Developmental and Operational Processes for Agent-Oriented Database Navigation for Knowledge Discovery

M. Brian Blake
Department of Computer Science
Georgetown University System
234 Reiss Science Building
Washington, DC 20057
blakeb@cs.georgetown.edu

Center for Advanced Aviation
Systems Development (CAASD)
The MITRE Corporation
McLean, Virginia 22102
bblake@mitre.org

Andrew B. Williams
Department of Electrical and Computer Engineering
University of Iowa
Iowa City, Iowa 52240
abwill@eng.uiowa.edu

ABSTRACT

Knowledge discovery in databases (KDD) is an area that has become important to organizations that search for trends and useful information from their raw database information. KDD can be a tedious and repetitive human-driven process with respect to extracting the relevant data-sets from databases for processing in the relevant learning algorithms. We investigate an approach where agents can control the extraction of the data-sets. We show a software developmental process and paradigm for programming information agents to extract data-sets based on a methodology we refer to as "extraction hints". We discuss what data modeling approaches can be used to allow these information agents to be reusable across various domains and databases. Lastly, using the aviation domain for motivation, we show the design of an agent architecture toward the further automation of KDD using agents.

1. Introduction

Over the past decade, government and industry organizations have enhanced their operations by utilizing emerging technologies in data management. Advances in data modeling approaches and database technologies have led to better storage and maintenance of business data and knowledge. Furthermore, the normalization of such data has increased the ability of organizations to extract useful knowledge from raw operational data. These approaches have led to the popularity of many areas, such as knowledge discovery in databases and data mining (KDD) [8]. Over the past decade, there has been increased acceptance of the maturing KDD technologies, but human-driven aspects need further automation.

1.1 Problems with Knowledge Discovery

Knowledge discovery and data mining (KDD) in databases consists of such phases as selection, pre-processing, transformation, data mining, and interpretation/evaluation. Excluding the data mining phase, where there are a plethora of automated learning algorithms and processes, other phases are mostly human-driven. Knowledge discovery is important in some domains, such as the aviation and bioinformatics, where raw data is complex and heterogeneous. This data can be represented in normalized and de-normalized databases.

A problem in such domains is that human experts must manage the KDD process, which tends to be iterative and time-consuming. In addition, human experts do not always know which subsets of data are most valuable for the data mining scenarios. In most domains, the number of possible attribute combinations makes it improbable for human-enacted "trial-and-error" learning sessions to be effective. Other algorithms for sampling and association rule pruning [18] are helpful in this domain but do not address the need for true automated approaches for iterative knowledge discovery sessions.

1.2 Applying Agents to KDD

Our approach seeks to apply agents for the semi-automated management of the KDD process and draws on applications in association rule discovery. Currently, there has been significant progress in the creation and enhancement of algorithms that create association rules from database records. Mostly, these technologies support human-driven learning sessions. Human domain experts suggest database columns as attributes and classifiers that are appropriate for rule learning algorithms. These attributes are tagged using a database schema viewer or extracted from the database (using human-proposed queries) as *datasets* into flat file formats.

From the initial suggestions of a human domain expert, an information agent will pre-process and transform data in preparation for data mining. The information agent submits the data to third-party, data mining software. From experience with the individual information agent, we lay the foundation for a future architecture toward the use of multiple learning agents to evaluate results and iteratively suggest the initiation of new KDD routines.

There are several significant research questions related to applying agents to KDD such as:

1. What database schema is most effective and reusable across domains for agent navigation through the database?

2. How can this information agent architecture be generic enough to be reusable across domains?

3. Can extraction heuristics, mainly defined with database-specific semantics, be derived from historical user actions and are such heuristics effective ?

This paper addresses *questions 1 and 2* and is organized as follows. In Section 2, we survey related projects that use agents for data mining. The following section will discuss the appropriateness of various relational data modeling techniques for agent navigation. Furthermore, there is the description of a reusable meta-information loading process to prepare databases for agent interaction. In Section 4, we discuss how human extraction hints can translate to agent programming using the aviation domain for motivation. In Section 5, we discuss the information agent architecture and user interface prototype developed in this work and briefly detail questions answered by the architecture. In Section 6, we discuss how results from this work motivated a new multi-agent architecture to address *question 3*. Finally, we detail our on-going agent architecture and design to further automate the KDD process.

2. Related Projects

In general terms, current technologies and applications that support knowledge discovery in databases can be categorized as tightly coupled approaches and loosely-coupled approaches. Tightly-coupled approaches [1][15][18] take advantage of database performance and capabilities by incorporating data mining models inside the underlying database. Thus, the knowledge discovery tools are bound to the underlying database. Loosely-coupled approaches, generically, extract data-sets from the database. This allows knowledge discovery tools to migrate to numerous database technologies and applications. Most commercial applications offer loosely-coupled tools [6][19][22].

Both of these approaches are most effective when human domain experts initiate and manage knowledge discovery sessions. While most researchers try to use fast, efficient algorithms to generate association rules and ways to reduce the number of irrelevant results, there is little work that provides automated data-set servers or even sophisticated user interfaces for data-set generation from databases. Similarly, these association rule-learning algorithms can discover attributes, which are most likely to have associations with other attributes. However, there is a shortcoming in the domain where users must identify complex qualifying events (i.e. the composition of unrelated attributes). Generating data-sets containing information for qualifying events is a time-intensive process for which there is currently minimal support. Moreover, current tools have little support for discovering the attributes or group of attributes that are most relevant to these qualifying events.

Considering related agent research, there are few approaches that use of agents in KDD. Maes [12] performed initial research using agents that control various information gathering and associated tasks. One such area was the use of personal assistant agents for machine learning. In fact, later implementations of this work led to the establishment of Frictionless Commerce. InfoSleuth [16] is a multi-agent architecture that gathers information from distributed sources and intelligently presents the composite information. Payne [16] details areas where interface agents can be used for learning and rule induction. IDM [4] is another multiple agent architecture that attempts to do direct data mining that helps businesses gather intelligence about their internal commerce. Helmer et. al [9] uses intelligent agents to mine security data.

Given the survey of related work, there were no findings of significant related projects that try to define agent heuristics and architectures for knowledge discovery in databases. The work of Maes and InfoSleuth are more data dissemination-oriented than the learning goals of this work. The goals of Payne and Helmer are more towards Internet-based domains while this work is toward a generic architecture and model that will allow learning directly from a relational database model. The architecture in IDM has some similarity to the architecture in this work, however the approach is aims toward the business domain and more toward applying learning algorithms as opposed to automating aspects of knowledge discovery.

3. Relational Models for Agent Navigation

Our current work has formalized a process and framework that combines the benefits of both tightly-coupled approaches and loosely-coupled approaches. This process incorporates the semantics for standardizing entities in the database through the inclusion of meta-information. This additional meta-information is not composed of data mining-specific models, but information that shows how entities are correlated (correlation attributes). Human-proxy agents or middle agents are used to populate the meta-information in the database, intelligently build data-sets, and also act as wrappers to the data mining software. This work also supports the specification that defines qualifying events as a function of database semantics. In this section, there is a description of our approach to database preparation/pre-processing for agent interaction and navigation.

3.1 Difficulties of Common Relational Modeling Techniques with Agent Navigation

The explanations in this section will be motivated around aviation-based data-sets. This is appropriate because, in later sections, our agent approach will be demonstrated on aviation data. In general, organizational databases follow a normalized or de-normalized approach [11]. However, most commonly, organizations take advantage of *hybrid* approaches that combine the benefits of both.

A main principle followed in this work is the development of agents that have relatively simple non-domain-specific functionality. Development in this fashion will allow agents to be reusable as they are deployed on different databases. The major problem with both normalized and de-normalized approaches is the specificity required for agent navigation. To motivate this problem, we consider the two modeling approaches (normalized and de-normalized) for data that represents airport terminal weather and airport terminal performance. Airport terminal weather [13] include fields such as visibility, ceiling, barometric pressure, temperature, wind speed, etc. Airport terminal performance [20] has information such as arrival/departure rates, arrival/departure cancellations, etc. This information is modeled in both normalized and denormalized form as represented in Figure 3.1. In the de-normalized tables, all information relevant to airport terminal weather and airport terminal performance are included in their respective tables. Related information are not connected by joining tables, however joins can be built on the tables using query languages, if the fields have the same basic data types and formats.

Though the normalization could be much stricter, for explanation purposes, normalization principles are demonstrated on the right side of Figure 3.1. Common fields of time and location are represented in other tables that are referenced by the main Terminal Weather and Terminal Performance tables.

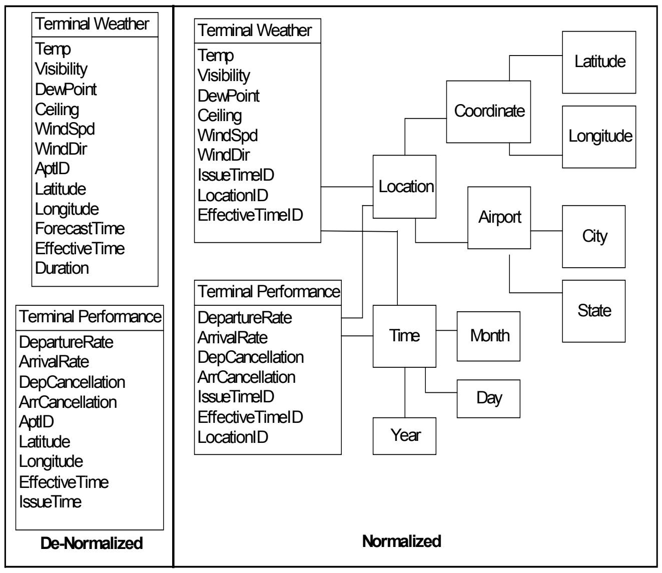


Figure 3.1 Different Relational Modeling Approaches

In initial studies, we attempted to model agents that navigate both modeling paradigms. We discovered that using either of these modeling paradigms would require autonomous agents to be tightly-coupled to the database. Answering questions, such as "What is the arrival rate trend when temperature is below zero degrees at Dulles International Airport?", required agents to have more than knowledge about the information, but also database semantic knowledge.

In the de-normalized models, agents would have to initially know which fields are used to relate tables. Thus, hard-coding agents to specific fields would be necessary. In addition, the assumption would be that these fields use similar data types and formatting which is not always true. Moreover, making joins across the de-normalized tables tended to be problematic with large data-sets.

In the normalized models, though common fields are modeled in the database, agents are required to have an understanding of the other descriptive tables. Most data mining software operates automatically from the normalized tables using numerous learning algorithms. However, such approaches still require human users to direct the KDD process. These data mining approaches use the key constraints found in relational system tables to understand table connectivity. In early approaches, we attempted to use these key constraints for agent navigation. This proved to be ineffective because of the variation of system tables across multiple relational database management systems. In addition, modeling behavior for tables having multiple keys to one another also tended to be problematic. Again, this required complex database semantic information to be programmed into the agents.

For agents to navigate through databases while also allowing the maximum possibility for reuse, we found it necessary to follow several conditions:

- ***Limit the coupling of database semantics in the agents***
  This would allow agents to be functional on multiple relational database management systems.
- ***Allow joins, however joins should not be mandatory***
  This will allow agents to be able to optimize their operations in some situations by using single table queries.
- ***Complex entity relationships should be made prior to data-set generation in a "data preparation" phase***
  Relating information real-time prove to be an ineffective approach the most effective approach

3.2 A Process for Programming the KDD Agents

In this modeling technique, we make the assumption that the initial database contains mostly de-normalized entities. This approach uses a process that standardizes the database into an environment that agents can navigate. This is a reproducable process that is valid for any database with entities that share related columns. This process includes the creation of *fact* tables [11] that connect entities. Information agents are configured to manage this loading and standardization process. Similar to tables above, consider entities that represent aviation terminal weather, terminal performance (arrival and departure rates), and terminal operations as illustrated in Figure 3.2.

The process for creating the database structure in Figure 3.2 is as follows.

1. Human domain expert identifies related columns across multiple entities and also specifies the target data types
2. The human also identifies specific range for the related columns/attributes (i.e. the range to be analyzed)
3. Assuming columns are sorted, information agents develop generic CorrelationAttribute entities and preload records for data within that specified range
4. Agents query the raw data tables and record-by-record create correlation records in the CorrelationSpace entities based on matching correlation attributes.
5. Additional detailed entities (Airports and Time) can be generated/connected to further describe the correlation attributes (This post-loading-process can be used in normalized models).

In Figure 3.2, the airport identification (AptID) and event times (EffectiveTime) are identified as common columns, as depicted with ovals and rectangles, respectively. In this example, these columns are obvious, but there are situations where the relations between entities are only evident to domain experts. By using the generic CorrelationSpace and CorrelationAttribute table structures, agent capabilities can be duplicated on any database containing these meta-information tables.

The agents are not limited by having to "hard-coded" column names and table locations, instead queries can be managed based on pre-defined conditions. Also, this process essentially pre-loads joins into the database model. During loading, agents do not create duplicate correlation records (CSpaceID), instead existing correlation records are reused. In this way, for example, when a weather record has the same correlation conditions (the same time and same airport) as a terminal performance record then both records have the same CorrelationSpace identification (CSpaceID).

The modeling concepts used to create this structure are not novel. These concepts are consistent with database normalization routines. The benefit is the separation of the domain-specific information from the database semantics. The main idea is for the agents to be able to perform basic text comparisons without knowledge of domain-specific concerns when generating data sets.

This schema makes it possible for agents to answer such questions as:

- What was the arrival rate at Dulles Airport on September 11th?
- What was the cancellation count of arrivals at La Guardia Airport when the temperature was less than 0 degrees?
- What was the arrival rate at Reagan National Airport on days when Ground Stops were issue?

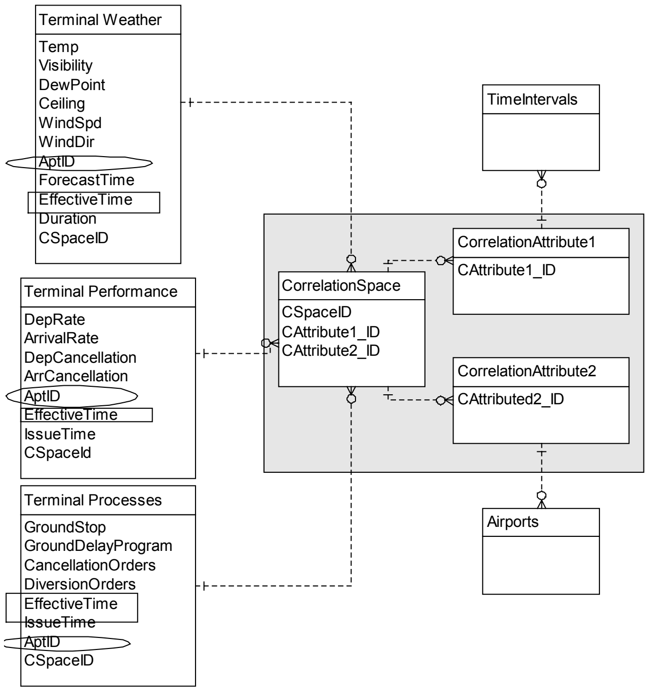


Figure 3.2 Database Preparation for Agent Navigation

The major benefit of this modeling approach is that agents can now vary the correlation attributes (in this example case, time and airport locations) to find deeper knowledge. For example:

- When the cancellation count at La Guardia Airport is less than one per fifteen minute period, then what is the trend for weather in Newark at that same time?
- What was the ceiling and visibility 4 hours before a ground stop is issued at Reagan National Airport?

The automation of these variations is not a current capability but will be possible in the architecture described in Section 7.0.

4. Programming Agents with Extraction Hints

Domain experts, at times, need to determine trends based on a composite list of attribute constraints. In this work, we specify these composite constraints as hints or *qualifying events*. In the example presented in Section 3.2, qualifying events can be defined as the combination of weather, processes, and performance conditions.

We define two major aspects of the qualifying event that human users can provide. The first aspect, called the *search criteria*, allows the input of a basic constraint. For example, a user might direct the agent to explore situations where the ceiling was greater than 1000 feet and temperature was greater than 90 degrees. This search criteria will specify records from the Terminal Weather table where the values of the ceiling column is greater than 1000 and the temperature column is greater than 90.

Once a search criteria is set, the user can also suggest *information points*. An information point is defined as other information related to the search criteria, as constrained by the correlation attributes (in this case, time and airport). For example, a user may specify an information point as the visibility at the same time and area. A user can also specify an information point for a different location for a different time, perhaps at another airport and for the time 3 hours before the time captured when the search criteria is met. Both the search criteria and information points are composed of the correlation attributes represented in the correlation records.

D	Generated Data Set
s	Resulting Search Criteria Data-points
i	Resulting Information Point Data-points
t	RDBMS table name
f	RDBMS column name
C	Set of all correlation attributes
X	Set of user-specified filter information
$q(C)$	Function that generates information from directly specified set of correlation attributes
$n(C,X)$	Function that generates information from directly specified set of correlation attributes while constraining returns by user-specified filters
c_{time}	Correlation attribute 1 (time-based)
c_{area}	Correlation attribute 2 (area/airport-based)

Table 4.0 Variables Describing Aspects of a Qualifying Event

The components of the search criteria and information points are defined in Table 4.0. For the variables in Table 4.0, the search criteria data-point, *s*, is created from a function, *n,* that gathers data from a relational database given the table name, *t*, and field name, *f*, and further constrained by correlation attributes, c_{time} and c_{area} and the set of all other database filters, *X*. This relation is further defined as

$$s = n_{t,f}(c_{time}, c_{area}, X)$$

The data-point from the information point specification, *i*, is created from a function, *q,* that gathers data from a relational database given the table name, *t*, and field name, *f*, and further constrained by correlation attributes, c_{time} and c_{area}. In this relation defined as

$$i = q_{t,f}(c_{time}, c_{area})$$

there is not a need for other database filters because the information points are based on instances related to the search criteria. Finally, the user-directed data-set, *D*, can be defined as the series of all search criteria datapoints and information points.

$$D = < \Sigma s, \Sigma i >, \text{ where some } s_n \text{ or } i_n \text{ is the classifier}$$

All data-sets can be captured in a flat file and later processed using data mining software. When processing the generated data sets using a association learning algorithm, one of the points from the search criteria, $s_{1\ldots n}$, or one of the information points, $i_{1\ldots n}$, must be designated as the learned class or classifier, based on all the other attributes.

5. An Information Agent for KDD

Thus far, the initial work described could have been achieved without the use of agent technologies. Agents have mostly been used in situations requiring autonomy to perform routine tasks [10]. Middle agents can act as proxies for their human counterparts. In the initial work, regular component-based concepts and development could fulfill the requirements in specifying and enacting complex knowledge discover sessions. However, in anticipation of further research, agent qualities were incorporated in the initial implementation. This architecture extended initial agent architecture research [2].

5.1 Functionality of the Information Agent

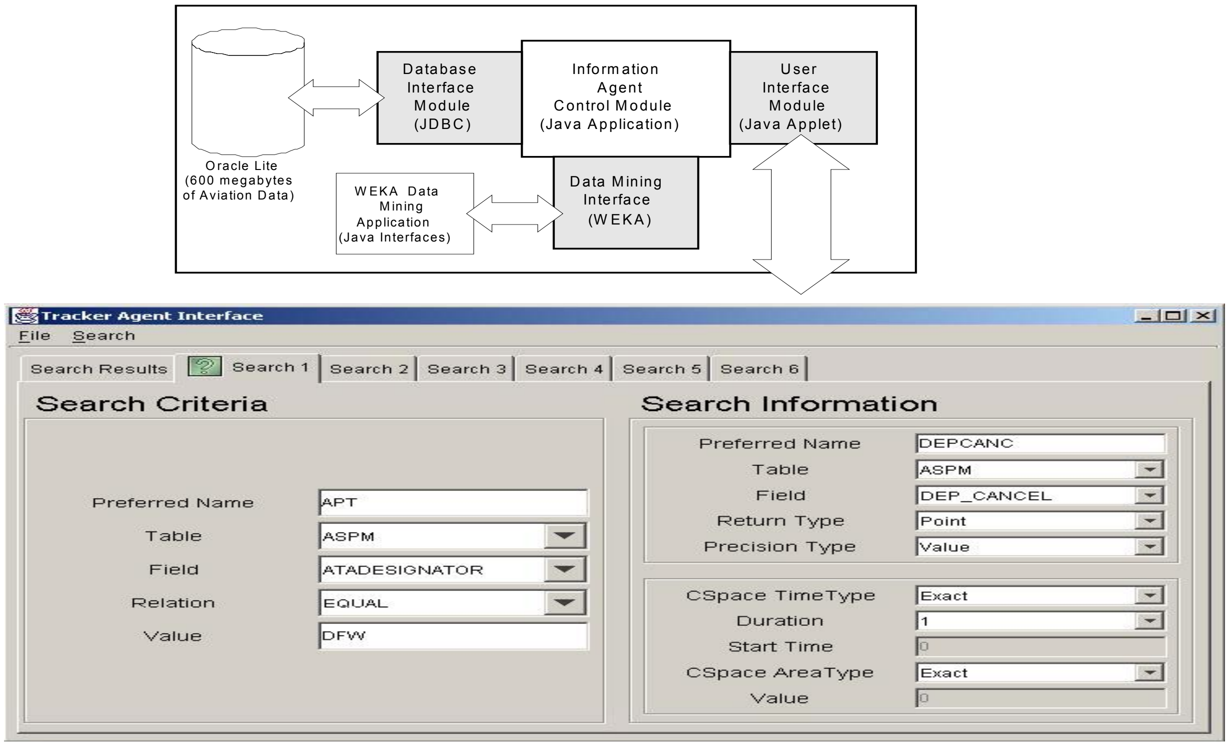
The main agent characteristic is the ability for the information agents to independently run a knowledge discovery session once the human user specified a qualifying event. Information agents in this work perform the following tasks during one session.

- Present graphical interface to human domain expert
- Extract qualifying event from interface settings
- Generate data-sets from the database
- Correct and record actions performed from incomplete data
- Submit data-sets to the data mining software
- Report results and corrective actions

The initial implementation is an excellent starting point to design and implement higher-level agents that analyze the results of the information agents. These higher-level agents or learning agents can trigger other information agents based on the results. Multiple learning agents can collaborate on results across multiple knowledge discovery sessions, thus gaining higher-order knowledge.

5.2 The Information Agent Architecture

The agent architecture consists of an information agent with a graphical user interface that allows human domain experts to specify qualifying events. The information agent has an internal database interface module that acts as an interface to a local relational database. This module utilizes Java Database Connectivity (JDBC) technologies and the Oracle 9i Lite Personal database. In addition, this information agent also has an interface to a third-party data mining tool, specifically the WEKA tool. The initial application contains specialized loaders supporting the data model approach Figure 3.2. This initial architecture and graphical user interface is illustrated in Figure 5.0.


Oracle Lite
(600 megabytes of Aviation Data)
Database Interface Module (JDBC)
Information Agent Control Module (Java Application)
User Interface Module (Java Applet)
WEKA Data Mining Application (Java Interfaces)
Data Mining Interface (WEKA)
Tracker Agent Interface
File
Search
Search Results
Search 1
Search 2
Search 3
Search 4
Search 5
Search 6

Search Criteria

Preferred Name	APT
Table	ASPM
Field	ATADESIGNATOR
Relation	EQUAL
Value	DFW

Search Information

Preferred Name	DEPCANC
Table	ASPM
Field	DEP_CANCEL
Return Type	Point
Precision Type	Value
CSpace TimeType	Exact
Duration	1
Start Time	0
CSpace AreaType	Exact
Value	0

Figure 5.0 A Information Agent Architecture and User Interface to Support KDD

The graphical user interface allows the user to insert search criteria and search information (information points) while varying the correlation attributes of time and area (CSpace TimeType and CSpace AreaType). Though the number tabs are configurable, the user interface in Figure 5.1 has 6 tabs corresponding to pairs of search criteria and information points. The *Exact* shown in TimeType and AreaType textboxes means there is no variation in the correlation.

6. Application to the Aviation Domain

The information agent developed in this work proved to be useful on aviation data-sets. This agent architecture was deployed on a subset of relational data from repositories owned by the Center for Advanced Aviation System Development (CAASD) at The MITRE Corporation. The entire repository contains over 600 gigabytes of data. For purpose of experimentation, the pre-processing method specified in Section 3.2 was executed on terminal weather, performance, and policy data (a simplified subset of the model shown in Figure 3.2) from the period of May 2001 – October 2001. This subset of data represented approximately 600 megabytes of data available for knowledge discovery.

6.1 Evaluating Validity of the Information Agent

One case performed to validate the initial information agent architecture was to re-engineer flight rules. When visibility is below a certain distance and ceiling (cloud-level) is below a certain altitude, the FAA institutes instrument flight rules (IFR) as opposed to the visual flight rules (VFR) during normal conditions. This means that any pilot flying a plane should be able to operate the plane using solely with instruments when IFR is instituted. How these rules are determined is a known heuristic, therefore if the right qualifying event is given by a human expert, the information agent should be able to discover the rules governing the FAA flight rules policies. Two data groupings corresponding to this scenario are listed below.

Weather	*FAA Processes*
Temperature, Ceiling, Visibility, Barometric Pressure, WindSpeed,Wind Direction, Dew Point	Ground stop, Ground Delay Program, Flight Rule, Miles-In-Trail Restriction, Cancellations, Diversions

Table 6.0 Sample Attributes from the Experimental Data Model

A human expert would set a search criteria to capture records when the ceiling (i.e. elevation of lowest cloud level) is less than 700 feet. The domain expert may also suggest that temperature, visibility and wind speed may also have some effect by setting these columns as information points. Finally, the expert would set another information point as the flight rule condition and set that column as the classifier.

In this evaluation scenario, there were 90% of the experiments where the information agent correctly discovered that only ceiling and visibility are pertinent to flight rules. These results showed promise to the domain experts when using the tool for other *unknown* associations. The trials that were not successful generally occurred when learning algorithms were applied to airports with relatively fair weather during the chosen six month period. The major results of the information agents were the lessons learned that have led to the design of our new multiple agent architecture detailed in the next section.

6.2 The Motivation for Multiple Agents

The use of the information agent at CAASD has been successful in discovering trends. Some interesting, unexpected trends have been the seeds for discovering new areas for analysis. One discovery in multiple trials of using the information agent is the trend of the human users to ask similar follow-up questions leading to follow-up KDD routines.

To motivate this point, we use the results determined in Section 6.1 that only ceiling and visibility are pertinent to flight rules. In response to these results, a domain expert may ask several follow-up questions that are not known:

1. Does Wind Direction or Dew Point also make a difference?
2. Does the same pertinent conditions of (ceiling and visibility) effect other processes like cancellations?
3. Does the same conditions effect cancellation in the past (12 hours prior) or in the future (12 hours after)?

The research question that arises from these follow-up questions is "Can these follow-up questions be defined with database semantics and can agents autonomously generate these follow-up questions?". Our current architecture inserts a user monitoring agent to analyze these follow-up questions and determine the differences between these questions and the previous questions. These differences are extracted in terms of the underlying database semantics. These semantics can be further transformed into heuristics used to program other agents, learning agents. Some heuristics in terms of the agent programming (defined in Section 4.0) are:

1. Create a new qualifying event where other attributes from the same data groupings are added while deleting non-pertinent attributes (as in question 1 and 2)
2. The correlation attributes (timeframe and location) can be varied (as in questions 3 and 4).

The above are just samples of the type of heuristics that can be programmed in the agent to allow new qualifying events to be generated and new knowledge discovery sessions to be processed concurrently.

7. A Multiple Agent Architecture for KDD

The design of the current multiple agent architecture is aimed towards a framework for multiple information agents and the introduction of higher order agents, which can be referred to as learning agents. In addition to learning agents, there are user monitoring agents that view the activities of domain experts that use the stand-alone information agents. Information agents will perform tasks similar to their described current functionality. Learning agents will incorporate the heuristics for the iterative instantiation of information agents. This architecture is illustrated in Figure 7.1.

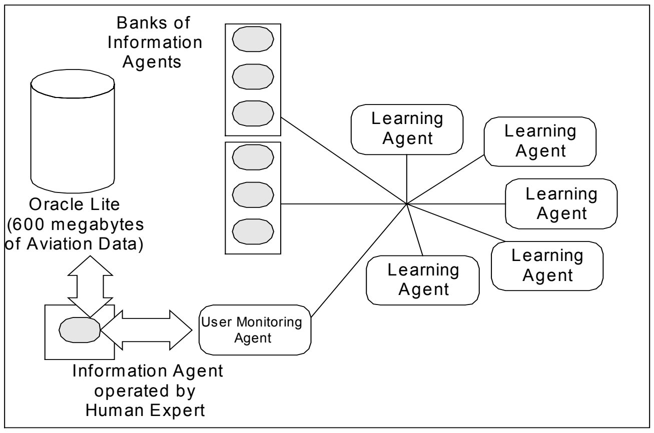

Banks of Information Agents
Learning Agent
Learning Agent
Learning Agent
Learning Agent
Learning Agent
Oracle Lite (600 megabytes of Aviation Data)
User Monitoring Agent
Information Agent operated by Human Expert

Figure 7.1. Multiple Agent Architecture Towards the Full Automation of KDD

The learning agents, information agents, and user monitoring agents will coordinate and communicate using Linda-based communication. Earlier work investigated the use of tuple-space communication for agent collaboration for workflow management [3]. This approach and existing modules using JavaSpace technology have been extended and incorporated into this architecture. The control module for the learning agent interprets the results from the information agents and heuristics created from the results of the user agents.

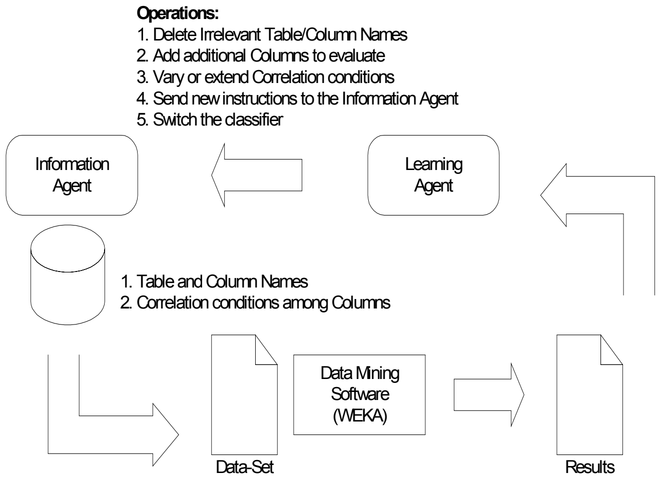

Operations:
1. Delete Irrelevant Table/Column Names
2. Add additional Columns to evaluate
3. Vary or extend Correlation conditions
4. Send new instructions to the Information Agent
5. Switch the classifier
Information Agent
Learning Agent
1. Table and Column Names
2. Correlation conditions among Columns
Data Mining Software (WEKA)
Data-Set
Results

Figure 7.2 Agent-Based KDD Operational Processes

The control flow of the agents that manage off-line knowledge discovery sessions is illustrated in Figure 7.2. Information agents create data-sets based on a qualifying event from a human user. The qualifying event, as explained in Section 4.0, consists of column names and how those values are correlated. Once the data-set is processed in the data mining software, association rules are created and sent to a learning agent. Also in this step, the columns that are most valid and the strength of the rules are combined with the learning agents' existing knowledge of other available database columns. The learning agent now uses heuristics and all the available information to produce a new set of column names and corresponding correlation conditions. This might require columns to be deleted or new ones to be added. The learning agent can also change the correlation conditions, such as time and location requirements in the aviation domain. The agent may also define a new column to be the classifier, thus starting a new thread.

8. Conclusion

In this paper, there is a description of an information agent that facilitates a human user in developing and executing KDD routines. This work represents a novel investigation of agent capabilities for use in knowledge discovery. This information agent has been useful as applied to the aviation domain and currently in use for analysis purposes at The MITRE Corporation. There is also a description of the on-going work an architecture that is more towards "automated" KDD routines. The use of the initial information agent was instrumental in determining the initial design and protocols for the multiple agent architecture. Future work is toward the evaluation of the multiple agent architecture.

9. Acknowledgements

Our thanks to several undergraduate students, Wendell Norman, Lindsay Blohm, Todd Cornett, and Denitsa Apostolava, who helped develop and test the agent modules. We thank Professor Marcus Maloof for his support in the use of the WEKA software and machine learning algorithms. We thank Professor Lisa Singh for discussions and some directions toward knowledge discovery in databases.

REFERENCES

[1] Agrawal, R. and Shim, K. "Developing Tightly-Coupled Data Mining Applications on a Relational Database System", Proceedings of the 2nd Int'l Conference on Knowledge Discovery in Databases and Data Mining, Portland, Oregon, August, 1996

[2] Blake, M.B. "Rule-Driven Coordination Agents: A Self Configurable Agent Architecture", 5th IEEE International Symposium of Autonomous Decentralized Systems (ISADS2001), pp 271-278, Dallas, TX March 2001

[3] Blake, M.B., "Using Agent Control and Communication in a Distributed Workflow Information System", Cooperative Information Systems, Lecture Notes in Computer Science, Springer-Verlag, 2519, pp 163-178

[4] Bose, R. and V. Sugumaran, " IDM: An Intelligent Software Agent Based Data Mining Environment". In Proceedings of the 1998 IEEE International Conference on Systems, Man, and Cybernetics, 2888-2893 San Diego, CA: IEEE Press

[5] C5.0: RuleQuest Research (2002) : http://www.rulequest.com/

[6] Clementine (2002): http://www.spss.com/spssbi/clementine/

[7] De Armon, J. et al, "Assessing NAS Performance: Normalizing for the Effects of Weather", 4th USA/Europe Air Traffic Management R&D Symposium, Sante Fe, Dec 3-7. http://atm2001.eurocontrol.fr/finalpapers/pap94.pdf

[8] Fayyad, U., Piatetsky-Shapiro, G., and Smyth,P. (1996) "Knowledge Discovery and Data Mining: Towards a Unifying Framework", *Proceedings of the Second International Conference on Knowledge Discovery and Data Mining (KDD-96)*, Portland, Oregon, AAAI Press

[9] Helmer, G. G., J.S.K. Wong, V. Honavar, and L. Miller, " Intelligent Agents for Intrusion Detection" Proceedings. IEEE Information Technology Conference, 121-124, Syracuse, NY: IEEE Press

[10] Jennings, N., K. Sycara., and M. Wooldridge, "A Roadmap to Agent Research and Development". Journal of Autonomous Agents and Multi-Agent Systems 1 (1) 7-38

[11] Kimball, R. The Data warehouse Toolkit: Practical Techniques to Building Dimension Data Warehouses, New York: John Wiley. 1996

[12] Maes, P. 1997. Agents that Reduce Work and Information Overload. Software Agents: AAAI Press/MIT Press

[13] National Weather Service METAR/TAF Information (2002): http://205.156.54.206/oso/oso1/oso12/faq.htm

[14] Nazeri, Z. and Jianping Zhang, "Mining Aviation Data to Understand the Impacts of Severe Weather on Airspace System Performance" Proceedings of the International Conference on Information Technology: Coding and Computing(ITCC'02)/ IEEE Press 2002

[15] Netz, A., S. Chaudhuri, U. Fayyad, and J. Bernhardt, "Integrating Data Mining with SQL Databases: OLE DB for Data Mining" Proceedings of the International Conference on Data Engineering (ICDE 2001), Heidelberg, Germany 2001

[16] Nodine, M., J. Fowler, T. Ksiezyk, B. Perry, M. Taylor, and A. Unruh, "Active Information Gathering in InfoSleuth". International Journal of Cooperative Information Systems 9:1/2, 3-28. 1998

[17] Payne, T.R. and P. Edwards "Interface Agents that Learn: An Investigation of Learning Issues in a Mail Agent Interface". Applied Artificial Intelligence, Vol. 11(1), pp 1-32.1997

[18] Sarawagi, S., Thomas, S., and Agrawal, R. "Integrating Association Rule Mining with Databases: Alternatives and Implications", *Data Mining and Knowledge Discovery Journal*, 4(2/3), July 2000

[19] Software Suites supporting Knowledge Discovery (2002): http://www.kdnuggets.com/software/suites.html

[20] The Aviation System Performance Metrics (ASPM) (2002): http://www.apo.data.faa.gov/faamatsall.HTM

[21] The National Convective Weather Forecast (NCWF) (2002): http://cdm.awc-kc.noaa.gov/ncwf/index.html

[22] WEKA (2002) http://www.cs.waikato.ac.nz/~ml/weka/