and computation are located in multiple independent sites and communicate through message passing. The communication price in distributed scenario is very expensive. Moreover, different distributed sites contain different constrains and privacy mechanism (Silva et al. 2005).

Recently, following the development of emerging applications and information technologies and the peculiarities of data sources, distributed data mining should face an increasing number of critical issues (Cao, Gorodetsky & Mitkas 2009, p. 64). Therefore, to build am agent-driven distributed data mining should concern the following requirements:

1) the multiplicity of data sources

In distributed environment, system need dynamically handle varied data sources. The first one attribute is the isolation of data sources. Normally data sources are isolated from each other physically, but the contents inside those data source will be related to each other in business analysis (Cao, Gorodetsky & Mitkas 2009, p. 64). There are two styles to store data in distributed data sources: homogeneous style and heterogeneous style. The global table is horizontally partitioned in homogeneous style; the distributed tables are the subsets of the global table. However in heterogeneous style, the global table is vertically partitioned; each of the distributed tables is part of the global table (Silva et al. 2005). The goal for DDM in business analysis is to collect all of the relevant data together through centralized integration or localized communication (Cao, Gorodetsky & Mitkas 2009, p. 64). In addition, the multiplicity of data sources also includes the mobility of source data.

2) the interoperability of DDM

The interoperability of distributed data mining contains the collaborations of agents and the interaction of agents. Therefore the architecture of the system must be open and flexible for agents to communication and interaction (Moemeng et al. 2009). However in most case, communication is a bottleneck for the whole platform. The reason for this phenomenon is the primary goal for many DDM methods is to limit the number of messages sent and the data column (Silva et al. 2005).

3) Privacy of data source

In distributed environment, privacy is a very important element in distributed data mining because some data sources do not permit other system to visit their data or manipulate their data (Silva et al. 2005). Therefore, data centralization is not possible in those sites. Agents must analyze those data in a distributed site instead of download or

transfer the data to other server (Silva et al. 2005). And also the agents must get authority to access the data firstly before performing a collection or analysis process (Cao, Gorodetsky & Mitkas 2009, p. 64).

4) Time Zone Constraint

Normally, two of the important constrain in distributed environments are time constraints and organizational constraint (Cao, Gorodetsky & Mitkas 2009, p. 64). The different of time zone will affect the data in different type of storage. In the other hand, organizational constraint will make DMM more complex (Cao, Gorodetsky & Mitkas 2009, p. 64).

2.1.2. The requirement of data mining-driven agent

Data mining-driven agent is an agent system that embeds data mining capability into agents, in this case, agents are able to learn and reason respecting the specific domain. In current system, the data mining processing is difficult and time-consuming for even the most powerful computer to handle (Cao, Gorodetsky & Mitkas 2009, p. 64).

In Multi Agent System (MAS), agents must be pro-active and autonomous. Agents analysis their situation and dynamically decide to perform tasks and interact with other agents. The knowledge of the agents will direct the agents' action, which base on the existing domain theory (Silva et al. 2005). However, recently in some complex situation, this knowledge is a result of data analysis to pre-existing domain knowledge. Scalable analysis of data may require advanced data mining for detecting hidden patterns (Silva et al. 2005).

2.1.3. The challenge for the integration and interaction

Multi Agent System (MAS) technology and Distributed Data Mining (DDM) can alleviate some of the other's critical challenges, but they should also face many enhancement and integration issues (Cao, Gorodetsky & Mitkas 2009, p. 64).

However, the mutual issues become the obstacles for people to implement the Agent Mining System, which include:

- Architecture and infrastructure
  Building a stable, flexible and extendable framework which can support the varied of data sources and a great number algorithm and functions is always a huge challenge (Cao, Gorodetsky & Mitkas 2009, p. 64).
- Human intelligence and human roles:
  It is a big challenge to add the human roles in problem-solving systems (Cao, Gorodetsky & Mitkas 2009, p. 64).
- Constraint and environment
  Since the agents and data mining need interact with environment, they must tackle the constraint around the system (Cao, Gorodetsky & Mitkas 2009, p. 64).
- Nonfunctional requirement
  Nonfunctional requirements are also very important in data mining and agent system, like: efficiency, effectiveness, action ability, user and business friendliness (Cao, Gorodetsky & Mitkas 2009, p. 64).

2.1.4. Extendible multi-agent data mining framework

In 2009, Albashiri & Coenen introduced a generic and extendible multi-agent data mining framework which is a hybrid peer to peer agent-based system comprise a collection of collaborating agents that exist in a set of containers. In their paper, they point out that because of different nature of DM and the wide range of task encompassed, the MADM systems must define a sophisticated communication mechanism. In the EMADS architecture, they use a system of mediators and wrappers coupled with an Agent Communication Language (ACL) such as FIPA ACL in JADE.

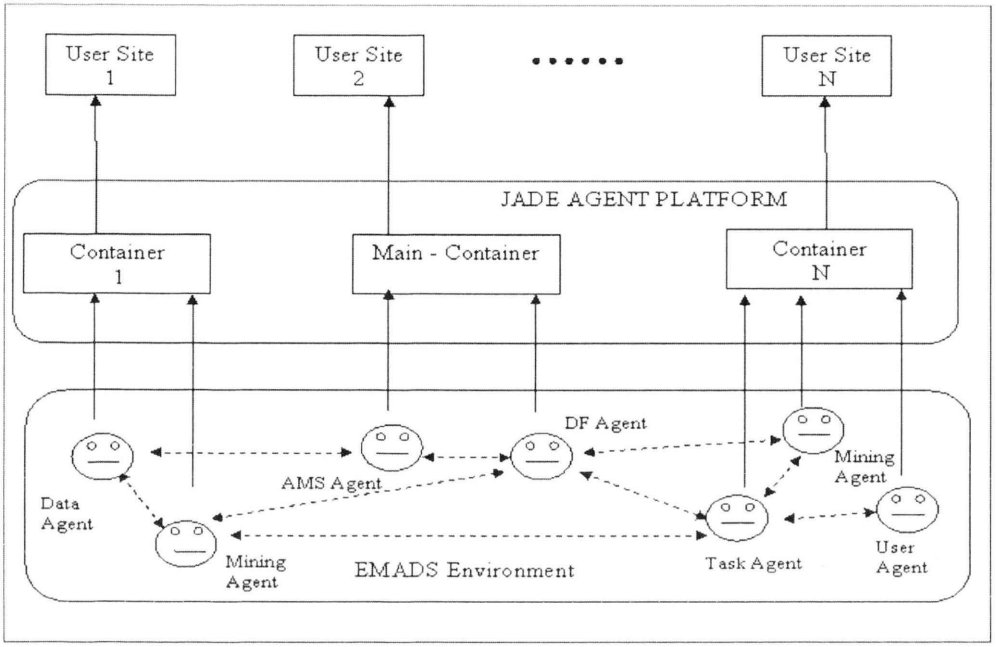

Fig.2.1. EMADS Architecture

Figure 2.1 shows the architecture of an extendible multi-agent data mining approach, which presents the relationship among the components in this system. User site can sent request to container. The housekeeping agents (an Agent Management System (AMS) agent and a Directory Facilitator (DF) agent) in the main container will synchronize the required agent and allocate the task (Albashiri & Coenen 2009).

In user site, a user agent runs on the user's local host. The user agent may have several different modes according to the nature of the participants (Albashiri & Coenen 2009). The participants may include:

- Developers: Developers have full access to the whole system and will mainly focus on the data mining algorithm development.
- Data miners: Use the User agents and task agents to set up data mining tasks, only have limited access privilege.

- Data contributors: The main job of them is to prepare data set, only have limited access privilege (Albashiri & Coenen 2009).

The main responsibilities of a user agent are to receive the user input, to setup the data mining task and to display the result. For example, system can use a web based interface through which they can access databases located at different sites and manipulate data mining facilities (Baik, bala and Cho 2004). The user can express a data mining plan and execute it by standard user interface, on completion the result will be sent back to user interface to display. Users have not necessary to define or know which agent or agents should be employed to perform this task (Albashiri & Coenen 2009).

The mediator (the DF and AMS agents) coordinates the communication between several by a knowledge base which includes the capabilities of the various EMADS agents. Normally three basic processing will be performed after users set up a task:

- **Service Registration** Where providers (agents who wish to provide services) register their capability specifications with a facilitator.
- **Request Posting** Where User Agent (requesters of services) construct requests and relay them to a task agent.
- **Processing** where the task agent coordinates the efforts of the appropriate service providers to satisfy the request (Albashiri & Coenen 2009).

One of the principal objectives of EMADS is to provide an easily extendible MADM framework which is implemented by a system of wrappers. The wrappers mechanism can wrap the data and tools as EMADS agents, so that this framework can easily accept new data sources and new data mining techniques (Albashiri & Coenen 2009).

The framework defines an abstract class of agent object and all of the specific agents should implement this class. In the abstract class, there are some simple attributes and

behaviors that all of agents should comply to. After agents inherit those interfaces, they can be introduced and used immediately as EMADS agents (Albashiri & Coenen 2009).

The wrappers include two types:

- Data wrappers:
  Data wrappers are used to wrap the data sources to data agents which include the detail info about data source location and Meta structure. So that other agents can access those data sources easily by communicating with these agents (Albashiri & Coenen 2009).
- Tool wrappers:
  Tool wrappers are mainly used to wrap the data mining tools like data mining algorithm, data pre-processing method and other data mining operation tools.

EMADS can tackle some critical issues of multi-agent data mining meet and provides some extent functions. Firstly, Framework become easily to accept new data sources and new data mining techniques, just wrap them as one of the agent inside the system (Albashiri & Coenen 2009).The wrapper mechanism allows system to integrate new resource or new functions without resetting the system and the of the resources will be kept in repository for future use. Moreover, housekeeping agents play very important roles in the agent management and knowledge discovery, which present autonomy and intelligence in several components of the proposed system (Moemeng, Cao & Zhang 2008). The principal advantages offered by the system are that of experience and resource sharing, flexibility and extendibility, protection of privacy and intellectual property rights and information hiding (Albashiri & Coenen 2009).

2.2 Service oriented architecture

Service-oriented architecture (SOA) is a flexible set of design principles used during the phases of systems development and integration in computing. CBDI recommends SOA is more usefully defined as:

> *The policies, practices, frameworks that enable application functionality to be provided and consumed as sets of services published at a granularity relevant to the service consumer. Services can be invoked, published and discovered, and are abstracted away from the implementation using a single, standards-based form of interface. (CBDI)*

A deployed SOA-based architecture will provide a loosely-integrated suite of services that can be used within multiple business domains. The adaption of SOA can alleviate the burden enterprises take to create redundant application for each of business requirements, increasing the value of current systems and automating new processes (Huhns and Singh 2005). Recently SOA become more and more mature and popular technique to guide the development of enterprise software infrastructure and application. Most all of the large software technology corporations like IBM, Microsoft and Oracle etc. raise their own service-oriented solutions for most of business requirements.

2.2.1. Elements of service-oriented architecture

Each service provides a platform-neutral interface contract that allows for service interaction across distributed and heterogeneous platforms in a uniform and universal manner (Chen, Cohen and Hamilton 2005). These services are well-defined business functionalities that are built as software components that can be reused for different purposes. A web service architectural model include three major parts: service provider, service requestor and service registry and broker (Fig 2.2). A service can be made available by publishing itself to service registry where it may be discovered by a service consumer.

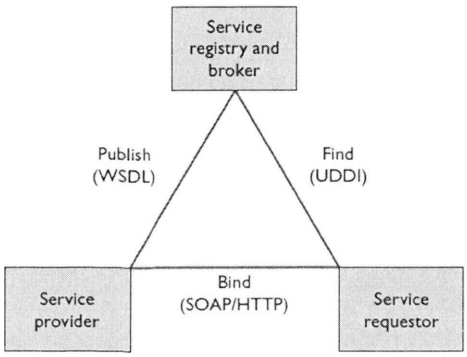

Figure 2.2 Web services architectural model (Chen, Cohen and Hamilton 2005)

- Service requestor: service requestor can be an application, a software module or even a service which require other services. Service requestors inquire the services description from service registry and binding services by using network connection. Then it can execute those services through interface.
- Service provider: service provider is a software entity which includes an exact network address. It receives and executes the request from service requestor. Service provider publishes the service description and interface to service registry, so that service requestor can easily find and execute the service.
- Service registry and broker: service registry and broker provides the services lookup function. It maintains a usable service list and allow service requestor inquire the service provider interface.

Every entity in service-oriented architecture may act as one or more roles of service provider, service requestor and service registry. Service-oriented architecture includes the following actions:

- Publish: to make the service detectable. Service description must be published so that requestor can find and invoke them.
- Find: to allocate the services. Service requestors look for the services they need from service registry.

- Bind and invoke: After service requestors find service description, they can bind and invoke the service through the information of service description.

2.2.2. Characters of service-oriented architecture

SOA is fundamentally about building modular applications, but modularity alone is not the only distinguishing characteristic of SOA. The most critical principles of the adaption of service-oriented architecture may include (Altman 2011):

- Each system capability is delivered as a discrete service. Service oriented is the primary principle guiding the SOA design paradigm. Service orientation reduces redundancy within a system and across systems. It also makes it easier to access a system's capability and business value (Altman 2011).
- Separation of concerns guides service design. A complex application will be separated into more easily developed modules, so that each part can be developed and maintained autonomously. Separation of concerns facilitates systems agility and component reuse (Altman 2011).
- Loose coupling reduces dependencies among services and makes the remaining dependencies explicit. Loose coupling improves a system's maintainability by reducing the impact that a change to one component has on other components (Altman 2011).

Services based on these principles are modular, distributable, loosely coupled, discoverable and swappable.

2.2.3. Three styles of SOA

Applications that provide the five characteristics of SOA can be broken down into three major styles:

- Remote procedure call (RPC)-style SOA (the traditional SOA)

RPC-style SOA formalizes and standardizes the interfaces between service requestors and providers by using document-style SOAP middleware or RPC-style SOAP middleware (Altman 2011).

- Web-style SOA (WOA)

  Web-style SOA are more loosely coupled that RPC-style SOA application because it follows a light weight web service principles (REST) and use the standards developed for the Web like URIS, XML or HTTP (Altman 2011).

- Event-driven SOA (the intersection of EDA and SOA)

  Event-driven SOA is more decoupled that the other types of SOA since the communication between event source and event consumer is based on an asynchronous relationship where event source sends notification to event consumer but receives no response (Altman 2011).

2.3 Ensemble learning

In machine learning, ensemble learning method is used to combines multiple models into one, which can improves accuracy and robustness over single model methods to obtain better predictive performance. Usually, when we have a set of training examples and a set of classifiers trained from them, we can take a sample of all the classifiers and aggregate them so as to reduce the classification error compared to the best individual classifier.

Bagging is one of the ensemble-based meta-learning algorithm which employs bootstrap sampling to generate many training set from the original training set, then building multiple base learners from each of those training sets and aggregating their predictions by simply averaging based on regression tasks or majority voting for classification tasks to make the final predictions (Zhou, Wu and Tang 2001). Bagging is known to be particularly effective when the classifiers are unstable for data dynamics or distribution.

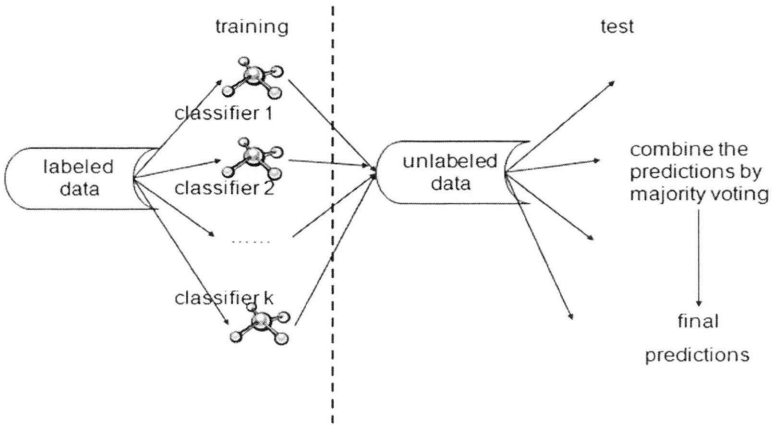

Fig.1. Ensemble of Classifiers-Consensus (Zhou, Wu and Tang 2001)

Fig. 1 shows the procedure of bagging. Firstly, it creates k data sets from a database applying the sampling with replacement scheme. Then, a learning algorithm is applied to each sample training data set to generate K classifier. After that, for an object with unlabeled data, it makes predictions with each of the K classifiers. In the end, it selects the most frequently predicted decision to be the final predictions. According to the tutorial presented by Seni and John in KDD 2007, bagging has been proved by server studies that it can improve the predictive performance by reducing the variance of the mean squared error, especially when the classifiers are unstable.

Consider idealized bagging estimator:

$$\bar{f}(x) = E\,(\hat{f}_z(x))$$

The mean squared error is:

$$E\left[Y - \hat{f}_z(x)\right]^2 = E\left[Y - \bar{f}(x) + \bar{f}\,(x) - \hat{f}_z(x)\right]^2$$

$$= E\left[Y - \bar{f}(x)\right]^2 + E\left[\bar{f}\,(x) - \hat{f}_z(x)\right]^2 \geq E\left[Y - \bar{f}\,(x)\right]^2$$

Boosting constructs a composite classifier by sequentially training classifiers while putting more and more emphasis on certain patterns. The principles of boosting are to boost a set of weak learners to a strong learner and make records currently misclassified more important. In contrast to the bagging algorithms for imbalanced data sets, Boosting has been proved more attractive since it has more complicate strategies (Hido and Kashima 2008).

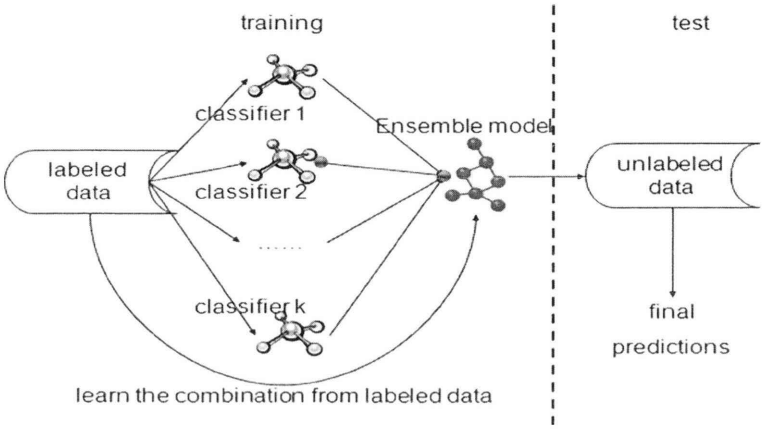

Fig.2. Ensemble of Classifiers-learn to combine (Zhou, Wu and Tang 2001)

AdaBoost is a widely used boosting method for classification. In the beginning, this algorithm maintains a probability distribution $D_t(i)$ over the original training set and set uniform weights on all the records. In each round *t*, it creates a bootstrap sample based on the weights. Then a classifier on the sample will be trained and applied on the original training set. After that, the records that are wrongly classified will have their weights increased and the records that are classified correctly will have their weights decrease. If the error rate is higher than 50%, the cycle will start over. Final prediction is the weighted average of all the classifiers with weights representing the training accuracy.

Fig 3 is the steps of AdaBoost algorithm:

Given: $(x_1, y_1), \cdots, (x_m, y_m)$ where $x_i \in X, y_i \in Y = \{-1, +1\}$

Initialize $D^1(i) = \frac{1}{m}$

For t = 1, ⋯, T:

1. Train base learner $h_t \rightarrow Y$ using distribution D^t
2. Choose weight updating parameter: a_t
3. Update and normalize sample weights:

$$D^{t+1}(i) = \frac{D^t(i)\exp(-\alpha_t h_t(x_i) y_i)}{Z_t}$$

Where, Z_t is a normalization factor.

Output the final classifier:

Fig.3. AdaBoost Algorithm (Sun et al. 2007)

Chapter 3

Agent-Based Service Oriented Architecture

In this chapter, we will introduce the agent-based service-oriented architecture used throughout this thesis. Firstly, we will present the infrastructure which integrates the multi-agent system and service-oriented computing. Then, we describe the major components in this architecture. Basically, Agent-Based Service Oriented Architecture is service-oriented architecture with distributed service bus (DSB) uses multi-agent technique to discover, match, plan and compose the services dynamically. We will explain the formalization used to specify interaction between agents and services and which is one of our research goals inside this framework. Finally, we present some capabilities that the agents inside the agent-based service-oriented architecture need in order to be able to apply the techniques that we will present in the following chapters of this thesis.

3.1. The integration methodology

Today's risk analysis not only has become more complex but also much more dynamic. From the business point of view this requires risk mining that is not only affordable but also highly flexible and adaptable. In order to achieve this, the idea of service-oriented architecture (SOA) came up for reusing resources and sharing compatible platforms or architectures. Service oriented architecture is a framework that enable application functionality to be provided and consumed as sets of services published at a granularity relevant to the service consumer. A deployed SOA-based architecture will provide a loosely-integrated suite of services that can be used within multiple business domains. The adaption of SOA can alleviate the burden enterprises take to create redundant application for each of business requirements, increasing the value of current systems and automating new processes. Also it leads to highly distributed and fluid software systems that may even cut across the boundaries of enterprises.

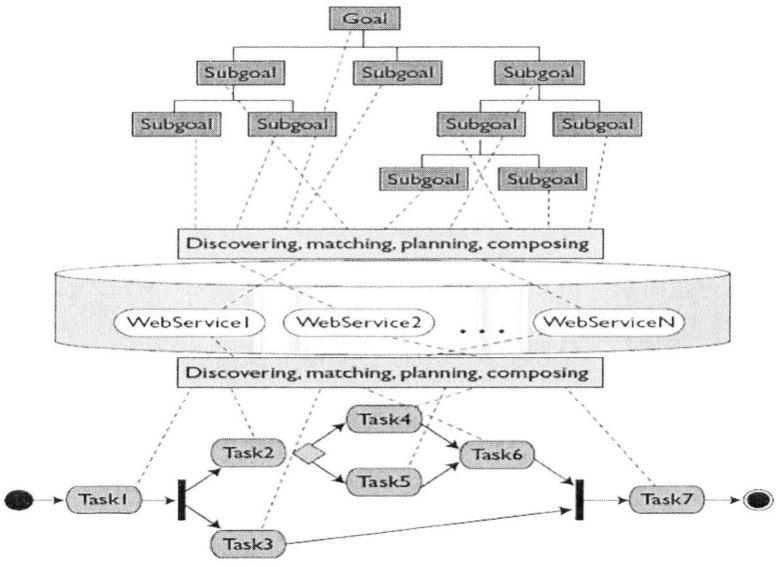

Figure 3.1 Service Oriented Architecture (Huhns and Singh 2005)

Engineering service-oriented architecture is a process of discovering and composing the proper services to satisfy a specification, whether it is expressed in term of a goal graph, a workflow, or some other model (Fig 3.1) (Huhns and Singh 2005). Therefore service composition plays a key role in this infrastructure. In this case, agent technology can be considered as a very successful support to SOA. Agents have a set of characteristics, such as autonomy, reasoning, reactivity, social abilities, pro-activity, mobility, organization, etc. which allow them to cover several needs for dynamic environments, especially ubiquitous communication and computing and adaptable interfaces. The most important role that agents should play in a SOA is to efficiently support distributed computing and to allow the dynamically composition of Web services. The combination of Web services and software agents provides a promising computing paradigm for efficient service selection and integration of inter-organizational business processes.

The merging of service-oriented and agent base approaches has been a hot topic of research in recent years. Cao et al. (2005) presented an agent service-oriented approach to address web services discovery and composition of web services. In their systems, providers' services are wrapped into providers' agents, and SOAP and WSDL messages are used to interact with agents. Liu et al. (2004) proposed a conceptual model of agent-mediated web services for intelligent service matchmaking. Rigole et al. (2006) have used java technology to create agents on demand into a home automation system, where each agent is defined as a service in the network. However, most of the above mentioned agent-based approaches are not open at all because the framework is closed and services and applications must be programmed using a specific programming language that support their respective proprietary.

In this chapter, we describe an Agent-Based Service Oriented Architecture (ABSOA). One of the most important characteristics is the use of intelligent agents as the main component in employing a service-oriented approach, focusing on distributing the majority of the

systems' functionalities into remote and local services and applications. The architecture proposes a new and easier method, where the functionalities of system are not integrated into the structure of the agents; rather they are modeled as distributed services and applications which are invoked by the agents acting as controllers and coordinators. The main ideas, advantages and resulting contributions of this framework are as follows:

- Dynamic business formation and effective selection of services. As the rate of business change increase, it may be hard to analyze the required services select them manually. In this infrastructure, the fixed pattern of business requirements can be descripted as predefined or dynamic business process models by BPEL4WS. Then the brokering service provided by agent technologies will be used to select and connect the business process to corresponding services efficiently.
- Service dynamic composition. Once new services are confirmed to join this platform, the services are required to register into service registry (UDDI) and compiled dynamically according to predefined abstract business rules so that a business process can be carried out.
- Flexible cooperation strategies. The dynamic characteristics of business collaboration generally require the introduction of flexible cooperation strategies in the consortium formation processes. An intelligent agent is an autonomous entity that every agent can communicate with their neighbors and decide when to collaborate and with whom to collaborate. The behaviors revealed by the agents can be directed by the cooperation strategies to achieve the best prediction performance.

3.2. Overview of the architecture

The agent-based service-oriented architecture (ABSOA) for risk mining is illustrated in Fig.3.1 where services and applications are managed and controlled by multi-agent platform. Proposed architecture consists four layers: user interface application, business

process, service choreography, and service application. This architecture can be viewed from the consumer perspective or provider perspective. The providers should not be interested in the application that the service is consumed in, but only focus on the services application architecture. Similarly the consumer should not be interested in the implementation detail of the service, but only focus on their application architecture and the business processes structure.

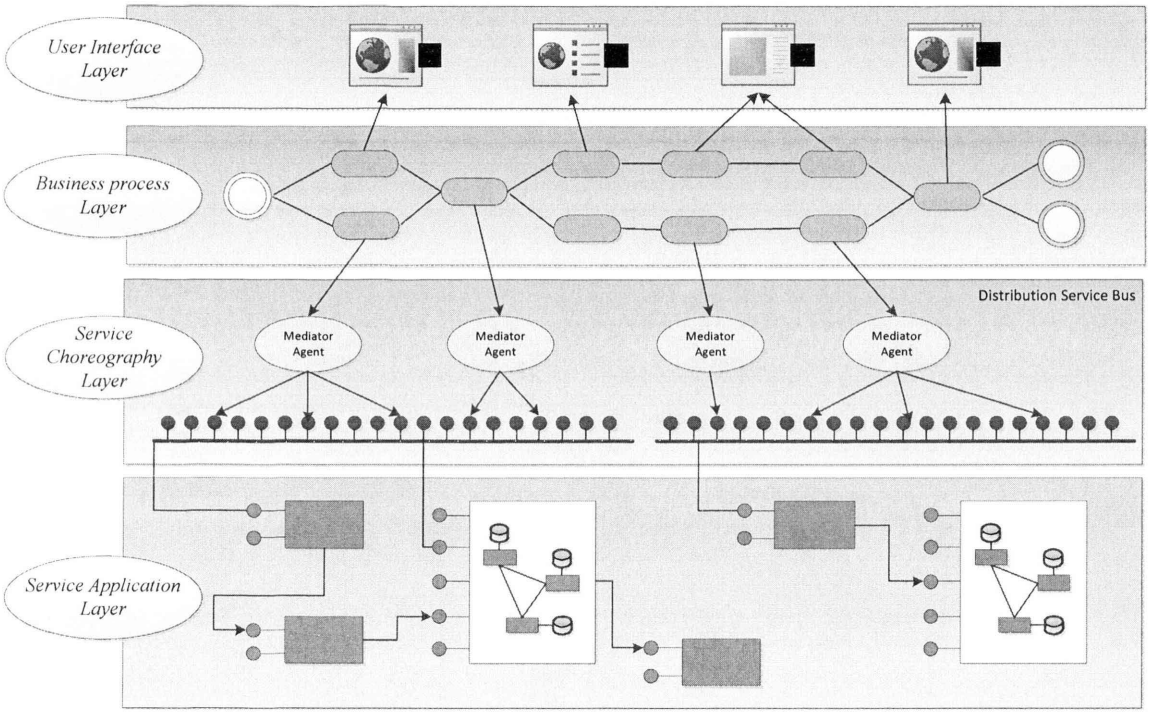

Fig.3.1. agent-based service-oriented architecture

Both of two perspectives are connected by service choreography which provides a bridge between the implementations and the implemented applications, creating a logical view of sets of services which are available for use, invoked by a common interface and management architecture. Agent-based service choreography are the core of ABSOA. There are different kinds of agents in the architecture, each one with specific roles,

capabilities and characteristics. They can be used to build high-level models with flexible interaction patterns. At the implementation level, UDDI, WSDL and SOAP proved such capacities as discovery, deployment and communication, which specifications such as BPEL4WS proved service composition and process enactment.

Major components of the proposed system are described as follows:

- *User Interface Layer* is responsible for presenting to the user with the system functionalities and allows content management through a provided GUI set can interact with business process and service choreography. User interface application can not only be web based, but also other SOAP client. From the user interface, the customers place the tasks, register their profiles, select algorithms, and setup the mining processes.
- *Business process Layer* is responsible for coordinating the workflow process, which includes the process of building the workflows, composing external web services and monitoring their execution. When a workflow process is triggered, the business process layer selects a related workflow definition from process definition repository. According to the defined relations and properties, the orchestration engine within the business process layer finds an execution task and then sends it to the service choreography for task allocation.
- *Service Choreography Layer* is a multi-agent platform that provides a bridge between the implementations and the consuming applications, creating a logical view of sets of services which are available for use. In this case, we also call it Distributed Service Bus (DSB). This is the core of the whole framework, integrating a set of agents, each one with special characteristics and behavior. They act as controllers and administrators for all applications, services, communication and performance of reasoning and decision-making.
- *Service Application Layer* encapsulates the data manipulation methods and data mining algorithms within appropriate interfaces and advertises them as one or more Web

services. It processes data sets and produces new data sets as output. As such, administrators easily add new services as long as they have the same service interface. Similarly, a user can use any service for their task, as long as the services allow it. Extending the services is simple: administrators can add a new database by instantiating a DS for it; they can increase computing power by either adding more nodes to existing clusters or by adding a new cluster as an independent mining server; or they can add new algorithms and metaphors just by advertising them.

Communication protocol is also a very important part in this architecture, which allows applications and services to communicate directly with the agents' platform. The protocol is based on SOAP specification to capture all messages between the platform and the services and application. All external communications follow the same protocol, while the communication among agents in the platform follows the FIPA Agent Communication Language (ACL) specification. Applications can make use of agents platforms to communicate directly (using FIPA ACL specification) with the agents in service choreography, so while the communication protocol is not needed in all instances, it is absolutely required for all services.

3.3. Distributed service bus

At the core of the ABSOA is the need to be able to manage services as first order deliverables. It is the service that we have constantly emphasized that is the key to communication between the provider and consumer. So we propose an agent-based approach for service bus, which called Distributed Service Bus (DSB). DSB is a logical view of the available and used services for a particular business domain. In this structure, there are two key components: the first one is the service register for hosting and managing services, the other one is the mediation capability that ensures that messages are delivered properly.

The DSB is built from WADE (Workflows and Agents Development Environment). WADE is an extension of JADE (Java Agent Development Environment) that eases distributed multi-agent system development. The platform comprises of multiple agent containers. The main container hosts a persistent main service agent, known as DAP Service Agent (DSA), as it is the starting point of autonomous process. The activity Runner (AR) containers perform data mining task as specified in the data mining model. Distributed AR containers can be pre-configured or on-spot configured to join a platform. DSB maintains a platform resource database which contains system resource, such as data sets, data mining models, etc.

The following will be the detail of the three specific agents in DAP:

- **DAP Service Agent (DSA)** – the agent as the starting point of the autonomous activity. The agent receives forwarded message from WSIG regarding the requested service; then it determines for an appropriate action to take. On request for instance execution, the agent verifies the request for the particular instance and forward to a Case Mediator agent.
- **Case Mediator Agent (CMA)** – the agent receives the information about an instance, it mediates the resources whether it is local or remote accessible, then it spawns a set of Activity Runner agents to perform the actual model execution. The CMA monitors the status of the execution, collects results, and notifies DSA.
- **Activity Runner Agent (ARA)** – the agent runs an activity, which is a component of a data mining model. ARA is the actual worker that performs the action defined in activity assigned by a CMA. After it finishes, it returns the result to the CMA.

Aside from DAP specific agents, WADE has added a few new agents that enhance JADE functionalities. In this work, we often refer to the two agents:

- **Configuration Agent (CFA)** – the agent is responsible for system configuration, such as add/remove agents and containers.
- **Control Agent (CA)** – the agent is responsible for system control, such as system monitoring, system stage alteration.

The agent communication and collaboration model is another critical issue which can significantly affect the system performance as a whole. JADE provides a communication mechanism that makes use of the FIPA ACL performatives. However, like Chaimontree, Atkinson and Coenen (2011) mentioned in their article that the FIPA ACL defines a series of stand format which use 22 standard performatives for agent to communication with each other. But because our architecture focuses on ensemble learning, the default 22 performatives are too broad for our needs. Therefore we have to define our own set agent communication language to engage the collaboration in the framework.

In this section, we describe the collaboration scheme of the three agents.

1. DSB Service Agent (DSA)
   a. Wait for "runInstance" request message from business process.
   b. Act as a monitor agent – maintain current status of all CMA and activities
   c. Search for an appropriate CMA to run the request instance – this is done by using status information in (b)
2. Case Mediator Agent (CMA)
   a. Wait for ACL message "run-instance" request message from DSA.
   b. Get instance configuration of the requested instance from system database.
   c. Maintain system objects in memory i.e. Model, Data Table, Activities
   d. Plan for model execution – this is additional to local execution. CMA spawns ARA to execute an activity. This can be done in the distributed fashion.
   e. Spawn an ARA to run an activity

f. Receive result Data Table of the activity from an ARA and update the model's datasets
g. Update DSA about the status of the running
h. Update prior to model started – number of activities, analysis of instance requirement
i. Update during the model run – status of each activity, and
j. Update after the model finished – outcome of the model run, statistical data of the run, size of data, time taken, etc.

3. Activity Runner Agent (ARA)
   a. Spawned by CMA.
   b. runs activity
   c. Submit result dataset to the respective CMA.

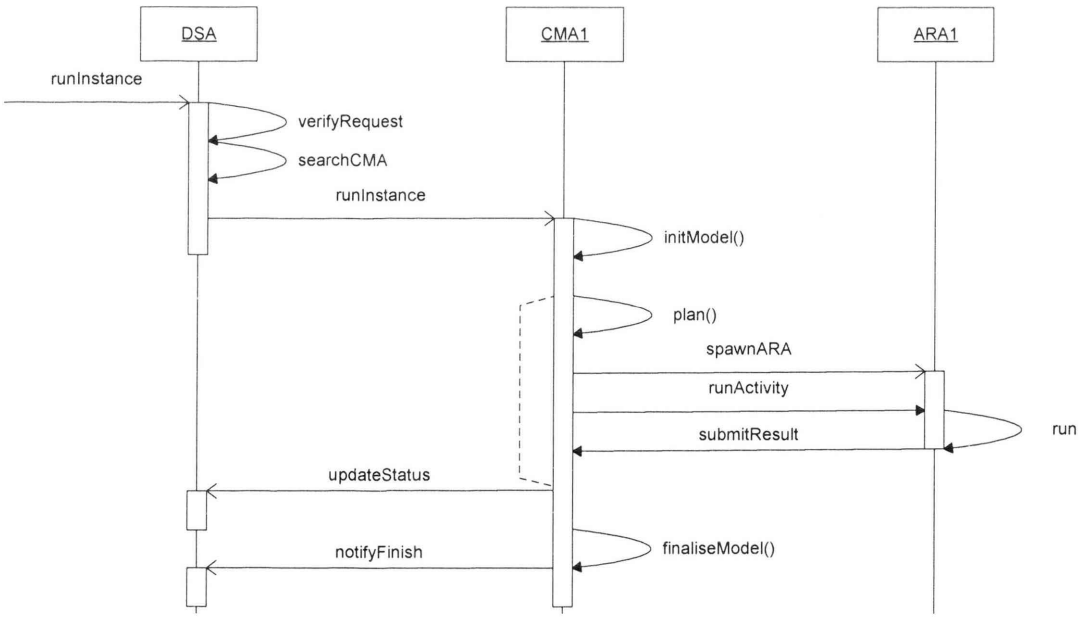

Fig.3.2. Sequence diagram for agent message-passing

In Fig 3.2, each function is described as in Table 1.

Function	**Input/Output**	**Description**
DSA.runInstance	IN: instanceID, userID	Run an instance of DMM. DSA replies immediately that the request is received, but not the result of the run because the run is a asynchronous process.
DSA.verifyRequest	IN: instanceID, userID	Verify the request whether the user is authorised to run the instance and the instance is ready to be run.
DSA.searchCMA	IN: instanceID OUT: CMA-AID	Examine the requirement of the instance and search for an appropriate CMA Agent ID (AID) to run the instance.
CMA.runInstance	IN: instanceID	Accept the instance ID to run
CMA.initModel	IN: instanceID OUT: DDM	Create a DMM from the given instance ID.
CMA.plan	-	A sophisticated method to determine several factors to spawn appropriate ARA to run a particular activity.
ARA.spawnARA	IN: activity OUT: ARA-AID	Spawn a new ARA provided with an activity for it to run.
ARA.runActivity	-	Receive run notification
ARA.run	OUT: activityOutput	Run the activity and capture the

		output of the activity. The output can be datasets, values, and trace messages.
CMA.submitResult	IN: status, activityOutput	Submit the run result to the CMA to collect into the DMM.
DSA.updateStatus	IN: status	Update the DSA with the current status of the DMM.
CMA.finaliseModel		Finalise the DMM, this includes saving files, closing streams, releasing objects, etc.
DSA.notifyFinish	IN: instanceID	Notify the completion of the instance.

Table 1. Function description of the system

3.4. Pluggable Mechanism

Lastly, the model execution engine is a complex Java class that loads configuration XML file and construct a series of activities and execute them. Each activity has a designated method which is invoked by the engine. Furthermore, each activity can be listened with Listener interface, which perfectly complies with the MVC model. Whenever an event occurred in an activity will notify every attached listener, therefore immediate update is achieved. This functionality is prepared for later agent integration, when agents are required to observe ongoing activities.

Now, let us come to the main concern with detailed issues.

(a) To extract information from an unseen algorithm, the algorithm, as a Java class, must extend the abstract class `Algorithm`, as it is to ensure the compatibility of the interface.

(b) Integrating the algorithm class is done through a dual step, the class loader and annotation processing. The algorithm class is loaded dynamically using a customised class loader (this class loader was done in the earlier phase). The algorithm is then configured with a set of parameters defined in the XML configuration file. However, the system must learn the parameter requirements from the algorithm, and to do so, the algorithm is annotated with a set of custom annotations.

Code annotation is a meta-programming technique which is available in most major programming languages, Java, C/C++. Meta-programming allows other programming code to observe information about the code without having to instantiate the object into the computer memory. The system can benefit from this by not risking to handling any unintentional run-time exception from unseen algorithms.

So far, we have created 4 annotations: `AlgorithmInfo`, `RequiredDataSetFields`, `DataSetField`, `FormalParameter`; which can annotate algorithm codes. The annotation is a meta-programming technique where the annotated objects do not need to be instantiated, hence non memory consumption and non run-time class casting error. It is a neat way to observe the class information while assure the system robustness.

```
@AlgorithmInfo(
        id = "12345698756433567945320 5",
        name = "SPAM",
        description = "A simple sequential pattern mining ",
        author = "Mr.Developer"
)
@InputDataSetInfo(
    dataMappingInfo = {
        @DataSetInfo(
                datasetName="dpriceData.1",
                fields = {

    @DataSetField(name="date",type=DataSetField.STRING),

    @DataSetField(name="price",type=DataSetField.DECIMAL)
                }
        )
    }
)
@FormalParameter
protected float minimumSupport;
@FormalParameter
protected int maxSequentialLength;
```

Table 2. Annotation

The code fragment from `SpamAlgorithm.java` in `spam` package shows how annotations are used.

- `AlgorithmInfo` denotes general information about the algorithm.
- `RequiredDataSetFields` defines data fields required by the algorithm. This is to facilitate data field mapping with external data sets
- `DataSetInfo` is a detail description of `fields`.
- `FormalParameter` denotes input or output parameters of the algorithm.

(c) Lastly, running the algorithm is achieved by providing a predefined activity, `RunAlgorithm`, to run any loaded algorithm class. The activity `RunAlgorithm` configures the algorithm with parameters provided in the XML configuration file.

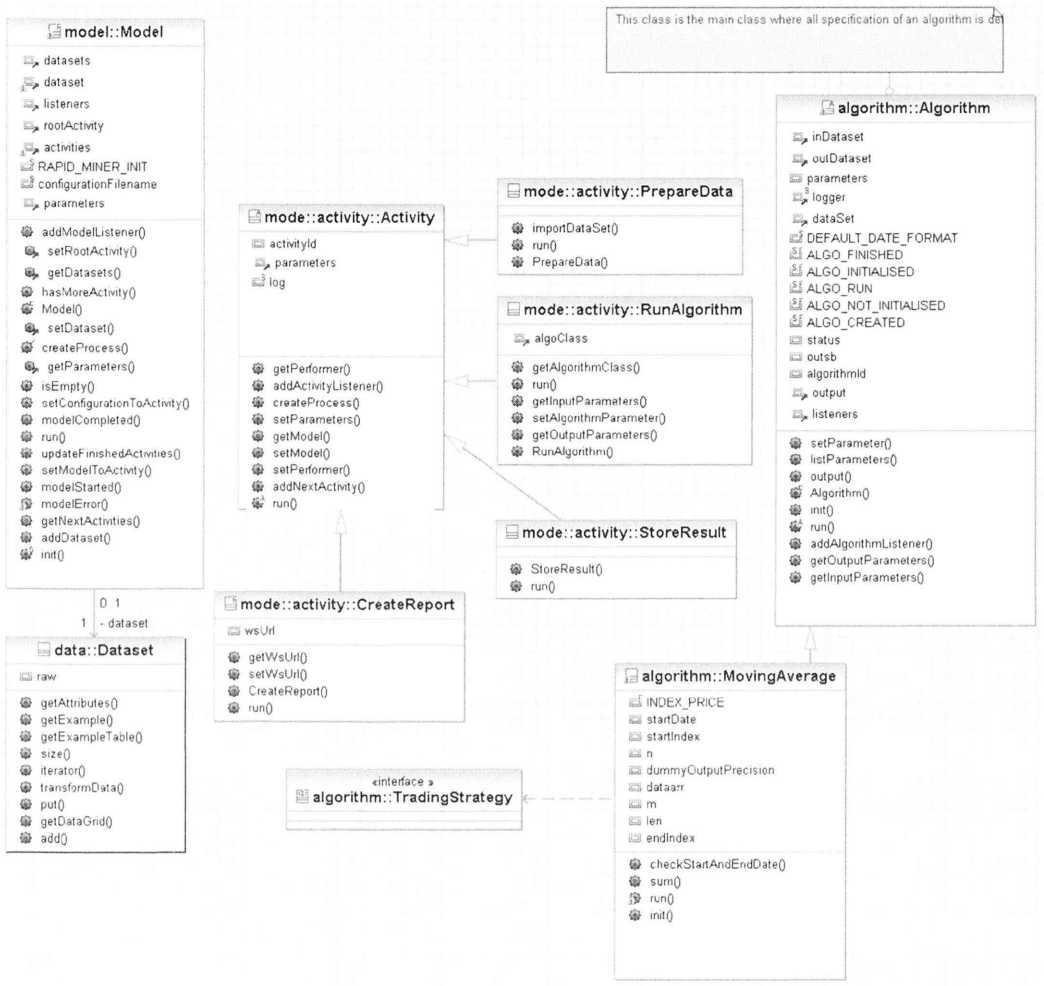

Figure 3.3. Class diagram of service platform

Figure 3.3 shows the overview of the service platform architecture. We try to simplify the algorithm development process by providing only single point of entry for algorithm developers. In general, the algorithm developer only needs to implement those required methods in the class `Algorithm` and properly annotate the code. Model construction and mapping are the task of model developer. In addition, the algorithm implements related interface for additional capability, such as, `TradingStrategyType` denotes that the

algorithm is a kind of trading strategy. Similarly, other interfaces are provided for particular data mining purposes, such as `ClusteringType`, `ClassificationType`.

3.5. Conclusion

In this framework, we propose a new and easier method, where the functionalities of system are not integrated into the structure of the agents; rather they are modeled as distributed services and applications which are invoked by the agents acting as controllers and coordinators. Therefore, techniques developed in this framework can improve the interoperability between different modules, distribution of resources, and independency of programming languages. Moreover, System administrators can add new resources (data sets, servers and algorithm) by simply advertising them as web service.

Chapter 4

Agent-Based Business Process Management

In the previous chapter we have shown that the Distributed Service Bus (DSB) component provides a service platform for hosting and managing services and mediation services that ensure that messages are delivered properly. In this chapter we are going to present the agent-based BPEL engine aimed at building workflows, composing the external Web services and monitor their execution. Specifically, we are going to present two alternative automatic procedures of risk mining, namely predefined workflow and dynamic workflow, and empirically evaluate them.

4.1. Introduction

In recent years, a lot of researchers focus their efforts on the usage of workflows in distributed systems and the use of agent technique for the management of workflows. One of the important contributions of our work in this chapter is firstly the use of agents as the support of all the services which involved in the execution of a business process, i.e. the distribution of workflow tasks, the control of their execution and finally the re-allocation of tasks in case of failure of some service components. Secondly, the integration of the agent technique with those techniques we consider crucial for accomplishing strategic business objectives.

It is obviously that continuous data mining and flexible adaption of the data mining process play crucial roles in the context of modern risk analysis and management solution. Therefore, in the system architectural design we provide an orchestration model (BPEL engine) to support the development of composite services, which can coordinate the execution of the services required in a predefined pattern or workflow at runtime. Our focus is to develop small modules with minimal dependencies that could be assembled to construct a full featured Business Processes Management System (BPMS). The goal of orchestration engine is to support users in the process of building the workflows, composing external web services and monitoring their execution.

Web Service Business Process Execution Language (WS-BPEL) is an OASIS standard executable language which defines an XML-based script for specifying actions within business processes with web services. A WS-BPEL workflow is a structured XML document composed of three main parts: (i) the definition of the process' attributes, (ii) the definition of the execution context and (iii) the activities to be executed. The main drawbacks of the existing software tools for WS-BPEL workflow design, specification and enactment are they enact the workflow in a centralized manner and furthermore they are not

able to dynamically exploit new web services in case of unpredictable event. In this work, the framework of orchestration engine we have realized is able to execute a BPEL process with distributed and dynamically composite web services.

4.2. Business Process Engine

The principal objective in the development of BPEL engine was to create a reliable, compact, and embeddable component capable of managing the execution of long-running business processes defined using the BPEL process description language. The BPEL process execution is constituted of three phases:

(i) Interpretation of the BPEL document

(ii) Creation of an internal process model, aiming at describing in a consistent way the business process characteristics and at the same time to make easy and efficient the execution of the business process itself,

(iii) Preparation of the execution context and distributed execution, possibly providing for the exploitation of new web services.

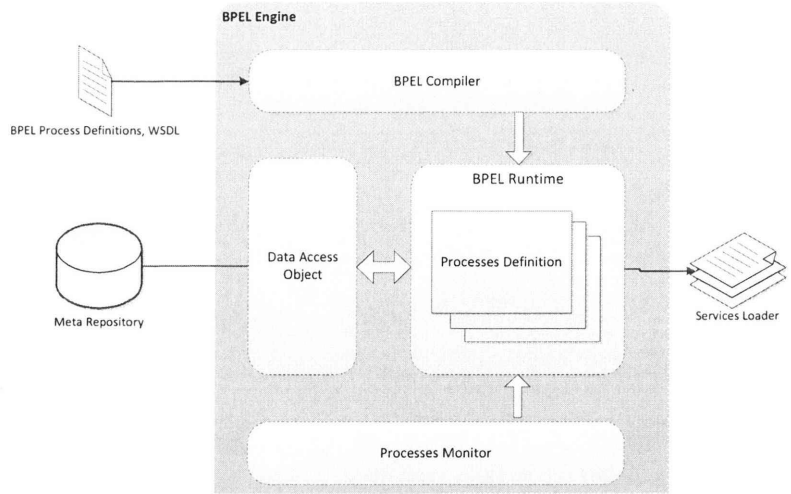

Fig.4.1. BPEL Engine

The key components of the BPEL Engine architecture include the BPEL Compiler, BPEL Runtime, Data Access Object, Log Manager and Processes Monitor. The relationships between these components can be summarized as: "The compiler converts BPEL documents into a form executable by the run-time, which executes them in a reliable fashion by relying on a persistent store accessible via the DAOs; the run-time executes in the context of web service specification which connects the engine to the Distributed Service Bus (DSB); The whole process should be monitored and controlled by log manager and process monitor.

BPEL Compiler: The BPEL compiler is responsible for the conversion of the source BPEL artifacts (i.e. BPEL process documents, WSDLs, and schemas) into a compiled representation suitable for execution. The compiled BPEL representation generated by the compiler is an object model similar in structure to the underlying BPEL process documents. However, the compiled representation has resolved the various named references present in the BPEL (such as variable names), internalized the required WSDL and type information, and generated various constructs (e.g. default compensation handlers). The compiled representation (typically a file with the .cbp extension) is the sole artifact required by the BPEL runtime.

BPEL Runtime: The BPEL Runtime is found in the bpel-runtime module and provides for the execution of compiled BPEL processes. The runtime handles the dirty work of process execution by providing implementations of the various BPEL constructs. The runtime also implements the logic necessary to determine when a new instance should be created, and to which instance an incoming message should be delivered. Finally, the runtime implements the Process Management API that is used by user tooling to interact with the engine.

To achieve reliable execution of processes in unreliable environments, the runtime relies on Data Access objects (DAOs) to provide persistence facilitates. The implementation of these

DAOs can be customized, but is typically provided by a transactional relational database. The DAOs are described in more detail in the next section.

The runtime implementation of BPEL constructs at the instance level is via processes executor. It provides an application-level concurrency mechanism (i.e. it does not rely on threads) and a transparent mechanism for interrupting execution and persisting execution state.

Data Access Object: Data Access Objects mediate the interaction between the BPEL Engine Runtime and underlying data store. Typically the data store is a JDBC relational database: in this case, the DAOs is implemented via the open JPA data access library. It is possible to create custom DAO implementations that employ a mechanism other that JDBC to achieve persistence, although no such implementation is provided. The BPEL Engine Runtime requires DAO objects to deal with the following persistence issues:

- active instance – keeping track of which instances have been created
- message routing – which instance is waiting for which message
- variables – the values of the BPEL variables for each instance
- partner links –the values of the BPEL partner links for each instance
- process execution state – the serialized state of the process executor "persistent virtual machine"

Processes Monitor: XML persistence for business process is the use of an XML data management system for business process analysis, which can be important in real-world business scenarios.

- Number of running/completed instances of process per week
- Number of errors for specific endpoints or partner links for a process
- Execution time for a specific process over a period of time

- Size of data exchanged per endpoint
- Availability or uptime of partner link (uptime)
- Average wait time for asynchronous endpoints

4.3. Integrating workflow into risk mining

To accomplish the complex business requirements, we define two alternative automatic procedures in this orchestration engine: predefined workflow and dynamic workflow. The first one is standard and common templates which extracted from previous experiments and real scenarios. Therefore the predefined workflows have been proved to be the summary of experience which can be applied in some general occasions. In these cases the duty of orchestration engine is to help the users to decompose the process into several sub tasks and then allocate them to service choreography to select the appropriate web services. All of the agents in the system have a common knowledge background so that they are able to select a matching service.

On the other hand, users have to create new workflows (or update from predefined workflow) according to the new business requirements which may include some atomic services available in the system. In this case, the duty of the orchestration engine is to present the users the web services that can be composed and possibly inform them when the realized workflow does not satisfy the composition rules. When a complete workflow is realized, it can delegate the workflow execution to service choreography and possibly replace those web services that are failed or no more available or cannot satisfy the execution constraints.

4.3.1. Risk mining model

A generic workflow composes of activities and transitions. Each activity receives inputs and produces outputs. Outputs from the prior task can be used as inputs for the subsequent task only if the structures of inputs and outputs are compatible. Transition is triggered by events, usually task completion or conditions.

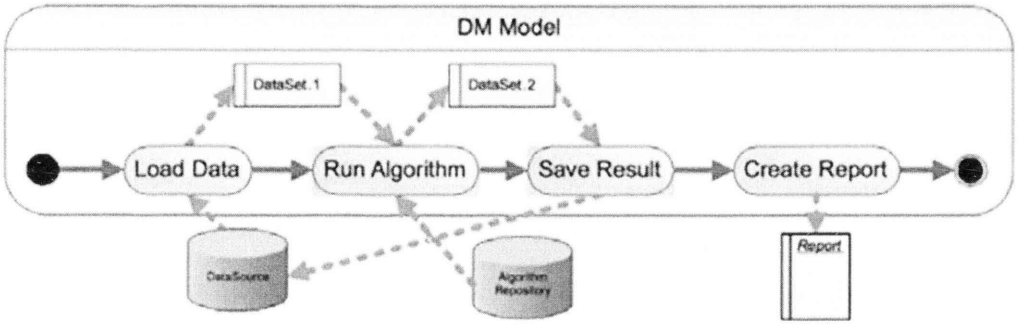

Fig.4.2. A simple data mining model

Data mining model (DMM) in the BPEL engine is represented as a workflow document which is based on a specific XML schema. DMMs are carried out by DAP. Figure 4.2 shows a simple data mining model defines a series of activities: LoadData, RunAlgorithm, SaveResult, and CreateReport. Each activity may produce any arbitrary data sets and store them in the model memory for other activities to consume. Sharing inputs and outputs does not occur in a single memory unit, due to distributed environment of the system. Inputs and outputs are determined to be transferred from one agent to another agent as needed. Since each activity is run by an ARA; ARA stays alive until there is no dependency to it.

4.3.2. Descriptive workflow evaluation

The schema of DMM is based on an XML Schema Definition (XSD); figure 4.3 shows a simplified schema. Model is the data mining model itself which contains activities and data

sets. Activity is an abstract class which defines generic structure of an activity. Each activity receives input and output dataset; while parameter specification is defined in the actual DMM contents. LoadData, SaveResult, RunAlgorithm, and CreateReport are sub classes extending from Activity. Each concrete activity defines its required parameters that are configured specifically for each model instance (describe in the later section). Figure 4.4 shows a sample content of LoadData in descriptive format. A system variable, i.e., ${dataset dir} is to be replaced with actual values of each agent environment. In this example, ARA loads a data set from a CSV file located at ${dataset dir}/dpriceData.csv into a dataset dpriceData.1.

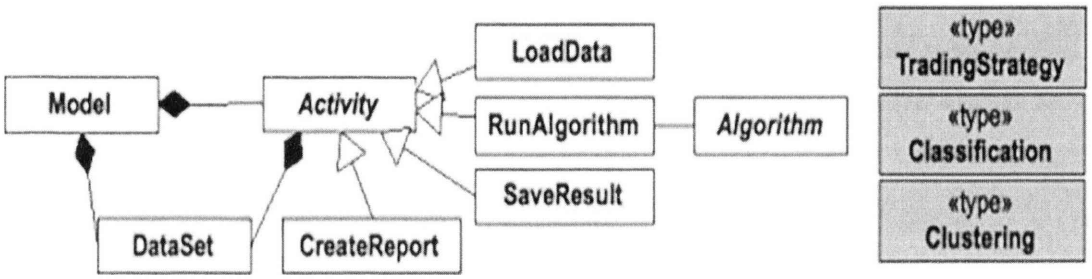

Fig.4.3. simplified DMM schema

```
<activity id="1" type="LoadData" name="LD1" nextActivityIds="2">
        <inputDataSources>
                <dataSource    id="1111"    name="dpriceData.1"    type="CSV"
                path="${dataset_dir}/dpriceData.csv">
                        <fields>...</fields>
                </dataSource>
        </inputDataSources>
</activity>
```

Fig. 4.5 A sample content of LoadData in descriptive format.

Another example, risk mining process allows users to upload new algorithms to the system; the algorithms are distinguished by namespaces. An algorithm may contain several related classes; these classes are packed into a file, in this case Java Archive (JAR) file. A descriptive algorithm name only defines fully qualified name of the algorithm including namespaces to be used in the data mining model, e.g., ianalyst.algorithm.Spam. The evaluation process for the algorithm name will add additional information, that are (i) the file path where the algorithm JAR file is located and (ii) the agent name is assigned to run this algorithm depending on the algorithm requirements, such as language and libraries. For example,

<algorithm name="ianalyst.algorithm.Spam"/>

is evaluated to

<algorithm name="ianalyst.algorithm.Spam" jar="/contents/algorithm/uploads/jar001.jar aid="ara001">

Similar to the algorithms, a descriptive data source only denotes the name with fully qualified name, e.g., stock/marketA/dpriceData2009. Other additional configurations can also be supplied, for instance, localAccessOnly indicates whether the activity should be executed on the remote platform where the database is accessible, hence the same network. For example,

<dataset name="stock/marketA/dpriceData2009" localAccessOnly="true"/>

is evaluated to

```
<dataset name="stock/marketA/dpriceData2009"
uri="jdbc:mysql:http://analyst.it.uts.edu.au:3306/ianalyst" aid="ara001">
   <fields>...</fields>
</dataset>
```

In case of view data sets, the data set that is generated from data processing instructions, e.g. SQL statements, the executable form will be as following:

```
<dataset name="stock/marketA/dpriceData2009"
uri="jdbc:mysql:http://analyst.it.uts.edu.au:3306/ianalyst"
aid="ara001" dpi="true">
        <dpiSequence>
                <dpi type="sql" key="proc01" value="SELECT * FROM stock
                WHERE date BETWEEN '2009-01-01' AND '2009-12-31'"/>
        </dpiSequence>
        <fields>...</fields>
</dataset>
```

4.3.3. Example

In this section, we demonstrate how the concept is implemented. From figure 4.6, system variables, e.g., ${project classpath}, ${dataset dir}, ${result dir}, ${report dir}, are to be replaced with local values of each agent environment. In this example, we load a data from a CSV file located at ${dataset dir}/dpriceData.csv into a dataset dpriceData.1. The dataset is visible throughout the model and of course shared with other activities. RA1 requires dpriveData.1 hence CMA sends the dataset to the RA1 ARA. Result of applying MovingAverage algorithm is the transferred into the dataset result. The last activity is to save the dataset into a file. In this example, the model exports the dataset to ${result dir}/resultData.csv.

```
<instance id="100679" name="Instance1">
        <dataSets>
                <dataSet id="12345" name="dpriceData.1">
                        <fields>
                                <field name="trading_date" type="xs:string" />
                                <field name="sum_daily_price" type="xs:decimal" />
                        </fields>
                </dataSet>
                <dataSet id="54321" name="result"><fields>...</fields></dataSet>
                </dataSets>
<activities first_activity_id="1">
        <activity id="1" type="LoadData" name="LD1" nextActivityIds="2">
                <inputDataSources>
                        <dataSource id="1111" name="dpriceData.1" type="CSV"
                        path="${dataset_dir}/dpriceData.csv">
                                <fields>...</fields>
                        </dataSource>
                </inputDataSources>
                <outputDataSets><dataSet id="12345" /></outputDataSets>
        </activity>
        <activity id="2" type="RunAlgorithm" name="RA1"
                previousActivityIds="1" nextActivityIds="3">
                <algorithm id="2222" name="MovingAverage"
                path="${project_classpath}"
                className="ianalyst.algorithm.impl.MovingAverage">
                        <parameters>
                        <inputParam key="m" value="5" type="xs:int" />
```

```
            ...
            </parameters>
        </algorithm>
        <inputDataSets>
            <dataSet id="12345">
                <fieldMapping>
                    <field localName="date" orgName="trading_date"
                    />
                    <field localName="price"
                    orgName="sum_daily_price" />
                </fieldMapping>
            </dataSet>
        </inputDataSets>
        <outputDataSets><dataSet id="54321" /></outputDataSets>
</activity>
<activity id="3" type="SaveData" name="SD1"
previousActivityIds="2" nextActivityIds="4">
        <inputDataSets><dataSet id="54321" /></inputDataSets>
        <outputDataSources>
            <dataSource id="3331" type="CSV"
            path="${result_dir}/resultData.csv">
            <fields>...</fields>
            </dataSource>
        </outputDataSources>
</activity>
```

```
        <activity id="4" name="CreateReport1"
        previousActivityIds="333" type="CreateReport">
                <dataFile path="${result_dir}/tt"/>
                <reportTemplate id="4444" name="CR1"
                path="${reporttemplate_dir}/ReportTemplate/RT01.zip">
                <fieldMapping>
                <fields>...</fields>
                </fieldMapping>
                </reportTemplate>
                <report path="${report_dir}/Report/yy" type="PDF"/>
        </activity>
        </activities>
</instance>
```

Fig. 4.6. Content of a DMM

The DMM is evaluated to an executable workflow as described in section 4.3. The executable workflow which has four activities is and monitored by the CMA. The CMA requests target platforms to spawn respective ARA. The CMA sends activity1 to the respective ARA1, in which the ARA must be on the same site where the data is available. ARA1 loads data from into the memory, maintains in its memory, and notifies the CMA of the completion. The CMA notifies ARA2 to run activity2. The CMA sends the algorithm content to ARA2. ARA2 acquires for input dataset which was produced by activity1. CMA notifies ARA1 to send the data to ARA2. In the planning stage, ARA1 and ARA2 are placed in the same or neighbor network to minimize the communication cost. ARA2 applies the algorithm on the data, then produces a result set, and notifies CMA for the completion. CMA notifies ARA3 to starts activity3. ARA3 fetches required data and saves to its local directory which is also visible to ARA4. After the completion of ARA3, ARA4

receives notification from the CMA and starts activity4. ARA4 merges the output data and a report temple into a report document, and places on the designated directory. ARA4 notifies the completion to the CMA. The CMA receives all completion confirmations and finally notifies the DSA for the workflow completion.

4.3.4. Evaluation

The integration method augments the existing ADDM architecture by introducing additional responsibility to the agents to perform according to the assigned activity. The risk mining models are improved to be used as workflows, so the system only maintains only one document, the descriptive one. The workflow is used to help planning of agents. With descriptively provided workflow, the responsible agent (CMA) can insert more details to the workflow for the execution. The benefit is that the system does not need to disclose internal information for the descriptive workflow producer, e.g., third-party front-end system, therefore data and system configuration are hidden. Model evaluation determines the resource utilization. Planning may concerns the cost of communication and computation time. Planning may result in a plan to execute all activities locally relatively to the CMA if the size of data is too excessive to transfer across the networks. A group of consecutive activities may be executed on the same containers as to minimize data transmission cost. In terms of robustness, exception handling is achieved through agent negotiation. Agents can be programmed to handle pre-deterministic exception internally, before acquiring for others assistance. The system benefits from less interruption from exceptions that can be handled. Non-deterministic exceptions are forwarded to the CMA and the CMA decides to start the activity elsewhere.

Chapter 5

Agent-Based Ensemble Learning Strategies

In this chapter we are going to introduce the collaboration strategies of a group of agents that joins together in order to improve model accuracy and robustness by using ensemble learning methods. Each ensemble agent individually gathers the evidence about model evaluation, and then ensemble learning method like bagging or boosting is used to obtain a global prediction from the individually gathered evidence. Moreover, we will present the ensemble learning methods to deal with imbalanced data and empirically evaluate its performance.

5.1. Introduction

Ensemble data mining method is a machine learning paradigm where multiple learners are trained to solve the same problem, which focus on creating ensembles in order to improve the classification accuracy of a given learning method. Although single model has good performance in classification, it is sensitive to samples and parameters setting, i.e. each single model has some bias. The effective way to reduce the bias is the ensemble method. In this chapter, our approach is to coordinate a set of agents (that can be considered as the individual classifiers) in order to take benefit from the ensemble effect.

Also, we proposed the concept of committee which is a group of agents that collaborate with each other in order to improve classification accuracy by using ensemble methods. It is the organizational form of an "ensemble of agents" from the point of view of multi-agent systems, defined to study the ensemble effect in multi-agent systems. Each member of the committee should be as competent as possible, but the members should be complementary to one another. Each individual agent has individually collected its own case base, and we cannot assume anything about them. If the members are complementary, then when one or a few members make an error, the probability is high that the remaining members can correct this error.

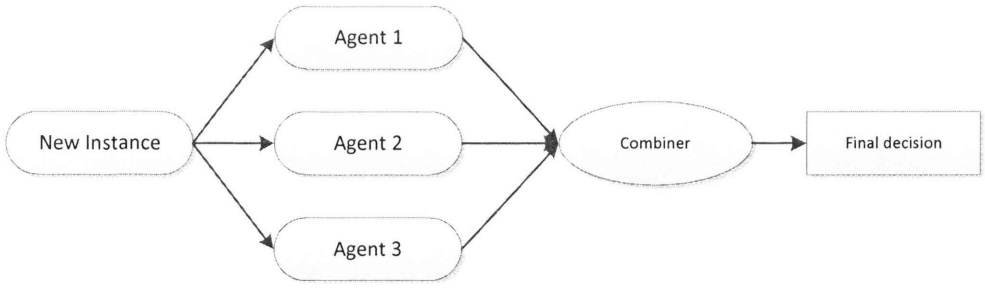

Figure.5.1. Framework of agent-based ensemble methods

Ensembles can provide a critical boost to risk mining where predictive accuracy is more vital than model interpretability. The most common and effective ensemble learning methods are Bagging and Adaboost, which often leads to over fitting in high imbalanced data situation. In this work, we adopt under-sampling based Bagging and cost-sensitive learning with Adaboost to overcome this drawback. Traditional ensemble methods often use instance weights to generate diverse input data and make the algorithms focus on the instances which are still high imbalance. In this way, we can balance the data structure by sampling method or separate the special groups which are difficult to classify by cost sensitive. Moreover, some measures are presented to evaluate the performance of the ensemble methods under different situations. Finally, we will present the committee collaboration strategy which enable a give set of agents work as an ensemble without compromising their autonomy.

5.2. Ensemble methods for risk mining

5.2.1. Under-sampling based bagging for imbalance learning

Bagging based under-sampling is probably the most straightforward way to further exploit the examples in $N \cap \overline{N^1}$. In this method, we independently sample several subsets $N_1, N_2, \ldots N_t$ from N. For each subset N. For each subsetN_i $(1 \leq i \leq T)$, a classifier H_i is trained using N_i and P. All generated classifiers are then combined for the final decision. Many learning algorithms can be employed to generate the individual classifiers.

Algorithm:

1. {Input: A set of minor class examples P, a set of major class examples N, $|P| < |N|$, and T, the number of subsets to be sampled from N.}
2. Randomly sample a subset N_i from N, $|N_i| = |P|$.
3. Train classifier H_i using P and N_i. Build a model $f^i(x)$
4. Combine all $f^i(x)$ into the aggregated model $f^A(x)$

Fig.5.2. Bagging Based Under-Sampling Algorithm

5.2.2. Cost-sensitive learning with adaptive boosting

Cost-sensitive learning targets the imbalanced learning problem by using different cost matrices that describe the costs for misclassifying any particular data example. In this case, we will consider the neural networks, cost-sensitive fitting the neural network can take four forms: first, cost sensitive modifications can be applied to the probabilistic estimate; second, the neural network outputs can be made cost-sensitive; third, cost-sensitive modifications can be applied to the learning rate ŋ; and fourth, the error minimization function can be adapted to account for expected costs.

- Cost-Sensitive classification: integrate cost factors into the testing stage of classification to adaptively modify the probability estimate of the neural network output.
- Adapting the output of the network: the actual outputs of the network are changed and appropriately scaled.

$$o_j' = \frac{\text{CostVector[j]}o_j}{\max_i CostVector[i]}$$

- Adapting the learning rate: increase the learning rate in Neural Network for high cost examples (that is, examples that belong to classes with high expected misclassification

costs). To ensure the convergence of the modified backpropagation procedure, the corrected learning rate should also be accordingly normalized (p is the current training example):

$$\eta(p) = \frac{\eta * \text{CostVector}[\text{class}(p)]}{\max_i CostVector[i]}$$

- Minimization of the misclassification costs: replaces the error-minimizing function by an expected cost minimization function K[i,j], i = desired class, j = actual class:

$$E = \sum_{p \in \text{Examples}} \frac{1}{2} \sum_{i \in \text{Output}} \left((y_i - O_i) * K[\text{class}(p), i]\right)^2$$

Algorithm:

1. {Input: A set of minor class examples P, a set of major class examples N, $|P| < |N|$, and T, the number of subsets to be sampled from N.}
2. i ← 0
3. **repeat**
4. i ← i +1
5. Randomly sample a subset N_i from N, $|N_i| = |P|$.
6. Train classifier H_i using P and N_i. H_i is an AdaBoost ensemble with weak classifiers $h_{i,j}$ and corresponding weights.
7. **until** i = T
8. Generate strong model $f^A(x)$

Fig.5.3. Cost-Sensitive Learning with Adaptive Boosting Aglroithm

5.3. Performance measurement

The conventional evaluation practice of using singular assessment criteria, such as the overall accuracy or error rate, does not provide adequate information in the case of imbalanced learning. Therefore, in this case we intend to introduce some other informative assessment metrics, such as the receiver operating characteristics curves, precision-recall curves, and cost curves, to evaluate the performance in the presence of imbalanced data.

Confusion matrix:

$$TP = True\ Positives;\ FP = False\ Positives;$$

$$FN = False\ Negatives;\ TN = True\ Negatives;$$

$$P_c = \text{Positive Column Counts}$$

$$N_c = \text{Negative Column Counts};$$

Precision is a measure of exactness.

$$\text{Precision} = \frac{\text{TP}}{\text{TP} + \text{FP}}$$

Recall is a measure of completeness.

$$\text{Recall} = \frac{\text{TP}}{\text{TP} + \text{FN}}$$

F-Measure combines precision and recall as a measure of the effectiveness of classification in terms of a ratio of the weighted importance on either recall or precision as determined by the β coefficient set by the user.

$$\text{F} - \text{measure} = \frac{(1 + \beta)^2 * \text{Recall} * \text{Precision}}{\beta^2 * \text{Recall} + \text{Precision}}$$

G-Mean evaluates the degree of inductive bias in term of a radio of positive accuracy and negative accuracy.

$$G - mean = \sqrt{\frac{TP}{TP + FN} * \frac{TN}{TN + FP}}$$

F-Measure and G-Mean are great improvements over accuracy, but they are ineffective in presenting the performance of different classifiers over a range sample distribution, especially in the case of highly skewed data sets. Therefore, we will use more matrixes to address such situations. He and Grarcia (2009) recommend Precision-Recall (PR) Curves and Cost Curves.

The PR curve is defined by plotting precision rate over the recall rate, which is an advantageous technique for performance assessment in the presence of highly skewed data. The cost curve is a cost-sensitive evaluation technique that provides the ability to explicitly express a classifier's performance over varying misclassification costs and class distributions in a visual format.

Chapter 6

Case Studies

This chapter presents the case studies which implements the framework we proposed in this thesis in two real industry projects. The first one is an online banking risk management system (i-Alertor) which we develop for one major Australian bank. i-Alertor integrates various features and data mining models, and aims to consolidate different sources of resources for systematic problem-solving. The second case we will demonstrate is a student risk management system (i-Education) for university. i-Education builds various business processes for different subjects, and aim to identify students who are potentially at risk of failing those subjects, also the key risk factors. These two applications have been proved to be a sophisticated, yet user friendly, risk analysis and management tool. They are modular, interactive, dynamic and globally oriented.

6.1. Online banking risk management system

6.1.1. Background

An online banking fraud detection system is a typical use case of risk mining methodology. We have developed an online banking risk management system: i-Alertor for one of the largest banks in Australia. i-Alertor integrates various features and data mining models, and aims to consolidate different sources of resources for systematic problem-solving. There are two main objectives in this system: the first one is to create a web based risk analysis and management system which can incorporate the three models (Contrast Miner, CNN and Decision Forest) to predict online banking fraud; the second one is to compare the combined risk scoring model with an existing rule-based system (we call it ExpertSystem) used in the major banks in Australia.

Our investigations in the bank show that real-world online banking transaction data sets and most online banking fraud has the following characteristics and challenges: 1) highly imbalanced large data set; 2) real time detection; 3) dynamic fraud behaviour; 4) weak forensic evidence; 5) diverse genuine behaviour patterns.

1) The data set is large and highly imbalanced. According to our study on one Australian bank's online banking data, online banking fraud detection involves a large number of transactions, usually millions. However, the number of daily frauds is usually very small. For instance, there were only 5 frauds among more than 300,000 transactions on one day. These results in the task of detecting very rare fraud dispersed among a massive number of genuine transactions.

Fraud detection needs to be real time. In online banking, the interval between a customer making a payment and the payment being transferred to its destination account is usually very short. To prevent instant money loss, a fraud detection alert should be generated as

quickly as possible. This requires a high level of efficiency in detecting fraud in large and imbalanced data.

2) The fraud behaviour is dynamic. Fraudsters continually advance their techniques to defeat online banking defences. Malware, which accounts for the greater part of online banking fraud, has been reported to have over 55,000 new malicious programs every day. This puts fraud detection in the position of having to defend against an ever-growing set of attacks. This is far beyond the capability of any single fraud detection model, and requires the adaptive capability of models and the possibility of engaging multiple models for leveraging the challenges that cannot be handled by any single model.
3) The forensic evidence for fraud detection is weak. For online banking transactions, it is only possible to know source accounts, destination accounts and dollar value associated with each transaction, but other external information, for example, the purpose of the spending, is not available. Moreover, with the exception of ID theft, most online banking fraud is not caused by the hijack of an online banking system but by attacks on customers' computers. In fraud detection, only the online banking activities recorded in banking systems can be accessed, not the whole compromise process and solid forensic evidence (including labels showing whether a transaction is fraudulent) which could be very useful for understanding nature of the deception. This makes it challenging to identify sophisticated fraud with very limited information.
4) The customer behaviour patterns are diverse. An online banking interface provides a one-stop entry for customers to access most banking services and multiple accounts. In conducting online banking business, every customer may perform very differently for different purposes. This leads to a diversity of genuine customer transactions. In addition, fraudsters simulate genuine customer behaviour and change their behaviour frequently to compete with advances in fraud detection. This makes it difficult to characterize fraud and even more difficult to distinguish it from genuine behaviour.

5) The online banking system is fixed. The online banking process and system of any bank are fixed. Every customer accesses the same banking system and can only use the services in a predefined way. This leads to good references for characterizing common genuine behaviour sequences, and for identifying tiny suspicions in fraudulent online banking.

The above characteristics make it very difficult to detect online banking fraud, especially for the risk mining: extremely imbalanced data, big data, model efficiency in dealing with complex data, dynamic data mining, pattern mining with limited or no labels, and discriminant analysis of data without clear differentiation. In addition, it is very challenging to develop a single model to tackle all of the above aspects, which greatly challenge the existing work in fraud detection.

6.1.2. Solution

We have implemented the online banking risk management system: i-Alertor based on agent-based service-oriented architecture. i-Alertor integrates various features and data mining models, and aims to consolidate different sources of resources for systematic problem-solving.

Fig. 6.1 shows the business process of our proposed online banking fraud detection system, i-Alertor. It consists of four tiers: database, data pre-processing, modelling and alerting, based on the mining process. The database tier locates data resources and connects them to retrieve related data. Relevant online banking fraud detection data is collected from heterogeneous data sources, including Internet banking real time transaction logs, recent and historical transaction data, customer demographic data, and other external sources. The types and format of the data also vary from different sources. To reduce the high volume of source data, we extract relevant information from raw data and transform it into required formats for the models. The relevant data involves real time online banking transactions, customer banking behaviour sequences, historical data, and customer profiles.

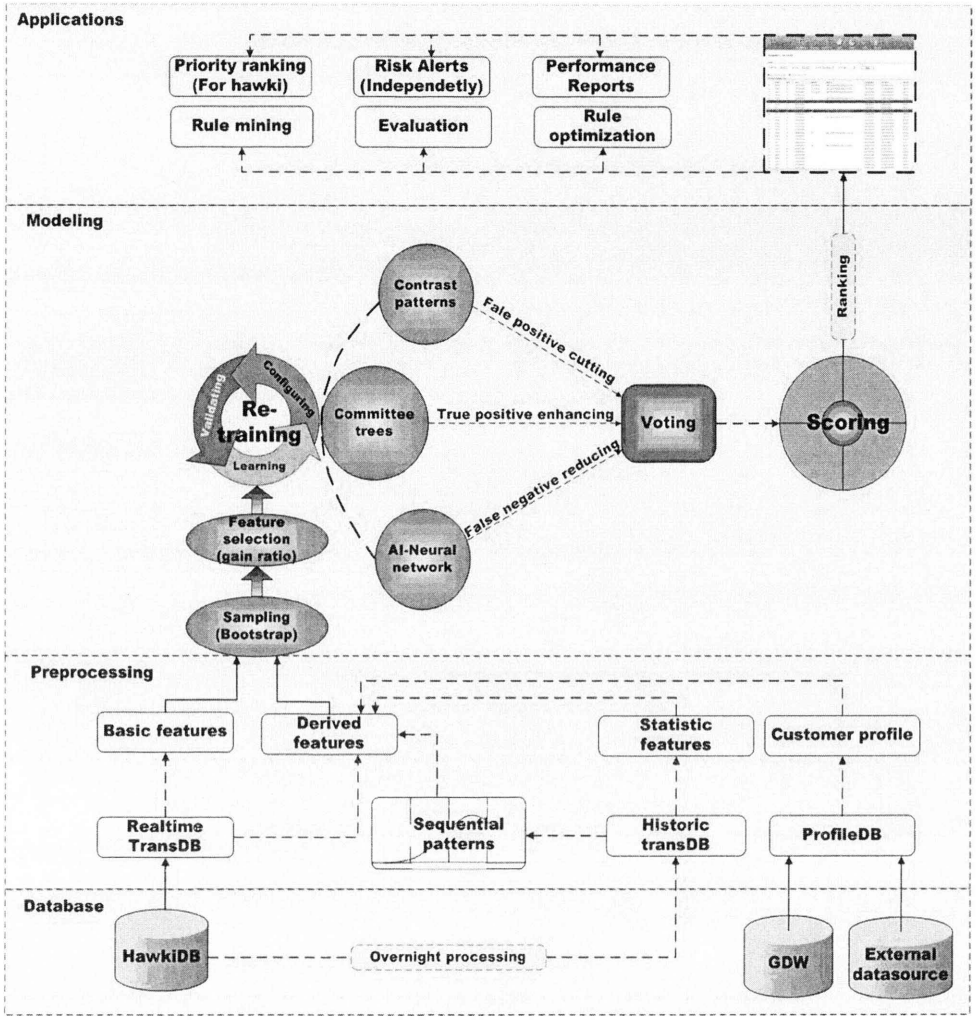

Fig.6.1 business process for online banking fraud detection

The pre-processing tier is in charge of real-time transaction accumulation, historical data maintenance, and preparing the data for model training and prediction. It also includes the function for selecting basic features and deriving features. There are two main tasks in the data pre-processing stage: data sampling and feature selection. As the online banking transaction data is extremely imbalanced, sampling is necessary before applying any data mining models. Because the number of genuine transactions is huge, we use under sampling to reduce the data volume and class distribution imbalance. In our system,

bootstrap sampling is applied to keep the data's statistical distribution. Feature selection is also crucial for the model. Too many features used in the model will highly affect its efficiency, which is very important for real time fraud detection. In our system, we calculate the relevance of a feature to each class in terms of its information gain ratio between two classes, and choose those which have relatively higher information gain ratio. They are the features with the highest discrimination.

The modelling tier provides the generation of models, such as model formation, parameter setting, task scheduling, model retraining, etc. Three data mining methods are adopted in the system:

- contrast pattern mining, which identifies contrast banking behaviour highly associated with online banking fraud;
- cost-sensitive neural network, which emphasizes the higher cost of making an error in the misclassification of a fraud compared to a genuine transaction;
- decision forest, which combines the power of individual decision trees in a weighted manner by cascading-and-sharing for constructing decision tree ensembles;

The alerting tier combines outputs from these three models, which are further combined according to a voting method in terms of their scores for each transaction. Finally, a risk score is generated for each transaction. An alert may be fired if a transaction has a score higher than the given threshold. The combination of three methods reduces the false positive rate and false negative rate, and increases the true positive rate.

All of the resources, data manipulation methods and data mining algorithms in this system are encapsulated within appropriate interfaces and advertises them as one or more Web services. Orchestration engine creates the risk mining processes we describe above, and then maps them to corresponding services by agents.

6.1.3. Outcome and evaluation

i-Alertor (Online Banking Risk Management System) has been implemented and installed in a major Australian bank for instant detection of at-risk online banking transactions. There are six major modules in this system: Dashboard, Workflow Editor, Risk Exploration, Model Management, Rule Management and Report Management. i-Alertor offers a comprehensive and instant solution for risk analytics and management for online banking services, on the basis of the following featured capabilities:

- **Real-time Overall Scoring for At-risk Transactions**
  i-Alertor provides an overall score for each online banking transaction in the real time, and prioritises Top N 'at-risk' transactions for immediate alerting and further investigation. i-Alertor supports one-off scanning and scoring of every transaction, leading to real-time and efficient monitoring of transactions debited and credited for bank accounts.
- **Mixed Data Mining Models for Risk Analytics**
  Our risk management platform embeds mixed data mining models specifically developed for online banking risk analysis. The system provides a flexible and friendly interface to plug into different models.
- **Inbuilt mechanisms for Adaptive Fraud Detection**
  i-Alertor has inbuilt mechanism to make mixed data mining models adaptive to online banking fraud dynamics. This ensures the security and highest possible detection rate of emerging suspicious transactions.
- **Comprehensive Case Investigation**
  i-Alertor provides comprehensive tools for investigating suspicious transactions and conducting case management from a multiple dimensional perspective.
- **Cost-effective Workload Management**
  i-Alertor recommends the optimal combination of fraud and anomaly detection models, the number of real-time alerts, with the best possible performance for detection of at-

risk transactions. This allows a cost-effective management of workload in online banking risk management.

Fig.6.2. Online banking risk management system: i-Alertor

This system incorporates and integrates several data mining models: cost-sensitive neural network, contrast pattern mining, and decision forest. Because different models discover fraud and genuine behaviour patterns from different angles, their combination captures behaviour patterns in a more comprehensive way.

We evaluated our system i-Alertor against the rule-based online banking fraud detection system ExpertSystem. Fig. 6.3 is an overview of distribution of fraud caught by ExpertSystem and our system i-Alertor. Here i-Alertor alerted the top 200 riskiest transactions of each day. The figure shows that there were 49.2% of frauds detected by both systems, while our system can detect an additional 16.8% of fraud. In total, our system has around 7% higher detection rates.

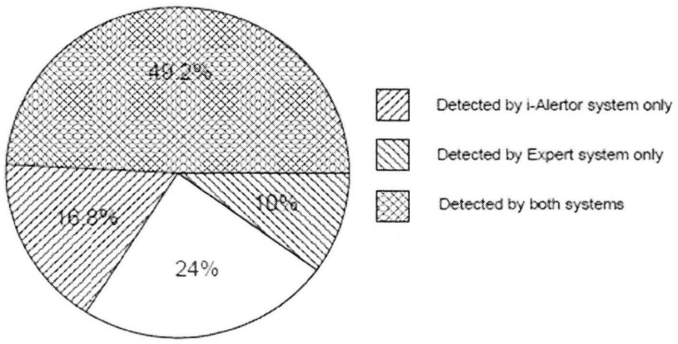

Fig.6.3. Distribution of fraud detected by different systems

Fig. 6.4 shows the detection rate under different alert volumes generated by ExpertSystem, ExpertSystem with ranked alerts, and i-Alertor. As there is no rank among original alerts from ExpertSystem, to obtain top *n* alerts, the alerts were randomly selected from ExpertSystem's daily alerts. For ranked alerts from ExpertSystem, the alerts are selected according to their risk score rank calculated by i-Alertor. From Fig. 6.4, we can see that ExpertSystem with ranked alerts performed much better than original alerts from ExpertSystem. i-Alertor worked even better than ExpertSystem with ranked alerts; at the same alert volume, i-Alertor sometimes even has a 10% higher detection rate. Overall i-Alertor always has better performance than ExpertSystem. It is mainly because of fraud dynamics and i-Alertor can catch high percentage of new fraud which is missed by rules in the ExpertSystem.

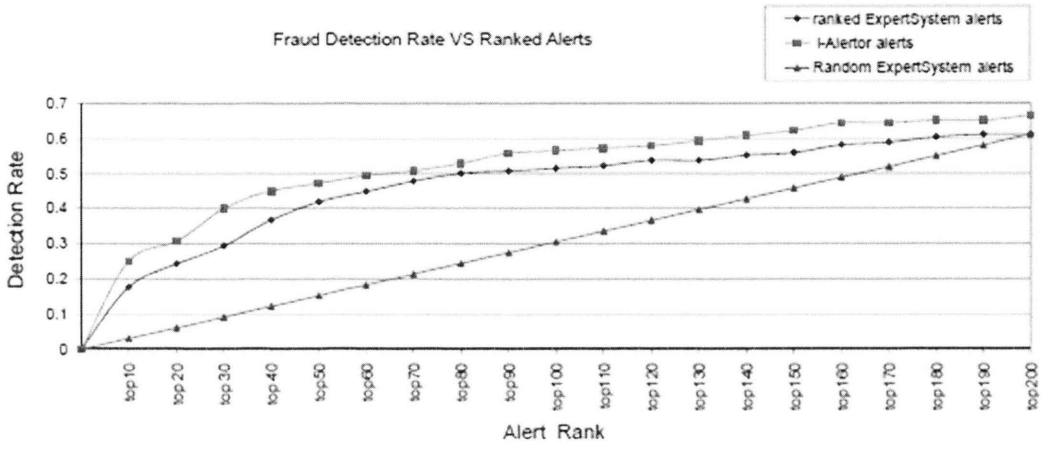

Fig.6.4. Detection rate comparison between i-Alertor and ExpertSystem with ranked alerts

Fig. 6.5 is the evaluation of i-Alertor on missing fraud detection. It shows that i-Alertor can catch 25% of fraud missed by ExpertSystem within 60 alerts. Therefore, if we combine ExpertSystem with i-Alertor, i-Alertor can help ExpertSystem detect more missing fraud by increasing very small alert volume.

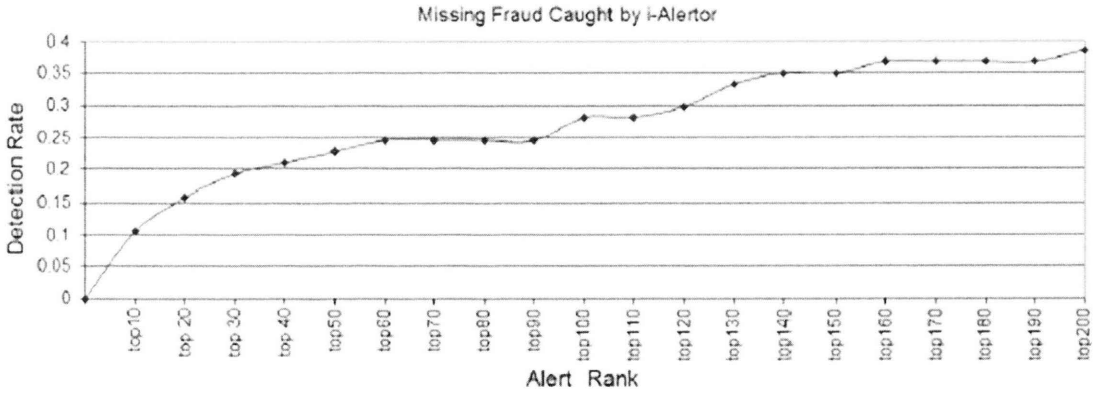

Fig.6.5. Missing fraud detection fate of i-Alertor

Therefore, i-Alertor (Online Banking Risk Management System) has been proved to be a sophisticated, yet user friendly, financial decision support tool. It is modular, interactive, dynamic and globally oriented. As an SOA, i-Alertor is easily extensible due to its use of web service – system administrations add new resources (data sets, servers and algorithms) by simply advertising them to the application servers. This architecture also uses a special distributed runtime environment, which can be used to deal with the distributed computing situation. An important aspect of i-Alertor is its visual metaphors, which help users from other areas of expertise understand the output of data mining algorithms and analysis result.

On the other hand, massive experiments in the bank show that our system and models have a higher detection rate and a lower false positive rate than any single classic data mining model, outperforming the existing rule based system used in all major Australian banks. In addition, our system generates comparably good detection performance on highly imbalanced data sets and the modified contrast pattern mining model is efficient on real time data.

6.2. Student risk management system

6.2.1. Background

The goal of learning analytics is to enable teachers and teaching administrators to better understand student behaviors and the driving forces associated with their performance. It also aims to predict issues and provide best practice services, in order to tailor the best educational opportunities to each student's level of need and ability.

The main objectives of this project may include:

- Provides a dashboard for student service staff to browse and investigate students at risk, as well as a model maintenance interface to arrange data mining models.

- Support risk analysis of –academically at-risk students at different levels: the whole population of students enrolled in the subject, a selected group of students, and an individual student.
- Provides the risk factors and risk patterns specific to each student within a subject; because these personalized risk factors and patterns are shown to be more actionable in relation to taking intervention.
- Provides profiles of each student such as engagement in learning, and the group profile of which the student is associated (i.e. the student's socio-economic and cultural background).
- Set up data connections and pre-processing between the i-Educator and the university data sources so that all relevant sources of data are automatically fed and processed.
- Develop the case management system and performance evaluation system so that the identified risky cases can be intervened in the corresponding business lines, and the performance of intervention is evaluated and refined.
- Expand the analytical models and process to cater for broader decision making purposes, such as student social relation analysis.
- Develop modules to process the privacy of the student information.

6.2.2. Solution

A system, called i-Educator, based on the infrastructure we proposed in this thesis has been implemented for the Outreach Program at the Student Service Unit for the early interventions of students at students at academic risk in the selected subjects. This system provides a solution for model training, student academic performance prediction, risk factor investigation and risky student list export. The system workflow consists of the following two steps.

- Model training. Training data set (historical student data with label) is fed into the system to train models. A range of metrics are used to evaluate the models accuracy. Once the accuracy meets the pre-defined requirements, the models are ready to predict student academic risk.
- Risk prediction. Test data set (target student data without label) is fed into the system. The trained models predict the student performance and associate a risk score to each student. Finally a list of risky student which is arranged in the order of risk score from the highest to the lowest is generated and displayed to users.

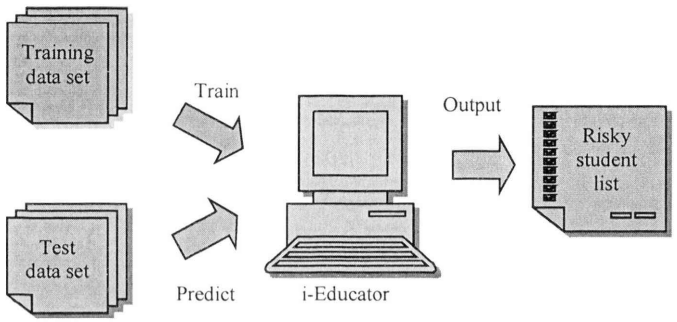

Fig.6.6. Workflow of i-Educator

Data mining models are built to identify key drivers and extract the target group needing help at different levels of priority. The detailed process of the predictive model-based approach is shown in Figure 6.6.

During students' studies in university, different kinds of data records are generated. These data records cover different aspects of students' procedures. For example, their demographic information, study behavior information, sanction information, dormitory information, subject performance information and their records of using different systems which related to their study performance. Basically, there is abundant data from different angles to address students' status and behaviors in study.

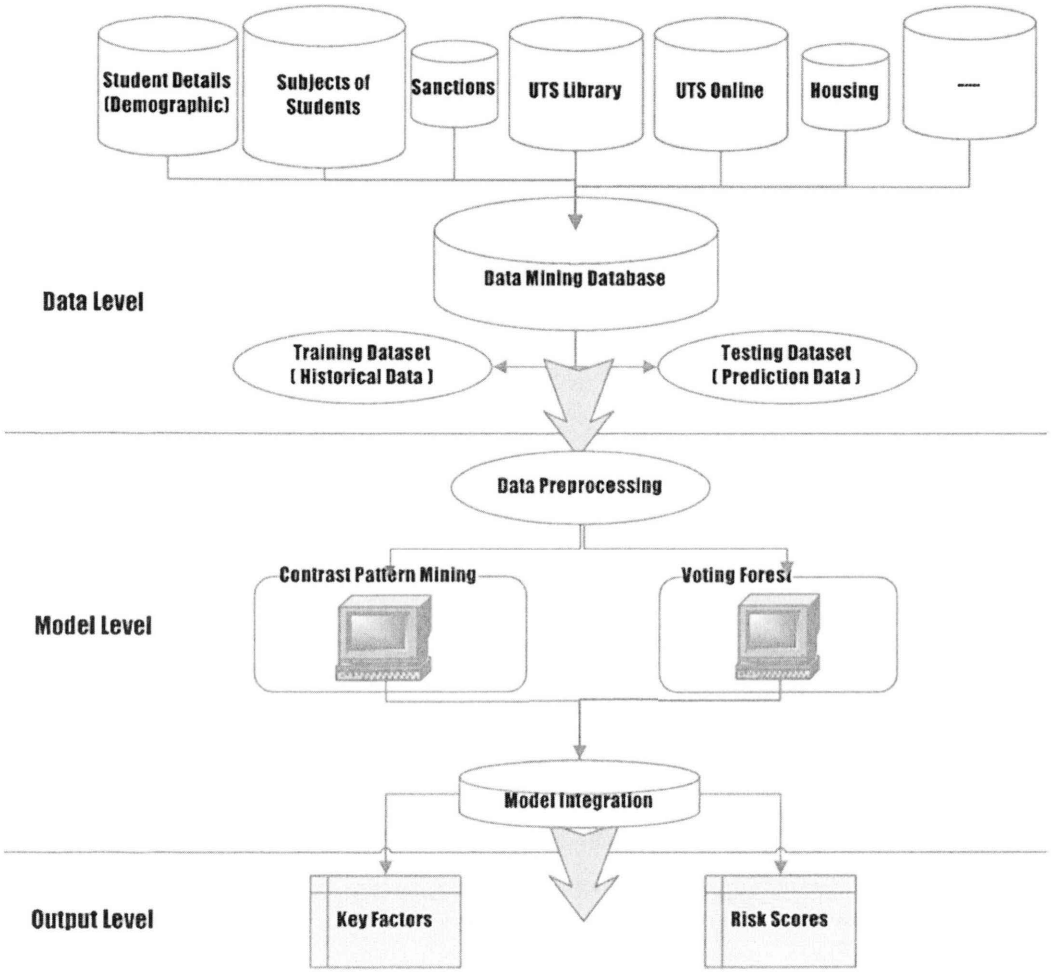

Figure.6.6. Predictive model based system structure

Firstly, we extract all historical student data, including demographic, sanctions and activities, subject details of students, etc., from different systems in the university. All the data will be processed to get meaningful nominal or numeric factors, such as demographic factors, subject factors, sanction and activity factors. For example, how many activities a student joined in the current semester, how many times he/she failed in a special subject, what is the socio economic status of the suburb the student lives in, etc. All of these factors

are then considered from both business and technical perspectives, and more meaningful and effective factors will be found gradually and added into data mining models to make their prediction more precise. Some other student data, such as the library records, student online records, various survey data, assignments scores, and so on, can be considered to get more useful and comprehensive information. Based on the historical student data, we can train a couple of data mining models to predict the failure risk of current students who are studying the targeted "killer subjects" and the scores still unknown, and then calculate risk scores for them. Two data mining models will be used for the prediction.

- The first one is the Contrast Pattern mining model, which can find discriminative factors between historical pass students and fail students. The model will tell us what kinds of factors or combined factors are likely to drive student failure. This will be further addressed in Section 6.2.
- The second one is called the Voting Forest. It improves the classic decision tree model by the ensemble approach, which is cascading-and-sharing for constructing decision trees 59 UTS Learning Analytics, Phase II and voting for risk score by all the trees. Based on this model, the risk score of each student is able to be calculated. Higher risk students will have higher scores.

Finally, by combining the above two models, high risk students and the corresponding trigger factors are listed as output. The generated risk lists of all students who enrolled in the current semester will be extracted to SSU, for them to further follow up those high risk students.

Usually, data pre-processing is the most time consuming task in a data mining related project. It may take as long as two thirds of the whole period of the project. The quality of data pre-processing directly determines the resultant quality of the data mining model and the outcomes.

The data pre-processing and integration procedure can be roughly divided into the following subtasks:

- Background knowledge learning and data field selection. To make full use of the data, one should have a deep understanding of the background knowledge. We took quite a long time to discuss with customers about the domain knowledge. Finally, we are able to pick out tables and fields related to the students' demographic information and performance records.
- Data Cleaning and preparing. We check data fields one by one, rectify errors in the data, and remove unwanted and meaningless fields like 'N/A' so that it is ready for data integration.
- Statistical feature construction. A lot of statistical features were constructed to enhance the feature set. These statistical features include but are not limited to: students' failures in different subjects, engagement features that integrate activities and sanctions, year gaps between graduation from secondary school and enrolment in UTS, failure rate by suburb, birth country, home language, etc.
- Data integration. All data integrated into the SQL Server is shown in Figure 6.6.
- Feature selection. Different feature selection methods like information gain and entropy are used to select candidate features for future analysis and model building in the later steps.

6.2.3. Outcome

There are two user interfaces in the system. One is the dashboard for investigating risky students and another is for building models. Figure 6.7 shows the dashboard providing risk rating and further investigation. Through this interface, users can know who are at risk, how risky they are, what their profiles are and why they are at risk. The investigation interface is composed of the following areas.

1. Search conditions, where users set the filters/conditions to select students to be rated in terms of risk. The conditions include entry year, subject, student type and risk rank. Other factors can be added to the filter set.
2. List of risky students, where lists the risky students who satisfy the search conditions. Each student is associated with a risk score and a risk rank to indicate the likelihood of risk the student may fail a certain subject in the selected group.
3. Selected group profile, which displays the statistical information about all the risky students shown in rectangle. The statistical information includes risk level of academic failure, Socio-economic level and education background. Other group-oriented factors can be added.

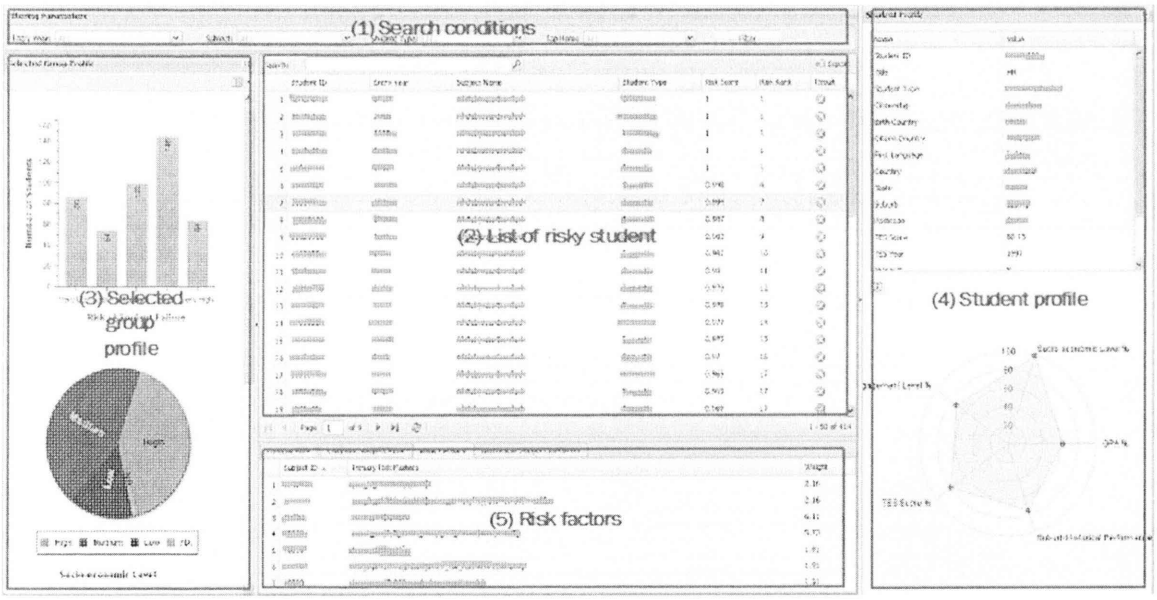

Fig. 6.7. the interface of dashboard

4. Student profile, where detailed profile information about an individual student is displayed, such as student citizenship, birth country, suburb, TES score, scholarship and etc. There is also a radar chart in this area. There are 5 dimensions in this chart to show

different aspects of students' status in the study: Engagement, GPA value, Socio-economic status, Suburb Performance and TES score. Explanations of the 5 dimensions are listed as below.

4.1. Engagement: To measure how many academic activities the student has engaged in, as well as the academic sanctions the student has been taken. A sigmoid function is used to combine students' activity records and academic sanctions together and shape them between 0 and 1.

4.2. GPA value (Grade Point Average): GPA is calculated by taking the number of grade points a student earned in a given period of time divided by the total number of credits taken.

4.3. Socio-economic status: Socio-economic status base on Australian population, derived from 2006 SEIFA index.

4.4. Suburb Performance: The historical performance in terms of failure rate for the suburb where the student is living in. In mathematical calculation, this field is defined as one minus suburb failure rate.

4.5. TES Score: Students' Tertiary Entrance Scores.

5. Risk factors. This area integrates functions relate to risk factors. There are several tabs in this area. They are overall risk, current subject risk, risk rules, risk factors and historical subject records.

5.1. Overall risk. This tab shows risk levels of all subjects this student is taking in the current semester. This gives the system user a rough idea of how the student is going on with every subject.

5.2. Current subject risk. This shows the risk level of the current subject.

5.3. Risk rules. This tab shows combined contrast rules which provide the reasons why a student fails. Typically, each reason is a combination of multiple factors. Each reason is associated with a weight, which indicates how important the reason is to the student failure.

5.4. Risk factors. To make user to understand the failure reasons much easier, we split combined rules and list out the most persuasive single factors in this tab. Every factor is associated with a weight, which indicates how important the reason is to the student failure.

5.4.1. Historical subject records. This tab shows students all historical subject records. This helps the system users to understand how the student performs in the past, if he/she has gained qualified background knowledge for the current subject.

Figure 6.8 shows the user interface of model builder, which is for data analyst users to build or update models. The interface consists of the following areas:

1. List of models, which show a list of models that have been built in the system. Analysts can add a new model.
2. Training data set, where users select the training data source for model building.
3. Training parameters, where user set the initial parameters for model building.
4. Test data set, where users select the test data source for evaluating model accuracy.
5. Operation buttons, which are the buttons to trigger model training, test and saving.
6. Message panel, which displays all messages during the course of model building, such as the model training progress, model accuracy and errors.

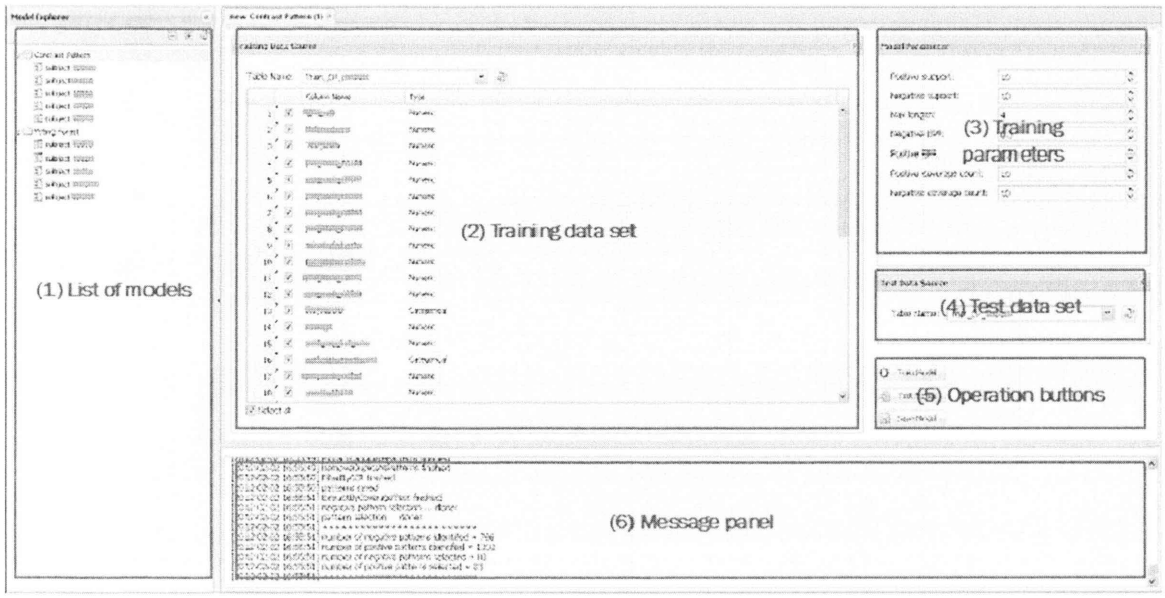

Figure.6.8. Interface of Model Builder

To evaluate the performance of this system, we compare our system's predictive accuracy against random selection. The data for evaluation was historical student data. The data was separated into two parts. The first part was used to build models while the second was used to calculate predictive accuracy.

As shown in Figure 38, the training data for subjects 48510, 48520, 48521 and 48530 covers four semesters from 2009 Autumn to 2010 Autumn, while the training data from subject 48540 covers seven semesters from 2002 Autumn to 2005 Spring. The reasons why we used 7-semester data for subject 48540 is that there were a few students enrolled in this subject per semester. Four-semester data is not sufficient for model training.

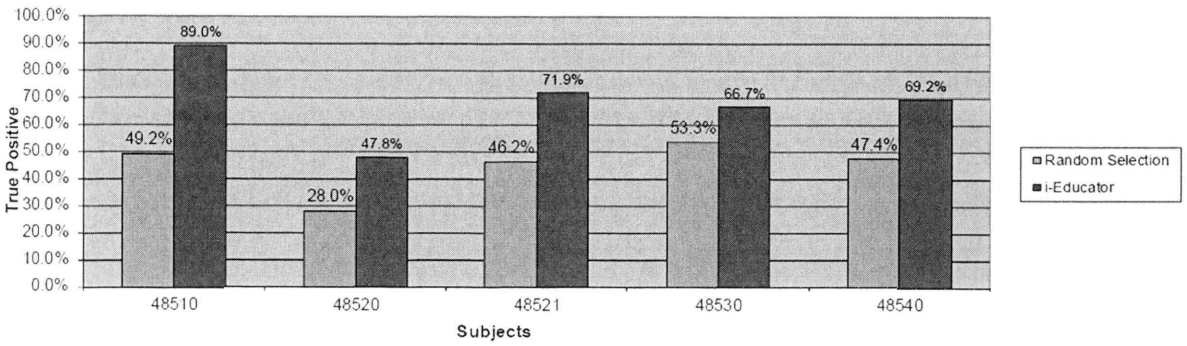

Figure.6.9. System Performance

After the models were built on the training data, we used test data to calculate our system's predictive accuracy. The test data come from the successive semester/s. For subjects 48510, 48520, 48521 and 48530, it was 2010 Spring. For subject 48540, they were 2006 Autumn and Spring. The predictive accuracy can be calculated by comparing the prediction results with the students' real results. As shown in Figure 6.9, in the case of 50% cut-off, our system's true positive is 89%, 47.8%, 71.9%, 66.7% and 69.2%, namely 1.8, 1.7, 1.6, 1.3 and 1.5 times of the random selection's accuracy, for subjects 48510, 48520, 48521, 48530 and 48540 respectively.

To make it clearer, we take subject 48510 as example. In semester 2010 Spring, there were 183 students who were enrolled in this subject. 93 students passed and 90 students failed. The failure rate was 49.2%. Our system assigned a risk score for each student and ranked the students according to their risk scores from the highest to the lowest. Among the top 91 risky students, that is 50% cut-off, there were 81 students who actually failed. Our system's true positive is 81/91=89% in the case of 50% cut-off, while random selection is equal to the failure rate of this subject, 49.2%. This means that our system is 1.8 times more accurate than the random selection method.

The evaluation results show that our system outperforms random selection for all four subjects and achieves reasonably good accuracy. It is clear that the system output is able to

help users easily understand students' risk levels, historical performances, current status, key reasons that cause the risks, and can even make some simple conclusions based on the key reasons (eg. student's educational backgrounds, background knowledge in fundamental subjects, engagement, cultural issues or language problems).

Chapter 7

Conclusion

This chapter presents a brief summary of the thesis, presenting a complete list of the techniques and strategies introduced, discussing then the main contributions of our work, and finally outlining some future lines of research.

7.1. Summary

In this section we are going to present a brief summary of the work presented in this thesis with the purpose of having a global view of the entire thesis.

With the purpose of designing a flexible and high performance risk analysis and management system, in this work we have focused on agent-based service-oriented risk mining architecture (ABSORM), that are the integration of multi-agent systems, service-oriented architecture and business process management. Moreover, agent-based ensemble learning methods can be used to achieve high accuracy and robustness of risk mining model.

The work on ABSORM architecture presented in this thesis can be grouped in three main areas, as Figure 7.1 shows:

- Work on agent-based service-oriented architecture, focusing on how multi agent system can be adopted to service-oriented architecture.
- Work on agent-based business process management, focusing on how agents can coordinate to work as business process that improves the system stability and flexibility.
- Work on agent-based ensemble learning strategies, focusing on how agents can coordinate to act as ensembles that achieve better performance that simple model.

Figure 7.1 shows these three areas of work, together with the specific techniques proposed or used in this thesis for each area.

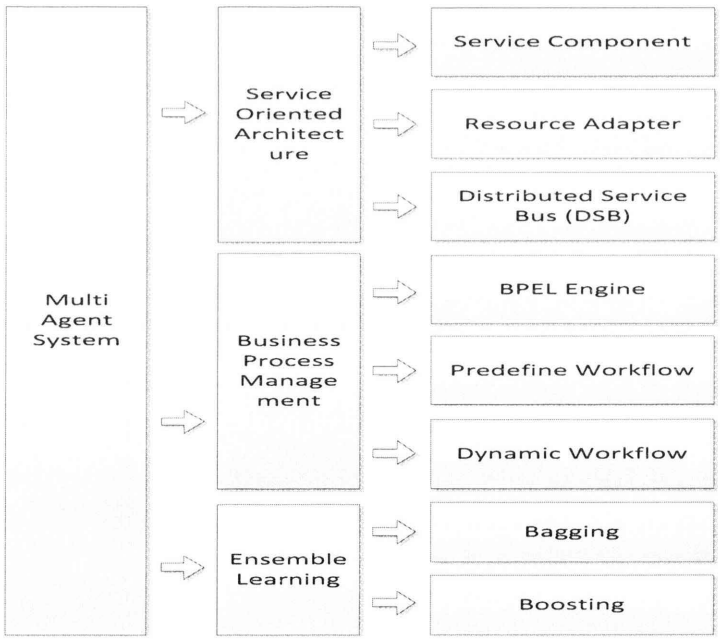

Figure 7.1 the different techniques presented in this thesis, group by areas.

7.2. Future work

There are several issues in our research that will be subject of future research. First of all, we would like to adopt the risk mining system to cloud computing to archive, analyze and mine large distributed data sets. Cloud computing is one of the most popular next generation architecture in software engineering. A lot of large companies are going to transfer their current systems to cloud environment. By a cloud, it provides resources and services over the Internet. A storage cloud provides storage services, while a compute cloud provides computing services. In that case, the infrastructure can be more loosely couple and agility.

The performance of agents is also a subject for future research. During our work, we have used a Case Mediator Agent (CMA) to control the collaboration between agents. However, one of the important features of agent is autonomous that every agent can communicate with their neighbors and decide when to collaborate and with whom to collaborate. In this way, we should develop learning techniques to improve the performance of the agents (both individually and as an ensemble). Specifically, we should focus on two types of learning: learning processes that allow agents to improve individual problem solving performance, and learning processes that allow agents to improve their collaboration.

We are also interested in applying our techniques to other domains than banking and education, such as stock market, health care or insurance etc. These new domains raise several challenges which can help us to perfect this system infrastructure.

Appendix A: List of Publications

Wei, W., Li, J., Cao, L., Ou, Y., Chen, J.: Effective detection of sophisticated online banking fraud on extremely imbalanced data. World Wide Web, 1-27. doi:10.1007/s11280-012-0178-0

Moemeng, C., Zhu, X., Cao, L. & Jiahang, C. 2010, 'i-Analyst: An Agent-Based Distributed Data Mining Platform', IEEE, pp. 1404-6.

Zhu, X., Yu, Y., Ou, Y., Luo D., Zhang, C., & Chen, J., 'System modeling of a smart-home healthy lifestyle assistant', ADMI 2012

Cao L., Ou, Y., & Chen J., 'i-Alertor: online risk analysis and management system', PAKDD 2012

Bibliography

Cao, L. 2009, 'Introduction to agent mining interaction and integration', in Longbing Cao (eds), *Data Mining and Multi-agent Integration,* Springer, pp. 3-36.

Cao, L., Zhang, C. & Dai, R. 2005, 'Organization-oriented analysis of open complex agent systems', *Int. J. on Intelligent Control and Systems*, vol. 10, no. 2, pp. 114-22.

Cao, L., Zhang, C. & Dai, R. 2005, 'The OSOAD methodology for open complex agent systems', *Int. J. on Intelligent Control and Systems*, vol. 10, no. 4, pp. 277-85.

Cao, L., Zhang, C. & Ni, J. 2005, 'Agent services-oriented architectural design of open complex agent systems', IEEE, pp. 120-3.

Moemeng, C., Zhu, X.H. & Cao, L.B. 2010, 'Integrating workflow into agent-based distributed data mining systems' ADMI 2010, LNCS 5980, pp. 4-15.

Albashiri, K.A. & Coenen, F. 2009, 'A generic and extendible multi-agent data mining framework', in E.Corchado et al. (eds), *HAIS 2009, LNAI 5572,* Springer-Verlag Berlin Heidelberg, pp. 203-10.

Baik, S.W., Bala, J. & Cho, J.S. 2004, 'Agent based distributed data mining', in K.-m.Liew et al. (eds), *PDCAT 2004, LNCS 3320*, Springer-Verlag Berlin Heidelberg, pp. 42-45.

Liu, X. 2008, 'An agent-based architecture for supply chain finance cooperative context-aware distributed data mining systems', *The Third International Conference on Internet and Web Applications and Services*, pp. 261-66.

Manfred, K., Warmuth & S.V.N. Vishwanathan. 2009, 'Survey of Boosting from an Optimization Perspective', ICML'09, Montreal, Canada.

Robert Schapire, 2007, 'Theory and Applications of Boosting', NIPS'07, Vancouver, Canada.

Giovanni Seni & John Elder, 2007, 'From Trees to Forests and Rule Sets-A Unified Overview of Ensemble Methods', KDD'07, San Jose, CA.

Moemeng, C., Gorodetsky, V., Zuo, Z.Y. & Zhang, C.Q. 2009, 'Agent-based distributed data mining: a survery', in Longbing Cao (eds), *Data Mining and Multi-agent Integration,* Springer, pp. 57-58.

Moemeng, P., Cao, L.B. & Zhang C.Q. 2008, 'F-TRADE 3.0: An agent-based integrated framework for data mining experiments', *IEEE/WIC/ACM International Conference on Web Intelligence and Intelligent Agent Technology*, pp. 612-5.

Silva, J.C.D., Giannella, C., Bhargava, R., Kargupta, H. & Klusch, M. 2005, 'Distributed data mining and agents', *Engineering Application of Artificial Intelligence 18*, pp. 791-807.

Zaidi, S.Z.H.Z., Abidi, S.S.R., Manikam, S. & Yu, C. 2004, 'ADMI: a multi-agent architecture to autonomously generate data mining services', *Second IEEE International Conference of Intelligent Systems*, pp. 273-9.

X.Y. Liu, J. Wu, and Z.H. Zhou, 2006, 'Exploratory Under Sampling for Class Imbalance Learning', Proc. Int'l Conf. Data Mining, pp. 965-9.

Cabri, G., Leonardi, L. & Puviani, M. 2007, 'Service-oriented agent methodologies', IEEE, pp. 24-9.

Castellano, M., Mastronardi, G., Aprile, A., Bellone de Grecis, G. & Fiorino, F. 2007, 'Applying a Flexible Mining Architecture to Intrusion Detection', IEEE, pp. 845-52.

Chen, Y. & Cohen, B. 2005, 'Data mining and service rating in service-oriented architectures to improve information sharing', IEEE, pp. 1-11.

Cheung, W.K., Zhang, X.F., Wong, H.F., Liu, J., Luo, Z.W. & Tong, F.C.H. 2006, 'Service-oriented distributed data mining', *Internet Computing, IEEE*, vol. 10, no. 4, pp. 44-54.

Cheung, W.K., Zhang, X.F., Wong, H.F., Liu, J., Luo, Z.W. & Tong, F.C.H. 2006, 'Service-oriented distributed data mining', *Internet Computing, IEEE*, vol. 10, no. 4, pp. 44-54.

Courbis, C. & Finkelstein, A. 2004, 'Towards an aspect weaving BPEL engine'. Dang, J., Huang, J. & Huhns, M.N. 2007, 'Workflow coordination for service-oriented multiagent systems', ACM, p. 249.

Erl, T. 2005, *Service-oriented architecture: concepts, technology, and design*, Prentice Hall PTR.

Gorodetskiy, V., Karsaev, O., Samoilov, V. & Serebryakov, S. 2006, 'Agent-based Service-Oriented Intelligent P2P Networks for Distributed Classification', vol. 2, IEEE, pp. 224-33.

Grundspenkis, J. & Pozdnyakov, D. 2006, 'An Overview of the Agent Based Systems for the Business Process Management', pp. 221-8.

Guedes, D., Meira, W. & Ferreira, R. 2006, 'Anteater: A service-oriented architecture for high-performance data mining', *Internet Computing, IEEE*, vol. 10, no. 4, pp. 36-43.

Huhns, M.N. & Singh, M.P. 2005, 'Service-oriented computing: Key concepts and principles', *Internet Computing, IEEE*, vol. 9, no. 1, pp. 75-81.

Kimlaychuk, V. 2008, 'Integrating Oracle enterprise service bus with JADE agents', IEEE, pp. 59-61.

Moemeng, C., Zhu, X. & Cao, L. 2010, 'Integrating workflow into agent-based distributed data mining systems', *Agents and Data Mining Interaction*, pp. 4-15.

Moemeng, C., Zhu, X. & Cao, L. 2010, 'Integrating workflow into agent-based distributed data mining systems', *Agents and Data Mining Interaction*, pp. 4-15.

Padma, S., Kumar, S.S. & Manavalan, R. 2011, 'Performance analysis for classification in balanced and unbalanced data set', IEEE, pp. 300-4.

Poggi, A., Tomaiuolo, M. & Turci, P. 2007, 'An agent-based service oriented architecture', pp. 157-65.

Spanoudakis, N. & Moraitis, P. 2008, 'An ambient intelligence application integrating agent and service-oriented technologies', *Research and Development in Intelligent Systems XXIV*, pp. 393-8.

Tapia, D., Rodríguez, S., Bajo, J. & Corchado, J. 2009, 'FUSION@, a SOA-based multi-agent architecture', Springer, pp. 99-107.

Wang, L., Wang, Q. & Ni, L. 2011, 'The Analysis and Design of SOA-Based Financial Data Mining System', vol. 1, IEEE, pp. 129-32.

Wen, Q. & He, J. 2006, 'Personalized recommendation services based on service-oriented architecture', IEEE, pp. 356-61.

Wu, L., Barash, G. & Bartolini, C. 2007, 'A service-oriented architecture for business intelligence', IEEE, pp. 279-85.