Annals of DAAAM for 2010 & Proceedings of the 21st International DAAAM Symposium, Volume 21, No. 1, ISSN 1726-9679
ISBN 978-3-901509-73-5, Editor B. Katalinic, Published by DAAAM International, Vienna, Austria, EU, 2010
Make Harmony Between Technology and Nature, and Your Mind will Fly Free as a Bird
Annals of DAAAM International

TESTING THE INTEROPERABILITY OF HL7-BASED APPLICATIONS USING TTCN-3

EGNER, A[lexandru]; MOLDOVEANU, F[lorica]; MOLDOVEANU, A[lin]; ASAVEI, V[ictor]; MORAR, A[nca] A[ndreea] & BOIANGIU, C[ostin] A[nton]

Abstract: *The testing of HL7 conformance has recently gained more and more popularity due to the importance of correct intercommunication between critical systems such as e-Health applications. The HL7 messages are very complex, and there are major differences between the versions of the HL7 protocol. This paper presents our testing solution, which offers a general and extensible way of addressing these issues and others. We used a standardized testing technology, TTCN-3, and its template matching mechanism to validate the messages exchanged by medical systems. The HL7 profile we tested was QED, but our method allows extension to any other profiles.*
Key words: *HL7v3; TTCN-3; IHE QED; SOAP; JAX-WS RI*

1. INTRODUCTION

HL7 protocol is very complex and defines a lot of optional fields (Hinchley, 2005). This problem has led to the emergence of many HL7 profiles, which aimed to narrow the protocol, constraining it to specific fields of activity. One example of HL7 profile is Query for Existing Data (QED), a standard developed by Integrating the Healthcare Enterprise (IHE). The increasing number of such HL7 profiles makes the testing troublesome. Our approach comes from the need of a generic testing solution, extensible and flexible. The main advantage of our approach is the use of a standardized testing technology, Testing and Test Control Notation version 3 (TTCN-3), which is reliable, very flexible and independent of the platform and the technology of the system under test (Willcock et al., 2005). In addition, the TTCN-3 test system is portable and modularized. This solution was adopted by the ReTeMeS (Reliability Testing of Medical Systems) partners. ReTeMeS is a Eureka project which aims for the development of new methods for medical applications testing.

2. GENERAL DESCRIPTION

The system under test (SUT) was Hospital Manager, an application developed by InfoWorld, one of the ReTeMeS partners. The communication with Hospital Manager was done with HL7v3 messages, through a web service. This enabled the possibility of remote testing. TTCN-3 allows both message and procedure-based communication, but in this case the message-based type was chosen.

Briefly, the testing method is as follows. Firstly, different TTCN-3 templates containing several queries and their expected responses are defined. The queries are processed and translated into HL7 messages by the TTCN-3 test system. The SUT receives these HL7 queries and replies to them. The replied HL7 message is then translated into a TTCN-3 template and compared to the expected response template. The TTCN-3 template matching mechanism enables then to establish if the response from the SUT is the one expected.

Before detailing the message flow, we mention two of the most important TTCN-3 test system components, the Codec and the Adapter. The Codec is responsible for encoding and decoding the TTCN-3 templates into and from Java. It is basically the link between the TTCN-3 scripting language and the TTCN-3 test system. The Adapter is responsible for assuring the correct communication with the SUT. The communication protocol between the Adapter and the SUT is Simple Object Access Protocol (SOAP).

The following figure illustrates the processing steps in a QED message exchange, based on our implementation.

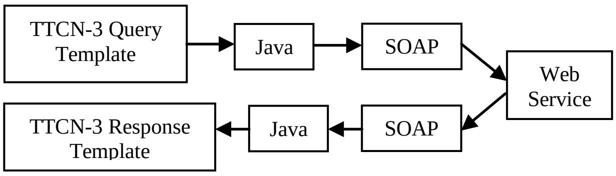

Fig. 1. Processing steps in a QED message exchange

The TTCN-3 template that describes the QED Query is sent to the Codec through the TTCN-3 Control Interface (TCI, 2007). The message is translated into a Java object that is passed on to the Adapter through the TTCN-3 Runtime Interface (TRI, 2007). The Java object is then serialized and embedded into a SOAP message. After the connection between the Adapter and the SUT is established, the SOAP message is passed on to the SUT, which, in our case, is represented by a web service. If the query is valid, the web service replies with a QED Response in SOAP format. The Adapter converts the SOAP message to a Java object and forwards it to the Codec which decodes it back to a TTCN-3 template. At this point, the Testing Execution (TE), another important TTCN-3 test system component, can evaluate the received response and establish if it matches the expected one, setting the verdict of the test case.

3. IMPLEMENTATION DETAILS

A. Codec

The Codec is an important TTCN-3 test system component. It has two basic functions: encode and decode. The TE interprets the TTCN-3 script describing the test case and automatically converts the TTCN-3 template representing the QED Query to a Java object, organized as a structure. After the conversion, the Java object is sent to the Codec.
When encoding, the Codec is responsible for translating this structure into a Java HL7 object. By doing so, the Codec assures that the Adapter receives a set of input data that it can handle. This translation is performed at runtime, with Java Reflection. The structure is being parsed, and each composing element is translated into the corresponding HL7 Java object. These objects are then merged into a single Java object representing the query.
The Codec is also responsible for sending this query request to the Adapter. The TRI defines the interaction between the TE and the Adapter. There is a restriction that the usage of TRI implies. All classes defining the messages used in the communication between the TE and the Adapter must

implement the TriMessage Java interface. This constrains the messages to be formatted as byte arrays. Thus, we had to develop a simple script to modify the JAXB generated HL7 classes, in order to make them Serializable.
The decoding function is just the opposite of the encoding one, meaning the Codec receives a TriMessage from the Adapter, containing the QED Response, deserializes it, converts it into a Java structure and forwards it to the TE.

B. Adapter

Another important TTCN-3 test system component is the Adapter. Its existence confers the TTCN-3 test system much flexibility. It is the only TTCN-3 test system component that establishes connections and handles the communication with the SUT. This means that the same test suite can be run on several SUTs with different platforms just by replacing this component. Our Adapter has two different roles: the encapsulation of the query and the extraction of the response from SOAP messages.
When encapsulating, it uses the TriMessage it receives from the Codec as input and translates it into a SOAP message which will later be sent to the SUT. Given the transparency of the test case to the Adapter, the conversion from Java to XML can only be done dynamically, at runtime, through Reflection. For this translation we used the Java API for XML Processing (JAXP), which provides the capability of validating and parsing XML documents. JAXP offers several parsing interfaces from which we chose the Document Object Model (DOM) parsing interface.
The DOM documents have a tree-type structure. They are composed of a root element, which represents the XML document, and several nodes, representing XML elements. The translation of the Java message to XML was implemented as follows. A DOM document was created based on the type of the query message. Then we used Reflection to obtain the object's fields list, each field representing a Java HL7 object. For each field in that list a DOM element was generated and added to the root element's children list. Afterwards, another DOM element was defined, based on the preconditions and added to the root's children list. The DOM document was then serialized and ready to be forwarded to the SUT.
When receiving the response from the SUT, the Adaptor deserializes the XML into a DOM document. The root element is used to generate the Java object representing the QED Response. The document is then parsed and each node is translated into the corresponding HL7 Java object. Using Reflection, these objects are set as fields of the Java-based QED Response object. After the parsing is finished, this object is serialized to a byte array and sent to the Codec via a TriMessage. The communication with the SUT was one of the Adapter responsibilities as well. Hospital Manager provided a web service to interface the communication with the Adapter. The communication protocol chosen for exchanging messages was SOAP. The web service used Web Services Description Language (WSDL) to define the type of QED messages it can handle, such as Query, Continue or Cancel (Chinnici et al., 2007). For the communication to take place, first the connection had to be established, and for that a Java client was needed. There are many tools that use the WSDL description to generate stubs and clients, and our choice was Java API for XML Web Services – Reference Implementation (JAX-WS RI). Once the client has been created, the methods for connection, sending and receiving QED messages were available and the communication was possible.

C. Defining test cases

We defined and ran several test cases to simulate various scenarios. We tested the SUT's behavior to queries of clinical information regarding patient allergies, immunizations, diagnostics or medication. The first step in building the test cases was the creation of the corresponding TTCN-3 templates.
We have first created several templates to match HL7 data types. For the flexibility, for some data types we created more than just one template, varying on the semantic content of the message that includes it. For instance, we created two CD templates. CD is a HL7 data type used for coding. The first CD template was defined to match any other CD template. The second CD template was defined to match only the ones containing certain field values. This approach provided much flexibility when creating the message templates.
The next step was to create the query and response templates. In creating the templates we took into account the preconditions, which represent information about the patient and the existing data to execute the queries on. The objective was to analyze the message both syntactically and semantically. For the syntax validation a simple template was enough. On the other hand, for the semantic validation more templates had to be defined for the same query.
The template matching mechanism helped then deciding which response was similar with the one expected. The verdicts established for these test cases enable to report the conformity of the application with the HL7 protocol.

4. ADVANTAGES

There are many advantages that come with our approach. Probably the most important one is the usage of a standardized scripting language, TTCN-3, and TTCN-3 testing system. The flexibility of TTCN-3 enables the adaption of the current approach to virtually any HL7 profile.
Moreover, our solution does not directly depend on the SUT, neither on its architecture, nor on the technologies it uses. The Adapter component is responsible for linking the SUT with the test suite, which means that it is the only component that needs to be replaced when testing other systems.
Another advantage is the automation. Test suites can be developed to thoroughly test several systems, without user intervention. Logs can be saved to allow post-processing.

5. CONCLUSIONS

The solution we proposed turned out viable, the results being a functional system that is reliable and can unambiguously determine the HL7 conformity of any application. The testing of Hospital Manager revealed there is a type of message the application handled erroneously, and also some minor semantic differences.
However, there are some enhancements that can be made. The TTCN-3 template definition is a bit difficult and requires solid knowledge both of TTCN-3 and HL7. A generating tool for defining these templates based on the preconditions the tester specifies would be a great addition.
Another extension would be the development of a generic Adapter which defines most of the common Adapters' behavior, making it easier to adapt the testing to new systems.

6. REFERENCES

Willcock, C.; Deiβ, T.; Tobies, S.; Keil, S.; Engler, F. & Schulz, S. (2005). *An introduction to TTCN-3*, John Wiley & Sons, 978-0-470-01224-6

***(2007) *Methods for Testing and Specification (MTS); The Testing and Test Control Notation version 3; Part 6: TTCN-3 Control Interface(TCI)*, ETSI ES 201 873-6 v3.2.1

***(2007) *Methods for Testing and Specification (MTS); The Testing and Test Control Notation version 3; Part 5: TTCN-3 Runtime Interface (TRI)*,ETSI ES 201 873-5 v3.2.1

Hinchley, A. (2005) *Understanding Version 3 – A primer on the HL7 Version 3 Communication Standard*

Chinnici, R.; Moreau, J.; Ryman, A. & Weerawarana, S. (2007) *Web Services Description Language (WSDL) Version 2.0 Part1: Core Language*