

Design, Implementation, Use, and Preliminary Evaluation of SEBASTIAN, a Standards-Based Web Service for Clinical Decision Support

Kensaku Kawamoto and David F. Lobach, M.D., Ph.D., M.S.

Division of Clinical Informatics, Department of Community and Family Medicine
Duke University Medical Center, Durham, North Carolina

ABSTRACT

Despite their demonstrated ability to improve care quality, clinical decision support systems are not widely used. In part, this limited use is due to the difficulty of sharing medical knowledge in a machine-executable format. To address this problem, we developed a decision support Web service known as SEBASTIAN. In SEBASTIAN, individual knowledge modules define the data requirements for assessing a patient, the conclusions that can be drawn using that data, and instructions on how to generate those conclusions. Using standards-based XML messages transmitted over HTTP, client decision support applications provide patient data to SEBASTIAN and receive patient-specific assessments and recommendations. SEBASTIAN has been used to implement four distinct decision support systems; an architectural overview is provided for one of these systems. Preliminary assessments indicate that SEBASTIAN fulfills all original design objectives, including the re-use of executable medical knowledge across diverse applications and care settings, the straightforward authoring of knowledge modules, and use of the framework to implement decision support applications with significant clinical utility.

INTRODUCTION

Adults in the U.S. receive only about half of recommended care,¹ and up to 98,000 Americans die each year as the result of preventable medical errors.² One of the most promising strategies for addressing this crisis in care quality is the use of computer systems to deliver patient-specific assessments or recommendations to clinicians.³ However, the availability of such decision support capabilities remains limited in most U.S. health care facilities.

While multiple factors have contributed to the limited use of decision support systems, an important factor has been the difficulty of re-using medical knowledge encoded in a machine-executable format.⁴ In attempting to overcome this problem, knowledge engineers have generally taken two approaches.⁴ As one approach, systems such as PRODIGY,⁵ SAGE,⁶ and First DataBank's Drug Information FrameworkTM⁷ provide access to their executable knowledge base using standard application programming interfaces. As a second approach, methods including GLIF3,⁸ GEM,⁹ and the Arden Syntax¹⁰ encode knowledge using a common

formalism, so that encoded rules can be consistently interpreted by system-specific interpreters.

Despite these significant efforts, a dominant framework has not emerged for sharing executable medical knowledge, due in part to the following challenges. First, some formalisms, such as the Drug Information FrameworkTM,⁷ focus on specific knowledge domains and are not extendable to other domains. Second, many formalisms are designed for use in specific types of decision support applications and are difficult to adapt for use in other types of applications. Third, many formalisms are difficult to understand due to their conceptual complexity. Fourth, many existing architectures require the client to provide the decision support engine with relatively unfettered read or write access to its clinical database. Finally, many existing methods require significant investments in infrastructure, such as a system-specific compiler¹¹ or a virtual medical record (vMR) implementation.⁶ In this paper, we describe the design, implementation, and use of a decision support Web service that overcomes each of these challenges.

DESIGN OBJECTIVES

Our primary design objectives were: (1) provide "write once, run anywhere" portability for executable medical knowledge; (2) allow both simple and complex knowledge modules to be authored in a straightforward fashion; and (3) minimize the effort required to understand and use the framework.

In addition, we pursued the following secondary objectives: (1) use available standards where appropriate; (2) leverage available tools; (3) facilitate knowledge maintenance; (4) support comprehensive testing; (5) optimize execution performance; and (6) enable clinically significant decision support systems.

DESCRIPTION OF FRAMEWORK

In order to meet the objectives just described, we designed a framework known as SEBASTIAN, an acronym for System for Evidence-Based Advice through Simultaneous Transaction with an Intelligent Agent across a Network. As indicated by the name, SEBASTIAN interacts synchronously with client software applications to deliver evidence-based decision support over the Internet.

Overall architecture. SEBASTIAN is implemented as a Web service, in which software functionality is provided over the Internet and extensible markup language (XML) messages are

used to communicate with client systems.¹² The framework is implemented as a Java servlet and is hosted by the Apache Tomcat servlet container. An overview of the SEBASTIAN architecture is provided in **Figure 1**. The salient features of this architecture are briefly described below.

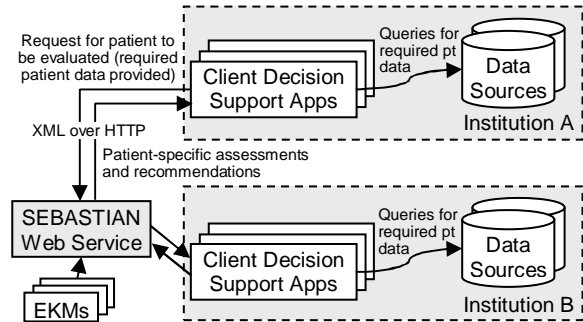


Figure 1. Overview of SEBASTIAN architecture. EKM = Executable Knowledge Module.

Patient information model. SEBASTIAN uses a patient information model based on the Health Level 7 (HL7) Reference Information Model (RIM),¹³ and concepts are identified using standard vocabularies included in the National Library of Medicine's Unified Medical Language System (UMLS).¹⁴

Executable Knowledge Modules. Medical knowledge in SEBASTIAN is captured in XML documents known as Executable Knowledge Modules (EKMs). Each module specifies the data requirements for assessing a patient, the patient-specific conclusions that will be returned by the module, and the logic that will be utilized to generate the conclusions using the specified patient data.

Services provided. SEBASTIAN uses the knowledge encoded in the EKMs to offer several services to client decision support applications. Because SEBASTIAN is a Web service, all services can be accessed by sending XML requests over HTTP. The core service offered is a patient evaluation service, in which patient data elements are received as the input and machine-interpretable decision support results are returned as the output.

Use of SEBASTIAN by clients. To receive patient-specific advice from SEBASTIAN, client decision support systems first retrieve the required patient data elements from one or more data sources. Then, the client sends the patient information to SEBASTIAN, receives structured decision support results in return, and uses those results as desired.

This next section describes each of these aspects of the SEBASTIAN framework in greater detail.

Patient information model. SEBASTIAN uses a standards-based patient model based on the HL7 RIM. Patients are modeled as entities described by demographic and "act" data, where an act refers to any act or service constituting health care services, such as an encounter, diagnosis, or procedure.¹⁵

A patient's demographic data consists of gender, race(s), age, and birth date. In addition, a patient can participate in health care acts. Each act possesses the attributes listed in **Table 1**. The encounter ID associated with an act is sometimes needed to identify which acts occurred as part of the same encounter, and the provider ID associated with an act is sometimes needed to identify which acts were performed by the same provider. The use of UMLS vocabularies is encouraged when identifying an act. Act subclasses may contain class-specific attributes, such as a value for an observation or a goal.

Table 1. Attributes common to all acts.

Attribute	Description and Example
act type	The type of act (e.g. encounter diagnosis)
identifying code	Code that identifies the act (e.g. "250.01" in ICD9CM version 2005)
act start / end time	The time when the act started / ended (e.g. 2005-03-15T00:00:00)
encounter ID (optional)	The identifier of the encounter associated with the act (e.g. Duke Encounter ID, 12345)
provider ID (optional)	The identifier of the provider associated with the act (e.g. Duke Provider ID, 67890)

At present, the patient model includes the following act types: encounters, encounter diagnoses, problem list entries, procedures, observations, goals, medication orders, program enrollments, provider assignments, invoices, and past EKM evaluation results. EKM results are the primary objects returned to client systems following the evaluation of a patient. In addition to the basic act attributes, EKM results possess the attributes listed in **Table 2**.

Table 2. EKM result attributes.

Attribute	Example
result code	003
result code short description	Diabetes_last_hgba1c_bet_5_and_6_mo
result code long description	Patient with diabetes, last HgbA1c test 5-6 months ago
patient-specific assessment	Patient with diabetes and last HgbA1c 5 mo and 21d back (8.5% on 10/15/04).
patient-specific recommendation	Recommend HgbA1c test to maintain test frequency at q6mo.
result parameters	lastTestDate → 2004-10-15; lastTestValue → 8.5; daysUnitDue → 9

Executable Knowledge Modules. SEBASTIAN encapsulates knowledge in XML-based Executable Knowledge Modules (EKMs). EKMs consist of maintenance, library, knowledge, and logic sections. Most module sections are edited using a functionally rich Microsoft InfoPath™ form. This form includes extensive terminology support. For example, authors can translate between vocabularies when specifying a clinical concept using multiple vocabularies.

Maintenance and library sections. The maintenance and library sections are very similar to the corresponding sections of Arden Syntax Medical Logic Modules.¹⁰ The maintenance section consists of general maintenance information, such as the title,

identifier, version number, and authors, while the library section consists of bibliographic information, such as keywords and references.

Knowledge section. The knowledge section defines the data requirements for evaluating patients using the module. Demographic requirements are specified by indicating whether gender, race, age, and/or birth date are required. Act requirements are specified by indicating the type of act required, the codes that identify the required act (codes can be specified in multiple vocabularies), how far back to look for that data, whether the associated encounter ID is required, and whether the associated provider ID is required. Of note, one module can require the results of another module. In such cases of “nesting,” the pre-requisite module is evaluated first and its results are made available to the dependent module.

The knowledge section also defines the machine-interpretable results that will be returned to the client. Each potential result code is accompanied by a description, and the result parameters that will be returned (e.g. lastTestDate, daysUntilDue) are also specified and described (**Table 2**).

Logic section. The logic section specifies how the patient data provided by the client will be used to derive the EKM results promised in the knowledge section. To do this, SEBASTIAN generates a corresponding Java class for each knowledge module in its repository. Within these Java classes, the module author can access the required data elements as native Java objects. For example, if a module requires encounter diagnoses for diabetes from the past 12 months, SEBASTIAN will auto-generate an array that will be populated at runtime by EncounterDiagnosis objects that meet the specified selection criteria. Given this mechanism for accessing required data elements, the author can use standard programming methods to generate and return an EKM result object conforming to the output specification described in the knowledge section.

Because the decision logic is expressed in native Java, authors can edit logic rules using powerful and widely available Java programming environments. Moreover, authors have significant latitude in deciding how to produce the required results given the available patient data. For example, authors can create utility classes to handle common operations, query medical knowledge stored in databases, or invoke external decision support engines.

Services provided. The primary service offered by SEBASTIAN is a patient evaluation service, in which client systems obtain machine-interpretable decision support results for specified knowledge modules. In requesting this service, the client can submit the minimum data set required by the knowledge modules, or it can submit a superset of the

required data (e.g. all available data). To facilitate testing, the framework permits clients to specify a time other than “now” for the time SEBASTIAN believes it is when evaluating a patient. A sample patient evaluation request is shown in **Figure 2**.

```
<?xml version="1.0" encoding="UTF-16"?>
<decisionEngineServiceRequest>
  <client id="ABC" password="XYZ"/>
  <request_ptEval-EkmsSpecified>
    <evalTime>Now</evalTime>
    <ekmToEvaluateList>
      <ekmToEvaluate id="diabetes_no_hgba1c" version="1.00"/>
    </ekmToEvaluateList>
    <decisionEnginePatient>
      <demographicData/>
      <actList>
        <encounterDiagnosisList>
          <encounterDiagnosis>
            <code voc="ICD9CM" vocVer="2005">250.01</code>
            <time start="2004-04-15" end="2004-04-15"/>
          </encounterDiagnosis>
        </encounterDiagnosisList>
      </actList>
    </decisionEnginePatient>
  </request_ptEval-EkmsSpecified>
</decisionEngineServiceRequest>
```

Figure 2. Sample patient evaluation service request.

In addition, SEBASTIAN offers the following auxiliary services: (1) a service that identifies the knowledge modules that meet client search criteria; (2) a service that provides descriptions of selected modules, including descriptions of the results that will be returned following patient evaluation; and (3) a service that specifies the consolidated data requirements for a set of knowledge modules.

Use of SEBASTIAN by clients. To use SEBASTIAN, the developer of a client system first identifies the set of knowledge modules that will best meet her application needs. Second, the developer verifies that she has access to the data required by the selected modules. Third, the developer ensures that when decision support functionality is needed, the client system will: (a) retrieve the required patient data, (b) send a request to SEBASTIAN to evaluate the patient using the relevant modules, (c) parse the EKM results that are returned, and (d) process the decision support results to meet user needs.

CURRENT USES OF SEBASTIAN

SEBASTIAN has thus far been used to implement four decision support systems. The four systems are briefly described below, and an architectural overview is provided for one of the systems.

Systems for population health management.

Three of the systems support the population health management of approximately 16,000 Medicaid beneficiaries in Durham County, North Carolina. On a nightly basis, SEBASTIAN is used to identify issues of concern within this population, such as overdue preventive care needs, poorly controlled diabetes, and frequent use of the emergency department for non-urgent care. These care needs are identified using Medicaid billing data, encounter data

obtained from two hospital information systems and two clinic management systems, and health risk data collected directly from patients using touch-screen kiosks. Based on the inferences made by SEBASTIAN, one system provides patients' primary care clinics with reports that list the patients most in need of services, along with identified care needs and recommended actions. A second system emails alerts to appropriate health care providers regarding care issues requiring follow-up, and a third system generates care reminder letters for patients in English and, when applicable, in Spanish.

Outpatient diabetes reminder system. We have also used SEBASTIAN to implement a Diabetes Reminder System (DRS) at the Duke Family Medicine Center (Figure 3). When a patient with diabetes checks into the clinic, the intake nurse accesses the DRS Web page and requests a diabetes care reminder sheet for the patient (arrow 1). The DRS controller application then requests all available information on the patient from the Duke Common Data Repository (CDR) through a Web service interface (arrow 2). The CDR Web service provides data on prior encounter diagnoses using ICD9 codes, procedures using CPT4 codes, past and scheduled encounters using Duke-specific codes, allergies using First DataBank codes, and laboratory results using a vendor-specific coding scheme. The DRS controller also retrieves data from a local, DRS-specific database that uses SNOMED CT to store data not otherwise collected in a coded format (e.g. whether a microfilament foot exam was done).

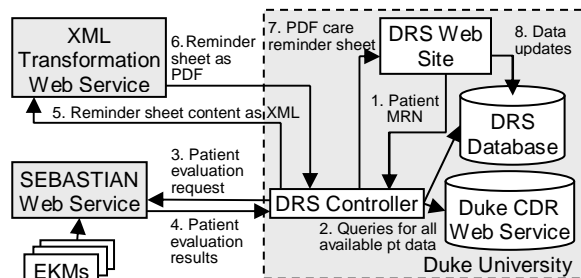


Figure 3. Overview of Diabetes Reminder System (DRS). MRN = medical record number. CDR = Common Data Repository. EKM = Executable Knowledge Module.

After consolidating the retrieved data into a single patient object, the DRS controller makes a request to SEBASTIAN to evaluate the patient using 13 EKM for diabetes management (arrow 3). Upon receiving the EKM results (arrow 4), the controller generates an XML document representing the contents of a reminder sheet. This XML document is passed to an XML transformation Web service (arrow 5), which uses an XSL stylesheet to convert the XML content into a PDF document (arrow 6). This PDF reminder sheet is streamed to the Web browser (arrow 7), so that the nurse can print out the sheet and attach it to

the patient's chart for clinician review. The reminder sheet consists of a section with relevant previous values, a section where clinicians can enter data not otherwise collected in a coded format, and a section that provides decision support on needed care (Figure 4). Any data updates are recorded by clinic support staff through the DRS Web site (arrow 8).

Focus	Status	Last Done	Recommendation
BP measurement	DUE NOW	06/05/03	every visit
Weight measurement	DUE NOW	06/05/03	every visit
Visual foot exam	DUE NOW	09/05/02	every visit
Foot microfilament	DUE NOW		every 6 months
HgbA1C	Not due	06/05/03	every 6 months
Urine microalb/cr ratio	Not due	12/20/02	annual, unless on ACE inhibitor or ARBs
Total cholesterol	Not due	12/20/02	annual
LDL cholesterol	Not due	12/20/02	annual
Ophthalmologic exam	Not due	06/30/03	annual
Influenza vaccine	DUE NOW	10/22/02	every fall, unless egg allergic
Pneumococcal vaccine	Not due	01/20/00	once, revacc. if >= 65yo & vacc'd before
Aspirin therapy (81 mg)	CONSIDER	not listed	if >=40yo. or >=30yo w/card RF. No cou

Figure 4. Decision support section of care reminder sheet.

EVALUATION OF FRAMEWORK

Our experiences to date indicate that the SEBASTIAN framework fulfills all primary and secondary design objectives, as outlined below.

Assessment of primary objectives. With regard to *knowledge portability*, we have found it easy to re-use EKM across applications and care settings. For example, all four of the decision support systems just described use the same EKM for diabetes care, despite significant differences in functionality and data sources. Also, many aspects of the SEBASTIAN framework facilitate *knowledge authoring*, including the use of functionally rich Infopath™ and Java authoring environments, the ability to “nest” EKM, and the ability to create utility functions to handle common logic patterns. Finally, we believe SEBASTIAN is *easy to understand and to use*. The only client infrastructure requirement is an Internet connection, and the framework was designed to be as complex as necessary, but as simple as possible.

Assessment of secondary objectives. With regard to the secondary objectives, SEBASTIAN leverages appropriate standards (e.g. Web services, XML, HL7 RIM, and UMLS) as well as *existing toolsets* (e.g. Microsoft InfoPath™ and Java development platforms). In addition, SEBASTIAN facilitates *knowledge maintenance* by encapsulating executable medical knowledge into modules that are independent of application code, version controlled, tagged with meta-data, and maintained centrally on behalf of multiple client applications. Furthermore, *comprehensive testing* is supported by a clear input/output interface and the ability to evaluate a patient as if it was any time in the past or future, and *execution performance* is also quite good. For example, a request to process the 13 EKM used by the DRS takes ~350 ms to complete, even when a superset of the required patient data is provided (in this case, data on 274 acts). Finally, SEBASTIAN is being used to *implement decision support systems*

with significant clinical utility. For example, we have preliminary randomized controlled trial data indicating that the provider alert system described earlier significantly reduces the rate at which patients with asthma or diabetes returns to the ED within 30 days of an initial ED visit (11.8% vs. 31.7%, $p = 0.01$, $n = 109$ index ED visits) (unpublished data).

DISCUSSION

In this paper, we have described a Web services approach to clinical decision support that provides effective mechanisms for both encapsulating medical knowledge into a machine-interpretable form, and for making that knowledge easily accessible to various medical software applications operating in many different settings. We have also described the use of this framework by multiple decision support systems.

Strengths and limitations of our approach. Strengths related to our primary design objectives include: the ability to use the same knowledge modules across multiple applications and institutions; the ability to author knowledge modules in a straightforward manner; and the ease with which the framework can be understood and used. Similarly, strengths related to our secondary objectives include: the use of appropriate standards; the leveraging of available tools; the support provided for knowledge maintenance and for comprehensive testing; fast execution performance; and the ability to implement clinically significant decision support systems using the framework. Another strength of our approach is that it could allow for the centralized management of the executable medical knowledge used by an institution¹⁵ or region. Finally, our approach allows a knowledge module to invoke other decision support engines from within the logic section. Thus, this framework could serve as a common platform for delivering executable medical knowledge that has been represented using different formalisms.

One limitation of our approach is that its usefulness has not yet been validated at other institutions or for several important types of decision support applications (e.g. computerized physician order entry systems). Also, because the patient information model was kept as simple as possible, we did not include some potentially useful relationships between patient acts, such as the relationship between a patient's medications and the problems for which they were prescribed. Finally, we have not yet formally reconciled our patient information model with the HL7 Reference Information Model through a process known as harmonization.

Implications. Through our involvement in the HL7 Clinical Decision Support Technical Committee and the HL7 Services Specification Project, we have come to realize that there is growing interest among

academic, commercial, and government stakeholders for establishing a common services framework for clinical decision support. We believe that the approach presented in this paper can serve as a useful foundation in this endeavor to specify a common architecture for authoring and delivering executable medical knowledge.

Future directions. In moving forward, we plan to work closely with our colleagues in HL7 to help establish a standard services framework for clinical decision support. In addition, we plan to use SEBASTIAN in other operational decision support applications to further demonstrate its flexibility and usefulness. We speculate that this approach will prove useful to others in their efforts to deliver clinical decision support at the point of care.

ACKNOWLEDGEMENTS

This research was supported by NIH grants T32-GM07171 and F37-LM008161; AHRQ grants R01-HS10472, R01-HS-015057, and R03-HS10814; and HRSA grant H2ATH00998. We thank Allen Mayers, Garry Silvey, and Jennifer Macri for their invaluable advice throughout the development process.

REFERENCES

1. McGlynn EA, Asch SM, Adams J, et al. The quality of health care delivered to adults in the United States. *N Engl J Med.* 2003;348:2635-2645.
2. Kohn LT, Corrigan JM, Donaldson MS, eds. *To Err is Human: Building a Safer Health System.* Washington, DC: National Academy Press; 1999.
3. Kawamoto K, Houlihan CA, Balas EA, Lobach DF. Improving clinical practice using clinical decision support systems: a systematic review of trials to identify features critical to success. *BMJ.* 2005;330:765-772.
4. Boxwala AA, Tu S, Peleg M, et al. Toward a representation format for sharable clinical guidelines. *J Biomed Inform.* 2001;34:157-169.
5. Johnson PD, Tu S, Booth N, Sugden B, Purves IN. Using scenarios in chronic disease management guidelines for primary care. *Proc AMIA Symp.* 2000;389-393.
6. Tu SW, Musen MA, Shankar R, et al. Modeling guidelines for integration into clinical workflow. *Medinfo.* 2004;2004:174-8.
7. First DataBank. Drug Information Framework Information Page. Available at: http://www.firstdatabank.com/integrated_content/drug_information. Accessed March 3, 2005.
8. Peleg M, Boxwala AA, Ogunyemi O, et al. GLIF3: the evolution of a guideline representation format. *Proc AMIA Symp.* 2000;645-649.
9. Shiffman RN, Michel G, Essaihi A, Thornquist E. Bridging the guideline implementation gap: a systematic, document-centered approach to guideline implementation. *J Am Med Inform Assoc.* 2004;11:418-426.
10. Pryor TA, Hripesak G. The Arden syntax for medical logic modules. *Int J Clin Monit Comput.* 1993;10:215-224.
11. Karadimas HC, Chaillolleau C, Hemery F, Simonnet J, Lepage E. Arden/J: an architecture for MLM execution on the Java platform. *J Am Med Inform Assoc.* 2002;9:359-368.
12. Cerami E. Top ten FAQs for Web services. Available at: <http://webservices.xml.com/pub/a/ws/2002/02/12/webservicefaq.html>. Accessed March 3, 2005.
13. Health Level 7. HL7 Data Model Development. Available at: <http://www.hl7.org/library/data-model/>. Accessed 3/3/05.
14. National Library of Medicine. Unified Medical Language System. Available at: <http://www.nlm.nih.gov/research/umls/>. Accessed March 3, 2005.
15. Greenes RA, Sordo M, Zaccagnini D, Meyer M, Kuperman GJ. Design of a standards-based external rules engine for decision support in a variety of application contexts: report of a feasibility study at Partners HealthCare System. *Medinfo.* 2004;2004:611-615.