# Testing Environments for Assessing Conformance and Interoperability

**R. Snelick**[1] **and L. Gebase**[1]
[1]National Institute of Standards and Technology (NIST), Gaithersburg, MD, State, USA

**Abstract** - *We present a classification framework of testing environments that supports conformance and interoperability testing of distributed systems. We describe each testing environment and state the applicability and requirements placed on a realization of a testing tool. The broad categories identified include data instance validation, isolated systems testing, and peer-to-peer systems testing. Our targeted systems include those that support data exchange standards. We list and define the types of conformance and interoperability testing that can be conducted and associate the types that can be performed in a given testing environment. We describe and illustrate a conceptual test tool design for each testing environment. The delineation of environments and their testing capacity is intended to facilitate a more structured approach to testing in which the relationship between testing and test requirements is more clearly defined and the capabilities and limitations of testing tools are better understood. In broader terms, this approach should help industry and testing bodies identify and describe the depth and scope of their testing endeavors.*

**Keywords:** Conformance; Data Exchange Standards; Evaluation Criteria; Interoperability; Messaging Systems; Testing Environments.

## 1 Introduction

We consider testing environments that can be used to support conformance and interoperability testing of distributed messaging systems. We have identified three distinct environments and for each we describe the testing activities that can be performed within an environment. Specifically, the environments are the Data Instance Test Environment, the Isolated System Test Environment, and the Peer-to-peer System Test Environment. The delineation of environments and their testing capacity is intended to facilitate a more structured approach to testing in which the relationship between test requirements and testing, along with an understanding of the capabilities and limitations of testing tools, is more clearly defined. In broader terms it is anticipated that this approach will help industry and testing bodies identify and describe the depth and scope of their testing events.

Recognizing that testing is a complex, multidimensional, and often incremental problem leads us then to consider the use of multiple environments for conducting testing. In what follows, we will examine more closely each of these testing environments. Each environment can be used for conducting a—possibly overlapping—component of conformance or interoperability testing.

Conformance testing is a multi faceted operation that can range from a simple assessment of the validity of a message value to a nuanced determination of a system's reaction to a complex sequence of events. Testing the full range of conformance requirements is not always practical, but the evaluation of specific conformance requirements such as data content may instead be of interest. Ultimately, the goal of conformance testing is to enable interoperability among different implementations. Conducting interoperability testing adds another dimension to the requirements for the testing environment.

A key focus of this document is to provide a clear explanation of what the objectives are in testing. We seek to address the following questions:

- What are the testing objectives?
- What can be tested in each environment?
- What testing environments can be used to meet our testing objectives?
- What requirements do testing environments place on a test tool design and implementation?
- and maybe most importantly, what does testing within an environment tell us about the implementation that was tested?

In addition to addressing these issues, we define and classify conformance and interoperability terms and concepts. We then propose a test hierarchy that can be used to guide and organize the testing plans and environments. We list and define the types of conformance and interoperability tests that can be performed to assess an application. Next we present three testing environments and for each describe the testing activities that can be performed within an environment. We describe each dimension of testing and document the

inter-relationships among each. Testing dimensions include the test modes such as automated and inspection testing, aspects of the test environment, e.g., are the systems under test on-site or will testing be conducted remotely over the Internet. These are important considerations when conducting testing. Finally we summarize with a brief overview of our plans to develop a testing infrastructure for evaluating implementations of healthcare data exchange standards. As a basis for building the testing infrastructure we draw from the structured and modular approach established in this work.

# 2 Definition of Terms

Below we define a set of terms used in this document. In some cases, these definitions may not be universally agreed upon, but establishing common definitions for use in this document will facilitate an understanding of the concepts presented.

**Conformance:** Conformance is defined as the fulfillment of a product, process, or service of specified requirements [1,2]. The concept of conformance is essential to any standard for providing an objective measure of how closely implementations satisfy the requirements defined in the standard.

**Syntactic Interoperability:** If two or more systems are capable of communicating and exchanging data, they are exhibiting syntactic interoperability. Syntactical interoperability is required for any attempts of further interoperability [3].

**Semantic Interoperability:** Beyond the ability of two or more computer systems to exchange information, semantic interoperability is the ability to automatically communicate information and have that information correctly interpreted by the receiving system. To achieve semantic interoperability, both sides must defer to a common information exchange reference model [3].

**Conformance Testing:** The assessment of an implementation to determine if its behavior is consistent with the requirements for behavior defined in a standard or other reference document. The objective of conformance testing is to determine how completely and correctly the requirements of the standard have been met by the implementation. Note that conformance cannot be definitively determined; only a degree of confidence can be derived based on the quantity and quality of tests performed. Conformance testing is *black box* testing in which the details of the implementation are unknown, only the inputs and outputs of the implementation are used for evaluation. [1].

**Syntactic Interoperability Testing:** The assessment of an implementation to evaluate its capability to syntactically interoperate with one or more distinct systems.

**Semantic Interoperability Testing:** The assessment of an implementation to evaluate its capability to semantically interoperate with one or more distinct systems.

**Testing Environments:** The testing environments establish the method of operation in which a conformance or interoperability test is being conducted. We have identified three such testing environments: *Data Instance Test Environment*, *Isolated System Test Environment*, and *Peer-to-peer System Test Environment*. For purposes here, we only define the testing environment necessary to enable interactions between the test system and the SUT or test object—i.e., the underlying test model. We don't discuss the hardware or software commonly associated with testing environments; this is implementation dependent.

**System-under-Test (SUT):** The software system that is being testing.

**Test Object:** The object that is being tested. The object may or may not have been created by the SUT. For example a message, document, or an application.

**Test Artifact:** Material used to test a test object. Testing requirements are derived from the test artifacts. For example, a standard, a specification, or implementation guide.

# 3 Test Organization Hierarchy

An important aspect in test planning is the organization and strategy that is administered by the tester. It is advantageous to have a comprehensive and structured testing plan to ensure that testing objectives are met, that is, to ensure that testing covers the evaluation of all aspects of the system that should be tested. Defining and applying an established organizational hierarchy at the outset of test suite development is important. It helps in the creation, management, and maintenance of test cases. In addition it can provide the necessary documentation for demonstrating the purpose and traceability of test cases. This is especially important when the SUT fails a test. **Figure 1** illustrates one such test organization hierarchy. What follows is an explanation of each component in the hierarchy.

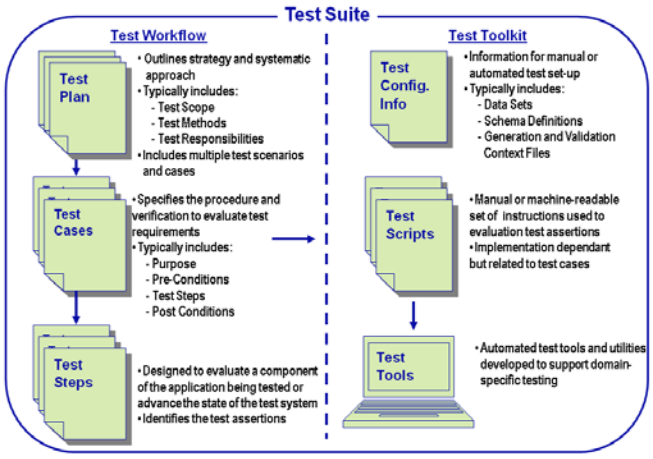


**Figure 1: Test Organization Hierarchy**

**Test Suite:** A collection of test cases and the associated machinery and process to execute the test cases. A *test suite* may include documentation of the overall testing strategy, configuration requirements, and may control, via tooling, the invocation of test cases that make up the test suite.

**Test Plan:** A plan that outlines the strategy and systematic approach to be used to evaluate an SUT. A *test plan* should minimally include test scope, test methods, and test responsibilities.

**Test Case:** A *test case* is a sequence of test steps, executed in the order in which they are listed unless a control mechanism is present that alters the order. Test steps are the smallest unit into which a test case can be divided. Test steps include commands for starting applications, changing state, or accessing data. The most important test steps, though, define assertions. Assertions may be used to evaluate SUT behavior, or they may be used to ensure that all actions necessary to carry out the next step in the test case have completed. A test case should include a test purpose, pre-conditions, and post conditions. The test case language may support control structures that allow the test case workflow to be repeated.

A test case may, in general, be thought of as consisting of procedure and verification information. The procedure information describes how to get the test items from the SUT during the test, while the verification information determines whether or not the SUT satisfies the test requirements by way of processing the test items [4].

**Test Script:** A script is synonymous with test case however it is sometimes distinguished from test case by referring to the set of instructions for a particular test that will be carried out by an automated test tool.

**Test Scenario:** A scenario is also synonymous with test case and is often used when the test case involves a multistep business workflow.

**Testing Steps:** Conformance and interoperability testing can generally be conducted most effectively by dividing the testing into a number of steps. Each step in the testing process is designed to evaluate a component of the application being tested or advance the state of the test system. Test steps are components of a test case.

**Test Assertion:** An atomic statement designed to evaluate an element or state of the SUT. Assertions are derived from the test criteria. For example, "the application shall populate the *administered code* element with a value identified in the vaccine administered code value set described in the CVX vocabulary."

**Test Criteria:** Criteria used to construct test case assertions that are designed to evaluate the SUT. The test criteria are derived from the test specification (i.e., the standard) that the system is being testing against.

**Test Configuration Information:** Configuration data necessary for executing the test cases that makes up the test suite. Typically includes data sets, schema definitions, generation and validation context files, and actor definitions and addressing.

**Test Tool:** An application that can be used in conducting conformance or interoperability testing. A test tool can interact with users or the SUT. The test tool executes the test scripts.

**Test Agent:** An implementation of functionality that—at a minimum—enables all interactions necessary for testing to be carried out with the SUT.

# 4 Test Evaluation Types

Conformance and interoperability testing can be effectively conducted by dividing the testing into a number of categories based on the evaluation specific criteria. The criteria address the evaluation of one or more components of the application being tested or one or more aspects of the application's behavior.

The system components and behavioral aspects to be tested include the following:

- Documents and messages.

- Transport protocol usage.
- Application behavior.
- Interoperability.
- Semantic interoperability.

Application behavior testing is designed to test conformance to the data exchange standard requirements and to test functional requirements. In the following, we elaborate on the methods for evaluating each criterion introduced above.

***Data Content Validation:*** For validating documents and messages, the data content of the message or document is evaluated independently of the means that was used to create the content. In other words, this is a procedure that is strictly designed to assess data content, and does not attempt to measure application conformance. Content validation is designed to evaluate a test object for adherence to the specification that defines valid instances of the test object.

***Data Content Conformance Testing:*** Documents and messages produced by an application may also be validated for adherence to a specification. The procedure for content conformance testing does not differ from the data content validation procedure. But since the data content is associated with the application that produced it, in this case, the application can also be evaluated.

***Transport Protocol Usage Testing:*** To test the application's use of the transport protocol, an evaluation of how an application packages and extracts messages and documents is made. This type of testing assesses an application's use of an allowed communications protocol. That is, on sending, a check to ensure the application correctly packages messages before sending them is performed, and on receiving the application is checked to make sure it correctly extracts the message content from the package it was received in.

***Testing Conformance to the Data Exchange Standard:*** One element of evaluating application behavior is evaluating an application's reaction to valid and invalid variations in data content. This type of testing examines application responses to received messages. For example, an application's reaction to variations in message structure and content are evaluated with this type of testing. Messages or documents are sent with variations in the encoding characters and valid and invalid content. A valid response from the receiving application is an indication that it processed a valid message (document) or recognized an invalid message (document).

***System Behavior Conformance Testing (from Application Functional Requirements):*** To test application behavior, an evaluation of the application's interpretation of message content is made. The evaluation is made based on the actions taken by the application when new messages are received. The application's response to user requests is also evaluated. Generally this type of testing consists of sending the application valid messages and evaluating the responses returned by the application for correct semantic content (when the SUT is a server). When the SUT is a client application it will be instructed to create a message or document, usually via a user interface. In order to conduct System Behavior Conformance Testing a test scenario is created in which a sequence of orchestrated transactions are composed to test adherence to specific functional requirements.

***Syntactic Interoperability Testing:*** Interoperability testing is designed primarily to establish that two applications are able to successfully exchange data. No evaluation of the application's processing of the data is made with this type of testing.

***Semantic Interoperability Testing:*** This type of testing is the second phase of interoperability testing. If two applications establish that they are capable of exchanging data, semantic interoperability testing attempts to determine if they correctly process the data exchanged as intended.

# 5 Testing Environments

In the previous sections we introduced a set of distinct testing types that can be used to evaluate the components of an application essential to its overall functioning. Below we introduce the conceptual models in which these testing types can be carried out.

**Data Instance Test Environment:** This testing environment is composed of one or more testing tools and the test object. The test tool uses the specification to evaluate the test object.

- ***Data Content Validation Testing***
  - No Context; not associated with an application
  - e.g., the content of an XML document evaluated against a schema
- ***Data Content Conformance Testing***
  - Test object is evaluated and is identified as having been produced by a specific application

**Isolated System Test Environment:** This testing environment consists of a SUT and testing tools designed to interact with the system. The SUT may interact with test agents or validation testing tools.

- ***Inherent Data Instance Testing Activities***
- ***Transport Protocol Usage Testing***

- ***Testing for Conformance to Data Exchange Standard***
  - Test range of conformance requirements
  - Valid and invalid instances
  - Multiple test cases conducted
- ***System Behavior Conformance Testing (from Application Functional Requirements)***
  - Test scenario is created to orchestrate a sequence of transactions

**Peer-to-Peer System Test Environment:** This testing environment consists of one or more vendor systems and a testing infrastructure designed to interact with and evaluate one or more of the vendor systems.

- ***Inherent Isolated System Testing Activities***
- ***Syntactic Interoperability Testing***
- ***Semantic Interoperability Testing***

### 5.1 Data Instance Testing Environment

In the Data Instance Testing Environment a test is conducted with a test object and a testing tool. The goal is to perform evaluation of data content against a set of conformance rules. The tool may be a validation service. An example of data instance testing is validating an HL7 message against an HL7 conformance profile [5].

The means of delivery is not important; any means by which the object can be delivered to the testing tool is satisfactory. That is, a file may be used to deliver the data or the test object may be delivered using the underlying transport protocol.

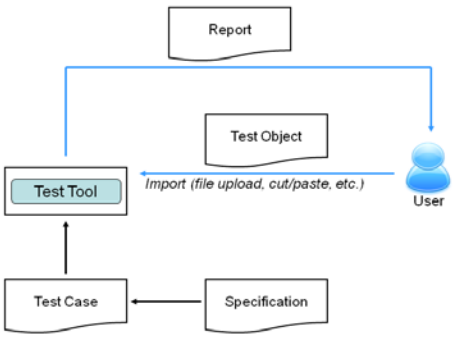


**Figure 2: Data Instance Testing Environment**

When no application is associated with a data object, no application conformance evaluation can be made, only an evaluation of the object's validity can be made. We define this procedure as *Data Content Validation*; there is no context and the test object is not associated with an application. When the application that produced the object is identified, a conformance evaluation of the application can be rendered. We define this procedure as *Data Content Conformance Testing*; in this case there is context and the application that produced the object is known.

The objective of data instance testing is to assess the adherence of the test object to the conformance rules defined by a specification. The specification will typically include rules defining the structure or syntax of the data, along with semantic rules for interpreting the data. The syntax may be defined by a formal grammar using a BNF (Backus-Naur Form) notation, an XML schema, or another notation. In all cases, precise evaluation of data items for compliance with the syntactic requirements will be possible. If a less formal mechanism is used for specifying the data syntax, determining syntactic correctness should still be possible, although ambiguous cases are possible. Generally, the specification will also include allowed values for instances of the data object and these will be evaluated for compliance with the requirements of the specification.

**Figure 2** depicts a test system environment for the Data Instance Testing Environment. In this illustration the test object is delivered manually (i.e., the user access point is via an uploaded file or cut-and-paste of the test object). Replacing the user icon with a system is another instance of this test environment in which the test object is delivered directly via the system that created the test object. The Data Instance Testing Environment validation component is leveraged and is an integral part of the test environments to follow.

### 5.2 Isolated System Testing Environment

In the Isolated System Testing Environment a test is conducted with the SUT and a test tool. The SUT may interact with test agents and validation testing tools. Since conformance testing is the main objective in using this model, data content conformance testing is subsumed in this model. Additionally, the Isolated System Testing Environment supports Transport Protocol Usage Testing, Testing for Conformance to the Data Exchange Standard and System Behavior Conformance Testing (from Application Functional Requirements).

Transport Protocol Usage Conformance Testing asserts that an application correctly implements an allowed communications protocol. That is, on sending, the application correctly packages messages before sending them, and on receiving the application correctly extracts the message content from the package it was received in.

The Testing for Conformance to the Data Exchange Standard evaluation type tests that an application correctly sends messages and that the application correctly responds to all messages, valid and invalid, that

the application receives. All responses are evaluated to ensure that they adhere to the requirements defined in the relevant specification. This type of testing evaluates an application's reaction to variations in message structure and content. Messages that the application is expected to support are sent to the application, changes are made in the encoding characters used, and valid and invalid variations in content are sent; often boundary conditions are tested and optional elements are included. The criterion for evaluation is receipt of an application response indicating that it processed a valid message or recognized an invalid message. No semantic evaluation of the response is made. These tests evaluate a range of conformance requirements from a specification and typically involve multiple test cases that can be executed in a batch mode.

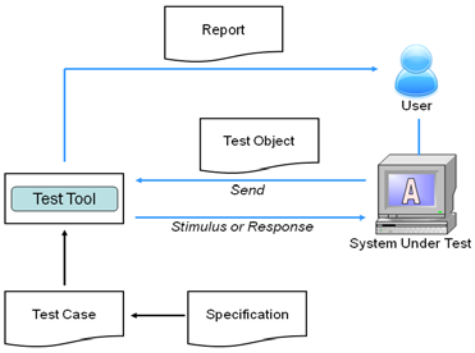


**Figure 3: Isolated System Testing Environment**

The objective of System Behavior Conformance Testing (from Application Functional Requirements) is to evaluate the behavior of an application. As with the previous testing method, it generally consists of sending the application valid messages and evaluating the returned responses. In this case, though, the response message is evaluated for their semantic content (when the SUT is a server). When the SUT is a client application it will be instructed to create a message or document, usually via a user interface. In order to conduct System Behavior Conformance Testing a test scenario is created in which a sequence of orchestrated transactions are composed to probe a certain functional requirement. In the behavioral analysis, typically a few data values are examined to determine validity.

Isolated system testing typically accounts for the majority of testing that is conducted. Once a system has successfully undergone conformance testing, interoperability testing usually proceeds more easily.

**Figure 3** depicts the Isolated System Testing Model in which the testing methods described in this section are conducted. The SUT interacts with a test tool designed to assess conformance of the SUT. In this model there is direct communication between the test tool and the SUT via the communication protocol. The test tool may include functionality of an application that an SUT would typically interact with in an operational environment. Often multistep tests are conducted in this environment involving numerous interactive communications between the test tool and the SUT.

### 5.3 Peer-to-peer System Testing Environment

Testing is conducted among a group of vendor systems. A vendor system may interact with a test tool or other vendor systems. Peer-to-peer system testing is designed to test interoperability among one or more systems. Conducting conformance testing prior to interoperability testing can greatly facilitate the ease with which interoperability testing can be performed. Peer-to-peer system testing may include some or all of the conformance testing described for isolated system testing. When conformance testing is conducted in advance, peer-to-peer testing then specifically targets syntactic interoperability testing and semantic interoperability testing.

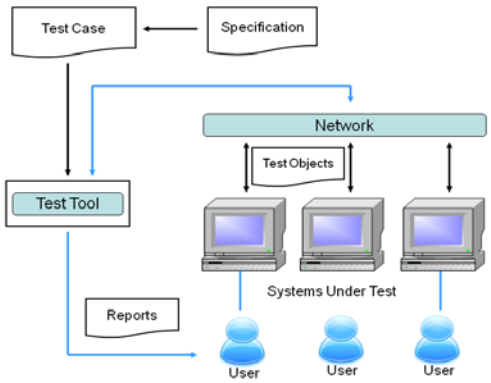


**Figure 4 : Peer-to-Peer System Testing Environment**

The objective of syntactic interoperability testing is to establish that two applications are able to successfully exchange data. No evaluation of the application's processing of the data is made with this type of testing. Semantic interoperability testing is the second phase of interoperability testing. If two applications establish that they are capable of exchanging data, this type of testing attempts to assess if they also correctly process the data exchanged.

**Figure 4** illustrates the Peer-to-peer System Testing Environment. This environment poses different and significant challenges in testing from what we have examined earlier. In this environment data exchange is made among a group of systems. The test environment no longer has direct interaction with the systems under test.

Here an intermediary or a proxy can be employed to intercept, log, and route messages to their intended destination. The conformance test cases that were developed for Isolated System testing can be leveraged in Peer-to-peer testing. The abstract test cases could be identical, however, execution of the test steps, configuration requirements, and assertion assessment will differ. By ascertaining that the conformance requirements are now met in an environment where the SUTs are interacting we can make a declaration of the interoperability capabilities of the systems.

# 6 Additional Testing Considerations

An important aspect to consider in testing is the testing mode. Automated testing, inspection testing, and hybrid testing are orthogonal to the testing environment (model) used. Automated testing mode implies that the evaluation of the SUT's behavior is automated. Inspection testing involves a human monitor. The hybrid mode is a combination of automated and inspection testing. The goal is to achieve automated testing whenever possible. However, in some circumstances it may not be possible or the cost of automation is too high—for example, a test case stating "display the patient's medical record on your EHR screen for the patient with id MR88408". The functional requirements of the SUT often will dictate the testing method.

Another important consideration in the testing environments is the operational environment. Tests can be conducted on-site in a closed network or conducted over the Internet. Network access and firewalls are a few of the issues that need to be addressed in an environment in which the SUTs are intended to communicate over the Internet.

# 7 Summary

We have examined a testing approach for evaluating an application's adherence to requirements defined in a standard or other specification. Our approach has been to divide the testing requirements into a number of categories that can be separately addressed. By decomposing the requirements in this way, we have been able to formulate a modular and structured approach to comprehensively testing an application. To accomplish this we have defined a testing strategy based on this decomposition and the development of a testing plan, which can be realized through the implementation of a test suite built up of a number of test cases. We have shown that by taking this approach it is possible to effectively evaluate systems for both conformance and interoperability.

To date, we have developed a number of tools for supporting testing in these environments. Additionally, as a basis for building a testing infrastructure for evaluating implementations of healthcare data exchange standards we draw from the structured and modular approach established in this work. The testing infrastructure will support a spectrum of healthcare data exchange standards. It will be consist of a set of reusable components that can be assembled to build out specific test tools. Key functionality includes generation and validation of test objects, test agents, and the communication infrastructure [6, 7]. A test harness will be employed to orchestrate the modules. Additional components to support test case development will be built. The concepts presented here will help shape the testing infrastructure design. This will enable us to further evaluate the techniques we have described with the intention of refining and improving our evaluation procedures. We also plan to investigate new techniques to further automate the testing, thereby minimizing the effort required to test wherever possible.

# 8 References


[1] ISO Reference - ISO/IEC 17000 Conformity assessment - Vocabulary and general principles, first edition 2004-11-02.

[2] Glossary of Conformance Terminology, Interoperability and Conformance Technical Committee, OASIS. http://www.oasis-open.org/committees/ioc/glossary.htm

[3] Institute of Electrical and Electronics Engineers. IEEE Standard Computer Dictionary: A Compilation of IEEE Standard Computer Glossaries. New York, NY: 1990.

[4] *Agile test framework for Business-to-Business Interoperability*. J. Woo, N. Ivezic, H. Cho. Internal NIST draft paper—not yet published.

[5] *Towards Interoperable Healthcare Information Systems: The HL7 Conformance Profile Approach*. R. Snelick, P. Rontey, L. Gebase, L. Carnahan. Enterprise Interoperability II: New Challenges and Approaches. Springer-Verlag, *London Limited 2007 pp*. 659-670.

[6] "*Conformance Testing and Interoperability: A Case Study in Healthcare Data Exchange*" L. Gebase, R. Snelick, M. Skall. 2008 Software Engineering Research and Practice (SERP08), WORLDCOMP'08 July 14-17, 2008, Las Vegas, NV.

[7] *A Framework for testing Distributed Healthcare Applications*. R. Snelick, L. Gebase, G. O'Brien. 2009 Software Engineering Research and Practice (SERP09), WORLDCOMP'09 July 13-16, 2009, Las Vegas, NV.