Multilingual Agents in a Dynamic Environment

Jeffrey Tweedale and Lakhmi Jain

Abstract. Experiments conducted by the Knowledge-Based Intelligent Information and Engineering Systems (KES) Centre use Java to gain its many advantages, especially in a distributed and dynamically scalable environment. Interoperability within and across ubiquitous computing operations has evolved to a level where *plug 'n' play* protocols that invoke common interfaces, provide the flexibility required for effective multi-lingual communications. One example includes: dynamic agent functionality within simulations that automatically adapt to incoming data and/or languages via scripts or messaging to achieve data management and inference. This has been shown using demonstrations at the Centre herein. Many aspects of the model involve web centric transactions, which involve data mining or the use of other types of Intelligent Decision Support System (IDSS). Section One of this paper provides an introduction, Section Two introduces the basic concepts of Decision Support System (DSS), Section Three discusses Intelligent Decision Support System (IDSS) enhancements, Section Four explains how agents use a multi-lingual dynamic environment, while Section Five highlights conclusions and future research direction.

Keywords: Multi-Agent System, Intelligent Decision Support System, Universal Plug and Play, XML Object Model.

1 Introduction

As technology advances, humans are increasingly introducing delays and/or errors through the lack of response within prescribed system limits. Natural language

Jeffrey Tweedale and Lakhmi Jain
School of Electrical and Information Engineering,
Knowledge Based Intelligent Engineering Systems Centre,
University of South Australia, Mawson Lakes, SA 5095, Australia
e-mail: Jeffrey.Tweedale@unisa.edu.au,
Lakhmi.Jain@unisa.edu.au

interfaces are slowly gaining acceptance; however many enhancements are still required before these become reliable enough for widespread use. Researchers also need to maintain the support of operators, therefore human involvement must be integrated to avoid system conflicts and isolate the operators from dirty, dull or dangerous tasks. Conversely, automation assists humans in becoming more productive, increases safety and reduces the cost of manufacturing or equipment operation. This research aims to develop an agent learning and teaming architecture suitable for controlling distributed systems or systems-of-systems as plug-able components in a Service-Oriented Architecture (SOA) using Web Services via common interfaces.

In 1980 Tim Berners-Lee enabled seamless connectivity using the World Wide Web (WWW)[1][1], while Marc Andreesen and Eric Bina coded Mosaic in (1992)[3][2] using HyperText Transfer Protocol (HTTP) to facilitate the presentation of requested information [4]. A series of follow-on developments resulted in a logic based protocol that could be cataloged into libraries and distributed in the format derived for Web-Services Description Language (WSDL) using an Extensible Markup Language (XML) protocol over HTTP. "This rich service provides a powerful mechanism of assembling information resources in context that require the agile construction of virtual organizations [5]". Many applications now use Web-Service style components, technology and type safe languages (like SOAP), to preserve the semantics, ontology and intent of the message transaction. AMAZON, YAHOO and GOOGLE all use electronic kiosk style applications to transform requests from semi-intelligent information retrieval, into information rich responses, that is tailored using some form of server-side Decision Support System (DSS) technology.

2 Decision Support System

DSS use "Systems" that value add to the data being collected (using a knowledge-base to create an inference), prior to promulgating a response. A simple thermostat uses a temperature sensor to extract environmental conditions (precise data), while a comparator[3] sets the threshold used to switch between heating or cooling modes. When not switching, the DSS simply waits for the temperature to exceed a limit set by the operator (rules). More complex environmental control

[1] Tim Berners-Lee is acknowledged as the founder of the public version of the WWW, however the original architects where a panel of 12 members board called the Internet Architecture Board (IAB), employed by Defense Advanced Research Projects Agency (DARPA) and headed by Vince Surf which first convened in 1974. This board created standards, using instruments called Request For Comments (RFCs). The first topics raised included: Domain Name Servers (DNSs), Network File Systems (NFSs), email, Universal Resource Locators (URLs) and a host of subsequent protocols [2].

[2] Purchased by Microsoft in 1994 and released as its Internet Explorer.

[3] This is the logic center containing actuators and reference limits which are used to decide to heat or cool.

systems may be required, where that system controls a building, a complete facility or even a complex variety of environments that are dispersed geographically. Regardless of the technology used to generate the decision, the outputs relate to a predefined set of judgments based upon readings changing in the environment being controlled.

A DSS uses Computation Intelligence to collect, fuse and analyze data in order to derive a decision to enhance a human's ability in a given environment[4]. An expansion of research into a variety of DSSs created greater confidence in the decision being generated by computers [6]. The context could be used to determine the problem solving technique used. Examples include: communication-driven, data-driven, document-driven, knowledge-driven and model-driven architectures.

The growing volume of data had an overall effect on the efficiency of these systems. A series of tools where developed to measure the validity and results of many DSS. Factors such as; availability, accuracy, user response and operator ability, were raised as constraints to be considered before specifying courses of action [7]. The history of these systems has bee well documented. Power (2007) [8] for instance gives an excellent historical description of DSSs, which discusses many of the fundamental issues, however misses the vital human aspects. Keen reports on a study relating to "the theoretical aspects of organizational decision making". A preliminary study was done at Carnegie in the 1950's, while a more concise study from MIT was completed in the early 1960s [9].

3 Intelligent Decision Support System (IDSS)

Intelligent agents are described as computational systems that have properties such as autonomy, social ability, reactivity, and pro-activity [10]. Agent technology has expanded to include the concept of IDSS as shown in Figure 1. Intelligence can be described as a computational process that enables intelligence and interaction. Agents are increasingly required to bridge the gap between humans and machines and can be integrated to form social structures, that include teaming and learning. To support a human in the supervisory loop, a dynamic interface has to constructed to overcome the external constraints [11]. According to Tweedale et al. (2006) [12], new research and technologies are beginning to focus on developing human-centric multi-agent system in order to achieve mutual goals [13]. To accomplish an automated task effectively computers, hardware, software and even firmware agents are normally employed [14]. They must react to changes in the environment (reactivity) and plan activities to solve their goals (pro-activity).

[4] McCarthy developed LISP, enabling Bachman to create the first database management system which was converted into an Intelligent Decision Support (IDS) application called SAGE, while Feighambaum labeled his expert system DENDRAL.

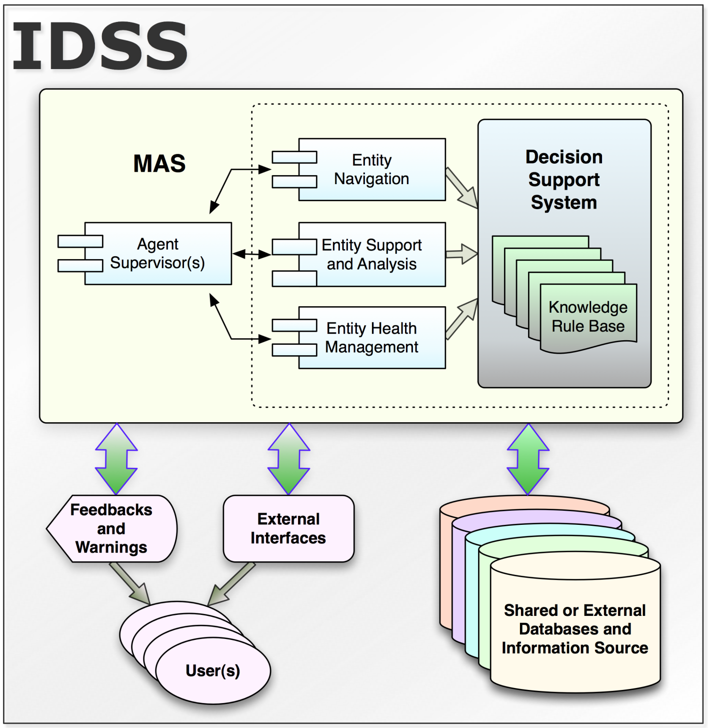

Fig. 1 The Principle of MAS within IDSS

4 Demonstration Examples

4.1 Teaming

Teams can be represented in many ways, however, experience from previous experiments indicates that Intelligent Agents using MAS architecture in a Beliefs, Desires, Intentions (BDI) framework that operates in a constrained environment is an effective model for this research. Agents have been defined in variety of ways as their functionalities have grown in different fields [12].

The technology of MAS is becoming more popular because agents can be ignored, informed, or forced to intelligently cooperate with one another, especially in the field of e-commerce where large-scale data mining is employed [15]. According to Dudek et al. (1993) [16], using MAS has advantages over a single agent. For example, replacing one complex robot with a group of simple robots that effortlessly explore an unknown area. This reduces the design complexity of each robot, is more economical, robust and scalable. The overall rate of failure would also decrease because if one or more robots fail, the whole system would still be able to function. The shared knowledge among agents should contain the knowledge of meaningful words and phrases the contextual information about particular issues or items of interest to one or more agent [17].

Adaptability refers to the MASs capacity to inter-operate and reorganize to form a new subgroup or dynamically adopt the functionality required to complete other tasks. For example, if the environment is small, it would limit the movement of agents within a specified hierarchy. Dynamic filters or components can be instantiated to achieve specific connectivity or customized configuration(s).

4.2 Component Messaging

Specialized techniques have been developed to enable the communication of agent based software components and the exchange of information between specific applications. Agents can communicate by sending messages directly to one and other using a prescribed format with rigid semantics. Theses messages are accepted by all the agents within the MAS and actioned as required [18]. Standards are required in order to establish compatible communication among any agent that gains access to that distributed MAS environment. Several common standards include: Agent Communication Languages (ACL), Knowledge Query Manipulation Language (KQML), Foundation of Intelligent Physical Agents (FIPA) and Simple Object Access Protocol (SOAP). Communication includes the delivery of complex attitudes or behaviours [17] through a specified architecture [12]. Therefore there is a requirement to identify the: syntax, semantics and pragmatism used, especially the symbols. Agents are able to communicate with other agents, using a specified precedence and hierarchy [16], however the need for immediate and broadcast messaging is becoming commonplace.

4.3 Decision Support

In some MAS, agents would be able to send direct messages to any individual agent by knowing their address and Identification [16]. Other topologies include agents only communicating with agents that are linked to them, for example, through a graph network or a hierarchy structure. In a hierarchical structure, there might be some controller agents who command and give tasks to other agents lower the hierarchy where those agents are only able to communicate with the controller agent. There are advantages and disadvantages to this type of structure. One advantage is that it reduces and simplifies the interactions between the agents because agents only interact with certain agents. However, one disadvantage is that communication among the agents in this type of structure is sensitive to the failure of one or few of the agents as this could cut the linkage of some group of agents from the rest of the swarm.

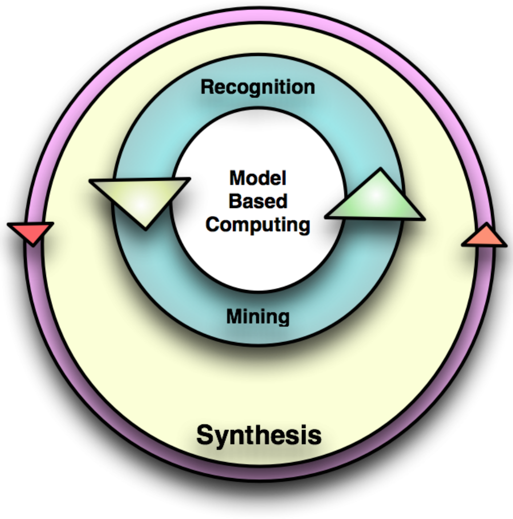

Fig. 2 The Concept of Micro-Simulated Decision Support [19]

4.4 Micro-simulated Decision Support

People rely on automated systems to make decisions, especially when time lines or expectations are compressed. There level of confidence and expertise in the system being used will also effect their response to any inference generated. Humans prefer simplicity and they attempt to streamline most processes to avoid stress, This may result in them skipping the "cognitive effort required to gather and process information [20]", and rely on intuition [21]. This method of coping often leads to mistakes via mode confusion, especially during intense periods of mission critical operation. *Automation bias* best explains the misuse of automated decision aids, although it cannot account for the lack of use in cases were it may have assisted [22].

Better training techniques and alternate models are being researched to solve some of these issues. The concept of using simulation to replicate the outcomes of a virtual reality is not new. The practice of using simulation to rehearse a number of possible outcomes, is becoming achievable, although real-time speeds are elusive. Through a series of sequencing, appending or combining the results of previous simulation, the system is able to recommend more precise courses action which would normally enhance the solution being sought. Doing this in parallel, at speeds that are faster than real-time (micro-simulations), data can be extrapolated to backfill missing variables or segments of a specific situation in order to gain an advantage in a variety of situations performed. To achieve this milestone, a combination of carefully chosen operator estimates and/or beliefs could be used as stimulants during specified scenario cycles. Each cycle must conform to a collection of rules, repeatable system states and consistent algorithms to stimulate the reality being attempted.

Thus using a combination of data analysis, mining and synthesis techniques in micro-simulated scenarios, researchers can employ an iterative approach to problem solving, using a variety of technologies or scenario instantiations that operate in parallel across a distributed environment. This promotes the enhanced concept of IDSS, which was initially described as Micro-Synthesized Simulated or Micro-Simulated Decision Support [19], however at its latest research day Intel combined a number of processes adopted above, they now call Model-Based Computing (MBC) as shown in Figure 2. The technique involves: the analysis of a volume of data to create a feasible model, use data to test instances of a parameter against a model and create an instance of a potential model. By increasing the number of processing elements within a system, analysts will be able to increase the number of parallel variables or constraints used to solve the problem [23].

5 Distributed Architecture

It is important for agents to gather knowledge about its environment to enable it to make intelligent decisions and actions [24]. Agents can be equipped with sensors to collect information. Communication between agents can be conducted directly between two agents or through a facilitator or interpreter. The exchange of information in a particular domain of knowledge that requires each agent to have shared knowledge of concepts in a particular domain is known as ontology. Each domain of knowledge would have its own ontology, although this can become flexible using more modern protocols (like SOAP)[5]. Figure 3 displays the physical architecture invoked in each computer/component to achieve distributed messaging based on this framework.

SOA models are constructed using loosely coupled components based on Web Services (Beans, Components or Enterprise Applications). Java classes are deployed as Web Services to consume targeted objects. Monson-Haefel (2003) postulated that Web Services, like distributed computing, is: "simply the hardest problem in computer science [28]". SOA provides an abstracted service at any granularity, which is great for hiding an entire system or sub-system. It can also become hierarchically dynamic using web service. Figure 3 displays the physical architecture invoked in each computer to achieve distributed messaging based on the SOAP framework. SOAP should have been a simple answer to interoperability, however the lack of proper standards initially made the job of distributing objects more frustrating than necessary. XML, SOAP, WSDL and Web Services retain the key topics in making successful distributed applications, although seamless integration of JAX-WS and JAXB is required to achieve the low-level functions and interfaces.

The SOA capability was included in Java 5.0 (Mar 2006) and was updated in Java 6.0 (Dec 2006). These Application Program Interfaces (APIs) reduced the expertise required by programmers to use Java Web Services in their SOA applications

[5] SOAP is an inter-application cross-platform communication mechanism. It defines a simple and flexible communication format that is based on XML documents passed across HTTP [25]. XML is used to represent data as an object instead of symbols and is passed in a standardized text-based format [26]. Using the SOAP protocol, information is packaged in SOAP messages that consists an envelope and body, with an optional header elements [27].

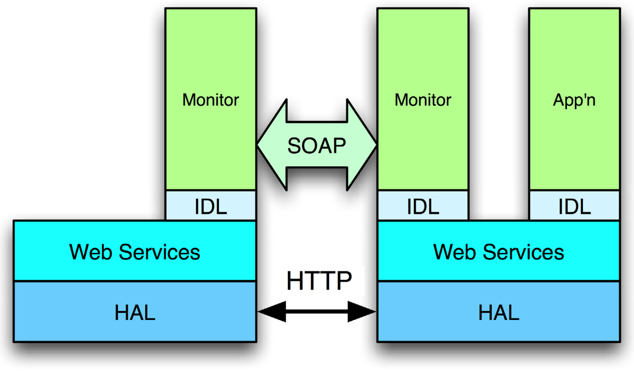

Fig. 3 Application Based on SOA Structure

(Knowledge about *Generics* and *Annotations* would aid productivity). Although developers will need to their update your knowledge on SOA, cross compilation of WSDL to Java and deployment before launching into creating a distributed application. Remote procedure code was includes in Java 4.0 (2002) with Web Services, Binding, Meta-data and Run-time behavior undergoing significant revision to trigger the *tipping point* of acceptance and wider spread implication.

6 Conclusion

Automation provides a path to increased productivity of many new machines and applications. Humans have traditionally been a bottleneck that impedes that goal, however technology is being used to erode that barrier. Agents, components, teams, DSS, IDSS, MBC and modern distributed computing techniques are being used to solve many of the issues encountered by researchers. Web services and SOAP are elements of that future and have evolved to a point were its evolution is nearing maturity. SOA makes it easier to product functional designs with limited functionality. Future testing and development is required to integrate these concepts with inter-operable computing technologies before world class applications mature commercially. The research conducted so far by KES has developed a blackboard design upon which segregated functions can be integrated into an application of aimed at achieving this goal.

References

1. Berners-Lee, T., Masinter, L., McCahill, M.: Uniform Resource Locators, RFC 1738. Internet Engineering Task Force, Internet Society, Geneva, Switzerland (1994)
2. Zittrain, J.: Interview: J. zittrain and d. clark, issues of domain names. Research Seminar on the Internet and Society. Harvard Law School, Cambridge (1997)

3. Salus, P.H.: Casting the Net: From ARPANET to Internet and Beyond.. Addison-Wesley Longman Publishing Co., Inc., Boston (1995); foreword By-Vinton G. Cerf
4. Mitchell, H.: Internet browser technology. Inventor of the Week, Massachusetts Institute of Technology, Cambridge, MA (Janurary 2001)
5. Spooner, J.G.: Intel breaks 3ghz speed barrier. ZDNet Australia, San Francisco, California, p. 1 (2002)
6. Yen, J., Fan, X., Sun, S., Hanratty, T., Dumer, J.: Agents with shared mental models for enhancing team decision makings. Decis. Support Syst. 41(3), 634–653 (2006)
7. Dhar, V., Stein, R.: Intelligent decision support methods: the science of knowledge work. Prentice-Hall, Inc., Upper Saddle River (1997)
8. Power, D.J.: A Brief History of Decision Support Systems. In: Quorum Books Division. Greenwood Publishing (March 2007) no. 156720497X
9. Keen, P.G.W., Morton, M.S.S.: Decision Support Systems: An Organizational Perspective. Addison-Wesley, Reading (1978)
10. Wooldridge, M., Jennings, N.: Intelligent agents: theory and practice. Knowledge Engineering Review 10(2), 115–152 (1995)
11. Finn, A., Kabacinski, K., Drake, S., Mason, K.: Design challenges for an autonomous cooperative of UAVs. In: Information Decision and Control, IDC 2007, Adelaide, Australia, February 2007, vol. 4477 (2007)
12. Tweedale, J., Ichalkaranje, N., Sioutis, C., Jarvis, B., Consoli, A., Phillips-Wren, G.: Innovations in multi-agent systems. Journal of Network and Computer Applications 30(3), 1089–1115 (2006)
13. Parsons, S., Pettersson, O., Saffiotti, A., Wooldridge, M.: Artificial Intelligence Today: Robots with the best of intentions. In: Veloso, M.M., Wooldridge, M.J. (eds.) Artificial Intelligence Today. LNCS, vol. 1600, pp. 329–338. Springer, Heidelberg (1999)
14. Nwana, H.S.: Knowledge engineering review, pp. 1–40 (1996)
15. Wooldridge, M.: Verifiable semantics for agent communication languages. In: Proceedings. International Conference on Multi Agent Systems, pp. 349–356. IEEE Computer Society, France (1998)
16. Dudek, G., Jenkin, M., Milios, E., Wilkes, D.: Taxonomy for swarm robots. In: International Conference on Intelligent Robots and Systems 1993, IROS 1993. Proceedings of the 1993 IEEE/RSJ, Yokohama, Jpn, July 26-30, 1993, vol. 1, pp. 441–447. IEEE, Piscataway (1993)
17. Labrou, Y., Finin, T., Peng, Y.: Agent communication languages: The current landscape. IEEE Intelligent Systems and Their Applications 14(2), 45–52 (1999)
18. Vaucher, J., Ncho, A.: Jade tutorial and primer (April 2003-2004)
19. Siotus, C., Tweedale, J.: Challenges and opportunities for a complex and networked world: Mission systems and simulators. In: Bhalla, J. (ed.) SimTecT 2006 Simulation Conference, SIAA, Melbourne, Australia, 29 May - 1 June 2006, pp. 1–6 (2006)
20. Mosier, K.L., Skitka, L.J.: Human decision makers and automated decision aids: made for each other. In: Cohen, M.S., Parasuraman, R., Freeman, J.T. (eds.) Automation and Human Performance: Theory and Applications. Lawrence Erlbaum Associates, Mahwah (1996)
21. Tversky, A., Kahneman, D.: The framing of decisions and the psychology of choice. Science 211, 453–458 (1981)
22. Dzindolet, M.T., Beck, H.P., Pierce, L.G.: Encouraging human operators to appropriately rely on automated decision aids. In: Research, O.H., Directory, E. (eds.) The 6th International Command and Control Research Program (ICCRTS), DTIC, Annapolis (2002)
23. Koehl, S.: What would you do with 80 cores? Research@Intel, Santa Clara, CA, USA, pp. 1–6, Tech. Rep. (2007)

24. Bigus, J.P., Bigus, J.: Constructing Intelligent Agents Using Java: Professional Developer's Guide, 2nd edn. Wiley, Chichester (2001)
25. Seely, S., Sharkey, K.: SOAP: Cross Platform Web Services Development Using XML. Prentice Hall PTR, Upper Saddle River (2001)
26. Benz, B., Durant, J., Durant, J.: XML Programming Bible. Wiley Publishing, Inc., New York (2003)
27. Graham, S., Davis, D., Simeonov, S., Daniels, G., Brittenham, P., Nakamura, Y., Fremantle, P., Koenig, D., Zentner, C.: Building Web services with Java: making sense of XML, SOAP, WSDL, and UDDI. Developer's Library (2002)
28. Hansen, M.: SOA Blog. Service Centric, San Fransisco, USA (2006)