MET3: AN INTEGRATIVE OPEN SOURCE BASED MULTI-AGENT CLINICAL DECISION SUPPORT SYSTEM

JELBER SAYYAD-SHIRABAD[1], SZYMON WILK[2,*], WOJTEK MICHALOWSKI[1,] KEN FARION[3]

[1] *University of Ottawa, Ottawa, Ontario, Canada*
[2] *Poznan University of Technology, Poznan, Poland*
[3] *Children's Hospital of Eastern Ontario, Ottawa, Ontario, Canada*

Clinical decision-making is a multi-stage process that involves a series of tasks requiring different types of clinical knowledge. Decision support can play an important role during the entire process. However, research on clinical decision making and clinical decision support systems has been focused on individual tasks. In this paper we present the design and implementation of MET3 – a multi-agent system providing an integrative support for tasks involved in the decision-making process. MET3 helps physicians with data collection, diagnosis formulation, treatment planning and finding supporting evidence. It integrates with external hospital information systems and runs on various computing platforms such as desktops and mobile devices available at the point of care. The system has been implemented using only open source and free software technologies.

1. Introduction

Clinical decision-making is a multi-stage process that involves a series of tasks including data collection, diagnosis formulation and treatment planning. Successful completion of these tasks requires different kinds of clinical knowledge[1]:

- data collection relies on *factual knowledge* of what to collect and how to collect;
- diagnosis formulation relies on *conceptual knowledge* of how to establish and evaluate potential diagnoses;
- treatment planning relies on *procedural knowledge* (often associated with the notion of evidence-based medicine) on how to develop such plan.

Considering the complexity and the kind of knowledge required, decision support can play an important role during the entire process. While significant research has been devoted to supporting individual tasks, less attention has been paid to integrative support that assists the physician throughout the entire

[*] Research described in this paper was conducted while Dr. Wilk was a postdoctoral fellow at the Telfer School of Management, University of Ottawa

process. A review by Berlin et al [2] indicates that the majority of available clinical decision support systems (CDSSs) are focused on specific tasks.

Additionally, to use a CDSS in diverse hospital settings it should be independent of computing platform and integrate with existing hospitals IT infrastructure.

In this paper we discuss the design and implementation of an integrative multi-agent CDSS called Medical Emergency Triage III (MET3), which supports tasks across decision making process. The system helps with collecting structured data, diagnosis based on this data, suggesting a treatment plan and finding evidence in support of the plan. MET3 integrates with external hospital information systems (HISs) to share patient data and runs on multiple computing platforms We show that one can develop such relatively complex system by using existing open source software (OSS). To the best of our knowledge this is the first system that provides this level of integration and solely relies on OSS and free software to do so.

In Section 2 we present the design of MET3. In Section 3 we briefly discuss the open source and free software technologies used to implement MET3. Testing the integration capabilities of MET3 with other HIS systems is covered in section 4. Finally, we present our conclusion in Section 5

2. MET3 Design

2.1. *Design Methodology*

Recent research [3] on CDSS design suggests that requirements mentioned above would be best satisfied with an architecture that supports service models, thus making both service and agent-oriented approaches as viable options.

In developing MET3 we followed the agent-oriented approach due to its advantages demonstrated in healthcare-related applications [4]. This was complemented with an ontology-driven design to help with creation of specific abstract models representing different kinds of clinical knowledge and essential system components, in a fashion similar to our earlier research [5].

We used the O-MaSE (Organization-based Multi-agent System Engineering) methodology to design MET3 [6]. O-MaSE views a multi-agent system (MAS) as an organization of agents with a common goal. It does not make assumptions regarding complexity or intelligence of agents. The environment in which MAS functions is modeled by a domain ontology, which is in line with the ontology-driven design.

Since MET3 agents are reactive, a basic O–MaSE process is sufficient. The tasks in this process are: goal modeling, domain modeling, agent classes modeling, protocols modeling and plans modeling. The outcomes of these tasks are the system goals, agent classes, agent plans, and inter-agent protocols. Due to limited space we will only present the domain model and agent class model, which identifies the individual agents used in the system. More detailed treatment of the analysis and design of the system can be found in [7].

2.2. *Domain Model*

Figure 1 shows the most essential concepts and relationships in MET3's domain model and ontology. The domain model is divided into five components:

- *meta-data ontology* – defines concepts describing patient data, e.g., presenting complaints, domain and accepted values for clinical attributes;
- *data ontology* – defines concepts that capture patient data handled by MET3 (e.g., values of clinical attributes);
- *interface ontology* – defines concepts representing various components of the MET3 user interface, e.g., screens, forms, widgets. It includes the concept of a platform to allow platform tailored user interfaces

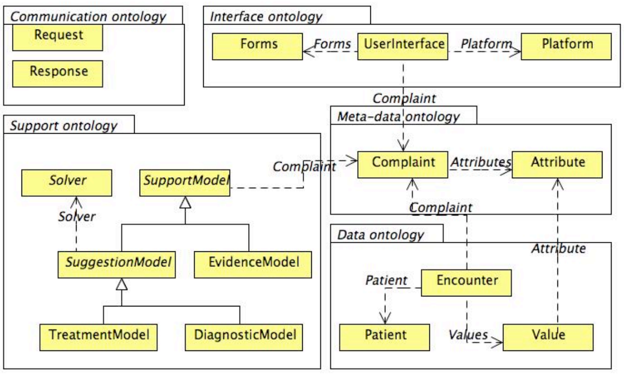

Figure 1. Domain model

- *support ontology* – defines concepts representing support models (further specialized into diagnostic, treatment and evidence models) and solvers. Solver corresponds to a generic processing mechanism that is capable of running a patient specific data through a support model to obtain a solution (e.g., possible diagnosis);
- *communication ontology* – defines concepts used by messages (requests and responses) exchanged by agents

2.3. *Agent Class Model*

Figure 2 shows the agent class model for MET3. System agents are identified by <<Agent>> keyword, while non-agent entities such as the user, model and evidence repositories, and HL7 integration engine are shown as actors. Requests sent from one agent to the other are represented as arrows. The direction of the arrow identifies the sender and the receiver agents in the communicative act.

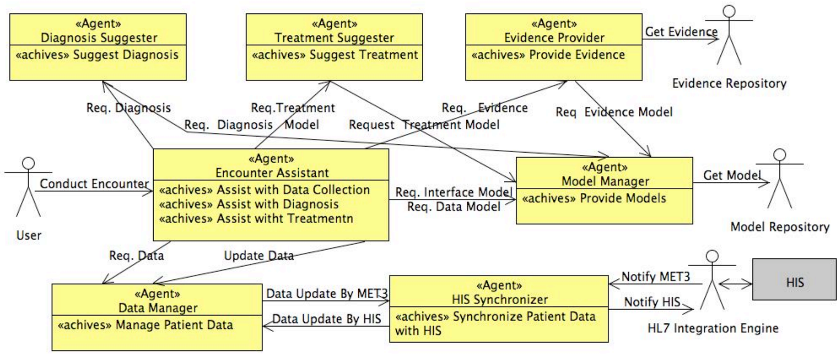

Figure 2. Agent class model

MET3 system consists of the following agents:

- *Encounter assistant* - provides a graphical user interface to a physician and interacts with other agents in the system based on the user's input;
- *Model manager* – manages abstract models stored in the model repository;
- *HIS synchronizer* - receives and passes relevant changes between external HIS and MET3 via the HL7 integration engine;
- *Data manager* – manages patient data stored locally in the data repository. It interacts with HIS Synchronizer agent to maintain a consistent and current view of the patient data in the system;
- *Diagnosis suggester* -- suggest a possible diagnosis based on the relevant patient data and the appropriate diagnosis models[1];
- *Treatment suggester* – suggests a treatment plan on the basis of the diagnosis verified by the physician and the appropriate treatment models[2];
- *Evidence provider* – provides clinical evidence (e.g., systematic reviews) from an evidence repository that supplements suggested treatment plans.

Encounter assistant (EA) acts as the physician's gateway to MET3. To update patient data, EA asks the *model manager* (MM) for appropriate interface model and the *data manager* (DM) for the current patient data. Then it generates the user interface, which allows the physician to modify the data. Such changes

[1] We developed these models by applying machine learning techniques to historic clinical data.

[2] For instance MET3's treatment model for asthma exacerbation is extracted from Guidelines for Emergency Management of Pediatric Asthma.

result in sending requests to DM to update the data repository. Subsequently the *HIS synchronizer* notifies HIS about this update.

To establish a diagnosis, EA passes this request to the *diagnosis suggester* (DS), which in turn requests a diagnostic model from the MM. Using the proper model/solver pair DS, responds with a diagnosis for the available data. EA reports this diagnosis to the physician.

Similarly, a treatment plan request is passed to the *treatment suggester* (TS), which applies the appropriate treatment model/solver pair to the data and interacts with the *evidence provider,* if needed, to retrieve relevant supporting evidence from the evidence repository. Again, EA presents the results to the physician.

3. Technologies Used to Implement MET3

MET3 has been developed entirely using open source or free software systems. The multi-agent "core" of MET3 is implemented using JADE (Java Agent DEvelopment Framework)[3]. JADE is a mature software framework and execution environment that runs on all computing platforms that support Java. All agents and their interactions as shown in Figure 2 are implemented through sub-classing appropriate JADE base classes.

The model and data repositories are implemented using Protégé[4], which allows one to create, maintain and programmatically manipulate an ontology. The repository holding clinical evidence is implemented using a MySQL[5] database and complemented by Hibernate Search[6], which combines the full text search and indexing power of Apache Lucene and object relational persistence.

Mirth[7] HL7 integration engine is used to filter and route messages between MET3 and external HIS systems. HL7 is a widely adopted standard and was also used by our collaborating hospital.

Sun Java Wireless Toolkit[8] is used to run EA on mobile devices.

Finally, we used WEKA[9], a collection of machine learning algorithms, to develop diagnosis models from retrospective data and to construct solvers that run patient data through these models to obtain a diagnosis.

---

[3] http://jade.tilab.com/
[4] http://protege.stanford.edu/
[5] http://www.mysql.com/
[6] https://www.hibernate.org/
[7] http://www.mirthcorp.com/
[8] http://java.sun.com/products/sjwtoolkit/
[9] http://www.cs.waikato.ac.nz/ml/weka/

4. Emergency Department Integration Testing

A deployed CDSS needs to operate within the existing HIS infrastructure. Essential to this operation is the ability to communicate with other HIS systems, as needed. Similar to most HIS systems, MET3 uses HL7 messages to communicate with other healthcare software. We simulated a typical HL7 message path through such an integrated setting, whereby a) a source node in HIS generates HL7 message for admitted patients b) an intermediate node built around Mirth, and used by MET3, captures and routes these messages c) HIS Synchronizer, which is the MET3 agent, receives these messages and forwards them to the data manager.

To do this we created an HL7 source node that simulates an ADT (admission-discharge-transfer) system by generating ADT admission messages, as it would be when a patient is admitted to a hospital. This application allows one to enter typical information such as patients name, age and complaint that are collected at patient admission time. The application then generates an HL7 message containing this information.

Based on publically available admission statistics, we simulated a typical day in Emergency Department of Children's Hospital of Eastern Ontario (CHEO). We computed the average number of admissions for all complaints in a day to be about 145 patients. Over a period of 12 hours we admitted 125 patients, which is a relatively heavy load. Amongst them there were 40 patients with abdominal pain or asthma, which would be further processed by MET3. We simulated 10 concurrent sessions of patient assessment using MET3. Along with showing successful test of HL7 message routing and processing, these tests covered the use of various functionalities of the MET3 system. The only difference between these tests and a deployment test would be the change in the source node that generates HL7 messages.

Additionally, we have evaluated Mirth based HL7 routing technology in CHEO in the context of a separate clinical trial. The system developed for this clinical trial integrates with CHEO's EPIC and SUNRISE systems. These systems are used for patient management and collecting clinical data and use HL7 messages to communicate. We use Mirth to route HL7 messages to the trial application, which generates patient visits records that are used during the patient assessment. In the first few months of deployment the Mirth component of this system had successfully processed over 100K of HL7 messages, and correctly routed the ones relevant to the trial for storage in a local database. These results further show that Mirth Engine is a robust solution for the purpose of integrating a CDSS with existing HIS systems.

5. Conclusions

In this paper we showed that one can develop a complex CDSS that supports various tasks involved in clinical decision-making process, i.e. data collection, diagnosis formulation, treatment planning and provision of clinical evidence. To the best of our knowledge, MET3 is the first system to implement such a diverse and complimentary set of functionality in an integrative manner.

MET3 can be used at the point of care on desktop computers and mobile devices. Moreover, it supports HL7 standard to communicate with an external HIS. In a successful pilot evaluation we tested system's functionality and performance in a simulated emergency department (ED) environment, while underlying HL7 integration technology has been in use as part of a clinical trial in a real ED setting since January 2009.

One of the important lessons learned from this exercise is that such systems can be successfully implemented using the existing open source and free software. We believe that MET3 shows that such a technology is strong alternative to closed source and proprietary alternatives used by commercial CDSS providers. However, in deciding what OSS among multiple alternatives to use, one should take into account the nature of the project. Industry and community supported OSS have a tendency to be maintained current.

References

1. Patel VL, Kaufman DR, Arocha JF. Emerging paradigms of cognition in medical decision-making. J. Biomed. Inform.35(1):52-75 (2002)
2. Berlin A, Sorana M, Sim I. A taxonomic description of computer-based clinical decision support systems. J. Biomed. Inform. 39:656-667 (2006)
3. Wright A, Sittig DF. A four-phase model of the evolution of clinical decision support architectures. Int. J. Med. Inform. 77:641-649 (2008).
4. A Moreno. On the Evolution of Applying Agent Technology to Healthcare. *IEEE Intelligent Systems*; 21 (6): 8-10 (2006).
5. Farion K, Michalowski W, Wilk S, O'Sullivan D, Rubin S, Weiss D. Clinical decision support system for point of care use: ontology driven design and software implementation. Methods Inf. Med. 2009:(in press).
6. Garcia-Ojeda JC, DeLoach SA, Robby, Oyenan WH, Valenzuela J. O-MaSE: a customizable approach to developing multiagent development processes. In: Luck M, editor. *Agent-oriented software engineering VIII*. Berlin: Springer-Verlag;. 1-15 (2007).
7. Sz. Wilk, W. Michalowski, D. O'Sullivan, K. Farion, S. Matwin: Engineering of a Clinical Decision Support Framework for the Point of Care Use. AMIA Annu Symp Proc. 6:814-818 (2008).