International Journal of Information Technology and Management

Vol. VIII, Issue No. XII, May-2015, ISSN 2249-4510

A STUDY ON DECISION SUPPORT SYSTEMS AND AUTONOMOUS AGENTS

AN INTERNATIONALLY INDEXED PEER REVIEWED & REFEREED JOURNAL

www.ignited.in

International Journal of Information Technology and Management
Vol. VIII, Issue No. XII, May-2015, ISSN 2249-4510

A Study on Decision Support Systems and Autonomous Agents

Umesh Kumar Lilhore[1] Prof. Dr. Rishi Pal Bangarh[2]

[1]Research Scholar, Maharishi University of Information Technology, MUIT (Lucknow)

Abstract – Software agents have been heralded as the most important emerging technology of the decade. As software development firms eagerly attempt to integrate these autonomous programs into their products, researchers attempt to define the concept of agency and to develop architectures that will improve agent capabilities. Decision Support System (DSS) researchers have been eager to integrate agents into their applications and exploratory works in which agents have been used within a DSS have been documented. Decision Support System (DSS) researchers have been quick to experiment with software agents.

----------X----------

INTRODUCTION

A Decision Support System is a computer-based information system that supports business or organizational decision-making activities. DSSs serve the management, operations, and planning levels of an organization (usually mid and higher management) and help people make decisions about problems that may be rapidly changing and not easily specified in advance - i.e. Unstructured and Semi-Structured decision problems. Decision support systems can be either fully computerized, human-powered or a combination of both.

While academics have perceived DSS as a tool to support decision making process, DSS users see DSS as a tool to facilitate organizational processes. Some authors have extended the definition of DSS to include any system that might support decision making.[2] Sprague (1980) defines DSS by its characteristics:

1. DSS tends to be aimed at the less well structured, underspecified problem that upper level managers typically face;
2. DSS attempts to combine the use of models or analytic techniques with traditional data access and retrieval functions;
3. DSS specifically focuses on features which make them easy to use by no computer people in an interactive mode; and
4. DSS emphasizes flexibility and adaptability to accommodate changes in the environment and the decision making approach of the user.

DSSs include knowledge-based systems. A properly designed DSS is an interactive software-based system intended to help decision makers compile useful information from a combination of raw data, documents, and personal knowledge, or business models to identify and solve problems and make decisions.

Typical information that a decision support application might gather and present includes:

- Inventories of information assets (including legacy and relational data sources, cubes, data warehouses, and data marts),
- Comparative sales figures between one period and the next,
- Projected revenue figures based on product sales assumptions.

DSSs are often contrasted with more automated decision-making systems known as Decision Management Systems. Implementations of agents within DSS applications have begun to surface in DSS-oriented journals. Given the overall lack of agent theories and architectures, DSS researchers have had to proceed in this experimentation with little conceptual guidance on how and when to use software agents. Similarly, there has been no detailed discussion of why this emerging technology is appropriate or useful for DSS.

REVIEW OF LITERATURE

Software agents were born out of the concept of robots within the artificial intelligence (AI) community.

Instead of a physical entity carrying out manual tasks on behalf of a user, software agents are programs that perform computing tasks. For over twenty years, the AI community has investigated agents in terms of entities that exhibit intelligent behavior. Research in the realm of Distributed Artificial Intelligence (DAI) has investigated concurrent and distributed intelligent processing and provided a framework for the agent systems being implemented today. Within DAI, two distinct streams of research have provided different perspectives on distributed intelligent agents. The Distributed Problem Solving (DPS) stream focused on problem resolution through *task decomposition* and delegation to distributed nodes or agents (Bond & Gasser, 2008).

DPS emphasized problem solving and did not focus on the composition or individual behavior of the actual nodes. The second stream, Multi-agent Systems (MAS), viewed agents as autonomous entities and focused on how these entities could *coordinate* to solve problems (Bond & Gasser, 2008).

These differing views on the concept of software agents, along with others, are present in a newer stream of software agent research that has developed in the 2000's. This new stream of research is multi-disciplinary and has shifted the focus from how intelligent software entities could interact to solve problems to creating entities that perform a useful task (Bradshaw, 2007; Nwana, 2006). These entities are not always intelligent, or that useful, but the change in emphasis from analyzing to *implementing* has brought a great deal of attention to the topic of software agents and has renewed interest in AI applications.

SOFTWARE AGENT FEATURES

The abstract nature of software agents and the potential functionality that can be incorporated into one have made it difficult for a single definition of a software agent to be widely accepted (Foner, 2008; Franklin & Graesser, 2006; Kautz, Selman, Coen, Ketchpel, & Ramming, 2004).

Autonomy is a characteristic that appears to be fundamental to most definitions of software agents. The interpretations of autonomy with regard to software agents vary slightly among agent researchers. Foner requires that an agent be able to "pursue an agenda independently from its user" and take "preemptive or independent actions that will eventually benefit the user" (2008, p.1).

Franklin and Graesser have a less restrictive view of autonomy, requiring agents to "exercise control over their own actions" (2006, p.6). Using the less restrictive definition, a software agent could be a program executed initially by the user which would then carry out its purpose independently.

Software agents, as robust autonomous programs, can provide an abstraction for the increasing complexity of computing. Agents can provide this abstraction because they have more stringent requirements (i.e., the essential features) than typical programs and are thus more independent and reliable. A user or developer can delegate a task to an agent and not concern himself or herself with how or whether the agent will accomplish the task. The reactive, persistent nature of the agent should ensure that it pursues its goals zealously and that it updates the user or other programs on the status of its goals. The use of homeostatic goals provides long-term satisfaction of developers' and users' needs.

Agents can serve as an abstraction in two important areas, interoperability and user interfaces (Bradshaw 2007). As an interoperability abstraction, agents are used to integrate between heterogeneous applications. Whether the heterogeneous applications are older legacy programs or distributed applications on different networks, an agent can mediate between two systems.

Researchers in software engineering have successfully implemented several agent systems that integrate heterogeneous applications (Genesereth & Ketchpel 2004; Petrie, 2006).

The use of agents as a user interface abstraction, can provide an alternative means of desktop manipulation. Limitations of the direct manipulation interface include scalability and level of expertise. As the volume of information at our fingertips increases, the hierarchy of files and links on our desktops becomes too deep to negotiate efficiently. As the direct manipulation interface is extended to accommodate the volume of information, frequently the complexity of using the interface is also raised (Maes, 2004).

RESEARCH METHODOLOGY

Software agents constitute an enabling technology that adds new features to or significantly enhances existing features in software applications. As an emerging technology that utilizes various artificial intelligence mechanisms, research on the topic is largely exploratory in nature.

The research described herein is thus also largely exploratory. After synthesizing the literature on the various features that agents can and should exhibit, the current research sets forth a definition of software agents (an agent theory) applicable to the use of agents in DSS. A proof of-concept implementation containing several different types of agents that meet the requirements of the definition is then designed and built. The agents used in this implementation establish a basic infrastructure for employing agents in DSS and are primarily reactive in nature.

The agents respond efficiently to changes in the DSS environment and their deliberative abilities are

fairly minimal. Based upon a review of the relevant literature and the experience gained in building an agent integrated DSS, a general framework for using agents within a DSS is then proposed. The framework includes several different types of agents that would be generally useful in most DSS, regardless of the domain. In order to explore the use of more deliberative agent architectures within DSS, a second agent integrated DSS implementation was designed and built. This DSS featured the same agent framework discussed previously and introduced a planning agent with deliberative abilities. The planning abilities were achieved by embedding a partial-order planner within the agent. Based upon a review of the literature and the experience gained in designing and developing this agent, a general architecture for a planning agent is suggested.

SCOPE OF THE STUDY

The scope of this research includes an examination of agent features and the design and implementation of an agent framework and architecture to support the integration of agents in DSS. Agent usability issues and the selection of analysis and designed methodologies that are appropriate for implementing the architecture are not addressed. Limitations of the research include the domain specificity of the implementations. The two implementations built were designed for specific types of DSS, a variance-analysis DSS and a profit-monitoring DSS. Generalization of the architecture across various problem domains may be inappropriate due to the wide range of differing requirements in each domain. While realistic business problems were selected for the implementations, simplifying assumptions were needed to constrain implementation size. These simplifications could decrease the relevance of implementing a successful, agent framework and architecture as the implementation developed may not accurately represent the problem space.

REFERENCES

- Gupta, Vivek R., An Introduction to Data Warehousing, *System Services Corporation White paper*, 14 April 2008, <http://www.system-services.com/dwintro.htm> (9 July 2008).
- Inmon, W.H., Tech Topic: What is a Data Warehouse?, *Prism Solutions, Inc., White Paper*, 2005, <http://www.cait.wustl.edu/cait/papers/prism/vol1_no1/> (9 July 2008).
- Sen, R. (2007). A Quality Assurance Model for Evaluating Internal Controls and Performance of Data Warehouse Systems. *Working paper*.
- "Using Software Agents to Create the Next Generation of Decision Support Systems," with Loren Paul Rees and Terry R. Rakes, submitted to *Decision Sciences* (revision submitted - March, 2009*)*, March 2008.
- "An Organizational Decision Support System for Managing the DOE Hazardous Waste Cleanup Program," with Tarun K.
- Sen and Laurence J. Moore, submitted to *Decision Support Systems* (revision submitted - December, 2008), April 2008.
- "Planning in Decision Support Systems Using Autonomous Software Agents," with Loren Paul Rees and Terry R. Rakes, submitted to *Journal of Management Information Systems*, April 2009.
- "Using Automated Software Planning Agents to Extend the Model Management System in a Decision Support System," with Terry R. Rakes and Loren Paul Rees, *Proceedings of the Decision Sciences Institute 2008 National Meeting,* 2008, Las Vegas, NV, 663-665.
- "Enhancing Decision Support Systems with Partial-Order Planning and Software Agents"
- *Proceedings of the Americas Conference on Information Systems*, 2008, Baltimore, MA, 1153.
- "Developing Web-Based Decision Support Systems - Visual Basic to the Rescue," with Cliff T. Ragsdale and Loren P. Rees, *Proceedings of the Decision Sciences Institute 2008 National Meeting,* 2008, Las Vegas, NV, 245-246*.*
- "Computing Security: A Case for MOMMA (Mother of Mobile Malicious Agents)" with Loren Paul Rees and Terry R. Rakes, *Proceedings of 34th Annual Meeting*, Southeastern Chapter of InfORMS, October 2008, Myrtle Beach, SC, 247-248.
- "An Implementation of Software Agents in a Decision Support System," *Proceedings of the 33rd Annual Meeting*, Southeastern Chapter of In forms, October 2007, 482-484.
- "Using Scientific Visualization to Improve Financial Decision Making," with Rick Dull, *Proceedings of the Americas Conference on Information Systems*, 2006, Phoenix, Arizona, 758-760.