 ISBN 0 9585370 9 7

An Ontology-driven Multi-agent approach for Healthcare

Gajun Ganendran, Quynh-Nhu Tran, Pronab Ganguly, Pradeep Ray and Graham Low

School of Information Systems, Technology and Management
The University of New South Wales, Kensington, NSW 2052, Australia

Abstract

Healthcare systems usually require a high level of collaboration amongst health entities. Maintaining consistency within this collaborative framework is a hurdle faced by healthcare professionals. This paper outlines a new approach to dealing with this issue through the development of an ontology-driven multi-agent system. It examines the case study of diabetes management.

1. Introduction:

Recent years have seen a remarkably growing interest in the "agent" technology and its application. Gartner forecasts recently that enterprise automation, which includes software agents and AI, will account for almost 50% of total IT spending in 10 years [5].Meanwhile, ontology has become increasingly popular in the computing community. Ontology can play an essential role in the design of multi-agent systems, leading to the notion of "ontology-driven multi-agent systems".

This paper examines the use of ontology-driven multi-agent systems in the healthcare domain, with a case study in diabetes management. Treatment of diabetes entails limited patient contact with multiple healthcare professionals such as the general practitioner (GP), specialist and clinicians. The interaction usually involves regular patient visits to the GP who collects blood samples to send to the clinical laboratory for testing. The result is reviewed by the GP to determine the patient's condition to prescribe medication. All complicated cases are referred to a specialist. Two major problems can be identified from the current system: 1) the results received by the GP or the specialist become outdated due to the speed of communication, and 2) the results may become inconsistent due to the lack of collaboration between health entities.

This paper aims to overcome the above problems by developing a system where: 1) the patient, laboratory clinicians, GPs and specialists *interact effectively* amongst each other to provide efficient health care to the patient, and 2) *consistency* within a heterogeneous environment is maintained.

Major components of the system are a *multi-agent architecture* and a diabetes *ontology*. Toshiba Beegent Platform and Protégé were used to develop the system and the FIPA standard-based ontology respectively.

2. Multi-agent Systems Development:

2.1 Multi-agent systems and existing development technologies:

A software agent is a piece of software that acts *autonomously* on behalf of human users to perform some set of tasks [16]. Most advanced applications of agents, including the one discussed in this paper, employ "intelligent" software agents, which are not only autonomous but also reactive, pro-active, and capable of interacting with each other in a flexible manner [16].

A multi-agent system offers the added value of an ensemble of agents. It presents a powerful and natural metaphor for conceptualising and designing many software applications [9], as will be illustrated with the diabetes-management scenario. Multi-agent systems also facilitate the interoperability of heterogeneous systems. The idea is to "agentify" the heterogeneous components, that is, to wrap these components with an agent layer that enables them to interoperate with each other via a uniform agent communication language [8].

Nowadays, software agents can be readily implemented to provide practical, industrial-strength help in everyday environment, thanks to the substantial progress in agent-enabling technologies. Languages for programming agents are now available (Java, Smalltalk, Python, Telescript and Perl5...), and so are agent communication languages (for example, KQML and AgentTalk). Developers can now find numerous agent construction toolkits and platforms, including Toshiba Beegent, JACK, JADE, Voyager, Aglets, MadKit and Concordia... The number of development methodologies for multi-agent systems, however, is still quite small. Only a few have been found that take into consideration the special features of agent-based systems, and that provide a reasonably complete and thorough coverage of the development lifecycle. These methodologies take inspiration either from object-oriented methods (for example, MaSE [15], GAIA [17], MESSAGE [2], Prometheus [12], MASSIVE [11], and Kinny et al.'s methodology [10]) or from knowledge engineering (for example, Cassiopeia [1] and MAS-CommonKADS [7]).

2.2 Role of Ontology in the development of Multi-agent systems:
Ontology is "a formal, explicit specification of a shared conceptualisation" [4]. A conceptualisation refers to an abstract model of a domain of interest. It captures the relevant concepts that exist in the domain, and the relationships that hold among them [6]. In simple terms, an ontology provides a vocabulary of concepts and relations with which to model a domain.

Accordingly, ontology can be used as a formal, declarative knowledge-representation mechanism to specify the application domain for a multi-agent system, and knowledge for individual agents [3]. Ontology is also essential to agent communication and coordination [3]. For agents to uniformly interpret the exchanged messages, they need to share the same understanding of the concepts conveyed in the messages. This is achieved by "committing" the agents to the same ontology, that is, to make the agents use a shared ontology in a coherent and consistent manner [6].

2.3 Ontology-driven Multi-agent System for Healthcare:
Multi-agent systems provide a powerful framework to help the patient, GPs, laboratory clinicians and specialists to interact and collaborate effectively. A typical doctor-specialist interaction would involve:

- Doctor contacting specialist about patient
- Specialist formulating treatment plan based on patient information
- Doctor formulating treatment plan based on patient information
- Doctor and specialist negotiating ideal treatment plan for patient

Each party in the above scenario can be modelled as an agent that interacts with other agents on the user's behalf. Different from the conventional objects, agents can engage in flexible and robust interactions as they can be endowed with sophisticated social skills (such as persuasion abilities), negotiation skills, coordination mechanisms, and flexible agent communication languages [8][9]. The support of multi-agent systems for interoperability is also useful in the healthcare domain. Each healthcare professional may use different systems to assist in the management of patient records and

treatment plans. Each system may reside on different platforms across a network. To unify these systems effectively, agents can be used as "wrappers" around each application and communicate with other wrappers via an agent communication language.

Consistency within a heterogeneous environment can be maintained by using ontology. Fig 1 describes a simple conceptual definition for the diabetes domain. Classes are used to describe concepts in the domain. In Fig 1, Diabetes represents the main class concept containing several sub concepts such as Treatment and Symptoms. Slots (not shown in the figure) are attributes associated with a class.

By defining relationships and attributes for these classes, we would be able to formally model the diabetes domain. Once developed into a knowledge base, agents can access these terms and relationships to reason and to derive answers to queries. The ontology can also be "committed" by communicating agents in order to resolve the issue of maintaining consistency with medical terminology and standards.

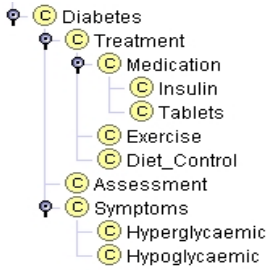

Fig.1. Simple conceptual definition of the diabetes domain

3. Development of Multiagent System for Healthcare: A Case Study:

3.1 Architecture of an ontology-driven agent system:
The conceptual model of a healthcare ontology-driven multi-agent system is shown in Fig. 2. It consists of the following components:

- *Diabetes Agent System*: provides an interface for the GP or specialist, sends requests from the user to the Ontology Agent (OA), and displays or responds to messages from OA.

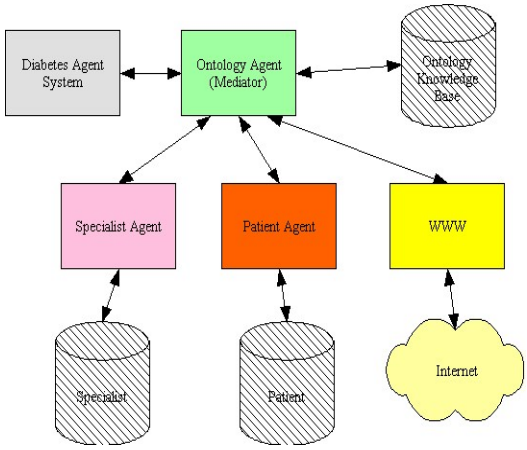

Fig.2. Conceptual model of an ontology-driven multi-agent system

- *Ontology Agent*: receives messages from any agent in the system and translates the messages

into primary concepts defined in the ontology (for example, if the message contains the term "web" and the primary concept defined in the ontology is "internet", "web" will be translated into "internet"). OA then queries all agent systems to determine which agent can satisfy the translated message. OA then sends the message to the destination agent.

- *Specialist Agent*: provides information about treatment options for a patient, and negotiates with Diabetes Agent System to arrive at optimal treatment plan.

- *Patient Agent*: responds to requests for patient information and interfaces with the patient to gather information.

- *WWW Agent*: provides access to the World Wide Web, performs a search on data requested by other agents, and launches an interface with the requested information.

3.2 Representation of Context Information in Domain Ontology:
The diabetes ontology is represented in a relational database management system (RDBMS). The relational schema should reflect the logical hierarchy of the key concepts in Fig 1 and retain all the constraints [13]. The format of our relational database schema is shown in Fig 3.

Column	Type	Description
Frame	Integer	frame id
Slot	Integer	slot id
Facet	Integer	facet id
value_index	Integer	maintains ordering of slot_or facet_value entries
value_type	Smallint	stores the type of the value stored in slot_or_facet_value.
Slot_or_facet_value	varchar(254)	stores slot or facet value
Long_slot_or_facet_value	longvarchar	stores slot of facet value greater than 254 characters.

Fig. 3. RDBMS schema to represent ontology

All classes, slots and facets are mapped as frames in the RDBMS schema. The hierarchy of our ontology is retained in the schema by the order of frame values. For example in fig 1, the concept of Medication contains two sub-concepts called Insulin and Tablets. Assume that Medication is assigned a frame id of 2000, Insulin and Tablets will be given consecutive numbers such that this hierarchy is maintained.

The attributes for a particular class that include instances and child sub-classes for that super-class are placed in the slot_or_facet_value field for the corresponding frame id. For example, assume that class Medication is assigned a frame id of 2000, any data stored in the field slot_or_facet_value corresponding to frame_id of 2000 will contain the attributes of class Medication. We can now derive the class-hierarchy and attributes of each class using this schema. This information can then be used to retrieve the conceptual understanding of the domain via simple database queries.

4. Implementation using Toshiba Beegent and Protege

Toshiba Beegent development framework is used for the design of user-level agents as well as the Ontology Agent. This agent system conforms to the FIPA Agent Communication Language (ACL) specification. The ACL framework in Beegent is encoded in XML to allow for transport via HTTP [14].

The Knowledge Base is developed using Protege. Protege allows for the development of a conceptual ontology using a graphical interface. Fig 1 shows the conceptual design of the diabetes ontology using Protege. Protege includes export capabilities to JDBC, allowing for access to the knowledge server via simple database queries. It also supports the inclusion of OKBC framework to access the knowledge server.

5. Conclusion

This paper described an ontology-driven multi-agent approach to the development of healthcare systems, with a case study in diabetes management. The approach used the multi-agent framework to enhance the doctors-specialists-clinicians collaborations, and ontology to maintain consistency within a heterogeneous environment. A prototype system was developed using the Toshiba Beegent framework and Protege. The prototype was based on the principles discussed in this paper and is being tested. The results gained from evaluating this system will help us determine the practical effectiveness of such systems.

References:

1 Collinot, A., Carle, P. & Zeghal, K. 1995. 'Cassiopeia: A Method for Designing Computational Organizations'. *Proceedings of the 1st International Workshop on Decentralized Intelligent Multi-Agent Systems*, 124-131.

2 Eurescom. 2001. 'Methodology for Agent-Oriented Software Engineering'. http://www.eurescom.de/public/projectresults/P900-series/907ti1.asp

3 Falasconi, S., Lanzola, G. & Stefanelli, M. 1996. 'Using Ontologies in Multi-Agent Systems'. *Proceedings of the 10th Knowledge Acquisition for Knowledge-Based Systems Workshop (KAW'96)*. http://ksi.cpsc.ucalgary.ca/KAW/KAW96/falasconi/

4 Fensel, D. 2001. *Ontologies: A silver Bullet for Knowledge Management and Electronic Commerce*. Springer-Verlag Berlin Hiedelberg.

5 Gengler, B. 2002. 'Silicon super-agents'. *The Australian*, Tuesday 30 April, pp1,4..

6 Gruber, T. 1995. 'Toward Principles for the Design of Ontologies Used for Knowledge Sharing'. In Guarino, N & Poli, R. (eds.). *Formal Ontology in Conceptual Analysis and Knowledge Representation*. Kluwer Academic Publishers.

7 Iglesias, C. A., Garijo, M., Gonzalez, J.C., & Velasco, J.R.1998. 'Analysis and Design of Multiagent Systems Using MAS-CommonKADS'. In Singh, M.P., Rao, A., Wooldridge, M.J. (eds.). *Intelligent Agents IV (ATAL'97)*. Springer-Verlag, Berlin.

8 Jennings, N.R. & Wooldridge, M. 1995. 'Applying Agent Technology'. *Applied Artificial Intelligence*, 9 (4), 351-359.

9 Jennings, N.R., Sycara, S. & Wooldridge, M. 1998. 'A Roadmap of Agent Research and Development'. *Journal of Autonomous Agents and Multi-Agent Systems*, 275-306.

10 Kinny, D., Georgeff, M., & Rao, A. 1996. 'A Methodology and Modelling Technique for Systems of BDI Agents'. *Agents Breaking Away: Proceedings of the 7th European Workshop on Modelling Autonomous Agents in a Multi-Agent World (MAAMAW'96)*, 56-71.

11 Lind, J. 1999. *MASSIVE: Software Engineering for Multiagent Systems*. PhD Thesis, University of Saarbrucken, Germany.

12 Padgham, L. & Winikoff, M. 2001. 'Prometheus: a methodology for developing intelligent agents'. Tutorial at the 14th Australian Joint Conference on Artificial Intelligence (AI'01). To appear in the proceedings of the First International Joint Conference on Autonomous Agents and Multi-Agent Systems (AAMAS 2002).

13 Protégé 2000 – Integrated Ontology Development Tool. http://protege.stanford.edu/

14 Toshiba. 2002. 'Bonding and Encapsulation Enhancement Agent'. http://www.toshiba.co.jp/beegent/index.htm

15 Wood, M. & DeLoach, S.A. 2000. 'An Overview of the Multiagent Systems Engineering Methodology'. *Proceedings of the 1st International Workshop on Agent-Oriented Software Engineering (AOSE-2000)*, 207-222.

16 Wooldridge, M. 1999. 'Intelligent Agents'. In Weiss, G. (eds.). *Multiagent Systems: A Modern Approach to Distributed Artificial Intelligence*. MIT Press, Cambridge, Massachusetts.

17 Wooldridge, M., Jennings, N.R. & Kinny, D. 2000. 'The Gaia Methodology for Agent-Oriented Analysis and Design'. *Autonomous Agents and Multi-Agent Systems*, 3(3), 285-312.

Contact: Email: p.ray@unsw.edu.au; Tel: 61 2 9385 5890; Fax: 61 2 9662 4061