Framework for distributed knowledge discovery systems embedded in extended enterprise

This item was submitted to Loughborough University's Institutional Repository by the/an author.

Additional Information:


Metadata Record: [https://dspace.lboro.ac.uk/2134/33990](https://dspace.lboro.ac.uk/2134/33990)

Publisher: © Elena Irina Neaga

Rights: This work is made available according to the conditions of the Creative Commons Attribution-NonCommercial-NoDerivatives 4.0 International (CC BY-NC-ND 4.0) licence. Full details of this licence are available at: [https://creativecommons.org/licenses/by-nc-nd/4.0/](https://creativecommons.org/licenses/by-nc-nd/4.0/)

Please cite the published version.
FRAMEWORK FOR DISTRIBUTED KNOWLEDGE DISCOVERY SYSTEMS EMBEDDED IN EXTENDED ENTERPRISE

By

ELENA IRINA NEAGA

A Doctoral Thesis
Submitted in partial fulfilment of the requirements
for the award of
Doctor of Philosophy of Loughborough University
July 2003

© by Elena Irina Neaga, 2003
ABSTRACT

One of the foremost challenges facing manufacturing industry nowadays is the large-scale integration of their enterprise systems, along with their associated models, data and information. The enterprise applications generate large amounts of data which are a valuable asset and potentially an important source of new information and knowledge for improving the business of the enterprise, gaining competitive advantage in fierce markets and coping with changes and managerial complexity.

The research reported in the thesis is focused on the modeling, design, development and implementation of knowledge discovery and data mining systems, by considering multiple views including knowledge, mining, information, data and application views defined at the level of enterprise reference architecture. An important aspect of the work has been to explore a common enterprise model that takes advantage of new technologies, yet also protects the investments that have already been made in existing systems and projects. This has resulted in an approach which integrates knowledge discovery and data mining into existing enterprise reference architectures and frameworks.

The research has defined a flexible and dynamic, unified object-oriented framework that can include an enterprise's current applications, integrate existing enterprise systems and provides an environment for future applications development. The architecture of the knowledge discovery system is supported by the unified object-oriented framework. This architecture has a standard structure that is platform-independent, and could be integrated with legacy, existing and future applications.

The research reported in this thesis also demonstrates that if an IT system follows reference architectures and models and also uses Object Management Group standards such as Unified Modeling Language, Common Object Request Broker Architecture and Model-Driven Architecture then standard Knowledge Discovery and Data Mining systems can be embedded in the manufacturing system design, and can be flexibly integrated within the enterprise reference architectures.
KEYWORDS: KNOWLEDGE DISCOVERY (KD), DATA MINING (DM), EXTENDED ENTERPRISE (EE), UNIFIED OBJECT-ORIENTED FRAMEWORK, REFERENCE ARCHITECTURE, INFORMATION AND DATA VIEWS, KNOWLEDGE VIEW, MINING VIEW, APPLICATION VIEW, COMMON ENTERPRISE MODEL, SUPPLY CHAIN MANAGEMENT (SCM), CUSTOMER RELATIONSHIP MANAGEMENT (CRM), ENTERPRISE RESOURCE PLANNING (ERP), UNIFIED MODELING LANGUAGE (UML), COMMON OBJECT REQUEST BROKER ARCHITECTURE (CORBA), COMMON WAREHOUSE META MODEL (CWM), EXTENDED ENTERPRISE COMMON WAREHOUSE META MODEL (EE_CWM), MODEL-DRIVEN ARCHITECTURE (MDA), MULTI-TIER ARCHITECTURE, ENTERPRISE JAVA BEANS (EJB)
DEDICATED

To my beloved son - ALEXANDRU NEAGA

And

In loving memory of my father – CONSTANTIN ROSCA
ACKNOWLEDGEMENTS

This research was fully supported by Wolfson School of Mechanical & Manufacturing Engineering of Loughborough University, Loughbourough, United Kingdom.

Dr Jennifer Anne Harding supervised me, with competitive professionalism, patience and kindness. I express my gratitude for her effort directed to fill the gaps between my previous education and experience acquired mainly at “Gh. Asachi” Technical University, Iasi, Romania, and the scientific requirements at Loughborough University, United Kingdom. Thank you!

I am very grateful to my families: Alexandru & Decebal Neaga, Aurora Rosca and Tedy & Ovidiu Rosca for their help, deep understanding and the acceptance to live away from home for such a long time.

Many thanks to All my Friends and Colleagues from different countries and Universities, for their unconditional trust, encouragements in difficult moments of doubts and giving up, and for sharing of our experiences.
# CONTENTS

<table>
<thead>
<tr>
<th>Chapter</th>
<th>INTRODUCTION</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.1</td>
<td>General Research Context</td>
<td>1</td>
</tr>
<tr>
<td>1.2</td>
<td>Aims and Objectives</td>
<td>6</td>
</tr>
<tr>
<td>Chapter 2</td>
<td>RESEARCH MOTIVATION AND CONTRIBUTION</td>
<td>9</td>
</tr>
<tr>
<td>Chapter 3</td>
<td>LITERATURE REVIEW</td>
<td>13</td>
</tr>
<tr>
<td>3.1</td>
<td>State of Art of Knowledge Discovery and Data Mining</td>
<td>13</td>
</tr>
<tr>
<td>3.1.1</td>
<td>Foundation and Definitions</td>
<td>13</td>
</tr>
<tr>
<td>3.1.2</td>
<td>Data Mining and Statistics</td>
<td>19</td>
</tr>
<tr>
<td>3.1.3</td>
<td>Basic Algorithms</td>
<td>19</td>
</tr>
<tr>
<td>3.1.3.1</td>
<td>Regression</td>
<td>20</td>
</tr>
<tr>
<td>3.1.3.2</td>
<td>Classification</td>
<td>21</td>
</tr>
<tr>
<td>3.1.3.3</td>
<td>Association Rules</td>
<td>23</td>
</tr>
<tr>
<td>3.1.3.4</td>
<td>Clustering</td>
<td>24</td>
</tr>
<tr>
<td>3.1.3.5</td>
<td>Sequential Patterns</td>
<td>25</td>
</tr>
<tr>
<td>3.1.3.6</td>
<td>Combined technique</td>
<td>26</td>
</tr>
<tr>
<td>3.1.3.7</td>
<td>On-Line Analytical Processing</td>
<td>28</td>
</tr>
<tr>
<td>3.1.4</td>
<td>Integration Issues</td>
<td>30</td>
</tr>
<tr>
<td>3.2</td>
<td>Extended Enterprise Application Areas</td>
<td>33</td>
</tr>
<tr>
<td>3.2.1</td>
<td>Application View</td>
<td>33</td>
</tr>
<tr>
<td>3.2.2</td>
<td>Supply Chain Management</td>
<td>36</td>
</tr>
<tr>
<td>3.2.3</td>
<td>Customer Relationship Management</td>
<td>39</td>
</tr>
<tr>
<td>3.2.4</td>
<td>Enterprise Resource Planning</td>
<td>42</td>
</tr>
<tr>
<td>3.3</td>
<td>Knowledge Discovery Projects, Methodologies and Standards</td>
<td>44</td>
</tr>
<tr>
<td>3.3.1</td>
<td>Introduction</td>
<td>44</td>
</tr>
<tr>
<td>3.3.2</td>
<td>Critical Overview of the Projects</td>
<td>46</td>
</tr>
<tr>
<td>3.4</td>
<td>Summary</td>
<td>51</td>
</tr>
<tr>
<td>Chapter 4</td>
<td>RESEARCH BACKGROUND</td>
<td>52</td>
</tr>
<tr>
<td>4.1</td>
<td>Architecture and Framework Concepts</td>
<td>52</td>
</tr>
<tr>
<td>4.2</td>
<td>Principles of Modeling and Associated Tools</td>
<td>54</td>
</tr>
<tr>
<td>4.3</td>
<td>Model-Driven Architecture</td>
<td>57</td>
</tr>
<tr>
<td>4.4</td>
<td>Common Warehouse MetaModel</td>
<td>59</td>
</tr>
<tr>
<td>4.5</td>
<td>Java™ Capabilities for Enterprise Applications</td>
<td>61</td>
</tr>
<tr>
<td>4.5.1</td>
<td>Understanding Enterprise JavaBeans</td>
<td>63</td>
</tr>
<tr>
<td>4.6</td>
<td>Summary</td>
<td>66</td>
</tr>
</tbody>
</table>
Chapter 1

INTRODUCTION

1.1 GENERAL RESEARCH CONTEXT

Manufacturing industry is changing fundamentally as multinational corporations become globalized and collaborative. Global economy determines that manufacturing companies are moving very fast from stand-alone applications, legacy systems and local databases to Internet-based distributed and collaborative environments supporting the extended or global enterprise. The extended enterprise is mainly characterized by the replacement of rigid supply chain structures by globally distributed and dynamic networks. On the other hand globalization is not moving at the same speed as the Internet revolution [Naisbitt, 2000]. Therefore in an economy where the only certainty is uncertainty, the one sure source of lasting competitive advantage is knowledge [Nonaka, 1998]. In order to cope with intensive changes and the fierce market place competition, manufacturing companies use integrating management strategies, human factors, production processes and control techniques. To achieve a high level of integration of managerial and technical elements, companies are resorting to standard reference architectures and common enterprise models usually developed within international projects. Also large organizations such as manufacturing companies could respond to changes and challenges in their business and production activities by using intensive and intelligent database processing in order to find new trends and to predict and improve their business performance. Knowledge discovery (KD) in databases is a promising solution and the extended enterprise architectures have to incorporate data mining (DM) systems which could significantly contribute to improving the decision making process and business performance. This research mainly addresses the area of applied DM within the framework of enterprise engineering and integration. It also deals with mining modeling aspects and their correlation with information models in manufacturing. One of the most important aims of the research reported in this thesis is to include and accommodate a generic DM system within the existing standardized and referenced enterprise architectures and models.

Knowledge discovery and data mining are interdisciplinary areas based on statistical analysis, database approaches and artificial intelligence (AI), especially machine learning. KD and DM incorporate complex algorithms from statistics and AI, including imaginative,
and intuitive processing. Like other evolutionary systems, especially those based on neural networks, DM applications are tending to use both rational and emotional intelligence defined as affective intelligence [Adami, 1998], [Goleman, 1995, 1997]. The main DM techniques are On-Line Analytical Processing (OLAP) and methods based on classification, association rules, clustering, decision trees, sequential patterns, fuzzy logic, and combinations of algorithms such as neural networks (NN) and case-base reasoning (CBR) [Adamo, 2001], [Agrawal, Srikant, 1995], [Bramer, 1999], [Fayyad et al., 1996], [Han, Kamber, 2001]. Support Vector Machines (SVM) are a new generation of machine learning algorithms based on recent advances in statistical learning theory, proposed by Vladimir Vapnik (1995) [Cristianini, Taylor, 2000]. KD and DM theory and main algorithms are discussed in detail in section 3.1 of chapter 3.

Generally, these algorithms and their implementations are applicable to manufacturing enterprises, and may be used to address the challenges faced by the extended manufacturing enterprise. This is mainly explained in section 3.2.

In chapter 5 additional views are defined within current enterprise reference architectures and standard models. The knowledge, information and data views support knowledge discovery and mining processing embedded in extended enterprise. The obtained mining models are included in the mining view.

Nowadays the trends in DM are towards standardization, using common methods and tools, and defining repeatable projects. CRISP-DM™ (Cross Industry Standard Process for Data Mining) SolEuNet (Data Mining and Decision Support for Business Competitiveness: A European Virtual Enterprise) Kensington Enterprise Data Mining (Imperial College, Department of Computing, London) and other projects have established methodologies, and developed languages and software tools for KD and DM processing. However most of the DM products are focusing on the DM technology rather than on the ease-of-use, integration, scalability and portability. Several integration aspects are still neglected as shown in sections 3.1 and 3.3.

Manufacturing enterprises rely on vast amounts of data and information that is located in large databases. This information is a valuable resource, but its value can be increased if additional knowledge can be gained from it. The exploration of database information, to identify and extract deep and hidden knowledge, is made possible by DM techniques. The existing databases of manufacturing enterprises, or indeed of most large organizations, are huge, but largely untapped sources of information, since they contain valuable records of
operational and market history. DM techniques can be used to improve strategic and operational planning activities, as databases can be explored to gain feedback on the past performance and business behaviour of the enterprise. Extended enterprises are based on advanced information technology, and DM applications provide valuable additional technologies for e-business operations and transactions.

Therefore the technology of DM and KD will become an integral part of the distributed enterprise architecture. In order to achieve this there is still a need for methodological approaches and architecture definitions and improvements that are correlated with the extended enterprise approaches and information modeling in manufacturing. The extended enterprise is described as the partnership between manufacturing enterprises, having the aim to achieve competitive advantage [Bernus, Nemes, 1996], [Berry, 2000] [Harding, Popplewell, 1999], [Szegheo, 1999], [Vernadat, 1996]. The best well-known enterprise architectures in the 1990s include CIMOSA, ARIS, PERA and GERAM [Vernadat, 1996]. However, research efforts have been especially devoted to looking for new architectures, in order to accommodate the changing conditions, recent technical advances and requirements of new manufacturing paradigms. It is argued that, in general, architectures with a constitution of function, information, resource and organization views, have been found to be sufficient [Toh, 1999], [Wang et al., 2002]. These architectures do not include specific views in order to support KD processing. This is important as KD processes should bring a considerable advantage in improving business performance and the flexibility of future manufacturing system design and re-engineering. However Wang et al. (2002) have introduced a few considerations about data sources integration and data warehouse, and Williams (1996) has defined the requirements for data shared between enterprise entities.

This research is directed at defining a framework for distributed KD/DM systems embedded in extended enterprise. In order to achieve this a multiple views approach is used, and knowledge and mining views are introduced into reference architectures for enterprise modeling and integration. Therefore this thesis is focused especially on the integration aspects of DM applications within the enterprise system architectures that have already been developed and adapted in manufacturing industry.

From the IT perspective an integrating infrastructure is a set of common services and functions available as middleware to all applications on different platforms of a distributed system [Vernadat, 1996]. Object Management Group (OMG) has developed Common Object Request Broker Architecture (CORBA) and most recently Model-Driven
Architecture (MDA) as very important and ambitious middleware projects. [Orfali et al., 1997], [Orfali, Harkey, 1998). Using CORBA services and the related Interface Definition Language (IDL) a set of DM-Corba system interfaces have been generated. Based on the Common Warehouse Metamodel (CWM) supported by MDA, an architecture of a DM system has been modeled and designed. The specifications and the capabilities of an identified system are defined and supported by generating the main class declarations. The research reported in the thesis demonstrates that the DM identified architectures are standard, especially by following Reference Model of Open Distributed Processing (RM-ODP) framework defined by ISO as well as DM standards.

The DM system included in an extended enterprise should be applied to the following application areas [Ayers, 2001], [Berson et al., 2000], [Groth, 2000], [Makridakis, Wheelwright, 1989], [Shim, 2000], [Wang, 1999]:

1. Supply Chain Management (SCM);
2. Customer Relationship Management (CRM);
3. Enterprise Resource Planning (ERP);
4. Decision making support;
5. Market Analysis and Product Specification;
6. E-business including e-transaction and e-commerce applications;
7. Production and Inventory;
8. Intelligent Process Monitoring and Control;

In order to support this wide range of applications related to manufacturing enterprises, the generic DM system uses the information modeling in manufacturing defined especially by the MOSES project [MOSES, 1992], [Maxfield et al., 1998] and includes the definition of additional elements which extend the standard manufacturing and product models.

The DM system analysis and design has been done using Unified Modelling Language (UML) implemented in Rational Rose.

The classes defined for the identified system have been generated and transferred in Oracle -JDeveloper which is Oracle's Java development tool for building, debugging and deploying Internet applications. The release 3.2. includes Java Data Base Connectivity (JDBC) capability for an Oracle database, and automatic generation of Corba interfaces. It also includes Oracle Business Components for Java, Extensible Markup Language (XML) - powered application component framework that significantly simplifies the development,
deployment and customization of multi-tier Java applications for the Internet. Furthermore, *Enterprise JavaBeans* (EJB) offer the possibility to develop reusable enterprise components which include knowledge discovery and data mining.

The research is mainly demonstrated using UML implemented in Rational Rose and JDeveloper.

During the late 1980s and early '90s there was an increasing interest in object-oriented languages, such as C++, Smalltalk, Java and even Object-Cobol, which attempted to unify code and data under the umbrella of objects. However, object-oriented programming could not cover all the problems related to code–data disfunctionality. In the pure object model, everything is an object, and the result is an often complex mixture of class relationships when the working environment is data-intensive oriented.

Component technologies such as Corba, Distributed Component Object Model (DCOM) and Enterprise JavaBeans (EJB) have moved the focus to the interfaces. Data are mere parameters in a component technologies approach. On the other hand, the emergence of XML is shifting the focus again to data. The separation of data from its implementation as relational or object-oriented databases is clearly defined within the OMG's CWM which is used in the context of this research for *common enterprise modeling* based on metadata concept.

The main aim of this research is to make DM more accessible to industry. Finally, its aims are to fill the gap between data and enterprise business intelligence through DM projects, by providing support for the process, techniques and appropriate access to advanced IT tools that are necessary to support business decision making.

This thesis has been organized into ten chapters as follows:

- Chapter 2 highlights the motivation and contributions of the research reported in this thesis in the context of both enterprise engineering and data mining approaches.
- Chapter 3 deals with the literature review revealing the gaps in KD/DM researches. It is focused on the theory of knowledge discovery, data mining and corresponding algorithms. It also includes the description of main application areas such as supply chain management (SCM), customer relationship management (CRM) and enterprise resource planning (ERP). Previous KD/DM projects, methodologies and standards are presented. This chapter briefly highlights the addition of some new elements which are approached in the following chapters and especially relate to methodological
Chapter 4 introduces the most recent OMG middleware projects such as MDA and CWM and capabilities of Java in order to develop distributed enterprise applications. Principles of systems analysis and modeling, and associated tools are also dealt with.

Chapter 5 deals with a new approach of reference architectures for enterprise modeling and integration and their associated methodologies which supports knowledge discovery and data mining systems. The previous enterprise engineering projects and frameworks such as CIMOSA, ARIS, PERA, GERAM and RM-ODP are intensively considered. This chapter therefore presents the multiple views approach based on enterprise reference architectures. The unified object-oriented framework for distributed knowledge discovery systems embedded in extended enterprise is presented in detail in chapter 7.

Chapter 6 includes proposed architectures for knowledge discovery and data mining systems using Corba/IDL and Enterprise JavaBeans. A multi-tier system architecture is also dealt with.

Chapter 7 provides a detailed modeling solution defined as a unified object-oriented framework for knowledge discovery and data mining distributed system.

Chapter 8 presents general implementation solutions in the context of this research.

Chapter 9 provides the description of the identified system, its specifications and capabilities.

Chapter 10 includes the conclusions of this work, and finally it presents discussions, future research directions, recommendations and suggestions derived based on the research reported in this thesis.

1.2 AIMS AND OBJECTIVES

The most important aim of this research is to provide a relevant framework for large-scale enterprise systems integration which includes knowledge discovery and data mining systems, existing enterprise applications and legacy systems. The research defines a unified object-oriented framework for the development of distributed KD/DM systems using existing tools and programmes which have implemented complex algorithms as well as enterprise
applications. The approach is correlated with enterprise modeling approaches and integration solutions based on business requirements and technological constraints. In order to achieve the main aim of the research additional objectives also have to be accomplished. These are as follows:

- To identify the potential for applying knowledge discovery and data mining in manufacturing industry even though they have not yet been widely used.
- To identify the benefits of adding knowledge discovery and data mining within current extended enterprise architectures.
- To define generic architectures for a KD/DM system covering a wide range of applications and adhering to standards and reference models applied to extended enterprises.
- To correlate and integrate reference architectures and standards with intelligent data analysis and exploration.
- To define knowledge and mining views and demonstrate that their introduction within the modeling phase of an enterprise at the generic, partial and particular levels will significantly improve enterprise performances.
- To demonstrate that the proposed KD/DM system architecture conforms to, and satisfies the requirements of DM standardization, manufacturing enterprise modeling and principles of integration.
- To demonstrate that a high level and generic architecture facilitates concrete implementation solutions by the inclusion of additional views and especially of the application view.
- To provide a methodology that addresses the following essential activities: project definition, data identification and experimental design, pre-processing, mining and evaluation of the results.
- To provide the methods and techniques for a manager or end-user with minimal DM experience to develop such systems in manufacturing organizations.
- To extend information modeling in manufacturing to enable representations of complex manufacturing, product and business entities at different processing levels such as pre-processing, knowledge or mining modeling.
• To hide the complexity of several algorithms or their qualities. By providing a good and flexible methodology that can improve the final results of the KD/DM even when inefficient or sub-optimal algorithms are applied.

• To provide a framework for the development of a distributed KD/DM system that is based on existing and emerging standards, and which is fully compatible with OMG middleware, associated models and DM standards. At the same time the system should conform to enterprise reference architectures and frameworks, and should be compatible with current enterprise applications and legacy systems.

The achievements of the above objectives are presented in chapter 2 which deals with the contributions of the research discussed in this thesis and in chapter 10 which draws the final conclusions against the initial aims of the research.
Chapter 2

RESEARCH MOTIVATION AND CONTRIBUTION

The research reported in this thesis mainly deals with large-scale integration of extended enterprise systems, along with their associated models such as data, information and knowledge models obtained through the attractive methods and techniques of knowledge discovery in large databases and data mining.

Figure 2.1 Framework for Distributed Knowledge Discovery Systems Embedded in Extended Enterprise
Therefore one of the most important contributions is the definition of a flexible and dynamic, unified object-oriented framework that includes an enterprise's current applications and provides an environment for future applications development.

The framework depicted in figure 2.1 provides a common enterprise model that takes advantage of previous projects regarding enterprise modeling, reference and standard architectures and methodologies. Figure 2.1 illustrates that the unified object-oriented framework defines an environment for integration of knowledge discovery and data mining commercial tools such as PolyAnalyst, Clementine etc. and source libraries of programmes such as Weka, ArMiner etc. with enterprise systems for CRM, SCM and ERP. The framework conforms to reference architectures for enterprise modeling and integration and it is based on OMG middleware definitions such as CWM and MDA. This framework supports the development of platform-independent systems.

In order to achieve a high level of integration it has been necessary to do a comprehensive analysis of enterprise reference architectures and models already used in manufacturing industry. The reasons for this are that even if they are still to be used, the enterprise reference architecture must be extended to include additional views such as knowledge and mining views. The knowledge view should facilitate the knowledge discovery, management and engineering. This thesis gives evidence of the need to approach knowledge discovery and data mining in manufacturing industry in a standard way.

The literature review related to the application of knowledge discovery and data mining in industry reveals the following aspects:

- Knowledge discovery and data mining provide a good potential to improve the business performance of an enterprise, and facilitate the re-engineering and re-design of manufacturing systems as well as new product introduction, design and manufacture.
- KD and DM are not widely used in manufacturing industry compared with other areas such as banking, finance, marketing and medicine.
- There is not any DM domain-dependent software system specifically for manufacturing application areas.
- There is not any reported research related to enterprise modeling, standard architectures and reference models that considers knowledge discovery and mining modeling. There are researches focused on information modeling, knowledge
representation and introducing data warehouse according to enterprise reference architectures and ISO standards.

- The existing standards for data mining and OMG's Model-Driven Architecture (MDA), Common Warehouse MetaModel (CWM) and Sun's Enterprise JavaBeans (EJB) have not previously been used for the design and development of enterprise embedded systems.

- KD and DM systems are still oriented to developing and implementing complex algorithms such as a new machine learning algorithm called Support Vector Machines and to optimizing the existing implementations for example Association Rules. The algorithms are usually described in pseudocode.

- Researchers in DM do not generally use UML in their development and implementation of applications. They tend to adopt less methodical system analysis and modeling approaches.

- There is not any reported research or project attempting to develop and implement data mining algorithms complemented with the required steps for production planning, product data management etc.

The research reported in this thesis attempts to make a bridge and to fill the gaps between theoretical knowledge discovery and data mining and their application in extended enterprises. This contributes in both theoretical and applied knowledge discovery and data mining even if the existing algorithms and their implementations are powerful enough to cover several engineering applications.

Furthermore nowadays from an engineering perspective there is not a practical need to research the way to shorten the execution of mining algorithms by a few seconds since computational power is continually increasing. This area of research has therefore been left for computer scientists and mathematicians since this is a very important research aspect which without any doubt will generates real progress related to algorithms theory and their implementations e.g. the optimization and the implementation of Association Rules algorithm for binary databases.

The whole area of knowledge discovery and data mining must address a combination of practical or industrial and business needs and requirements for investigating these areas whilst satisfying the passions driving researchers to develop outstanding theories. The
research included in this thesis is at the boundary, approaching application of knowledge discovery and data mining in manufacturing areas.

Therefore the related contributions are described below:

• A multiple views approach for the development of distributed KD/DM systems embedded in extended enterprise.

• Extending the CIMOSA and ARIS views (function, information, organization and resources) with knowledge and mining views.

• A unified approach to data mining application areas in extended enterprise. The identified areas are Supply Chain, Customer Relationship Management Systems and Enterprise Resource Planning and associated support systems. These areas define the application view which facilitates implementation of KD/DM systems. Application view provides the data and information models for EE_ApplicationSpecification.

• Supporting the design of generic DM systems architecture by using a combination between CIMOSA and ARIS.

• Conforming generic DM architectures using distributed object-technologies such as CORBA and MDA to RM-ODP framework.

• Using UML for DM complex application development in manufacturing industry.

• Creating a metamodel of the enterprise information model defined as Extended Enterprise Common Warehouse MetaModel (EE_CWM) using metadata concept.

• Defining a unified object-oriented framework for manufacturing, product, mining and knowledge models and associated support systems based on OMG’s CWM.

• Producing the specification of a prototype system and defining its capabilities and properties.

• Investigating multi-tier architectures based on Corba infrastructure and EJB, and demonstrating their potential for enterprise systems including knowledge discovery and mining processing.
Chapter 3

LITERATURE REVIEW

3.1 STATE OF ART OF KNOWLEDGE DISCOVERY AND DATA MINING

3.1.1 FOUNDATION AND DEFINITIONS

Knowledge Discovery in Databases (KDD) and Data Mining (DM) have only recently emerged as recognized areas even though several techniques have been used for about 50 years. The terms “Knowledge discovery and data mining” have been introduced in 1990’s. The historical evolution of KDD and DM is outlined below [KDnuggets, 2001]:

- 1960s: data mining based on statistical analysis,
- 1980s: data mining for large databases;
- 1989s: knowledge discovery in databases;
- 2000s: web-mining, text mining, DM agents, integrated and collaborative DM

There are several books, research reports and articles dealing with KDD and DM theory. They are mainly focused on the techniques which have their foundation in Statistics, Artificial Intelligence and Machine Learning. The most outstanding approaches and research are described in [Agrawal, Srikant, 1995], [Bramer, 1999], [Piatetski-Shapiro, Frawley, 1991], [Ebecken, 1998], [Fayyad et al., 1996], [Fayyad et al., 1996a], [Han, 1998], [Han, Kamber, 2001], [Kohonen, 1997], [Levine, 2000], [Liu, Motoda, 1998], [Liu, Siau, 2001], [Riccia et al., 2000], [Weiss, Indurkhya, 1998].

One of the first applications of DM was the analysis of transaction data stored by a supermarket in order to improve the way in which the products are arranged on shelves [Cristofor, 2002]. In the last 10-12 years, the most successful area of applications of DM is marketing, especially direct marketing, and most recently customer relationship management. In industry, the evolutionary systems have been applied since 1980’s which were the most successful time for artificial intelligence technologies. However during the last 10-15 years all these methods and techniques which have as input large amounts of data have been joined under a common umbrella called DM. The main reason is that nowadays the amount of data generated by many applications has dramatically increased, and this data is a valuable source
for the discovery of new information and knowledge. Also, the eruption of data has caused a comparable explosion in the need to analyze it which is possible by the increase of computational power which might at one time have been too computationally expensive. On the other hand the applications based on advanced algorithms such as neural networks, genetic algorithms etc. cannot neglect the large amounts of data and its structure stored in databases and obtained by running other collaborative systems including web applications.

The most important source of information on Data Mining, Knowledge Discovery, Text Mining, Web Mining, Decision Support Systems and other related topics is the KDNuggets Newsletter edited by Gregory Platetsky-Shapiro, http://www.kdnuggets.com.

The current research directions in DM are the theory of algorithms, applied DM, web mining and the hottest topics are interdisciplinary, and they are moving towards integrated DM approaches.

Liao (2003) in the article in press, presenting the review from 1995 to 2002 of knowledge management technologies and applications, includes DM as a specific topic, and classifies knowledge management technologies using the following seven categories: knowledge management framework, knowledge-based systems, data mining, information and communication technology, artificial intelligence/expert systems, database technology and modeling, together with their applications for different research and problem domains. However some of these areas have sub-topics which overlap such as databases and data mining or the topics related to knowledge discovery, representation, acquisition, reusing etc. [Liao, 2003].

Therefore at the beginning of the 21st century it is almost impossible to clearly separate research areas and their applications, and research is usually cross-disciplinary and interdisciplinary oriented.

Knowledge discovery and data mining are defined in several ways, but from the perspective of computer science the best known definitions are [Adamo, 2001], [Adriaans, Zantinge, 1996], [Bramer, 1999], [Ebecken, 1998], [Fayyad et al., 1996a], [Han, 1998], [Liu, Motoda, 1998], [Liu, Siau, 2001], [Piatetski-Shapiro, Frawley, 1991], [Weiss, Indurkhy, 1998]:

- The process of searching and retrieving or visualization of valuable information and knowledge in large volumes of data;
Knowledge Discovery in Databases (KDD) represents the extraction of interesting knowledge (rules, regularities, patterns, constraints) from data in large databases;

- Dealing with the discovery of new correlations, hidden knowledge, unexpected information, patterns and new rules from large databases;

- Representing the exploration and analysis by automatic, or semi-automatic means of large quantities of data usually stored in databases. Cristofo (2002) clearly specifies that there is no restriction to the types of data that can be used as input for DM. The input data can be a relational, or object-oriented database, a data warehouse, a web server log or a text file. DM is associated with large amounts of data, but for research and testing applications, the test data sets are of a limited length, and are usually flat files.

- The key element or the core of a process called Knowledge Discovery in Databases (KDD) dealing with several processing techniques for data especially included in large databases and data warehouse. Data warehouse is a central store of data extracted from operational data.

- Extracting previously unknown, valid and actionable information from large databases and then using the information to make critical decisions.

Nowadays knowledge discovery and data mining have incorporated ideas from business, marketing and other areas because the focus has slightly moved from theory to applications. Researchers of applied data mining have observed that the theory needs a few new elements, but instead it requires more methodologies and standards to apply it. They make contributions related to theory as well as to specific areas of application. Several research projects are inter or cross-disciplinary with respect to data mining as well as to business, finance and marketing areas. These approaches define data mining as follows [Berry, Linoff, 2000], [Berson et al., 2000], [Delmater, Hancock, 2001], [Helberg, 2002], [Pyle, 2003]:

- The process of utilizing the results of data exploration to adjust or enhance business strategies. It is based on discovered patterns, trends and exceptions found through data exploration to support the business. The information produced by data mining engines requires intelligent review by knowledgeable human experts. Discovered patterns and regularities in the data; whether or not they are significant and useful must still be
determined by a human expert who knows the domain specification very well, and can turn knowledge into action.

- Data mining finds patterns and relationships in data by using sophisticated techniques to build models. It is based on well-established statistical and machine learning techniques. A model is an abstract representation of reality which is useful to understand the business and for making decisions. There are two main kinds of models in data mining: predictive and descriptive. Predictive models can be used to forecast explicit values, based on patterns determined from known results. They could predict financial trends, market evolution, customer behaviour etc. For example from a database of customers who have already responded to a particular offer, a model can be built that predicts which prospects are most likely to respond to the same offer. Descriptive models describe patterns in existing data, and are generally used to create meaningful subgroups.

- The process by which business analysts utilize analytical technology (i.e. quantitative methodologies) to gain greater insights into important factors that drive their processes. Data mining identifies reliable relationships between variables that drive a process and a certain performance measure. With this information, managers can better control their processes and more effectively carry out strategies. Processes refer to any type of business activity (i.e. sales and marketing, pricing, costs, customer relationship management, manufacturing and others).

- An optimizing technique with several applications in business and industry which uses pattern recognition technologies as well as statistical and mathematical techniques. It provides a systematic way of finding and exploiting useful information latent in large, complex data sets, but it is not magic. Real data mining is more a methodology with the support of technology.

- It helps uncover trends in time to make the knowledge actionable.

- Within every organization is an amount of data which can describe the past performance of the organization through knowledge discovery and data mining.

The typical DM process is shown in figure 3.1, and it could be based on an existing data warehouse. Data warehouse (DW) is defined as extraction and integration of data from multiple sources and legacy systems, in an effective and efficient manner [Harding, Yu,
One of the most important purposes for using a data warehouse is to avoid duplicated or partially duplicated data in separated databases. Therefore, a data warehouse supplies the necessary information by creating data marts as shown in figure 3.1. Data marts are departmental data subsets focused on selected subjects for example a marketing data mart may include customer, product and sales information. Data warehouse represents the main store of data that has been obtained from operational data. The information in a data warehouse is subject-oriented, non-volatile, integrated and time dependent. Usually a data warehouse contains large data sets [Adriaans, Zantinge, 1996].

The KDD is formed by the following sub-processes [Adriaans, Zantinge, 1996], [Cristofor, 2002]:

- **Selection of the application area** which represents the identification of a possible application and the definition of a problem to be solved using KDD.

- **Data selection** which represents the process of choosing necessary operational data for the identified application areas and storing in a separate database in order to perform KDD.

![Figure 3.1 Data Mining through a Data Warehouse](image-url)
• **Data profiling** which is the process of checking the integrity of a database by comparing with existing models of a database schema definition.

• **Data cleaning** represents excluding duplicate records and checking the consistency of the domains.

• **Data enrichment** is the process of adding new attributes related to the existing records.

• **Data coding** represents transforming data according to SQL operations or performing conversion of database records according to specific rules in order to obtain a suitable data set for pattern finding and recognition algorithms. For example, customer's addresses could contain very detailed information and may be converted into regional codes. Coding, therefore may be a creative activity that has to be performed repeatedly in order to obtain the best result.

• **Mining data** is the stage in which an algorithm is running.

• **Data reporting** represents the process of retrieving the results of mining.

According to [Pyle, 1999] DM is also only one step in KDD. Other steps include identifying the problem to be solved, collecting and preparing the right data, interpreting and deploying models and monitoring the results. The real key to success, however, is to have a thorough understanding of the data and the application area. Algorithms can provide meaningful results only when sensibly directed.

Usually, within the framework of DM and KDD the following types of knowledge have been identified [Adriaans, Zantinge, 1996]:

• **Shallow knowledge** which represents the information that can be easily retrieved from databases using a query tool such as SQL.

• **Multi-dimensional knowledge** which is the information that can be analyzed using OLAP tools.

• **Hidden-knowledge** which is data that can be found by using pattern recognition or machine-learning algorithms

• **Deep knowledge** which is information and data that is stored in the database but can only be located by search algorithms applied to an indicated area of the database. The difference between deep and hidden knowledge is also determined by the search algorithm used. Hidden knowledge is the result of a search algorithm applied with
additional information such as the key to facilitate finding encrypted information into a database. The encrypted information usually is almost impossible to locate without a defined key.

3.1.2 DATA MINING AND STATISTICS

Statistics is an embedded component of a DM process and in the steps of KDD such as selection and cleaning data. In addition, several DM algorithms have their roots in statistical analysis. However according to the strict definitions of DM and Statistics, they do not overlap, and the main differences are presented below [Pyle, 1999]:

- Statistics assume a pattern and the algorithms attempt to prove it;
- DM describes a kind of pattern and the algorithms find them;
- In order to do DM processing, data is usually given as a database or a large flat file;
- The objective of DM is to find patterns, knowledge and valuable new information in data and through statistical analysis data is processed according to a defined objective of analysis;
- Statistics consider data variation, but this is not considered in DM;
- In DM residual data is useful, and is processed, and in statistics it is removed from the original data set.

Generally if statistics do not give optimal results then DM can be used in order to obtain unusual and unexpected, but valuable information. However DM software systems have incorporated programmes based on pure statistics such as regression.

3.1.3 BASIC ALGORITHMS

The kernel of KDD processing is composed of several advanced and complex algorithms which have been presented in detail in many reference books and articles such as [Bramer, 1999], [Cristianini, Taylor, 2000], [Ebecken, 1998], [Fayyad et al., 1996], [Han, 1998], [Liu, Motoda, 1998], [Liu, Siau, 2001], [Piatetski-Shapiro, Frawley, 1991], [Weiss, Indurkhya, 1998]:

This paragraph presents a selection of the main algorithms such as:

- Regression;
• Classification;
• Association Rules;
• Clustering;
• Sequential Patterns.

3.1.3.1 Regression

DM employs from statistics linear and non-linear regression which are widely used for correlating data. Statistical regression requires the specification of a function over which the data is fitted. In order to specify the function it is necessary to know the forms of equations governing the correlation for a data set [Wang, 1999]. Even if regression is considered to be a statistical technique, the distinction is arbitrary because DM deals with predictive modeling, and regression does exactly the same [Berry, Linoff, 2000].

According to [Fayyad et al., 1996a] and from the perspective of AI, regression is learning a function that maps a data item to a real-valued prediction variable. There are many applications of regression, for example predicting consumer demand for a new product as a function of advertising expenditure and predicting time series where the input variables can be time-lagged versions of the prediction variable.

Using this method it is possible to obtain qualitative knowledge about input-output relationships. However if prior knowledge is not available, it is necessary to find out the most probable function by trial-and-error which may be very time consuming. Feedforward neural networks do not need functions to be fixed in order to learn and they have shown remarkable results in representing non-linear functions. However the resulting function in this situation could be difficult to understand [Wang, 1999].

Sophisticated DM commercial tools such as KXEN (Knowledge Extraction Engine) implement Robust Regression called K2R which is defined as a regression algorithm that allows models to be built to predict categories or continuous variables. K2R maps a set of descriptive attributes (model inputs) to target attributes (model outputs). It uses an algorithm patented by KXEN, which is a derivation of a principle described by Vladimir Vapnik as “Structured Risk Minimization.” Instead of looking for the best performance on a known data set, K2R automatically finds the best compromise between quality and robustness. The resulting models are expressed as a polynomial expression of the input numbers. The only
element specified by the user is the polynomial degree. To improve modeling speed, K2R can also build multi-target models.

KXEN provides the following example of using K2R, and their implementation engine:

- Collect information about the past performance of dealers two years ago and associate how much of a product they sold 1 year ago;
- Train a model to predict how much a dealer will sell based on the available information;
- Check the quality and robustness of the model;
- Apply the model to all of the dealers today: the model associates each dealer with an estimation of how many products a specific dealer will sell;
- Sum up the estimates to predict how much a specific dealer will sell next year. This is the base line for the sales forecast.

3.1.3.2 Classification

Classification also known as segmentation has been defined in [Helberg, 2002] as the process of examining known groups of data to determine which characteristics can be used to identify (or predict) group membership. Examples of classification include the classification of trends in financial markets, grouping customers based on their past transactions and predicting their response to a particular product promotion [Fayyad et al., 1996], [Helberg 2002].

When a database is processed by classification records are grouped into different classes based on common properties (attributes). According to a machine learning approach, data classification develops a description or model for each class in a database, based on the features present in a set of class-labeled training data sets. The associated algorithms for classification or segmentation are as follows [Bramer, 1999],[Helberg, 2002], [Piatetski-Shapiro, Frawley, 1991]:

- Statistical algorithms: discriminant analysis, logistic regression etc.
- Decision trees are also called tree-based segmentation using a divide-and-conquer approach to classify data based on a tree representation. This class of methods also includes CHAID, classification and regression trees (C&RT), QUEST, CART and C4.5/C5.0.
- Artificial neural networks (ANN) emulate the pattern-finding capacity of the human brain and hence some researchers have suggested applying ANN algorithms to pattern-
mapping. However they have been applied successfully in only a few applications that involve classification because they require more time for processing than other decision trees algorithms. This time is called training time and it varies depending on the number of training cases, the number of weights in the network and the setting of the learning algorithm’s parameters.

- Genetic algorithms are optimization techniques that use processes such as genetic combination, mutation and natural selection in a design based on the concepts of natural evolution.

- Nearest neighbour method classifies each record in a data set based on a combination of the classes of the k record(s) most similar to it in a historical data set. Sometimes called the k-nearest neighbour technique.

- Rule induction: The extraction of useful if-then rules from data based on statistical significance.

Classification-rule learning involves finding rules or decision trees that partition given data into predefined classes. For any real-world problem domain of the classification-rule learning, the set of possible decision trees is too large to be searched exhaustively. The following algorithms for classification-rule discovery are similar with those for AI-machine learning [Piatetski-Shapiro, Frawley, 1991]:

- ID3 (Quinlan) is a decision tree building algorithm which determines the classification of objects by testing the values of their properties. It builds the tree in a top down manner starting from a set of objects and a specification of properties.

- C4.5 algorithm was proposed by Quinlan (1993). The C4.5 algorithm generates a classification-decision tree for the given data set by recursive partitioning of data. The decision tree is grown using a depth-first strategy. The algorithm considers all the possible tests that can split the data set and selects a test that gives the best information gain. For each discrete attribute, one test is considered and this produces as many outcomes as the number of distinct values of the attribute being tested. For each continuous attribute, binary tests involving every distinct value of the attribute are considered. In order to gather the entropy gain of all these binary tests efficiently, the training data set belonging to the node under consideration is sorted for the values of the continuous attribute. The entropy gains of the binary cut are based on each of the distinct
values that are calculated in one scan of the sorted data. This process is repeated for continuous attributes.

- **SLIQ (Supervised Learning In Quest)** algorithm is a decision tree classifier designed to classify large training data sets. It uses a pre-sorting technique in the tree-growth phase. This helps avoid costly sorting at each node. SLIQ keeps a separate sorted list for each continuous attribute and a separate list called the class list. An entry in the class list corresponds to a data item, and has a class label and name of the node it belongs to in the decision tree. An entry in the sorted attribute list has an attribute value and the index of a data item in the class list. SLIQ grows the decision tree in breadth-first manner. For each attribute, it scans the corresponding sorted list and calculates entropy values of each distinct value of all the nodes in the frontier of the decision tree simultaneously. After the entropy values have been calculated for each attribute, one attribute is chosen for a split for each node in the current frontier, and they are expanded to produce a new frontier. Then one more scan of the sorted attribute list is performed to update the class list for the new nodes.

**3.1.3.3 Association Rules**

*Association Rules* were introduced by R. Agrawal, T. Imielski and A. Shami, in 1993, and the most used algorithm *Apriori*, in 1994, by R. Agrawal and R. Srikant. L. Cristofor in his PhD thesis, 2002, analyses and optimizes the algorithms for Association Rules, and the prototype system implemented has been used to demonstrate this research.

The basic idea of association rules is to search the data for patterns of the following form: *IF (some conditions are true) THEN (some other conditions is probably true)* Each condition extracted from data is called an association rule, or simply a rule. Association Rules generate *rule-based models*. Association Rules have two main characteristics associated with them that measure their value: coverage and confidence. *Coverage* describes how much evidence is in the training data set to back up the rule. It usually ranges between 0 and 1 (0% and 100%). *Confidence* describes how likely the rule is to give a correct prediction. It is also in the range between 0 and 1 (0% and 100%). In addition, the algorithm of association rules uses the *support* of a rule which is the number of records or transactions which confirm the rule [Cristofor, 2002].

The classical association rule mining algorithm, Apriori has been developed for finding all association rules in a database whose support is greater than the *minimum support*
and whose confidence is greater than a minimum confidence. This algorithm uses the itemset concept which is defined as a non-empty set of items. The itemset is called frequent if its support is greater than the minimum support. A frequent itemset that is maximal is called large.

The problem of mining association rules is decomposed in the following steps:
1. Finding all combinations of items that have transaction support above minimum support.
2. Calling those combinations frequent item sets.
3. Using the frequent item sets to generate the desired rules.
4. Holding the rule which has the confidence \( c \geq \) minimum confidence and \( s \geq \) minimum support.

Within a database an association rule could be the rule which implies certain association relationships among a set of objects such as "occur together" or "one implies the other". Given a set of transactions, where each transaction is a set of literals called items an association rule is an expression of the form \( X \rightarrow Y \), where \( X \) and \( Y \) are sets of items. The intuitive meaning of such a rule is that transactions on the database which contain \( X \) tend to contain \( Y \). Mining association rules are particularly important when trying to find relevant associations among items in a given transaction within a database [Sousa et al., 1998].

The algorithm is easy to understand and apply, but it can generate large lists of rules, and only some of them are useful. The purpose of association rules is to derive or induce if-then rules that can be used to predict outcomes, and typical applications include finding relationships among products that people tend to purchase together or identifying the buying pattern of customers [Helberg, 2002].

3.1.3.4 Clustering

Clustering like segmentation identifies groups of similar cases, but it does not predict outcomes or target categories [Helberg, 2002]. For this reason clustering algorithms are also called unsupervised classification, and they process a group of physical and abstract objects into classes of similar objects. Clustering analysis supports the construction of meaningful partitions of a large set of objects based on the divide-and-conquer methodology which decomposes a large-scale system into smaller components to simplify design and implementation [Sousa et al., 1998]. It is directed at identifying clusters embedded in the data where a cluster is a collection of data objects that are similar. Similarity is expressed by
distance functions, and it is intended to maximize similarity between cases assigned to the same group and to maximize difference between groups.

An example of clustering is presented in [Helberg, 2002], and it relates to identifying customers that would make good targets for a new product marketing promotion. Because this is a new product there is not any data available about its impact on the market, however there may be some information available about customer attributes that might be used to predict whether they will be interested in the product. This knowledge may be extracted by clustering this information from the customer database and by identifying homogeneous groups of customers. Each group can then be examined to derive a group profile, which describes typical customers in that group. By comparing the profiles, segments may be identified that will respond positively to the marketing promotion.

The clustering methods are divided into:

- **Hierarchical clustering** which represents the combination of cases and clusters that are similar to each other, one pair at a time.
- **K-Means clustering** which is based on the assumption that the data falls into a known number \( K \) of clusters. This method starts by defining initial profiles called cluster centers, for the \( K \) clusters, sometimes using random values for the clustering characteristics or sometimes using dissimilar cases from the data set.

DM research has been focused on high quality and scalable clustering methods for large databases and multi-dimensional data warehouses.

### 3.1.3.5 Sequential Patterns

*Sequential patterns* are part of sequential analysis. The main goal of the sequential patterns approach is to find all sequential patterns with a pre-defined minimum support represented by a data sequence [Sousa et al., 1998]. The input data is represented by a list of sequential transactions and there is often an associated transaction-time.

This algorithm is briefly outlined below. For this algorithm, the following terminology is used:

- The length of a sequence is the number of itemsets in the sequence.
- A sequence of length \( k \) is called a \( k \)-sequence.
• The sequence formed by the concatenation of two sequences \( x \) and \( y \) is denoted as \( x.y \). The support for an itemset \( i \) is defined as the fraction of customers who bought the items in \( i \) in a single transaction. Thus the itemset \( i \) and the 1-sequence \( i \) have the same support. An itemset with minimum support is called a large itemset or \textit{litemset}.

• Each itemset in a large sequence must have minimum support. Hence, any large sequence must be a list of itemsets. In the algorithms, \( L_k \) denotes the set of all large \( k \)-sequences, and \( C_k \) the set of candidate \( k \)-sequences.

The steps of this algorithm are as follows [Agrawal, Srikant, 1995]:

1. \textit{Sort Phase}. This step implicitly converts the original transaction database into a database of sequences.

2. \textit{Litemset Phase}. In this phase we find the set of all litemsets \( L \). We are also simultaneously finding the set of all large 1-sequences, since this set is just \( I \mid I L \).

3. \textit{Transformation Phase}. This is required to repeatedly determine which of a given set of large sequences are contained in a data sequence. To make this test fast, it is necessary to transform each data sequence into an alternative representation. In a transformed data sequence, each transaction is replaced by the set of all litemsets contained in that transaction. If a transaction does not contain any litemset, it is not retained in the transformed sequence. If a data sequence does not contain any litemset, this sequence is dropped from the transformed database. However, it still contributes to the count of total number of customers. A data sequence is now represented by a list of sets of litemsets.

4. \textit{Sequence Phase}. Use the set of litemsets to find the desired sequences.

5. \textit{Maximal Phase}. Find the maximal sequences among the set of large sequences. In some algorithms this phase is combined with the sequence phase to reduce the time wasted in counting non maximal sequences.

### 3.1.3.6 Combined technique

A very important DM method is to combine different algorithms such as the knowledge extraction process based on rules with neural networks (NN) and \textit{Case Base Reasoning} (CBR) [Hui, Jha, 2000]. Case Base Learning represents the process of acquiring knowledge represented by cases using \textit{reasoning by analogy}, and for this reason it is also called CBR.
CBR is particularly useful for processing data which has complex structure, and large historical data sets are not required.

The methods of obtaining the knowledge base by generating NN models and the rule base is shown in figure 3.2. In this case, the NN model extracts knowledge used to develop a knowledge base formed by rules and NN models for classification and clustering. The types of NN used in data mining could be the supervised learning vector quantization (LVQ3) neural network and the unsupervised Kohonen self-organizing map (KSOM) neural network [Kohonen, 1997], [Hui, Jha, 2000].

Supervised learning is usually performed with feedforward NN where training patterns are composed of two parts, an input vector and an output vector, associated with the input and output nodes respectively. A training cycle consists of the following steps:
1. An input vector is presented at the input layer together with a set of desired responses, one for each node, at the output layer.
2. A forward pass is done and the errors or discrepancies, between the desired and actual response for each node in the output layer, are found.
3. These are then used to determine weight changes in the net according to the prevailing learning rule.

![Figure 3.2 Knowledge Extracting by Neural Networks and Rules](image)

However, supervised NN are usually not employed in data mining applications because of their low computational efficiency and because of the following reasons [Hruschka, Ebecken, 1998]:
- it is difficult to interpret their models;
- they are represented by many real parameters (connection weights);
they represent non-linear functions;
- the hidden unit can learn distributed representations of knowledge.

3.1.3.7 On-Line Analytical Processing

OLAP and DM are considered to be two complementary techniques for analyzing large amounts of data in databases and/or data warehousing environments. OLAP is defined as a way of performing multi-dimensional analysis on relational databases. A multi-dimensional representation related to a product family is shown in figure 3.3. According to [Chen, 1999] the dimensions or measurements along a set of dimensions such as product family, manufacturing company, location of the company etc. consist of the following conceptual tokens:
- dimension: each dimension is described by a set of attributes for example, the product dimension may consist of four attributes: the category and the industry of the product, year of its introduction and the average profit;
- measure: each of the numeric measures depends on a set of dimensions, which provide the context for that measure; the dimensions together are assumed to uniquely determine the measure and, therefore, multidimensional data provides a view of a measure as a value in the multidimensional space of dimensions;
- domain hierarchy: for example, country and state.

There are at least two different ways of combining OLAP and data mining:
1. DM can be incorporated into an existing OLAP engine (i.e. OLAP techniques and data mining techniques are inter-twined together to analyze the same data);
2. DM is applied before the OLAP analysis. In this case, the results of the data mining activity are
incorporated as intentional historical data, and used as inputs for the current data analysis.

Multi-dimensional analysis solutions are commonly referred to as OLAP solutions, and offer an extension to the relational model by providing a multi-dimensional view of the data. For example, in multi-dimensional analysis, data entities may represent different dimensions. Multi-dimensional solutions provide the ability to:

- analyze potentially large amounts of data with very fast response times;
- "slice and dice" through the data, and drill down or roll up through various dimensions as defined by the data structure;
- quickly identify trends in areas that would otherwise be missed.

Multi-dimensional data structures can be implemented with multi-dimensional databases or extended RDBMSs. Relational databases can support this structure through specific database schema intended for multi-dimensional analysis and highly indexed or summarized designs. These structures are sometimes referred to as relational OLAP (ROLAP)-based structures.

The integration of DM and OLAP is done by using complex algorithms for processing data cube and hypercube structures described in [Chen, 1999], [Bramer, 1999], [Piatetski-Shapiro, Frawley, 1991].

OLAP can be used in an integrated architecture as shown in figure 3.4, and it offers consistently rapid response to queries, regardless of the size and complexity of the database records. OLAP engine supports the user synthesized enterprise information through comparative, personalized viewing, as well as through analysis of historical and projected data in various "what-if" data model scenarios. Usually the OLAP application runs on a server as shown in figure 3.4, and the associated tools perform an intermediate stage in the DM process. However, OLAP tools are different to DM processing because of the following characteristics [Adriaans, Zantinge, 1996]:

- OLAP tools do not have learning capabilities;
- They do not create new knowledge;
- They cannot search for new solutions;
- They do not work directly on a relational database.

DM is more powerful than an OLAP tool because of the difference of multi-dimensional processing of a database and the fact that new knowledge, and hidden information can be
extracted through DM. However, as discussed above, DM can be combined with OLAP as they are two complementary techniques for analyzing large amounts of data in databases and/or data warehousing environments. At present, few commercial data mining systems have implemented OLAP technologies into their commercial tools. Sometimes an OLAP representation defines a high number of dimensions for a data cube which are not always relevant to answer the business query. Intelligent OLAP (IOLAP) identifies and visualizes only the relevant information into a multidimensional representation. IOLAP is implemented by Knowledge Extraction Engine (KXEN).

3.1.4 INTEGRATION ISSUES

The integration of KD systems and DM applications has been recently approached in some researches and projects such as [Aubrecht, Kouba, 2001], [Chua et al, 2002], [Dabbas, Chen, 2001], [Helberg, 2002], [Heinrichs, Lim, 2003], [Kamath, Cantu-Paz, 2003], [Kotásek, Zendulka, 2003], [Shahbaz, Harding, 2003], [Wang et al, 2002].

This paragraph presents integrated KD/DM systems especially for manufacturing engineering applications, and a separate section is included to deal with related projects, methodologies and standards.

At present, only a few research projects address manufacturing applications, and besides that very little research examines the integration aspects. Furthermore these research projects do not consider the enterprise reference architectures, frameworks, OMG middleware projects and standards such as Corba, and most recently MDA and CWM. Also these approaches usually deal with integration at the level of a single-dedicated application as explained below:

- Modeling and collecting semiconductor manufacturing data from multiple databases, transforming, integrating these data and retrieving reports used to monitor a factory's performance and improve its productivity. This application has been used at Motorola for wafer fabrication processing [Dabbas, Chen, 2001];
- Exploring manufacturing production processes data collected from different locations in order to generate complex models. This application intends to use XML for data integration and transformation [Shahbaz, Harding, 2003]. Using XML for data preprocessing, and exchanging mining models has been also approached at Department
of Computer Science and Engineering from Brno University of Technology, Czech Republic. This research suggests an XML framework for KDD without any domain-specification, but based on creating an ontology library for KDD [Kotásek, Zendulka, 2003].

In [Wang et al., 2002] an architecture that encompasses view models, flow models and object-oriented models has been proposed to support the modelling, design and implementation of process manufacturing systems. This architecture integrates into a common enterprise data warehouse, data related to the following concepts:

- Business Management (finance, supply, product databases and an associated knowledge base)
- Production Management (resource, production, material etc. databases)

This approach defines a combination of objects and agents for building the models of process manufacturing systems. Objects are used to describe the passive constituent entities that possess no autonomy and conduct no functions, whereas agents are used to model the active entities that possess decision-making capabilities and perform operations on objects. One important advantage of this architecture is that it provides executable models at the implementation stage, which can be directly converted into application programs coded in object-oriented languages. An information infrastructure based on the Corba standard is proposed for integrating the distributed data sources.

Therefore this approach is a very good basis for the research reported in this thesis even though it does not explicitly consider data mining, but the enterprise data warehouse is used for performance analysis and production statistics.

Chua et al. (2002) introduces the intelligent middleware for linear correlation discovery which integrates data management and data analysis tools to improve traditional data analysis in three perspectives:

1. identify an appropriate linear correlation program from a software system such as SPSS or SAS in order to perform analysis based on the semantics of a data set;
2. execute appropriate programmes contained in the data analysis packages;
3. derive useful knowledge from data analysis.

In this research, the following major problems in leveraging on the existing software for data analysis and KD/DM have been analyzed, and improvements have been attempted through using an intelligent middleware:
Several companies own separate software for data definition, data manipulation and data analysis. For example, while business data may be stored in a Microsoft Access database, necessary data analysis and mining functions may be contained in heterogeneous data analysis packages such as SPSS or SAS.

Scarce data analysis expertise: Few users have formal training with advanced data analysis methods such as data mining and on-line analytical processing (OLAP) and data management tools such as data warehouses.
• Affordability of integrated tools. While integrated prototype and commercial database/data analysis systems do exist, many companies are either unable or unwilling to adapt these products due to technical, economic, operational feasibilities and many other reasons.

• Lack of a well-accepted data analysis communication standard. To transfer data from a database to a data analysis package, it is necessary to create an export file in a format the data analysis package understands, and manually import it to the data analysis package.

The approach related to using intelligent middleware for linear correlation discovery provides and supports an integration at the level of programmes which are defined as functions and implemented in packages which define KD/DM software products.

Unified Modeling Language (UML) is not widely used for KD/DM system analysis and design and its advantages are obvious for creating object-oriented frameworks. The detailed approach is included in the following sections of this thesis. However Kamath, Cantu-Paz (2003) have used UML for the design of a parallel object-oriented DM toolkit. The UML class diagram for decision trees supports the design of the experimental system.

3.2 EXTENDED ENTERPRISE APPLICATION AREAS

3.2.1 APPLICATION VIEW

The design, development and implementation of the Distributed Knowledge Discovery and Data Mining System Embedded in Extended Enterprise is supported by a unified object-oriented framework composed by multiple views described as follows:

• **Information and Data Views** which provide the description of all enterprise objects, data models and the metadata concept used for a generic data mining system design. These views are very important and are mainly based on the metadata concept defined by OMG's Common Warehouse MetaModel. The design and development issues according to these views are presented in chapters 5 and 7.

• **Knowledge and Mining Views** are introduced in this research, to the reference architectures for enterprise modeling and enterprise integration such as CIMOSA, ARIS, PERA and GERAM. These views are explained in detail in chapter 5 and they define high level architectures of generic KD/DM systems. The high level
architectures and associated methodologies offer a basis to integrate and adapt or specialize a KD/DM system to suit the requirements and environment of specific enterprises. However this approach does not facilitate concrete implementation solutions.

- **Object-Oriented View** according to OMG middleware standards and projects such as UML, Corba, MDA and CWM. This enables integration with existing systems that comply with these standards. This view is approached in detail in chapters 5 and 6.

- **Application View** which provides an environment for application development and implementation. According to this view the KD/DM system should be applied to a range of existing systems [Ayers, 2001], [Berson et al, 2000], [Groth, 2000], [Makridakis, Wheelwright, 1989],[Shim, 2000], [Wang, 1999]:
  1. Supply Chain Management (SCM);
  2. Customer Relationship Management (CRM);
  3. Enterprise Resource Planning (ERP);
  4. Decision making support;
  5. Market Analysis and Product Specification;
  6. E-business including e-transaction and e-commerce applications;
  7. Production and Inventory;
  8. Intelligent Process Monitoring and Control;

This view depends on the domain specifications and requirements of individual systems and it is therefore domain-dependent. It is also supported by the unified object-oriented framework for a data mining system modeling and design which is presented in detail in chapter 7.

- **Industry Standard View** which incorporates the models processed, shared and exchanged using Extensible Markup Language (XML) and Predictive Modeling Markup Language (PMML). This view is not part of this research, but the intention is this should support networked, distributed systems such as the approach presented in [Shahbaz, Harding, 2003]. The brief outline of this view shows the modalities by which the approach included in this thesis could be applied at the implementation level for future application development.
The literature reviewed in the remainder of section 3.2 all provides relevant background for the Application View and the range of existing systems which should be supported by this view. This paragraph is focused on the Application View and provides the necessary background information for class definitions such as ApplicationInputSpecification EE_ApplicationSpecification included in the Extended Enterprise Common Warehouse MetaModel (EE_CWM) which is presented in chapter 6. The general CWM description and some specifications are presented in the following chapter. The remainder of this chapter deals in detail with SCM, CRM and ERP as these are considered to be of vital importance for integrated enterprise architecture definitions that include DM. These applications are presented in relation to extended enterprise approaches.

Sometimes from a logical viewpoint, concepts related to these applications overlap or at least are related to each other, this is especially true for the SCM and ERP areas.

For example CRM includes a large number of functions and attributes for both e-commerce and supply chain because a customer demand can be considered to be entered into a SCM system through a CRM application [Ayers, 2001]. For an extended enterprise it is better to consider all these areas because a particular company may focus on implementing an ERP system, whilst others concentrate on SCM. In this way they may generate redundant and identical data which needs to be pre-processed or aggregated in a data warehouse. Also some companies, particularly those with technical products may use Product Data Management (PDM) applications in order to improve their supply chains because a collaborative PDM system has the capabilities not only to define product components but also to share this design information along the supply chain.

Figure 3.4 shows the general structure of a data mining and OLAP system which processes enterprise databases composed of data generated from existing ERP, CRM and SCM systems. The system includes data warehouse processing which is a repository for data organized in a different format that is suitable for database queries. A data warehouse is built from the operational data related to the above applications for fast retrieval, and efficient analysis. The single purpose data warehouse is defined as a data mart.
3.2.2 SUPPLY CHAIN MANAGEMENT

The supply chain is a complex area defined as life cycle processes formed by physical, information, financial and knowledge flows whose purpose is to satisfy end-user requirements with products and services from multiple, linked sources. Supply chain management deals with the design, maintenance and operation of supply chain processes necessary to accomplish end-user needs [Ayers, 2001], [Martel, Vankatadri, 1999] [Huang et al., 2002].

Supply chain management is also the logical integration of strategy definitions and formulation, marketing, operations and distribution as well as inventory and production planning. Therefore the consideration of supply chain approaches from the engineering and business perspectives results in the graphical representation depicted in figure 3.5 [Huang et al., 2002].

![Supply Chain Approaches](image)

Figure 3.5 Supply Chain Approaches based on [Huang et al., 2002]
The traditional supply chain management systems were faced with several problems which are briefly outlined below. Some of them have been solved by the definition of virtual and extended enterprises especially based on multi-agents systems, and for others an optimal solution is to use intelligent data acquisition and processing complemented with data warehouse and data mining systems as part of an extended enterprise architecture.

These problems were as follows [Ayers, 2001], [Martel, Vankatadri, 1999], [Huang et al., 2002]:

• "Bullwhip effect" which represents the discrepancy between some of the parameters of supply chain systems such as the number of products to be manufactured, the quantity of raw material needed and the market requirements. This is also called supply chain distortions or oscillations and it causes unstable supply chains. Even in distributed and networked supply chains the bullwhip effect is caused by the lack of proper coordination and communication between suppliers, but it can be reduced and controlled using multi-agent systems.

• Stand-alone manufacturing enterprises could not form strategic alliances with their suppliers.

• Distribution was not recognized for its important role in cutting down costs and lead-time and increasing the availability of finished products. It could also be used to delay product differentiation till that stage so that customization could be achieved and over-stocking and stock-outs could be avoided.

In the extended enterprise, an integrated supply chain is defined globally, and includes distributed networks of suppliers with the purpose of achieving specific long-term business objectives. An aim for data warehouse and mining should be to reduce the bullwhip effect and improve the business performances of a distributed supply chain system, since they might be used to remove duplication and inconsistencies in the data, and reduce ambiguity. To date, the modalities of smoothing out the oscillations of a supply chain especially through data mining have not been researched. However these modalities are outlined as follows:

• Finding and understanding customers and suppliers patterns and then, collaboratively processing these to improve information quality and to discover knowledge related to compressing cycle times throughout the entire process.

• Finding and understanding product demand patterns at each stage of the supply chain.
• Monitoring the actual demand for a product to as near a real time basis as possible through temporal data mining.
• Minimizing or eliminating information queues that create information flow delays.
• Including CRM applications.

Several projects related to integrated supply chains based on multi-agent systems have been developed and implemented. The most important are briefly introduced below [Fox, 1996], [Frayret et al., 2001], [Montreuil et al., 2000]:

• **TOVE** which had the aims of developing a shared representation for supply chain knowledge, identifying a corresponding decomposition of supply chain functions and encapsulating these into agents. This project also developed an incremental model of problem solving for each functional agent in order to provide rapid responses to unplanned events. Protocols, strategies and tools for the communication of information, coordination of decisions, and management of change within multi-agent environments are also included.

• **NetMan** project had the main objective of developing a computer-based infrastructure, to support the operations of a network of business units or entities defined for networked manufacturing enterprises. The infrastructure is implemented based on multi-agent systems. This project defined a network-oriented organizational strategy according to which a manufacturing business dynamically organizes its operations through the configuration and activation of a distributed network of interdependent responsible manufacturing centers. It offered a framework and methodology for designing hierarchical networks of business entities in order to enable agile manufacturing. Business units or entities are developed as NetMan centres which create a NetMan network. It also addressed information sharing on a manufacturing network and its value creation process, and it investigated networking strategies where business relationships are characterized by the information exchanged between firms on demand, price and capacity. The project uses economic and management theories such as business rules-of-the-game. From a technical perspective of the development and implementation of business entities, the results of this research suggest that higher networking flexibility is achieved when firms favour tight web relationships. However despite the amount of data shared, information
exchanged and creation of business models, the project has not considered knowledge discovery and data mining or web mining.

In the context of this research the main aspect is that these projects do not address knowledge discovery and data mining even though distributed supply chain management systems create large amounts of data and information. Furthermore as there is a growing interest in e-business, the implication should be the creation of e-supply chains and the exploration of these should be done through web mining. However, this area of work has not been reviewed here as research in these areas is more focused on e-commerce and related topics which are beyond the objectives of the research reported in the thesis.

3.2.3 CUSTOMER RELATIONSHIP MANAGEMENT

*Customer Relationship Management (CRM)* is a comprehensive approach and concept, which provides seamless integration of several areas of business that are related to customer modeling, customer behaviour and interactions. These areas are as follows: marketing, sales, customer service and a support environment for the integration of human resources, processes and technologies. These areas take full advantage of the Internet capabilities, databases and data mining. Nowadays, CRM has become very important because the competitive markets are increasingly saturated and competitive, and the marketing model is changing from the product-centered stage to the customer-centered stage [Xu et al., 2002]. Applications of data mining in CRM and marketing are classic examples of using these techniques, and a traditional system deals with customer profiling [Adriaans, Zantinge, 1996]. However the application of data mining in CRM is still an evolving topic under research and investigation. Attempts to apply more sophisticated discovery and mining techniques for CRM and e-commerce are described in [Rygielski et al., 2002]. This article presents two data mining techniques: Chi-square Automatic Interaction Detection (CHAID) and Neural Networks and their application for CRM, Business-to-business (B2B) and Business-to-Consumer (B2C).

Therefore CRM is the process by which a company manages its interactions with customers, and CRM systems and associated methodologies are applications that define the interface between a company and its customers [Ayers, 2001], [Berson et al., 2000]. CRM is the core activity of e-business, and in the framework of manufacturing enterprises it could be
integrated with SCM and ERP systems as well as market applications. Several companies complement their CRM and ERP applications with other Business Intelligence (BI) systems. These are defined as more broad applications that implement additional systems such as financial analysis and accounting. According to the DM Review Newsletter at: http://www.dmreview.com/, BI is a global concept describing the processes, techniques and support tools that facilitate decision-making and customer relationships management systems. CRM systems are composed of the following sub-systems which use data mining as explained below [Berson et al., 2000], [Delmater, Hancock, 2001], [Pyle, 2003]:

- **Customer Profiling** which is the process of discovering consistent patterns within customer or prospect data which provides information and knowledge that can be used to make business decisions and take appropriate actions. For example, customer profiling might be used to identify which prospects are most likely to respond favorably to a certain type of selling campaign or offer. Advanced customer profiling can use predictive data mining in order to identify which prospects are most likely to make a particular type of purchase. Cluster analysis is used for grouping customers and products based on different criteria. This process is called **segmentation**; it could also be directed at acquiring new customers. Customer profiling is mainly divided into **customer acquisition** and **customer retention** which may also be identified as customer loyalty. Customer profiling is also required for finding out additional selling, and cross-selling opportunities. **Cross-selling** is the process of offering new products and services to existing customers. Customer profiling is directed at understanding the customer and creating a long-term, lasting relationship with loyal customers and is defined as one-to-one marketing.

- **Customer Profitability** uses data mining in order to understand, optimize and improve it. Customer profitability is logically linked to customer loyalty.

- **Customer Segmentation** applies data mining, and especially clustering to discover discrete segments in the customer database by considering additional variables in addition to applying traditional statistical analysis.

- **Predicting Customer Behaviour** includes churning which represents the process of moving from one company to another.

- **Customer Interaction Center** represents the kernel of a CRM system and is achieved using knowledge discovery, data mining and feature extraction algorithms.
Additional details about CRM are presented in chapter 7 in order to support the analysis and modeling of a CRM system linked to knowledge discovery and data mining processes.

Kurt Thearling (2001) has defined the main data sources for CRM that are also useful for data mining applications. These data inputs are as follows [Thearling, 2001]:

- **Structured sources** such as relational data bases and multi-dimensional data representation;
- **Unstructured sources** such as web logs and rich content files i.e. images, sound and video.

Harding et al. (2001) approached the accomplishment of customer requirements using identified needs that were structured as a set of variables related to a product. The attributes are first fuzzified according to the sets of linguistic variables of the relevant model variables. The attributes can then be modified by the fuzzy set boundaries, as appropriate, in order to fully elaborate on the customer attributes. The knowledge about customer attributes, requirements, product attributes and their relationships is included in a fuzzy knowledge system which could be explored using data mining techniques such as classification.

Consideration of how data mining may be applied to CRM in particular, makes it possible to draw the following conclusions.

- Marketing and CRM are successful areas of application of data mining, although at first sight these areas may seem not to be directly linked to manufacturing industrial environments. However the new approaches of extended enterprises are more based on business modeling and strategic optimization than on production processes and this is especially useful in order to gain competitive advantage.
- Several specific applications of data mining for CRM can be developed and implemented but this is not the focus of the research reported in this thesis. Rather the purpose of this section of the thesis is to generalize and to find common elements for CRM, ERP and SCM in order to define associated classes to describe these systems in the extended enterprise.
- An enterprise data warehouse is a critical component for implementing a successful CRM strategy.
- Extraction of hidden predictive information and discovering knowledge from large enterprise databases can identify valuable customers, predict their future behaviours, and enable a company to make proactive, knowledge-driven decisions.
• The automated, feature oriented analyses made possible by data mining move beyond the identification of past events typically provided by history-oriented tools such as decision support systems.
• Data mining systems should answer business questions that in the past were too time-consuming to pursue, and if possible enable appropriate actions to be taken that are based on these identified answers.

3.2.4 ENTERPRISE RESOURCE PLANNING

Enterprise Resource Planning (ERP) systems are complex programmes that link all the enterprise functions and associated departments such as manufacturing, finance, sales and human resources. An ERP system also provides the source data for different analysis such as qualitative analysis for production planning or sales forecasting [Ayers, 2001]. Therefore an ERP system allows an enterprise to integrate its resources. Furthermore over more than 30 years manufacturing companies have maintained large databases to manage their data and information about their processes, products and human resources. Systems such as Material Resource Planning (MRP) were critical components, but as the potential for information storage and processing increased, the shortcomings of MRP and MRPII could be tackled. In addition the need for integration between all functions within the business led to enterprise-wide resource planning which attempts to integrate CRM applications and other programmes for accounting and sales management [Gupta, 2000]. In [Botta-Genoulaz et al., 2001] a comprehensive analysis of ERP systems is presented for agile enterprises which must be able to continuously improve their management, and quickly react to changes in their business environment. The paper makes the link between enterprise architectures and modeling methodologies, and ERP systems development and implementation with the aim of standardization of ERP solutions. It also includes an analysis of software systems supporting ERP implementations such as Baan, J.D.Edwards, Oracle Application, PeopleSoft and SAP.

ERP systems attempt to integrate the old standalone computer systems in accounting, finance, human resources, manufacturing and distribution, and replace them with a single unified software program divided into software modules that roughly approximate the old standalone systems. However, in practice particular sectors such as finance, manufacturing and the warehouse still use their own software, except now the software is linked together.
Most vendors of ERP software provide solutions that are flexible enough so that it is possible to install some modules without buying the whole package. Many companies, for example, will just install an ERP finance or the human resource module [Koch et al., 1999].

Therefore the need to apply and integrate data mining in ERP systems is obvious, and the premises for that have been created. However, in practice, the following situations occur:

- Knowledge discovery and data mining are not explicitly considered in existing ERP solutions;
- ERP solutions include data warehouse, data conversion, integration, analysis and elements of evolutionary programming, but they are not defined for data mining;
- A DM system increases the cost of ERP systems and the processing speed, and this may lower information delivery performance.

Generally, there are several difficulties in implementing an ERP system which can be increased by the adoption of a data mining system without a specific methodology. The shortcomings of implementing an ERP solution are outlined below [Botta-Genoulaz et al., 2001], [Gupta, 2000]:

- Organizational problems such as the organization not being well prepared for ERP implementation. This includes lack of clarity in the requirements, inadequate business processes re-engineering etc.
- Human problems such as lack of commitment and motivation from the top management, communication errors, failures in the implementation team leaders or members, internal staff inadequacy, training shortcomings, change management underestimated etc.
- Technical problems such as database integration;
- Problems adapting the generic software to the specific needs of the enterprise. ERP vendors have been required to provide tools and utilities that will allow the enterprise's in-house IT personnel or independent consulting firms to customize the software. Actually, customization and implementation of ERP software has become an industry in its own right.
3.3 KNOWLEDGE DISCOVERY PROJECTS, METHODOLOGIES AND STANDARDS

3.3.1 INTRODUCTION

The best known, used and referred projects related to a methodological and standardized approach of knowledge discovery in databases (KDD) and data mining (DM) are as follows:

- **CRISP-DM™** (Cross-Industry Standard Process for Data Mining) which has developed an industry- and tool-neutral DM methodology. The project started from the KD processes applied in industry and intended to respond directly to user requirements. This project has also defined and validated a DM process that is applicable in diverse industry sectors, and it attempts to make any DM project faster, cheaper, reliable and manageable [Helberg, 2002].

- **SolEuNet** (Data Mining and Decision Support for Business Competitiveness: A European Virtual Enterprise) The main aim of this project is to apply the technologies of DM and Decision Support (DS) in order to enhance efficiency, effectiveness and quality of operations in business, industry and public services. A virtual enterprise model has been proposed as a dynamic problem-solving link between advanced DM and DS systems. The project has emphasized data pre-processing methods and associated software systems such as Sumatra, which is a scripting language especially for data transformation. It has also dealt with model exchanging techniques based on Predictive Modeling Markup Language (PMML). This project provides some insights into the possibilities of integrating DM and DS, with Expert Systems (ES) establishing a strong connection between these apparently different research areas [Lavrac, 2001], [Aubrecht, Kouba, 2001], [SolEUNet, 1999].

- **Kensington Enterprise Data Mining** (Imperial College, Department of Computing, London, UK) project has developed Kensington Discovery Edition (KDE) which is an enterprise-wide platform that supports entire processes of KD, including dynamic information integration, large-scale interactive discovery and knowledge management. KDE provides an extensible layer of generic and scalable analytical components, a uniform framework of dynamic knowledge management for verifying
and warehousing discovered knowledge as well as a wide range of deployment tools for generating analytical applications from well verified discovery processes.

However most of the DM software systems are focused on DM technology rather than on ease-of-use, integration, scalability and portability. Several attempts have been made by standards organizations and consortia to agree on a standardized way to use DM together with data management products like SQL, object-oriented databases and data warehouses. Figure 3.6 shows a graphical overview of the main data mining standards and the relation between them. This figure also illustrates that CRISP-DM methodology does not relate to any of these standards. These standards are as follows [Thess, 2002, 2003]:

1. ISO/IEC JTC1 SC32 WG4: SQL/MM Part 6: Data Mining. A collection of SQL user-defined types and routines to compute and apply data mining models.

2. Data Mining Group (DMG): Predictive Model Markup Language (PMML), which is an open standard format based on XML specification for exchanging DM models between applications running on different platforms.

3. OMG: Common Warehouse Metamodel (CWM): Data mining - A UML/XML based specification for data mining metadata as described in the following chapter.

4. Microsoft-OLE DB for DM is a major step toward the standardization of DM primitives, and it defines the DM object model for relational databases.

5. Oracle9i DM is an extension to Oracle9i Database Enterprise Edition that embeds DM algorithms for classifications, predictions and association rules. All models and functions are accessible through Java-based Application Programming Interfaces (API) called Java Data Mining (JDM).

6. SEMMA (Select, Explore, Modify, Model, Assess) methodology elaborated by SAS Institute Inc. To be applied successfully, the SAS data mining solution called Enterprise Miner, must be interpreted as a process rather than as a set of tools. It is also a methodology that clarifies and applies exploratory statistical and visualization techniques, selects and transforms the most significant predictive variables, models the variables to predict outcomes and confirm the accuracy of a model. The steps of the SEMMA process are as follows:
   - **Sample** the data by extracting a portion of a large data set containing enough significant information, but having optimal dimension to be manipulated quickly.
> *Explore* the data by searching for unanticipated trends and anomalies in order to understanding ideas and the trends of the data set.

> *Modify* the data by creating, selecting and transforming the variables to focus the model selection process.

> *Model* the data by allowing the software to search automatically for a combination of data that reliably predicts a desired outcome.

> *Assess* the data by evaluating the usefulness and reliability of the findings from the data mining process.

---

**Figure 3.6 Overview of Data Mining Standards [Thess, 2002, 2003]**

### 3.3.2 CRITICAL OVERVIEW OF THE PROJECTS

*CRISP-DM™* contains a high level methodology which includes the following phases graphically represented in figure 3.7 [Chapman et al., 1999, 2000], [Helberg, 2002]:

**Business Understanding**

This initial phase focuses on understanding the project objectives and requirements from a business perspective and then converting this knowledge into a data mining problem definition and a preliminary plan designed to achieve the objectives.
Data Understanding

The data understanding phase starts with an initial data collection and proceeds with activities in order to identify data quality problems, to audit the data set, or to detect interesting subsets to form hypotheses for hidden information.

Data Preparation

The data preparation phase covers all activities required to construct the final data set (data that will be included into the modeling tool(s)) from the initial raw data. Data preparation tasks are likely to be performed multiple times, and not in any prescribed order. Tasks include table, record and attribute selection as well as transformation and cleaning of data for modeling tools.

Modeling

In this phase, various modeling techniques are selected and applied, and their parameters are calibrated to optimal values. Typically, there are several techniques for the same DM problem type. Some techniques have specific requirements on the form of data. Therefore, stepping back to the data preparation phase is often needed.
Evaluation
At this stage in the project a model (or models) has been built that appear to have high quality, from a data analysis perspective. Before proceeding to final deployment of the model, it is important to more thoroughly evaluate the model and review the steps executed to construct the model, to be certain it properly achieves the business objectives. The main objective is to determine if there is some important business issue that has not been sufficiently considered. At the end of this phase, a decision on the use of the data mining results should be reached.

Deployment
Creation of the model is generally not the end of the project even if the purpose of the model is to increase knowledge of the data, the knowledge gained will need to be organized and presented in a way that the customer can use it. Depending on the requirements, the deployment phase can be as simple as generating a report or as complex as implementing a repeatable data mining process. In many cases it will be the customer, not the data analyst, who will carry out the deployment steps. However, even if the analyst will not carry out the deployment phase it is important for the customer to understand up front what actions will need to be carried out in order to actually make use of the created models.

CRISP-DM™ is supported by SPSS, and its module for DM is called Clementine. The general methodology is defined as an iterative process, and it attempts to produce a standard. The defined phases need to be refined in order to apply the methodology for the extended enterprise areas of applications that were presented in section 3.2.

According to CRISP-DM™ the essential activities for a DM project relate to the identified areas of applications that have been established in the extended enterprise, and they are presented in figure 3.9 [Neaga, Harding, 2002]. The main weakness of CRISP-DM™ is the high level approach which does not offer enough details for implementation solutions of a defined project. In practice CRISP-DM™ is a general roadmap for further detailed instructions according to the specification of application areas. Furthermore this project does not make any link between an algorithm and its suitability for a specific area of application. Each domain of application also needs a specific modeling methodology if DM is considered to be domain-dependent. The pre-processing phase is also not described in sufficient detail.

SolEuNet project emphasizes the data pre-processing phase and develops a novel software architecture that integrates DM and DS support systems in order to implement rapid
remote collaborative data mining projects. The architecture includes Sumatra, a scripting language which specifies data transformation tasks. The integration of DM models is defined at the level of representation, that is common for all models, and is based on PMML and the following models are defined [SolEUNet, 1999]:

1) data mining models;
2) decision support models;
3) data transformation models.

Figure 3.8 KDD/DM Processing Defined by SolEuNet Project [SolEUNet, 1999]

Knowledge discovery in databases and data mining (KDD/DM) process defined by SolEuNet is shown in figure 3 8. This figure illustrates that the DM algorithm generates a model which can be stored in PMML format. The PMML model can be used by other tools, the main aim of which is to visualize the model or to apply the discovered knowledge. Therefore PMML language is used to support the exchange of DM models between different applications and visualization tools. However this project does not include many applications in engineering as examples.

A step forward was made by Kensington Enterprise Data Mining project which developed a distributed platform that supports intelligent information processing, including dynamic information integration, large-scale interactive discovery and knowledge management embedded in a generic Enterprise Data Analysis framework. Knowledge Discovery Edition (KDE) provides an extensible layer of generic and scalable analytical components, a uniform framework of dynamic knowledge management for verifying and
warehousing discovered knowledge as well as a wide range of deployment tools for generating analytical applications from validated discovery processes. The novelty of this project is defining and implementing distributed data mining by using Java and Enterprise JavaBeans. However this architecture does not have any correlation with other projects and standards in both knowledge discovery and industry. Therefore it is a domain-independent project, and it lacks a methodology for its application.

Figure 3.9 Essential Activities in a Data Mining Project

In order to fill the gap between DM theory and its application the research reported in this thesis proposes a domain-dependent methodology because using the algorithms the following situations appear:

1) It is almost impossible to interpret the results of mining a large database without any domain specification for the applied algorithms;

2) If DM is considered to be domain-dependent, the modeling should follow the information modeling requirements of the application class. In this situation for every application class it should be possible to define or simulate generic data sets in order to test the application. Additional data sets could be defined for training and execution.

It has been concluded that a methodology for intelligent data analysis should include:

1) Understanding the domain involves formation of the picture of what exists in the domain, and gathering the relevant facts about the domain;

2) Exploring the problem space;

3) Exploring the solution space;
4) Specifying the implementation methods or algorithms;
5) Mining the data which includes preparing, surveying and modeling the data;
6) Creating mining models.

A methodological approach of KD/DM in manufacturing industry is especially, based on object-oriented modeling techniques and it defines a unified framework for DM systems following reference architectures and environments such as CIMOSA, RM-ODP and MDA. The KDD generic architecture incorporates DM and OMG standards. This could be called KD/DM unified system modeling approach and it represents a contribution of this research which is detailed in chapter 7.

3.4 SUMMARY

This chapter presents the literature review, revealing the gaps in existing knowledge and research projects, and introducing the elements of a new approach and of identified solutions which will be detailed further.

Therefore this chapter deals with state-of-art of KD and DM theory, basic algorithms and their applications in extended enterprise. In addition the integration issues and KD/DM project are described.

Despite the continuous DM system development and upgrade, many integrating aspects remain to be solved. The research described in this thesis proposes a multiple views approach of DM system development which uses a unified object-oriented framework to facilitate the proposal of different system architectures. In order to do that this chapter includes a comprehensive presentation of Supply Chain, Customer Relation Management and Enterprise Resource Planning systems which have been identified as major areas for application of DM in manufacturing industry.
Chapter 4

RESEARCH BACKGROUND

4.1 ARCHITECTURE AND FRAMEWORK CONCEPTS

Object Management Group (OMG) produces and maintains software specifications for interoperable enterprise and Internet applications [OMG, 2003]. With well-established standards covering software production from design, through development, to deployment and maintenance. OMG supports a full-lifecycle approach to enterprise integration. Modeling standards include Unified Modeling Language (UML), XML Metadata Interchange (XMI) and Common Warehouse MetaModel (CWM). Common Object Request Broker Architecture (CORBA) is a standard open platform which has associated services, facilities and a language for interfaces definition, IDL. There are also several profiles, additional standards and models widely used together with UML for complex system analysis, modeling, design and architecture definitions. However architecture definition and development are very complicated tasks [Quatrani, 2000]. Furthermore research communities have not accepted a standard, unique and conceptual definition for the term architecture that is valid from the perspective of computer engineering as well as for enterprise architecture research. According to [Booch et al., 1999] the software architecture is the set of significant decisions about the organization of a software system, the selection of the structural elements and their interfaces by which the system is composed, together with their behaviour as specified in the collaborations among those elements, the composition of these structural and behavioural elements into progressively larger subsystems, and the architectural style that guides this organization and associated elements and their interfaces, their collaborations and their composition. According to this definition there is no doubt about using object modeling techniques to describe a software architecture that includes a comprehensive software and hardware structure. This is also emphasized in the definition given by Quatrani (2000) who takes into consideration the decision making process in architecture options in the development phase. An architecture is also defined as the logical and physical structure of a system, forged by all strategic and tactical design decisions applied during development [Quatrani, 2000].
In order to reduce the effort required to build an application a framework can be used. Usually a framework defines a partial design and implementation of a system in a given domain. Therefore a framework is an incomplete system or a generic application which can be tailored to create complete applications. Frameworks are generally used and developed when several similar applications will be developed. A framework is also an architecture for the assembly of software components because it usually implements the common elements between applications and interfaces.

According to [Gurp, Bosch, 2000] based on IBM specification for building object-oriented frameworks, they are grouped into following three categories:

- **Application frameworks** aim to provide a full range of functionality typically needed in an application. This functionality usually involves GUI (Graphic User Interfaces), documents, databases etc. An example of an application framework is MFC (Microsoft Foundation Classes). MFC is used to build applications for MS Windows. Another application framework is JFC (Java Foundation Classes). The latter is interesting from an Object Oriented (OO) design point of view since it incorporates many ideas about what an OO framework should look like. Many design patterns were used in this framework.

- **Domain frameworks** can be helpful to implement programs for a certain domain. The term domain framework is used to denote frameworks for specific domains such as manufacturing, finance and banking. Domain specific software usually has to be tailored for a company or developed from scratch. Frameworks can help reduce the amount of work that needs to be done to implement such applications. This allows companies to make higher quality software for their domain while reducing the time to market.

- **Support frameworks**. Support frameworks typically address very specific, computer related domains such as memory management or file systems. Support for these kinds of domains is necessary to simplify program development. Support frameworks are typically used in conjunction with domain and/or application frameworks.

In the following chapter, the differences between a framework and a reference architecture for enterprises and their information systems are explained.

This research has adopted the solution to define a unified object-oriented framework for distributed DM systems fully described in UML and using the Common Warehouse...
MetaModel which is defined at the heart of the Model-Driven Architecture as shown in figure 4.1. The framework is domain-dependent and has major constraints which will be explained in chapter 7.

The unified framework for KD/DM applications could support several common subsystems for business modeling, description of manufacturing models and data mining as presented in detail in chapters 5 and 7.

4.2 PRINCIPLES OF MODELING AND ASSOCIATED TOOLS

UML is defined as a standard graphical language for specifying, visualizing, constructing and documenting the artifacts of software systems as well as for business modeling and other non-software systems [OMG, 2001b]. For the development and implementation of an advanced enterprise solution, UML has an important role to accomplish the following requirements:

- Defining enterprise system frameworks and architectures that underpin several applications and are platform-independent;
- Strategic reuse and systems engineering capabilities;
- Enterprise Application Integration (EAI).

It also offers a direct support for the representation of the conceptual system structure including business processes description and system functions as well as concrete development components represented by generated C++/Java classes, database schemas, reusable software modules etc. [Eriksson, Penker, 1998], [Fowler, Scott, 1997], [Quatrani, 2000], [OMG, 2001b], [Stevens, Pooley, 2000].

The main reasons for using modeling techniques are as follows:

- to communicate the desired structure and behaviour of the system;
- to visualize, control and deploy the architecture of a system;
- to comprehensively understand the system and expose opportunities for optimal design and reuse;
- to manage risk.

The choice of models and diagrams definitions has a deep influence upon how a problem is attacked and how a corresponding solution is shaped. Abstraction, the focus on relevant
details while ignoring others, is a key to system learning and communication. Because of this [OMG, 2001b]:

- Every complex system is best approached through a small set of nearly independent views of a model. Usually a single view is not sufficient.
- Every model may be expressed at different levels of complexity.
- The best models are connected to reality.

In terms of the views of a model, UML includes the following graphical diagrams:

- **use case and class diagrams** which define the functionality and the logical view of the system;
- **statechart and activity diagrams** which define the behaviour of the system;
- **sequence and collaboration diagrams** which define the interaction within the system;
- **component and deployment diagrams** which define the implementation issues.

The most important tools which implement UML are Rational Rose, Poseidon and Eclipse. There is no doubt that the most commonly used is *Rational Rose* which enhances a visual modeling approach providing support for component-based and controlled iterative development. Besides the implementation of UML, Rational Rose uses Component Object Modeling (COM), Object Modeling Technique (OMT) and Booch'93 method. It also has the following main capabilities [Quatrani, 2000], [Rational Co., 2000]:

- Identify and design business objects and then map them to software components;
- Partition services across a multi-tier model and/or architecture;
- Design a distributed object architecture;
- Code generation directly from the model;
- Use reverse engineering to create models from existing components and applications;
- Use round-trip engineering facilities in order to keep the designs synchronized with code.

Rational Rose extends visual modeling for dynamic behaviours such as business requirements, business scenarios analysis with sequence and collaboration diagrams, state modeling and additional code generation capabilities for CORBA/IDL along with scripting language definition.

*Rational Quality Architect* is a collection of integrated tools for testing middleware components built with technologies such as Enterprise Java Beans (EJB) and Component Object Model (COM), using a script language [Rational Co., 2001].
The Professional Edition of Poseidon for UML has more facilities including flexible code generation, round-trip engineering, Java/Jar file import and HTML documentation generation. One of the most valuable features of Poseidon for UML is its code generation and the Professional Edition gives a full access to it. The code generation mechanism is based on a template technology, which defines the syntax of the outcome which can cover Java, C++, XML, HTML etc. Also sophisticated round-trip engineering for Java allows you to read in existing Java code and generate a UML model for it or to continuously synchronize your code with the model. It is possible to change the generated code or redesign the model without losing the consistency of the model. Using the Jar file import capability it is possible to read existing libraries and use these in your models. The HTML documentation generator allows the export of your models to an HTML format and enables it to be shared with others over the web or intranet. This outcome is similar to Javadoc, but includes all the information related to the UML model including the diagrams. The professional edition can import Rational Rose files (mdl).

UML has an important function in the definition of OMG middleware projects especially, Model-Driven Architecture (MDA). The basic idea behind the MDA structure is the development of a base Platform-Independent Model (PIM) that maps to one or more Platform-Specific Models (PSM) which can be implemented. These models are defined in UML. Also at the core of MDA is a combination of UML, Meta-Object Facility (MOF) and Common Warehouse MetaModel (CWM). The role of UML in CWM is described below [OMG, 2000]:

- It is used as MOF which is equivalent to a meta-metamodel. UML or the part that corresponds to the MOF Model, UML Notation and OCL (Object Constraint Language) are used as the modeling language, graphical notation and constraint language, respectively, for defining and representing the CWM.
- It is used as the Foundation metamodel within the definition of the CWM. The Foundation, Behavioural_Elements and Model_Management packages of UML are used for the design of the CWM Foundation from which other metamodels inherit classes and associations.
- It is used as the object-oriented metamodel. UML, especially its Foundation package, is relied on for representing object-oriented data resources.
4.3 MODEL-DRIVEN ARCHITECTURE

*Model-Driven Architecture* (MDA) is the latest OMG initiative, developed in order to support companies and organizations to integrate new applications into existing systems.

MDA has been defined by OMG as a high-level abstract architecture based on UML methodology and existing profiles. The MDA core is supported by Meta-Object Facility (MOF), Common Object Request Broker Architecture (CORBA), XMI/XML and Common Warehouse Metamodel (CWM) as described below [OMG, 2001a]:

- **CORBA** is a standard middleware platform for enterprise distributed software architectures and its combination with JAVA™ and Enterprise JavaBeans (EJB) offers performant solutions for distributed DM systems as presented in chapters 5, 6 and 9. However, several enterprises have chosen different middleware solutions.

- **MOF** provides a standard repository for the model and defines a structure which facilitates team development and implementation because everyone will view the
model in a standard format. MOF could be considered as a library of models or a repository using template definitions.

- CWM is a data warehouse standard for data repositories, database schema definitions and it includes OLAP and data mining models.

- XML Metadata Interchange (XMI) is a mapping which expresses UML models in XML and allows them to be exchanged in the extended enterprise.

MDA has a structure formed by layers as shown in figure 4.1. This architecture has the following advantages:

- Integrating already existing software systems developed using UML and associated profiles with the systems under development and their future versions;

- Incorporating the previous OMG standards and creating an open framework to add new definitions, support languages and specifications;

- Flexibility regarding the constant changing environments;

- Increasing the lifetime of systems, lowering maintenance cost and raising ROI.

The relation between MDA, RM-ODP and other architectures defined in chapter 5 such as CIMOSA is explained below [OMG, 2001a]:

- The term abstraction is used in the MDA in the same sense as it is defined in the RM-ODP Part 2, i.e. the suppression of irrelevant details.

- A model that is based on specific abstraction criteria is often referred to as a model from the viewpoint defined by those criteria, or as a view of the system.

- While the MDA provides guidelines on the architecture of models, the choice of viewpoints to be used in a system specification is a modeling choice.

- In the MDA, the term platform is used to refer to technological and engineering views that are irrelevant to the fundamental functionality of a software component.

- UML provides an important modeling element that is relevant for separating viewpoints, levels of abstraction and refinements as packages that correspond to particular viewpoints of the system, and hence another package will correspond to a different viewpoint. At the level of diagrams, Use Cases Diagrams describe the functionality of the system and the Class Diagrams model a static system view defined as its logical architecture.
MDA separates the platform-independent application model from the implementation solutions. The MDA approach considers the model to be at the heart of development, and aims to implement platform-independent models (PIMs). When applied to these models, model transformation and code generation tools produce architecture and platform-specific models (PSMs) and code dedicated to the different target platforms.

4.4 Common Warehouse MetaModel

Common Warehouse MetaModel (CWM) defines a generic model that enables data exchange and sharing across databases or even data warehouses across enterprises [OMG, 2000], [Poole et al., 2002]. It is a new open industry standard recently adopted by Oracle, SAS and other software companies which are progressing towards incorporating this standard in their implementation. CWM is a common meta model which should be independent of any specific data warehouse implementation, but which becomes domain specific in association with domain specifications. The metamodel is developing as a set of packages which describes metadata. Metadata is data describing data or information about data, and generally, it comprises a description of information structures and models. Metadata is very important in achieving large-scale integration and interoperability between dissimilar software products and applications. It is proposed that metadata as defined by OMG’s CWM may be intensively used for the integration of different software systems, applications and components from multiple vendors. It has the following general layered structure [OMG, 2000], [Poole et al., 2002], [Poole et al., 2003]:

- The Foundation consists of a UML metamodel and CWM Foundation which supports the general concepts and structures that are shared by all packages.
- The Relational, Record, Multidimensional and XML packages support the definition of various types of data sources and data targets and their exchange mappings.
- The Transformation, OLAP and Data Mining packages define data transformation, analytical processing and data mining.

The different packages included in the model are quite independent allowing the developers and implementors to ignore packages which are not relevant to their needs.

For the purposes of the research reported in this thesis, the DM package has been dealt with and applied in combination with information modeling for manufacturing enterprises.
The data mining generic models included in CWM-DM are described in UML, and they are composed by the following diagrams [Poole et al., 2002], [Poole et al., 2003]:

- The **main model diagram** that represents a general DM model that is defined as a generic representation of data input, application specifications, and the output generated by the execution of a DM algorithm;

- The **settings diagram** that defines the mining algorithm settings and their usage relationships to the attributes of the input data specification. The mining algorithms that are included are as follows: Statistics, Clustering, AssociationRule, Classification, and Regression and CostMatrix.

- The **attributes diagram** describes the mining attributes as NumericAttribute, CategorialAttribute and OrdinalAttribute.

In addition, the CWM includes the following packages [Poole et al., 2002], [Poole et al., 2003]:

- **Core Package** contains basic classes, and associations used by all other CWM packages. This package does not depend on other packages.

- **Behavioural Package** collects classes that provide CWM classes in other packages with behavioural characteristics frequently found in object-oriented systems. By including this package into a software system it is possible to create classifiers that are capable of incorporating object-oriented concepts of operation, method, interface, and event that are specified as part of their definition in a CWM model.

- **Business Information Package** provides general-purpose services to other CWM packages for describing a business environment in which a data warehouse and/or data mining is processing.

- **Business Nomenclature Package** captures generic business concepts in the form of a structured vocabulary defined as an ontology which is independent of any specific implementation or data model. This package also includes a generic description of business concepts as a collection of terms that can be organized into hierarchies of taxonomies and glossaries.

- **Software Deployment Package** defines a high-level architecture including software and hardware elements for a data warehouse and data mining system. This package describes a minimal configuration which is needed by other CWM packages. It
includes a DataManager and a DataProvider which make the link with specified data management interfaces such as ODBC and JDBC.

- **XML Package** defines metamodel classes needed to support the description of XML documents as data resources in a data warehouse and it is compatible with XML 1.0 specification.

- **OLAP Package** contains tools for on-line analytical processing in relational and multidimensional databases, or a hybrid of both.

- **Information Visualization Package** includes the tools for graphical presentation and summarization of information recorded in the warehouse.

**CWM-OLAP** offers support for analytic application software systems that process manufacturing and business data stored in a multidimensional format. The multidimensional format usually includes the consolidation of data drawn from multiple and diverse information sources. Unlike more traditionally structured representations (i.e. the tabular format of a relational database), the multidimensional orientation is a more natural expression of the way business enterprises view their strategic data. The ultimate objective of OLAP is the efficient construction of analytical models that transform raw business data into strategic business insight.

The following features are generally found in OLAP systems [OMG, 2000]:

- Multidimensional representation and processing of manufacturing and business data.

- Upward consolidation of multidimensional data in a hierarchical manner, possibly with the application of specialized processing rules.

- The ability to navigate a hierarchy from a consolidated value to the lower level values forming it.

- Support for time-series analysis; i.e. OLAP users are generally concerned with data and consolidations at specific points in time.

- Support for modeling and scenario analysis.

### 4.5 JAVA™ CAPABILITIES FOR ENTERPRISE APPLICATIONS

Enterprise applications (EA) have became very complex over the past several years, reflecting evolutionary progress in software technology as well as the changes in industrial technology and their businesses. In addition their business activities are much more directed
to e- business and sharing e-marketplaces. Some of common characteristics of all EA should be:

- EA are clearly distributed and the Internet has became the usual distribution environment.
- EA typically require the ability to handle a large number of users and large amount of data and/or flexibility to expand quickly.
- EA require certain services, such as security in order to prevent unauthorized access and complex transaction processing as well as database access.

These requirements can be accomplished using Java™ and associated environments such as JDeveloper, JBuilder and Java 2 Enterprise Edition (J2EE). The technologies within Java™ mainly address server-side development requirements such as [Eckel, 1998], [Dorsey, Koletzke, 2001]:

- JavaBeans are used for building reusable components called beans which are defined as classes. The beans have methods, properties and events. Enterprise JavaBeans slightly differ from JavaBeans, and their concepts are described in the next section.
- Java Servlets, which provide the means for interactions with Web clients.
- JavaServer Pages (JSP), which allow developers to create dynamic content for thin clients.
- Java DataBase Connectivity (JDBC) Standard Extension in order to link an application to a relational database such as Oracle.
- Java™ API for XML Parsing (JAXP) for any XML-based EA.
- Incorporating distributed objects such as CORBA and Remote Method Invocation (RMI) by importing specific classes (import org.omg.CORBA.*)
- Java Authentication and Authorization Service (JAAS) for enabling a security layer to the applications.

The syntax of Java is similar to C++ because Java was derived from C++. However there are several differences intended to be improvements, but this was not always possible because of the processing speed. The common elements and the differences are explained in detail in [Eckel, 1998], and a few of them are as follows:

Because Java runs in the first step as an interpreter, it is in the range of 20 times slower than C++. Nothing prevents the Java language from being compiled and there are just-in-time compilers appearing at the time of writing that offer significant speed-ups. It is not
inconceivable that full native compilers will appear for the more popular platforms, but without those there are classes of problems that will be insoluble with Java because of the speed issue.

Java is fully object-oriented, everything must be in a class, and class definitions are roughly the same form in Java as in C++. All objects of non-primitive types can be created only via `new`. There is no equivalent to creating non-primitive objects “on the stack” as in C++. All primitive types can be created only on the stack, without `new`. No forward declarations are necessary in Java, it is possible to use a class or a method before it is defined, and the interpreter ensures that the appropriate definition exists. Inheritance in Java has the same effect as in C++, but the syntax is different. Java uses the `extends` keyword to indicate inheritance from a base class and the `super` keyword to specify methods to be called in the base class that have the same name as the method used. However, the `super` keyword in Java only allows access to methods in the parent class, i.e. one level up in the hierarchy. Base classes implemented in C++ can have access methods that are deeper in the hierarchy. The base class constructor is also called using the `super` keyword. As mentioned before, all classes ultimately and automatically inherit from `Object`. There is no explicit constructor initializer list like in C++, but the interpreter forces all base class initialization to be performed at the beginning of the constructor body. Member initialization is guaranteed through a combination of automatic initialization and exceptions for uninitialized object handles. Java uses overriding method to create a method in a sub-class with the same name, return types, and arguments as the method in superclass.

Java has no preprocessor. To use classes in another library, it is necessary to call `import` specifying the name of the library. There are no preprocessor-like macros.

### 4.5.1 UNDERSTANDING ENTERPRISE JAVABEANS

JavaBeans is an architecture and platform-independent set of classes for creating, using software components implemented in Java and developing component-based systems. It is used to implement beans which are a reusable software component that can be visually manipulated in a framework. JavaBeans offers the following services [Eckel, 1998], [Vogel, Rangarao, 1999]:

- Graphical User Interface merging by using the `container` concept;
Persistence defined as a mechanism by which components can be stored and retrieved within a specific context;

Event handling defining the interactions between components;

Introspection by which the components can be internally read by other components;

An application builder support for the components.

JavaBeans is similar, and can be used in conjunction with other visual programming environments such as Visual Basic and Borland Delphi.

The original JavaBeans specification describes the standard behaviour and properties of Java components that run primarily on the client side of a client/server system. The introduction of Enterprise JavaBeans enhances JavaBeans at a conceptual level. EJB is a component architecture for creating scalable, multi-tier, distributed applications, and it makes possible the creation of a dynamically-extensible application server. The usual bean and an Enterprise JavaBean do not have much in common from a programming perspective, even if the concept around them is similar, i.e. to enable developers to use reusable components in their applications.

Usual beans are Java classes, typically graphical user interface (GUI) components, designed to conform to a series of programming conventions so that integrated development environments like Symantec Visual Cafe or IBM's VisualAge for Java can inspect the beans and hook them together into applications. The development environment can then generate appropriate Java code to work with the beans. For example, a bean might represent a special kind of text field or list box, and the development environment could then ease the code development process by graphically allowing you to configure this bean and call the right methods for the desired functionality.

EJBs are likewise components, but the associated beans follow a completely different set of conventions and interfaces and are not for use inside development environments. The purpose of EJBs is to encapsulate business logic (for example, the steps involved in depositing money into an account, calculating income tax, or selecting which warehouse to ship an order from) into server side components.

In the EJB paradigm, an application is implemented as a set of business-logic-controlling EJB components that have been configured in application-specific ways inside an EJB container such as an application server. Clients are then written to communicate with the EJB
components and handle the results. The standardized interfaces exist to allow the EJB container to manage security and transactional aspects of the bean.

EJB and Corba/IDL are two complementary technologies addressing the application server components. The EJB specification will allow CORBA clients to interact with EJB components by presenting them through a CORBA-accessible interface.

Figure 4.2 Enterprise JavaBeans Architecture [Vogel, Rangarao, 1999]

EJB architecture is depicted in figure 4.2 [Vogel, Rangarao, 1999]. The Enterprise JavaBeans specification is defined by Sun Microsystems at:

http://www.java.sun.com/products/ejb, and it is as follows:

Infrastructure is delivered by EJB Server provider which is the vendor with expertise in distributed infrastructure and associated services. EJB server provider implements a platform which facilitates the development of distributed applications and provides a runtime environment for them. The EJB Container provider is part of the infrastructure and is defined as an expert in distributed systems, transactions and security. A container is a runtime system for one or multiple enterprise beans. It links different beans and EJB server.
At the level of Application, the Enterprise Bean Provider and the Application Assembler are defined. The Enterprise Bean Provider is an expert in the application domain which implements the logic of the application, its transaction system and security. The Application Assembler unifies the components and adds dedicated and standard interfaces. The Deployment and Operation of an application using EJB are in the charge of the Deployer and System Administrator.

An Enterprise Bean provider is concerned with the following three components [Vogel, Rangarao, 1999):

- The Home Interface specifies operations to create, find and remove an enterprise bean;
- The Remote Interface specifies the business logic of the application provided in the enterprise bean;
- The Implementation classes implement the functionality defined in the home and remote interfaces.

The application of Enterprise JavaBeans in this research is discussed in chapter 9 and the corresponding classes for the above components are presented in figure 9 4.

4.5 SUMMARY

This chapter deals with the background material for the research considered in this thesis and includes basic concepts and definitions of a software architecture and a framework, principle of modeling, the OMG standards and middleware projects. This research background provides a support for the next chapters as it presents Model-Driven Architecture (MDA) and Common Warehouse MetaModel (CWM).

The Java capabilities have been presented here in order to explain the selection of this language for the implementation examples which demonstrate the theory reported in this thesis. Enterprise JavaBeans (EJB) are introduced as their main purpose is building distributed applications with pluggable components from different software providers. They can enable developers to implement solutions that take advantage of previously difficult-to-implement features like distributed transaction management and multithreading, which now are provided by the EJB container. Using Enterprise JavaBeans means that it is possible to
link the whole family of other Java technologies such as RMI, JDBC, JNDI and OMG’s CORBA into an enterprise common platform as presented in chapter 9.
Chapter 5

MULTIPLE VIEWS APPROACH BASED ON ENTERPRISE ARCHITECTURES

5.1 ENTERPRISE ENGINEERING OVERVIEW

The purpose of this chapter is to describe the multiple views of the suggested framework for distributed knowledge discovery and data mining systems embedded in extended enterprise which is shown in figure 2.1. This chapter also demonstrates that the suggestion of introducing knowledge and mining views is in accordance with reference architecture for enterprise modeling, ISO standards and frameworks. The suggested framework also uses the existing views of enterprise reference architectures and frameworks.

Reference architectures are fundamental for enabling an enterprise to analyze its behaviour and to cope with changes in response to external dynamics and uncertainties. The concepts of enterprise reference architectures and frameworks are also evolving, and they are slightly different to those explained in the previous chapter. However the earlier modeling approaches provide a valuable basis for analysis of current enterprise integration solutions in order to incorporate new systems such as KD/DM and develop re-engineering solutions. Based on other references, Molina (1995, 2002) and Bell (2002) point out the distinction between a reference architecture and a framework because these concepts were ambiguously used in manufacturing area. This distinction is completed and briefly explained below:

- A framework offers an incomplete and general design and implementation roadmap or guideline for all enterprise information systems.

- A reference architecture provides a standardized and comprehensive model including the relations and interactions between elements, which should be totally or partially adopted in order to integrate a system into an existing enterprise. According to [Vernadat, 1996] an architecture is a finite set of interrelated components put together to form a consistent whole that is defined by its functionality, and the reference architecture is defined for a given domain and is a generic architecture from which other architectures can be compared or derived. An architecture can also be related to hardware and software systems components of an enterprise.
Thus, framework is a more general concept than architecture, meaning that different architectures can be developed within a defined framework [Vernadat, 1996].

Zachman (1997, 1999) who is a world's recognized expert on Enterprise Architecture and author of a Framework for Enterprise Architecture, defines a strong connection between business processes, organization strategies and enterprise architectures. This approach also emphasizes that an enterprise must produce models in order to deliver systems implementations in the short-term and at the same time for the long-term, instantiate the architecture process in order to ensure on-going coherence of system implementations and to build an enterprise environment conducive to accommodating high rates of change.

In the previous chapter, the concepts of architecture and frameworks were also presented in detail. Based on all these considerations this research suggests the definition of a unified object-oriented framework for distributed KD/DM systems embedded in extended enterprise. The framework intensively takes into consideration previous enterprise modeling approaches and related solutions and it supports the definition of a range of system architectures as explained in chapter 6.

However, Zachman (1999) stated that:

"The intense world-wide preoccupation with "data warehouse" should have been highly predictable as somehow, we had to find an approach to compensate for the historical lack of Enterprise-wide data architecture. The data in the existing systems, the "legacy" is so discontinuous, so inconsistent, so incorrect, so redundant, it borders on useless for management purposes as cited in a CEO survey. The legacy data may serve to get the transactions processed, but when validity of its values and its meaning become vital to understand what is happening in or to the Enterprise, it is seriously deficient and therefore contributes substantially to the frustration the Enterprise is feeling toward information systems. The creation of the data warehouse (i.e. creating the integrated semantic structure) and extracting, transforming, cleansing, integrating and distribution of the data is simply after-the-fact effort to redeem a very tenuous Enterprise-frustrating, legacy situation.

I do not want to minimize the very significant advances in decision support systems, analytical processing, data mining, and so on that nearly universal focus on data warehouse has brought to the information discipline, but setting these universal, industry-wide advances aside for a moment, the expense of building a data warehouse in a given Enterprise is substantial.

Further I would suggest that if the issues of Enterprise Architecture are not encompassed in any current data warehouse implementation, the ultimate result will be increased Enterprise frustration and the expenditure will be perceived as dissipated as the implementation will simply be one more un-architected, redundant, legacy file with a new name "Data Warehouse."

The best known enterprise modeling approaches and reference architectures such as CIMOSA, ARIS, PERA and GERAM constitute a common environment and methodology for systems integration, comparative analysis, design and re-design of a manufacturing
enterprise using advanced software tools [Bernus, Nemes, 1996], [Bernus et al., 1996], [ESPRIT, 1993], [Edwards et al., 1998], [Harding et al., 1999], [Harding, Popplewell, 1999], [Toh, 1999a], [Toh, 1999b], [Vernadat, 1996], [Wang et al., 2002]. Besides these projects which aim to define, analyse and apply reference architectures and models, TOVE (Toronto Virtual Enterprise) dealt with ontologies for enterprise modelling, agent architectures, coordination, and applied them to integrated supply chain management systems, knowledge-based design and enterprise engineering systems [Fox, 1996]. TOVE was developed at the University of Calgary, Canada within the Enterprise Integration Laboratory. Most recently the GLOBEMAN21 (Global Engineering and Manufacturing in Enterprise) international projects [Berry, 2000], [Szegheo, 1999] addressed three main aspects of manufacturing: sales and services, inter-enterprise management and engineering and based on industrial requirements specifications the work is coordinated and integrated into an IT architecture for enterprise networks and virtual enterprises. Generally, these projects define reference models and associated methodologies or guidelines, based on which IT support for virtual enterprise networks can be set up and operated. Furthermore a definition of an extended enterprise is provided that shows the main difference between it and a virtual enterprise. Extended enterprise is described as a long-term co-operation and partnership based on information and knowledge exchange [Szegheo, 1999]. Virtual Enterprise is the temporary link between enterprise systems based on Internet and intranet technologies. According to [Bernus et al., 1996], [Vernadat, 1996] the definitions of virtual and extended enterprise are overlapping, and they are formed and dissolved as quickly as the underlying business processes are changed. Within the new approach of extended enterprise developed by GLOBEMAN projects, the emphasis is on integrated supply chain and customer relationship management systems.

These international projects include the MISSION (Modelling and Simulation Environments for Design, Planning and Operation of Globally Distributed Enterprises) project which provides a comprehensive architecture to support manufacturing system modelling, simulation and operation [ESPRIT, 1998, 2001], [ESPRIT, 2001].

This chapter investigates and uses previous enterprise reference architectures and shows how they may be interpreted or adapted in order to support KD/DM. According to [Molina, Bell, 2002] the use of reference architectures and models has been found to be a helpful concept to allow a systematic approach of a system development and implementation. It does, however, offer challenges due to the complexity of producing complete models.
Generally, the investigation of reference models does present a major challenge, but it is not essential to produce complete reference models in order to gain considerable value from them in particular areas of relevance to the industry, such as the introduction and integration of systems and enterprise modeling.

This chapter also introduces the background of information modeling in manufacturing, starting with the MOSES project [MOSES, 1992] but subsequently under continuous development according to the latest requirements in the area and new paradigms such as global, agile and holonic manufacturing [Maxfield et al., 1998]. The description of manufacturing models is based on outstanding researches done in the Department of Manufacturing Engineering at Loughborough University, and includes [Costa et al, 2001], [Dorador, Young, 2000], [Harding, Popplewell, 1996], [Harding, Yu, 1999], [Harding et al., 1999], [Harding, Popplewell, 1999], [Molina, 1995], [Zhao et al., 1999], [Ye, 2002].

The reference architecture is also considered as a conceptual area and an intellectual paradigm which facilitates analysis and accurate discussion of a specific area of investigation [Bernus et al., 1996], [Vernadat, 1996]. Moreover reference architectures are very useful in different aspects of enterprise integration by defining a perspective or viewpoint from which the enterprise is considered, and hiding the irrelevant aspects in order to reduce complexity. Harding et al. (1999b) have defined the view as a particular interpretation of the enterprise design and operation information, or a tool for building, interpreting and analyzing a factory model, reflecting a particular design paradigm.

CIMOSA (Computer Integrated Manufacturing – Open System Architecture) defines a model-based enterprise engineering framework as shown in figure 5.1 (CIMOSA Cube). This enterprise reference architecture includes function, information, resource and organization views which are described as follows [Bernus et al., 1996], [ESPRIT, 1993], [Rolstadas, Andersen, 2000], [Toh, 1999], [Vernadat, 1996], [Wang et al., 2002]:

- **Function view** provides a hierarchically structured description of the functions, dynamic behaviour and static functional structure of an enterprise;
- **Information view** provides the description of a structured set of enterprise objects and information models which are identified in other views;
- **Resource view** provides a description of the resource organization of the enterprise;
- **Organization view** provides the organizational structure of the enterprise.
The number of views is not limited, and it can be expanded as necessary, but it is recommended that the number is kept to the minimum possible. Therefore the product view has been introduced in order to reflect changing customer demands, to manage the product data efficiently and to provide customized products [Toh, 1999a], [Wang et al., 2002] [ Wortmann et al., 2001]. Within [Harding, Popplewell, 1999], [Harding et al., 1999a], [Yu et al., 1998] etc. A multi-view approach defined as the Factory Design Process (FDP) has also been suggested. This additionally included the following views:

- **Strategic View** provides the description of business aims and objectives of an enterprise in order to identify enterprise strategies;
- **Business Process View** includes the interaction between a group of processes and their performance.
- **Performance View** provides the information needed to determine whether the enterprise can satisfy the identified objectives.

FDP and its associated Factory Data Model (FDM) have been particularly applied to the design and quick redesign of an enterprise for new product introduction. Furthermore the associated databases could be used in conjunction with data warehousing and further data mining as presented in [Harding, Yu, 1998].

### 5.2 KNOWLEDGE AND MINING VIEWS

The existing views that have been reviewed enable designers and users to better understand and communicate the structure, purposes, capabilities, resources and relationships within the enterprise and a network of enterprises. However, they do not particularly support them to identify and extract knowledge that exists within the system mainly because they do not capture knowledge in a systematic and organized manner. Hence, different views of the enterprise, and of the extended enterprise are needed to enable efficient knowledge extraction through data mining. Therefore it is necessary to define the knowledge view as the description of processed information with an associated meaning, which leads to an action that adds value to the initial data. The knowledge view facilitates enterprise integration and extended enterprise collaboration and coordination from the knowledge perspective. Enterprise knowledge could be classified as follows:

- Knowledge about the past which is stable, voluminous and accurate;
- Knowledge about present which is unstable, compact and may be inaccurate;
- Knowledge about the future which is hypothetical.

Knowledge discovery and data mining are critical processes applied to existing enterprise databases in order to find new information, knowledge and patterns which show the future enterprise behaviour and improve its business performance. Furthermore the knowledge view supports other areas of knowledge engineering and management, and it makes possible the distinction between tacit (implicit) and explicit knowledge [Polanyi, 1966]:

- **Tacit knowledge**: implicit, mental models and experiences of individuals.
- **Explicit knowledge**: formal models, rules and procedures.
The conventional model to turn data into information and further into knowledge is defined as follows:  \[ \text{data} \rightarrow \text{information} \rightarrow \text{knowledge} \]

However one the main goal of data mining is to gain new information and knowledge from databases. Therefore data mining may process the information embedded in the information view and the new information, knowledge and patterns could be captured in the same views. However to obtain a distinct separation, an additional view of the enterprise is suggested and this is called the mining view. The mining view should include the description of mining and business models obtained through knowledge discovery, OLAP, data mining and other advanced data exploratory techniques. The knowledge and mining views included in the CIMOSA cub are shown in figure 5.1.

Figure 5.2 Aris Views based on [Scheer, Kruse, 1994]

Generally the main objective of CIMOSA is to develop a CIM reference architecture [Bernus et al., 1996]. The modeling framework which is explained in detail, in the section 5.5 is formed of the following parts: a: the reference architecture and b: the particular architecture. Obtaining useful information and patterns from input data or preprocessing an initial data set should complete the information view defined by CIMOSA or refine the information and knowledge embedded in the other views. Also because the CIMOSA
reference architecture contains embedded information and knowledge, this can be usefully process at to improve the enterprise’s future business and reengineered solutions.

*ARIS* (Architecture of Integrated Information Systems) is similar to CIMOSA, but instead of focusing on manufacturing systems, it deals more with the business-oriented issues of an enterprise and the corresponding information systems. ARIS has been developed by Prof. Scheer, at IDS, an international company recognized for its focus on e-business and engineering. Starting with a top-down business problem-definition stage, users can build views of their business in terms of organization, data or resources, process control and function or organization [Scheer, Kruse, 1994].

The data view is used to define semantic data models, in terms of entity-relationship diagrams, and then to translate them into relational schema before implementing the associated database. ARIS provides a basis for adding new information, knowledge and mining views as shown in figure 5.2. The ARIS architecture definition also includes a top-down business that is necessary for DM systems.

The ARIS associated toolset and its add-on components enable the extended enterprise and its global definition, and improve the design and modelling of business processes as well as their analysis and optimization. Using this tool it is possible to make quick decisions about the management of e-business processes. The ARIS toolset provides realistic simulations of resource utilization, activity-based cost calculations (i.e. for make-or-buy decisions), as well as web-based communication of modelled and optimized company processes.

This architecture presented in figure 5.2 combines a high-level business-oriented view with layers in order to address document workflow, software engineering, activity-based costing and project management. Based on the analysis of ARIS architecture of information system and toolset it is possible to conclude that an enterprise applying ARIS could successfully include KD/DM systems. Furthermore within ARIS, UML is used for modelling business processes and translating problem definitions into an information system according to UML methodology.

Generally, adding new views or extending the interpretation of existing views offers support for the development of applications based on knowledge and business intelligence solutions. This modeling approach should be directed to the definition of a common enterprise model. However CIMOSA, PERA, ARIS, GERAM etc. are high level and abstract reference architectures which do not offer implementation solutions especially for specific IT systems such as DM applications which process particular data stored in defined databases. It
is therefore suggested that the implementation solutions should be based on an *application view* obtained through unified perspective of application areas such as SCM, ERP and CRM.

The examination of these high level abstract reference architectures and standard frameworks such as RM-ODP alongside implementation solutions such as CORBA, MBA and EJB presented in chapter 6 is essential to the work reported in this thesis, since this research is based on the principle that:

*IF an IT system follows reference architectures and models and also uses OMG standards such as UML, CORBA and MDA THEN standard KD/DM systems can be incorporated in its design, and can be flexibly integrated within the enterprise reference architectures.*

### 5.3 INFORMATION AND DATA VIEWS

In the previous section it has been explained that the different enterprise reference architectures handle data and information in different ways. As previously stated ARIS is an architecture that directly supports information and knowledge retrieval through data mining. In this section, the general data and information requirements for the inclusion of a knowledge discovery and data mining system are considered.

Generally, the mining process operates on data and creates mining models as already presented. Data is at the basis of KD and DM, and it can be organized in databases or in separate files. The data associated with a specific mining process is called a *data set*. It is very important to have access to suitable data sets in order to implement successful DM projects.

The *data view* represents the abstract level of all data manipulated by different software systems into an enterprise information system. Therefore a data view has to be considered at the level of enterprise architecture and the definition of a high level enterprise data architecture should significantly support data mining. The data view as part of ARIS has already been explained and has been used as a basis for the inclusion of additional views such as information, knowledge and mining views as depicted in figure 5.2.

Williams, T.J. (1996) has specified the following regarding data and its organization [Bernus et al., 1996].
• Standardized methods of database organization, management and system-wide access should be emphasized;
• Shared data, data used by multiple functions or processes, should have a standardized and neutral format;
• Data is not necessarily tied to any particular control level; it resides in a database and is managed by a separate administration system;
• Data exchanged between enterprise entities is of the following types (figure 5.3):
  1. Raw data;
  2. Clean data which is smoothed and conditioned;
  3. Information which has been associated with a semantic.

An Enterprise Data Architecture is based on data views and data models. A data view is clearly defined within the ARIS approach and it is used to define coherent data models for databases. In addition ARIS, being developed for e-business support systems, has a structure flexible enough to support more DM approaches than other enterprise models.

![Figure 5.3 Different Stages of Data Processing (Williams, 1996)](image)

Generally, enterprise engineering approaches do not consider a data-driven model which would be very useful in a data warehouse and effective data management system. However a purely data-driven model will not produce an adaptive enterprise system architecture because it could not incorporate changes without extracting information and knowledge [Bernstein, Marca, 1998].

More problems arise in practice particularly when DM processes single flat files obtained from an enterprise database. This practice results in redundant processing, degradation of data, lack of flexibility and integration.
The main directions of enterprise data integration such as common relational and/or object-oriented databases supporting knowledge discovery and data mining are as follows:

- XML represents a critical future direction for effective management of data, metadata, as business process representation and will play an increasingly important role in systems development using XML and Enterprise Application Integration (EAI).
- XML-based EAI technologies and Model Driven Architecture (MDA) permit implementation with minimal or no change to the existing applications and associated data.
- Using a central data warehouse by creating data marts which represent a common data repository defined as an environment containing a specialized set of related data that has been customized for a specific application. Usually there are three types of data marts: embedded, dependent and independent.
- Profiling data which is the process of checking the integrity of a database by comparing it with existing models of a database schema definition.
- Encapsulating Data using object-oriented approach.

This research is also intended to fill the gap between the amount of data available to an enterprise and its ability to use data effectively especially through a methodological DM processing. This gap is considered to be bridged using the unified object-oriented framework presented in chapter 7. The information models obtained from data through data mining are incorporated in the information view which is defined within the CIMOSA reference model and this is explained in sections 5.1 and 5.2. However CIMOSA does not have the specification for data organization included in a data view such as ARIS. Therefore a good support for knowledge discovery and data mining should be provided by applying a combination of the CIMOSA reference architecture and the ARIS model.

5.4 ENTERPRISE MODELING AND DATA MINING

A model is a representation of some complex real-world entity such as a manufacturing company, built for a particular purpose. A model therefore describes the entity in some ways, often using information about the entity or information about resources and processes owned by the entity or knowledge about how the entity exists or operates.
Inevitably then some of this information and knowledge is *common* as it is relevant to more than one model of the entity, i.e. the models overlap.

If *enterprise modelling* is considered as the process of building models of the whole or part of the enterprise such as process models, data models, resource models etc. based on knowledge about the enterprise, previous models and/or reference models as well as domain ontologies and model representation language [Vernadat, 1996], then *mining models* are directed to logically fit or overlap with *enterprise models*, except that they are obtained by knowledge discovery. Descriptive enterprise models are widely used and usually are based on diagrams such as IDEF, SADT etc. These models can be complemented and improved with *mining descriptive models* describing patterns in existing data about an enterprise behaviour, and performances. *Predictive models* may evaluate initial business goals, their achievement and forecast future business. Enterprise modeling supported by mining has not been considered in previous enterprise engineering approaches. Common enterprise and mining modeling may be able to quickly analyze all available data and identify what information is important. It may enable managers to understand exactly how to respond to each individual event and change, and to improve performance. Enterprise Modeling starts by identifying the key performance indicators for corporate business goals. Common Enterprise and mining models can be used to predict and rank the profitability of each initiative, and decide where best to invest to achieve the desired outcome.

The CIMOSA cube depicted in figure 5.1 is used to assist the enterprise/manufacturing system designer to build a particular enterprise architecture which is a set of models documenting the CIM environment from requirements to implementation.

*Function Views* are used to design a particular system from a generic one as explained below:

*Generic functions* are performed in every enterprise independent of its size, organization and business area. These are defined as control of workflow, administration of information, integration of resources and management of communications, and they should be performed by generic system services.

*Specific functions* are dependent upon the individual enterprise. Examples include design of products and production processes, generation of production plans, scheduling of production, shipment of products, maintenance of equipment, processing of orders, accounting etc. Specific functions may be performed by machines, humans and computers. CIMOSA separates functions using the following two interrelated concepts:
The CIMOSA Modelling Framework in which specific and generic functions are clearly separated.

The CIMOSA Integrating Infrastructure supporting execution of generic functions and linking specific functions. It is effectively the communication system which interconnects all of the functions in a CIM system.

CIMOSA applies the general views as follows:

- The Function View describes workflows;
- The Information View describes the information models.
- The Resource View describes the structure of resources (human resources, machines and information systems);
- The Organization View defines authorities and responsibilities;
- The added Knowledge View describes the knowledge models incorporated or obtained from the above views;
- The added Mining View describes the common enterprise and mining models useful in keeping a history of enterprise models.

The CIMOSA Modelling Framework provides the user with architectural constructs and guidelines for the structured description of business requirements and their translation into a particular system design and implementation. Enterprise modeling offers a systematic way to design any system integrated into an enterprise. Mining models provide information and knowledge about past business behaviour of the enterprise, and could help decision making support system design and implementation.

The Derivation Process guides the user through the three modelling levels: from the definition of enterprise business requirements (Requirements Definition) through the optimization and specification of the requirements (Design Specification) to implementation (Implementation Description). On each modeling level the enterprise is analyzed from different viewpoints (Modelling Views). The modeling based on knowledge and mining views is directed at keeping a history of past enterprise definitions and modeling solutions which can be used to the optimization of manufacturing system design.

To reduce modeling effort CIMOSA uses the Instantiation and defines three levels of genericity from purely generic to highly particular:
• **Generic Level** is a reference catalogue of basic CIMOSA architectural constructs called also building blocks for components, constraints, rules, terms, service functions and protocols.

• **Partial Level** contains a set of partial models applicable to a specific category of manufacturing enterprises.

• **Particular Level** is related to one particular enterprise and is defined in the Instantiation Process by the modeller using already prepared building blocks from the Generic and Partial Level and developing new particular enterprise specific components.

The *Generation Principle* recommends the modeling of the manufacturing enterprises according to the views included.

**PERA** (Purdue Enterprise Reference Architecture) has a layering structure and it has been created to cover the life cycle of an enterprise from inception and objectives definition to the operational level [Vernadat, 1996]. **PERA Enterprise Model** includes the three main components of any enterprise: Production Facilities, People, Control and Information Systems. The related methodology identifies parts of enterprise which are considered, then the products and services offered, and the basic functional requirements. An additional DM system added to this architecture could process the objectives definition during the life cycle of enterprise.

**GERAM** (Generalized Enterprise Reference Architecture and Methodology) provides a comprehensive description of the elements recommended in enterprise engineering and integration and thereby sets the standard for the collection of tools and methods from which any enterprise should benefit to more successfully tackle initial integration design and the change processes which may occur during the enterprise operational lifetime. It does not impose any particular set of tools or methods, but defines the criteria to be satisfied by any set of selected tools and methods. GERAM views enterprise models as an essential component of enterprise engineering and integration; this includes various formal forms of design descriptions utilized in the course of design. The purpose of GERAM is to define a reference providing a consistent modeling environment, a detailed methodology and standard models for products, processes, management, enterprise development and strategic management [Bermus, Nemes, 1996], [IFAC/IFIP, 1999], [Vernadat, 1996].
The GERAM framework has the following main components [IFAC/IFIP, 1999], [Vernadat, 1996]:

- **GERA** - Generalized Enterprise Reference Architecture is the basic concept to be used in enterprise engineering and integration including enterprise entities, life cycles and life histories of enterprise entities.

- **EEMs** - Enterprise Engineering Methodology describes the processes of enterprise engineering and integration, and they may be expressed in the form of a process model or structured procedure with detailed instructions for each enterprise engineering and integration activity.

- **EMLs** - Enterprise Modelling Languages define the generic modelling constructs for enterprise modelling adapted to the needs of people creating and using enterprise models. These languages enable the modeling of the human part in the enterprise operation as well as the parts of business processes and their supporting technologies.

- **EMs** - (Particular) Enterprise Models represent all or part of the enterprise operations, including its manufacturing or service tasks, its organization and management and its control and information systems. These models can be used to guide the implementation of the operational system of the enterprise (EOSs) as well as to improve the ability of the enterprise to evaluate operational or organizational alternatives (for example, by simulation) and thereby enhance its current and future performance.

- **EETs** - Enterprise Engineering Tools which support the methodology and the languages used for enterprise modeling.

Potentially, all proposed enterprise reference architectures and methodologies could be included in GERAM so that developers of particular architectures could gain from being able to commonly refer to the capabilities of their architectures, without having to rewrite their documents to comply with GERAM. Users of these architectures should benefit from GERAM because the GERAM definitions would allow them to identify what they could (and what they could not) expect from any chosen particular architecture in connection with an enterprise integration methodology and its proposed supporting components [IFAC/IFIP, 1999].
5.5 A MINING VIEW SUPPORTED BY RM-ODP

The Reference Model for Open Distributed Processing (ISO/IEC DIS 10746-1:1995, or RM-ODP) has been defined by ISO as a general coordinating framework of open distributed processing (ODP). Based on RM-ODP it is possible to produce standard architectures which support distribution, inter-networking, interoperability and portability.

---

**Figure 5.4 KD/DM System Adhering to RM-ODP**
RM-ODP provides a coordinating framework to support the design of distributed systems in heterogeneous environments based on the standardization of open distributed processing. It identifies the main properties of open distributed systems such as openness, integration, flexibility, modularity, federation, manageability, provision of quality of service, security and transparency from the point of view of the application.

The RM-ODP consists of the following parts [Costa et al., 2001], [Dunlop et al., 1999]:

**Part 1 — Overview** states the motivation and objectives of ODP systems. It includes an overview of the ODP architecture explaining the key components in the architecture, ODP functions and distribution transparency. It introduces the architecture to the developers of ODP systems, explains how it can be applied to future systems and how to use conformance assessment for existing systems.

**Part 2 — Foundation** specification contains the definition of the concepts and analytical framework for the description of ODP systems such as object, interface, object state and communication. The foundation specification also introduces the principles of conformance to ODP standards and the way in which they are applied.

**Part 3 — Architecture** contains the specification of the required characteristics that qualify a distributed processing system as being open. ODP systems are defined from five viewpoints:

- **Enterprise viewpoint** describes the requirements and the functionality of an ODP system;
- **Information viewpoint** describes the information system structure, flow, and manipulation constraints;
- **Computational viewpoint** describes the methods, techniques for information processing;
- **Engineering viewpoint** describes the engineering resources supporting ODP;
- **Technology viewpoint** describes the components used to develop and implement the system.

**Part 4 — Architectural semantics** is a formalization of the ODP modeling concepts.

RM-ODP has been investigated and applied for the application of UML in the creation of manufacturing and product models for integrated enterprise systems such as described in [Dorador, Young, 2000], [Costa et al., 2000] etc. However the previous research does not consider intelligent data analysis for producing information models.
A KD/DM system adhering to RM-ODP is presented in the figure 5.4 and it is a general framework for designing these systems. This framework approaches the design of a KD/DM system by dividing the KD/DM application design into viewpoints according to RM-ODP standard, and each viewpoint represents a certain level of abstraction. ODP viewpoints set out a systematic way to decompose the design and implementation of distributed KD/DM systems. The Enterprise viewpoint describes the data mining application specifications and requirements. It includes also a description of the system functionality using Use Case Diagrams which are presented in detail in section 7.3. This view could include additional modeling techniques for data and process descriptions such as IDEFx. The information view includes the information models used as input by the knowledge discovery and data mining system and mining models that resulted through the mining processing. The research reported in this thesis mainly deals with UML modeling and design based on CWM-DM which is also defined as standard metamodel by OMG and was explained in detail in sections 4.3 and 4.4. The computational view describes the knowledge discovery and data mining algorithms implemented by different software systems. This view also includes the business logic of the application. The engineering and technological views deal with the distributed systems to support an ODP system as explained in chapter 6. The design of a KD/DM system that conforms to RM-ODP offers a good basis for integration with other enterprise systems and of comparison with systems for exploratory data analysis. However adhering to a reference model does not solve the practical design and implementation issues involved in software engineering [Costa et al., 2001]. Also within RM-ODP it is possible to use different modeling, representation and processing techniques which do not properly communicate or interface with each others. Therefore even though at the conceptual level, the reference model provides general and coherent views that organize a KD/DM-ODP system, at the implementation level the system could become dissimilar. Also RM-ODP is not intended to standardize the components of the system nor to unnecessarily influence the choice of technology. The possible implementation solutions that conform to engineering and technological views of RM-ODP are presented in chapter 6.
5.6 INFORMATION MODELING TO SUPPORT DATA MINING

Based on the reference architectures and frameworks previously described, the standard information models of manufacturing data and product have been defined by MOSES project [Dorador, 2001], [Dorador, Young, 2000], [Harding, Popplewell, 1996], [Molina, 1995], [Molina, Bell, 2002], [Zhao et al., 1999].

Figure 5.5 Manufacturing Model Including Additional Classes
A product model is a comprehensive representation of a product and it contains detailed data description, information, objects and knowledge in order to support extended enterprise applications. The manufacturing model describes available manufacturing processes, resources and strategies, and it also implicitly captures manufacturing facilities and their capabilities [Dorador, Young, 2000], [Harding, Popplewell, 1996]. The initial aim of the product and manufacturing models defined within MOSES project, was to support concurrent engineering systems, their integration, exchanging of information and knowledge sharing.

The basic manufacturing model is composed of classes describing resources, processes and strategies. The representation of resources and processes are consistent enough to capture facilities and their capability. Also the resources class was based on the resource view defined by CIMOSA which represents an enterprise’s capabilities and management [Rolstadas, Andersen, 2000].

Generally, the manufacturing and product models interact with each other. A manufacturing model can be described using UML diagrams as shown figure 5.5 [Dorador, 2001], [Dorador, Young, 2000]. It includes the representation of the level of an enterprise of factory, shop, cell and station in order to offer support for global enterprise systems and modeling.

A KD/DM system included in an extended enterprise processes integrated data sources, and the first step in creating suitable data sets source is the development of comprehensive information models according to the previously discussed multiple views such as resources, organizations etc. The models are associated with suitable schema definitions. The basic manufacturing model considered as a reference in this research is a good basis for obtaining new information, knowledge and patterns. However it is suggested that the basic manufacturing and product models should be refined especially by the addition of classes which describe business and market information associated with a manufactured product. Therefore the new classes added are called Competitive Capabilities, Business Product Specification (P³TQ), Core Capabilities [Braganza, 2002], [Pyle, 2003], [Swink, Hegarty, 1998].

The addition of new classes should be necessary for supporting data mining applications in the areas of Supply chain, Customer Relationship Management and Enterprise Resource Planning. In addition the classes which extend the capabilities could contribute to enterprise integration such as presented in [Braganza, 2002]. This research states that enterprise initiatives differ by their purposes, and it suggests a framework for enterprise integration.
based on specific capabilities developed for the organization. These capabilities are defined as Competitive Capabilities, and are described by the class CompetitiveCapabilities which captures the information related to enterprise competitiveness, organizational consequences and its position in the market place.

Business Product Specification class is based on a simple principle by which is shown that all companies face a fundamental problem of having the right product, in the right place, at right time, in the right quantity and for the right price (P³TQ- Product, Place, Price, Time, Quantity) [Pyle, 2003]. The P³TQ concept attempts to include JIT paradigm defined as the successful execution of all manufacturing activities from design to delivery including all stages of conversion from raw material to the final product in order to reduce procurement delays and stocks.

The class Core Capabilities intends to make a stronger link between manufacturing and marketing. According to [Swink, Hegarty, 1998] consistent relationships between market and manufacturing strategies have not been clearly established, and also core capabilities have not been sufficiently defined. Capabilities could also derive more from manufacturing infrastructure: people, management and information systems, learning and organizational enterprise resources. Core manufacturing capabilities are also distinct from the concept of manufacturing competence. Manufacturing capability refers to a fundamental proficiency in manufacturing and the manufacturing competence could be defined as the degree to which manufacturing performance supports the strategic objectives of the enterprise. Core capabilities contained within a company’s manufacturing processes enable it to differentiate its products from competitors’ products [Brown, 1996], [Leavy, 1996], [Swink, Hegarty, 1998].

The Market Strategies class describes the new products introduction and distribution according to the identification of new market segments, pricing and customers requirements and profiling.

The basic manufacturing model presented below is considered as a reference according to enterprise architectures, models and ISO standards. The additional classes such as CompetitiveCapabilities, CoreCapabilities, BusinessProductSpecification and MarketStrategies are linked to manufacturing strategies and business specification. This aspect is useful because the objectives of data mining are related to business behaviour of an enterprise and to gain competitive advantage. Also the basic manufacturing model does not follow the new OMG’s CWM which was introduced in chapter 4. The CWM is an open
standard defining a common metamodel and XML-based interchange format for storing metadata in databases and data warehouses.

Therefore an important focus of this research has been to refine the reference manufacturing model using OMG's CWM and add new classes to support DM application areas and/or existing systems for SCM, ERP etc. A complete model based on CWM is presented in chapter 7.

However, this model could be restricted by constraints mainly as a result of several conditions involved by the criteria of integration adopted.

5.7 CONCLUDING REMARKS

This chapter suggests a new approach of enterprise modeling and integration especially by including explicit knowledge and mining views linked to the CIMOSA cube. To the ARIS should be added information, knowledge and mining views. The associated views are data, information, knowledge and mining views which create a novel approach to enterprise and mining modeling at conceptual level.

The main ideas around this approach are as follows:

- Knowledge discovery and data mining could be applied to the databases of a particular enterprise implemented as a result of CIMOSA, ARIS or GERAM models if these include additional views such as the suggested knowledge and mining views.
- Including a knowledge view at the generic and partial level of models involves the creation of a framework for applying KD/DM at particular level in a standard and unified way. Therefore in this situation generic data sets are used and knowledge which captures previous information about enterprise models. Also at this generic level, the framework does not deal with real data residing in databases, but instead only with their schema definitions.
- The purpose of this approach is to obtain a coherent data structure which supports KD/DM, but at a high-level of abstraction provides a lack in concrete system development and implemented solutions.
- The enterprise reference architectures include or have associated methodologies which could be applied to knowledge discovery at generic, and partial level, but not at the particular implementation level.
• Knowledge view contributes to the high level design of related knowledge engineering systems and tools supporting the extended enterprise.

• Enterprise modeling is supported by predictive models obtained through knowledge discovery and data mining. Predictive and descriptive models could evaluate the business aims, describe the present enterprise models and predict the enterprise’s future behaviour. However the enterprise modeling workbenches and software tools do not consider data mining.

A KD/DM distributed system adhering to RM-ODP completes the framework defined by CIMOSA, ARIS and GERAM. The UML diagrams for an extended common enterprise model based on Common Warehouse MetaModel (CWM) which is also an abstract model, are presented in the chapter 7.
Chapter 6

ARCHITECTURES OF DISTRIBUTED KNOWLEDGE DISCOVERY AND DATA MINING SYSTEMS

6.1 INTRODUCTION

Despite the fact that distributed computing has been used since the beginning of computer technology, the concepts are still evolving and new solutions are being developed. Therefore enterprise applications have been moved from distributed computing environments coded usually in C++ to web-based Internet environments using Java technologies such as applets, servlets, Java Server Pages (JSP), Java DataBase Connectivity (JDBC) and Enterprise JavaBeans (EJB).

The OMG middleware projects such as CORBA and MDA provide standard solutions for implementing distributed systems using object technology. Corba extends JavaTM with a distributed object infrastructure. Enterprise JavaBeans (EJB) enhance distributed computing by defining object-web solutions and server-side components included in a multi-tier architecture for enterprise systems. [Orfali et al., 1997], [Orfali, Harkey, 1998], [Vogel, Rangaro, 1999].

This chapter deals with the potential architectures to be defined within the unified object-oriented framework. The suggested architectures and their implementation solutions are used in manufacturing industry, but without any specification for data mining systems. Therefore especially the multi-tier architecture shown in figure 6.3 constitutes a contribution of this research. These architectures are part of the technological and engineering viewpoints of RM-ODP standard framework explained in section 5.5 and figure 5.4. However the multi-tier architecture and Model-Driven Architecture with its implementation solutions for distributed applications, based on EJB and Java DataBase Connectivity (JDBC) are still under research and in the development phase by system architects.
6.2 MULTI-TIER ARCHITECTURE

Distributed object and web-based component technologies enable distributed collaborative KD/DM supported by high performance client/server architectures, based on different hardware such as massively parallel computers and intelligent middleware. However in implementing client/server systems many difficulties arise especially regarding the connection of applications. Applications may be connected as depicted in figure 6.1. into a classical client/server system. In this architecture the code of the application called business logic is repeatedly written from the server into the client system memory, and it can be run using different databases [Liu, 2001]. This architecture is memory consuming, and has shortcomings in data security. Furthermore for data mining systems, replication of an algorithm implementation causes redundancy in the system and inefficient usage of processing time and memory. The multi-tier architecture offers clearly defined and separated interfaces and coordination between components. Such a system is defined as layered applications or a multi-tier architecture. The multi-tier approach offers a high-level of scalability and flexibility. Usually the first layer contains the presentation of the application which is also called user interface, or front-end. The business logic of the application resides in the second-tier and the third-tier is the data layer. A multi-tier architecture can usually be formed by three-tiers or four-tiers for complex applications.

![Figure 6.1 Classical Client/Server System](image_url)
The research reported in this thesis also uses CORBA infrastructure to connect objects as explained in section 6.3 and figure 6.3 and it provides a better separation and security between client and server components.

Figure 6.2 Multi-tier Architecture

A KD/DM multi-tier architecture has the following main characteristics:

- Data can be accessed from anywhere, mined on a local workstation and co-ordinated for the presentation of the results at the client.

- Data can be imported from any JDBC interface such as Oracle and Microsoft SQL Server and from other data sources such as a data warehouse or an OLAP database.

- Distributed knowledge processing capabilities.

- Facilities to integrate legacy applications and their databases using a wrapping technique based on creating object wrappers, by which the legacy system becomes a reusable component in a multi-tier system. An object wrapper is defined as an interface to an existing programme or component. An object wrapper is essentially a black-box transformation of legacy code providing a working interface to an existing program or program component. The wrapping layer can communicate with the
legacy system using a predefined Application Programming Interface (API) [Coyle, 2000].

- A multi-tier application can run on more than one server using a pre-defined application protocol.
- It has a high-level of scalability providing support to dynamically add or change components
- It could adapt a Corba distributed object system and/or a purely client/server application.
- It could be based on several Java technologies such as JDBC and EJB.
- It could adopt HTML, XML interfaces for data, information and mining models exchange.

A DM system supported by CORBA, MDA and EJB is integrated into a multi-tier architecture which has a distinct separation of its components. This multi-tier architecture is depicted in figure 6.2. Further investigation of multi-tier architectures is needed before they can be applied in manufacturing system design. Therefore this architecture should add a contribution to enterprise architectures as well as to distributed DM systems.

CORBA offers implementation solutions especially through the definition of a standardized language for interface development (OMG/IDL). MDA joins CORBA, XMI/XML, UML and CWM solutions which enable the development of new applications by using existing systems. Furthermore MDA clearly separates the system specification from implementation solutions. CORBA does not define a server-side component co-coordinator framework but this could be developed using EJB which is a higher-level component-based architecture for distributed applications that incorporate transaction systems. The relationship between Corba and EJB is explained below [Orfali, et al., 1997], [Orfali, Harkey, 1998], [Vogel, Rangaro, 1999]:

- EJB augments CORBA in relation to its transactional system, and by defining a server-side component framework, and deployment and security solutions;
- CORBA augments EJB with a distributed object environment.
6.3 CORBA/IDL APPLIED TO KNOWLEDGE DISCOVERY SYSTEMS

CORBA has been developed as an infrastructure for building systems involving distributed objects and it consists of the following main components [Farley, 1998], [Orfali, Harkey, 1997]:

- An Object Request Broker (ORB) which provides clients and servers of distributed objects with an environment to make and receive requests from each other. ORBs can also provide object services such as Naming services that let the client look-up objects by name, or Security Services that provide secure inter-object communications.

![Diagram of CORBA/IDL applied to DM Systems](image)

Figure 6.3 CORBA/IDL applied to DM Systems based on [Orfali, Harkey, 1997]

- Methods for specifying the interfaces that objects in the system support. These interfaces specify the operations that can be requested from the object and any data encapsulated in the object. There are two ways to define object interfaces:

  - **Interface Definition Language (IDL)** is used for the definition of static language- and platform-independent interfaces. IDL is a declarative language without implementation methods. The interfaces developed using IDL are
called stubs and skeletons for the object and they must conform to CORBA specifications. In practice Corba IDL-to-Java and Java-to-IDL it is widely used for automatic mutual mapping, and are implemented in software products such as Visibroker for Java, OrbixWeb, and JavaSoft’s Java2IDL. By using Corba IDL, it is possible to make existing code look like an object on ORB even if it is implemented in a different language or if it runs on a different platform.

- **Dynamic Invocation Interface (DII)** which provides the dynamic invocation methods for a run time object.

- A binary protocol for communication between ORBs, called Internet Inter-ORB Protocol (IIOP).

Corba/IDL has become a major technology for enterprise integration projects. Even though the adoption of this technology for knowledge systems included in extended enterprise is not done yet, and only a few data mining commercial tools may have CORBA interfaces.

6.4 CONCLUDING REMARKS

This chapter describes a multi-tier architecture for a KD/DM system which uses CORBA and EJB. UML, CORBA and CWM are defined at the heart of Model Driven Architecture. Therefore the system by default is adhering to OMG’s MDA. The UML diagrams for an extended enterprise common model based on CWM which is also an abstract model, are presented in the next chapter.
Chapter 7

A UNIFIED OBJECT-ORIENTED FRAMEWORK

7.1 EMBEDDED SYSTEMS

It has already been emphasized that the research reported in this thesis is focused on large-scale integration of dissimilar software systems running in an extended enterprise.

In an extended enterprise there are considered to be the following systems and databases:

• Complex software systems implemented to support Customer Relationship Management (CRM), Supply Chain Management (SCM) and Enterprise Resource Planning (ERP) applications. Examples of these systems are DecisionCraft, J.D. Edwards, MicroStrategy 7i, Baan ERP, Oracle Applications, PeopleSoft, SAP etc.;

• Knowledge Discovery and Data Mining products such as PolyAnalyst, Clementine, Weka, ArMiner etc.

• Manufacturing and products models stored in databases such as Oracle, Object Store, and associated database management systems and applications.

The integration is defined at a conceptual level by including the knowledge view, and also the concepts and associated solutions to design and develop embedded system are explained. Generally, an embedded system is a combination of hardware and software which has specific functions and capabilities, and it is designed for a particular or dedicated application. Initially the embedded systems have especially dedicated hardware and interfaces. Further the concept is used to define a high-level of integration for dedicated systems. In order to obtain this for the systems considered above, the following solutions are suggested:

• Including abstract views capturing knowledge and mining models called knowledge and mining views into an enterprise reference architecture such as CIMOSA, as shown in chapter 5.

• Using Application Programming Interfaces(API) for the systems included in extended enterprise. Some of these interfaces are already defined by the software vendors.

• Using a unified object-oriented framework as explained in this chapter. The unified framework considers, as an integration basis, data exchange and data sharing by dissimilar software products, and metadata and metamodel are defined.
7.2 METADATA AND METAMODEL

Generally, traditional KD/DM programmes, which implement complex algorithms and have been reported in the literature, do not intensively take advantage of UML capabilities for optimal system modeling, design and development.

Also, Knowledge Discovery and Data Mining (KD/DM) software systems and enterprise’s usual applications do not use the OMG’s Common Warehouse Metamodel (CWM) which provides a standard way to exchange and share data between dissimilar applications that are commonly run in an extended enterprise. Enterprise data mining applications are not different from other applications which require access to multiple data sources that are usually stored in databases. Therefore the integration could be approached at the following levels:

- Application Programming Interfaces(API) which have the aim to make dissimilar software systems run in the same environment;

![Figure 7.1 Metadata Concept](image)

- Integration based on data and information models which is approached in detail within this chapter. This approach is considered to be a key item for the successful inclusion of KD/DM systems into an extended enterprise.
The background of OMG's CWM in relation with other standards and projects is presented in chapter 4, and chapters 5 and 6 show its application in the enterprise engineering approach. The metadata concept depicted in figure 7.1 is crucial for large-scale enterprise systems integration because dissimilar software products use different formats of data. Figure 7.1 illustrates the metadata concept. This figure also demonstrates that mining techniques can be applied to any databases or files of data owned by, or available to a manufacturing system. The results of mining may lead to a formal model which captures the structure and relationships of data, or may be the identification of further (possible previously unknown) information about data. Alternatively, the results of mining may be a combination of these results. Any of these alternatives may be referred to Metadata. However the metadata concept does not capture knowledge. Mining models are incorporated in metadata which is defined as information about data or as formal models.

In order to interoperate effectively, software components of an extended enterprise must be capable of easily sharing data, and this is also important for mining processing. Sharing data requires a common definition and structure.

According to [Poole et al., 2002] metadata is also characterized by the following assumptions:

- A formal or generic model of an information structure is the metadata describing that information structure. The metadata is essentially a formal model of the data that it describes.
- Metadata, when expressed as a formal and platform-independent model, can be translated to any number of different platform-specific models, each representing a different target platform.
- Metadata when expressed as a formal and platform-independent model can exist outside of, and independently of any particular target platform.

Metadata integration is achieved through building metadata bridges which are software interfaces capable of translating data from one system into a format accepted by other systems.

The unified object-oriented framework depicted in figure 2.1 provides a flexible integration between the enterprise systems classified above. The classification has been considered especially to facilitate knowledge discovery and mining processing. It is also intended to conform to the standard reference architectures as explained in chapter 5. Therefore this
approach involves a set of restrictions and constraints which are intended to be solved by the modeling approach based on metadata and metamodel concepts defined within CWM. It is suggested that this large-scale integration across an extended enterprise can be achieved by using the unified framework described in detail in this chapter. It is obtained by using a metamodel for extended enterprise also called an extended enterprise model. OMG’s CWM is a specification and not an implementation, but it provides a generic model that is domain-independent which allows domain-specific models to be built [Poole et al., 2003]. These models form a basis for the design of an abstract integration architecture based on data.

The framework uses the reverse engineering capability of Rational Rose in order to generate a model based on a specific implementation, and this model can be merged into a current model [Quatrani, 2000].

The modeling has been accomplished in the following stages:

- Adding to the standard manufacturing information models additional classes to capture suitable data representation and information in order to facilitate knowledge discovery and data mining. This is partially explained in chapter 5.

- Creating UML diagrams describing a generic KD/DM application for extended enterprise.

- Analysing the class diagrams of CWM-DM specifications presented in chapter 4. In this research only the specifications of the model have been used and these have recently been realized as version 1.1 of CWM. For the purposes of this research, the model has been created in Rational Rose based on its initial specifications. CWM is domain-independent.

- Creating diagrams of a new domain-dependent metamodel for the manufacturing area which captures the manufacturing and product models, and application specifications and requirements. This model is called EE_CWM.

- The EE_CWM employs UML in order to specify data mining related objects, including model representations, model building settings and results of model operations. The unified object-oriented framework for application of DM in manufacturing can be used for further development and implementation solutions. EE_CWM defines a common basis to integrate dissimilar enterprise applications which can be described as follows:
➤ Static integration defines the static connections between the various systems which have specific interfaces;
➤ Dynamic integration defines the behaviour and software processes using a central repository of integrated data.

7.3 MODELLING OF A DATA MINING SYSTEM

The first stage in modeling and developing an application is to identify the functionality and behaviour of the system which is documented in a use case model that shows the system’s intended functions [Quatrani, 2000].

![Use Case Diagram](image)
Figure 7.2 shows the Use Case Diagram of the application of KD/DM in a manufacturing enterprise. The main role of this model is to communicate the way of applying KD/DM within manufacturing and to provide a common basis for end-users and developers. The defined *Actors* are as follows:

- **General Manager** who is responsible and interested in applying knowledge discovery and data mining in order to improve business performance, productivity and position in the marketplaces of the enterprise based on historical data stored in several databases. He is also interested to use the enterprise past information in reengineering processes and to optimize manufacturing systems design and production processes. The General Manager must be able to appreciate whether the data mining results are useful.

- **Business Analyst** who is responsible for applying business intelligence and analytical applications in an enterprise. Sometimes he/she is assisted by a knowledge worker and information consumers who are people monitoring the enterprise’s behaviour and its position in the market by examining several reports. A Business Analyst is in charge of evaluating how data mining improves business performance of the enterprise.

- **Manufacturing System Designer** who deals with manufacturing system design, reengineering processes, new product introduction etc. He/she has to be an expert in manufacturing engineering who could evaluate the results obtained using KD/DM and compare these with traditional data analysis methods.

- **Data Miner** is responsible for mining processing and building models. He/she has experience in data mining, data analysis and knowledge extraction, but he/she does not know in detail the specific areas of application related to manufacturing enterprises. Also he/she has abilities to use DM software systems and in developing new interfaces according to end-user requirements. Sometimes it is possible to work together with software developers part of a software company which develop DM products. Data Miner can offer consulting to an enterprise by showing hidden and unknown aspects incorporated in an enterprise database. This is usually called indirect data mining [Berry, Linoff, 2000]. The direct data mining is the retrieval of new information and knowledge in order to accomplish objectives of mining
activities. The establishment of the objectives is done by General_Manager, Business_Analyst and Manufacturing_System Designer.

- **DB_Admin** creates and manages the enterprise databases. He can also contribute to KD/DM by creating a data warehouse and/or a central data repository in order to facilitate DM.

The actors are persons responsible for the tasks described above, or Departments within an enterprise because KD/DM are very complex applications which require several people to be involved. The above Use Case Diagram is presented in a simple and understandable way to show the functionality of a DM application. The corresponding Activity Diagram presented in figure 7.3 shows the dynamics of the part of the system related to business intelligence of an enterprise.

![Figure 7.3 Activity Diagram (1) of Business Intelligence/DM](image-url)
The major areas of application have been identified as Customer Relationship Management, Supply Chain Management and Enterprise Resource Planning, and they have been presented in detail in chapter 3. These applications are described using Business_Intelligence_Areas_Description Activity or their identification could be obtained by applying reverse engineering to the software systems which support the above applications. A specific area with an associated data set is identified by the Activity of Identify_Specific Area and Select data.

Figure 7.4 Activity Diagram (2) for Product Development Life Cycle

Figure 7.3 illustrates the Activity Diagram (1) which contains the main activities such as Business_Intelligence_Area_Description, Identify_Specific Area, Pre-process Data, Mine Data, Evaluate the Results and Improve Business performance. It contains also the transitions between activities and decision points.

The Activity Diagram (2) presented in figure 7.4 contains the activities and corresponding transactions for the Product Development Life Cycle. These activities are product planning, design, manufacture and maintenance. Other activities related to market and sales can occur in parallel. Therefore this model generates product design and manufacturing models as well as a product business model based on customers needs. This Activity Diagram corresponds to Selected_Application as presented in figure 7.2 which
depicts the Use Case Diagram. This diagram also shows the place where information could be incorporated or captured.

![Figure 7.5 Collaboration Diagram](image)

Figure 7.5 depicts the associated collaboration diagram, and the sequence diagram presented in figure 7.6 shows the interactions between some of the objects arranged in time sequence.

In Figure 7.5, the MiningModel, BI Applications and Performance are visual representations of the objects which interacts with actors as follows: 1: General_Manager defines the need to explore BI_Applications and data mining; 2: Business_Analyst describes potential areas as BI Applications; 3: MiningModel improves the Performance; 4: Business_Analyst evaluates MiningModel; 5: General_Manager interprets MiningModel; 6: MS_Designer evaluates Performance of a manufacturing model incorporating mining model; 7: General_Manager evaluates the results of data mining; 8: General_Manager validates a data mining project and sets up the objectives.

In Figure 7.6 is illustrated the order of interactions in time sequence as follows: 1: Discussion between a General_Manager and a Business_Analyst; 2: Description of potential areas as BI_Applications for data mining; 3: Producing a Mining_Model; 4: Evaluation1; 5: Evaluation2; 6: Evaluation3 (4, 5 and 6 represent different stages of the Evaluation of a Mining_Model) 7: Improving enterprise’s Performance.

The class diagram presented in figure 7.7 shows a Generic DM application for an extended enterprise.
This diagram uses the classification of DM systems that is presented in the next chapter, in section 8.4. It also includes the software system PolyAnalyst™ as an instance of a DM software system that has an added Corba Interface. The class diagram presents the general view of the system which can include commercial KD/DM software systems such as PolyAnalyst™ or a dedicated system that has been specifically designed for the manufacturing area. This research is oriented to flexibly integrate dissimilar software systems using OMG's CWM. Therefore the extended enterprise models are presented in next paragraphs.

The model presented illustrates a useful and simple way to introduce and apply DM in an enterprise. However this model does not really incorporate the product and manufacturing models or any other input data from files, databases or data warehousing, and it does not adhere to any existing standards.
Figure 7.7 Class Diagram of a Generic DM Application
7.4 EXTENDED ENTERPRISE MODELS BASED ON CWM-DM

7.4.1 EXTENDED MAIN MODEL

![Diagram of CWM-DM Main Model](image)

Figure 7.8a CWM-DM Main Model [OMG, 2000]
Extended Main Model represents a comprehensive KD/DM model in combination with the classes describing ApplicationInputSpecification which is related to the identified application areas presented in chapter 3 as well as being represented in the class diagram depicted in figure 7.7. Figure 7.7 illustrates the ApplicationSpecification class which describes CRM, SCM, ERP and other applications. The OMG's CWM-DM main model is depicted in figure 7.8a. It consists of MiningModel defined as a representation of the mining model itself, MiningSettings, which is at the heart of the model, ApplicationInputSpecification, which specifies the set of input attributes for the model, and MiningModelResult, which represents the result set produced by the testing or application of a generated model.

The class SupervisedMiningModel extends MiningModel to include supervised learning such as classification and regression. Hence, this class requires a TargetAttribute which provides the correspondence between ApplicationAttribute and the obtained SupervisedMiningModel. For obtaining this model it is applied supervised learning to a data set which describes an Application identified by ApplicationInputSpecification and ApplicationAttribute.

The attribute function of MiningModel describes the DM function class (i.e. AssociationRules) whereas the attribute algorithms are used to specify the concrete algorithm (i.e. decisionTree). These attributes are identical to those in MiningSettings, and will be described later in more detail. This model also contains classes corresponding to the algorithms such as [OMG, 2000], [Poole et al., 2003]:

- StatisticsMiningModel: statistical models;
- AssociationRulesMiningModel: association rules model;
- SequentialMiningModel: sequential analysis model;
- SupervisedMiningModel for classification and numeric prediction;
- ClusteringMiningModel: clustering model.

Figure 7.8b shows the added classes including EE_ApplicationSpecification Class describing identified application areas or combinations of these. It also incorporates product, manufacturing and market models. The classes related to application areas are described below.
The **CRM Class** is defined based on the details presented in chapter 3. Generally it describes the set of applications which represent the process by which manufacturing organizations manage their interactions with customers. The set of sub-systems include and are not limited to the following [Berson et al., 2000]:

- **Customer retention.** Sophisticated customer-retention applications begin with modeling those customers who have defected to identify patterns that led to their defection. These models are then applied to the current customers to identify likely defectors so preventive actions can be initiated.

- **Sales and Customer Service:** In today’s highly competitive environment, superior customer service creates the sales leaders. When information is properly aggregated and delivered to front-line sales and services professionals, customer service is greatly enhanced. If customer information is available, rule-based applications can be employed to automatically recommend products. Applications such as market-basket analysis (analysis of transactional databases to find sets of items that appear frequently together in a single purchase) have already shown phenomenal gains in cross-selling ratios, floor and shelf layout and product placement improvements, and better layout and web pages.

- **Marketing:** Marketing depends heavily on accurate information to execute *retention campaigns*, lifetime value analysis, trending, targeted promotions etc. Promotions can be better targeted by having a complete customer profile, and targeting dramatically increases response rates and thus decreases campaign costs.

- **Risk Assessment and Fraud Detection** An accessible customer base significantly reduces the dangers of entering into undo risk.

CRM applications involve using information about customers and prospects to more effectively predict their interactions in different stages of the relationship. These stages are defined as the customer life cycle, and they are as follows:

- Acquiring customers;
- Increasing the value of customer;
- Retaining good customers.
Figure 7 8b Extended Main Model
The *SupplyChain (SCM) Class* describes the flow of products and information between distributed supply chains of the extended enterprises including procurement of materials, transformation of materials into finished product and distribution of that product to end customers. This class is a generic representation of distributed supply chain systems which are very complex. Generally supply chains are defined as networks of logistic and manufacturing activities starting with raw material sourcing and ending with the distribution of finished goods to markets. A complete supply chain system usually involves aggregated data from several companies and several production facilities. The performance of a supply chain for a given product-market critically depends on two interdependent strategic issues [Martel, Vankatadri, 1999]:

- the network structure, i.e. the number, location, mission, technology and capacity of the facilities of the company involved;
- the material flow planning and control processes used, i.e. the methods used to compute requirements, lot-sizes, safety-buffers etc. and to schedule operations for the various production/distribution stages in the network.

The *Production_Inventory Class* is an abstract model of the machines and production processes. This class is useful in exploring the productivity and the failure rate of production lines. Usually, *Production_Inventory* models link the production process to the demand process to ensure that shortages are kept at a low level and no excessive inventory is built up. The models vary in terms of characteristics of the production process and the response to shortages and the inventory review procedures.

*ERP(Enterprise Resource Planning) class* provides a complete description of an enterprise resource system. This class and associated sub-classes describe relationships between functional areas such as different departments, relationships with partners, suppliers and customers. According to [Botta-Genoulaz et al., 2001] ERP class has the following main characteristics:

- Modeling several basic processes with the goal of integrating information across the enterprise and eliminating complex, expensive links between different software systems.
- Creating a comprehensive image of the major business processes of an enterprise, such as customer order fulfillment and manufacturing.
• Ensuring in real time the update of all the information related to the different business functions of a company: purchasing orders, production orders, sale orders, inventory and product management. This updating is followed in relation with the other support functions like accounting, financial and human resource management.

• Attempting to integrate all departments and functions across an enterprise onto a global information system that can assist all those different departments' particular needs. Common components include Finance modules, Human Resources software, as well as manufacturing and logistics applications.

**EE_ApplicationSpecification class** has the main role of eliminating the redundant information which could appear in the classes described above. This class is a generic class describing metadata and associated metamodels generated by CRM, SCM or ERP depending what applications the enterprise run.

Using a one to one multiplicity link the class **ApplicationInputSpecification** defines the set of input attributes for the mining model.

### 7.4.2 EXTENDED SETTINGS MODEL

The **Settings Model** is presented in figure 7.11a and it defines **MiningSettings** and their usage relationships to the attributes of the input specification [OMG, 2000].

**MiningSettings** has four subclasses representing settings for different types of algorithms such as **StatisticsSettings**, **ClusteringSettings**, **SupervisedMiningSettings**, **AssociationRulesSettings** [Poole et al., 2003].

The **SupervisedMiningSettings** has further subclasses identified as **ClassificationSettings**, **RegressionSettings** and this is a **CostMatrix** which is defined for representing cost values associated with misclassifications. **AttributeUsageRelation** consists of attributes that further classify the usage of **MiningAttributes** by **MiningSettings** (i.e. relative weight). Several associations are also used to explicitly define requirements placed on attributes by certain subclasses of settings (i.e. **target**, **transactionId** and **itemId**).

**MiningDataSpecification** is the collection of mining attributes specifying the interpretation of input data attributes.

The **Extended Settings Model** incorporates the **Basic Manufacturing Model** as described in section 5.6. In addition to the classes described above, the
ExtendedEnterpriseStrategies and GenericProductData classes have been added. The complete model description is presented in figure 7.11b.

The ExtendedEnterpriseStrategies class is shown in figure 7.9 and is based on the following assumptions and definitions:

- An enterprise strategy is the definition of the generic modalities of manufacturing to assist the achievement of business objectives [Platts, Gregory, 1991], [Tidd, 1994]. Generally, manufacturing strategy is concerned with combining responsibility for resource management as well as achieving business requirements [Brown et al., 2000].

- The application of competitive strategy to manufacturing enterprises requires the formulation of a competitive manufacturing strategy which looks for positions in market where the enterprise can meet its long-term objectives and satisfy customer requirements. A competitive strategy involves also using the business to maximize the value and capabilities that distinguish an organization from its competitors [Porter, 1998].

- Manufacturing strategy could be defined from different perspectives [Wang, Bell, 1994]:
  - the role of a manufacturing strategy: is to describe the competitive leverage required of the manufacturing functions;
  - the aim of a manufacturing strategy: is to analyse the entire manufacturing functions relative to the ability to provide such leverage, on which task it then focuses each element of the manufacturing structure;
  - the use of a manufacturing strategy: is to allow the manufacturing structure to be managed, not just for the short-term, to control the operational details of cost, quality and delivery;
  - the implementation of a manufacturing strategy: is that it spells out an internally consistent set of structural decisions designed to forge manufacturing into a competitive weapon.

- In order to formulate a strategy for an extended enterprise it is necessary to define the basic manufacturing objectives which are quality, delivery, cost, flexibility and innovation as short-term objectives as well as long-term objectives.
Figure 7.9 ExtendedEnterpriseStrategies Class and associated classes

Figure 7.10 GenericProductData Class Diagram
Figure 7.11a CWM-DM Settings Model [OMG, 2000]
Figure 7.11b Extended Settings Model
The **ExtendedEnterpriseStrategies class** and its associated sub-classes which link strategic management and operations are presented in figure 7.9. These classes logically describe several concepts such as JIT (Just-in-time) and TQM (Total Quality Management), LP (Lean Production), MRPII (Manufacturing Resource Planning), ERP (Enterprise Resource Planning), FMS (Flexible Manufacturing Systems) etc. which capture the changes in the global economy and market places.

The **GenericProductData class** and its sub-classes describe the following items [Dorador, 2001], [Dorador, Young, 2000], [Harding, 1996], [Wortmann et al., 2001]:

- CAD geometrical product model including feature-oriented product description;
- STEP (Standard for the Exchange of Product Model Data) and Express product neutral model class which describes the neutral data format for the representation and exchange of product data;
- Product Data Management (PDM) representation.

This class is depicted in figure 7.10.

### 7.4.3 EXTENDED ATTRIBUTES MODEL

The **Extended Attributes Model** overlaps the **Attributes Model** which extends the class MiningAttribute used in Settings model. According to [OMG, 2000] the Attributes model defines two sub-classes of MiningAttribute: NumericAttribute and CategoricalAttribute as shown in figure 7.12. CategoryDescription class represents the category properties and values that either a CategoricalAttribute or OrdinalAttribute might possess, while CategoryHierarchy represents any taxonomy that a CategoricalAttribute might be associated with. This diagram is shown in figure 7.12.

The Attributes model contains variables for MiningDataSpecification. The model above implements a PMML interface for mining model exchange which has been added besides the OMG's CWM-DM attributes model.
Figure 7.12 Attributes Meta Model

7.5 CONCLUDING REMARKS

This chapter approaches in a new way the design of data mining applications which are to be included in extended enterprises. First of all it uses UML to describe a generic application and then the model is integrated with existing information models in manufacturing by using OMG's CWM-DM. This approach is completely new and it defines a unified object-oriented framework which provides a general and standard model for enterprise systems supporting knowledge discovery and data mining. The framework is also directed at developing an
enterprise architecture based on metadata and metamodel concepts according to the OMG standard CWM-DM. CWM enhances the metadata integration architecture by considering data exchange and sharing across enterprise applications as a crucial factor to achieve a large-scale integration. Data is also essential for knowledge discovery and data mining. This modeling approach which incorporates product and manufacturing models supports an embedded data mining system. Therefore this approach fills the following identified gaps in the reviewed research and literature:

- Using UML which enables complicated systems to be designed efficiently. Yet it is important to use UML if KD/DM systems are to be integrated with other information systems and models used in manufacturing industry. However UML also needs to be complemented with other models and standards in order to achieve a large-scale integration of dissimilar software products.

- OMG's CWM has been selected to accomplish enterprise systems integration based on the concept of metadata and metamodel. However, only the part of Common Warehouse MetaModel which addresses data mining (CWM-DM) has been used, and it should also be useful to make a model according to transformation package which covers data preprocessing. The OLAP package is useful for multidimensional data representation. Unifying the complete CWM with product and manufacturing models should create a complete extended enterprise model which can support a wide range of applications. OMG's CWM has only been recently released and it is not currently used by software vendors. In this research the specifications of the model have been used and the diagrams have been created in Rational Rose Enterprise Edition. The released version (V1.1) of CWM includes the packages description which can be imported in Rational Rose.

- Enterprise systems architecture adheres to the MDA which is a new approach for software application development and integration focused on data and business intelligence application views more than on implementation issues. The implementation solutions have been neglected because the research reported in this thesis is oriented to intensively use existing software systems as this is considered to represent the current situation in manufacturing enterprises more realistically.
• The unified object-oriented framework is defined as a development environment which is intended to leverage multiple enterprise software systems emphasizing knowledge discovery and data mining and their applications.

• The unified framework based on CWM covers data preprocessing by the application of Transformation package. Related to integration of data from different data sources, the unified framework does it by default because the application integration is based on data.

OMG's CWM will be implemented in Java but this version is not yet available for downloading. It will have the capabilities to generate the interchange format of the model in XML, and could have an IDL interface in order to access data through a CORBA interface. Software vendors such as Oracle, SAS etc. will need to deal with this model in order to incorporate it into their software products. Therefore, in the future if an Oracle database will be used to store product and manufacturing models they will automatically adhere to MDA and CWM. However, this is for the future as it is not done yet. There is also an additional area of integration that has not yet been considered by commercial software suppliers. This remaining gap relates to integrating dedicated enterprise systems such as Product Data Management (PDM) which could be integrated and deployed using this approach.
Chapter 8

SOME KD/DM IMPLEMENTATION SOLUTIONS

8.1 DISTRIBUTED KNOWLEDGE DISCOVERY AND DATA MINING SYSTEM SPECIFICATIONS

The essential components and requirements of a distributed KD/DM system embedded in extended enterprise have been identified through analysis of research projects and associated software systems which were mainly described in chapter 3. A specification list has been produced that is based also on the unified modeling and design framework explained in chapter 7, and the main elements of this specification are listed below:

- A distributed KD/DM System must be based on existing and emerging standards. It therefore is defined using UML methodology and is fully compatible with OMG’s MDA and CWM.
- It should be platform-independent.
- It should provide standard interfaces, i.e. CORBA/IDL.
- It should be possible to integrate the distributed KD/DM system with existing third-party software solutions (provided that they are based on an open system architecture)
- Input to DM applications must be possible from all type of files and information models that are currently in use in manufacturing enterprises.
- It should be possible to extend the distributed KD/DM system to accommodate standards such as XML, PMML etc.

8.2 AN IDENTIFIED KNOWLEDGE DISCOVERY AND DATA MINING SYSTEM

Experiments have been carried out with prototype systems to satisfy the requirements of the above specification. The main software components of the proposed distributed KD/DM system embedded in an extended enterprise have been identified, and are shown in figure 8.1. This figure shows a potential system architecture composed of the following modules:
• KD/DM Core Package which implements the main DM algorithms. However, within this research, two versions of Association Rules from different Java source libraries have been considered.

• KD/DM Interfaces are defined as API corresponding to different support languages used such as JAVA or C/C++. The system defines a CORBA/IDL interface for accessing a DM Server as explained in this chapter, and also a JAVA interface which implements CWM is designed.

• The dedicated databases which contain Manufacturing and Product Models are accessed through Java DataBase Connectivity (JDBC) and Open DataBase Connectivity (ODBC).

• Additional data and information models related to Extended Enterprise are stored in flat files.

• The dedicated data management systems could run on the EE Server.

Other architectural solutions could use RMI or EJB and other middleware. This is not specified for any particular Database Management System, but the JDeveloper part of Oracle toolkit has been used for demonstrating this research. The proposed architecture has the following main capabilities and properties:

1. It is composed of a KD/DM core package and associated interfaces.
2. It is based on UML methodology and the CWM and it is fully compatible with MDA;
3. It incorporates manufacturing and product models as described in chapters 3 and 6, and defines additional classes and associated packages related to extended enterprise strategies, manufacturing and product models, and business behaviour.
4. The UML description has been produced using Rational Rose Enterprise Edition.
5. It incorporates data mining source packages from Weka Environment, ARMiner system and DM software systems such as PolyAnalyst™.
6. It includes generated CORBA/IDL interfaces.
7. It is platform-independent being completely modeled in UML and corresponding interfaces depicted in figure 8.1 should be developed.
8. It is realized to show a clear separation between the different system levels of modeling and design, interfaces and application programming interface (API).
9. It can be extended in order to support several DM standards such as PMML, SQL/MM, OLE DB for DM, Java DM (JDM) etc.
It is realized to show a separation of data input stream from algorithm/programme specifications and defined as depicted in figures 7.7 and 7.11b. Also for the extended enterprise common model, EE_CWM, the classes ApplicationInputSpecification, and MiningDataSpecification, are separated from EE_ApplicationSpecification as described in chapter 7, figures 7.8b and 7.11b. The system could include specific application programming interfaces for manufacturing and product models as shown in figure 8.1.

The system could be implemented in Java or C/C++, based on the code generation capability of Rational Rose Enterprise Edition. However, this chapter mainly deals with Java implementation issues. The main advantages and disadvantages of using Java vs. C++ were presented in chapter 4.

Code generated and written in Java is also platform-independent and can run on any computer which implements the Java Virtual Machine.

KD/DM-Beans can be created using Enterprise JavaBeans as explained in chapters 4 and 9.

The system could incorporate as components several DM systems including a DM server (DMS) like the example at http://dms.irb.hr/. The software system PolyAnalyst™ has been evaluated and included in the KD/DM distributed software system with a designed CORBA/IDL interface DMS is an internet service for online data analysis based on knowledge induction. The programme allows a user to upload data to the server, where computations are performed and then the results are returned to the users. Internet browsers enable communication with the server in both directions. Data are analyzed by the Inductive Learning by Logic Minimization (ILLM) system, which was conceived and developed at the Laboratory for Information Systems, Department of Electronics, R. Boskovic Institute, Zagreb, Croatia [http://dms.irb.hr].

A specification has been defined for a proposed distributed KD/DM system with the above capabilities. The KD/DM system has been designed and modeled as shown in chapter 7, in figures 7.8b and 7.11b which propose a comprehensive extended enterprise model called EE_CWM.

In order to implement the whole system a comprehensive analysis of KD/DM software systems has been required and details of the main software systems considered follow.
Figure 8.1 KD/DM System Embedded in Extended Enterprise

KD/DM Core Package

KD/DM Interfaces

C/C++

C++/API

CWM/Java

Corba/IDL

Java/API

ManufacturingModel/ API

ProductModel/API

ExtendedEnterprise Strategies/API

Flat Files

JDBC

ODBC
8.3 DATA MINING SOURCE SYSTEMS

In order to experiment with integration and interfacing aspects of the proposed distributed KD/DM system it has been important to work with existing third-party DM software and systems. Several dissimilar sets of programs and systems have been considered, but the following have been found to be particularly useful.

*Waikato Environment for Knowledge Analysis (Weka)* developed at the Department of Computer Science from University of Waikato, New Zealand is a very flexible collection of source programmes implementing several machine learning and DM algorithms for solving real-world data mining problems. Its main strengths lie in the classification area, where all current machine learning approaches, and quite a few older ones have been implemented within a clean, object-oriented Java class hierarchy. Regression, Association Rules and Clustering algorithms have also been implemented. The algorithms can either be applied directly to a data set or called from a specific Java programme. For example the source code of the Classification programme can be modified by adding specific source code for product management. By this means it is possible to obtain a dedicated Classification system for product management. Usually PDM systems do not contain mining processing capabilities. Generally, Weka contains sub-systems for data pre-processing, classification, decision tree, regression, clustering, association rules and visualization. It is also well-suited for introducing new machine learning schemes [Witten, Frank, 2000]. In order to demonstrate the research described in this thesis, source programmes from Weka have been used for CORBA/IDL interface generation and the development of a bean using Enterprise JavaBeans.

It is also possible to run the Weka environment as a stand-alone system and use it for processing data mining experiments. It has the following components:

- *Weka Experiment Environment* which enables the user to create, run, modify and analyze experiments in a more convenient manner than is possible when processing the schemes individually.

- *Knowledge Explorer* module, which processes a flat file, a database, or a URL connection.

The Association Rules programme composed by Apriori and ItemSet is presented in Appendix and it has been used as support for components integration and interface generation.
ArTool represents a collection of algorithms and associated programmes for mining of association rules in binary databases. ARTool is a project developed at Department of Computer Science from University of Massachusetts Boston, USA, and it includes the ArMiner programme for Association Rules which has been used in order to demonstrate the research included in this thesis. ArMiner is a client/server application implemented in Java, and the whole programme is presented in Appendix. However this programme has been designed to process binary databases and it does not have a standard structure and architecture.

The algorithms for finding Association Rules have been presented in chapter 3, which is dedicated to literature review and to revealing the gaps in applied data mining in industry and past and present research projects. The identification of these gaps highlighted the need for an integration framework as proposed in this thesis.

8.4 ANALYSIS OF KNOWLEDGE DISCOVERY AND DATA MINING INTEGRATED SYSTEMS

The description of KD/DM Integrated Software Products and their classification is difficult especially because of the fact that there exist more than 100 such software systems. Finding a general classification criteria is not always possible. The classification has been required for the modeling and design of a generic DM application in UML, depicted in figure 7.7. The classification below is especially based on algorithm and corresponding domain specification.

a. Subject-oriented Systems which are dedicated application domain specific tools The most developed systems of this category are for analysis of financial markets based on the methods of technical analysis. Technical analysis represents a combination of different techniques for forecasting of prices dynamics and selecting the optimal structure of investment portfolio, based on various empirical models of the market behaviour. These methods range from very simple, for e.g. trend subtraction, to those having elaborate mathematical bases, such as fractal theory or spectral analysis. These systems usually provide a major advantage to the user as they operate in specific terms of the application area. These terms are very clear to traders and financial analysts in the case of dedicated system for financial markets. Often such
systems have special interfaces for loading financial data. There exist several subject-oriented systems based on technical analysis, and the most important are MetaStock (Equis International), SuperCharts (Omega Research), Candlestick Forecaster (IPTC), Wall Street Money (Market Arts).

For the manufacturing area there are not any domain specific systems.

b. Statistical Software Systems for which statistical methods are augmented by some elements of data mining. However, their main data analysis methods remain the classical statistical analysis such as correlation, regression, factor analyses and other techniques. Such systems cannot determine the form of hidden dependencies in data, and require the user to provide the hypotheses that are to be tested by the system. One of the major drawbacks of such systems is that they do not allow the data analysis to be performed by the user who does not have a thorough training in statistics. Usually these programmes are very complicated and special training is necessary. Another disadvantage of statistical systems is that during the data exploration the user has to perform several times a set of some elementary operations. Tools for automation of the process are either non-present, do not exist, or require programming in some internal language. Sometimes statistical packages are inefficient for solving complex real-world problems. Some examples of such systems are SAS with Enterprise Miner (SAS Institute), SPSS with Clementine (SPSS), Statgraphics (Statistical Graphics) and Statistica (Statsoft).

c. Software Systems Implementing classical AI and Data Mining Algorithms which are based on classification, association rules, clustering decision trees and sequential pattern finding. There exists several data mining software products of this type, each having different levels of performance. Therefore selecting the right system to achieve optimal mining processing is determined by several factors such as the performance of the implemented algorithm, functionality, friendly user-interfaces, data pre-processing and integration capabilities, accessibility to different databases and others. The best known DM systems are Clementine (Integral Solutions) associated with SPSS, Enterprise Miner associated with SAS, PolyAnalyt (Megaputer), DBMiner, Witness Miner (Lanner), WizSoft etc.

d. Software Systems Based on Evolutionary Programming such as fuzzy logic, genetic algorithms and neural networks. These implementations have been directed to
performant systems for forecasting, financial and market analysis, and medical data analysis. The best known systems are PolyAnalyst, NeuroShell etc.

c. Software Systems Implemented Regression and Support Vector Machines such as Knowledge Extraction Engine (KXEN) which provide predictive and descriptive data modeling for decision making support systems.

d. Visualization Tools which perform mining and visualization tasks associated with virtual reality processing capabilities, i.e. Cygrom(DataScope)

Even this classification does not include all the systems according to all criteria, it is concluded that the quantity and quality of existing systems is an important argument to demonstrate that the DM community and industry do not need new tools and algorithms, but more importantly that they need to find improved and standardized ways of applying DM technology. Also choosing appropriate DM tools is a key element for a successful application implementation.


It is also possible to conclude that PolyAnalyst™ is one of the best performing data analysis tool combined with a powerful data mining system. Besides the usual capabilities of a DM system it can explore arbitrary complex structures using some data primitives. [Kiselev, et al. 1998], http://www.megaputer.com/:

Also, this data mining software system supports the solution of tasks of using clustering, classifying, segmenting, predicting and explaining data as well as finding association rules in transactional data. PolyAnalyst™ implements the following data mining algorithms:

- Association rules, decision trees and classification using fuzzy logic;
- Find Laws algorithm predicts values of numerical variables and generates explicit symbolic rules that represent nonlinear dependencies in data;
- Nearest Neighbour algorithm uses a memory-based classification system, predicting the value of an attribute in a data record based on the proximity of this record to other data records and the best coefficients for the distance function are calculated with genetic algorithms;
- Polynet Predictor represents a neural network tool used in order to predict numerical attributes in large training data sets;
- Find Dependencies algorithm spots relationships in data and identifies the most influential attributes with respect to the target attribute;
- Regression which is implemented as stepwise linear regression and logistic regression;
- Clustering is based on an unsupervised data mining algorithm;
- Summary statistics;
- Market Basket Analysis represents the examination of transactional data in order to determine which items are most frequently purchased together. Market basket analysis can run together with association rules, and in this situation, the aim is to discover patterns across a large number of transactions, and to identify buying patterns. The further application of the market basket analysis and association rules is to forecast the future transactions based on the current transactions.

The above algorithms can be accessed via a menu system in the PolyAnalyst™ interface, as shown in figure 8.2. The menu shown in figure 8.2 also illustrates the exploration of data using the algorithms implemented in the PolyAnalyst™ and the possibility of doing a Basket Analysis (BA) or a Summary Statistics (SS).

The algorithms implemented in the PolyAnalyst™ system are defined according to the machine learning technique. With a combination of these exploration engines and the simple overviews from the Summary Statistics engine, the user can derive previously unknown knowledge from quantities of raw data. Each of these algorithms has a different purpose and produces different results using a different method. As a result, to perform an optimally...
useful analysis with PolyAnalyst™, it is vital to know about the inputs and outputs of each machine learning algorithm. However, due to this variety of approaches, PolyAnalyst™ can produce far better results than other data mining tools that implement only one or two machine learning algorithms, such as neural networks or decision trees.

Also, PolyAnalyst™ includes model application to external data in any format through a standard protocol for processing large databases, output of created models in XML/PMML format and web mining capabilities.

However, PolyAnalyst™ is not based on OMG’s CWM and does not have a CORBA/IDL interface.

Therefore PolyAnalyst™ has been chosen in order to demonstrate the research included in this thesis for the following reasons:

- The implemented algorithms cover specific applications in manufacturing area;
- Lack of Corba/IDL interfaces, other OMG’s standards and models even it creates models in XML/PMML which do not satisfy the integration requirements of legacy systems;
- Lack of large-scale integration with enterprise systems for CRM, ERP and SCM which usually may use only one or two algorithms for data analysis.

8.5 CONCLUDING REMARKS

The research reported in this thesis deals deeply with different solution approaches at high-level and at implementation level. The implementation level is presented in this chapter and in chapter 9. It includes the definition of the specifications for a distributed KD/DM system embedded in extended enterprise. The description of its capabilities and properties are also shown. This chapter presents a classification of KD/DM software systems used for modeling and design of generic applications. The Java source programmes and software packages used to carry out the experiments are also dealt with.
Chapter 9

DEMONSTRATING THE RESEARCH

9.1 GENERATED JAVA™ SOURCE CODE FOR EE_CWM

Rational Rose Java provides the following facilities for code generation [Rational Co., 2000]:

- Design, modeling and visualization of Java constructs, including packages, classes, interfaces, imports, inheritance, fields, methods and modifiers.
- Automatic synchronization between the model and the code.
- Automatic generation of Java source code from a model.
- Integration with VisualAge for Java, JBuilder, Forte for Java and Visual Cafe.
- Possibilities to include java classes, zip, jar and cab files into a class or component diagram for reverse-engineering.
- User-controlled generation of default constructors, finalizers and static initializers as well as user-defined generation of field-name prefixes.
- Component mapping on code generation.
- Bean property generation.
- Support for modeling J2EE, Enterprise JavaBeans and Servlets.
- Support for large frameworks including: JDK Class Library, Java Database Connectivity (JDBC), Application Foundation Classes (AFC), Windows Foundation Classes, Enterprise Java Frameworks (JNI, EJB, JTS)

Java source generation is based on the component specification rather than on the class specification. In order to realize Java source code, after diagrams for classes have been created, every class is assigned a valid Java component. In practice the following methods are used [Rational Co., 2000]:

1. Generating Java source from Class Diagram;
2. Generating Java source from a Component Diagram

Some generated declarations for the extended model are presented below:

```java
public class ExtendedEnterpriseStrategies {
```
public MarketStrategies theMarketStrategies;
public ManufacturingStrategies theManufacturingStrategies;

public ExtendedEnterpriseStrategies()
{
}

//Source file: D:\JAVA\BIN\CostMatrix.java

public class CostMatrix
{
    public ClassificationSettings theClassificationSettings,
    public CostMatrix()
    {
    }
}
//Source file: D:\JAVA\BIN\RegressionSettings.java

public class RegressionSettings
{
    public SupervisedMiningSettings theSupervisedMiningSettings;
    public RegressionSettings()
    {
    }
    public AlgorithmSpecification theAlgorithmSpecification,
    public SupervisedMiningSettings theSupervisedMiningSettings,
    public CostMatrix theCostMatrix;
    public ClassificationSettings()
    {
    }
}
//Source file: D:\JAVA\BIN\StatisticsSettings.java

public class StatisticsSettings extends MiningSettings
{
    public MiningSettings theMiningSettings;
    public StatisticsSettings()
    {
    }
}
//Source file: D:\JAVA\BIN\MiningAttribute.java

public class MiningAttribute
{
    public StatisticsSettings theStatisticsSettings;
    public AttributeUsageRelation AttributeUsage[];
    public MiningAttribute()
Having these class definitions in Java makes it possible to call a Java source programme as follows:
Download the source programme (.java) or the compiled version (.class) and make the link with the package or components described by the class diagram.

9.2 GENERATED CORBA/IDL INTERFACES

CORBA/IDL interfaces generation could be done at the level of system modeling and design using Rational Rose before creating the model or by assigning every class to a component whose language is set to Corba. The screen supporting this option is shown in figure 9.1.

![Figure 9.1 Rational Rose Corba Generation](image)

Generally, Rational Rose provides two standard mechanisms for controlling the model elements and code generation [Rational Co., 2000]:

- Model properties, which provide global settings for a project and its classes, attributes, operations, components and roles;
- Specifications, which control individual classes, attributes, operations, components and roles;

Rational Rose CORBA also provides a set of context-based custom specifications which extract settings from both the Rational Rose specifications and the model property settings, enabling the display and modification of certain CORBA-specific information. Custom CORBA specifications are available for project, class, attribute, operation and component.
For the logical diagram defining the classes of a Generic DM Application, presented in chapter 7, figure 7.7, the following interfaces have been generated:

Source file: C:\Program Files\Rational\Rose\CORBA\MyCorbaInterface.idl
#ifndef MYCORBAINTERFACE_DEFINED
#define __MYCORBAINTERFACE_DEFINED
/* CmIdntification
  %X%%Q%Z%W% */
#endif

Source file: C:\Program Files\Rational\Rose\CORBA\PolyAnalyst.idl
#ifndef POLYANALYST_DEFINED
#define __POLYANALYST_DEFINED
/* CmIdntification
  %X%%Q%Z%W% */
#endif

CORBA objects defined in IDL can be mapped to Java classes and interfaces using IDL-to-Java compiler [Orfali, et al., 1997], [Orfali, Harkey, 1998]. Generating Corba/IDL interfaces using the JDeveloper environment is based on Java™ IDL technology for distributed systems which enables objects interacting on different platforms across a network. Java IDL is similar to RMI, which supports distributed objects written entirely in the Java. However, Java IDL enables objects to interact regardless of whether they are written in the Java programming language or another language such as C, C++, COBOL or others [Orfali, Harkey, 1998]. This feature is very useful for a data mining system processing a legacy database.

Figure 9.2 shows the basic one-method of distributed objects shared between a CORBA client running a DM programme, accessing the database on the server. On the client side, the application stand-in for the
method is being called remotely. The stub is wired into the ORB, so that calling it invokes the ORB's connection capabilities, which forwards the invocation to the server. On the server side, the ORB uses skeleton code to translate the remote invocation into a method call on the local object. The skeleton translates the call and any parameters to their implementation-specific format and calls the method being invoked. When the method returns, the skeleton code transforms results or errors and sends them back to the client via the ORBs.

Between the ORBs, communication proceeds by means of a shared protocol, IIOP--the Internet Inter-ORB Protocol. IIOP, which is based on the standard TCP/IP Internet protocol, defines how CORBA-compliant ORBs pass information back and forth.

In addition, CORBA-compliant ORBs can provide a number of optional services defined by the OMG. These include services for looking up objects by name, maintaining persistent objects, supporting transaction processing, enabling messaging and many other abilities useful in the distributed, multi-tiered computing architecture presented in chapter 6.

The communication between the client and the server is presented as follows:

1. The Java client which is an applet or an application invokes a specific operation on the data stored on the server database.
2. The ORB transfers that invocation to the server object registered for that IDL interface.
3. The server method runs, returning an initial value for data.
4. The ORB transfers the data back to the client.
5. The client subsequently processes the data.

The generation process using JDeveloper is shown in figure 9.3. The following declarations have been obtained:

```java
//Company:   LU
//Description: 
package com;
public interface MyCORBAInterface extends org.omg.CORBA.Object {
    ...
}
package Aprion;
public interface MyCORBAInterface extends org.omg.CORBA.Object {
    ...
}
package Aprion;
/**
<p>
</p>
<ul>
<li> <b>Java Class</b> Apriori._example8i_MyCORBAInterface
```
Source File: Apriori_example8i_MyCORBAInterface.java

IDL Absolute Name: ::Apriori::MyCORBAInterface

Repository Identifier: IDL:Apriori/MyCORBAInterface:1.0

IDL definition:

```java
#pragma prefix "Apriori"

interface MyCORBAInterface {
};
```

```java
public class _example8i_MyCORBAInterface extends Apriori._MyCORBAInterfaceImpIBase implements oracle.aurora.AuroraServices.ActivatableObject {
    /**
     * Construct a persistently named object.
     */
    public _example8i_MyCORBAInterface(java.lang.String name) {
        super(name);
    }
}
```

Figure 9.3 Corba Generation using JDeveloper
The packages used for generation of the Corba project are AssociationRules and Apriori which are presented in Appendix.

9.3 CREATING ENTERPRISE JAVABEANS CLASSES

Enterprise JavaBeans link the whole Java technologies such as RMI, JDBC, JNDI and OMG's CORBA into an enterprise common system. EJB technology defines a model for the development and deployment of reusable Java server components as explained in section 4.5.1. Components are pre-developed pieces of application code that can be assembled into an enterprise application system. EJB is a technology which enhances the application server capabilities by implementing clearly separated components. Therefore if the EE_CWM classes are considered to define an application then it can be deployed as an EJB on a server. The components can be considered at different level of integration. Figure 9.4 depicts EJB classes generated by Rational Rose.

EJBs use Deployment Descriptors as a way for a container to determine the EJB persistence, security and transaction information. Rose Java derives this information from the EJB Specification and the Project Specification.

Rose Java automatically generates the Deployment Descriptor for a XML interface <ejbName>.xml (or <servletName>.xml file when you generate code for the EJB Implementation or Servlet class.

The Rose EJB Specification can define the Container Manager containing the following beans:

- **Session Bean** which can be Stateless (each call to the bean's methods is independent of the following call) or Stateful (a client has exclusive use of the bean's methods and maintains state between method calls);

- **Entity Bean**;

- **Message-Driven Bean**.
The Code Generation facility allows users to customize the way Rose generates code for the modeled Java classes. However the complete implementation of an application using EJB is as follows [Vogel, Rangarao, 1999]:

- Write remote and home interfaces, which define the methods of creating, finding and interacting with an EJB.
- Write the main EJB class.
- Write the deployment descriptor component.
- Compile EJB classes and interfaces.
Figure 9.4 Extended Enterprise Common Model using EJB
Figure 9.5 shows the artefacts and their dependencies involved in using a MDA approach based on EJB. The business logic of the application is specified in the platform independent model by using the UML standard. This figure shows that the Platform Specific Model of Application is obtained by mapping the application described in UML for EJB as shown in figure 9.4. The standard UML can be automatically mapped to the UML class diagram which incorporates EJB classes, and then generates a platform specific model. The Platform Specific Model is a base to generate Java class descriptions which together with the deployment descriptor are included in the EJB Container and used to join the components of the application.

9.4 CONCLUDING REMARKS

This chapter attempts to demonstrate the research by presenting the process of generation of Java source code and Corba/IDL interfaces. In addition the Enterprise JavaBeans class diagram for Extended Enterprise Common Warehouse MetaModel (EE_CWM) is shown. The MDA approach for EJB could be applied for all UML class diagrams. The practical approach included in this chapter is in line with the latest software engineering trends related to using distributed component technologies. However in computer engineering software development while significant advances in building reliable, scalable and secure system have been made there is still a long way to progress until it will be possible to obtain a component-based distributed system for a given set of requirements in a systematic manner.
Chapter 10

CONCLUSIONS

This chapter concludes the research which has been carried out and discussed in the previous chapters. Firstly this chapter provides a brief summary of all the previous chapters. The research contributions described in detail in chapter 2, are outlined. Finally, the conclusions are drawn against the research objectives set in chapter 1.

10.1 SUMMARY OF THE THESIS

The main purpose of this research is to make knowledge discovery and data mining more accessible to manufacturing industry, and to bridge the gap between their theoretical approaches regarding advanced algorithms, their implementations and their applications. The research adds new elements in modeling of an enterprise and its information and knowledge to better support business intelligence application in manufacturing industry. The modeling approach uses a Common Warehouse Metamodel developed by OMG which is directed to integrate the existing enterprise systems, and their associated data and models.

Although knowledge discovery and data mining were clearly defined at the beginning of '90s, there are still open topics, and the most important issues are related to standardization, integration, portability and scalability of these systems. On the other hand research into the extended enterprise engineering approach is far from exhausted even though the research reported in this thesis takes into consideration the earlier enterprise modeling and integration issues because they are of an outstanding value, reference, and very useful. Furthermore it is stated that everything old could be new again because the perspective is always changing.

The literature review reveals the gaps between knowledge discovery and data mining theories and their applications. The shortcomings of the previous and current research projects have been identified in the context of data mining, knowledge discovery and extended enterprise.
The background of the research includes the description of basic concepts regarding a software architecture and a framework, the principle of modeling and the OMG standards and middleware projects. The Java capabilities for enterprise application have been presented in order to explain the selection of this language for implementation purposes. Enterprise JavaBeans are introduced as having the main scope to build distributed applications using components from different software providers. Using Enterprise JavaBeans means that it is possible to link the whole family of other Java technologies such as RMI, JDBC, JNDI and OMG’s CORBA into an enterprise common system.

The research described in this thesis takes into consideration both sides of the extended enterprise as the integrated enterprise and the information systems integration within the enterprise. This approach generates constraints which are dealt with.

The research reported in this thesis suggests a multiple views approach of KD/DM system development considering the previous enterprise models, architectures and standards. Knowledge discovery and data mining standards are also considered. The third chapter includes a comprehensive description of Supply Chain, Customer Relation Management and Enterprise Resource Planning systems which have been identified as major areas for application of KD/DM in manufacturing industry.

The novelty of this research is approaching both knowledge discovery and enterprise engineering at a conceptual level as a whole, and demonstrating the potential of the application of data mining in manufacturing. The contributions detailed in chapter 2 embrace interdisciplinary areas of knowledge discovery and enterprise engineering. Furthermore though there are several knowledge based enterprise approaches there is not any research joining the field of information retrieval using data mining within the extended enterprise. The research clearly suggests that in order to include data mining within an enterprise reference architecture such as CIMOSA, ARIS etc additional views defined as information, knowledge and mining views are necessary to improve the enterprise performance supported by business intelligence applications. A common enterprise and mining modeling solution is discussed.

The research also defines a unified object-oriented framework which provides a general and standard model for enterprise systems supporting knowledge discovery and data mining. The framework is also directed to design an integrated enterprise architecture based on the concepts of metadata and metamodeling using OMG’s MDA and CWM-DM.
The implementation level deals with the specifications, capabilities and properties for a distributed KD/DM system embedded in extended enterprise fully modeled in UML. The implementation level is demonstrated by generation of Java source code for an Extended Enterprise Common Warehouse MetaModel. A set of Corba/IDL interfaces have been generated at the modeling level as well as implementation level. In addition Enterprise JavaBeans class diagram for Extended Enterprise Common Warehouse MetaModel is shown.

It is hoped that the research reported in this thesis meets almost all of the objectives stated in chapter 1 as follows.

- The initial assumption is that mining databases of an extended enterprise should improve business performance and provide optimal solutions for manufacturing systems re-design, enterprise re-engineering, new product introduction etc. However knowledge discovery and data mining techniques are not currently sufficiently exploited in manufacturing industry.

- This research clearly shows that there are not many applications of data mining in manufacturing compared especially with financial and banking industry, and medicine. This is shown in chapter 3, sections 3.1.4, 3.2.1 and 3.3.

- The gaps in current research and projects regarding integration aspects from the perspective of knowledge discovery and data mining and extended enterprise approaches have also been shown in sections 3.1.4, 3.3 and 5.1.

- An integrated approach of enterprise engineering issues and knowledge discovery by the definition of a unified object-oriented framework for modeling and design of data mining applications has been proposed. The proposed framework provides a good basis for the correlation of existing enterprise reference architectures, dedicated software systems for CRM, SCM, ERP and data mining tools. However this framework cannot solve all the identified gaps and challenges. The framework produces a complex Extended Enterprise Common Warehouse MetaModel which could be difficult to apply and implement. Therefore investigations should follow related to the implementation issues using this environment in order to keep its coherence. Usually a framework provides only the general guidelines which cannot solve real implementation solutions even though it has been attempted to avoid this aspect by using OMG’s CWM and an application view.
The research reported in this thesis attempts also to take into consideration the disadvantages of including knowledge discovery and data mining into an enterprise high level architecture. The disadvantages could be multiple data processing and the potential new information and knowledge may be useful such as list of rules which do not make sense for a business analyst or a manufacturing system designer, high cost of good data mining tools, data warehouse organization and perhaps others.

The suggested unified information modeling approach in manufacturing is based on the latest development issues of OMG's and is directed to obtain a standard common enterprise model useful in extended/global enterprise approaches. The success of MDA and CWM is based on the willingness of major software vendors to adopt these models. Therefore using these models for information modeling in manufacturing creates the background for the development and implementation of advanced dedicated enterprise applications. The unified framework provided is flexible enough to optionally include knowledge discovery and data mining modules.

The methodological approach of data mining in extended enterprise could be achieved by applying the same methodologies as for enterprise modeling.

10.2 DISCUSSION AND FURTHER RESEARCH DIRECTIONS

The general contribution of this work is related to the areas of enterprise engineering as well as knowledge discovery and it is achieved by a novel and comprehensive approach of the topics involved such as standardization of data mining application in manufacturing industry, extended enterprise, reference architectures, Supply Chain Management, Customer Relationship Management and Enterprise Resource Planning.

However the proposed unified object-oriented framework does not have a complete original implementation solution for the following reasons:

- The research itself intends to use previous projects, enterprise applications and knowledge discovery systems.
- UML methodology recommends that a very comprehensive analysis and design of the system using standarization methods should be carried out before writing code.
- Even though this research intends to bridge the gap between high level architecture definition and a particular implementation using multiple views approach and
especially the application view. There still remains open topics such as component-based architecture.

For extended enterprises information retrieval and refinement by web-mining is essential, but this is different to knowledge discovery and data mining, and it is not discussed in this thesis.

**Research Directions** derived from this thesis:

- Investigating web mining and applying it for extended enterprise information and knowledge retrieval.
- Extending information modeling in manufacturing using Common Warehouse MetaModel, and applying other packages such as OLAP and XML.
- Implementation of a dedicated knowledge discovery system in manufacturing area according to the design methodology suggested in this thesis.
- Parallel object-oriented databases and associated systems for information extraction through data mining.
- Assimilation of new standards such as Java Data Mining (JDM) and Oracle9i Data Mining which is an option to Oracle9i Database Enterprise Edition that embeds data mining functionality for making classifications, predictions and associations. The model-building and scoring functions are accessible through a Java-based API.
- Applying Common Warehouse MetaModel V1.1 which is slightly different to the specifications of the model used in this research, but it will have an implementation version in Java.
- E-supply chain in manufacturing and associated support systems.

10.3 FINAL THOUGHTS

"Discovery consists of seeing what everybody has seen and thinking what nobody has thought."
- Albert von Szent-Gyorgyi

"It is by intuition that we discover and by logic we prove."
- Henri Poincaré

Isaac Newton (1642 - 1727) has said "If I have seen further it is by standing on the shoulders of giants."
This research used and attempted to develop and demonstrate new ideas based on several researches accomplished by outstanding people involved in manufacturing, artificial intelligence, software engineering and business research. I therefore realize that it should bring me a lot of happiness if the modest contributions reported in this thesis should be a beginning of further investigations, new research directions, successful projects and controversial debates.
REFERENCES

FRAMEWORK FOR DISTRIBUTED KNOWLEDGE DISCOVERY SYSTEMS EMBEDDED IN EXTENDED ENTERPRISE


[Han, 1998] Han, J., "Data Mining: An Overview from a Database Perspective", Tutorial Outline, PAKDD Conference, Simon Fraser University, Canada, 1998, at: http://kais.mines.edu/~xwu/icdm/icdm-01.html

[Han, Kamber, 2001] Han, J., Kamber, M. "Data Mining: Concepts and Techniques" Morgan Kaufman.


FRAMESWORK FOR DISTRIBUTED KNOWLEDGE DISCOVERY SYSTEMS EMBEDDED IN EXTENDED ENTERPRISE


GLOSSARY

**Actor** is someone or something external to a system that must interact with the system under development [Quatrani, 2000].

**Analytic Application** is a software system that meets three distinct conditions: process support, separation of function, time-oriented and integrated data. Analytic applications expand the reach of business intelligence to an extended user base, packaging these technologies into a business context [DM Review Newsletter, 2003].

**Application Programming Interface (API)** is a formally defined programming language interface between a program (system control program or commercial tool) and an end-user application or a user.

**Apriori** is a classical association rule mining algorithm which finds all association rules in a database whose support is greater than the minimum support and whose confidence is greater than a minimum confidence.

**Architecture** is a finite set of interrelated components put together to form a consistent whole defined by its functionality [Vernadat, 1996].

**Architecture of Integrated Information Systems (ARIS)** is similar to CIMOSA reference architecture, but instead of focusing on manufacturing systems, it deals more with business-oriented issues of an enterprise and the corresponding information systems.

**Association Rules** is a mining algorithm used to search the data for patterns of the following form: \( \text{IF} \left( \text{some conditions are true} \right) \, \text{THEN} \, \left( \text{some other conditions is probably true} \right) \)

**Bean** is a reusable software component that can be visually manipulated in a framework, see also Enterprise JavaBeans (EJB).

**Bullwhip effect** called also supply chain distortion represents the discrepancy between some of the parameters of supply chain systems such as the number of products to be manufactured, the quantity of raw material needed and the market requirements [Martel, Vankatadri, 1999].

**Business Intelligence (BI)** is a global concept describing the processes, techniques and support tools that facilitate decision-making and customer relationships management systems [DM Review Newsletter, 2003].

**Business-to-Business (B2B)** is an aspect of e-commerce dealing with advanced transaction systems using Internet technologies.
Business-to-Consumer (B2C) is an aspect of e-commerce dealing with buying and selling products and services to individual customers using Internet technologies.

Case Based Reasoning (CBR) or Case Based Learning is the process of acquiring knowledge represented by cases using the mechanism of reasoning by analogy.

Central Repository is defined as a data warehouse or other centralized store of operational data.

Chi-Square Automatic Interaction Detector (CHAID) is a decision tree algorithm that uses contingency tables and the chi-square test to create the tree.

Class is the description of a group of object with common properties (attributes), common behaviour (operations), common relationships to other objects (associations and aggregations) and common semantics [Quatrani, 2000].

Classification is the process of examining known groups of data to determine which characteristics can be used to identify (or predict) group membership [Helberg, 2002].

Classification-rule learning involves finding rules or decision trees that partition given data into predefined classes.

Clustering is a data mining algorithm for the identification of groups of similar cases, but it does not predict outcomes or target categories [Helberg, 2002].

C4.5 is a data mining algorithm which generates a classification-decision tree for the given data-set by recursive partitioning of data.

Component Object Model (COM) is defined by Microsoft as a well-defined, mature, stable and freely available specification as well as a reference implementation, which has been widely tested and adopted worldwide as a de-facto standard.

Common Object Request Broker Architecture (CORBA) is defined as an infrastructure for building systems involving distributed objects.

Common Warehouse Metamodel (CWM) is defined as a set of standard interfaces that can be used to enable easy interchange of warehouse and business intelligence metadata between warehouse tools, warehouse platforms and warehouse metadata repositories in distributed heterogeneous environments.

http://www.omg.org/technology/documents/modeling_spec_catalog.htm#CWM.

Component(s) are pre-developed pieces of application code that can be assembled into an enterprise application system.
Computer Integrated Manufacturing – Open System Architecture (CIMOSA) is an enterprise architectural framework to support enterprise integration and modeling developed by the AMICE Consortium within ESPRIT Projects 688, 5288 and 7110.

Confidence describes how likely the rule is to give a correct prediction. It is in the range between 0 and 1 (0% and 100%).

Container is defined and developed as a Java component which includes and organizes other Java components such as Abstract Windowing Toolkit (AWT) or EJB components.

Coverage describes how much evidence is in the training data set to back up the rule. It usually ranges between 0 and 1 (0% and 100%).

Cross-Industry Standard Process for Data Mining (CRISP-DM™) is a data mining project which has developed an industry- and tool-neutral data mining methodology [Helberg, 2002].

Customer Relationship Management (CRM) is a comprehensive approach and concept, which provides seamless integration of several areas of business that are related to customer modeling, customer behaviour and interactions.

Data is a collection of unanalyzed observations of worldly events [Pyle, 2003].

Data Mart is a particular data subsets focused on selected subjects such as a marketing data mart may include customer, product and sales information.

Data Mining (DM) is a complex process applied to any type and amount of data and supported by several methods, techniques and algorithms of statistics, machine learning and affective intelligence.

Data warehouse (DW) is defined as extraction and integration of data from multiple sources and legacy systems, in an effective and efficient manner [Harding, Yu, 1999].

Decision trees are a class of data mining and statistical methods based on building trees consisting of nodes and branches, starting from a single root node.

Decision making is the process of making a decision assisted by artificial intelligence support systems, but it is demonstrated that the existence of emotions is also very helpful in this process.

Descriptive model describes patterns in existing data, and is generally used to create meaningful subgroups.

Distributed Component Object Model (DCOM) is defined by Microsoft as a protocol that enables software components to communicate directly over a network in a reliable, secure and efficient manner. DCOM is designed for use across multiple network transports,
including Internet protocols such as HTTP and it is based on the Open Software Foundation’s DCE-RPC specifications and it works with both Java applets and ActiveX® components through its use of the Component Object Model (COM).

**E-business** deals with conducting business on the Internet.

**E-commerce** is an aspect of e-business dealing with buying and selling of goods and services on the Internet, composed by business-to-business (B2B), business-to-consumer (B2C) and electronic data interchange (EDI) such as online catalogues, books etc.

**Enterprise modeling** is the representation of an enterprise which can be composed of several submodels including but not limited to process models, data models, resource models and organizational models [Vernadat, 1996].

**Enterprise Resource Planning (ERP)** systems are complex programmes that link all the enterprise functions and associated departments such as manufacturing, finance, sales and human resources [Ayers, 2001].

**Enterprise JavaBeans (EJB)** is a component architecture for creating scalable, multi-tier, distributed applications, and it makes possible the creation of dynamically-extensible application server.

**Embedded system** is a combination of hardware and software which has specific functions and capabilities, and it is designed for a particular or dedicated application.

**Extended Enterprise (EE)** is described as a long-term co-operation and partnership based on information and knowledge exchange [Szegheo, 1999].

**Extended Enterprise Common Warehouse MetaModel (EE_CWM)** is a common enterprise model developed using OMG’s CWM and information models in manufacturing.

**Extensible Markup Language (XML)** is designed to improve the functionality of the Web by providing more flexible and adaptable information identification using an extensible format or a language for describing other languages which facilitate the design of own customized markup languages for different types of documents.

**Feedforward neural networks** are NN with training patterns composed of following two parts an input vector and an output vector, associated with the input and output nodes respectively.

**Framework** is the specification of a partial and/or a general development and implementation roadmap for generic or domain-dependent systems. Therefore within a framework could be defined several architecture.
Genetic algorithm is a method for solving optimization problems using parallel search, based on Darwin's biological model of natural selection.

Generalized Enterprise Reference Architecture and Methodology (GERAM) provides a comprehensive description of the elements recommended in enterprise engineering and integration and thereby sets the standard for the collection of tools and methods from which any enterprise should benefit to more successfully tackle initial integration design and the change processes which may occur during the enterprise operational lifetime.

Global Enterprise see Extended Enterprise.

Hierarchical clustering represents the combination of cases and clusters that are similar to each other, one pair at a time.

ID3 is a decision tree building algorithm which determines the classification of objects by testing the values of their properties.

Information is a summary and communication of the main components and relationships contained within the data and presented within a specific context [Pyle, 2003].

Information modeling is a modeling technique used for the description of a real-world entity.

Intelligent On Line Analytical Processing (IOLAP) represents the identification and visualization of the relevant information into a multidimensional representation.

Itemset concept is defined for Apriori algorithm as a non-empty set of items.

Interface definition language (IDL) is developed within Corba infrastructure and it is used for the definition of a static language and platform-independent interfaces.

Internet Inter-ORB Protocol (IIOP) is a binary protocol for communication between ORBs.

JavaBeans is an architecture and platform-independent set of classes for creating, using software components implemented in Java and developing component-based systems.

Java Database Connectivity (JDBC) is an Application Programming Interface (API) that provides access virtually to any data source from the Java programming language. It also provides cross-DBMS connectivity to a wide range of SQL databases and other data sources such as spreadsheets or flat files.

Java Naming and Directory Interface (JNDI) implements (in Java) the mechanism for components to locate each other within a distributed architecture.

Just-in-time (JIT) is a manufacturing paradigm which is directed to reduce procurement delays and procurement stocks as much as possible [Vemadat, 1996].

K-Means clustering is a clustering method based on the assumption that the data fall into a known number \(K\) of clusters.
FRAMEWORK FOR DISTRIBUTED KNOWLEDGE DISCOVERY SYSTEMS EMBEDDED IN EXTENDED ENTERPRISE

**K-nearest neighbour** see Nearest neighbour.

**Knowledge** is an interrelated collection of procedures for acting toward particular results in the world with associated references for when each is applicable along with its range of effectiveness [Pyle, 2003].

**Knowledge Discovery (KD)** deals with data integration, cleaning, refinement, profiling, mining and visualization.

**Lean Manufacturing** is directed to minimize the product development costs by eliminating non-productive activities or outsourcing too costly ones, and making organizational and cultural changes in the enterprise enforcing higher employee involvement and optimal using of proprietary knowledge [Vemadat, 1996].

**Legacy system** or application is a programme written in a programming language which has become obsoleted such as Cobol, and is not well documented. It may run on a dedicated hardware, and its maintenance is almost impossible or very expensive.

**Litemset** is defined for Apriori algorithm as an itemset with minimum support.

**Logistic Regression** is the development of standard linear regression for the situation where the dependent variable takes one or two (maybe three or four) values while in standard linear regression the dependent variable must be a continuous one taking a range of values.

**Manufacturing Model** describes available manufacturing processes, resources and strategies, and it also captures manufacturing facilities and their capabilities [Dorador, Young, 2000], [Harding, Popplewell, 1996].

**Metadata** is data which describes data or information about data, and generally, it comprises a description of information structures and models.

**Metamodel** is a description of a real-world entity using other models, for a particular purpose.

**Middleware** is a development and run-time infrastructure for distributed applications, defining and providing all or most of a set of common services including object or service location and binding, network protocol (which may be synchronous or asynchronous defined as loosely coupled) scalable server typically with transactional capability, directory, security and other services. Current middleware platforms include CORBA, Enterprise JavaBeans (EJB) and .Net http://www.omg.org/gettingstarted/terms_and_acronyms.htm.

**Model** is an abstraction that includes the essentials aspect of a complex and real situation or structure in order to process or use it.

**Model-Driven Architecture (MDA)** is a high-level abstract architecture based on UML methodology and existing profiles, and it is supported by Meta-Object Facility(MOF),
Common Object Request Broker Architecture (CORBA), XMI/XML and Common Warehouse Metamodel (CWM) [OMG, 2001a].

Meta Object Facility (MOF) is an extensible model driven integration framework for defining, manipulating and integrating metadata and data in a platform independent manner. http://www.omg.org/technology/documents/modeling_spec_catalog.htm#MOF.

Multi-tier architecture is a software solution composed by layers which offer clearly defined and separated interfaces and coordination between components.

Nearest neighbour method classifies each record in a data set based on a combination of the classes of the k record(s) most similar to it in a historical data set. Sometimes called the k-nearest neighbour technique.

Neural Networks (NN) is a computer model based on the architecture of the human brain which consists of multiple simple processing units connected by adaptive weights.

Object is a concept, an abstraction or a thing with sharp boundaries and meaning for an application [Quatrani, 2000].

Object Request Broker (ORB) provides clients and servers of distributed objects with an environment to make and receive request of each other.

Object wrapper is a black-box transformation of legacy code providing a working interface to an existing program or program component.

Object Linking and Embedding (OLE) is a method used in Microsoft Windows to connect documents, data and images of one type to another.

On Line Analytical Processing (OLAP) is defined as a way of performing multi-dimensional analysis on relational databases.

Open DataBase Computing (ODBC) is an Application Programming Interface (API) that provides a Windows interface for accessing a database.

Pattern(s) in the context of knowledge discovery and data mining are defined as similar structures in a file or a database that are relevant and repetitive.

Platform-Independent Model (PIM) is a model that abstracts away technological and engineering details [Poole et al., 2003].

Platform-Specific Models (PSM) is a model that contains technological and engineering details [Poole et al, 2003].

Predictive model is a mining model used to forecast explicit values, based on patterns determined from known results.
Predictive Modeling Markup Language (PMML) is an open standard format based on Extensible Markup Language (XML) specification for exchanging data mining models between applications running on different platforms.

Product Data Management (PDM) is a comprehensive area dealing with all product data and information related to product design, manufacture and delivery by using dedicated software systems.

Product Model is a comprehensive representation of a product and it contains detailed data description, information, objects and knowledge with their associated relations and interactions in order to support extended enterprise systems.

Purdue Enterprise Reference Architecture (PERA) is a reference architecture which has a layering structure and has been created to cover the life cycle of an enterprise from inception and objectives definition to the operational level [Vemadat, 1996].

Rational Rose is a software tools for visual modeling providing support for component-based and controlled iterative development [Rational Co., 2001].

Reference Architecture is a standardized and comprehensive system model including the relations and interactions between elements, which should be totally or at least partially adopted in order to integrate a system into an existing enterprise and further architectures are compared or derived.

Reference Model of Open Distributed Processing (RM-ODP) is a general coordinating framework for open distributed processing (ODP) defined by ISO and it facilitates the creation of standard architectures which support distribution, inter-networking, interoperability and portability.

Regression defined as linear and non-linear is a statistical method widely used for correlating data.

Reverse Engineering is a capability of Rational Rose Modeler tool for creating models from existing components and applications [Quatrani, 2000].

Remote Method Invocation (RMI) enables the programmer to create distributed Java applications in which the methods of remote Java objects can be invoked from other Java virtual machines, possibly on different hosts.

Robust Regression (K2R) is defined as a regression algorithm that allows models to be built to predict categories or continuous variables.

Round-trip Engineering is a capability of Rational Rose Modeler tool for keeping the designs synchronized with the code [Quatrani, 2000].
Rule-based model is a data mining model generated with an Association Rule algorithm.

Scalability is the ability of an application or product (hardware or software) to continue to function well when it (or its context) is changed in size, volume and complexity in order to meet user needs.

Segmentation see Classification.

Sequential Patterns are part of sequential analysis and they have the main goal to find all the sequential patterns with a pre-defined minimum support represented by a data sequence [Sousa et al., 1998].

Similarity in the context of data mining is expressed by distance functions, and it is intended to maximize similarity between cases assigned to the same group and to maximize difference between groups.

Supervised Learning In Quest (SLIQ) is a decision tree classifier designed to classify large training data.

Supply Chain Management (SCM) deals with the design, maintenance and operation of supply chain processes necessary to accomplish end-user needs [Ayers, 2001].

Support of a rule is used by the association rules algorithm and it indicates the number of records and transactions which confirm the rule [Cristofor, 2002].

Support Vector Machines (SVM) are learning systems that use a hypothesis space of linear functions in a high dimensional feature space, trained with a learning algorithm from optimization theory that implements a learning bias derived from statistical learning theory [Cristianini, Taylor, 2000].

Standard for the Exchange of Product Model Data (STEP) defines a neutral data format for the representation and exchange of product data (ISO10303).

Tree-based segmentation is decision trees algorithms which use a divide-and-conquer approach to classify data based on a tree representation.

Total Quality Management (TQM) means that product quality is a general business activity from product design to manufacturing, delivery and customer service.

Unified Modelling Language (UML) is a language used to specify and document the artifacts of an object-oriented system under development [Quatrani, 2000].

Unified object-oriented framework (especially in the context of this research) is defined as a development environment which is intended to leverage multiple enterprise software systems emphasizing knowledge discovery and data mining and their applications.

Unsupervised classification see clustering.
**View** or viewpoint is a perspective from which an enterprise is considered in order to reduce complexity by hiding the irrelevant aspects.

**Virtual Enterprise** is the temporary link between enterprise systems based on Internet and intranet technologies.

**XML Metadata Interchange (XMI)** is a standard for the exchange of metadata and metamodels such as UML models.

**Web Mining** is the process of mining information from the World Wide Web and it is composed of Web content, Web structure and Web usage Mining.
APPENDIX

The Appendix presents Java source code of the programmes used to demonstrate the research.

Package AssociationRule.java

ArMiner is a programme, part of a collection of tools for data mining called ArTool developed in the Department of Computer Science from University of Massachusetts Boston, USA. ArMiner is an application for finding Association Rules according to the algorithms presented in the chapter 3, dedicated to literature review.

```java
package com;
/**
 * Java Class com_example81_MyCORBAInterface
 * Source File com_example81_MyCORBAInterface.java
 * IDL Absolute Name com:MyCORBAInterface
 * Repository Identifier IDVcomIMyCORBAInterface:10
 */

IDL definition.

```
An association rule has two parts: the antecedent of the rule and the consequent of the rule, both of which are sets of items. Associated with these are a support and a confidence. The support shows how many rows of a database support this rule, the confidence shows what percentage of the rows that contain the antecedent also contain the consequent.

```java
public class AssociationRule implements java.io.Serializable {
    public static final int ANTECEDENT_SIZE = 1;
    public static final int CONSEQUENT_SIZE = 2;
    public static final int SUPPORT = 3;
    public static final int CONFIDENCE = 4;

    /**
     * The antecedent.
     */
    @serial
    private int[] antecedent,

    /**
     * The consequent
     */
    @serial
    private int[] consequent,

    /**
     * The support of the association rule
     */
    @serial
    private float support;

    /**
     * The confidence of the association rule.
     */
    @serial
    private float confidence,

    /**
     * Creates a new association rule.
     *
     * @param antecedent the antecedent of the association rule
     * @param consequent the consequent of the association rule
     * @param support the support of the association rule
     * @param confidence the confidence of the association rule
     * @exception IllegalArgumentException <code>antecedent</code>,
     *      <code>consequent</code>, <code>support</code>
     *      or <code>confidence</code> are null or not between 0 and 1
     */
    public AssociationRule(int[] antecedent, int[] consequent, float support, float confidence)
    {...
```
public AssociationRule(Itemset antecedent, Itemset consequent,
        float support, float confidence)
{
    if (antecedent == null || consequent == null
        || support < 0 || support > 1
        || confidence < 0 || confidence > 1)
        throw new IllegalArgumentException("constructor requires itemsets as arguments");

    this.antecedent = new int[antecedent.size()];
    for (int i = 0; i < antecedent.size(), i++)
        this.antecedent[i] = antecedent.getItem(i);

    this.consequent = new int[consequent.size()];
    for (int i = 0; i < consequent.size(), i++)
        this.consequent[i] = consequent.getItem(i),

    this.support = support,
    this.confidence = confidence,
}

/**
 * Return size of antecedent
 * @return size of antecedent
 */
public int antecedentSize()
{
    return antecedent.length;
}

/**
 * Return size of consequent
 * @return size of consequent
 */
public int consequentSize()
{
    return consequent.length;
}

/**
 * Return support of association rule
 */
public float getSupport()
{
    return support;
}

/**
 * Return confidence of association rule.
 */
public float getConfidence() {
    return confidence;
}

/**
 * Return i-th item in antecedent.
 *
 * @param i the index of the item to get
 * @exception IndexOutOfBoundsException <code>i</code> is an invalid index
 * @return the <code>i</code>-th item in antecedent
 */
public int getAntecedentItem(int i) {
    if (i < 0 || i >= antecedent.length)
        throw new IndexOutOfBoundsException("invalid index");
    return antecedent[i];
}

/**
 * Return i-th item in consequent.
 *
 * @param i the index of the item to get
 * @exception IndexOutOfBoundsException <code>i</code> is an invalid index
 * @return the <code>i</code>-th item in consequent
 */
public int getConsequentItem(int i) {
    if (i < 0 || i >= consequent.length)
        throw new IndexOutOfBoundsException("invalid index");
    return consequent[i];
}

/**
 * Compare two AssociationRule objects on one of several criteria
 *
 * @param ar the AssociationRule object with which we want to
 * compare this object
 * @param criteria the criteria on which we want to compare, can
 * be one of ANTECEDENT_SIZE, CONSEQUENT_SIZE, SUPPORT or CONFIDENCE.
 * @exception IllegalArgumentException <code>ar</code> is null
 * or criteria is invalid
 * @return a negative value if this object is smaller than
 * <code>ar</code>, 0 if they are equal, and a positive value if this
 * object is greater.
 */
public int compareTo(AssociationRule ar, int criteria) {
    if (ar == null)
        throw new IllegalArgumentException("method requires association rule as argument");
}
float diff;

if (criteria == ANTECEDENT_SIZE)
    return this antecedent length - ar antecedent.length;
else if (criteria == CONSEQUENT_SIZE)
    return this consequent length - ar consequent.length;
else if (criteria == SUPPORT)
    diff = this support - ar support;
else if (criteria == CONFIDENCE)
    diff = this.confidence - ar confidence;
else
    throw new IllegalArgumentException("invalid criteria");

if (diff < 0)
    return -1;
else if (diff > 0)
    return 1;
else
    return 0;

/**
 * Compare two AssociationRule objects on one of several criteria
 * @param ar the AssociationRule object with which we want to
 * compare this object
 * @param criteria the criteria on which we want to compare, can
 * be one of ANTECEDENT_SIZE, CONSEQUENT_SIZE, SUPPORT or CONFIDENCE.
 * @return true if the objects are equal in terms of antecedent
 * and consequent items; false otherwise.
 */
public boolean equals(Object obj)
{
    if (!(obj instanceof AssociationRule) || obj == null)
        return false;

    AssociationRule other = (AssociationRule)obj;

    if (antecedent length != other antecedent.length)
        return false;

    if (consequent length != other.consequent length)
        return false;

    for (int i = 0; i < antecedent.length; i++)
        if (antecedent[i] != other antecedent[i])
            return false;

    for (int i = 0; i < consequent.length; i++)
        if (consequent[i] != other consequent[i])
            return false;

    return true;
/**
 * Return a String representation of the AssociationRule.
 *
 * @return String representation of AssociationRule
 */
public String toString()
{
    String s = "{";
    for (int i = 0; i < antecedent.length; i++)
        s += antecedent[i] + ",";
    s += "}" + (" + support + "," + confidence + ");
    return s;
}

/**
 * for testing purposes only !!!
 */
public static void main(String[] args)
{
    Itemset is1 = new Itemset();
    Itemset is2 = new Itemset();
    is1.addItem(7);
    is1 addItem(3);
    is1 addItem(15);
    System.out.println("is1: " + is1);
    is2.addItem(12),
    is2 addItem(5);
    is2 addItem(8);
    System.out.println("is2: " + is2);
    AssociationRule ar = new AssociationRule(is1, is2,
        (float)0.5055,
        (float)0.3033);
    System.out.println("ar: " + ar);
}

Packages APRIORI and ITEMSET

The package for AssociationRules including classes Apriori and ItemSet is presented below. This package is part of WEKA [Witten, Frank, 2000]:

175
package weka.associations;

import java.io.*;
import java.util.*;
import weka.core.*;

/**
 * Class implementing an Apriori-type algorithm. Iteratively reduces the minimum support until it finds the required number of rules with the given minimum confidence. <p>
 * Valid options are:<p>
 * -N required number of rules <br>
 * The required number of rules (default: 10). <p>
 * -C minimum confidence of a rule <br>
 * The minimum confidence of a rule (default: 0.9). <p>
 * -D delta for minimum support <br>
 * The delta by which the minimum support is decreased in each iteration (default: 0.05). <p>
 * -M lower bound for minimum support <br>
 * The lower bound for the minimum support (default = 0.1). <p>
 * -S significance level <br>
 * If used, rules are tested for significance at the given level. Slower (default = no significance testing). <p>
 * -I <br>
 * If set the itemsets found are also output (default = no). <p>
 *

public class Apriori implements OptionHandler {

    /** The minimum support. */
private double m_minSupport;

/** The lower bound for the minimum support. */
private double m_lowerBoundMinSupport;

/** The minimum confidence. */
private double m_minConfidence;

/** The maximum number of rules that are output. */
private int m_numRules;

/** Delta by which m_minSupport is decreased in each iteration. */
private double m_delta;

/** Significance level for optional significance test. */
private double m_significanceLevel;

/** Number of cycles used before required number of rules was one. */
private int m_cycles;

/** The set of all sets of itemsets L. */
private FastVector m_Ls;

/** The same information stored in hash tables. */
private FastVector m_hashtables;

/** The list of all generated rules. */
private FastVector[] m_allTheRules;

/** The instances (transactions) to be used for generating the association rules. */
private Instances m_instances;

/** Output itemsets found? */
private boolean m_outputItemSets;

/**
 * Constructor that allows to sets default values for the
 * minimum confidence and the maximum number of rules
 * the minimum confidence.
 */
public Apriori() {
    resetOptions();
}

/**
 * Resets the options to the default values.
 */
public void resetOptions() {

m_delta = 0.05;
m_minConfidence = 0.90,
m_numRules = 10;
m_lowerBoundMinSupport = 0.1;
m_significanceLevel = -1;
m_outputItemSets = false;
}

/**
 * Method that generates all large itemsets with a minimum support, and from
 * these all association rules with a minimum confidence.
 *
 * @param instances the instances to be used for generating the associations
 *
 * @exception Exception if rules can't be built successfully
 */
public void buildAssociation(Instances instances) throws Exception {

double[] confidences, supports;int[] indices;
FastVector[] sortedRuleSet;
int necSupport = 0;
if (instances.checkForStringAttributes()) {
    throw new Exception("Can't handle string attributes!");
}

// Decrease minimum support until desired number of rules found.
m_cycles = 0;
m_minSupport = 1.0;
do {

    // Reserve space for variables
    m_Ls = new FastVector();
m_hashtables = new FastVector();
m_allTheRules = new FastVector[3];
m_allTheRules[0] = new FastVector();
m_allTheRules[1] = new FastVector();
m_allTheRules[2] = new FastVector();
    sortedRuleSet = new FastVector[3];
    sortedRuleSet[0] = new FastVector();
    sortedRuleSet[1] = new FastVector();

    // Find large itemsets and rules
    findLargeItemSets(instances);
    if (m_significanceLevel != -1)
        findRulesBruteForce();
    else
        findRulesQuickly();

    m_cycles++;
    m_minSupport = m_minSupport * 0.95;
}

}
// Sort rules according to their support
supports = new double[m_allTheRules[2].size()];
for (int i = 0; i < m_allTheRules[2].size(); i++)
    supports[i] = (double)(ItemSet)m_allTheRules[1].elementAt(i)).support();
indices = Util.stableSort(supports);
for (int i = 0, i < m_allTheRules[2].size(), i++)
    {
        sortedRuleSet[0].addElement(m_allTheRules[0].elementAt(indices[i]));
        sortedRuleSet[1].addElement(m_allTheRules[1].elementAt(indices[i]));
        sortedRuleSet[2].addElement(m_allTheRules[2].elementAt(indices[i]));
    }

// Sort rules according to their confidence
m_allTheRules[0].removeAllElements();
m_allTheRules[1].removeAllElements();
m_allTheRules[2].removeAllElements();
confidences = new double[sortedRuleSet[2].size()];
for (int i = 0; i < sortedRuleSet[2].size(); i++)
    confidences[i] = (double)sortedRuleSet[2].elementAt(i).doubleValue();
indices = Util stabilizationSort(confidences);
for (int i = sortedRuleSet[0].size() - 1;
    (i >= (sortedRuleSet[0].size() - m_numRules)) && (i >= 0); i--)
    {
        m_allTheRules[0].addElement(sortedRuleSet[0].elementAt(indices[i]));
        m_allTheRules[1].addElement(sortedRuleSet[1].elementAt(indices[i]));
        m_allTheRules[2].addElement(sortedRuleSet[2].elementAt(indices[i]));
    }
m_minSupport -= m_delta;
 necSupport = (int)(m_minSupport *
    (double)instances numInstances() + 0.5);
m_cycles++;
} while ((m_allTheRules[0].size() < m_numRules) &&
    (Util grOrEq(m_minSupport, m_lowerBoundMinSupport)) &&
    (necSupport >= 1));
m_minSupport += m_delta;

/**
* Returns an enumeration describing the available options
* *
* @return an enumeration of all the available options
*/
public Enumeration listOptions() {

String string1 = "\tThe required number of rules. (default = " + m_numRules + ")",
string2 = "\tThe minimum confidence of a rule. (default = " + m_minConfidence + ")",
string3 = "\tThe delta by which the minimum support is decreased in\n",
string4 = "\teach iteration. (default = " + m_delta + ")",
string5 =

179
The lower bound for the minimum support. (default = " + m_lowerBoundMinSupport + ")",
string6 = "If used, rules are tested for significance at the given level. Slower. (default = no significance testing)",
string7 = "If set the itemsets found are also output. (default = no)"

FastVector newVector = new FastVector(6);
newVector.addElement(new Option(string1, "N", 1, 
"-N <required number of rules output>"));
newVector.addElement(new Option(string2, "C", 1, 
"-C <minimum confidence of a rule>"));
newVector.addElement(new Option(string3 + string4, "D", 1, 
"-D <delta for minimum support>"));
newVector.addElement(new Option(string5, "M", 1, 
"-M <lower bound for minimum support>"));
newVector.addElement(new Option(string6 + string7, "S", 1, 
"-S <significance level>"));
newVector.addElement(new Option(string8, "S", 0, 
"-I"));

return newVector.elements();

/**
 * Parses a given list of options. Valid options are:
 * -N required number of rules <br>
 * The required number of rules (default: 10). <p>
 * -C minimum confidence of a rule <br>
 * The minimum confidence of a rule (default: 0.9). <p>
 * -D delta for minimum support <br>
 * The delta by which the minimum support is decreased in each iteration (default: 0.05).
 * -M lower bound for minimum support <br>
 * The lower bound for the minimum support (default = 0.1). <p>
 * -S significance level <br>
 * If used, rules are tested for significance at the given level. Slower (default = no significance testing). <p>
 * -I <br>
 * If set the itemsets found are also output (default = no). <p>
 * @param options the list of options as an array of strings
 * @exception Exception if an option is not supported
public void setOptions(String[] options) throws Exception {

resetOptions();
String numRulesString = Utils.getOption('N', options),
minConfidenceString = Utils.getOption('C', options),
deltaString = Utils.getOption('D', options),
minSupportString = Utils.getOption('M', options),
significanceLevelString = Utils.getOption('S', options);

if (numRulesString.length() != 0) {
    m_numRules = Integer.parseInt(numRulesString);
}
if (minConfidenceString.length() != 0) {
    m_minConfidence = (new Double(minConfidenceString)).doubleValue();
}
if (deltaString.length() != 0) {
    m_delta = (new Double(deltaString)).doubleValue();
}
if (minSupportString.length() != 0) {
    m_lowerBoundMinSupport = (new Double(minSupportString)).doubleValue();
}
if (significanceLevelString.length() != 0) {
    m_significanceLevel = (new Double(significanceLevelString)).doubleValue();
}

m_outputItemSets = (new Boolean(m_outputItemSets)).booleanValue();

/*
 * Gets the current settings of the Apriori object.
 * @return an array of strings suitable for passing to setOptions
 */
public String[] getOptions() {

String[] options = new String[11];
int current = 0;

if (m_outputItemSets) {
    options[current++] = "-I";
}
options[current++] = "-N"; options[current++] = "" + m_numRules;
options[current++] = "-C"; options[current++] = "" + m_minConfidence;
options[current++] = "-D"; options[current++] = "" + m_delta;
options[current++] = "-M"; options[current++] = "" + m_minSupport;
options[current++] = "-S"; options[current++] = "" + m_significanceLevel;

while (current < options.length) {
    options[current++] = "";
}

return options;
}
/**
 * Outputs the size of all the generated sets of itemsets and the rules.
 */
public String toString() {

StringBuffer text = new StringBuffer();

if (m_Ls.size() <= 1)
    return "\nNo large itemsets and rules found!\n";

   text.append("\n    \n    Apriori\n    \n    Minimum support: \n    Minimum confidence: \n    Significance level: \n    Number of cycles performed: \n    \n    Generated sets of large Itemsets:\n    \n    \n    Large Itemsets L(\n    \n    \n    \n    Best rules found:\n    \n    \n    return text.toString();
}

/**
 * Get the value of minSupport.
 */
*  
*  @return Value of minSupport.
*/
public double getMinSupport() {
return m_minSupport;
}

/**
 * Set the value of minSupport.
 * @param v Value to assign to minSupport.
 */
public void setMinSupport(double v) {
    m_minSupport = v;
}

/**
 * Get the value of lowerBoundMinSupport.
 * @return Value of lowerBoundMinSupport.
 */
public double getLowerBoundMinSupport() {
    return m_lowerBoundMinSupport;
}

/**
 * Set the value of lowerBoundMinSupport.
 * @param v Value to assign to lowerBoundMinSupport.
 */
public void setLowerBoundMinSupport(double v) {
    m_lowerBoundMinSupport = v;
}

/**
 * Get the value of minConfidence.
 * @return Value of minConfidence.
 */
public double getMinConfidence() {
    return m_minConfidence;
}

/**
 * Set the value of minConfidence.
 * @param v Value to assign to minConfidence.
 */
public void setMinConfidence(double v) {
    m_minConfidence = v;
}

/**
 * Get the value of numRules.
 * @return Value of numRules.
 */
public int getNumRules() {
    return m_numRules;
}

/**
 * Set the value of numRules.
 * @param v Value to assign to numRules.
 */
public void setNumRules(int v) {
    m_numRules = v;
}

/**
 * Get the value of delta.
 * @return Value of delta.
 */
public double getDelta() {
    return m_delta;
}

/**
 * Set the value of delta.
 * @param v Value to assign to delta
 */
public void setDelta(double v) {
    m_delta = v;
}

/**
 * Get the value of significanceLevel.
 * @return Value of significanceLevel.
 */
*/
public double getSignificanceLevel() {
    return m_significanceLevel;
}

/**
 * Set the value of significanceLevel.
 * @param v Value to assign to significanceLevel
 */
public void setSignificanceLevel(double v) {
    m_significanceLevel = v;
}

/**
 * Method that finds all large itemsets for the given set of instances.
 * @param the instances to be used
 * @exception Exception if an attribute is numeric
 */
private void findLargeItemSets(Instances instances) throws Exception {

    FastVector kMinusOneSets, kSets;
    Hashtable hashtable;
    int necSupport, i = 0;

    m_instances = instances;

    // Find large itemsets
    necSupport = (int)(m_minSupport * (double)instances.numInstances()+0.5);

    kSets = ItemSet.singletons(instances);
    ItemSet.updateCounters(kSets, instances);
    kSets = ItemSet.deleteItemSets(kSets, necSupport);
    if (kSets.size() == 0)
        return;
    do {
        m_Ls.addElement(kSets);
        kMinusOneSets = kSets;
        kSets = ItemSet.mergeAllItemSets(kMinusOneSets, i);
        hashtable = ItemSet.getHashtable(kMinusOneSets, kMinusOneSets.size());
        m_hashtables.addElement(hashtable);
        kSets = ItemSet.pruneItemSets(kSets, hashtable);
        ItemSet.updateCounters(kSets, instances);
        kSets = ItemSet.deleteItemSets(kSets, necSupport);
        i++;
    } while (kSets.size() > 0);
private void findRulesBruteForce() throws Exception {

    FastVector[] rules;
    // Build rules
    for (int j = 1; j < m_Ls.size(); j++) {
        FastVector currentItemSets = (FastVector)m_Ls.elementAt(j);
        Enumeration enumItemSets = currentItemSets.elements();
        while (enumItemSets.hasMoreElements()) {
            ItemSet currentItemSet = (ItemSet)enumItemSets.nextElement();
            rules = currentItemSet.generateRulesBruteForce(m_minConfidence, m_hashtables, j + 1,
                                                        m_instances, numInstances(),
                                                        m_significanceLevel);
            for (int k = 0; k < rules[0].size(); k++) {
                m_allTheRules[0].addElement(rules[0].elementAt(k));
                m_allTheRules[1].addElement(rules[1].elementAt(k));
                m_allTheRules[2].addElement(rules[2].elementAt(k));
            }
        }
    }
}

private void findRulesQuickly() throws Exception {

    FastVector[] rules;
    // Build rules
    for (int j = 1; j < m_Ls.size(); j++) {
        FastVector currentItemSets = (FastVector)m_Ls.elementAt(j);
        Enumeration enumItemSets = currentItemSets.elements();
        while (enumItemSets.hasMoreElements()) {
            ItemSet currentItemSet = (ItemSet)enumItemSets.nextElement();
            rules = currentItemSet.generateRules(m_minConfidence, m_hashtables, j + 1),
            for (int k = 0, k < rules[0].size(); k++) {
                m_allTheRules[0].addElement(rules[0].elementAt(k));
                m_allTheRules[1].addElement(rules[1].elementAt(k));
                m_allTheRules[2].addElement(rules[2].elementAt(k));
            }
        }
    }
}
**Main method for testing this class.**

```java
public static void main(String[] options) {

    String trainFileString;
    StringBuffer text = new StringBuffer();
    Apriori apriori = new Apriori();
    Reader reader;

    try {
        text.append("n\nApriori options:\n\n");
        text.append("-t <training file>\n");
        text.append("The name of the training file.\n");
        Enumeration enum = apriori.listOptions();
        while (enum.hasMoreElements()) {
            Option option = (Option) enum.nextElement();
            text.append(option synopsis() + "\n");
            text.append(option description() + "\n");
        }
        trainFileString = UtilisgetOption('t', options);
        if (trainFileString.length() == 0)
            throw new Exception("No training file given!");
        apriori.setOptions(options);
        reader = new BufferedReader(new FileReader(trainFileString));
        apriori.buildAssociations(new Instances(reader));
        System.out.println(apriori);
    } catch (Exception e) {
        System.out.println("n" + e.getMessage() + text);
    }
}
```

---

**Package Statistics.java**

Statistics is part of Weka library.

```java
package weka.core;

/**
 * Some constants */
```
private static double logSqrtPi = Math.log(Math.sqrt(Math.PI));
private static double rezSqrtPl = 1/Math.sqrt(Math.PI),
private static double bigx = 20.0;

/**
 * Computes standard error for observed values of a binomial
 * random variable.
 * @param p the probability of success
 * @param n the size of the sample
 * @return the standard error
 */
public static double binomialStandardError(double p, int n) {
    if (n == 0) {
        return 0;
    }
    return Math.sqrt((p*(1-p))/(double) n),
}

/**
 * Returns chi-squared probability for given value and degrees
 * of freedom. (The probability that the chi-squared variate
 * will be greater than x for the given degrees of freedom.)
 * Adapted from umxstat by Gary Perlman
 * @param x the value
 * @param df the number of degrees of freedom
 */
public static double chiSquaredProbability(double x, int df) {
    double a, y = 0, s, e, c, z, val;
    boolean even;
    if (x <= 0 || df < 1)
        return (1);
    a = 0.5 * x;
    even = (((int)(2*(df/2))) == df);
    if (df > 1)
        y = Math.exp(-a), 0.0: Math.exp (-a));
    s = (even ? y : (2.0 * normalProbability(-Math.sqrt (x))));
    if (df > 2){
        x = 0.5 * (df - 1.0);
        z = (even ? 1.0 : 0.5);
        if (a > bigx){
            e = (even ? 0.0 : logSqrtPi),
            c = Math log (a);
            while (z <= x){
                e = Math log (z) + e;
                val = e*z-a-e;
                s += Math.exp (val); 0.0: Math.exp (val));
                z += 1.0;
            }
        }
    }
public static double FCriticalValue(double p, int df1, int df2) {
    double fval;
    double maxf = 999990;  /* maximum possible F ratio */
    double minf = .000001;  /* minimum possible F ratio */

    if (p <= 0.0 || p >= 1.0)
        return (0.0);

    fval = 1.0 / p;  /* the smaller the p, the larger the F */

    if (Math.abs(maxf - minf) > .000001) {
        if (FProbability(fval, df1, df2) < p) /* F too large */
            maxf = fval,
        else /* F too small */
            minf = fval,
        fval = (maxf + minf) * 0.5;
    }

    return (fval),
}

/**
 * @param F the F-ratio
 * @param df1 the first number of degrees of freedom
 * @param df2 the second number of degrees of freedom
 * @return the probability of the F-ratio
 */
public static double FProbability(double F, int df1, int df2) {

```java
int i, j,
int a, b;
double w, y, z, d, p,

if ((Math.abs(F) < 10e-10) || df1 <= 0 || df2 <= 0)
    return (1.0);

a = (df1%2 == 1) ? 1 : 2;
b = (df2%2 == 1) ? 1 : 2,
w = (F * df1) / df2;
z = 1.0 / (1.0 + w);

if (a == 1) {
    if (b == 1) {
        p = Math.sqrt (w);
y = 1/Math.PI; /* 1 / 3.14159 */
d = y * z / p;
p = 2.0 * y * Math.atan (p),
    } else {
        p = Math.sqrt (w * z);
d = 0.5 * p * z / w;
    }
} else if (b == 1) {
    p = Math.sqrt (z),
d = 0.5 * z * p;
p = 1.0 - p;
}
else {
    d = z * z;
p = w * z;
}

y = 2.0 * w / z;
for (j = b + 2; j <= df2; j += 2) {
    d *= (1.0 + a / (j - 2.0)) * z;
p = (a == 1 ? p + d * y / (j - 1.0) * (p + w) * z),
}
y = w * z;
z = 2.0 / z;
b = df2 - 2;
for (i = a + 2; i <= df1; i += 2) {
    j = i + b;
d *= y * j / (i - 2.0);
p += z * d / j;
}

// correction for approximation errors suggested in certification
if (p < 0.0)
p = 0.0;
else if (p > 1.0)
p = 1.0;
return (1.0 - p),
}

/**
* Returns probability that the standardized normal variate Z (mean = 0, standard
* deviation = 1) is less than z
*/
public static double normalProbability(double z) {

double y, x, w;

if (z == 0.0)
    x = 0.0,
else
    y = 0.5 * Math.abs(z);

if (y >= 3.0)
    x = 1.0;
else if (y < 1.0) {
    w = y * y,
    x = (((((0.0000124818987 * w
            -0.001075204047) * w +0.005198775019) * w
            -0.019198292004) * w +0.059054035642) * w
            -0.151968751364) * w +0.319152932694) * w
            -0.531923007300) * w +0.797884560593) * y * 2.0;
}
else {
    y = 2.0;
    x = ((((((0.000045255659 * y
            +0.000152529290) * y -0.00019538132) * y
            -0.000676904986) * y +0.001390604284) * y
            -0.000794620820) * y -0.002034254874) * y
            +0.011630447319) * y -0.009279453341) * y
            +0.005353579108) * y -0.002141268741) * y
            +0.000535310849) * y +0.999936657524,
    }
}

return (z > 0.0 ? (x + 1.0) / 2.0) . ((1.0 - x) / 2.0);}

/**
* Computes absolute size of half of a student-t confidence interval
* for given degrees of freedom, probability, and observed value
* @param df the number of degrees of freedom
* @param p the probability
* @param se the observed value
* @return absolute size of half of a student-t confidence interval
*/
public static double studentTConfidenceInterval(int df, double p,
        double se) {
    return Math.sqrt(FCriticalValue(p, 1, df))*se;
}

/**
* Main method for testing this class.
public static void main(String[] ops) {
    System.out.println("Statistical Analysis launched");
    System.out.println("Binomial standard error (0.5, 100): "+
        Statistics.binomialStandardError(0.5, 100));
    System.out.println("Chi-squared probability (2.558, 10): "+
        Statistics.chiSquaredProbability(2.558, 10));
    System.out.println("Normal probability (0 2): "+
        Statistics.normalProbability(0 2));
    System.out.println("F critical value (0.05, 4, 5): "+
        Statistics.FCriticalValue(0.05, 4, 5));
    System.out.println("F probability (5.1922, 4, 5): "+
        Statistics.FProbability(5.1922, 4, 5));
    System.out.println("Student-t confidence interval (9, 0 01, 2). "+
        Statistics.studentTConfidenceInterval(9, 0 01, 2));
}