December 2003

Using Agents and XML for Knowledge Representation and Exchange: An Intelligent Distributed Decision Support Architecture (IDDSA)

Rahul Singh
University of North Carolina, Greensboro

A. Salam
University of North Carolina, Greensboro

Lakshmi Iyer
University of North Carolina, Greensboro

USING AGENTS AND XML FOR KNOWLEDGE REPRESENTATION AND EXCHANGE: AN INTELLIGENT DISTRIBUTED DECISION SUPPORT ARCHITECTURE (IDDSA)

Rahul Singh
University of North Carolina, Greensboro
rahul@uncg.edu

Al Salam
University of North Carolina, Greensboro
amsalam@uncg.edu

Lakshmi S. Iyer
University of North Carolina, Greensboro
lsiyer@uncg.edu

Abstract

This paper presents an Intelligent Distributed Decision Support Architecture (IDDSA) for knowledge management by Intelligent Agents. This research contributes to existing knowledge through the development of a methodology to represent modular, rule-based knowledge using XML Schema and decision trees and to share this knowledge through intelligent agent architectures. We show the application of this methodology to create problem-specific knowledge modules that are easily distributed over the Internet and support the design of distributed IDSS. Such an approach will further the research in intelligent decision support by providing the required knowledge representation and decision analytical support on distributed platforms such as the internet. We present directions for future research that utilize the benefits of this research through a conceptual framework for agent-enabled IDSS to manage the generation and use of decision models that support decision specific tasks on distributed platforms.

Keywords: Intelligent agents, multi-agent systems, knowledge representation, intelligent decision support, eXtensible Markup Language

Introduction

Recent advances in systems support for problem solving and decision-making tasks have seen increased use of artificial intelligence (AI) based techniques for knowledge representation (Whinston, 1997). Knowledge representation utilizes multiple techniques including the incorporation of business rules, decision analytical models and models generated from the application of machine learning algorithms through data mining techniques. Intelligent decision support systems (IDSS) incorporate intelligence in the form of knowledge about the problem domain, with problem representation to inform the decision process and reduce the cognitive load of the decision maker.

Enterprise-wide and cross-enterprise DSS are increasingly important as organizations attempt to fully exploit their data repositories and data warehouses to enable collaborative business processes with suppliers and customer organizations over the Internet. Such advances create the need for DSS to focus on providing support to problem solving activities on distributed platforms by providing problem specific data and knowledge to a decision maker anywhere on the Intranet. Such support, while still in its infancy, is of significant importance as organizations increasingly rely on the Internet as the distributed platform to support various information and knowledge centric processes that extend beyond traditional transaction processing.

In this paper, we present an Intelligent Distributed Decision Support Architecture (IDDSA) that provides knowledge representation and knowledge exchange among intelligent agents and knowledge storage in a knowledge repository. This provides the basis for the agent-based architecture to support knowledge management, including knowledge creation, knowledge representation, knowledge exchange and the use of knowledge for decision support on a distributed platform. Our research incorporates the use of XML to represent decision trees and other rule-based knowledge, thereby creating the foundation for agent knowledge representation and exchange. Additionally, we propose the use of a knowledge repository to store knowledge, captured in XML documents, to be used and shared by software agents within the multi-agent architecture. The proposed IDDSA incorporates the recently adopted Web Services architecture whereby agents can provide distributed intelligent decision support by exchanging their knowledge. Details of the implementation of the larger architecture are presented to illustrate the initial proof of concept. Implications and directions for further research in this area by academics and practitioners are discussed.

Relevant Literature Review in DSS, IDSS, XML

Models of decision problems provide analytical support to the decision maker by facilitating a greater understanding of the problem domain and allowing the decision maker to assess the utility of alternative decision paths with respect to achieving the objective of the decision task. Decision Trees are a popular modeling technique with wide applicability to a variety of business problems. The performance of a particular method in modeling human decisions is dependent on the conformance of the method with the decision makers' mental model of the decision problem (Kim et. al., 1997). Decision Trees represent a natural choice for IDSS whose goal is to generate decision paths that are easy to understand, to explain and to convert to natural language. The choice of decision trees as the modeling methodology affords the ability to incorporate inductive learning in the IDSS. Decision trees are among the most commonly used inductive learning techniques used to learn patterns from data. The ID3, C4.5 and SEE5 algorithms provide a formal method to create and model decision rules from categorical and continuous data. Sung, et. al. (1999), compared multiple machine learning techniques in predicting bankruptcies and found that decision tree technique had the most interpretive power. Additionally, decision trees solutions lend themselves to automatic generation of structured queries to extract pertinent data from organizational data repositories (Adriaans and Zantinge, 1996). This makes them particularly useful to provide insights and explanations for the non-technical user (Apte, 1997). Decision trees are especially suitable for decision problems that require the generation of human understandable decision rules based on a mix of classification of categorical and continuous data (Sung et. al., 1999. They provide clear indication of the importance of individual data fields to the decision problem and are therefore useful in reducing the cognitive burden of the decision maker. It is clear that decision trees represent a powerful and understandable technique to model business decisions that can be reduced to a rule-based form. The benefits of the technique highlighted above provide a strong basis for choosing decision trees as a component for intelligent decision support systems.

Decision support systems research continues to look for ways to incorporate modular, reusable knowledge components in the design of Intelligent DSS (IDSS) to assist decision makers located anywhere on a distributed platform such as the Internet. As organizations continue to use distributed computing platforms and architectures to enable business processes, in general, and the use of the Internet to provide information to clients and decision makers, in particular, the need to provide decision support on such distributed platforms will continue to grow (Bui and Lee, 1999). In this respect, decision support systems must be capable of providing the decision maker with decision specific models and data to help analyze and solve decision problems in a distributed manner. Such a need and an approach to achieve this are investigated in Bhargava, et. al. (1997). The recent volume of work in model management and knowledge sharing over the Internet platform (Kim, 2001; Cheng, 2000; Liu, 2001; and Chung and Kim, 2002) uses XML as the enabling technology to provide an open platform for the exchange of knowledge over the Internet.

Intelligent Agents

An intelligent agent is "a computer system situated in some environment and that is capable of flexible autonomous action in this environment in order to meet its design objectives. (Jennings and Wooldridge 1998) The terms agents, software agents and intelligent agents are often used interchangeably in the literature. All agents do not necessarily have to be intelligent. Jennings and Wooldridge (1998) observe that agent-based systems are not necessarily intelligent and require that an agent be flexible to be considered intelligent. Such flexibility in intelligent agent based systems requires that the agents should be (Jennings and Wooldridge, 1998; Bradshaw, J.M., 1997):

- cognizant of their environment and be responsive to changes therein;
- reactive and proactive to opportunities in their environment;
- autonomous in goal-directed behavior;
- collaborative in their ability to interact with other agents in exhibiting the goal-oriented behavior; and
- adaptive in their ability to learn with experience.

Agent based systems may consist of a single agent engaged in autonomous goal-oriented behavior, or multiple agents that work together to exhibit granular as well as overall goal directed behavior. The general multi-agent system is one in which the interoperation of separately developed and self-interested agents provide a service beyond the capability of any single agent model. Such mutli-agent systems provide a powerful abstraction that can be used to model systems where multiple entities, exhibiting self directed behaviors must coexist in a environment and achieve the system wide objective of the environment.

In our current state of the art, the agent paradigm takes the form of specialized objects (Shoham, 1997). This representational form provides the basis for the business and development communities to model agency in emergent information systems. Intelligent Agents are action-oriented abstractions in electronic systems, entrusted to carry out various generic and specific goal-oriented actions on behalf of users. The agent abstraction manifests itself in the system as a representation of the user and performs necessary tasks on behalf of the user. This role may involve taking directions from the user on a need basis and advising and informing the user of alternatives and consequences (Whinston, 1997). The agent paradigm can support a range of decision making activity including information retrieval, generation of alternatives, preference order ranking of options and alternatives and supporting analysis of the alternative-goal relationships. In this respect, intelligent agents have come a long way from being digital scourers and static filters of information to active partners in information processing tasks. Such a shift has significant design implications on the abstractions used to model information systems, objects or agents, and on the architecture of information resources that are available to entities involved in the electronic system. Another implication is that knowledge must be available in formats that are conducive to its representation and manipulation by software applications, including software agents. The previous section addresses this issue through the use of XML and web services to allow for the representation and exchange of knowledge and its exchange between software applications.

This research builds on existing bodies of knowledge in intelligent agents, knowledge management, XML and web services standards. Our research focuses on the translation between XML and Decision Trees by software agents, thereby creating the foundation for knowledge representation and exchange by intelligent agents that support decision makers. Additionally, we propose the use of a knowledge repository to store knowledge, captured in XML documents, to be used and shared by software agents within the multi-agent architecture. In this paper, we present an architecture that provides knowledge representation and knowledge exchange among agents and knowledge storage in a knowledge repository (See Figure 3). The proposed IDDSA incorporates the recently adopted Web Services architecture whereby agents can provide distributed intelligent decision support by exchanging their knowledge using XML and related set of standards. Details of the implementation of the larger architecture are presented to illustrate the initial proof of concept. Implications and directions for further research in this area by academics and practitioners are discussed.

The following section provides details of the agents used in the IDDSA and of the agent hierarchies and agent types used in the design of the decision support architecture. We also describe in detail the flow of knowledge between agents and the flow of information between agent and users and present an overall architecture for IDDSA. Section 4 illustrates the prototype of the architecture through the use of an example decision problem of whether to play tennis given certain climate variables. This example is adapted from Mitchell (1997) and is used to illustrate the use of decision trees and their representation using XML and DTDs that can be used by intelligent agents to help users make decisions. We present the implementation of various components of the architecture, including the agent classes and the representation of the knowledge as XML documents and DTDs.

XML for Knowledge Management

Despite its efficiency for presenting information in human readable format, HTML is very limited in extensibility and customization of markup tags and description of the data contained in those tags. This is a severe constraint that limits the use of HTML by application software for information sharing in a distributed computing environment where application programs (including intelligent agents) are expected to work with available data (and rules and knowledge) without human intervention.

The use of eXtensible Markup Language (XML) and the related set of standards developed by the W3C (http://www.w3c.org), have helped to overcome these limitations. XML allows for the creation of custom tags that contain data from specific domains. For example, a company in the furniture industry may develop customized tags for the representation of content to serve its business domain. By creating custom tags, the company can represent the data in a more meaningful and flexible way than it would be ever possible using HTML. The company may also develop documents that represent business-rules using XML. This description of structures in XML documents is provided by a set of standards called XML Schema and the Document Type Definition (DTD) language (http://www.w3c.org). The XML schema document describes the specific elements (the XML tags), their relationships and the specific types of data that can be stored in each of these tags. Essentially, the XML schema documents describe the structure of XML documents and their contents. XML parsers written in C, C++ or Java can process and validate XML documents (containing business rules and data) based on XML schemas written based on either the DTD or the XML Schema specification. Application software appropriate parser utilities are able to read and/or write to XML documents following the W3C standards and specification. This provides the foundation technology, built upon an agreed and accepted standard from W3C, for the capture, representation, exchange and storage of knowledge, represented by business rules and related data in XML format, that can be potentially used and shared by software agents.

Recently, there have been some interesting and promising initiatives to develop technologies for what is being called the "Semantic Web" (Berners-Lee, Hendler and Lassila, 2001) The purpose of these initiatives is to make the content of the web unambiguously computer-interpretable, thus making it amenable to agent interoperability and automatic reasoning techniques (McIlraith et al. 2001). Two important technologies for developing Semantic Web are already in place- XML and the Resource Description Framework (RDF). The W3C developed the RDF as a standard for metadata to add a formal semantics to the Web, defined on top of XML, to provide a data model and syntax convention for representing the semantics of data in standardized interoperable manner (McIraith, Son and Zeng, 2001). The RDF working group also developed RDF Schema (RDFS), an object-oriented type system that can be effectively thought of as a minimal ontology modeling language. Recently, there have been several efforts to build on RDF and RDFS with more AI-inspired knowledge representation languages such as SHOE, DAML-ONT, OIL and DAML+OIL (Fensel, 2000). Even though these initiatives are extremely promising for agent interoperability and reasoning, they are at their early stages of development

Intelligent Distributed Decision Support Architecture (IDDSA)

IDDSA comprises intelligent software agents as the basic abstraction used for knowledge representation and knowledge exchange in supporting decision making activities. The agent abstraction is created using objects as the base class and incorporates additional features as warranted by agent functionality. Each agent class contains a decision tree component to represent domain knowledge. Knowledge agents also interact with a knowledge repository to actively affirm the accuracy of the decision tree models used by the knowledge agents and serve as the active component of the decision support system architecture. Figure 1 below shows a class diagram depicting the relationship between agents in use by IDDSA.

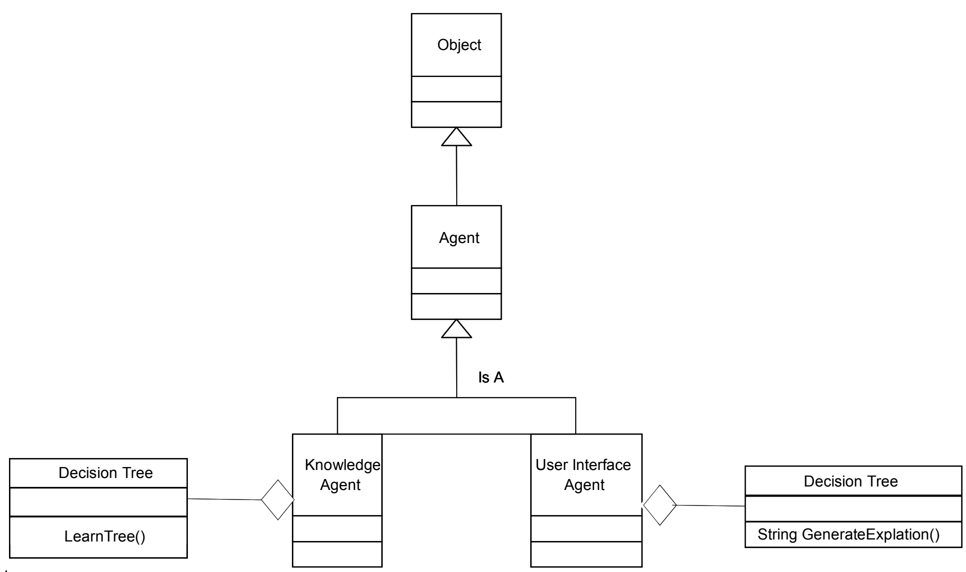


Figure 1. Class Diagram Showing the Intelligent Agents Classes and Decision Tree Classes Involved in IDDSA

The two types of agents employed by IDDSA include a Knowledge Agent and a User Interface agent. In addition, each agent comprises a decision tree component which is the primary knowledge representation mechanism. The Knowledge Agent has methods to learn its representation of the problem domain and does this on an active basis. The primary function of this agent is to maintain current representations of knowledge of the problem domain. It communicates with the User Interface agents through exchange of the decision models using XML and DTDs. The purpose of the User Interface Agents is to interact with the user and provide decision support, provide feedback for decisions and provide explanation of choices made through the reasoning capabilities afforded to it through explanations generated from its decision tree component. The User Interface maintains current knowledge of the problem domain by synchronizing its knowledge with the knowledge agent. Hence, the decision support facility provided to the user is active in its maintenance of the most current knowledge available. Figure 2 illustrates the structure of the User Interface and Knowledge Agents.

Each agent contains members that represent the contents and structure of the XML documents to distribute knowledge to other agents and to the user. In the case of the user interface agent, the agent contains methods to take parameters of the decision problem and provide textual explanations of the decision. This is done by parsing the learned decision tree and generating the explanations in a format that is easy for the user to understand. The User Agent contains a learned decision tree that represents the current state of knowledge that the system has about the problem domain. The tree can be used to generate decision rules and explanations to support decision making by the user. The Knowledge Agent contains methods to generate rules to support ad-hoc queries by the user agent. This is supported through the interactions of the Knowledge Agent with the Knowledge Repository of the system which is implemented as a set of XML documents that can be stored in a repository which is capable of storing XML documents such as the Oracle 9i family of information management products.

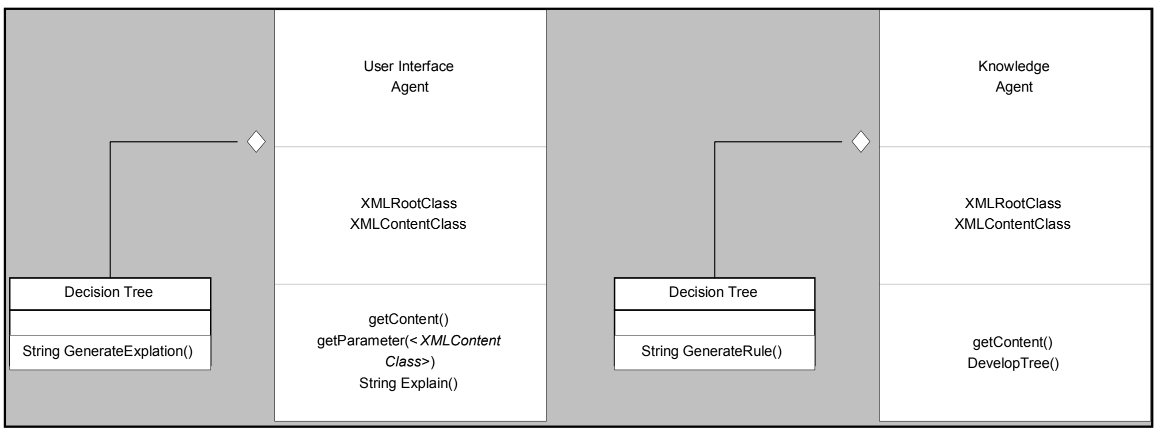


Figure 2. Class Diagram Showing the User Interface and Knowledge Agent Details

These agents, with their knowledge representation capability, using decision trees, and their knowledge exchange capabilities, using XML objects, form the building blocks for the Intelligent Distributed Decision Support Architecture discussed in this paper. The flow of information and the architecture are presented in figure 3 below. The user interacts with the User Interface agent by asking question about the decision problem and receives responses containing decision alternatives and explanation of the choices made by the agent. This is achieved through parsing the decision tree based on the parameters supplied by the user. The User Interface agents interact with the Knowledge agents in order to receive information from them regarding any updates that are required to the knowledge representation that is presented to the user. The knowledge agents also serve as the means to service any ad-hoc queries that cannot be answered by the user interface agents, such as queries regarding knowledge parameters that are not available to the user interface agents. In such cases, the Knowledge agent, with direct access to the knowledge repository can provide such knowledge to the user agents, for the benefit of the user. In this respect, the interactions among the agents in this system are modeled as collaborative interactions, where the agents in the community work together to provide decision support and knowledge-based explanations of the decision problem domain to the user.

The IDDSA consists of intelligent agents as discussed above that are able to provide intelligent decision support to the end-users. All of the agents in the architecture are fully FIPA compliant in terms of their requirements and behavior. In this architecture, decision trees are used for representing rule-based modular knowledge that can be learned, used and shared by the agents in the agent community. The knowledge agents primarily learn and acquire this knowledge from the repository, while the user agents help the users make decisions on specific problems using the knowledge contained in the decision trees. The exchange of knowledge between agents and between users and agents is achieved through sharing of content information using XML. The agents work on a distributed platform and enable the transfer of knowledge by exposing their public methods as Web Services using SOAP and XML.

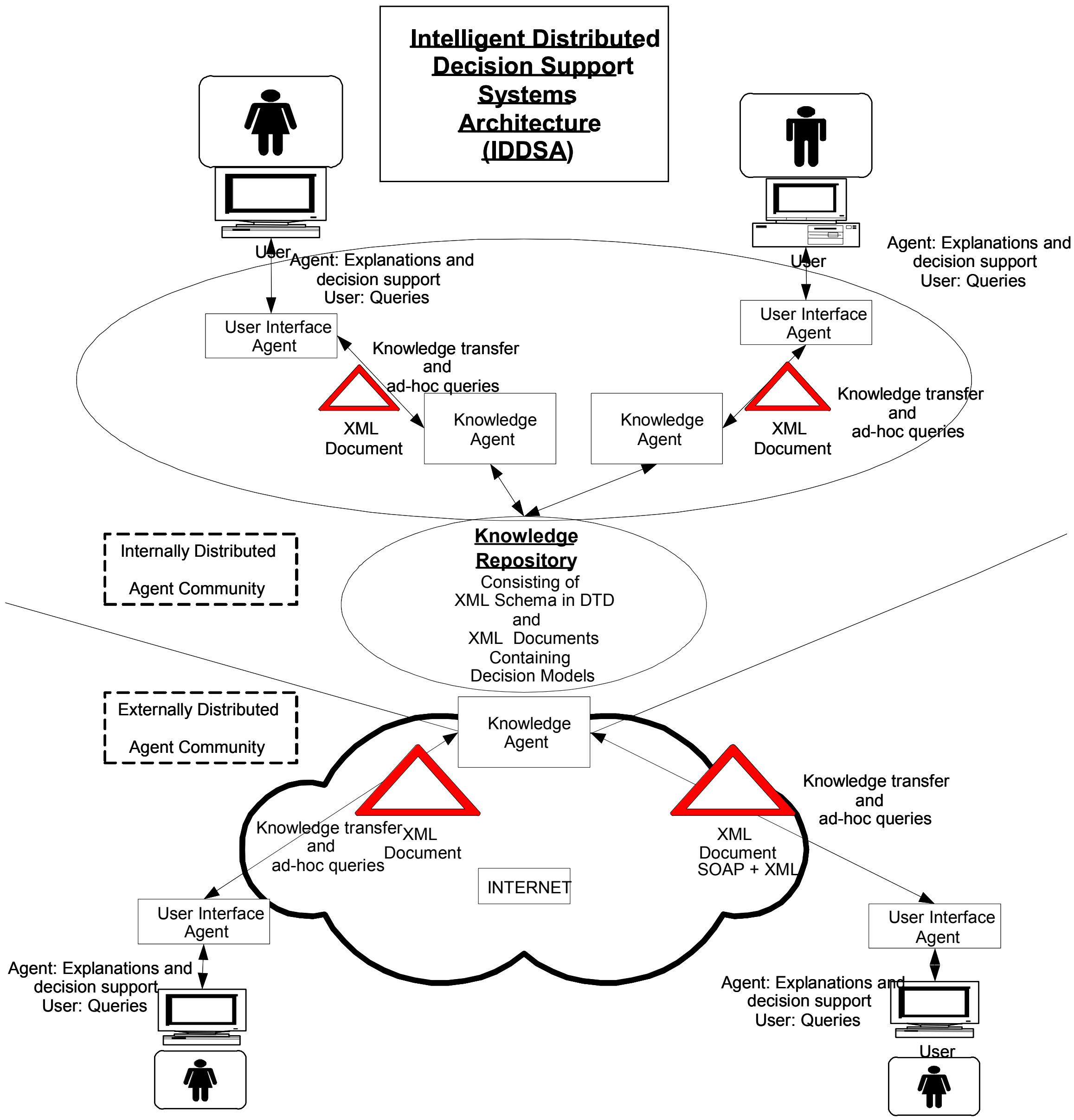


Figure 3. Intelligent Distributed Decision Support Systems Architecture (IDDSA)

The rule-based modular knowledge in the form of decision tree is captured in XML documents that then can be used and shared by agents. Capturing the modular knowledge in XML format also facilitates their storage in a knowledge repository- a repository that enables storage and retrieval of XML documents. The architecture allows for multiple such knowledge repositories depending upon the problem domain. The benefits of such knowledge repositories are the historical capture of knowledge modules that are then shared among agents in the agent community. This minimizes the learning curve of newly instantiated agents who are instantiated with the current knowledge that is available to the entire system. This is achieved in IDDSA since agents have captured rule-based knowledge modules and have stored such knowledge modules in XML format in the knowledge repository for the benefit of the entire agent community and the system.

The proposed IDDSA architecture also provides a decision explanation facility to the end-users where agents are able to explain how they arrived at a particular decision. This has three important benefits:

(1) The end-user is able to understand how the decision was made by the software agent,

(2) the end-user is able to make a clear assessment of the viability of the decision and
(3) the end-user is able to learn and gain more knowledge about the problem domain by studying the decision paths used by the agent.

Agents are able to explain the rules and parameters that were used by the agent in arriving at the stated decision. This explanation facility is a natural extension of using decision trees in general for solving rule-based decision problems. Non-technical end-users are able to easily understand how a problem was solved using decision trees compared to other existing problem-solving methods such as neural networks, statistical and fuzzy logic-based systems (Sung, 1999).

The IDDSA architecture is divided into two main parts: one focusing on intelligent distributed decision support that may be internal to the company and the other focusing on providing intelligent distributed decision support that may be external to the company. In the second case, the proposed architecture incorporates the W3C Web Services architecture that uses the simple object access protocol (SOAP) and XML. The incorporation of this architecture creates a flexible means of exposing the services of the agents using the Web Services architecture by a company to its potential or existing global population of customers and suppliers.

Play Tennis: An Illustrative Example

The problem domain selected for the initial proof of concept is the play tennis decision problem (Mitchell, 1997) using the decision tree method. The decision problem is to decide whether to play tennis on a particular day or not based upon climatic conditions such as the day's outlook, the level of humidity and the wind conditions. Figure 4 shows a schematic of the decision problem under consideration. The leaf nodes of the decision tree represent the final outcome of the decision of whether to play tennis on a certain day, based on what the weather is like.

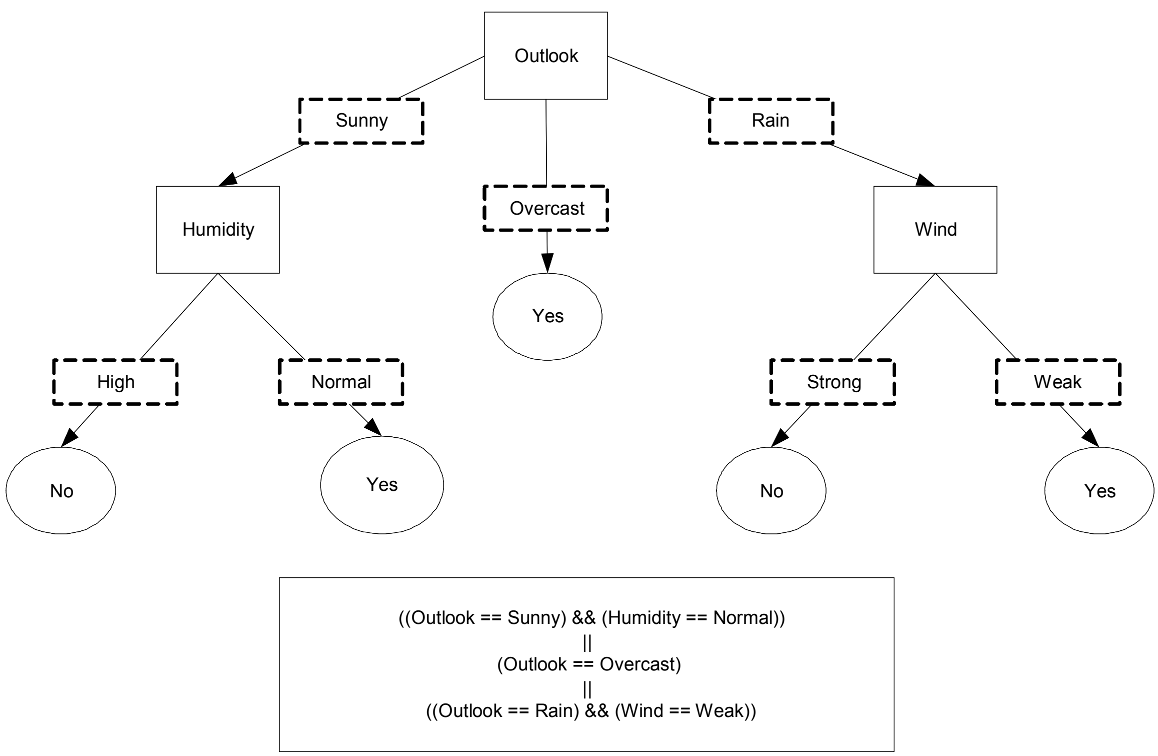
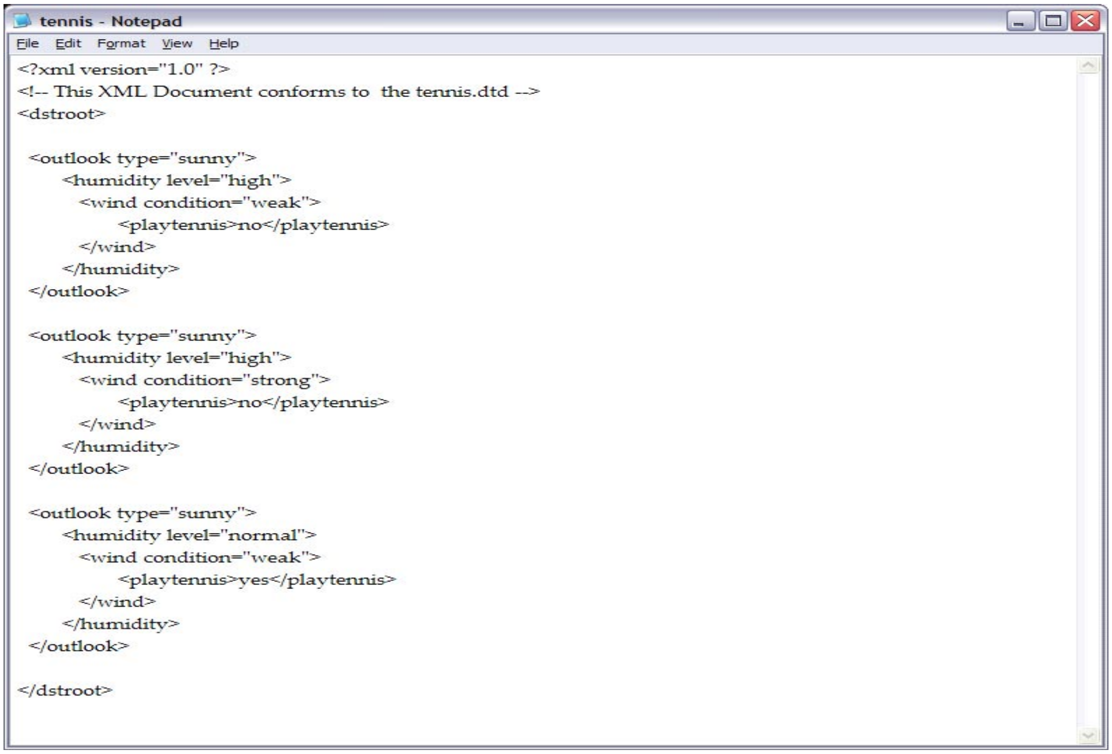


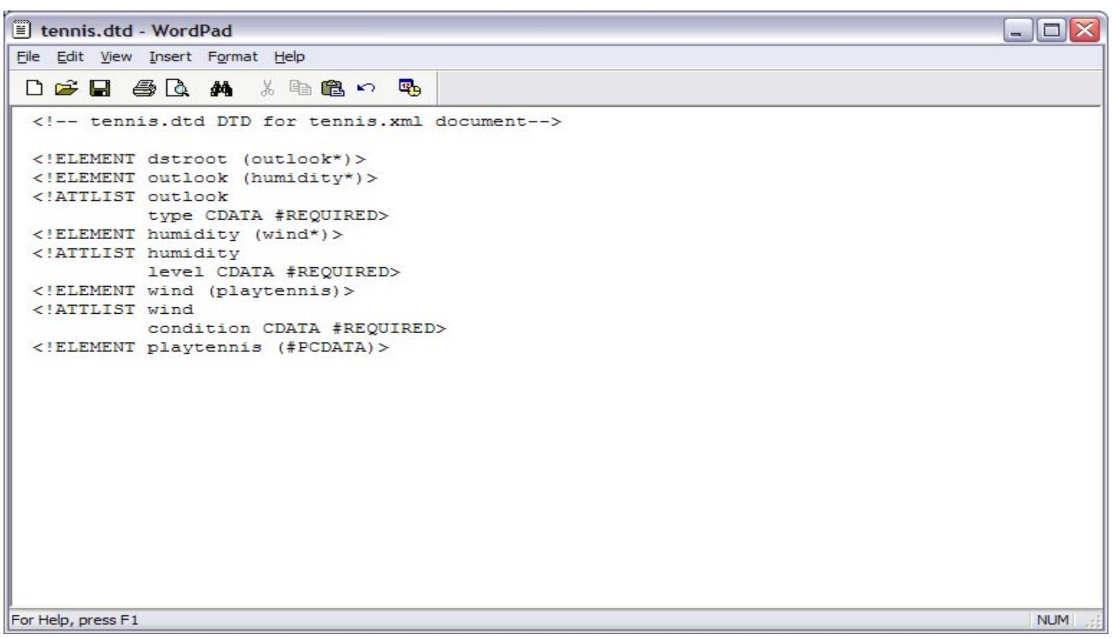
Figure 4. Decision Tree Representation of the Play Tennis Problem (adapted from Mitchell, 1997)

As an input the end-user provides the existing condition to the software agent and the agent makes a decision and presents the decision to the end-user whether or not tennis can be played that particular day given the conditions entered by the user. The problem is simple to understand, yet it illustrates the fundamental requirements of the system and provides an elegant way to test the various features of the agents and the architecture.




```
<?xml version="1.0" ?>
<!-- This XML Document conforms to  the tennis.dtd -->
<dstroot>

 <outlook type="sunny">
    <humidity level="high">
      <wind condition="weak">
          <playtennis>no</playtennis>
      </wind>
    </humidity>
 </outlook>

 <outlook type="sunny">
    <humidity level="high">
      <wind condition="strong">
          <playtennis>no</playtennis>
      </wind>
    </humidity>
 </outlook>

 <outlook type="sunny">
    <humidity level="normal">
      <wind condition="weak">
          <playtennis>yes</playtennis>
      </wind>
    </humidity>
 </outlook>

</dstroot>
```

Figure 5. Decision Tree Representation of the Rule-Based Knowledge Module for the Play Tennis Problem




```
<!-- tennis.dtd DTD for tennis.xml document-->

<!ELEMENT dstroot (outlook*)>
<!ELEMENT outlook (humidity*)>
<!ATTLIST outlook
          type CDATA #REQUIRED>
<!ELEMENT humidity (wind*)>
<!ATTLIST humidity
          level CDATA #REQUIRED>
<!ELEMENT wind (playtennis)>
<!ATTLIST wind
          condition CDATA #REQUIRED>
<!ELEMENT playtennis (#PCDATA)>
```

Figure 6. Document Type Definition (DTD) for the Play Tennis Problem

In the prototype implementation of the proposed IDDSA architecture, we use the Java programming language (J2EE - JDK 1.4 – http://java.sun.com) to implement the agents as extensions of objects. The choice of the Java programming language was based upon the widely accepted advantage of Java providing portable code and XML providing portable data. In addition, we use the Oracle 9i Database and Application Server platforms (http://www.oracle.com) to implement the knowledge repository and use the Sun Microsystems Java XML API toolkit to interface the agents with the XML repository. The decision tree implementation consists of tree nodes with branches for every category of the node variable. Each traversal from the root node of the decision tree to a leaf node leads to a separate decision path as illustrated in Figure 4. The agents contain methods to traverse the decision tree and obtain a decision path which can then be translated into an XML object and an XML document using a Document Type Definition (DTD) file. These form the basis for the generation of decision alternatives and for the explanations of decisions by the agents. The agents are implemented as java beans and their explanations are available to the user through calls made to their public methods that are exposed as services, and presented to the user as dynamically generated web content by using JSP (Java Server Pages) technology (http://java.sun.com/products/jsp/index.html) Figure 5 presents an XML document that contains the rule-based knowledge module in the form of a decision tree representing the rules of the play tennis problem. Figure 6 shows the DTD document that standardizes the creation and validation of XML documents pertaining to this decision problem.

Conclusion

This research presents a methodology to represent modular, rule-based knowledge using the Extensible Markup Language (XML) and the Document Type Definition (DTD) standards from the World Wide Web Consortium (W3C). Using this methodology, we have shown how such an approach can be used to create problem-specific knowledge modules that can easily be distributed over the Internet to support distributed IDSS design. Such an approach will facilitate intelligent decision support by providing the required knowledge representation and the decision analytical support. We show the conceptual architecture of such a distributed IDSS and provide details of the components of the architecture, including the agents involved and their interactions, the details of the knowledge representation and implementation of knowledge exchange through a distributed interface. We also provide indication of how such architecture might be used to support the user and assume the role of an expert and provide explanations to the user, while retaining the benefits of an active DSS through extensible knowledge generation by incorporating machine learning algorithms.

The limitations inherent to decision trees and such techniques are also the limitation of this architecture. Therefore, further research needs to be conducted to understand how this architecture can be expanded to incorporate other types of learning and rule induction or rule creation to be shared and used by software agents. Despite this limitation, this research contributes significantly to the advancement of our understanding of how emerging technologies can be incorporated into intelligent agent-based architecture to enhance the value of such systems in distributed intelligent decision support systems. The play tennis example used here is simple, intuitive and elegantly achieves its purpose of illustrating the use of the architecture while minimizing complications inherent to a more complex problem domain. We are conducting further research on elaborating this architecture for a variety of problems that lend themselves to rule-based, inductive decision making with a need for user interactions and which benefit from greater understanding of the problem domain by the user.

References

Adriaans, P., and Zantinge, D., *Data Mining*. Harlow, UK: Addison-Wesley, 1996.

Berners-Lee, Tim, Hendler, James and Lassila, Ora, "The sematic Web" *Scientific American*, May 2001, pp. 34-43

Bhargava, H. K., R. Krishnan, and R. Muller, "Decision Support on Demand: Emerging Electronic Markets for Decision Technologies," *Decision Support Systems*, Vol. 19, pp. 193-214, 1997.

Bradshaw, J. M. (Ed.) (1997), *Software Agent*s, Boston: MIT Press.

Bui, Tung and Lee, Jintae An agent-based framework for building decision support systems, *Decision Support Systems,* Volume 25, Issue 3, Pages 225-237, April 1999.

Cheng, E. C., An object-oriented organizational model to support dynamic role based access control in electronic commerce, *Decision Support Systems,* Vol. 29. pp. 257-369, 2000

Chung, Tae-Sun, and Kim, Hyoung-Joo, Extracting indexing information from XML DTDs, *Information Processing Letters*, Vol. 81, pp. 97-103, 2002

Dolk, D.R., Model management and structured modeling: the role of an information resource dictionary system, *Communications of the ACM*, v.31 n.6, p.704-718, June 1988.

Fensel, D. , *IEEE Intelligent Systems*, Novmber/Decmber 2000 page 67

Jennings, N. R. & Wooldridge, M., *Agent Technology: Foundations, Applications, and Markets*, London: Springer. 1998.

Kim, C. N., Chung, H. M., and Paradice, D. B. Inductive Modeling of expert decision making in loan evaluation: a decision strategy perspective. *Decision Support Systems*, 21, 2, 83-98. 1997

Kim, H., An XML-based modeling language for the open interchange of decision models, *Decision Support Systems,* Vol. 31. pp. 429-441, 2001

Krishnan, R. and K. Chari, Model Management: Survey, Future Research Directions and a Bibliography, *Interactive Transactions of ORMS*, Vol. 3, No. 1, 2000.

Liu, L., Pu, Calton and Han, W., An XML-enabled data extraction toolkit for web sources, *Information Systems*, Vol. 26. pp. 563-583, 2001

McIlraith, Sheila, Son, Tran C. and Zeng, Honglei, Mobilizing the Semantic Web with DAML-Enabled Web Services, *Proceedings of the Semantic Web Workshop*, Hong Kong, China, 2001

Mitchell, T. M., *Machine Learning*. McGraw-Hill, 1997.

Shoham, Y. An Overview of Agent-Oriented Programming. In *Software Agents,* Bradshaw, J. M. (Editor). Menlo Park, CA. AAAI Press. 1997

Tsai, Yao-Chuan, Model Integration using SML, *Decision Support Systems,* Vol. 22. pp. 355-377, 1998.

Whinston, A., Intelligent Agents as a basis for Decision Support Systems, *Decision Support Systems,* Vol. 20, No. 1, 1997.