Authors' accepted manuscript of the article published in
Proceedings of the 10th IEEE International Conference on Information Technology and Applications in Biomedicine
http://dx.doi.org/10.1109/ITAB.2010.5687715

Interoperability testing of integration profiles based on HL7 standard version 3

Jean-François Pambrun and Rita Noumeir

***Abstract*— With the deployment of regional and national Electronic Health Records, interoperability in healthcare has become especially crucial. Interoperability requires the use of standards such as the Health Level Seven (HL7), to enable the communication of electronic health information between heterogeneous systems. A framework for the implementation of standards, such the one provided by Integrating the Healthcare Enterprise (IHE), is also required to ensure semantic and functional interoperability. Various types of testing are also necessary: testing peers' ability to exchange data, to extract information from messages, and to react to the extracted information. In this paper, we propose a web infrastructure for interoperability testing of IHE profiles based on HL7 version three; we present the system architecture and describe how it can be extended to add new tests.**

I. Introduction

Health level 7 (HL7) is an application level standard related to electronic health information communication. For the past few years, the third version (v3) of the standard has been in development; it is based on web services and xml messaging. HL7 covers most of the health information exchange needs while digital imaging communication is handled by the Digital Imaging and Communications in Medicine (DICOM).

Standards are necessary to implement an Electronic Health Record (EHR). EHR enables access to patients relevant diagnostic information independently from the geographic location of the point of access or the institution where the information was initially gathered. It includes information such as observations, laboratory results, imaging reports, drugs, discharge summaries and allergies. EHR deployment is expected to improve the quality of care by enabling access to all relevant information at the diagnostic decision moment; it is also expected to improve the efficacy and efficiency of the overall healthcare system by improving productivity and by reducing duplication of tests. EHR requires interoperability between various heterogeneous distributed systems; therefore standards are essential because they ensure technical interoperability. However, semantic and functional interoperability require the definition of a framework for the implementation of standards. This lead to the creation of the Integrating the Healthcare Enterprise (IHE) initiative. IHE started in 1998 and was initially jointly sponsored by the Radiological Society of North America (RSNA) and the Healthcare Information and Management Systems Society (HIMSS). Currently, several other associations sponsor IHE that has expanded over several clinical domains and who benefits from broad international support. IHE follows an approach where care providers identify key interoperability problems they face, and where healthcare manufacturers and information technology experts agree upon an implementation that uses established standards to provide a solution for each identified interoperability problem. IHE integration profiles are documented and are publicly available as part of the IHE technical framework[1]. IHE's first involvement with HL7 v3 was with the IT Infrastructure Patient Identifier Cross-Reference (PIX) and Patient Demographic Query (PDQ) integration profiles.

Ensuring interoperability requires various types of testing: testing peers ability to communicate and exchange data; testing peers ability to parse and extract information from the successfully exchanged messages; and testing peers ability to react to the extracted information by changing information in their systems or by influencing subsequent workflow actions. Interoperability testing in healthcare is very new. To our knowledge, it started in 1999 with the first IHE connect-a-thon. It is a face-to face testing event where hundreds of systems from various healthcare manufacturers test their software implementation of IHE profiles by executing real clinical scenarios. By putting in place this testing event, IHE has been a pioneer in healthcare testing.

To prepare for the live-testing event, participants test their implementation with a software-testing tool, beforehand. This testing tool consists of documents and software that simulates communication partners, in addition to providing test data and test plans. Succeeding in all tests for a specific actor in a specific integration profile is required to participate in the face-to-face testing event, and to subsequently participate in demonstrations. Several live-testing events are taken place around the world every year. Even though IHE testing early purpose was to support educational demonstrations, its uniqueness, its technical team experience and the large number of systems tested, have contributed to propel this event to the level of the de-facto testing in healthcare.

In this paper we present the HL7 v3 testing software used to test the IHE PIX and PDQ integration profiles. We describe the software architecture and functionalities, and discuss their extensibility and limitations.

II. Overview of the PIX and PDQ architecture

A clinician may need patients diagnostic information that was gathered in a different hospital. Because the same patient

This work was supported in part by the Natural Sciences and Engineering Research Council of Canada and by a grant from Canada Health Infoway.

J. Pambrun and R. Noumeir are with the Department of Electrical Engineering, École de Technologie Supérieure, University of Quebec, 1100 Notre-Dame West, Montreal, Quebec, Canada, H3C 1K3 rita.noumeir@etsmtl.ca

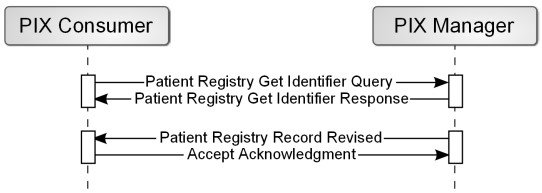

Fig. 1. The process flow of a PIX query.

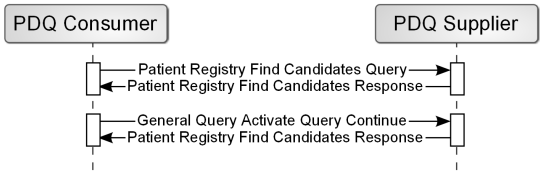

Fig. 2. The process flow of a PDQ query.

is usually identified with different identifiers in various domains, mapping all these identifiers together is needed. The PIX integration profile addresses this need. A PIX consumer can query a PIX manager with a patients identifier within a specific domain and requests the patients identifier within another domain. PIX Managers can also notify consumers when a record has been merged. Those transactions are based on HL7 v3 standard; they are respectively Get Patient Identifier and Record Revised as depicted in Fig. 1.

On the other hand, the PDQ integration profile addresses the need of querying for demographic information and identifiers of a specific patient, knowing only a subset of demographic information about that patient. For example, a PDQ consumer issues a query to a PDQ supplier using patients name and birth date, and asking for patients identifiers. The query transaction is based on HL7 v3 standard; it is Find Candidates Query as depicted in Fig. 2. Query Activate Query Continues is used when multiple response messages are necessary to retrieve all matching records.

III. Communication Infrastructure

HL7 v3 messages are encoded in XML and can be sent using web services technologies. Therefore, each server would implement a web service that is specified with a Web Services Description Language file (WSDL) [2]. A WSDL file describes every supported interaction including request and response messages as well as the communication methods that can be handled by the endpoint. The XML message is wrapped in a Simple Object Access Protocol (SOAP) envelope and can be sent using a protocol such as HTTP. The SOAP [3] envelope should also contain a standardized web service addressing (WS-Addressing) segment that can be used for routing.

IV. Related Work

Testing can be either to validate conformance to standards or to validate interoperability. Conformance testing can be seen as a prerequisite to interoperability but does not guarantee it. Interoperability testing is a way to ensure that different systems can co-operate to perform a specified

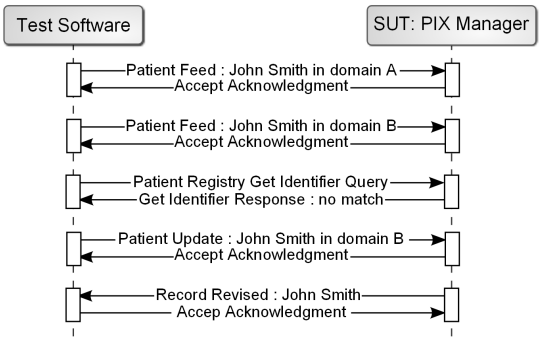

Fig. 3. Test case sequence diagram for Update Notification by a PIX Manager

business sub-process. By exchanging messages that conform to one or multiple standards, the co-operating systems react to the received information by changing their internal data or state, or by triggering actions or message exchanges with the same peer or others.

The common approach to healthcare system testing is the Upper-Lower Tester [4]. Using this approach, the system under test communicates with a Lower Tester via a specific communication protocol, and with the Upper Tester being the user or the business application under test. This approach is widely used but suffers from the lack of business level testing that can be achieved with the actor based testing. An actor is an application that has specific business responsibilities and communicates with other actors according to constrained messages that all cooperating actors agree on. Actor based testing allows the construction of an environment that simulates the real operation environment in which the system under test is expected to operate. Actor based testing requires the business process to be specified in terms of profiles between actors. Our testing system is based on an actor testing approach. Testing can take place at the communication level, the content level by validating the content of messages, and at the business level by validating that the system under test has either changed its internal data or triggered data exchange with other parties. IHE Mesa testing tools [5] is based on actor testing. IHE Gazelle project, also an actor based approach, is a multi-organization work in progress effort that allows the testing of multiple systems.

Very few projects exist for interoperability testing in healthcare. In [6], a test framework is proposed to design and execute testing of HL7 communication, document, and business layers. The business layer is described in terms of scenarios that usually require the exchange of messages between two actors. This framework enables the fast and easy design of new tests.

V. Implementation

A. Test cases

The PIX and PDQ integration profiles define five actors: PIX manager, PIX consumer, PDQ supplier, PDQ consumer, and ID source. Each test case is designed to validate the

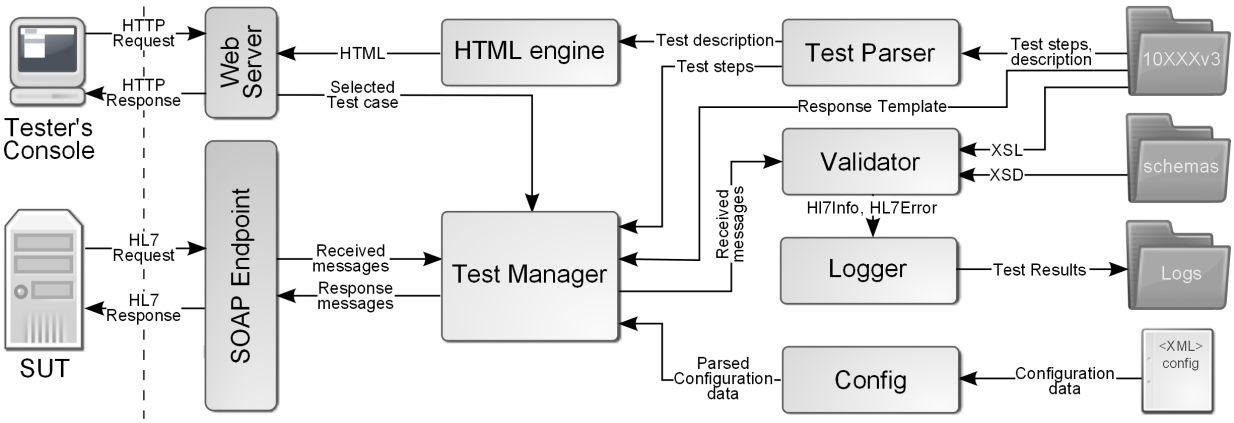

Fig. 4. System architecture.

interoperability of a specific actor, by testing specific transaction sequences, business logics, or functional behaviours. To conduct the testing, the test software plays the roles of all other actors that are required to perform certain tasks. For instance, when testing partial name queries performed by a PDQ supplier, the testing software plays the role of an ID source by registering multiple patients, and then plays the role of a PDQ consumer to perform a query.

All messages received from the system under test, including the acknowledgement message, are analyzed for well formed syntax and for content. This is possible because the database of the system under test is populated by the testing software since this later issues messages for the registration of patients; therefore, the test designer knows exactly what should be the content of a response message including the error cases (e.g. unknown patient error, unknown domain). Controlling the test data allows the validation of technical, semantics and functional interoperability.

Test cases for PIX manager and PDQ supplier are more complex and usually require multiple messages to validate functional interoperability. One such example is the update notification mechanism provided by a PIX manager. Figure 3 shows the interactions used to implement this specific test case. It involves the following steps:

- The testing software registers patient John Smith in domain A
- The testing software registers patient John Smith in domain B with a different date of birth.
- The testing software queries for the patient's identifier in domain B; this should return no result as the demographics are incorrect in domain B.
- The testing software updates patient John Smith in domain B; records from domain A and B should now be matched.
- The system under test is expected to issue an update notification indicating the ID of John Smith in domain B.

B. Testing Software Architecture

Development was carried out using the JAVA programming language because it can be used to build multi platform applications and because the JAVA open source community provides all the tools required to build web applications. The software can be shipped as a simple Web Archive (.war) and be used within the Tomcat[7] or Jetty[8] web containers. It uses log4j[9] for logging, SOAP with Attachments API for Java (SAAJ) [10] to produce and decode SOAP envelopes and WS-addressing segments. Xerces[11] and Xalan[12] are used as XML parser and XLST[13] processor respectively.

The complete architecture is presented in Fig. 4; it is composed of the following blocks:

1) Tester's console: A graphical users interface (GUI) that uses a web browser and permits to select a test case, provides test instructions and displays test results. It communicates over the network via HTTP with the Web Server. This GUI can run on the same machine as the server or the system under test.

2) SUT: The system under test. It interacts with the SOAP Endpoint over the network via HTTP.

3) Web Server: Built on JavaServer Page, it allows the tester to select an actor to test, to select a test case, to read instruction, to start the test and to review results. It uses the HTML engine to generate the actor list, the test cases list and the test description.

4) HTML Engine: Responsible for generating HTML content from files describing actors and test cases. It uses the Test Parser to generate test description pages based on the content of the test.xml file located in the specific test subfolders.

5) SOAP Endpoint: When a test has been started and incoming messages are expected, the SOAP Endpoint Servlet will listen for HL7 messages; a received message is passed to the Test Manager. The Test Manager response is sent back to the SUT.

6) Test Manager: The main component of the software. When a test is active, the Test Manager receives the incoming HL7 message from the SUT, validates it, and constructs an appropriate response. It uses a Configuration module to determine the response endpoint, the SOAP version to use, and the disk location where to save the generated log files. It relies on the Test Parser to get the list of the test steps, as

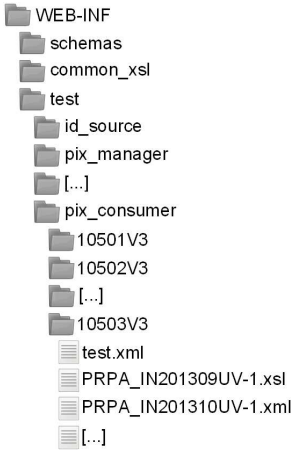

Fig. 5. File system architecture.

well as the incoming and outgoing messages that are required to complete the test. Received messages are handed to the Validation module. Errors are reported trough the Logger module. XML templates with embedded style sheets are used to generate response messages. They are stored within the test case folders and the style sheets are used to fill dynamic fields before responses are sent back to the client.

7) Validator: Analyses the received messages using XML schemas and XSL style sheets to verify the messages syntax and content. Schemas validation is performed by Xerces parser while the message and SOAP envelope contents are validated using the Xalan XSLT processor. Xalan allows calling java code from a style sheet using the Xalan-Java Extension enabling issue reporting directly to the Logger.

8) Config: Helper module to access settings from the config.xml file.

9) Logger: Based on Apache Log4j that allows customized logging with multiple levels (info, warn, error) and provides output options ranging from console prints to emails messages. This module receives info, warnings and errors notifications from all other modules. Two new levels were developed, HL7info and HL7error, to better represent problematic issues with the received messages from the SUT. All log levels are saved in an application log file. HL7info and HL7error log levels contain interoperability issues; they are saved in files specific to each test case. Logs are generated in HTML and XML formats and contain the evaluation results.

C. Adding a new test

We have implemented a simple mechanism to be able to easily add or modify test cases without rebuilding the software. Users are only required to modify files contained within a ”test” folder located at the file system root of the web application (Fig 5). The tester needs to select the actor under test as well as a specific test case. To present the list of actors to the tester, subfolders of the test folder are automatically scanned and interpreted as actors. Subfolders of each actor folder are interpreted as individual test cases. The list of test case is automatically generated and presented to the tester when an actor is selected. Each test case has a mandatory test.xml file that serves two purposes: providing a summary description of the test case along with instructions intended for the tester of the SUT and describing the steps needed by the system to complete the test. Moreover, each test case has several XML and XSL files that provide a specification of inbound and outbound messages along with their validation style sheets.

VI. Conclusion

We have presented the functionalities and architecture of a web application for testing interoperability in healthcare based on the HL7 standard v3. The software tests technical, semantic and functional interoperability of patient's identifiers and demographics queries, as specified by the IHE technical framework. It has been used by several healthcare system manufacturers to test their implementations as a prerequisite to their participation in IHE connect-a-thons. We have also described how to extend the system in order to add additional tests. The testing software is available from [14, 15] and can be used by system implementers to test their implementations. It can also be used by site integrators to verify and test the interoperability of their systems, or by developers to understand specifications ambiguities, and to resolve implementations difficulties.

References

[1] IHE International. Patient Identifier Cross-Reference (PIX) and Patient Demographic Query (PDQ) HL7 v3. 2009, [Online] Available at: http://www.ihe.net/Technical_Framework/upload/IHE_ITI_TF_ Supplement_PIX_PDQ_HL7v3_TI_2009-08-10.pdf, June 5, 2010

[2] Web Services Description Language (WSDL) Version 2.0 Part 1: Core Language, [Online] Available at: http://www.w3.org/TR/wsdl20/, June 5, 2010

[3] SOAP Version 1.2 Part 0: Primer (Second Edition), [Online] Available at: http://www.w3.org/TR/soap12-part0/, June 5, 2010

[4] Gebase L., Snelick R., and Skall M., Conformance testing and interoperability: a case study in healthcare data exchange, Proceedings of the 2008 International Conference on Software Engineering Research & Practice, pp. 143-9, 2008

[5] MESA software, IHE Test Tools. [Online]. Available: http://ihedoc.wustl.edu/mesasoftware/index.htm, April 1, 2010.

[6] Namli T., Aluc G., and Dogac A., An interoperability test framework for HL7-based systems, IEEE Transactions on Information Technology in Biomedicine, vol. 13, no. 3, pp. 389-99, May 2009

[7] Apache Tomcat, [Online] Available at: http://tomcat.apache.org/, June 5, 2010

[8] Mort Bay Jetty WebServer, [Online] Available at: http://jetty.codehaus.org/jetty/, June 5, 2010

[9] Apache log4j, [Online] Available at: http://logging.apache.org/log4j/index.html, June 5, 2010

[10] SAAJ Standard Implementation, [Online] Available at: https://saaj.dev.java.net/, June 5, 2010

[11] Apache Xerces Java Parser, [Online] Available at: http://xerces.apache.org/xerces-j/, June 5, 2010

[12] Apache xalan-java, [Online] Available at: http://xml.apache.org/xalan-j/, June 5, 2010

[13] XSL Transformations (XSLT) Version 2.0, [Online] Available at: http://www.w3.org/TR/xslt20/, June 5, 2010

[14] PIX/PDQ V3 Software, [Online] Available at: http://ihe.etsmtl.ca, June 5, 2010

[15] Testing tools source code of the Integrating the Healthcare Enterprise (IHE) PIX and PDQ v3 Integration Profiles, [Online] Available at: http://sourceforge.net/projects/ihe-pixpdqv3, June 5, 2010