Architectural Recommendations for e-Business Global Interoperability Test Bed

Nenad IVEZIC[1], Jungyub WOO[1], Yunsu LEE[1], Hyunbo CHO[2], Oriol BAUSA[3], Jacques DURAND[4], Tim FOWLER[5], Yildiray KABAK[6], Christine LEGNER[7], Oliver STEIN[8]

[1]NIST, 100 Bureau Drive, Gaithersburg, MD, 20899, USA
Tel: +001 301 975-3536, Fax: + 001 301 975 4482,
Email: {nivezic|jungyub.woo|yun-su.lee@nist.gov}
[2]Department of Industrial and Management Engineering, Pohang University of Science and Technology, San 31, Hyoja, Pohang 790-784, Korea
Tel: +82 54 279-2204, Fax: +82 54 279-2870, Email: {hcho@postech.ac.kr}
[3]Invinet Sistemes, Ribes 31, 08013, Barcelona
Tel: +34 639465171, Email: oriol@invinet.org
[4]Fujitsu Computer Systems, 1250 East Arques Ave., Sunnyvale, CA 94085, USA
Tel: +001 408 746 6000, Email: JDurand@us.fujitsu.com
[5]Automotive Industry Action Group, 26200 Lahser Rd., St. 200, Southfield, MI 48034, USA
Tel: +001 248 358-3570, Fax: +001 248 358-3570, Email: {fowlertj@aol.com}
[6]Dept. of Computer Engineering, Middle East Technical University, İnönü Bulvari, Ankara, 06531, Turkey
Tel: +90 312 2405598, Fax: + 90 312 2101259, Email: yildiray@srdc.com.tr
[7]HEC Lausanne, Universite de Lausanne, Internef, bureau 127.3, CH-1015 Lausanne, Switzerland,
Tel: +41 21 692 3432, Email: christine.legner@unil.ch
[8]Ludwig-Erhard-Strasse 52-56, 72760 Reutlingen, Germany,
Tel: +49 (7121) 144 89-60, Email: oliver.stein@ipoint.de

Abstract: We offer a collection of recommendations to architect a comprehensive and coherent architectural solution for a global interoperability testing framework for eBusiness systems. Requirements from an initial set of three industrial use cases drove the architectural requirements analysis and identification of recommendations. Both functional and non-functional use case requirements were analyzed to arrive at the architectural recommendations.

1. Introduction

While e-Business specifications are implemented and adopted by an increasing number of organizations, it is still cumbersome to achieve and demonstrate full compliance to standard specifications. In large part, this is due to two main issues: (1) achieving e-Business interoperability typically requires that a full set of standards – from open Internet and Web Services standards to industry-level specifications and e-business frameworks – are implemented; and (2) there are only limited and scattered testing facilities available. As the testing facilities are typically provided by one of the standards developing organizations, they have a narrow focus on a particular standard. Particularly, they do not encompass testing of the entire e-Business scenarios, e.g. complex transactions with several partners.

In 2009, Phase 1 of the Global e-Business Interoperability Test Bed (GITB) project was completed [1,2,3]. In that phase of the international project, CEN (The European Committee for Standardization), ETSI (The European Telecommunications Standards Institute), EIC (Enterprise Interoperability Center), NIST (National Institute of Standards and Technology), KorBIT (Korean Business-to-Business Interoperability Testbed), AIAG (Automotive Industry Action Group), and others collaborated to perform a comprehensive analysis of requirements, current capabilities, and risks to enable modular testing infrastructure that addresses testing requirements in e-Business scenarios. The collaborative team also proposed a collection of recommendations for the architectural and governance approach to make such a global testing framework a reality.

In 2011, Phase 2 of the GITB project started with the objective to provide architectural and governance solutions for the global testing framework. The project continues to be supported by international stakeholders who expect the proposed architectural and governance foundation to support future collaborative work in e-Business testing. With participation of experts and stakeholders from Europe, Asia and North America, the workshop team has a potential to deliver a promising architectural and governance foundation.

2. Objectives

This paper describes the recommendations for architecting phase of the Global e-Business Interoperability Test Bed that were identified in Phase 1 and further refined in Phase 2 of the GITB project.

3. Architectural Recommendations

After analyzing three representative use cases – from long distance supply chain, public procurement, and healthcare domains – and current testing capabilities across a number of test beds available, a series of key recommendations have been proposed during Phase 1 [1,2]. These recommendations are refined further in the on-going Phase 2 of the project [4].

Recommendation 1: GITB needs a new test framework.
According to the results of the Phase 1 analysis, the existing test beds and test methodologies do not fully provide the engineering-level functional capabilities required by the initial three use cases. Also, they lack several non-functional capabilities, such as modularity, extensibility, and plug-and-play capability. This means that the existing test beds and test methodologies are difficult to reuse to meet the GITB test requirements. Specifically, there are five cases where the "Union of the Existing Test Beds' Capabilities" does not satisfy the "Union of Use cases' Functional Requirements" [2]: (1) Capability of providing setup information to systems under test (SUTs); (2) Capability of test case customization; (3) Capability of generating a message template from a schema; (4) Capability of employing existing validation engines; and (5) Capability of recovery from errors. These missing requirements are related with the non-functional capabilities, such as extensibility and testing tool governance.

Simple addition of missing functional capabilities to existing capabilities may not guarantee interoperability between added components and existing capabilities. Moreover, the existing test beds and test methodologies were designed for particular standard specification and testing environments, making it hard to apply them for other domain standards and environments.

Hence, to enhance the reuse of existing capabilities, basic issues need to be addressed, such as common architecture, common platform, representation of test cases, and access method. As a result, a new test framework should be developed to satisfy the requirements extracted from the three use cases and to enhance the reuse of the existing capabilities.

Recommendation 2: A GITB test framework should be structured into two parts for efficient development and management: an independent test case model and a test execution model.
A test case encodes a set of conditions under which a test user verifies whether the target SUT works correctly. Test execution method processes and executes each step defined in

the test case. The test execution model guides the implementation of a test bed to validate eBusiness solutions and documents against regulations and agreements between trading partners, while the test case model describes the contents and structure of the test cases.

Most of the existing test case models are tightly coupled with their own test execution model. That is, the existing test cases cannot be interpreted without knowledge of the specific test execution model. This represents a significant roadblock to efficient reuse of testing materials and components: a test case for a specific test requirement should be developed independently of a specific test bed system.

The GITB test framework should consist of both a test execution model and a test case model so that the test case model is independent of the test execution model, making the framework and components more reusable and manageable. That is, once developed by domain experts, other test users should easily reuse a test case even though the test user does not have knowledge of a specific GITB test bed. This should be possible because the GITB test case would have sufficient information about the test procedure and verification requirements for both a specific test bed and SUT(s). In addition, the test case could be used for other test bed systems if the systems conform to the GITB technical requirements.

Recommendation 3: Test case structure and scripting grammar should be standardized.
According to the engineering level functional requirements, most of the functional requirements for test execution depend on the functional requirements of the test case design. For example, the definition of "capability of test preparation and setup" for test execution depends on how "test configuration information" is represented. Therefore, the structure and grammar necessary to represent a test case should be considered first.

Recommendation 4: A GITB test case may be generated using a test case library to enhance reusability.
A test case library is a collection of resources used to develop a test case. Test case libraries contain scripts and data that are a basis for constructing executable test cases. This allows sharing and changing the scripts and data in a modular fashion. The test case library consists of procedure artifacts and validation assertion artifacts.

In order to represent a business testing scenario and procedure, the artifacts make use of the following concepts, as shown in Figure 1:

- Actor: An actor represents a role played in relation to the business by someone or something in the business environment.
- Message: A message is an object of communication. Conceptually, it is a kind of event, and may reference a schema, or conformance profile in order to define the message syntax and structure.
- Transaction: A transaction represents transfer of information (messages) between two or more actors. It generally consists of the initiating Actor, message, and a responding Actor.
- Collaboration: A collaboration is a sequence of transactions between two (or more) actors that have a specific business meaning.

In order to represent a rule or regulation abstractly, a validation assertion artifact is designed. Each procedure artifact is associated with a validation assertion artifact at the corresponding level as follows:

- Message-level Validation Assertion: This is a kind of test assertion related to regulation or rules for a specific message. These assertions are mainly used to check whether the message correctly uses its grammar and structure as defined in a schema.
- Transaction-level Validation Assertion: This is a kind of test assertion related to regulation or rules for a specific transaction, which indicates a message exchanged between specific actors. These assertions are mainly used to check a static, fixed value constraint or a dynamic value check in a single message exchanged between two actors.
- Collaboration-level Validation Assertion: This is a kind of test assertion related to regulation or rules for a specific collaboration which consists of one or more transactions. These assertions are mainly used to check a data correlation specification across multiple messages in collaboration.

A validation assertion describes a rule that needs to be verified using a natural language like English, while a rule script is machine-readable and interpretable by test verification engine. These rule scripts could be reused if the same verification language is used across different test beds. Thus, actual test case library may contain not only validation assertions but also associated rule scripts.

Using a test case library, a test case developer should be able to rapidly generate own specific test cases. This should be possible because the library artifacts do not contain a specific test configuration information that define a test harness and bindings of test components. On the other hand, a core test case is a machine-readable low level description of a test case. A core test case can be derived from a test case library using specific testing environment information such as transport and communication protocol. This approach provides two benefits: 1) A test case library can be developed in a more generic manner without considering test bed system details, and 2) many test cases used for existing test frameworks may be executed by a GITB test bed as an executable test case.

Recommendation 5: The GITB test case model should include specification of test configuration, specification of obtaining test items, and specification of verification of test items.
The specification of a test configuration may include various information about the test mode, the connection between SUTs and test bed, and the details about the human test manager involvement. The specification of the test configuration should consider some of the test modes as listed below:

- Instance/Document Testing: Instance or conformance/unit testing is defined as testing an artifact (e.g., an HL7 V2 message) against the rules defined in the specification. This form of testing does not directly involve a system under test, but rather a testing artifact that was produced by the system under test. Examples of instance testing include validating a CDA document instance against the CDA general rules and document type rules, and validating an HL7 V2 message instance against an HL7 V2 conformance profile.
- Conformance/Isolated System Testing: Isolated system or integration testing involves a single vendor system conducting a test with respect to one or more standard specifications. The vendor system may interact with multiple test agents.
- Interoperability/Peer-to-Peer Testing: Peer-to-Peer or interoperability testing is defined as testing conducted among a group of vendor systems either in a closed-network or remote setting. Peer-to-peer testing uses two or more vendor systems,

but can use one or more standards specifications. An example of interoperability system testing would be the IHE Connectathon.

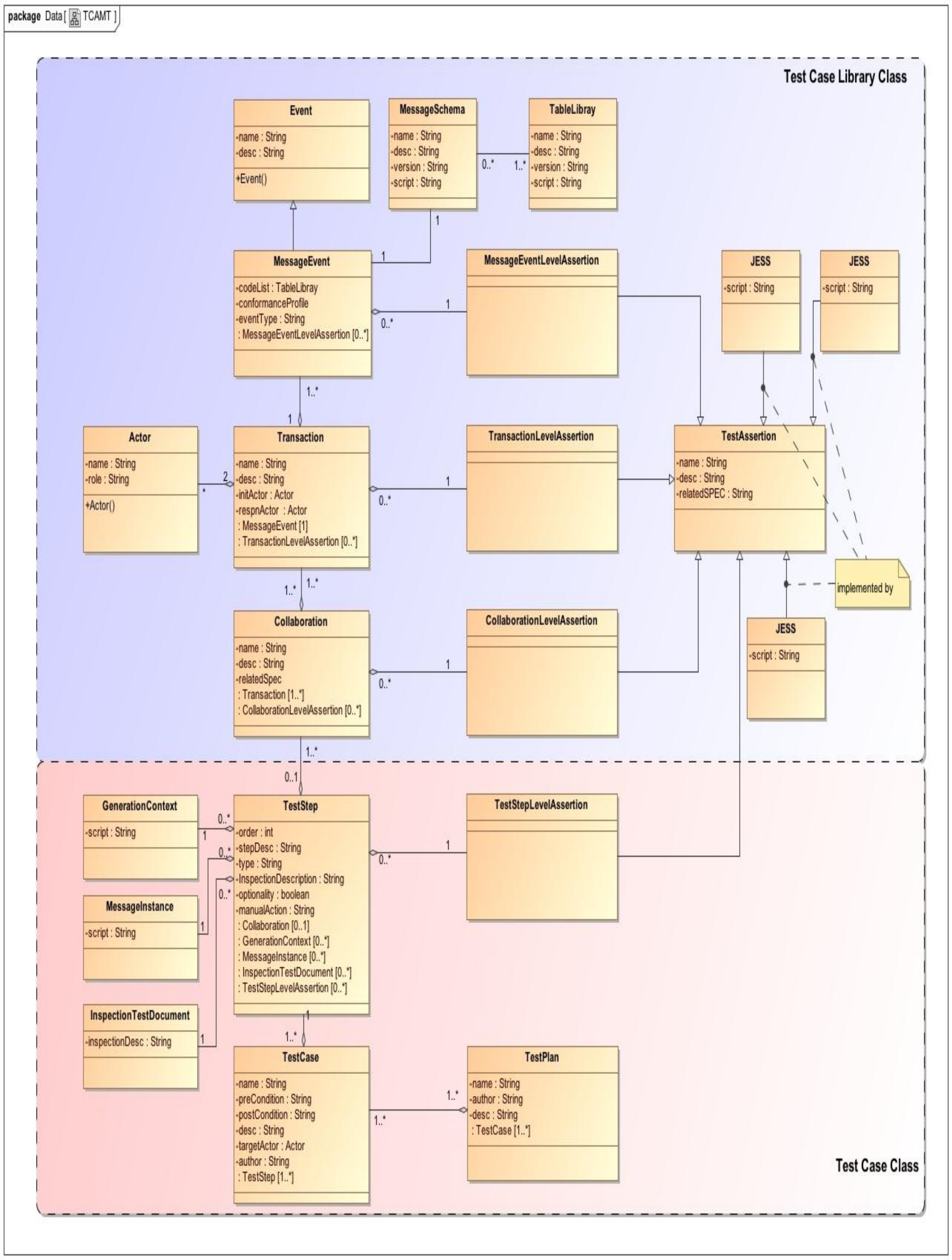

Figure 1: Test Case and its Library Model

A specification of obtaining test items includes description of a specific situation applied to the SUT(s) in order to obtain test items. This specification generally includes the test scenarios and used protocols in the exchange between SUT(s) and a test bed, In particular, the specification should indicate what actor is SUT or test agent (test bed).

The specification of verification of test items contains the expected returns from the SUT(s), against which the actual returns are verified. This specification should indicate the verification language and verifier engine(s) to be used.

Recommendation 6: Test case authoring and management tool should be provided by GITB in order to efficiently generate test cases for the test user.
Typically the test case generation is a time consuming activity because multiple domain experts need to cooperate. An authoring and management tool may help various test case developers save significant design time. The tool should provide services such as version control and life cycle management of both newly created and legacy test cases.

Recommendation 7: GITB should provide a repository management service for the test users to search the test case library and test case scripts.
In order to enhance access and reusability of the test cases, all artifacts in the test case library should be stored in a public repository. In this way, the generated test case should be also registered in the public repository for the efficient management as shown in Figure 2.

A typical test case generation procedure using this repository model is described as below:

1. Analyze test requirements, which come from the standard specification and user-specific requirements.
2. Draw the test procedure from the test requirements.
3. Search relevant procedure artifacts in the test case library.
4. Define the procedure artifacts for the missing parts and then update them in the library repository.
5. Implement required rules scripts for the validation assertion and update them in the library repository.
6. Implement rule script for the test step in the test case.
7. Register the test case in the test case repository.

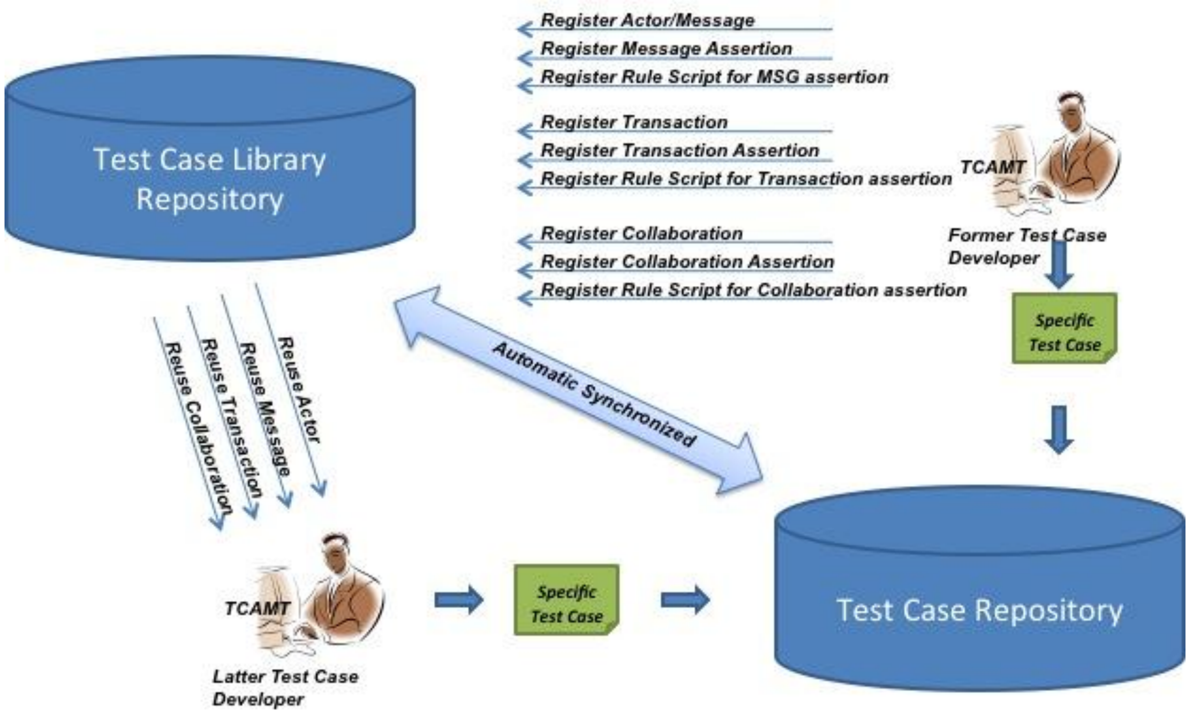
Register Actor/Message
Register Message Assertion
Register Rule Script for MSG assertion
Register Transaction
Register Transaction Assertion
Register Rule Script for Transaction assertion
Register Collaboration
Register Collaboration Assertion
Register Rule Script for Collaboration assertion
Test Case Library Repository
TCAMT
Former Test Case Developer
Specific Test Case
Automatic Synchronized
Reuse Actor
Reuse Message
Reuse Transaction
Reuse Collaboration
TCAMT
Latter Test Case Developer
Specific Test Case
Test Case Repository

Figure 2: Test Case and Library Registry/Repository

Recommendation 8: The functional requirements should be implemented as plug-and play components if they are related to existing standards.
Some testing functional requirements should be implemented to support any standard. For example, the functional requirements for sending and receiving message payloads should be implemented to support existing transport and communication protocols such as ebMS, Web Service, MLLP, EDI Messaging, X12, and AS2. Each protocol should be implemented as a plug-and-play component to enhance configurability of the test bed. The candidate plug-and-play components are as follows: (1) Message exchange component; (2) Message pre/post-processing component (e.g., parsing retrieving values, and transforming values); (3) Message validation component.

Recommendation 9: The functional requirements should be implemented as plug-and play components if they support specific requirements of use cases.
Some testing functional requirements should be implemented to support the specific requirements of use cases. For example, the functional requirements for requesting and storing user information may differ from case to case. The corresponding requirement may be implemented as a proprietary module, but should be plug-and-play capable for the specific test bed. The candidate plug-and-play components are as follows: (1) Test step controller component; (2) Binding component (i.e., binding user information to test case); and (3) Reporting component.

Recommendation 10: The functional requirements should be implemented as infrastructure components if they are neither standard-specific nor use case-specific.
It may be inefficient to implement each functional requirement differently for different purposes. For example, since test preparation, setup, and test step control functions are common to any testing, they should be implemented and embedded into the infrastructure that may be reused in any test bed. The candidate infrastructure components include (1) Test preparation and setup component; (2) Test flow and progress display component; (3) Uploading/Downloading message component; (4) Message generation component (to generate a message template from a schema) and Test data generation component (for a specific message template); and (5) Error recovery component.

4. Conclusions and Future Work

There is an increasing need to facilitate interoperability and develop testing methodologies for e-business scenarios, which requires global collaboration among the institution form different geographic regions. The Global e-Business Interoperability Test Bed project is in a position to deliver an architectural and governance solution to lay a foundation for such global collaboration. The present recommendations represent an initial set of a growing number of directives that, however, can help advance the state of art in testing while making the testing processes both more affordable and more consequential.

5. Acknowledgements

This work has been done in collaboration with the European Commission-funded Global eBusiness Interoperability Test Bed Workshop which runs as a CEN activity. We gratefully acknowledge support of Ms. Barbara Gatti and Mr. Jaap van der Marel.

6. Disclaimer

Certain commercial software products are identified in this paper. These products were used only for demonstration purposes. This use does not imply approval or endorsement by NIST, nor does it imply these products are necessarily the best available for the purpose.

7. References

[1] Global eBusiness interoperability test bed methodologies, GITB, accessed June 2011. Available at http://www.ebusiness-testbed.eu/.

[2] Feasibility Study for a Global eBusiness Interoperability Test Bed (GITB), accessed June 2011. Available at http://www.ebusiness-testbed.eu/publicaties/4765.

[3] Ivezic, N., Cho, H., Woo, J., Shin, J., Kim, J., Abidi, M., Dogac, A., Namli, T., Legner, C., Fischer, F. Towards a Global Interoperability Test Bed for eBusiness Systems. In Procedings of *the eChallenges 2009 Conference and Exhibition*", IIMC International Information Management Corporation, (2009).

[4] Global eBusiness Interoperability Test Bed Methodologies, GITB2, Business Plan, accessed June 2011. Available at http://www.ebusiness-testbed.eu/dynamics/modules/SFIL0100/view.php?fil_Id=1008.