



Available online at [www.sciencedirect.com](http://www.sciencedirect.com)



Engineering Applications of Artificial Intelligence 20 (2007) 1097–1111

Engineering Applications of  
**ARTIFICIAL  
INTELLIGENCE**

[www.elsevier.com/locate/engappai](http://www.elsevier.com/locate/engappai)

## Data mining for agent reasoning: A synergy for training intelligent agents

Andreas L. Symeonidis<sup>a,b,\*</sup>, Kyriakos C. Chatzidimitriou<sup>c</sup>,  
Ioannis N. Athanasiadis<sup>d</sup>, Pericles A. Mitkas<sup>a,b</sup>

<sup>a</sup>Electrical and Computer Engineering Department, Aristotle University of Thessaloniki, 54124, Thessaloniki, Greece

<sup>b</sup>Intelligent Systems and Software Engineering Laboratory, Informatics and Telematics Institute/CERTH, 57001 Thessaloniki, Greece

<sup>c</sup>Department of Computer Science, Colorado State University, Fort Collins, CO 80523, USA

<sup>d</sup>Istituto Dalle Molle di Studi sull'Intelligenza Artificiale, CH-6928 Manno, Switzerland

Received 27 March 2006; received in revised form 16 February 2007; accepted 21 February 2007

Available online 18 April 2007

### Abstract

The task-oriented nature of data mining (DM) has already been dealt successfully with the employment of intelligent agent systems that distribute tasks, collaborate and synchronize in order to reach their ultimate goal, the extraction of knowledge. A number of sophisticated multi-agent systems (MAS) that perform DM have been developed, proving that agent technology can indeed be used in order to solve DM problems. Looking into the opposite direction though, knowledge extracted through DM has not yet been exploited on MASs. The inductive nature of DM imposes logic limitations and hinders the application of the extracted knowledge on such kind of deductive systems. This problem can be overcome, however, when certain conditions are satisfied a priori. In this paper, we present an approach that takes the relevant limitations and considerations into account and provides a gateway on the way DM techniques can be employed in order to augment agent intelligence. This work demonstrates how the extracted knowledge can be used for the formulation initially, and the improvement, in the long run, of agent reasoning.

© 2007 Elsevier Ltd. All rights reserved.

**Keywords:** Data mining; Agent technology; Agent training; Agent reasoning; Knowledge model

### 1. Introduction

#### 1.1. Technology coupling

The application domain of data mining (DM) and its related techniques and technologies has been greatly expanded in the last few years. The development of automated data collection tools and the tremendous data explosion, the imperative need for the interpretation and exploitation of massive data volumes, along with the existence of supporting tools, has resulted to the develop-

ment and flourishing of sophisticated DM methodologies. Issues concerning data normalization, algorithm complexity and scalability, result validation and comprehension have been successfully dealt with (Adriaans and Zantige, 1996; Witten and Frank, 1999; Han and Kamber, 2001), while numerous approaches have been adopted for the realization of autonomous and versatile DM tools, which foster all the appropriate pre- and post-processing steps that constitute the process of knowledge discovery in databases (KDD) (Piatetsky-Shapiro and Frawley, 1992; Fayyad et al., 1996; Chen et al., 1996).

Since DM systems are comprised of a number of discrete, nevertheless dependent tasks, they can be thought of as networks of collaborative, yet autonomous, units that regulate, control and organize all distributed activities involved in data cleaning, data transformation and reduction, algorithm application and result evaluation.

\*Corresponding author. Intelligent Systems and Software Engineering Laboratory, Informatics and Telematics Institute/CERTH, 57001 Thessaloniki, Greece. Tel.: +30 2310 99 6399; fax: +30 2310 99 6398.

E-mail addresses: [asymeon@iti.gr](mailto:asymeon@iti.gr), [asymeon@ee.auth.gr](mailto:asymeon@ee.auth.gr) (A.L. Symeonidis).

Research literature on intelligent agent system architectures has proven that such kinds of problems that require the synergy of a number of distributed elements for their solution can be efficiently implemented as a multi-agent system (MAS) (Ferber 1999). This is why multi-agent technology has repeatedly been used as a powerful technology for developing DM systems (Stolfo et al., 1997; Kargupta et al., 1997; Zhang et al., 2003; Moham-madiam, 2004).

In an MAS realizing a DM system, all requirements collected by the user and all appropriate tasks are perceived as distinguished roles of separate agents, acting in close collaboration. All agents participating in an MAS communicate with each other by exchanging messages, encoded in a specific agent communication language (ACL). Each agent in the MAS is designated to manipulate the content of the incoming messages and take specific actions/decisions that conform to the particular reasoning mechanism specified by DM primitives.

Considerable effort has been spent to formulate efficient agent models for enhancing the DM process. Moving towards the opposite direction (Fig. 1), we envision the creation of knowledge models by the use of DM techniques and their dynamic exploitation by agents operating in diverse environments.

On the other hand, the interesting, non-trivial, implicit and potentially useful knowledge extracted by the use of DM (Fayyad et al., 1996) would be expected to find fast application on the development and realization of intelligence in agent technology (AT). The incorporation of knowledge based on previous observations may considerably improve agent infrastructures while also increasing reusability and minimizing customization costs. Unfortunately, limitations related to the nature of different logics adopted by DM and AT (inductive and deductive, respectively), hinder the unflustered application of knowledge to agent reasoning. If these limitations are overcome, then the coupling of DM and AT may become feasible.

### 1.2. Related bibliography

Going briefly through related work, attempts to couple DM and AT already exist. Galitsky and Pampapathi (2003) in their work use both inductive and deductive

reasoning, in order to model and process the claims of unsatisfied customers. Deduction is used for describing the behaviors of agents (humans or companies), for which we have complete information, while induction is used to predict the behavior of agents, whose actions are uncertain to us. A more theoretical approach on the way DM extracted knowledge can contribute to AT performance has been presented by Fernandes (2000), who attempts to model the notions of data, information and knowledge in purely logical terms, in order to integrate inductive and deductive reasoning into one inference engine. Kero et al. (1995), finally, propose a DM model that utilizes both inductive and deductive components. Within the context of their work, they model the discovery of knowledge as an iteration between high level, user-specified patterns and their elaboration to (deductive) database queries, whereas they define the notion of a meta-query that performs the (inductive) analysis of these queries and their transformation to modified, ready-to-use knowledge.

In general, existing agent-based solutions can be classified according to the granularity of the agent system and inference mechanism of the agents. Fig. 2 attempts a qualitative representation of the MAS space; agent reasoning may fall under four major categories, ranging from simple heuristics to self-organizing systems. The shaded region delineates the area of interest of the proposed synergy.

### 1.3. Presented work

Advancing on the way earlier research work has dealt with the coupling of technologies, we believe that intelligence should not be hard-coded in an agent, since this option reduces flexibility and puts a heavy burden on the programmer's shoulders. On the other hand, ill-equipped agents are seldom effective. Taking the middle ground, a rather simple (dummy) agent can be created and then trained to learn, adapt, get smarter and more efficient.

The embedded intelligence in an agent should be acquired from its experience of former transactions with humans and other agents that work on behalf of a human or an enterprise. Hence, an agent that is capable of *learning* can increase significantly its effectiveness as a personal collaborator and yield a reduction of workload for human users. The learning process is a non-trivial task that can be facilitated by extracting knowledge from the experience of other agents.

In this paper, we present a methodology and the corresponding tool for transferring DM-extracted knowledge into newly created agents. DM is used to generate knowledge models, which can be dynamically embedded into the agents. As new data accumulate, the process can be repeated and the decision structures can be updated, effectively **retraining** the agents. Consequently, the process is suitable for either upgrading an existing, non agent-based application by adding agents to it, or for improving the already operating agents of an agent-based application.

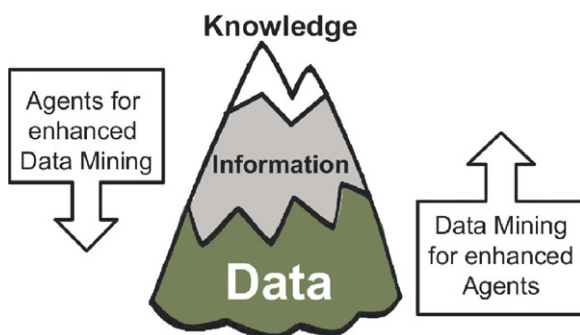


Fig. 1. Mining for intelligence.

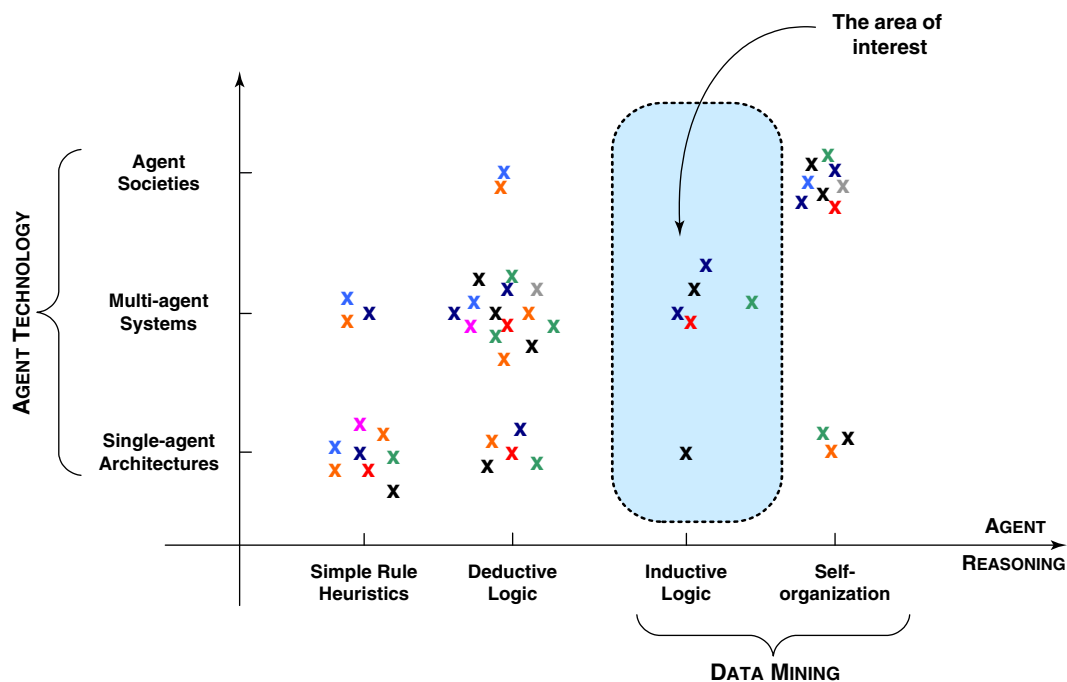


Fig. 2. Agent-based applications and inference mechanisms.

The methodology relies heavily on the inductive nature of DM, while taking into account its limitations.

#### 1.4. Paper outline

The rest of the paper is organized as follows: Section 2 establishes the difference between deductive and inductive logic, and presents a mechanism that permits the seamless marriage of the two logic paradigms, with each one contributing its strengths, leading to agent-based systems with a good knowledge of the application domain. Section 3 provides an overview of the Data Miner, an open-source platform developed for supporting and automating a large part of the mechanism. In order to demonstrate the feasibility of our approach, Section 4 provides a rough sketch of two prototypes developed following the presented mechanism. Finally, Section 5 states our conclusions and denotes some open issues still to be considered.

## 2. Coupling data mining with intelligent agents

### 2.1. Logic and limitations

The increasing demand for sophisticated and autonomous systems, along with the versatility and generic nature of the multi-agent technology paradigm has led to the employment of AT in a variety of disciplines. Requirements of such systems are, among others, the processing of vast amount of information, the cooperation and coordination of different processes and entities leading to unified solutions, even the development of intelligent recommendations that humans can incorporate into their decisions. Based on this motivation, researchers in the areas of

artificial intelligence (AI) and software engineering (SE) are oriented towards hybrid approaches that combine different theoretical backgrounds and algorithms. Basic prerequisite for the merging of technologies is the existence of a “common denominator” to reference on. In our case, this “common denominator” for AT and DM is the inference procedures they deploy, which can be expressed by Eq. (1):

$$Data \wedge Knowledge = Information. \quad (1)$$

AT and DM, though, assess Eq. (1) in two different ways: AT employs the deductive logic paradigm, whereas DM employs the inductive logic paradigm. In the first case, an agent system is designed with a model of its environment in mind (*Knowledge*) in order to act on its sensor inputs (*Data*) and to produce decisions (*Information*). In the second case, DM algorithms are trying to discover the *Knowledge* necessary in order to translate *Data* into *Information*.

Deduction is defined as the logical reasoning process in which conclusions must follow their premises. In other words, a knowledge model is applied to data in order to produce information, either in the form of data or knowledge (Fernandes, 2000). In general, deductive systems are truth preserving, that is, they transform their inputs while conserving their true value. This feature along with the assumption that initial knowledge is correct, allows the semantic correctness of the knowledge form to be assumed (Talavera and Cortes, 1997).

Usually, deduction is used when the process and the goals are well-defined and the human expert, who constructs the knowledge base, has a fine grasp of the problem’s underlying principles. Then the system can be modeled appropriately and conclusions can be drawn

based on the rules and procedures implemented, satisfying efficiency and soundness.

Nevertheless, deductive rationality has certain limitations, since it breaks down under complication. On one hand, the logical apparatus ceases to cope beyond certain complexity, while on the other, in interactive situations of complication, agents are forced to guess their behaviors, leading to a world of subjective beliefs where objective, well-defined assumptions cease to apply and therefore the problem becomes ill-defined (Arthur, 1994). In addition, deductive reasoning systems lack adaptivity, as they are disembodied from their environment (Wooldridge, 1999).

These drawbacks can be overcome, though, by the use of induction, defined as the inference from the specific to the general (Chen, 1999). Induction follows a different approach towards the realization of Eq. (1). Data and information are known, (data tuples and their classification, respectively) and the objective is to find the knowledge model that “transforms” the former into the latter. The main primitives of induction are the discovery of previously unknown rules, the identification of correlations and the validation of hypotheses in supplied datasets. On the other hand, the discovered knowledge may not always be valid, since inductive learning systems transform their inputs by means of generalizations (Kodratoff, 1988). Therefore, induction alone is unable to satisfy system soundness.

It is therefore obvious that neither of the above approaches is a panacea in real-life applications. Even though both inductive and deductive reasoning have been used separately in a variety of fields, we argue that the real coupling of these two practices (through the coupling of their representative technologies) may constitute the basis for enhanced and more efficient market-oriented MAS. In our approach, we exploit the ability of deductive reasoning for facilitating the well-known and established processes/functionalities of the MAS, while we utilize inductive reasoning for the discovery and exploitation of previously unknown knowledge, i.e. the adaptation of the MAS to real-world, real-time needs.

## 2.2. The mechanism

Fig. 3 illustrates the mechanism for developing MAS that can exploit knowledge extracted by the use of DM techniques. On the one hand, standard agent-oriented SE (AOSE) processes are followed, in order to set up the ontology, the behaviors and types of agents, as well as their interactions. On the other hand, DM techniques are applied for the extraction of the appropriate knowledge models.

The methodology pays special attention to two issues: (a) the ability to **dynamically** embed the extracted knowledge models into the agents and (b) the ability to repeat the above process as many times as deemed necessary.

The steps one has to follow in order to build a DM-enhanced MAS are (a) to develop the application ontology,

(b) design and develop agent behaviors, (c) develop agent types realizing the created behaviors, (d) *apply DM techniques on the provided data set*, (e) *extract knowledge models for each agent type*, (f) create the agent instances for the application, (g) dynamically incorporate the knowledge models to the corresponding agents, (h) instantiate the multi-agent application, (i) monitor agent actions, and, finally (j) *periodically retrain the agents of the system*.

In this way not only do we equip our MAS with the capabilities of this coalition, but we also automate (or semi-automate) a number of system procedures. Moreover, from an SE point of view, we increase the adaptability, reusability and versatility of MAS, just by reapplying DM on different data sets, and by incorporating the extracted knowledge into the agents of the MAS. This way we can easily propose MAS as an add-on solution for the enhancement and increase of value of legacy systems.

## 3. Data miner: a tool for training and retraining agents

In order to enable the incorporation of knowledge into agents, we have implemented a tool that is agent oriented. The data miner, as its name implies, is a DM suite that provides users with the capability of applying a number of DM algorithms on application specific and agent-behavior-specific data, and of incorporating the extracted decision models into JADE agents (Bellifemine et al., 2000), augmenting that way their intelligence. The data miner is one of the core components of the Agent Academy platform,<sup>1</sup> a platform that (semi)automates a number of MAS development tasks outlined in Section 2.2 (Mitkas et al., 2002, 2003). Data miner can also function as a standalone tool for classification, association rule extraction and clustering. A stable version of the data miner, along with user documentation can be found at: <http://sourceforge.net/projects/aadataminer>.

### 3.1. Prerequisites

During the design phase of an MAS, the application developer is called to take action on two lines:

1. Identify how DM techniques can be exploited, in order to enhance the system under development (technique to be selected, data to be cleaned, etc.)
2. Decide on which of the MAS agents will incorporate and diffuse DM-extracted knowledge. Not all agent types have to bear the inductive reference mechanism.

<sup>1</sup>Agent Academy (Agent Academy Consortium, 2004a) is a platform for developing MAS architectures and for enhancing their functionality and intelligence through the use of DM techniques. For more information on Agent Academy, please visit the official project site: <http://sourceforge.net/projects/agentacademy>.

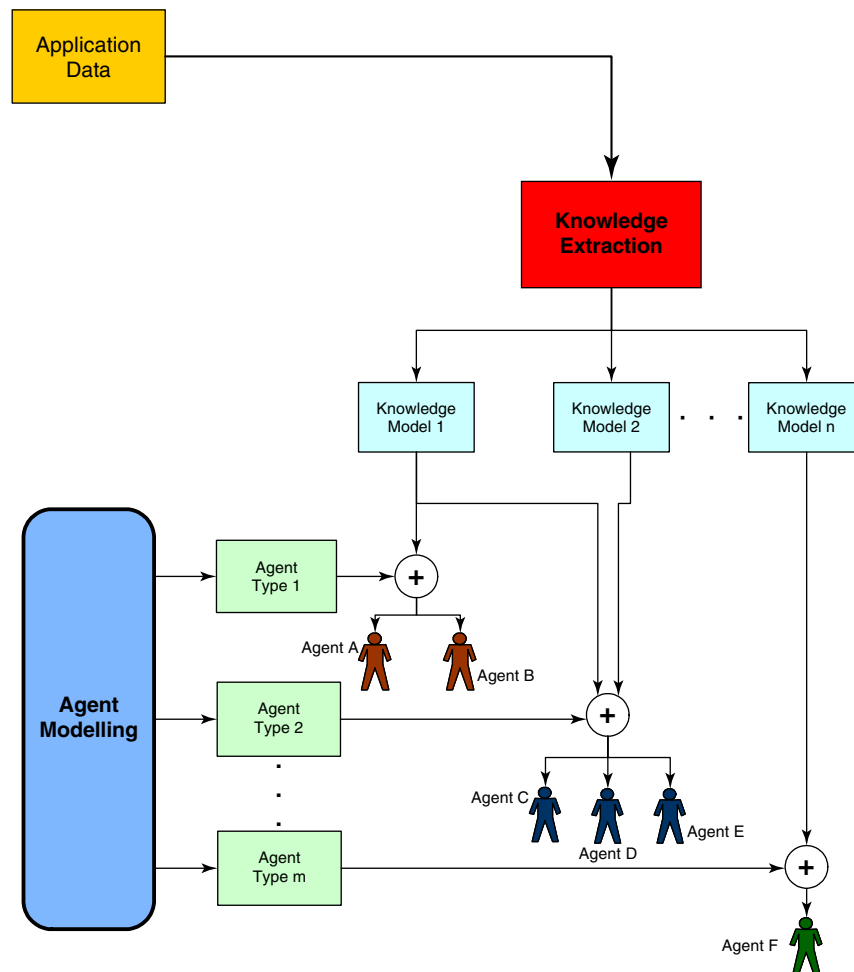


Fig. 3. The MAS development mechanism.

### 3.2. Data miner overview

The mechanism adopted for embedding rule-based reasoning capabilities into agents is illustrated in Fig. 4.

The data miner comprises three modules:

1. The preprocessing unit, which receives and conditions the input data.
2. The miner, which is the core component, where DM takes place.
3. The evaluator, which is the front-end of the data miner and provides algorithm performance metrics and visualization capabilities.

Data, either application-specific or agent-behavior-specific come in XML format. Each data file contains information on the name of the JADE agent the file belongs to and on the decision structure of the agent it will be applied on. The XML file is then inserted into the *Preprocessing Unit* of the data miner, where all the necessary data selection and data cleaning tasks take place. After that data are forwarded to the *Miner*, where the user decides on the DM technique, as well as on the specific

algorithm to employ. Table 1 illustrates the available DM techniques and the corresponding DM algorithms.

After DM is performed, the results are sent to the *Evaluator*, which is responsible for the validation and visualization of the extracted model. In the case that the user accepts the constructed model, a PMML<sup>2</sup> document describing the knowledge model is generated. This document expresses the referencing mechanism of the agent we intend to train. The resulting decision model is then translated to a set of facts executed by a rule engine. The implementation of the rule engine is realized through the Java Expert System Shell (JESS) (Friedman-Hill, 2003), which is a robust mechanism for executing rule-based agent reasoning. The execution of the rule engine transforms the

<sup>2</sup>PMML is an XML-based language, which provides a rapid and efficient way for companies to define predictive models and share models between compliant vendors applications. It allows users to develop models within one vendor's application, and use other vendors' applications to visualize, analyze, evaluate or otherwise use the models. The fact that PMML is a data mining standard defined by DMG (Data Mining Group, 2001) provides the data miner with versatility and compatibility to other major data mining software vendors, such as Oracle, SAS, SPSS and MineIt.

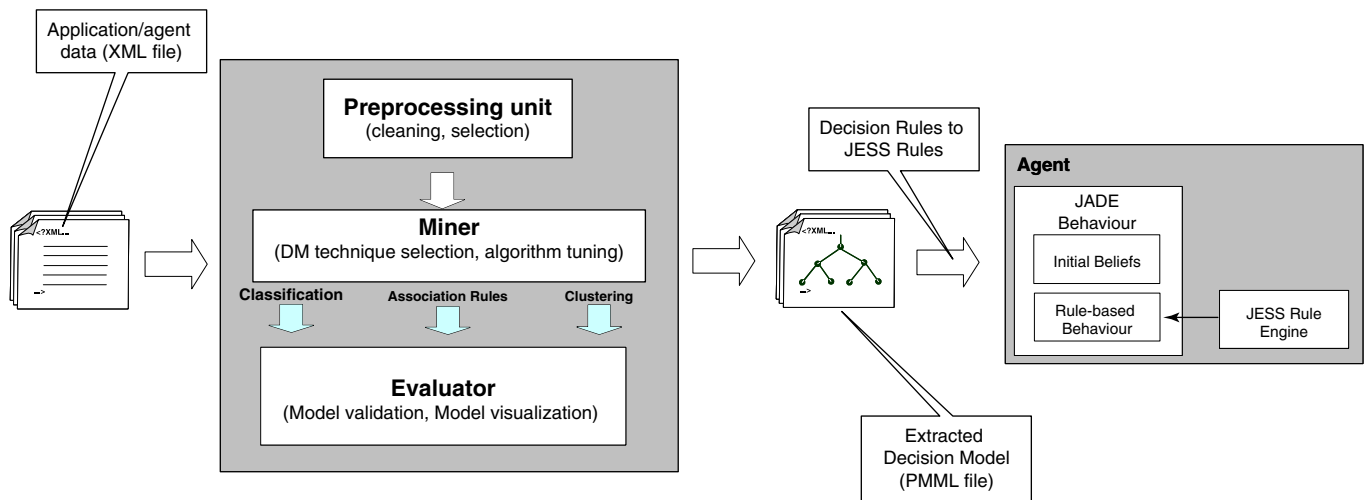


Fig. 4. The agent training mechanism.

Table 1  
Data miner provided techniques and algorithms

	DM technique		
	Classification	Association rules	Clustering
DM algorithm	ID 3 C 4.5 CLS $\sigma$ -FLNMap <sup>a</sup>	A priori DHP DIC $\kappa$ -Profile <sup>b</sup>	K-means PAM EM —

<sup>a</sup> $\sigma$ -FLNMap is a new algorithm that formalizes the solution using the notion of fuzzy lattices. Training data are clustered into fuzzy lattices, each one of which is assigned to a certain class, based on a criterion called *inclusion measure*. In this way, the extracted decision model consists of a set of fuzzy lattice rules (Kaburlasos et al., 2006).

<sup>b</sup> $\kappa$ -Profile is a new algorithm that attempts to determine agent dynamic traversal rules, by dealing with their behaviors as states. These rules can be utilized as a knowledge base for classifying agent actions, according to their domain roaming behavior. Based on web log mining,  $\kappa$ -Profile creates an intelligent segregation of roaming attitudes for predicting agent behaviors (Agent Academy Consortium, 2004b).

data miner extracted knowledge into a living part of the agent's behavior.

### 3.3. Training and retraining with the data miner

Following the design primitives imposed by the above-mentioned mechanism, the data miner has been developed as a GUI-based wizard that guides the user from the initial step of data loading, until the final step of creating the PMML document and embedding it into the reasoning mechanism of the agents.

At first, the user launches the data miner and specifies whether he/she will train a new agent or retrain an existing one (Fig. 5i). Then he/she has to define an agent ontology to load data for (Fig. 5ii) and to specify the *agent decision attributes* that will be represented as internal nodes of the extracted decision model. The corresponding XML data

file is selected through the selection panel (Fig. 5iii), it is parsed, and then validated for syntactical correctness and ontology consistency.

The user can decide on the attributes on which DM will be performed, in case the dataset provided represents a superset of the desired one (Fig. 6i). On the next step, the appropriate DM technique, with respect to the problem addressed, and the most suitable algorithm to use (Fig. 6ii) are selected. For each one of the algorithms selected different fine tuning options are provided, in relation to the DM mechanism employed (Fig. 6iii). Training, testing and validation options are specified (Fig. 6iv), i.e. whether training on the specified data set is going to be performed or testing against another data set, cross-validation or percentage splitting—always with respect to the DM technique used.

Advancing through the procedure flow, the user can decide whether he/she would like to save the output buffer of the data miner, which holds information on evaluation metrics for each technique (mean square root errors, prediction accuracy, etc.) or the output model, which describes the extracted-to-be knowledge. In addition, through this panel visualization capabilities can be determined (Fig. 7i). An overview of the data set options and the algorithm tuning, training and output defined parameters is provided in the last panel of the data miner (Fig. 7ii), in order for the user to check for any errors or omissions. Then the DM procedure is initiated. If the resulting decision model is judged as satisfactory by the user, then it is accepted and the corresponding PMML document is constructed.

This document is then parsed, translated into JESS blocks, and inserted into the agent's behavior, according to the mechanism described previously. Fig. 8i illustrates part of the initial XML document, while Fig. 8ii the corresponding PMML output, after the C4.5 algorithm (Quinlan, 1993) has been applied on the dataset provided. Fig. 9 summarizes the whole procedure.

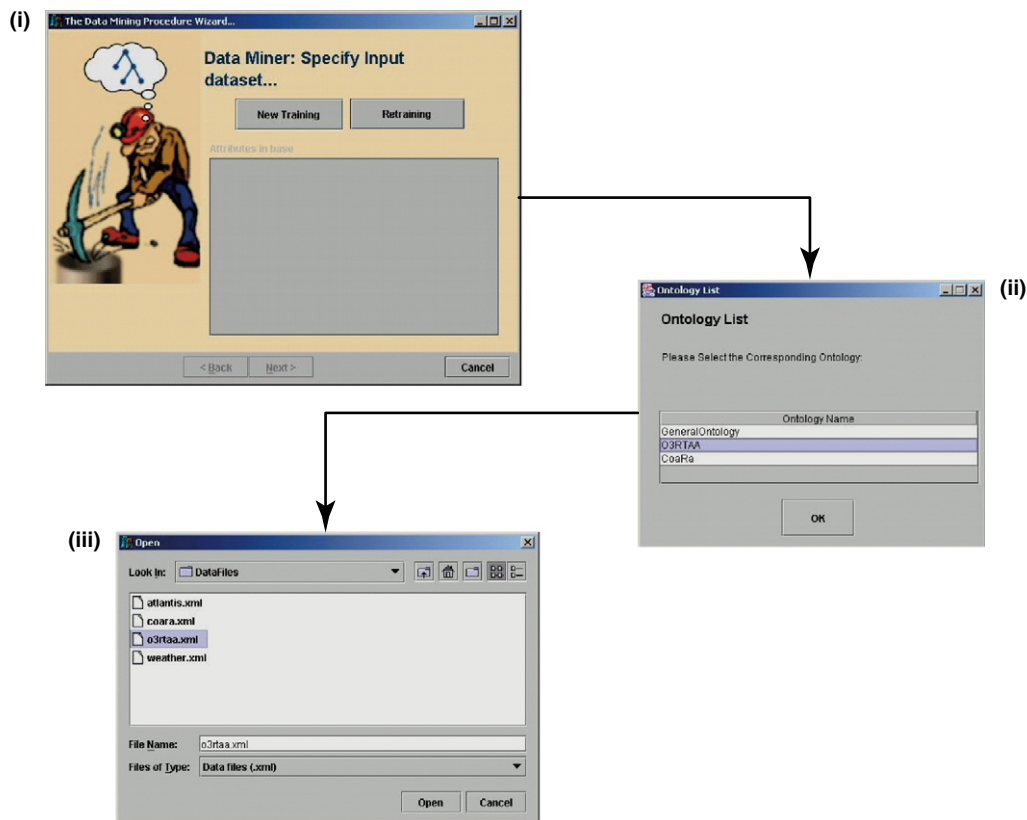


Fig. 5. Launching data miner and defining the initial data set.

#### 4. Data miner assessment: representative test case scenarios

In order to prove the validity of our concept, we have developed a number of test cases that employ our approach and provide sophisticated solutions by incorporating domain knowledge and exploiting dynamically formatted knowledge. Two of the most popular application domains were selected and an equal number of pilot multi-agent applications were developed, deployed and evaluated. These applications are:

- (a) An Enterprise Resource Planning (ERP) add-on, which provides intelligent and adaptive policy recommendations in the context of Customer and Supplier Relationship Management,
- (b) An Environmental Monitoring Information System (EMIS), which monitors a number of environmental measures and provides multiple-level notifications, alerts and alarms, based on certain attribute dependencies.

Even though these applications reside in different domains, they share a common denominator, as they are both multi-agent decision-support systems that can exploit DM techniques capabilities. An outline of these test-case scenarios is hereon depicted, along with their points of strength with respect to other approaches.

It should be denoted that, although it is not necessary for all systems' components to be implemented as agents, the adoption of the Agent Academy framework for embedding DM-extracted knowledge into agents, led to fully AOSE schemes. Both system architectures were designed keeping in mind that each agent type should be mapped to a specific software process or DM problem. The DM algorithms selected—according to their relevance to the problem—were provided by the data miner.

##### 4.1. Agents and data mining in enterprise information systems

Enterprise information systems (EIS) have been enjoying acceptance in the last two decades, therefore leading to the development of many IT systems (especially Enterprise Resource Planning systems—ERP) that can store and manipulate the vast amount of data daily produced by the companies. Though largely in use, legacy ERP systems are only transactional IT systems, while the need for analytical interpretation of data is increasing (Shapiro, 1999). In order to manipulate these large reservoirs in a more efficient way, reduce data overflow and provide useful recommendations, we have developed an Intelligent Policy Recommendation multi-Agent system (IPRA). IPRA is coupled with a legacy ERP and can efficiently manage one of the most important business assets: customer ordering. IPRA comprises agents representing the most important

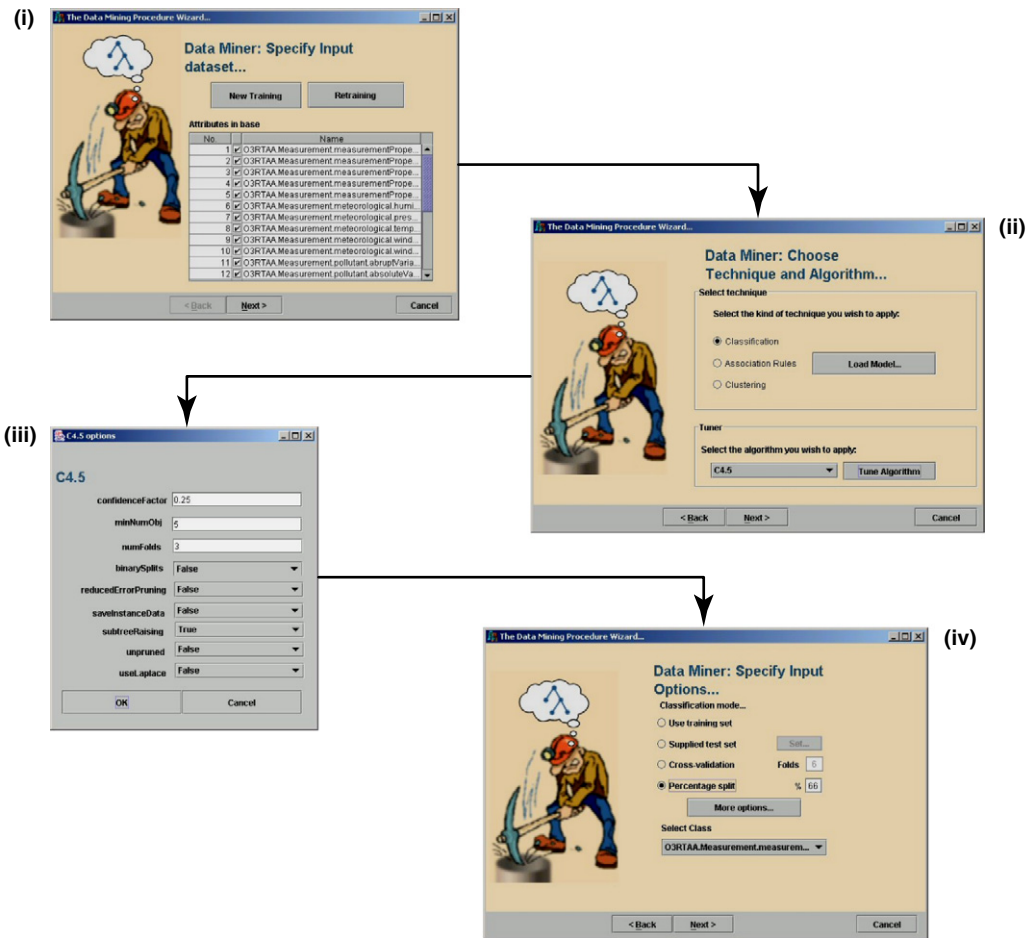


Fig. 6. Selecting input options (agent attributes, DM technique, algorithm, training parameters).

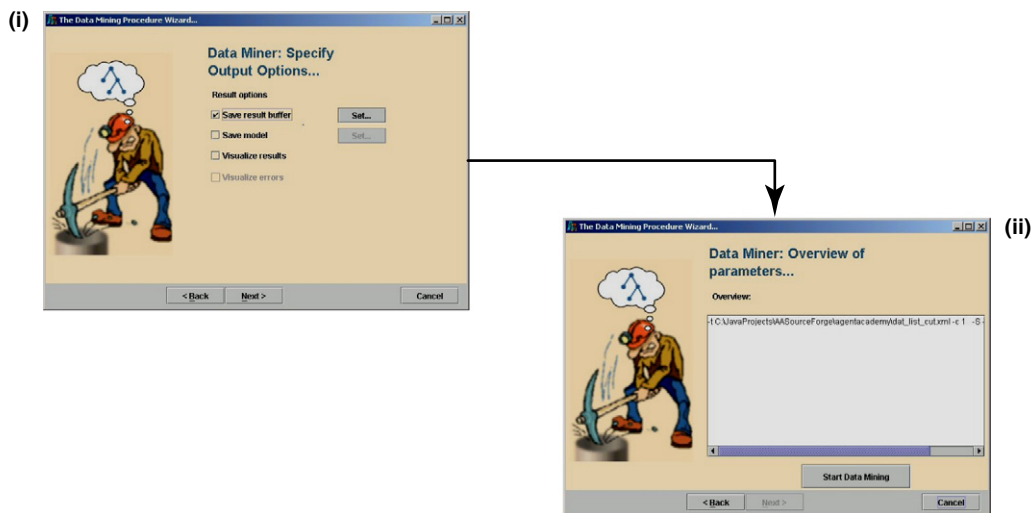


Fig. 7. Specifying data miner output and final error check.

supply-chain entities, such as customers, inventories, suppliers and the company itself. The constructed MAS works in close collaboration with an already implemented ERP system, in order to provide recommendations about customer orders.

IPRA consists of five cooperating layers. Each layer is manned by agents that perform different tasks related to the recommendation and coordinate through messages, using FIPA standards for agent communication. The architectural diagram of the system is depicted in Fig. 10.

```

<?xml version="1.0" encoding="UTF-8" ?>
-<INSTANCES task_agent_id="123456789" decision_structure_id="0">
-<ATTRIBUTES>
  <!-- ATTRIBUTES DEFINITIONS -->
  -<ATTRIBUTE>
    <NAME>MeasurementProperties.earlyAlarmFlag</NAME>
    <TYPE>numeric</TYPE>
  </ATTRIBUTE>
  -<ATTRIBUTE>
    <NAME>MeasurementProperties.measureRequest</NAME>
    <TYPE>numeric</TYPE>
  </ATTRIBUTE>
  -<ATTRIBUTE>
    <NAME>MeasurementProperties.measurementReliability</NAME>
    <TYPE>string</TYPE>
  </ATTRIBUTE>
  -<ATTRIBUTE>
    <NAME>MeasurementProperties.measurementResolution</NAME>
    <TYPE>string</TYPE>
  </ATTRIBUTE>
  -<ATTRIBUTE>
    <NAME>MeasurementProperties.measurementTimestamp</NAME>
    <TYPE>string</TYPE>
  </ATTRIBUTE>
  -<ATTRIBUTE>
    <NAME>Meteorological.humidity</NAME>
    <TYPE>real</TYPE>
  </ATTRIBUTE>

```

---

```

<?xml version="1.0" encoding="UTF-8" ?>
<DOCTYPE pmml_2_0.dtd (View Source for full doctype...)>
-<PMML>
+<Header copyright="issel.ee.auth.gr" description="The tree model of ONDA_base1">
+<DataDictionary numberOfFields="9">
-<TreeModel modelName="ONDA_base1">
+<MiningSchema>
-<Node score="0">
  <SimplePredicate field="O3_15" operator="lessOrEqual" value="61" />
  -<Node score="0">
    <SimplePredicate field="O3_15" operator="lessOrEqual" value="52" />
  +<Node score="0">
    -<Node score="0">
      <SimplePredicate field="O3_15" operator="greaterThan" value="52" />
    -<Node score="0">
      <SimplePredicate field="O3_15" operator="lessOrEqual" value="57" />
    -<Node score="0">
      <TRUE />
    </Node>
  </Node>
  -<Node score="0">
    <SimplePredicate field="O3_15" operator="greaterThan" value="57" />
  +<Node score="0">
    -<Node score="0">
      <SimplePredicate field="NOx" operator="greaterThan" value="33" />
    -<Node score="0">
      <SimplePredicate field="TEM" operator="lessOrEqual" value="13" />
    -<Node score="0">
      <TRUE />
    </Node>
  </Node>

```

Fig. 8. The initial XML file and the corresponding PMML output.

The workflow, top-to-bottom, is as follows: customers communicate with the MAS operator about their orders, therefore initiating a *Request For Recommendation* (RFR). Except from the products and their related quantities, customers also supply the operator with information such as desired payment terms, backorder policies and type of product transportation costs. The operator provides these preferences into the MAS through a graphical user interface (GUI) agent, the *customer order agent* (COA). COA is responsible for transmitting customer preferences into the system, initiating the recommendation procedure.

Order preferences are communicated to the *recommendation agent* (RA), the agent responsible for developing the final recommendation. At this point, the decision making process is initiated with RA requesting the profiles of the customer, the profiles of the ordered products and the profiles of their corresponding suppliers from the information processing layer agents. These profiles are then integrated into the final decision.

The three types of the information-processing layer agents apply DM techniques on legacy data sources, with the help of the *ERP Agent*. These types are *customer profile identification agent* (CPIA), *inventory profile identification agent* (IPIA) and *supplier Profile identification agent* (SPIA). CPIA and SPIA perform *clustering* on customer and supplier financial and managerial attributes in an attempt to categorize customers, while IPIA performs *market basket analysis*, in order to provide potential additional purchase items to customers. In general, the profiles and rules mined for customers, suppliers and products are generated from historical data, which constitutes a widely accepted predictor for future behavior. Data are the given input of the Data Miner, a specific technique and algorithm is then selected, DM is performed, and knowledge is extracted. This knowledge, which in fact represents the reasoning models of the agents, is stored into a profile repository that is updated periodically, by retraining the agents, i.e. reapplying DM on data. That

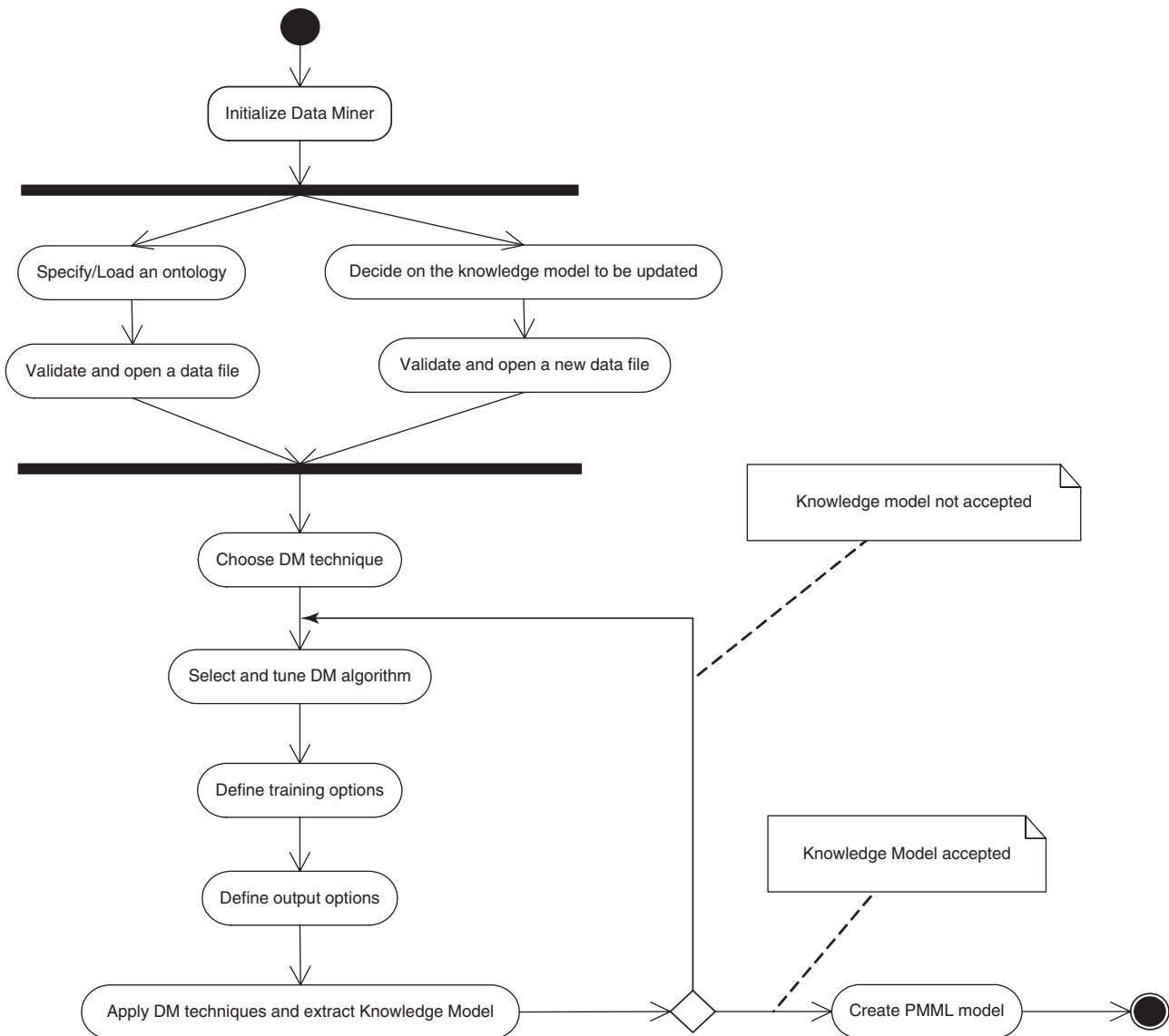


Fig. 9. The functionality of data miner.

way the system preserves its efficiency, autonomy and adaptability.

Table 2 illustrates the techniques and algorithms used, as well as indicative results on a specific test case. Association analysis was performed on 14,125 transaction records, while clustering was performed on 8,000 customers and 500 suppliers, respectively. Time limitations rose only on the initial association analysis phase, which lasted 62.856.3 s (Pentium III processor @800 MHz, 512 MB RAM). Retraining time was significantly less (~521 s for 200 new records). Nevertheless, since training is performed off-line and thereafter the produced model (.pmml file) is automatically incorporated into the agents' reasoning mechanism (at runtime), such time constraints have been considered as acceptable.

The final recommendation is produced by the RA. RA is responsible for gathering all profiles created by agents in the Information Processing layer, and producing the final

recommendation. This agent employs a rule-based reasoning mechanism implemented, employing JESS. The rules fired mirror enterprise policies applied to customer orders. In particular there are three distinct rule types RA can realize (Symeonidis and Mitkas, 2005):

1. Simple <If...Then...> statements, i.e. additional discounts or burdens to the total price of the order, based on the profiles returned.
2. Rules describing mathematical formulas, i.e. identification of re-order and order-up to level thresholds, for procuring additional components and products for the inventory.
3. Rules providing solutions to search or constraint satisfaction problems, i.e. identification of the most appropriate suppliers based on their corresponding value given by SPIA and their location of the depleted storage facility among others.

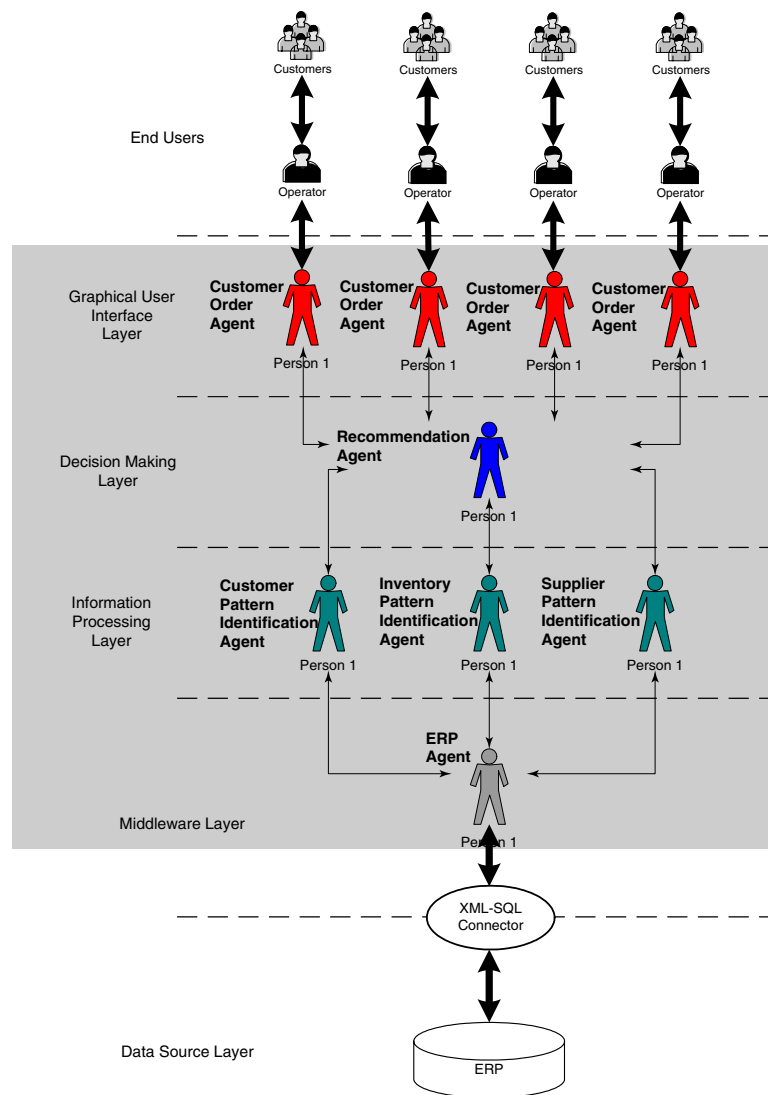


Fig. 10. The five cooperating layers, the agent types and other related entities.

RA demonstrates the importance of the coalition of merging knowledge from inductive DM techniques and expert knowledge in the form of rules. In our case, human experts are able to work more easily with higher level classification of customers and suppliers.

Further analysis of the IPRA system can be found at Symeonidis et al. (2003) and Kehagias et al. (2004). Concluding on IPRA, one may outline the key enhancements provided with respect to Legacy ERP systems:

1. IPRA allows the *on-the-fly* modification of static business rules, simply by replacing the rule documents used by RA, in contrast to legacy systems, where static rules are hard coded by the ERP vendor during the customization phase.
2. Dynamic business rules are applied both to data and knowledge extracted, while in legacy systems they are applied only on data.
3. IPRA supports the application of market basket analysis, customer and supplier evaluation automati-

cally, while in most ERP systems these processes are not supported, unless external modules (MBA, CRM and SRM, respectively) are employed. This, of course would imply increased (almost unaffordable) ERP costs.

4. Recommendations are automatically generated and the decision cycle-time is short, since it is not related to the database size, a parameter that also affects the reduction of information overload. This is not the case in other systems, though, since recommendations are produced through reports and their response is highly interdependent to their database.
5. IPRA proves highly adaptable, merely by adding new agents or modifying the policy of the existing one, while the cost of its enhancement is low, since the use of the Agent Academy platform allows easy design, instantiation and modification of the IPRA architecture. In legacy systems, expensive customization or third party COTS employment is essential.

Table 2  
Data mining for IPRA

DM-level	Technique	Algorithm (s)	Aim	Agent type	Results
Customer Management	Clustering	Maximin K-means	Customer profiling	CPIA	5 Clusters  Customer discount: 0–30%, customer priority: low–medium–high
Supplier Management	Clustering	Maximin K-means	Supplier profiling	SPIA	5 Clusters Supplier value: low–medium–high
Inventory Management	Association analysis	A priori	Market-basket analysis	IPIA	25 Rules  Support = 2%, Confidence = 90%

#### 4.2. Agents and data mining in EMIS

Environmental informatics (Enviromatics) is the research initiative that deals with issues concerning continuous monitoring of environmental measures and rapid changes, aiming to provide sound and valid decision-support solutions. More sophisticated applications are welcome to eliminate the time-overhead that lies between data producers (meteorological stations, environmental institutes and universities) and data consumers (interested parties, such as hospitals, governmental authorities, sensitive groups of people and the community in general). In order to fill the gap, cooperation of different functions is needed. Environmental surveillance, data storage, data manipulation, and alarm distribution to the end users are processes that must be modeled and coordinated properly, for the system to function according to the requirements.

Again, MASs coupled with DM can provide a successful paradigm capable of assessing the requirement for monitoring, management and distribution of environmental changes. O<sub>3</sub>RTAA acts a middleware application between field sensors and end-users, triggering alarms on air quality. The environmental attributes that O<sub>3</sub>RTAA monitors are shown in Table 3. Agents are cooperating in a three-layer pyramid, satisfying monitoring, management and distribution tasks, basic requirements of an environmental IT.

The architecture of the O<sub>3</sub>RTAA MAS is illustrated in Fig. 11. Information flows from the meteorological field sensors to the end-users of the system. Air-quality measurements are introduced into the system by *Diagnosis Agents*. At this layer preprocessing, such as measurement validation and estimation of missing values takes place. These preprocessed measurements are then forwarded to the *Alarm Agent*, which is responsible for triggering alarms, either formal or custom, with the former indicating dangerous situations (e.g. air-quality exceeding legal thresholds), while the latter alerting end-users about situations of their concern (in case of asthmatic people for example). Finally, the distribution agent forwards the

alarms raised by the Alarm Agent to the appropriate users by querying a user profile repository and by selecting the appropriate medium of transmission (i.e. e-mail or SMS).

DM is performed both at the contribution, as well as the management layer. The diagnosis agent performs classification on historical data for the validation of incoming measurements (incoming measurement validation—IMV) and for the estimation of missing values (missing value estimation—MVE), in case of hardware sensor malfunction. For the IMV process, the C4.5 algorithm is applied, and measurements are classified either as ‘valid’ or ‘invalid’. If a measurement is characterized as invalid by the system, the MVE process is triggered, in order to decide on an estimation of its value. This process employs the  $\sigma$ -FLNmap algorithm for the construction of the estimation decision model.

On the other hand, the alarm agent utilizes DM techniques in order to differentiate between custom alarms. The identification of custom alarms (ICA) process employs the C4.5 algorithm in order to create the certain custom alarm classes, and to classify all instances on them. Table 4 summarizes the above. The size of the initial dataset used was 105.216 records for three reporting stations, resulting to 315.648 instances. Initial training lasted 98.150, 7 s for  $\sigma$ -FLNmap and 27.114, 7 s for C4.5 (Pentium III processor @800 MHz, 512 MB RAM). Retraining lasted significantly lower. As stated above, these time periods are considered acceptable, since DM is performed off-line and this does not affect MAS efficiency.

O<sub>3</sub>RTAA employs DM in a different context with respect to other EMIS approaches: rather than predicting future states, our system attempts to evaluate the current state, a requirement imposed by state-of-the-art (near) real-time reporting systems. The overall goal of O<sub>3</sub>RTAA is to diminish human intervention, by successfully validating incoming information in real time, and by diffusing knowledge extracted to all interested parties, in a comprehensive and efficient manner. Further analysis of the O<sub>3</sub>RTAA system and its advantages can be found at (Athanasiadis and Mitkas, 2004).

Table 3  
The environmental attributes monitored by O<sub>3</sub>RTAA

Monitored attributes										
SO <sub>2</sub>	O <sub>3</sub>	NO	NO <sub>2</sub>	NO <sub>x</sub>	Wind velocity	Wind direction	Temperature	Relative humidity	Radiation	Pressure

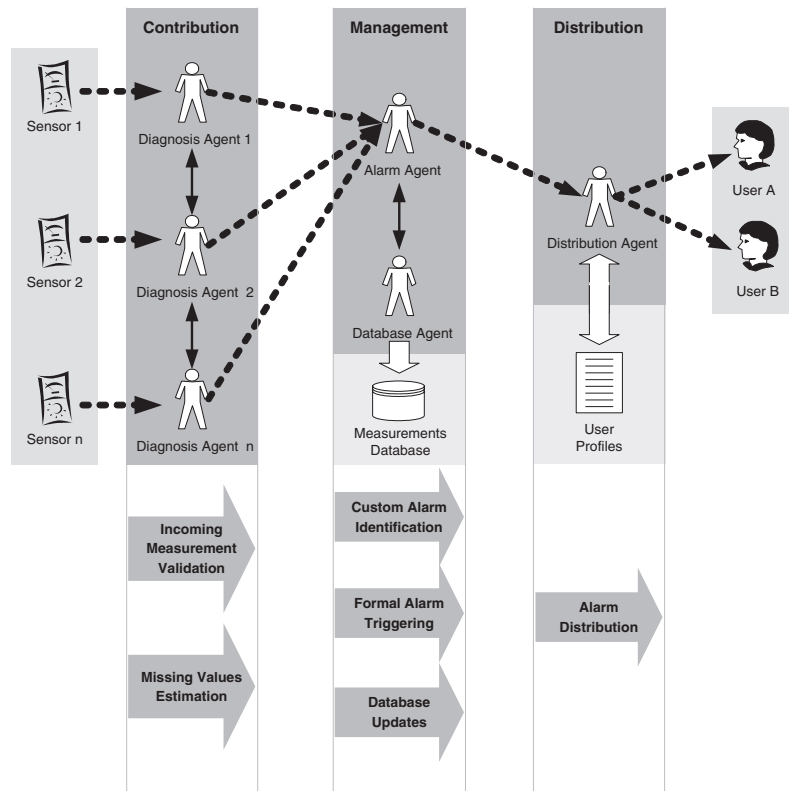


Fig. 11. The O<sub>3</sub>RTAA architecture, along with the processes executed by the agents.

Table 4  
Data mining for O<sub>3</sub>RTAA

DM-level	Technique	Algorithm (s)	Aim	Agent type	Results
Contribution level	Classification	C4.5	Measurement validation	Diagnosis	Classes: 2 (valid-erroneous) Prediction accuracy: 99.71%
Contribution level	Classification	$\sigma$ -FLNmap	Missing value estimation	Diagnosis	Value estimation Correlation coefficient: 0.9826
Management level	Classification	C4.5	Custom alarm identification	Alarm	Classes: 2 (low-med) Prediction accuracy: 93.80%

### 5. Conclusions

Through the brief presentation of two test cases, we have shown the feasibility of combining two powerful trends in computing: intelligent agents and DM. The use of AT leads to systems characterized by both autonomy and a distribution of tasks and control (Jennings et al., 1998a), while the utilization of data-mined knowledge offers the

capability of deploying “*Interesting (non-trivial, implicit, previously unknown, and potentially useful) information or patterns from data in large databases*” (Fayyad et al., 1996).

There are several examples of systems (Jennings and Wooldridge, 1998b), where agent intelligence is based on simple rules, leveraging the cost of development, tractability and maintenance. In dynamic and complicated environments like those of our applications, though, simple

expert engines prove insufficient. A series of complex and dynamic rules need to be applied, leading to conflicts and high human involvement, which results to the depletion of rationality of human experts. Moreover, automatic adaptability is not an issue, since it is exhausted to probabilistic inference (Caglayan et al., 1997). In the presented work, we argue that DM working in association with agent-based applications offers a strong autonomous and adaptable framework.

In our approach, we have revised the applicability potential of the three dominant DM techniques on MAS. Clustering grouping, Classification categorization and prediction, and Association Rule Extraction for correlation discovery. By the use of the data miner, we have created a uniform mechanism for incorporating resulting knowledge into agent reasoning. The benefits of this approach span in many directions:

- The combination of autonomy (MAS) and knowledge (DM) provides by definition adaptable systems. The process itself rejects profiles, rules and patterns not in use, while it adopts others that are frequently encountered.
- The rigidity and lack of exploration of deductive reasoning systems is overcome. Rules are no longer hard-coded into the system and their modification is only a matter of retraining. In addition, techniques such as association rule extraction have no equivalent in expert systems and provide the agents with an ability of probing and searching.
- Real-world databases often contain missing, erroneous data and/or outliers. Through clustering, noisy logs are assimilated and become a part of a greater group, smoothing down differences (i.e. IPRA), while outliers are detected and treated accordingly. In fact CPIA was able to identify a 2% of the customer population with very high value to the company, labeling it accordingly. Through classification, certain data records can be validated and estimated with regard to previous situations (i.e. O<sub>3</sub>RTAA). Rule-based systems cannot handle such data efficiently without increasing their knowledge base and therefore their maintenance cost. The presented approach favors the combination of inductive and deductive reasoning models. In both test cases, there were agents deploying deductive reasoning models (e.g. RA, alert agent), ensuring therefore system soundness. Nevertheless, these agents decide on data already preprocessed by inductive agents. That way, the dynamic nature of the application domains is satisfied, while deductive result sets (knowledge bases of deductive agents) become more compressed and robust.
- Even though the patterns and rules generated from DM cannot be defined as sound, there are metrics to evaluate the performance of the algorithms. Total mean square error (Clustering), support confidence (association rules) and classifier accuracy (classification) among others, exist for evaluating the different knowledge models and

approaches. Our approach takes under serious consideration the need for knowledge model evaluation and provides through the data miner a series of functionalities for visualization, model testing and model comprehension.

It is therefore our strong belief that DM extracted knowledge could and should be coupled with AT.

## Acknowledgments

This work has been partially supported by the European Commission, under the 5th Research Framework Programme (IST-2000-31050: Agent Academy: A Data Mining Framework for Training Intelligent Agent).

## References

- Adriaans, P., Zantige, D., 1996. Data Mining. Addison-Wesley, London.
- Agent Academy Consortium, the, 2004a. The Agent Academy Project [cited 1/09/2004]. Available from the World Wide Web <<http://AgentAcademy.iti.gr>>.
- Agent Academy Consortium, the, 2004b. Requirements and specifications of Agent Academy [cited 1/09/2004]. Available from the World Wide Web <<http://AgentAcademy.iti.gr/downloads.htm>>.
- Arthur, B.W., 1994. Inductive reasoning and bounded rationality. *American Economic Review* 84 (2), 406–411.
- Athanasiadis, I.N., Mitkas, P.A., 2004. An agent-based intelligent environmental monitoring system. *Management of Environmental Quality Journal* 15 (3), 238–249.
- Bellifemine, F., Poggi, A., Rimassa, G., 2000. Developing multi-agent systems with JADE. The Seventh International Workshop on Agent Theories, Architectures, and Languages (ATAL-2000), Boston, MA.
- Caglayan, A., Harrison, C., Harrison, C.G., 1997. Agent Sourcebook: A Complete Guide to Desktop, Internet, and Intranet Agents. Wiley, New York.
- Chen, Z., 1999. Computational Intelligence for Decision Support. CRC Press, Boca Raton, FL.
- Chen, M.S., Han, J., Yu, P.S., 1996. Data mining: an overview from a database perspective. *IEEE Transactions on Knowledge and Data Engineering* 8 (6), 866–883.
- Data Mining Group, the, 2001. Predictive Model Markup Language Specifications (PMML), ver. 2.0 [cited 1/09/2004]. Available from the World Wide web: <<http://www.dmg.org>>.
- Fayyad, U., Piatetsky-Shapiro, G., Smyth, P., 1996. Knowledge discovery and data mining: towards a unifying framework. In: Proceedings of The Second International Conference on Knowledge Discovery and Data Mining. AAAI/MIT Press, Portland, USA, pp. 82–88.
- Ferber, J., 1999. Multi-agent Systems—An Introduction to Distributed Artificial Intelligence. Addison-Wesley, London.
- Fernandes, A.A.A., 2000. Combining inductive and deductive inference in knowledge management tasks. In: Proceedings of the 11th International Workshop on Database and Expert Systems Applications (First International Workshop on Theory and Applications of Knowledge Management—TAKMA 2000). IEEE Computer Society, Silver Spring, MD, pp. 1109–1114.
- Friedman-Hill, E.J., 2003. Jess, The Expert System Shell for the Java Platform, version 6.1, CA, Sandia National Laboratories [cited 1/09/2004]. Available from the World Wide Web <<http://herzberg.ca.sandia.gov/jess/>>.
- Galitsky, B., Pampapathi, R., 2003. Deductive and inductive reasoning for processing the claims of unsatisfied customers. In: The Proceedings of the 16th International Conference on Industrial and Engineering

- Applications of Artificial Intelligence and Expert Systems (IEA/AIE 2003). Springer, Heidelberg, pp. 21–30.
- Han, J., Kamber, M., 2001. *Data Mining: Concepts and Techniques*. Morgan Kaufmann, San Diego.
- Jennings, N.R., Wooldridge, M.J. (Eds.), 1998b. *Agent Technology: Foundations, Applications and Markets*. Springer, Berlin.
- Jennings, N.R., Sycara, K., Wooldridge, M.J., 1998a. A roadmap of agent research and development. *The International Journal of Autonomous Agents and Multi-Agent Systems* 1, 7–38.
- Kaburlasos, V.G., Athanasiadis, I.N., Mitkas, P.A., 2006. Fuzzy lattice reasoning (FLR) classifier and its application on ambient ozone estimation. *International Journal of Approximate Reasoning*, paper in press, available online August 2006.
- Kargupta, H., Hamzaoglu, I., Stafford, B., 1997. PADMA: PArallel Data Mining Agents for scalable text classification. In: *The Proceedings of High Performance Computing '97*.
- Kehagias, D., Chatzidimitriou, K., Symeonidis, A.L., Mitkas, P.A., 2004. Information agents cooperating with heterogeneous data sources for customer-order management. *The 19th Annual ACM Symposium on Applied Computing (SAC 2004)*, Nicosia, Cyprus, pp. 52–58.
- Kero, B., Russell, L., Tsur, S., Shen, W.M., 1995. An overview of data mining technologies. *The KDD Workshop in the 4th International Conference on Deductive and Object-Oriented Databases*, Singapore.
- Kodratoff, Y., 1988. *Introduction to Machine Learning*. Pitman Publishing, London.
- Mitkas, P.A., Symeonidis, A.L., Kehagias, D., Athanasiadis, I., 2002. An agent framework for dynamic agent retraining: agent academy. *Workshop on Challenges and Achievements in e-business and e-work*, Prague, pp. 757–764.
- Mitkas, P.A., Kehagias, D., Symeonidis, A.L., Athanasiadis, I.N., 2003. A framework for constructing multi-agent applications and training intelligent agents. In: *Proceedings of the Fourth International workshop on Agent-Oriented Software Engineering (AOSE-2003)*, held at Autonomous Agents & Multi-Agent Systems (AAMAS 2003), Melbourne, Australia, Springer, Berlin, pp. 1–16.
- Mohammadiam, M., 2004. *Intelligent Agents for Data Mining and Information Retrieval*. Idea Group Inc.
- Piatetsky-Shapiro, G., Frawley, W.J., 1992. *Knowledge Discovery in Databases*. MIT Press, Cambridge, MA.
- Quinlan, J.R., 1993. *C4.5: Programs for Machine Learning*. Morgan Kaufmann, San Mateo.
- Shapiro, J.F., 1999. Bottom-up vs. top-down approaches to supply chain modelling. In: Tayur, S., Ganeshan, R., Magazine, M. (Eds.), *Quantitative Models for Supply Chain Management*. Kluwer, Massachusetts.
- Stolfo, S.J., Prodromidis, A.L., Tselepis, S., Lee, W., Fan, D.W., Chan, P.K., 1997. Jam: Java agents for meta-learning over distributed databases. In: *Proceedings of the Third International Conference on Knowledge Discovery and Data Mining*, AAAI Press, Newport Beach, CA, pp. 74–81.
- Symeonidis, A.L., Mitkas, P.A., 2005. *Agent Intelligence through Data Mining*. Springer, USA.
- Symeonidis, A.L., Kehagias, D., Mitkas, P.A., 2003. Intelligent policy recommendations on enterprise resource planning by the use of agent technology and data mining techniques. *Journal of Expert Systems with Applications* 25 (4), 589–602.
- Talavera, L., Cortes, U., 1997. Inductive hypothesis validation and bias selection in unsupervised learning. In: *Proceedings of the Fourth European Symposium on the Validation and Verification of Knowledge Based Systems (EUROVAV-97)*, Leuven, Belgium, pp. 169–179.
- Witten, H.I., Frank, E., 1999. *Data Mining: Practical Machine Learning Tools and Techniques with Java Implementations*. Morgan Kaufmann, New Zealand.
- Wooldridge, M., 1999. *Intelligent agents*. In: Weiss, G. (Ed.), *Multiagent Systems*. MIT Press, Cambridge, MA.
- Zhang, Z., Zhang, C., Zhang, S., 2003. An agent-based hybrid framework for database mining. *Journal of Applied Artificial Intelligence* 17, 383–398.