A TTCN-3-based Test Automation Framework for HL7-based Applications and Components

Diana Vega, Ina Schieferdecker, Technical University of Berlin

George Din, Fraunhofer Fokus Berlin

Abstract

Information systems used in health care use more and more a common approach of structuring and representing the health care data. One of the standards addressing such an approach is HL7 Standard (Health Level Seven) [HL7 08]. It supports the inter-communication of various components in different medical facilities known as HIS (Healthcare Information Systems).

So far, little attention has been given to a common methodology for assuring the quality of HIS systems and their testing. In this paper we present a test framework for HIS-based health care applications, which is based on the standardized test technology TTCN-3 providing both a testing language for specifying tests and a test architecture for the automated execution of tests. TTCN-3 has been successfully applied to test applications in telecommunication, automotive, etc., its application for health care systems is however in the beginning. We show how TTCN-3 can be applied to analyze medical information and how existent TTCN-3 based test methodologies can be re-used in the health care area. Due to the increased level of complexity of medical information used in HIS-based systems, manual testing is almost impossible and a very tedious task. Therefore, our framework aims at a higher degree of test automation.

1 Introduction

The electronic health industry is among the fastest-growing industries in all industrialized countries, especially in US. Electronic health applications are used in a variety of settings such as in a hospital, where health care professionals treat patients, administer and document diagnostic and therapeutic procedures, take notes, look up data from previous treatments, etc. Peculiarities in the medical workspace are related to three major roles: *physician*, e.g. make decisions, *nursing*, e.g. sustain ongoing care, and *billing*, e.g. handle reimbursement. These roles need to be clearly distinguished by a medical application; overlapping functionality should be avoided, e.g., a nurse is not allowed to provide a diagnosis. In reality, potential overlaps between roles caused by erroneous implementations represent a major issue which should be identified before application deployment by adequate testing.

An Electronic Health Record (commonly known as an EHR) is a secure and private lifetime record of an individual's health and care history, available electronically to author-

ized health care providers. In [Eic 05], a detailed survey of EHR standards is provided. It is notable that an important part of the standardization efforts in this area is concerned with the communication among medical components e.g. DICOM [NEM 08], ISO/IEEE 11073[ISO 04], and Health Level 7 (HL7) [HL7 08]. HL7 is an ANSI [ANS 08] standard for health care specific data exchange between medical system components. The name refers to the top layer (level 7) of the ISO Open Systems Interconnection (OSI) protocol stack used also for medical systems.

HL7 consists of standardized messages and message exchange protocols so that clinical data can be shared among medical system components, and correctly understood by all. By use of HL7, system components are able to communicate with one another without the need for superfluous information conversion, data correction, etc. However, HL7 messages may not be properly constructed. Faulty messages can be created by including proprietary or faulty fields. There are also different flavors of HL7, which along with non-conforming implementations, result in interoperability issues and reliability decrease. Typically, two HL7 devices, produced by two different vendors, do not work together properly. Hence, there is a strong demand for testing methodologies that are suitable for the medical domain, that are flexible enough to cope with different variants and configurations and that demonstrate and certify the quality level of tested components, devices and systems.

Hence, the development of medical systems suggests the development of an appropriate test environment: parallel to the risk analysis the testing strategy, test priorities and the test designs should be defined and precisely specified. The development and execution of tests can take up to 50% of the overall system development time and resources. Improving the efficiency of the test process will bring large economic effects - especially for the HL7 domain, where the technology still lacks clear guidance for checking the conformance to the HL7 standard. As the predecessors of the Testing and Test Control Notation (TTCN-3) have been originally developed for conformance and interoperability tests of OSI protocol stacks up to protocol layer 7, there is a natural link between the two standards: the OSI-oriented protocol stack HL7 and the testing technology TTCN-3, which is well applicable to protocol testing.

The approaches and results presented in this paper are concerned with the outcome of an ongoing project in Germany which started in December 2007: Test Automation for the Next Generation of Medical Systems (TestNGMed) [Pro 08]. The concepts presented in this paper regard the contribution of the authors of this paper to the overall project demands. The general target of TestNGMed is to develop a TTCN-3 based testing methodologies for HL7 based medical systems and their components. The intended result is a test framework providing prefabricated test configurations, test data and test procedures which will enable an easy, flexible, reusable and scalable testing of HL7-based medical systems.

The rest of the paper is organized as follows: in Section 2 we address the issues of testing medical systems. Next, in Section 3 we present our approach and the developed principles for building a TTCN-3 test framework for the electronic health care domain. Section 4 presents methods for the development of a TTCN-3 test system for HL7: the guidelines for using TTCN-3 for HL7-based applications and the resulting mapping from HL7 specifications to TTCN-3 test suites. In the end, we give our conclusions.

2 Testing in the Electronic Health Care Universe

In a typical medical environment, a large variety of data formats is used. An Electronic Health Record (HER) includes information such as observations, laboratory test results, diagnostic imaging reports, treatments, therapies, drugs administered, patient identifying information, legal permissions, etc. The Health Level 7 (HL7) standard is broadly used, but it does not solve the integration problems in electronic health care [Jab 04]. The main issues in electronic health care systems with regard to data interoperability aspects may be revealed with the help of the following types of testing:

- *HL7 interface unit testing* – typically interface specification based aiming to confirm that HL7 messages sent and/or received from a medical application conform to the HL7 interface specification.
- *HL7 interface integration testing* – testing of business scenarios to ensure that information is able to flow correctly between medical applications.
- *HL7 interface system testing* – end-to-end scenario testing focused on ensuring all relevant components of all relevant medical applications are able to interoperate correctly.

Other testing may regard functional testing or end-user acceptance testing and will typically follow interface testing. Further testing issues need to be addressed as well:

- system level test design for the overall system reliability,
- automated test generation from message and system interaction specifications,
- risk-oriented test strategies and test selection, or
- test coverage metrics for a quantifiable reliability analysis.

The testing approach presented in this paper regards HL7 Version 2.5.1 [HL7 08]. We refer to this version throughout the rest of the paper whenever we mention HL7. The motivation for selecting this version is based on the fact, that a multitude of electronic health care applications exist already on the market and are built on top of this version.

3 TTCN-3 Based Framework

3.1 A Short TTCN-3 Overview

The TTCN-3 [ETS 05] standardized testing language is a text-based language, which is for testers easy to learn and to use. It has been specifically designed for testing and inherits the most important typical programming language artifacts, but includes additionally important features required for the specification of tests. It constitutes a domain-specific language for testing specification and implementation.

A TTCN-3 based test specification is called Abstract Test Specification (ATS). It usually consists of various TTCN-3 modules grouped into file folders and subfolders. A *module* is the top level element of the TTCN-3 language which is used to structure the test definitions of:

- *test data*: the structure of test data being defined as **types** of messages, and the concrete test data being instances of these types called **templates**,
- *test configurations*: **ports** and test **components** to define the interfaces to and the active entities of the test system,
- *test behavior*: **functions**, **altsteps**, and **testcases** which implement the interactions between the test components and the SUT, which are follow the test configuration and make use of the test data along the test behavior,
- *control*: the global behavior of the test system to control the flow of test case execution

TTCN-3 permits also the import of data types from other languages than TTCN-3 like ASN.1, IDL, or XML Schema [Jea 04], enabling the direct applicability of TTCN-3 to test applications from the automotive domain, telecommunications or based on Web services.

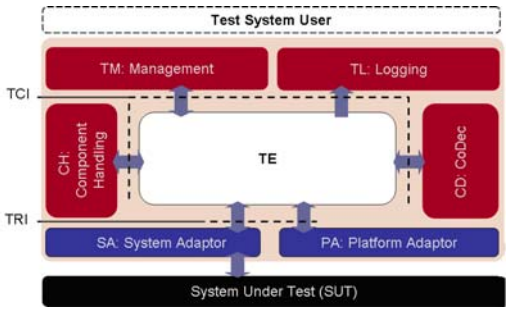

Figure 1 TTCN-3 Test System Architecture

In order to get an executable test system based on TTCN-3, in addition to the abstract test specification, a TTCN-3 execution environment has to be provided as well. According to the general TTCN-3 test system architecture [TRI 05], one has e.g. to ensure the real communication between the SUT and the test system by an *adaptation layer* component and an encoding/*decoding* component, shortly called *adapter* and *codec*. All components building a TTCN-3 test system architecture are illustrated in the Figure 1. Typically, the components are provided by a TTCN-3 tool environment, except of the ATS (out of which the executable tests are being generated), the adapter and the codec which are specific to the SUT. In this paper, we focus on how various artifacts composing the TTCN-3 test specification, i.e. the ATS for testing HL7 based systems, can be defined and/or generated.

3.2 The TTCN-3 based Test Framework for HL7

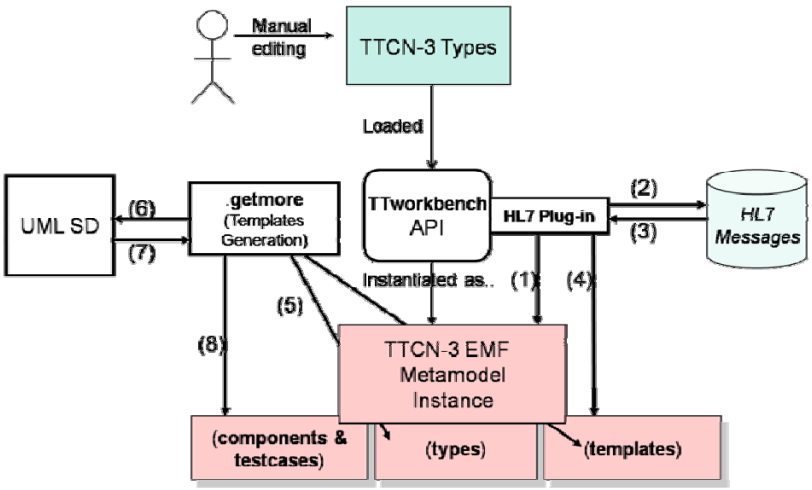

Figure 2 The TTCN-3 Framework Architecture for HL7

Figure 2 presents our approach for developing tests for HL7 based applications. It combines manual test specifications with automated processes for partial test generation and fully automated test execution. Two proprietary tools are foundation pillars of our approach: TTworkbench [Test 08] and .getmore [sep 08]. Our contribution regards the components directly linked with TTworkbench [Test 08], which is used to specify and execute TTCN-3 test specifications. Therefore, we focus on discussing the TTCN-3 types and templates for HL7. The .getmore tool uses the produced types and templates to generate the logic of the tests out of the HL7 use case scenario described in UML, e.g. Sequence Diagrams (SD). However, test case generation is not the subject of this paper.

An important component of the whole architecture constitutes the TTCN-3 Metamodel [Sch 04], technically realized by using the Eclipse Metamodeling Framework (EMF) and encapsulated within the TTworkbench platform. It allows the construction of TTCN-3 artifacts in a model-oriented paradigm and plays the role of a bridge between interacting tools and the TTCN-3 language elements.

Data type definition. Every TTCN-3 ATS requires a type system in accordance with the SUT interface input and output data types. In electronic health care applications, the SUT exposes interfaces whose interaction messages are described in HL7. Consequently, the test system must provide a type system that can understand HL7 messages. Therefore, a mapping from HL7 messages to TTCN-3 types is necessary. The conceptual mapping will be presented in Section 4. From an implementation point of view, this step may be achieved either manually, by analyzing the format of each HL7 message, or by implementing an automated mapping, i.e., a transformer. The second alternative requires that a formal description of HL7 messages is given, e.g. a database, XML, etc.

Templates. The type definition provides the basis for TTCN-3 template definitions constituting the test data. Templates play the role either of SUT stimuli or of SUT response verification patterns. In the first case, fully specified type instances have to be specified. They represent inputs that an HL7 application should understand. In the case of response verification pattern templates, not only concrete values can be used, but also so called matching mechanisms including wildcards "*", indicating the presence of an arbitrary value or the omission of a value, or "?", indicating the presence of an arbitrary value. They are useful for setting the verdict of the test when the expected responses and the real outputs of the SUT are analyzed and matched against each other. For example, not all fields of a message may count in the verdict decision making which decide about the correctness of the SUT responses, therefore wildcards may replace concrete values in the response verification pattern templates.

In order to provide TTCN-3 templates based on a type system, one has the following possibilities:

- *manual template definition*: the templates are defined manually. A larger effort is demanded to achieve this. The templates are described directly in the TTCN-3 language, so that later changes are possible. However, due to the manual development a lot of inconsistencies may occur in the specification.
- *templates generated from a data pool*: a data pool which conforms to the messages of interest is provided. A *transformer* is needed to transform the data given in the data pool to TTCN-3 templates. The transformer is based on the data pool message structure and may produce TTCN-3 templates directly or may use an intermediate format as indicated in Figure 2 with the box "TTCN-3 Metamodel Instance". After transformation, the test data is not only available in the data pool, but also in TTCN-3.

- *templates loaded via external functions*: also in this case, a data pool is required. The templates are not visible to the TTCN-3 test specification, but are loaded dynamically at execution time into existent template variables. Afterwards, the loaded templates can be referenced like normal TTCN-3 templates. This mechanism is similar to the serialization mechanism in the Java language. It has the advantage that the decoding function of the test system can be used instead of a transformer, but has the disadvantage that the template specification is not visible and no changes can be made in the template definition.

The second option is by far the most convenient. Although the implementation of the transformer requires quite some efforts, the approach has the big advantage that the transformer can be reused for any data pool. In our implementation we use the TTCN-3 metamodel to create TTCN-3 templates by instantiating directly the HL7 types loaded in the TTworkbench tool. Then, a transformer fills the the metamodel instances with concrete data extracted from a data pool, which itself is generated by an HL7 message editor.

3.3 A Test Configuration Example

The templates are used by functions implementing the test behavior to interact with the SUT. The functions are started on TTCN-3 parallel test components (PTCs). A test case has always a main test component (MTC) which runs the test itself and manages the creation and execution of PTCs. The MTC and PTCs can communicate with each other or with the SUT via ports. Each test component can have an arbitrary number of ports.

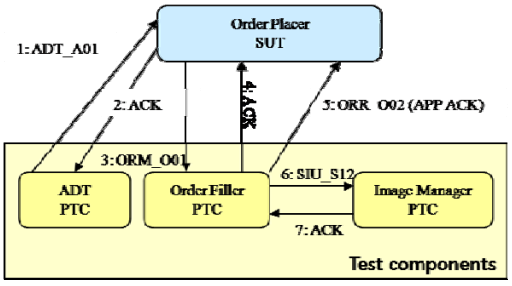

Figure 3. TTCN-3 Test Configuration

The test components and their connections via ports is called *test configuration.* For testing HL7 based applications, the test configurations may become quite complex involving many PTCs and ports. An example of such configuration is described in [Sne 07] and is depicted in Figure 3. The test configuration consists of three PTCs implementing three different functionalities: **ADT**, **Order Filter** and **Image Manager**. The PTCs interact with the SUT via *system* component, in this case the **OrderPlacer** component. As the figure illustrates, many interactions between the test components and the SUT take place. Obviously, the test system has to simulate the real behavior of the three

functionalities realized as PTCs and it also has to validate the content of the messages received from SUT. The sequence of messages (indicated in the figure) also needs to be validated to ensure that the messages come in the correct order.

Many such interaction use cases involving laboratory, radiology and cardiology components are made publically available by the Integrating the Health Care Enterprise standard [IHE 08].

4 An HL7 to TTCN-3 Mapping

This section discusses in detail the mapping of HL7 data structures to TTCN-3 type systems so that TTCN-3 can be used directly for the testing of HL7 based application. HL7 data structures are of two categories for 1) *messages between* application systems, and for 2) *events triggering* message based interactions.

4.1 The HL7 Message Format

An HL7 *message* is the atomic unit of data transferred between system components. Each message has a message type defining its purpose. For example, the ADT (Admit/Discharge/Transfer) message type is used to transmit portions of a patient administration data from one system to another. A trigger *event* causes the data to flow among systems.

A message consists of a set of *segments.* The segments define logical groups of data fields which are string of characters. The structure of messages is illustrated in Figure 4. Within a HL7 message, a segment may be optional, i.e. it is marked with "[]" brackets, or it may repeat for a number of times (up to a specified number) forming the so called *groups* indicated by "{}" brackets.

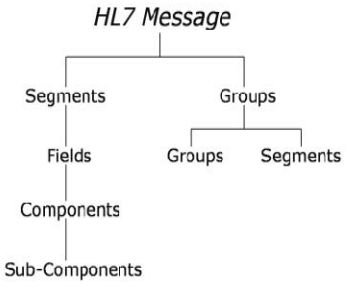

Figure 4 HL7 Message Structure

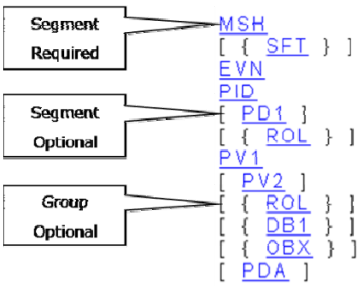

Figure 5 ADT_A02 Message Structure

Figure 5 shows an example of the HL7 message structure for a patient transfer event. The message format is of the ADT category type and is associated to the trigger

event A02. An A02 event is issued whenever a patient is changing his or her assigned physical location. As result of combining the ADT category type with the trigger event A02, the message type is ADT_A02. The encapsulated segments provide a logical grouping for *data elements*. For example, the Patient Identification segment (PID) is mandatory in the ADT_A02 message and includes fields for identifying information such as patient name, social security number, medical record number, account number, and miscellaneous details. The segment PD1 (Patient Additional Demographic) is an optional segment in the ADT_A02 message. The segment may appear once or may be omitted. The same rule applies for ROL segment, with the difference that the mark "{}" indicates that if the segment is present it may repeat more than once.

4.2 The Mapping

The mapping rules of HL7 types to TTCN-3 data types follow closely the structure of the HL7 message format. In Table 1, we summarize the composing parts of a HL7 message and how they relate to TTCN-3 constructs.

Table 1 Mapping of HL7 Entities to TTNC-3 Elements

HL7 Entities	Mapping in TTCN-3
Functional group & Message Type. The HL7 specification document defines various functional groups according to a common application function, for example, ADT, Order Entry, Finance. Within a functional group one or more message types are defined.	These map to many **record types** whose name start with the name of the HL7 functional group. For example, all messages of the ADT functional group are mapped into record types named *ADT_*_Message_Type.* The * is replaced with the appropriated triggering event.
Message definitions describe sets or combinations of segments that make up a properly formed message. For example, ADT distinguishes among more than thirty separate message definitions based on "trigger events". Each message definition includes one or more segments.	These map to TTCN-3 **record types** containing fields of other TTCN-3 types corresponding to HL7 segments as the order of fields matters. The order plays an important role in defining the sequence of segments within an HL7 message. *Listing 1* shows an example on how the ADT_A02 HL7 message structure maps to a TTCN-3 record. The *optional* segments map to record fields that are marked with the optional keyword. The record fields may be of type *Group_Type* as for instance SFT_Group_Type (line 3) in Listing 1. In TTCN-3, we define a group as a **set of** data structures (Listing 2).
Segment definitions provide a logical grouping of other data elements. Segments can be *required* or *optional*, can be nested, and can repeat.	These map to TTCN-3 **record types** having as fields other simple, e.g., **charstring**, or structured types, i.e., **record**. The fields may be also of type **set of** when the fields are repeatable.

Fields. HL7 uses abstract data types to define the types of the fields. Some fields may hold more than one data element. In addition, many fields are (or can be) coded, and the standard includes a variety of code *tables* to define acceptable contents.	These map to **charstring** based types with length restrictions. For instance, in Listing 3, the ST data structure with a specific length maps to a charstring based type with length restriction. Restrictions can be defined also for the allowed values as presented in Listing 4, i.e. HL7 tables. The HL7 fields holding more than one data element map into **record types.** These records contain fields of either basic types or of record types.

Listing 1. Mapping Example – ADT_A02 Type Description in TTCN-3

```
1.  type record ADT_A02_Message_Type  {
2.      MSH_Segment_Type MSH_Segment,
3.      SFT_Group_Type SFT_Group optional,
4.      EVN_Segment_Type EVN_Segment,
5.      PID_Segment_Type PID_Segment optional,
6.      ROL_Group_Type  ROL_Group_1 optional,
7.      PV1_Segment_Type PV1_Segment,
8.      PV2_Segment_Type PV2_Segment optional,
9.      ROL_Group_Type  ROL_Group_2 optional,
10.     DB1_Group_Type  DB1_Group optional,
11.     OBX_Group_Type  OBX_Group optional,
12.     PDA_Segment_Type PDA_Segment optional
13. } ;
```

Listing 2. Group Example – STF Group Description in TTCN-3

```
type record length (1..infinity) of SFT_Segment_Type SFT_Group_Type;
```

Listing 3. Length Restriction Example – ST20 Description in TTCN-3

```
type charstring ST20 length (0..20);
```

Listing 4. Table Example – YesNo Table Description in TTCN-3

```
type charstring ID;
type ID YesNo_Indicator_Table("Y","N");// Table 0136
```

Along the mapping, not only the structure of messages needs to be considered. The attributes of various elements are also very important. In the following, we analyze the type of attributes that may accompany segments of data structures within a HL7 message:

- **Usage** indicates how the element can be used: Required, Optional, Not Supported, Conditional, Required or Empty. In TTCN-3, we have the option to define whether a field is mandatory or not by using the keyword *optional.*
- **Cardinality** indicates how many times an element can appear, e.g. [0..0], [0..1], [1..1], [0..3], [3..5], [0..*]. TTCN-3 language does not offer the possibility to specify such attributes, but instead it is possible to define **set of** types with specific length constraints. Hence, any field, segment, etc., having a cardinality greater than one is mapped to a **set of** type (*"_Group_Type"*) encapsulating the demanded restrictions. An example of cardinality is provided in Listing 2. For cardinality of type [0..1], no group type is introduced but the element is made optional.
- **Code Sets (Tables)** indicate sets of valid values for a given HL7 primitive element (e.g. Table 0136 has a Yes No indicator). We defined for each table a new charstring based type (postfixed by*_Table*) which indicates the allowed values (e.g. Listing 4).
- **Length** indicates the maximum length of an element. This translates in TTCN-3 to a charstring based type with length restriction (e.g. Listing 3).

4.3 Mapping Guidelines

As long as the test specification contains thousands of definitions, it is extremely important to be consistent in writing TTCN-3 test definitions and in maintaining a clear test suite and module structure. To keep the test specifications consistent, readable, reusable and well structured, we developed and applied a set of guidelines for writing and structuring the TTCN-3 specifications. A more general approach for defining guidelines for TTCN-3 test specification has been proposed in [Din 08]. In the following we elaborate some of our HL7 guidelines for TTCN-3:

Naming conventions. Identifiers for TTCN-3 elements are required to follow certain construction rules. The naming conventions concern prefixing and postfixing rules and apply to all TTCN-3 elements which require an identifier: types, templates, functions, altsteps, testcases, groups, modules, variables, etc. For easier localization, the TTCN-3 identifiers can be prefixed with a string indicating a group of definitions of the same category. For example, the message types can be prefixed by strings such as Type, type, type_, T etc. Multiple prefixes can occur. For example, type definitions can be grouped into types of messages to be sent to SUT, e.g. Send_Msg, and types of messages to be received, e.g. Received_Msg. If multiple prefixes are used, they can simply be concatenated or separated by the "_" character.

Table 2. Naming conventions for mapping HL7 entities to TTCN-3

TTCN-3 identifiers	Naming Convention
Message types	**<HL7Message(Profile)Name>_Message_Type**
Segment types	**<HL7SegmentName>_Segment_Type**
Group types field instances	**<TableName|<HL7FieldName>_Segment Name>_Group_Type[_#Nr.]**
Message field names	**<HL7FieldName>_Segment[_#Nr.]**
Tables	**<HL7TableName>_Table**

In our approach, we use the naming conventions presented in Table 2. The TTCN-3 types corresponding to HL7 messages and segments are postfixed appropriately, e.g., *_Message_Type*, *_Segment_Type*. Similarly, the group identifiers are postfixed by *_Group_Type*. The fields of messages are postfixed by *_Segment* and possibly a number in case that the HL7 message contains several segments with the same name. The HL7 tables are mapped in TTCN-3 as charstring based types postfixed by *_Table*.

Module structuring rules. The structuring rules are related to the grouping of TTCN-3 definitions into groups and modules. This can be realized in many ways. One approach is to group *by categories.* The definitions of the same category are grouped together, e.g., types in a group of types, templates in a group of templates. Another approach is to group *by libraries*, which are constituted by TTCN-3 modules. The reusable definitions which are at the same time also general enough to apply to different test suites should be grouped into libraries.

5 Conclusion

In this paper we investigated issues in electronic health care systems where the quality can be substantially improved. Based on the successful application of TTCN-3 testing methodologies in domains like telecommunication, automotive, we analyzed the applicability of TTCN-3 for testing HL7 based applications. This approach allows us to build a TTCN-3 test bed for medical applications, whose communication means are based on HL7.

After developing a principal approach for testing HL7 based medical applications with TTCN-3, we outlined the method to develop appropriate test specifications. The first step in the definition of TTCN-3 test specification is the development of the type system. For that, we developed a conceptual mapping between HL7 version 2.5 data formats [HL7 08] and TTCN-3 types and put it already to practice along an ongoing project [Pro 08].

An additional target of our work is to automate not only the test execution, but to automate also at least parts of the test generation. Hence, we investigated test data aspects, which are for information systems like HL7 based applications are of particular importance. We conceived a method of generating TTCN-3 templates out of specified

types and existing pools of HL7 messages. The next step will be the generation of test cases out of real use case scenarios as defined for the electronic health care domains by the IHE standard for the work of the laboratory, radiology, and alike [IHE 08].

In long term, the standardization of TTCN-3 test suites for HL7 based system is a further target. Furthermore, there is also the requirement to extend our TTCN-3 based test framework towards testing non-HL7 based health care systems as typically a mixture of standards and technologies is used for the realization of medical applications.

References

[Eic 05] M. Eichelberg, T. Aden, J. Riesmeister: A Survey and Analysis of Electronic Health care Record Standards. ACM Computing Surveys, Vol. 37, No. 4, December 2005, pp. 277–315

[IEE 90] IEEE Standard Computer Dictionary: A Compilation of IEEE Standard Computer Glossaries, IEEE, 1990

[IHE 08] IHE International: Integrating the Health care Enterprise. http://www.ihe.net/ (2008)

[HL7 08] HL7: HL7 Standard Version 2.x. https://www.hl7.org/library/bookstore/ (2008)

[Din 08] G. Din, D. Vega, I. Schieferdecker: Automated Maintainability of TTCN-3 Test Suites based on Guideline Checking. The 6th IFIP Workshop on Software Technologies for Future Embedded & Ubiquitous Systems (SEUS 2008). Springer. October 2008

[ETS 05] European Telecommunications Standards Institute (ETSI): ETSI Standard es 201 873-1 v3.1.1 (2005-06): The Testing and Test Control Notation Version 3; Part 1: TTCN-3 Core Language. Sophia-Antipolis, France, 2005.

[TRI 05] European Telecommunications Standards Institute (ETSI): ETSI Standard (ES) 201 873-5 V3.2.1 (2007–02): Methods for Testing and Specification (MTS); The Testing and Test Control Notation version 3; Part 5: TTCN-3 Runtime Interface (TRI). Sophia-Antipolis France (February 2007)

[NEM 08] NEMA: Digital Imaging and Communications in Medicine (DICOM). http://medical.nema.org/. 2008

[ISO 04] ISO/IEEE: ISO/IEEE 11073, Point of Care Medical Device Communication Standards, http://www.iso.org/iso/iso_catalogue/catalogue_tc/catalogue_detail.htm?csnumber=36347 . 2004

[EST 05] Euro: CMM-Assessments. Munich, 1999

[ANS 08] American National Standards Institute (ANSI) http://www.ansi.org/. 2008

[Test 08] Testing Technologies : TTworkbench – an Eclipse based TTCN-3 IDE http://www.testingtech.com/products/ttworkbench.php. 2008

[Jea 04] D.-M. Jeaca, G. Din, A. Rennoch: Importing XML Schema datatypes into TTCN-3, Proceedings of 3rd Workshop on Systems Testing and Validation (STV04). 2004

[Sch 04] I. Schieferdecker, G. Din: A Meta-model for TTCN-3. M. Núñez et al. (Eds.): FORTE 2004 Workshops, LNCS 3236, pp. 366–379, 2004.Springer-Verlag Berlin Heidelberg 2004

[Pro 08] ProInno: Test Next Generation Medical Systems (TestNGMed). EUREKA financed german Project (2008-2009)

[sep 08] sepp.med: .getmore: UML based Test Generator. http://www.seppmed.de/getmore.65.0.html. 2008

[Jab 04] S. Jablonski, R. Lay, C. Meiler, S. Müller: Process Based Data Logistics: a Solution for Clinical Integration Problems. Data Integration in the Life Sciences. Springer, ISBN 978-3-540-21300-0, 2004, pp. 31-46

[Sne 07] R. Snelick, P. Rontey, L. Gebase, L. Carnahan: Towards Interoperable Health care Information Systems: The HL7 Conformance Profile Approach. IESA 2007

[IHE 08] Integrating the Health care Enterprise (IHE). IHE Technical Framework Volume II Transactions (Scheduled Workflow); http://www.ihe.net/tf/. 2008