Advancing HL7 v2 to New Heights: A Platform for Developing Specifications, Test Plans, and Testing Tools

Robert Snelick[1]

[1]National Institute of Standards and Technology (NIST)

Abstract

Development of HL7 v2 data exchange interface specifications has long been problematic, plagued with ambiguous and inconsistent requirement specifications. This situation leads to potential misinterpretation by implementers, thus limiting the effectiveness of the specification and creating artificial and unnecessary barriers to interoperability. Likewise, the ability to test implementations effectively for conformance to the specifications is hindered. The current approach of specification development and test plan creation relies on word processing tools, meaning implementers and testers must read and interpret the information in these documents and then translate it into machine-computable requirements and test assertions. This approach is error prone—a better methodology is needed. We present a set of productivity tools in an integrated platform that allow users to define and constrain HL7 v2 specifications and to develop test plans that result in machine-computable artifacts. A testing infrastructure and framework subsequently uses these artifacts to create conformance testing tools automatically. We present and demonstrate the utility of a platform for developing specifications, writing test plans, and creating testing tools. The value proposition of this end-to-end methodology is explained for authors writing HL7 v2 specifications, for developers implementing interfaces, and for testers creating validation tools.

Keywords

Conformance; Healthcare Data Exchange Standards; Healthcare Information Systems; Interoperability; Specification Development Tools; Testing Tools.

Correspondence to:

Robert Snelick
National Institute of Standards and Technology (NIST)
100 Bureau Drive Stop 8970 Gaithersburg, MD 20899, USA
robert.snelick@nist.gov

1 Introduction

For 30 years, HL7 (Health Level 7) Version 2 (v2) has been the predominant standard used for the exchange of healthcare administrative and clinical data. Healthcare information systems use the HL7 v2 protocol to develop standardized interfaces to connect to and exchange data with other systems. HL7 v2 covers a broad spectrum of domains including Patient Administration, Laboratory Orders and Results, and Public Health Reporting. The base HL7 v2 standard [1] is a framework that contains many message events, and for each event it provides an initial template (starting point) that is intended to be constrained for a specific use case. The application of constraints to a message event is referred to as profiling [2,3]. For example, the VXU V04 (Unsolicited Vaccination Record Update) message event is a generic template for communicating information about a patient's immunization related

events. The base message template is composed of mostly optional data elements. For a given use case, e.g., Send Unsolicited Immunization Update for the US Realm [4], the message template is "profiled". That is, elements can be constrained to be required, content can be bound to a set of pre-coordinated codes, and so on. The base message event (e.g., VXU V04) that has been constrained for a particular use (e.g., submitting immunization events) is referred to as a conformance profile[1]. An implementation guide is a collection of conformance profiles organized for a workflow (e.g., submitting, acknowledging, querying, and responding to/for immunization events). In this example, four conformance profiles exist, each with different message events; one for submitting an immunization event, for sending an acknowledgment, for querying for an immunization history, and for providing an immunization history. To date, HL7 v2 implementation guides have been created using word processing programs, which has resulted in ambiguous and inconsistent specification of requirements. This practice has hindered consistent interpretation among implementers, which has created an unnecessary barrier to interoperability.

We present an end-to-end methodology and platform for developing specifications (implementation guides), writing test plans, and creating testing tools in the HL7 v2 technology space [5]. The platform includes three key foundational components:

- A tool to create implementation guides and conformance profiles
- A tool to create test plans, test cases, and associated test data
- A testing infrastructure and test framework to build testing tools

A key to the approach is that the "normal" process of creating implementation guides, test plans, and testing tools is "reversed". Instead of creating requirements using a natural language and subsequently interpreting the requirements to create test plans and test assertions, the requirements are captured with tools that internalize the requirements as computable artifacts.

Figure 1 illustrates a high-level overview of the methodology. Domain experts develop use cases, determine the message events that correspond to the interactions in the use cases, and then proceed to define the requirements. Using the methodology, they accomplish these tasks by entering this information into the Implementation Guide Authoring and Management Tool (IGAMT). During this process, the domain experts constrain the message events according to the requirements needed by the use case. Section 2 will elaborate more on this process and on the details of how the requirements are constrained. The output of IGAMT is a set of artifacts that are represented in Word, HTML, and XML formats. The complete implementation guide, including the narrative and messaging requirements, can be created in IGAMT and then exported in Word or HTML. Such formats are suitable for ballot at standards development organizations such as HL7 or IHE (Integrating the Healthcare Enterprises [6]). In May 2017, two HL7 v2 implementation guides that were generated by IGAMT were submitted for ballot. Each conformance profile can be exported as XML[2]. The XML format contains all the messaging requirements in a machine-computable representation, which is the most important aspect of IGAMT, since the XML conformance profiles have many uses including a computable definition of the message interface, message validation, test case and message generation, and source code generation.

[1] Also, referred to as a message profile.

[2] The XML format is defined by NIST and is publicly available but is not yet standardized. NIST intends to propose the format to HL7 for adoption. Additionally, there is no relationship between this format and other HL7 profiling formats such as the Templates Implementable Technology Specification (ITS) standard and FHIR.

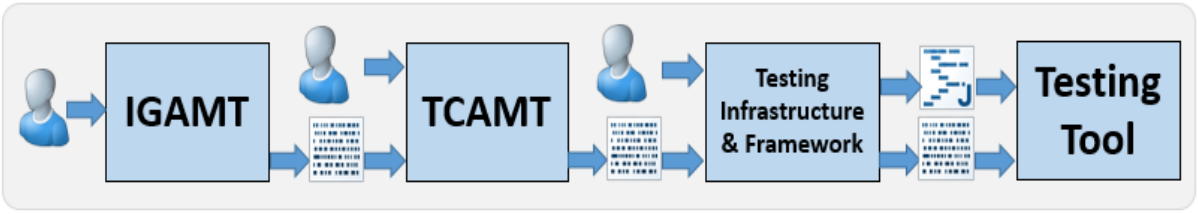

Fig. 1. NIST HL7 v2 Standards Development and Testing Platform Overview

The XML conformance profiles can be imported into the Test Case Authoring and Management Tool (TCAMT). TCAMT is used to create targeted test cases for interactions (profiles) defined in the implementation guide. The output is an additional set of constraints in an XML format. The entirety of the output generated from IGAMT and TCAMT is called a "resource bundle"[3].

The NIST platform includes a testing infrastructure of common utilities used for testing, such as a message validation engine, along with a testing framework that provides various testing tool components, such as a communication framework and a profile viewer. Testing Tool instances are then created using both the testing infrastructure and framework components as well as the resource bundle output generated from IGAMT and TCAMT.

The NIST platform allows end users to create conformance testing tools by means of a set of productivity tools. This streamlined approach can greatly reduce today's problems with conformance test tools. These problems include: tools often don't exist, they are expensive to build, they are difficult to update in a timely fashion, they are not adaptable for local refinements, and their time to market is lengthy. Additionally, the platform provides value through enforcing consistent and rigorous rules for requirements specifications.

The remainder of this paper explains the NIST platform in more detail in the context of how it can be applied in real-world use case settings. We first describe how IGAMT is used to define and constrain conformance profiles. One important aspect is the application of recently developed methods and best practices for requirements specification. Additionally, a brief overview of the validation process is given. Next, an explanation of how a set of targeted test cases are created in TCAMT is provided. In Section 4 we discuss a testing infrastructure and framework components. Next, an overview of the resulting test tools and how they are created is presented. Finally, there is a discussion on how the platform supports testing capabilities beyond the scope of the HL7 v2 interoperability specification. One goal of this paper is to inform the reader about the ease with which HL7 v2 implementation guides, test cases, and testing tools can be created using the NIST platform compared to the current laborious methods used today.

2 IGAMT

IGAMT [5] is a tool used to create HL7 v2.x implementation guides that contain one or more conformance profiles. The tool provides capabilities to create both narrative text (akin to a word processing program) and messaging requirements in a structured environment. Our focus in this paper is on the messaging requirements.

[3] Is not related to a resource bundle in FHIR (Fast Healthcare Interoperability Resources).

IGAMT contains a model of all the message events for every version of the HL7 v2 standard. Users begin by selecting the version of the HL7 v2 standard and the message events they want to include and refine in their implementation guide. For example, the message events VXU^V04, ACK, QBP^K11, and RSP^K11 are used to create eight conformance profiles in the immunization implementation guide [4]. Each message event is profiled (constrained) to satisfy the requirements of the use case. The QBP and RSP message types are used more than once to specify different uses.

Rules for building an abstract message definition are specified in the HL7 message framework, which is hierarchical in nature and consists of building blocks generically called elements [1]. These elements are segment groups, segments, fields, and data types (i.e., components and sub-components). The requirements for a message are defined by the message definition and the constraints placed on each data element. The constraint mechanisms are defined by the HL7 conformance constructs, which include usage, cardinality, value set, length, and data type. Additionally, explicit conformance statements are used to specify other requirements that can't be addressed by the conformance constructs. The process of placing additional constraints on a message definition is called profiling. The resulting constrained message definition is called a conformance profile (also referred to as a message profile). An example of a constraint is changing optional usage for a data element in the original base standard message definition to required usage in the conformance profile.

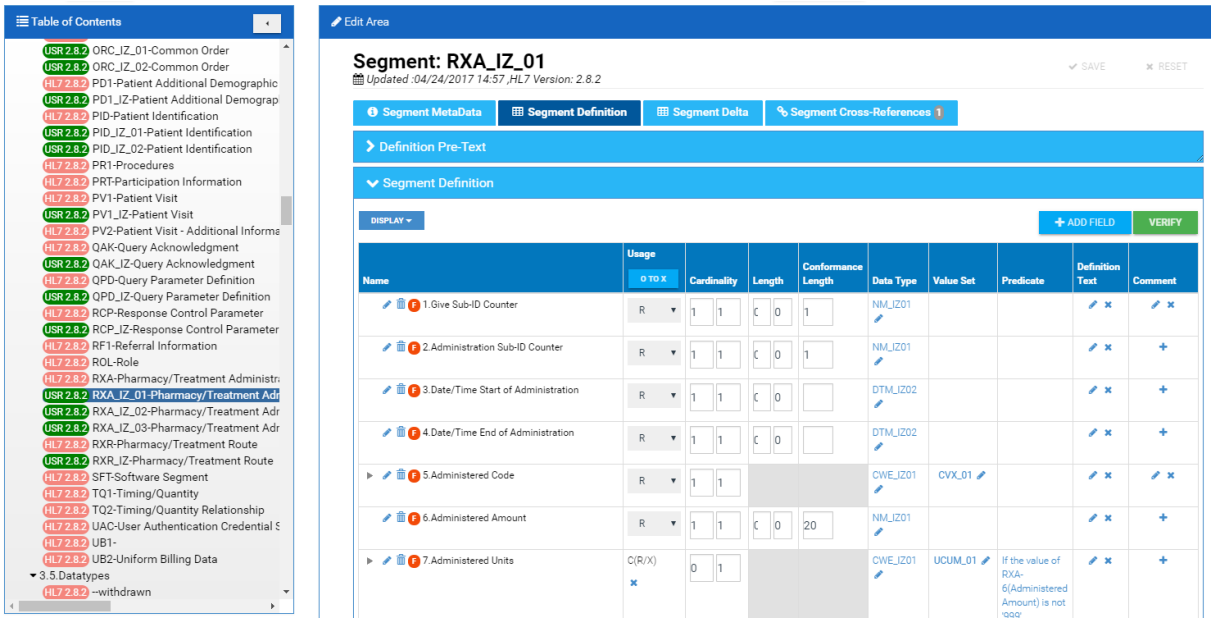

Fig. 2. IGAMT Screen Capture: Navigation and Segment Profiling View

IGAMT provides, in a table format user interface, the mechanisms to constrain each data element at each level in the structure definition. The rows of the table list the data elements according to the structure definition being constrained (segments, fields, and data types). The columns list the conformance constructs that can be constrained for a data element, including the binding to a value set. Figure 2 shows a screen capture of the navigation and the segment profiling panels. On the left-hand side, the user can select the object to edit. The right-hand side displays the list of fields in the segment and the requirements that can be specified for the field.

One key philosophy of IGAMT is the capability of creating and reusing building block components. These lower level building blocks can be used to create higher level constructs efficiently. The building blocks include data type flavors, segment flavors, and profile components. A base data type can be constrained for a given use; the resulting data type is called a data type flavor (or data type specialization). A given base data type may have multiple data type flavors. These flavors can be saved in libraries and reused as needed. A similar process applies to creating segment flavors.

A profile component represents a subset of requirements that can be combined with other profiling building blocks. One such example is the definition of a profile for submitting immunizations. The Centers for Disease and Control and Prevention (CDC) creates a national level profile, however, individual states may have additional local requirements that can be documented in a profile component. Only the delta between the national and local requirements is documented in the profile component. Combining the national level profile and the state profile component yields a complete (composite) profile definition for a given state. Another example is for the case of sending laboratory results and reportable laboratory results to public health. The use cases are very similar. The reportable laboratory results have additional requirements; therefore, a profile should be created for sending laboratory results, followed by a profile component for reportable laboratory results. A composite profile for the public health use case can be created by combining the profile and the profile component. This design principle provides a powerful and effective approach for leveraging existing profiles and profile components [2].

A utility for creating and managing value sets is also provided. Specific value sets can be created and bound to data elements. For example, a base HL7 v2 table can be cloned and modified ("constrained") to create a value set for a specific use, thus enabling more granular value set bindings [2]. Instead of binding an entire HL7 v2 table to an element (typical practice), a value set containing only codes relevant to that element for a particular use is specified. Using this approach, multiple value sets are derived from a single HL7 v2 table, which provides clear requirements for implementers. Mechanisms for creating value set libraries are provided to promote reuse.

2.1 Improved Requirements Specification

In the effort to create conformance test tools for the Office of the National Coordinator (ONC) certification in support of the US Centers for Medicare and Medicaid Services (CMS) Meaningful Use (MU) program, it quickly became apparent that the HL7 v2 specifications named in the ONC rule were ambiguous, under-specified, and inconsistent. This made it difficult to create rigorous, comprehensive, and meaningful test tools and test cases to adequately validate vendor implementations for the ONC stated goal of enabling interoperability. If implementers can interpret and implement requirements in different ways, interoperability is impeded. To improve this situation, NIST worked closely with the specification authors and other stakeholders to gain clarity and subsequently co-published addendums and errata. This effort revealed deficiencies in the mechanisms for specification of requirements and approaches for creating implementation guides. As a remedy, new and improved methods for specifying requirements emerged along with a set of best practices. IGAMT incorporates these methods and encapsulates, automates, and simplifies how the requirements are specified. Table 1 in the Appendix provides a list of the most important methods, concepts, and best practices for improved specifications (beyond current practices).

2.2 IGAMT Message Model and Validation Process

IGAMT has an internal model of all HL7 v2 messages for each version of the standard (Figure 3). HL7 v2 publishes the standard in human readable text documents. Message definitions and accompanying structures are *codified* into a data base, which is available from HL7. IGAMT reads the data base and converts the message definitions into the IGAMT message model. The message model is the anchor on which all IGAMT functions and features are based. IGAMT reveals the model via a graphical user interface (GUI) where the user can constrain the message as needed. The user interface displays panels for the Message, Segment, Data Type, Value Set, Profile Components, Condition Predicates, and Conformance Statements. IGAMT exports the constrained message definition (a profile) as an XML profile instance. IGAMT ensures that the XML profile instance adheres to the rules of the Profile Schema. Validation is performed by validating a message instance against the constraints defined in the XML Profile. The validation engine interprets the requirements as documented in the XML Profile and makes assertions against the message instance accordingly. A Validation Report is generated. The validation process forms the basis of the conformance test tools.

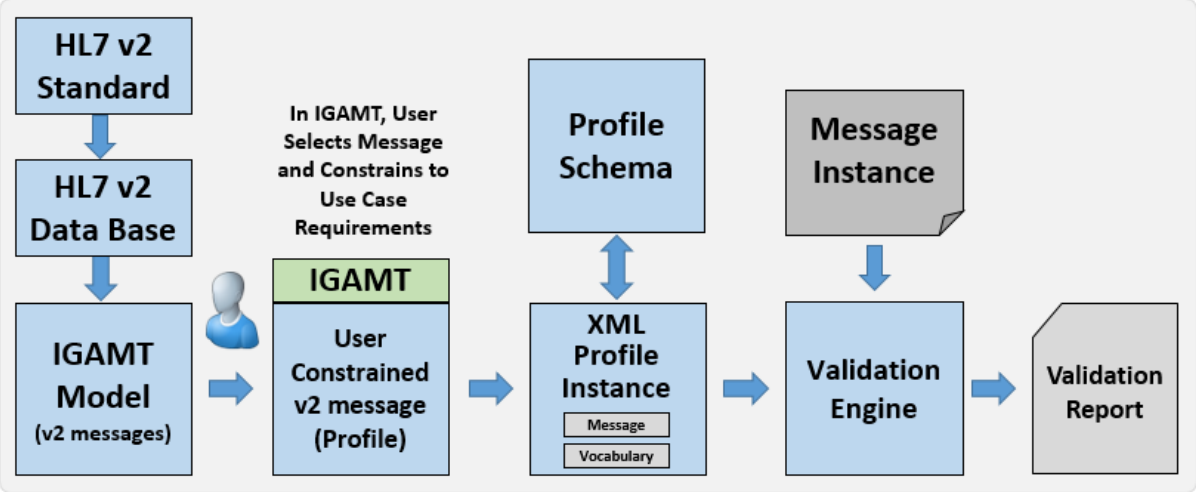

Fig. 3. IGAMT Message Model and Validation Process

3 TCAMT

TCAMT [5] is a tool used to create HL7 v2.x test plans that contain one or more (typically many) test cases. Key features in TCAMT include test plan creation (narrative and computable), IGAMT XML profile import, HL7 v2 message creation and import, constraint editing, constraint and messaging templates, and multiple export formats. A test case can consist of one or more test steps. A test step can be an HL7 v2.x interaction or a manual step such as visually inspecting the contents of an application's display screen. Each test case and test step can consist of a test description, pre- and post-conditions, objectives, evaluation criteria, and additional notes and comments. Test steps for an HL7 v2.x interaction contain an HL7 v2 message (with specific data) that aligns with the XML conformance profile created from IGAMT[4].

[4] Not necessarily conformant data; invalid data may be used in the testing process

Targeted test cases are critical for assessing the capabilities of a system. TCAMT allows domain experts to create test cases (that include example messages) for certain scenarios and capabilities. Test cases provide context, which expands the scope of testing. Without context, a validation tool cannot test a message exhaustively to all requirements specified in the implementation guide. For example, elements with "required, but may be empty (RE)" usage, elements with "conditional usage (C)", or elements with cardinality greater than "1" cannot be assessed without targeted tests. A message that is validated against the requirements of a conformance profile without any provided context is called "context-free testing". A message that is validated against the requirements of a conformance profile and with a provided context is called "context-based testing" [2]. The test cases provide context, and TCAMT is a tool that allows users to create the test cases.

A key design component in TCAMT is its use of the XML profiles created in IGAMT as a foundation. The message definition defined in the profile provides the foundation such that data associated with each message element of interest can be specified. TCAMT also allows the user to enter additional assertion indicators based on what they want to test. For example, for an element with a usage of "RE", the user can provide data that are expected to be entered into the sending system for the element and can select an assertion indicator. There are several assertion indicators that could be selected, for example, "presence". In this case, if the user provides test data and selects the indicator of "presence", a constraint is generated by TCAMT and is provided to the validation. For elements with "RE" usage, the element must be supported by the system-under test (SUT), but in a given message instance the element may not be populated. For this construct, the tester wants to ensure that the implementation has, in fact, included support for the element.

In a context-free environment, the absence of data in a message is not a conformance violation for elements with "RE" usage. However, in the example test case described above, data were provided and a presence constraint was specified. Now, when a message created for this test case is validated, the additional constraint triggers an assertion for the presence of data for this element. This method is one way to determine support for the element.

Via TCAMT, the user can create an unlimited number of test cases and test a broad spectrum of requirements. Other constraint indicators can be used to test for specific content or for the non-presence of an element. Additionally, test data can be provided to trigger conditional elements. In other instances, support for certain observations may need to be ascertained. In such cases, test data for specific observations (e.g., in an immunization forecast, the vaccine group, earliest date to give, and due date) can be provided, requiring the message instance to contain an OBX segment for each observation. The test case might be set up to expect certain LOINC[5] codes to ensure each observation (capability) is implemented by the system. TCAMT provides the mechanisms to conveniently and consistently create test cases. Output from TCAMT provides the additional constraints that are interpreted by the validation engine.

4 Testing Infrastructure and Framework

NIST has built an HL7 v2.x testing infrastructure and framework to aid in the process of creating conformance testing tools. The testing infrastructure provides a set of services utilized by the test tool framework to build specific instances of tools. A test tool can be built for a specific need or to be a general-purpose tool to handle multiple implementation guides

[5] Logical Observation Identifiers Names and Codes

and profiles. The latter tool is a web application where a user can upload implementation guides, conformance profiles, and test plans to "create" a test tool. The test tool is "built-on-the-fly" and can be generated as a by-product "for free" once the XML profile and associated artifacts have been created (in IGAMT and TCAMT). This process allows domain experts to "build" the test tool. Alternatively, the framework can be leveraged, customized, and installed locally. Using the framework, developers can choose to create customized, specific, or general-purpose web application conformance test tools, and they can access the validation via web services or incorporate validation via a JAR (Java Archive) file or source code. Regardless of the use, the platform can significantly improve the quality of implementation guides, assist in the creation and maintenance of test plans, expedite the stand-up of a validation tool, and, overall, reduce the cost and time of the entire process.

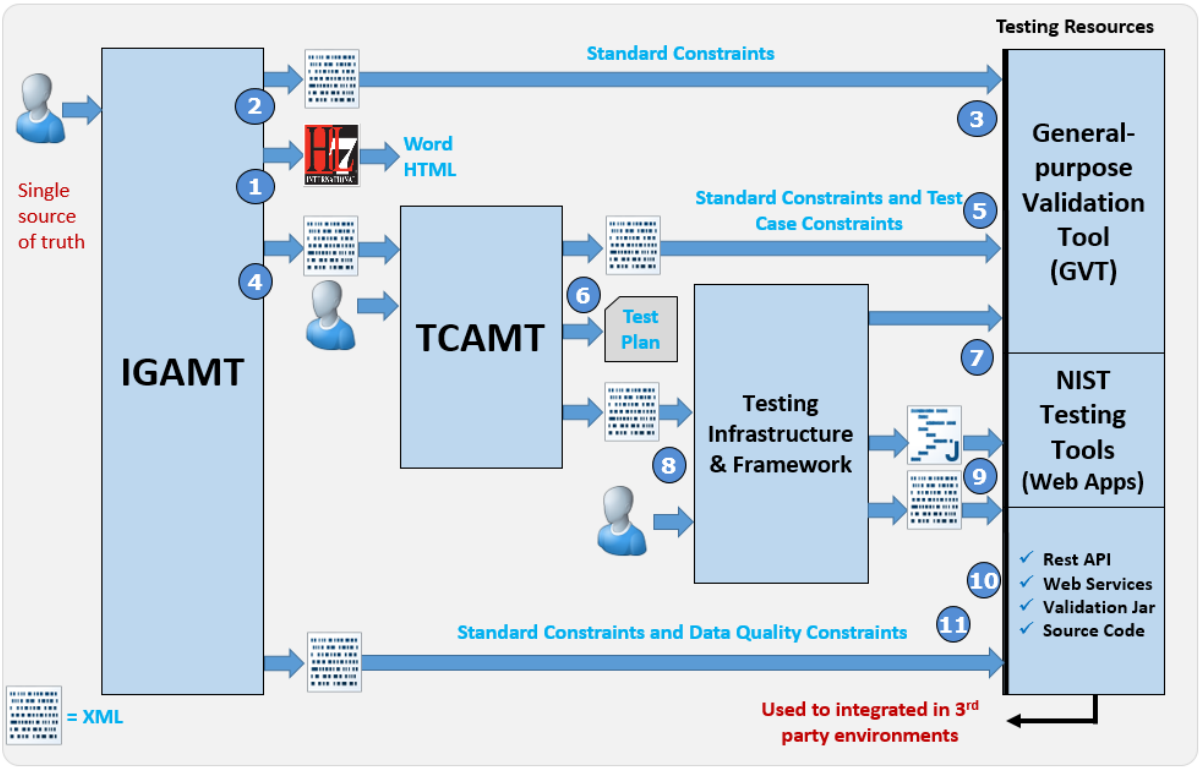

Figure 4: NIST HL7 v2 Standards Development and Testing Platform Architecture

Figure 4 shows in more detail the end-to-end methodology and platform. A key design principle is that there is a single source of truth in the creation of implementation guides and test plans. Modifications are made in one place and are propagated to associated services, utilities, and tools. IGAMT is a tool used by domain expert authors to define requirements for interface specifications. Human readable (1) and machine computable (2) artifacts are exported. A context-free conformance test tool is automatically generated when the IGAMT XML profiles are loaded in the general-purpose validation tool (3). At this level, validation is based on the technical requirements defined in the profile. No context is associated when validating the message instance against the requirements defined in the profile. This type of validation is called context-free testing.

Point (4) shows the XML Profile as input into TCAMT. Test scenarios provide a context, that is, a real-world story with associated data. Additional constraints are generated from having context. The profile and context constraints are loaded into the general-purpose validation tool

to create a context-based validation tool automatically (5). Point (6) indicates a human readable export of the Test Plan.

Point (7) indicates that the testing infrastructure and framework components are used as the basis for the general-purpose validation tool. The general-purpose validation tool is itself a tool that takes as input the resource bundle (XML Profile, TCAMT constraint file, etc.) to automatically generate a conformance test tool. Points (8) and (9) indicate the process by which developers can leverage the testing infrastructure and framework to create customized conformance test tools. Point (10) indicates that validation can be accessed via other methods that allow a user to integrate it into their local environments. The platform provides access to the tool validation via REST and web services. Additionally, the validation JAR and source code are available. Point (11) indicates that additional constraints can also be included that go beyond the scope of typical interface requirements. These can include data quality business rules, for example, ensuring that a vaccine dose reported is consistent in terms of the manufacturer, lot number, and date given. More on this topic is given in Section 6.

5 Conformance Test Tools

As shown, conformance testing tools are built using the testing infrastructure and framework, the IGAMT-produced conformance profiles, and the TCAMT-produced test plan. Testing tools are web-based applications that can support both context-free and context-based validation [5]. In addition to performing message validation, the tools provide a browse-able view of the requirements for each conformance profile. In the context-based mode, the test story, test data, and an example message are provided for each test step.

In the context-free mode, the user simply selects the conformance profile to validate against and then imports the message. The validation is performed automatically and a report is given. In the context-based mode, the user selects the test step and imports the message to validate. The test tool sets the validation to the conformance profile linked to the test step, performs the validation, and provides a report. In both modes, a tree structure of the message is shown on the left panel of the validation screen and can be used to inspect the content of individual data elements.

Test plans can be executed in non-transport mode and transport mode. Non-transport mode provides an interface to upload (cut/paste or load file) a message into the validation edit box. Transport mode allows an application to connect to the test tool to exchange messages interactively. The test tool can act as an initiator or responder as directed by the test plan. Various transport protocols are supported including MLLP and SOAP. Test Cases can also include manual test steps in addition to automated test steps that contain an HL7 v2 message exchange.

6 Requirements beyond the Interface Specification

The intent of HL7 v2 is specifically scoped to defined requirements for exchanging data between applications. The specifications typically do not impose any requirements on how the data are processed. Other specifications, in conjunction with the interface specification may specify such requirements (e.g., IHE integration profiles and functional requirements specifications). In real world settings, exchange partners need to account for more than just conformance to the exchange requirements. Data quality, business rules, and functional requirements are necessary to satisfy the desired outcome of the use case scenario.

Mechanisms to define such requirements, and testing support that can verify that the complete workflow is implemented as intended, are beneficial.

The generic constraint generation utility in IGAMT can be used to create data quality constraints. Certain business rules can be applied to a message to determine if it meets the requirements necessary for incorporation by the receiver. A simple data quality rule for reporting an immunization record is that the date of administration must be after the date of birth. This constraint likely is never given in an HL7 v2 interface specification, however, data quality rules such as these are important at the local level. Users can create these rules in IGAMT that provide additional validation (point (11) in Figure 4).

TCAMT can be used to create test cases to test functional requirements. For example, a scenario can be crafted in which three different immunization records for the same patient are created from different providers and sent to an immunization information system (IIS). A subsequent query to the IIS to return a complete immunization history can be performed. The response message can be examined to see if the consolidated record contains the expected combined immunization history. TCAMT provides the capability to create such a scenario and the additional content validation constraints. Testing for invalid (or negative) test case scenarios also can be created. The platform provides the capabilities for the tester to create unlimited test scenarios using convenient and powerful tooling.

7 Conclusion

We presented an end-to-end methodology and platform for developing standards, writing test plans, and creating testing tools in the HL7 v2 technology space. The platform includes three key foundational components: (1) a tool to create implementation guides and conformance profiles; (2) a tool to create test plans, test cases, and associated test data; and (3) a testing infrastructure and test framework to build testing tools. Requirements are captured in IGAMT and exported as conformance profiles. TCAMT is used to create a set of test cases based on the conformance profiles. A conformance test tool is created by combining the validation and associated artifacts with the testing infrastructure and framework.

8 Appendix

Table 1: Methods, Concepts, and Best Practices for Improved Specifications

Concept	Issue	Feature/Improvement
Explicit Condition Predicates	Conditional usage is specified but lacks conditional statement or an explicit conditional statement	Explicit condition predicate with defined format, style, and pre-defined patterns
Condition Predicate True/False Outcomes	Limited True/False outcomes for conditional usage (C and CE only)	Full range of true/false outcomes; for example, C(R/RE) and C(RE/O)
Explicit Conformance Statements	Statements that hinted at being requirements are hidden in narrative sections of the specification	Explicit conformance statements with defined format, style, identification, and pre-defined patterns
Data Type Flavors	Conflated specializations of data type constraints, in-line constraints, un-managed data type flavors	Explicit data type flavor definitions, naming conventions, and style
Data Type Flavor Library	No notion of creating a library of data flavors for reuse by the community at-large	Master set of data type flavors and defined process for user defined flavors; promote consistency and reuse
Segment Flavors	Segments typically are defined to account for requirements for use in more than one	Provide mechanisms to allow specific segment definition via segment flavors,

	message definition—resulting in conflation of requirements	profile components or explicit conformance statements
Profiling Multiple Occurrences	Capability to assign different data type flavors to multiple occurrences to a field element; defined in v2.8	Implemented in IGAMT and in XML profile instance; can vary by "type code", "order", and "one of"
Co-constraints	Missing, inconsistent, or lack of detailed specification of relationship among data element content; typically, in elements OBX-2, OBX-3, and OBX-5	Mechanism to define data element content relationships and dynamic data type flavor mapping for OBX-2 and OBX-5[6]
Value Set Specification	No explicit value set or code table specifications; often the base HL7 or HL7 User table is bound to an element (or elements) with no further constraints	Explicit value set definition creation and value set binding strength
Value Set Profiling	No formal methodology to constrain code systems for specific element binding and use	Explicit value set definition usage indicator for codes and attributes to indicate extensibility and stability
Profile Components	No constructs or methods to define profile building blocks of constraints for reuse	Profile components are introduced to defined a set of arbitrary requirements that when combined with a profile or other profile components create a complete profile (Composite Profile)
Delta Profiles	Complete specifications for closely related use cases	"delta" specifications can be created leveraging the concept of profile components
IG Template	No guidance on what implementation guides should contain	IGAMT incorporates several default templates and export options
Conformance Keywords	Non-existence and inconsistent definition and use of verbs to express requirements	Explicit definition and use of conformance keywords as part of the IG template; based on RFC 2119

Acknowledgements

I'd like to thank the NIST development and analysis team: H. Affo, W. Jung Yub, H. Tamri, I. Mellouli, A. El Ouakili, C. Rosin, N. Crouzier, M. Lefort, and S. Martinez; reviewers S. Taylor and F. Oemig, and finally C. Newman for providing user feedback on the tool suite.

References

[1] Health Level 7 (HL7) Standard Version 2.7, ANSI/HL7, January, 2011, http://www.hl7.org.

[2] *Healthcare Interoperability Standards Compliance Handbook*. F. Oemig, R. Snelick. Springer International Publishing Switzerland, ISBN 978-3-319-44837-4, December 2016.

[3] *Principles for Profiling Healthcare Data Communication Standards*. R. Snelick, F. Oemig. 2013 Software Engineering Research and Practice (SERP13), WORLDCOMP'13 July 22-25, 2013, Las Vegas, NV.

[4] HL7 Version 2.5.1 Implementation Guide for Immunization Messaging; Release 1.5, October 1, 2014. http://www.cdc.gov/vaccines/programs/iis/technical-guidance/downloads/hl7guide-1-5-2014-11.pdf

[5] *NIST Resources and Tools in Support of HL7 v2 Standards*. http://hl7v2tools.nist.gov/

[6] Integrating the Healthcare Enterprise (IHE). http://www.ihe.net

[6] For example, based on different codes in OBX-3, different data type flavors of the same base data type can be specified in OBX-2 that indicates the requirements in OBX-5. This enables precise requirements definition.