MEDINFO 2010
C. Safran et al. (Eds.)
IOS Press, 2010

doi:10.3233/978-1-60750-588-4-811

Integration of Workflow and Rule Engines for Clinical Decision Support Services

JaeHoon Lee[a], JeongAh Kim[b], InSook Cho[c], Yoon Kim[d]

[a] *Division of Industrial Engineering, Ajou University, Suwon, Korea*
[b] *Division of Computer Education, Kwandong University, Gangneung, Korea*
[c] *Nursing Department, Inha University, Incheon, Korea*
[d] *Medical Department, Seoul National University, Seoul, Korea*

Abstract

Although recent studies have suggested the feasibility of integrating workflow and rule technology in a Clinical Decision Support System (CDSS), their implementation has not been verified yet. This paper proposes a knowledge engine which integrates workflow and rule engine as a tool for interpretation and execution of computer interpretable clinical guidelines. The objective of this paper is to validate its feasibility in two perspectives: clinical knowledge coverage and execution performance. The two open source engines which were selected and integrated were chosen due to their reliability and consistency. Implementation of workflow and rule engine integration has shown that the integrated knowledge engine (uBrain) is an effective CDSS for the execution of clinical guidelines.

Keywords:

CDSS, Workflow, Rule engine

Introduction

Over the decades, many researchers have tried to develop and implement effective CDSSs. The basic concept of a CDSS is to provide an intelligent tool which can provide clinicians or patients with clinical knowledge and patient-related information to enhance patient care experience. In addition to the revolutionary advances in the field of information technology, recent CDSSs are based on artificial intelligence or knowledge engineering technologies. Furthermore, the architecture of CDSSs has changed from standalone systems to standard-based and service-oriented systems. These advances demonstrate that future research and trends of CDSSs development will aim to maximize the interoperability through separation of their components such as data, rules, processes and services.

Recently, CDSS development basically employs two main components: 1) a clinical knowledge base for what to dispose and 2) an inference engine for how to dispose. The studies on various knowledge bases mainly have focused on computerized and standardized representation of clinical knowledge. One of the most widely used formats is the Computer Interpretable Guidelines (CIGs) which is used as a generic template to facilitate the translation of guidelines from their published formats into computer interpretable algorithms.

The inference engine of a CDSS performs such functions as interpreting and executing the guidelines encoded in the specific representation formats. There are generally two approaches in developing a clinical inference engine. One approach is to develop an engine for a specified clinical guideline format. Well-known guideline formats generally contain their own execution engines [1]. The other approach is to adopt common knowledge tools into clinical decision support. Since the core logic of clinical guidelines consist of complicated rule sets, commercially available rule engines have been adopted on a wide-scale basis. Several researches have reported that commercial rule engines can be an execution engine for clinical knowledge [2].

Although rule engines have been verified as having an alternate core function as an inference engine, they cannot possible provide complete coverage for clinical guidelines. The external part of clinical guidelines consists of triggers, actions and decision-makings. Recent research has focused on reinforcing functions that control the main flow of guidelines, invoking rule execution and interfacing with local applications. The rule invokers were originally embedded in the CDS applications, but as the coverage of clinical guidelines became wider, the necessity to make them independent and to standardize them has increased.

Recent studies have suggested the adoption of the workflow concept into clinical guidelines, particularly in terms of pattern coverage, execution and knowledge representation [3-5]. It is very promising because as the coverage of knowledge bases in CDSS becomes wider spread, so does the original use of workflow as a business process management which can be adequately utilized in clinical processes. The clinical guideline can be effectively separated into the combination of workflow and rule models. Therefore in this paper, we integrate workflow and rule engines as a knowledge engine, and validate its availability.

In this study, a workflow engine is used as an interpreter of the guideline process model, and a rule engine is used as an inference engine. The workflow definition model represents the logical flow of the guideline model, and the rule model implies unit logics such as clinical concepts, constraint, and criterion. When a service is triggered, the workflow engine invokes the rule engine for execution generating actions according to the results. In this study to increase the integrity in the execution of guidelines, we integrated two open source engines which were verified in a decision supporting system for management information system domain.

The verification of the proposed work is done at two critical points of the CDSS: 1) accuracy of knowledge processing and 2) execution performance. The accuracy of the proposed system can be verified using test cases. The representative clinical guideline models and their test cases can measure the accuracy and reliability of the knowledge engine. Performance was measured as the availability of practical implementation at local institutions. The performance criteria included were service response time, data fetching time, and durability against stress. Acceptable performance and time parameter for a CDSS is two transactions per second [2].

The results of the study show that the elements of the clinical guideline were correctly executed and made within the parameter mentioned above. Additionally, the integrated knowledge engine provided wide coverage and the expression power of the integrated knowledge engine can be extended to other formats for clinical guideline formats. The contribution of this paper is that it verified the availability of an integrated engine using workflow and rule models.

Materials and Methods

Workflow management system and rule engine

A workflow definition model and its management system are widely used in modeling and automation of business processes. A workflow management system employs various tools to support the entire life cycle of workflow from its de-sign to its execution and analysis [6]. The core function of a workflow management system is the enactment service which interprets the workflow definition model step by step, delivers each task to the actual workers and controls and monitors the status of the process. This software component, which acts as the workflow enactment service, is called the workflow engine

Since a workflow model can represent diverse patterns of logical flows such as data, resources, and controls, approaches to workflow for clinical guidelines have been adopted and tried. One of the more significant approaches was the comparison of two concepts in terms of their expression power [4]. These studies systematically analyzed and clustered the process patterns and control structures towards their ability to be automated. The conclusion of these studies showed that CIG modeling languages are remarkably close to traditional workflow languages from the control flow perspective, but cover many fewer workflow patterns. Consequently, workflow management systems may be suitable and applicable for clinical guideline applications [3].

A rule engine is commonly provided as a component of a business rule management system to execute rules. In any IT application, business rules are changed more frequently than the rest of the application code. Rule engines are the pluggable software components that execute business rules that have been externalized from application code. This allows business users to modify the rules frequently without the need of IT intervention and hence allows the applications to be more adaptable with dynamic rules.

Verification of knowledge coverage and performance

In order to apply workflow and rule model into a clinical guideline, the feasibility should be evaluated by measuring the similarity between the two concepts. The verification has two perspectives: 1) knowledge coverage and 2) physical performance. The former is related to the relevance of the proposed method, and the latter is about verifying the applicability for the real clinical field.

Generally speaking, clinical knowledge bases are very complicated and specified to domain experts so it makes it difficult to evaluate and compare with knowledge bases in different domains. One of the significant methods is to compare with generalized models. Recent research concentrates on the similarity of workflow models and CIGs in terms of patterns. The result of the comparison shows that CIG languages such as Abru, EON, GLIF3.5, and PROforma are very similar to process languages of workflow management systems although they do not make use of many of the workflow patterns in such systems [3].

Another dominant point for verification is the physical performance of a knowledge engine, and this may be the biggest benefit for adopting commercial engines. The factors which determine the performance of a knowledge engine are as follows: 1) delivery time of patient data to the engine, 2) response time to make actions when a large number of rule sets are loaded, 3) loading time of rule models. In case of an engine which is a pre-loading type, loading time is negligible [2].

Generally, the operation time of the CDS service is very short; less than a second. Therefore, in case of occurring redundant service requests, the system should endure the stress of multiple accesses to prevent waiting too long a time for the response. Another major bottleneck in service performance is delivery of the data to the engine. In order to minimize the number of round trips between a rule service and an external repository, the rule service should be primed with a large swath of patient data. Consequently, the CDS service system should satisfy not only providing quick response time but also avoid overloading adjacent systems.

Selection of engines

In order to select most suitable workflow and rule engines, the following elements are considered.

- Integrity: In order to achieve fast response and correctness of execution, the two engines should be easily integrated. The ingredients of integrity are the same programming language, fully object-oriented design, and simple and extensible interfaces.

- Reliability: The engines are required to contain industrial references in order to assure stable performance against physical stress particularly in practical uses in local institution.
- Extensibility: The framework of engines should be based on a well-known architecture (J2EE etc.) so that its components can be easily added or reconfigured.
- Open source: Because this study was involved from a national perspective, the proposed works should be non-profitable and open to the public.

Consequently, two open source engines; uEngine and BRAIN were selected. uEngine is a workflow engine which has advanced in convenient development of workflow activity types so that it can integrate the other modules with ease. BRAIN is a business rule engine based on an object rule model and <if then> rule expression. Object rule model defines the domain model and interface model. Domain model defines the domain scope and standard value or basis for criterion. Interface model defines the variables to be compared with standards. Various criteria were expressed in if-then statements.

The two engines were fully developed in the Java language platform and based on object-oriented design patterns. These engines were already verified in decision supporting module of management information applications.

Development

The integrated knowledge engine (uBrain) was designed to take a layered approach to partitioning the functions which are provided by the components. Basically, a clinical guideline contains diverse elements which have their own features and functions. The most closed part to users is related to clinical actions such as retrieving patient data, triggering interference, making recommendations or notifications. These kinds of actions can be separated and represented effectively in workflow model. The workflow engine employs basic activity types which have same functions to clinical actions, so a knowledge author can design clinical workflow with them.

The part of clinical guideline for inferences was separated as rule models, and conducted by a rule engine. Logical elements of clinical guidelines such as variables and their values, presence of a status, and composite logics can be interpreted as rule functions in the rule engine. The rule engine reads input data sets at once, executes, and returns the result sets. The role of the workflow engine is activation of rules, delivery of input data sets and the execution results.

The external feature of uBrain was developed as a client-server system which is based on the assumption that there exist many physically distributed clients. Also, standardized clinical guidelines for each disease were defined and registered in the service registry so that a client can find and invoke for the CDS service for a specific type of guideline. Consequently, uBrain defines the business logic of a guideline model, the local EMR provides data, and the associated clinical applications will support the interactions between the users and a guideline implementation system. This architecture is based on service oriented architecture.

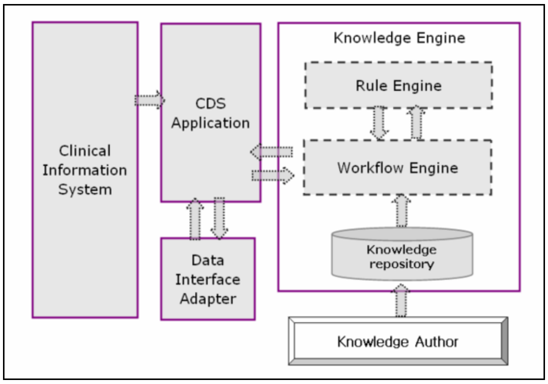

Figure 1- Proposed CDSS architecture

The overall architecture of uBrain is shown in Figure 1. The system architecture was designed to provide flexibility for integration with clinical information system to be taken by a local institution. Clinical guidelines may be stores in a knowledge repository after being encoded by a knowledge author. The engine retrieves the knowledge from the repository according to a request (event) from the CDS application. To load the patient data from the clinical information system to the knowledge engine, the CDS application should fetch the patient data in run time through data interface adapter (DIA).

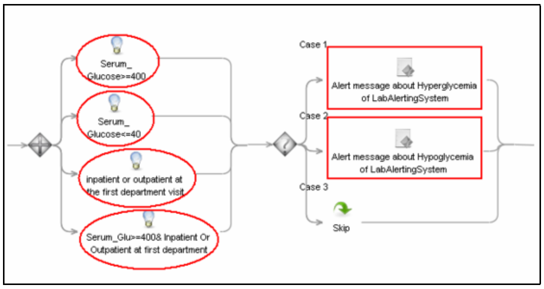

Figure 2- Integration of workflow and rule model

Figure 2 shows the integrated feature of workflow and rule as a guideline model. The rule activity icons (in circles) indicate the invocation points for rule execution, and the result values may be stored at the variables in the process model. At the decision-making points (branching points), the workflow model selects a path to execute based on the rule execution results and activates actions (activity icons in squares), and finishes the process.

Results

Stress test of the integrated engine

To validate the proposed framework, example clinical guidelines were selected. The guidelines are to validate performance, knowledge coverage and extensibility. Lab alerting; a simple type of clinical guideline is first implemented to see the response time of the engine. Lab alerting consists of seven types of sub guidelines which contain one or more rule sets. The stress test is under the assumption that there is a server for guideline service and multiple clients, service requesters. The evaluation points are 1) how the response time decreases according to multiple accesses and 2) how the engine is affordable against overload. The results are shown in Table 1.

Table 1 – Execution results for load test

No. of connections	No. of test cases	Processing time (msec)	
		Average	Std. dev.
1	30	48.3	4.9
2	60	111.3	67.5
3	90	1,368.1	500.9
5	150	4,303.5	562.4
10	300	6,414.3	2,746.6
20	600	14,154.5	7,538.7
30	900	20,823.7	23,638.5

The stress tests were conducted under various conditions. The number of simultaneous requests increased from 1 to 30. In table 1, the average processing time increases proportional to the amount of requests. Standard deviation values increases more sharply than average processing time. This implies that the stressed environment influences the quality of service in both indicators of means and variances. In particular, the largest stress condition makes the variance of processing time much higher so that sometimes users give up waiting for the response.

Performance test in real usage

One of the critical issues in a real environment for CDS uses is the fetching time for the patient data. A DIA was also developed to efficiently retrieve local patient data and make an input data set for the knowledge engine. A test bed which includes a DIA was established based on a clone of the local EMR database which contains real patient data. In all, 323,445 test cases were generated from the database, and the accuracy of uBrain execution results compared to the original lab results was 100%. The performance results for the processing time are shown in Table 2. The results represent that the performance of the system highly depends on the amount of data and rules which should be disposed.

Table 2 – Execution results with test cases

Test name	No. test cases	No. Alerts	Average processing time (msec)		
			DIA	Engine	Total
CBC	39,893	41	137	55	192
Glucose	44,494	229	276	102	378
HCT	62,764	764	457	36	493
Rh typing	10,439	38	787	55	842
WBC	62,612	73	400	32	432
Sodium	51,405	156	446	48	494
Potassium	51,838	349	165	53	218
Total	323,445	1,650	346	52	398

Coverage test

A hypertension guideline was modeled as a case for the complicated knowledge base. It consists of 247 rules and encoded in Sage format. The guideline consists of three recommendation sets; a main guideline which has branches according to existence of diabetes mellitus, and two sub guidelines which make actions for recommendation according to the rule execution results. 201 representative test cases were selected from clinical experts, and the results shows that the integrated engine can cover complicated types of guidelines.

Discussion

The dominant trend in CDSS development is separation of clinical knowledge and reasoning, and their independent disposing. The workflow engine and rule engine were physically integrated in architecture, but independently operated by its own roles. One of the main obstacles which make it hard for CDSSs to spread widely is the hardness to identify or separate the knowledge from other application functionalities and, separation of these components may contribute to increased reliability and maintainability of CDS services.

In the stress test, the engine showed excellent performance and can endure a few simultaneous requests. This will be a factor which needs to be determined to assure the capacity of a CDS server in restricted resources. In run time test, the processing time was found to be divided into three elements: workflow engine processing, rule engine processing, and DIA data fetching. The workflow processing time was almost fixed around 20 micro seconds. The rule processing time was proportional to the number of rule sets. And the data fetching time depends on how many elements should be prepared to execute knowledge engine.

The results of the performance analysis indicate the direction of future works. The number of rule operations is a variable factor to determine the entire processing time so it should be minimized. Some parts of this can be accomplished by caching for the similar or same conditions. Data fetching operation was still the biggest bottleneck, but this study verifies that the strategy of ‘fetching data before execution at a time’ is feasible and promising.

Conclusion

The integration of workflow and rule engines is successful in the perspectives of architectural efficiency and availability in clinical domains. The knowledge coverage of the integrated engine was verified by translating and executing Sage based guidelines. Also the engine shows acceptable performance in practical use of CDS services through generation and execution of test cases. Future work is expected utilizing the extensibility and applicability of the proposed methodologies.

References

[1] Wang D, Peleg M, Tu SW, Boxwala AA, Ogunyemi O, Zeng Q, Greenes AR, Patel VL, and Shortliffe EH, Design and implementation of the GLIF3 guideline execution engine, Journal of Biomedical Informatics, 2004: 37; pp. 305–318.

[2] Goldberg HS., Vashevko M, Postilnik A, Smith K, Plaks N, and Blumenfeld BM, Evaluation of a commercial rule engine as a basis for a clinical decision support service, AMIA 2006 Symposium Proceedings; 2006; pp. 294.

[3] Sedlmayr M, Rose T, Röhring R, and Meister M, A workflow approach towards GLIF execution, 17th European Conference on Artificial Intelligence (ECAI): 2006: pp. 3.

[4] Mulyar N, Aalst WMPV, and Peleg M, A Pattern-based Analysis of Clinical Computer-interpretable Guideline Modeling Languages, Journal of the American Medical Informatics Association, 2007; 14 (6); pp. 781-787.

[5] Heard KM, Huang C, Noirot LA, Reichley RM, and Bailey TC, Using BPEL to define an executable CDS rule process, AMIA 2006 Symposium Proceeding; 2006; pp. 947.

[6] Aalst WMPV, Hofstede, A. H. M. t. and Weske, M., Business process management; a survey, Lecture Notes in Computer Science, 2003; 2678; pp. 1-12.

Address for correspondence

Jaehoon Lee: shamvala@gmail.com