

A Method and Architecture for Building Compliance Agents

James P. Davis

HealthMagic Corporate Technology Center
1901 Main Street, Suite 1570
Columbia, SC 29201 USA
jjmd@healthmagic.com

Michael N. Huhns
Ronald D. Bonnell

Center for Machine Intelligence
University of South Carolina
Columbia, SC 29208 USA
huhns@sc.edu
bonnell@sc.edu

Introduction

This position paper presents preliminary results on defining a method and architecture for building agents that monitor compliance to specified plans or protocols in the healthcare domain. This work is being pursued as a precursor to constructing an automated, problem-solving task-specific, knowledge acquisition environment for agent-based compliance monitoring software applications for healthcare.

The status of our work is as follows. One agent system has been developed and evaluated for treatment compliance monitoring in a disease management sub-domain, while a second agent system is being developed for compliance monitoring of occupational safety in the healthcare workplace. We believe that our work in creating an architecture for compliance monitoring problems can be used as a basis for building a knowledge-level, task-specific agent development tool set, along the lines of those created for generic-task based knowledge acquisition environments (Musen 1989; Tu et al. 1992; Linster and Musen 1992).

Our current emphasis is on articulating the method devised for creating these agent systems, and its underlying system architecture, that is enabling us to create a model-driven acquisition environment for building domain-independent compliance agent applications. Instead of using knowledge-level analysis techniques (Musen 1989) for articulating our problem-solving architecture, we have adopted object-oriented analysis techniques for this purpose. We have found these useful for capturing problem semantics that are precise and rigorous, yet amenable to automation through a well-understood mediating interface required of a knowledge acquisition tool kit.

Rationale for Agents

Intelligent software agents are becoming recognized as a preferable technique over more traditional, monolithic

knowledge systems for constructing personalized, collaborative, distributed, intelligent applications in many domains (Huhns and Singh 1997). In healthcare, this approach is of particular interest (Szolovitz et al. 1994; Hayes-Roth et al. 1996) because of the increasing demand for individualization and personalization at the point where the healthcare system makes contact with people—including patients as well as caregivers. Agent-oriented systems techniques are relevant and useful in healthcare problems where there is benefit to the human in having personalized “advocates” acting on their individual behalf in the domain.

One of the key problem-solving tasks in healthcare is monitoring compliance to specified protocols and plan regimens. Protocols are operational guidelines defined for areas of practice, including prescribed treatment regimens (as in the case of patient disease management) and workplace occupational practices (as in the case of safety guidelines for healthcare workers).

Our interest is in defining an analysis method that will allow us to look across several different compliance domains to elucidate the underlying problem-solving task structure, and in devising an appropriate system architecture that will allow us to capture the appropriate domain knowledge to operationalize a cadre of personalized agents into service in different compliance monitoring domains in healthcare.

Analysis Method and Architecture

We proceeded with the following approach in our analysis:

Partition and decompose problem space. In our compliance domains, we start with partitioning the problem domain into strategic, tactical, and operational components.

Identify and create inventory of knowledge-level components and behaviors. Next, we identify the goals, actions and knowledge structures of the decomposed system, referred to as knowledge-level analysis (Musen 1989). We selected modeling languages contained in the UML formalism (Rational 1997), specifically using the Use Case notation, as the mechanism for identifying and describing the knowledge-level interactions between the system and the domain “actors”.

Identify and localize compliance monitoring activities. Given an inventory of Use Case interactions created from the previous step, we refine each of these so as to (1) highlight the essential functional transformations of the interaction, (2) define the order and direction of interactions, and (3) identify the important “artifacts” of interaction, namely, the data storage elements required to process the interaction. We used the dataflow diagram notation (Edwards 1993) for this purpose. Also at this time, we identify opportune locations in processes supporting the system interactions that are candidates for having autonomous agents associated with them.

Create an inventory of agents. Once the agents are identified relative to the system processes, we create an inventory, indicating the agent category name, the actions that trigger the agent to take action, the data sources the agent uses in its reasoning activities, the basic nature of its output actions, and the destination “actors” who receive some notification of compliance information. We use a tabular form to represent this information.

Identify and model the essential compliance lifecycle. In our formulation of compliance problem solving, each compliance agent manages one or more “artifacts” of compliance, each having a sequence of state transitions constituting a “lifecycle” (Shlaer and Mellor 1992). Each specific category of compliance agent has its own lifecycle objects that it manages. We have had good success in using the state transition diagram (Edwards 1993) notation as an analysis and documentation medium for assessing the completeness of lifecycle depiction for our agents.

Map the abstract architecture onto the agent environment. At this point, we need to map each compliance agent onto its portion of the logical architecture where it will be delivered. In addition, we must indicate how the agents will interact with one another in that environment. It has been our experience that additional “helper” agents are needed to facilitate distributed, coordinated agent behavior (Huhns and Singh 1997). To that end, we model the sequences of interactions among the agents comprising the agent protocol. We use UML sequence diagrams (Rational 1997) for this purpose.

The principal observation about identifying agent behaviors is that compliance agents track “artifacts” of compliance through a lifecycle, and that deviations from prescribed compliance protocols indicate opportunities for compliance agents to take remedial action, such as sending reminders and alerts. Thus, remedial goal-setting is done relative to where the agent is currently located in the lifecycles of the artifacts it is tracking.

We think of the architecture in terms of logical and physical components. The logical, knowledge-level components consist of the conceptual structures and behaviors for the agents, their knowledge sources, goals and agendas, and interaction protocols. These are mapped onto a physical architecture consisting of the JATLite agent toolkit coupled with the JESS inference engine and JDBC interfaces to a backend database.

Summary

This position paper has presented our approach to building tools for agent-based systems. We selected a narrow task-specific problem domain in which agents have high value, and analyzed the problem-solving structure for the compliance monitoring task, devising an appropriate architecture fitting the results of knowledge-level analysis.

We have used a number of graphical notations as a basis for articulating and exploring the problem-solving architecture for these systems. Using these notations made it easier to look across multiple agent applications and find common problem-solving architectural patterns.

The future direction is to use the method and architecture formulations to drive construction of appropriate knowledge acquisition tools for compliance agents. The goal is to have tools allowing compliance agent behaviors to be programmed by clinical practitioners for healthcare applications for which personalized advocacy in a collaborative, distributed workflow is a desirable trait

References

- Edwards K. 1993. *Real-Time Structured Methods—System Analysis*. New York: John Wiley.
- Huhns, M. N., and Singh, M. P., 1997. Agents and Multiagent Systems: Themes, Approaches, and Challenges. To be published.
- Larsson, J. E., and Hayes-Roth, B. 1998. Guardian: An Intelligent Autonomous Agent for Medical Monitoring and Diagnosis. *IEEE Intelligent Systems*, 12:58-64.
- Linster, M., and Musen, M. A., 1992. Use of KADS to Create a Conceptual Model of the ONCOCIN Task. *Knowledge Acquisition* 4, 55-87.
- Musen, M. A. 1989. Conceptual Models of Interactive Knowledge Acquisition Tools. *Knowledge Acquisition* 1, 73-88.
- Rational 1997. *UML Notation Guide*, Version 1.1.
- Shlaer, S., and Mellor, S. J. 1992. *Object Lifecycles—Modeling the World in States*. Englewood Cliffs, NJ: Yourdon Press.
- Szolovitz, P., Doyle, J., Long, W. J., Kohane, I., Pauker, S. G., 1994. Guardian Angel: Patient-Centered Health Information Systems. TR-604, MIT Laboratory for Computer Science.
- Tu, S. W., Shahar, Y., Dawes, J., Winkles, J., Puerta, A., and Musen, M. A., 1992. A Problem-solving Model for Episodic Skeletal-plan Refinement. *Knowledge Acquisition* 4, 197-216.