Proceedings of the ASME 2008 International Design Engineering Technical Conferences & Computers and Information in Engineering Conference
IDETC/CIE 2008
August 3-6, 2008, Brooklyn, New York, USA

DETC2008-49807

DEVELOPMENT LIFE CYCLE AND TOOLS FOR DATA EXCHANGE SPECIFICATION

KC Morris **Puja Goyal**

Simon Frechette

National Institute of Standards and Technology
Gaithersburg, Maryland 20899, U.S.A.

ABSTRACT

In enterprise integration, a data exchange specification is an architectural artifact that evolves along with the business. Developing and maintaining a coherent semantic model for data exchange is an important, yet non-trivial, task. A coherent semantic model of data exchange specifications supports reuse, promotes interoperability, and, consequently, reduces integration costs. Components of data exchange specifications must be consistent and valid in terms of agreed upon standards and guidelines. In this paper, we describe an activity model and NIST developed tools for the creation, test, and maintenance of a shared semantic model that is coherent and supports scalable, standards-based enterprise integration. The activity model frames our research and helps define tools to support the development of data exchange specification implemented using XML (Extensible Markup Language) Schema.

1 INTRODUCTION

The motivation for this work comes from experience in working with industries to develop standards for data exchange [11]. We have found that data exchange models or specifications evolve in a fragmented and distributed fashion. To make integration and interoperability more efficient and scalable, the fragmented specifications need to fit into a coherent, semantic model. That is, they need to be logically consistent and contain minimal duplication. Additionally, semantically overlapping data structures should be related or annotated. In our previous work [11], we proposed an activity model capturing the activities involved in creation and maintenance of a coherent, semantic model of data exchange specifications (DES). The focus of this paper is on the tools to support the activities surrounding the development and use of the DES. First a high-level overview of those activities is presented in order to describe the tools and the context in which they are used.

2 DES DEVELOPMENT LIFE CYCLE

The highest-level activity, called the *Manage DES Development Life Cycle* is shown in Figure 1 using IDEF0 [15][1]. This section describes that activity along with the subactivities with particular emphasis on the inputs and outputs as they are indicative of the objective of the activity. The controls are described in some detail where NIST has developed tools to support them in the XML Schema [23] context. Other controls and the mechanisms of the activities are described in more details in other reports [11][13].

The objective of the Manage DES Development Life Cycle activity is to design, create, extend, or modify data exchange specifications for systems integration projects such that data/content level interoperability is achieved over multiple specifications. This will reduce integration costs over the long run. To achieve that reduction, the activity produces and maintains a library of semantically coherent DES as the integration projects evolve.

The inputs to the activity are the *Data Exchange Requirements* (DER) and *Sample Exchange Data*. The DER includes detailed information requirements for integration. The DER may be captured in a number of different data models including class diagrams, database schema, and entity-relationship diagrams. The *Sample Exchange Data* is representative data to be exchanged in the actual integration scenario. The more sample exchange data in hand the better the quality of the DES produced.

[1] In IDEF0, each activity box is defined with the inputs on the left, outputs on the right, controls/constraints on the top, and, mechanisms from the bottom.

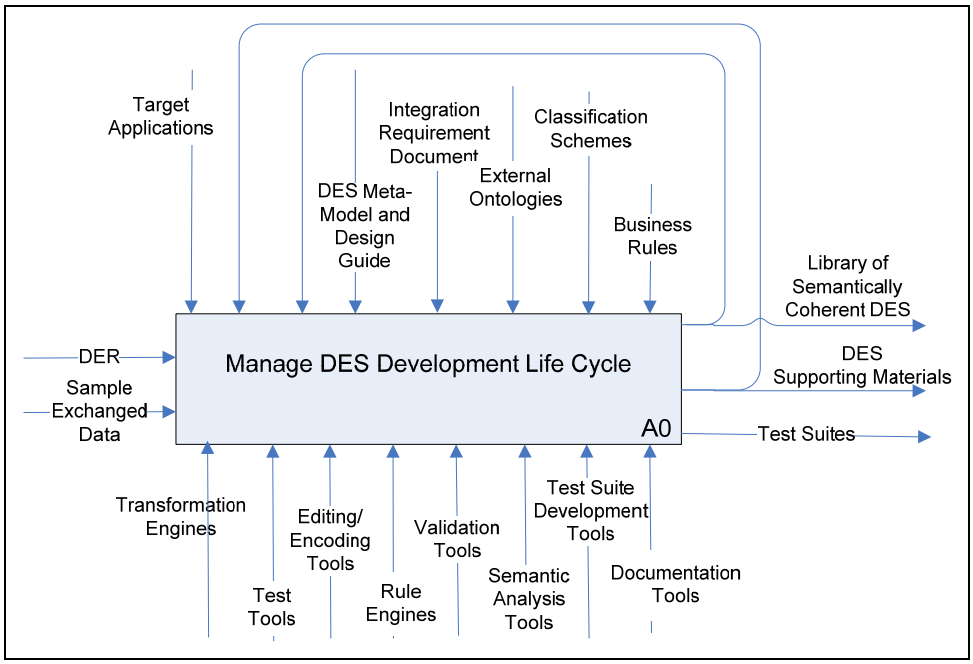

Figure 1: Activity A0 – Manage DES Development Life Cycle

The output of the activity is a *Library of Semantically Coherent DES*. This is a collection of data-interchange terms and structures captured in a computer interpretable representation such as XML Schema. The library may incorporate standards-based data specifications and will presumably include new data specifications as well. Terms and data structures may contain unique semantics, overlapping semantics, or properly annotated duplicate semantics. Overlapping semantics must be clearly represented. Where direct relationships cannot be established internally via the DES normative representation or duplications cannot be eliminated, they must be properly annotated. Additionally, *DES Supporting Material* will be produced and includes information kept along with the DES to help maximize the reusability and comprehensibility of its terms and data structures. Supporting data include, but are not limited to, a table of terms (controlled vocabulary), classification schemes for categorization, DER and integration requirements documents, DES documentation, sample exchange data, and more expressive semantic models.

The final output, *Test Suites*, is important for enabling standards-based integration[2]. Testing is an indispensable activity in systems integration [6, 7]. The test suite provides test data and other materials needed for integration testing.

The manage DES development life cycle activity (*A0*) is decomposed into the five subactivities shown in Figure 2. While this activity model has been developed independently from any particular implementation form, i.e., specifications represented in a particular language such as XML Schema, the tools we have developed support an XML Schema based implementation. As such this section will include references to XML Schema. Interactive systems integration, implementation of translators, model evolution, and retirement are beyond the scope of this activity model. These activities, A1–Discover DES, A2 –Validate DES, A3 – Integrate DES, A4 – Pilot DES, A5 – Register DES, are described in this section.

Discover DES

The DES discovery activity targets reuse of existing DES. Reusing DES is strategically important for minimizing long-term interoperability costs. Typically integration projects first try to identify existing XML Schemas that support their requirements (the DER). If none are found, they may make the decision to build their own XML Schemas. The activity is decomposed into three subactivities: *Select DES for Reuse*, *Extend/Adjust DES*, and *Create New DES* as depicted in Figure 3.

The select DES for reuse activity searches the *Library of Semantically Coherent DES* for models that closely match the new DER; requirement gaps may be discovered from a requirement coverage analysis. This activity is typically executed by a senior technical member or members of the organization, but as the use of integration specifications becomes more ubiquitous the task will become more daunting. We envision having a tool which can intelligently search the library (*Semantic Aware Lookup Tool)* to assist the domain expert or system engineer with this activity. The tool should exploit other information associated with the DER and DES in finding such matches. That information includes text descriptions, integration models, sample data, *Classification Schemes*, and other expressive logical axioms describing the semantics of the DES data elements in *External Ontologies*.

One outcome of the DES selection activity is that the DES can be used as-is. This outcome is preferable, because

[2] We refer to standards-based integration as the integration approach that relies on implicitly agreed upon DES semantics (i.e., complete and formally expressed semantics of the DES is not available).

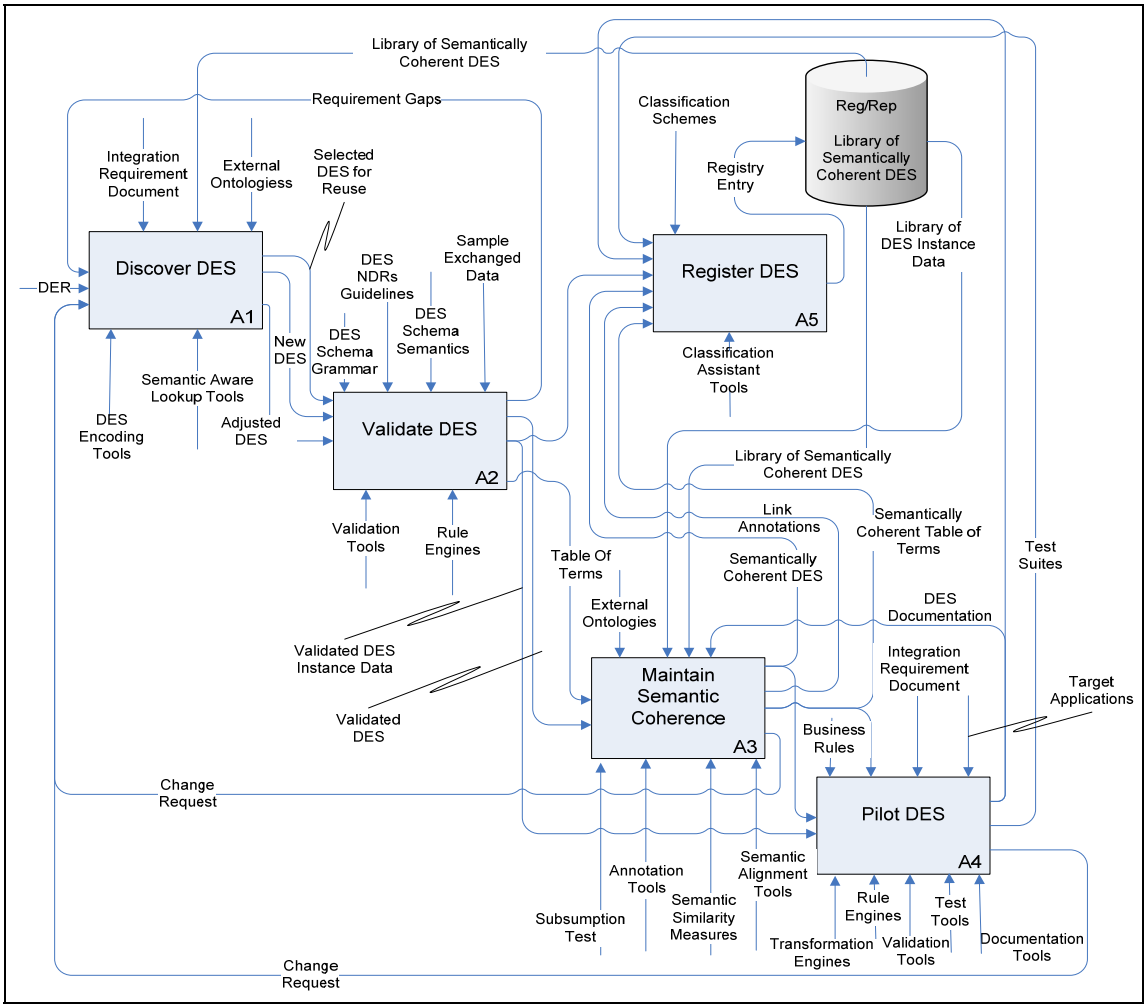

Figure 2: Decomposition of the DES Development Life Cycle

there will not be a new DES to maintain. In addition, the integration will be eased, because the new interface implementation can be readily connected to the same interfaces already implemented.

In many cases, the existing DES from the library only partially supports the new DER. In such cases, a partially reusable DES is extended or is adjusted to meet the uncovered requirements. This is the objective of the *Extend/Adjust DES* activity. An extension is an addition to an existing DES. An adjustment is a direct modification to the reusable DES, such as a relaxation of some constraints to accommodate a new type of data. For uncovered requirements that have no relationship to existing DES, a new DES is created.

Validate DES

The objective of the *Validate DES* activity [13] is to ensure the quality of the DES including that the DES satisfies its requirements as laid out in the discovery activity. Figure 4 illustrates subactivities of the DES validation. The *Validate DES* activity takes as input an initial information specification, e.g., the XML schema, produced by the *Discover DES* activity. Before the schema is deployed it should be tested. Releasing a schema that is not of a high enough quality will result in frustration for both the users and the software developers and could result in failure of the entire project; therefore, the *Model Validation* activity includes tests for quality of design.

Validate DES involves two types of quality validations. The first validation, represented in activity A2.1, is qualify DES. In this activity an XML Schema is tested not only against the standard specification for XML Schemas, xml-schema.xsd [XSD] but also with multiple tools that implement that specification. The XML schema is also checked for compliance with the project's *Naming and Design Rules* (or *NDR*). These rules ensure that modeling practices are used consistently, which enhances the specification's intelligibility tremendously, thereby avoiding confusion during the piloting and implementation phase of the integration project. Naming conventions are a form of design guidelines; however, their importance should not be underestimated and they, therefore, are called out and make use of the *Table of Terms*. The NDR should be established, documented, and enforced as early as possible in the development of the DES to avoid rework.

This activity may seem redundant with the creation subactivity of the Discover DES activity, which has the same

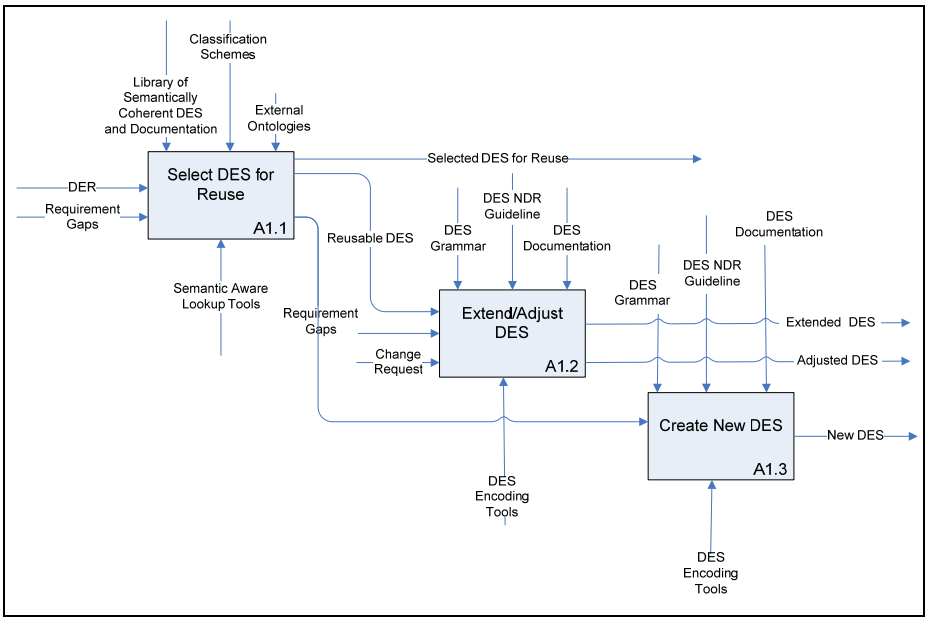

Figure 3: Activity A1 – Discover DES

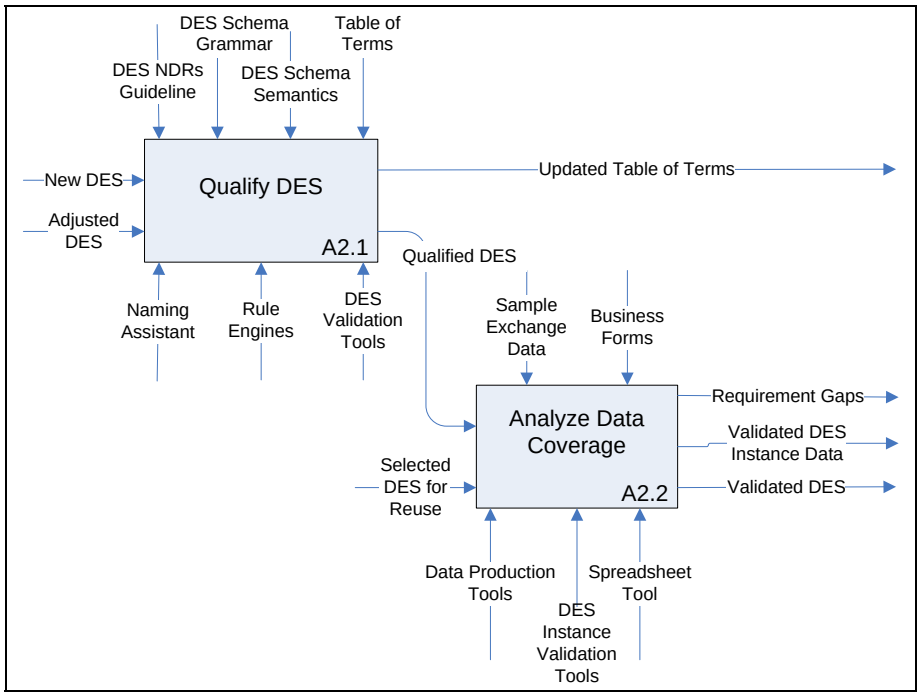

Figure 4: Activity A2- Validate DES

controls; however, it is a very practical step. If the DES has been developed using a specific tool that enforces the grammar and NDR Guidelines then this step may seem unnecessary; however, it is particularly useful when different tools are being used by project partners, as is typically the case in an integration activity. The activity checks that a DES is compatible not only with the tools used in its development but also with others *before* the DES is disseminated widely.

Activity A2.2, *Analyze Data Coverage*, is the second type of validation and ensures that the model meets the original information requirements. The most direct way of doing this is to analyze the relationship between an XML schema and the application data, the *Sample Exchange Data*. Instance data for the DES is gathered and mapped into the XML Schema checking for complete coverage of both the data by the schema and the schema by the data. For this approach, the *Data Production Tools* include data editing and/or data generation tools that generate sample exchange data from target applications in the DES-compliant representation. This may be a manual process often accomplished with the use of a spreadsheet to map from data fields in the systems to be integrated into the XML schema, and vice versa. The output from this activity is identified requirement gaps, which can be fed back into Discover DES activity, at which point the

process is reiterated. In this activity when problems are uncovered in validating the instance data with the XML schema, the problems are often indicative of the problems in XML schema or its supporting material and not just in the instance data. Resolution of the problems should result in improvements to either the exchange specificaion or the supporting documentation to clarify the intention.

Validate DES is an iterative activity the end result of which is a valid schema meeting a given set of quality criteria along with other artifacts illustrating the DES and how it is to be used. These artifacts are the *Validated DES Instance Data* created as a reference from the sample exchange data and the table of terms containing controlled vocabulary and data-element definitions.

Given the current state of the technology many aspects of this activity are automatable. For some parts of the activity commercial tools are available. To better support quality validation NIST has developed the three tools described below to supplement the functionality available in commercial tools:

- Naming Assister [8]. One artifact of the qualify DES activity is a table of terms to be used for naming in the XML schema. An initial table may have been provided by the Discover DES process. NIST has prototyped a tool, known as the Naming Assister, to help with naming. The Naming Assister specifically aids in creating consistent compound names by verifying the construction of these names against a table of allowable terms.

- Quality of Design Tool (QOD) [12]. The QOD Tool provides a repository of design rules and a framework to publish and execute the rules. The tool allows users to create their own rules or select their own set of rules against which to check XML schemas. The database contains tests for rules from several different organizations including, but not limited to, the following specifications:
  - Department of Navy's Naming and Design Rules [4]
  - UN/CEFACT's Naming and Design Rules [21]
  - OAGi's Naming and Design Rules [16]
  - IRS [9]

- XML Validation Page. NIST developed an XML Validation page which allows users to upload XML schema and XML instance files and have them validated using a selection of XML tools. This tool is similar to web pages made available by others with the important distinctions being 1) that it checks the files using multiple, 3rd-party XML tools, and 2) that it contains repositories of XML Schema files for specific projects.

The DES qualification is especially important for reuse of the DES. To prevent future difficulties only a high-quality DES that conforms to the NDR and is in other ways error free should be allowed into the library for reuse by others. This activity ensures that modeling practices are used consistently. It enhances the specification's understandability and helps avoid confusion during the pilot and implementation phases of the integration project.

Maintain Semantic Coherence

The *Maintain Semantic Coherence* activity is important as a long-term interoperability strategy. The objective is to ensure that new schemas and extensions are not only consistent with established design practices as in the previous activity but also are semantically coherent with the growing collection of DES. The activity can be viewed as a monitoring or certification function before the DES and associated artifacts are made available for reuse. The activity ensures (1) that the new, extended, or adjusted DES does not conflict with existing uses; and, (2) new terms or data structures that are semantic duplicates or overlaps with existing ones are not created without proper relationship and documentation. Consequently,

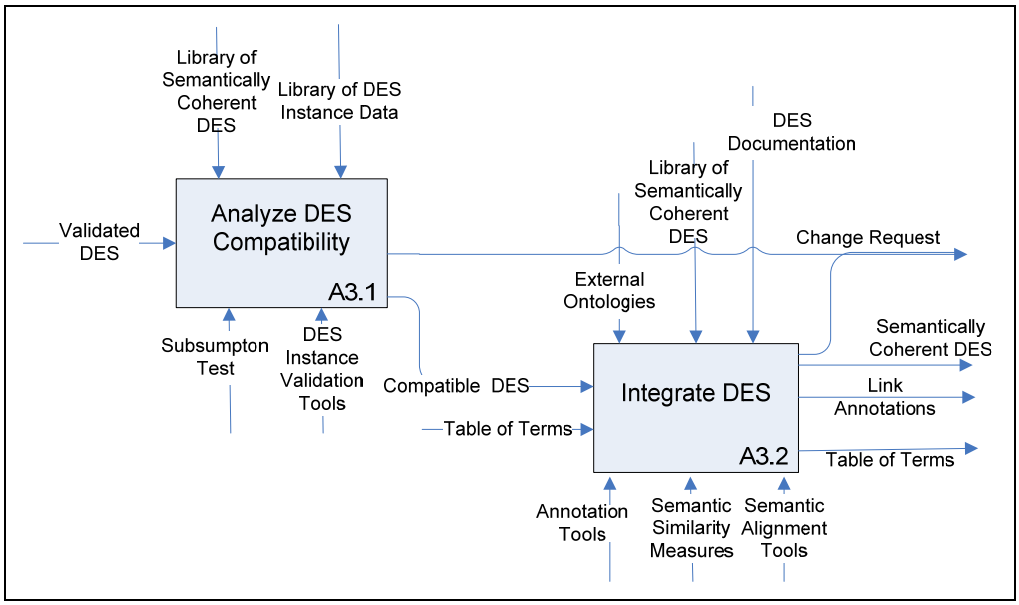

Figure 5: Activity A3 - Maintain Semantic Coherence

the activity is decomposed into two subactivities, *Analyze DES Compatibility* and *Integrate DES*, as shown in Figure 5.

DES compatibility analysis ensures that the *Validated DES* produced from the earlier activity is compatible with any existing DES. A DES versioning scheme should be documented in the NDR to help ensure compatibility. In some circumstances, compatibility is broken in order to achieve semantic coherence. In such cases, clear versioning indicates potential incompatibility and additional intelligence can be built into the corresponding interfaces to handle this. The output from the compatibility analysis is either a *Compatible DES* when there are no compatibility issues or a *Change Request* to fit the new DES into the broader model. As described, compatibility may be left broken. The decision, which is a business as well as technical decision, depends on the long-term impact on interoperability, migration strategy, development stage, and cycle time.

Two approaches to compatibility analysis have been suggested in the diagram. The first approach is an empirical one that uses a *DES Instance Validation Tool* and a *Library of DES Instance Data*. The existing DES instance data is validated against the adjusted/extended DES. If the existing instance data is not invalidated and it fully covers the applications, it is likely that there are no compatibility issues. The other approach is to perform a subsumption test. If the adjusted/extended DES subsumes the previous version of the DES in the *Library of Semantically Coherent DES*, then there is no compatibility issue - the new version is backward compatible with the previous version. Note that 'A' subsumes 'B', if all possible instances of 'B' are also instances of 'A'. An example is when a structure 'A' is simply a less restrictive version of 'B'.

The compatible DES is fed into the *Integrate DES* activity to analyze and maintain the semantic coherence with the existing DES. The activity seeks to ensure that the DES does not create semantically duplicate terms or data structures and that any semantically overlapping terms or data structures are properly related. Typically, the activity would first identify terms and data structures that are semantic duplicates and/or overlaps. Where possible, duplicates should be eliminated by sending a *Change Request* to the activity A1 to reuse. When elimination is not possible, such as when the DES is already in use or when it is a standard controlled by an outside party, *Link Annotations* are created across the terms or structures. Similarly, a preferred approach to resolving overlaps would be to restructure and establish a relationship using a schema construct available in the DES. When that is not possible, cross-links between the overlaps should be annotated to ensure that the relationships can be identified and managed. Consequently, if there is no change request to the earlier activity, the compatible DES and table of terms are output from the activity along with the link annotations where necessary.

Analyzing semantic duplicates and overlaps can be a complex and tedious task particularly when there is semantic ambiguity in the model. A manual approach would require the domain expert to comb through the whole library of DES for each term and structure in the new DES. We envision semantic analysis tools to assist with this task. The semantic similarity measure assists in identifying semantic duplication and overlaps by providing quantitative guidelines for assessing the semantic proximity of terms and structures. Semantic alignment tools would (1) discover the relationships between the new terms or structures and the existing ones, and (2) suggest changes to accommodate the new relationships. Much research is on-going in semantic similarity measures [1, 3, 5, 17, 18, 19] and semantic alignment [2, 20]. These tools use information such as *External Ontologies* and *DES Documentation* to get more clues when comparing the target DES with the *Library of Semantically Coherence DES*. Since DES integration is a topic of ongoing research, the list of tools and reference materials here is by no means exhaustive.

Pilot DES

To solve a real integration problem, we must exchange information between specific software applications, which may impose additional requirements on the discovered DES. While the discovered DES presumably covers most of the DER, certain additions or modifications may be necessary. For instance, additional usage criteria specific to the applications being integrated may be needed, or adjustments for the deployment environment or community where the applications will be integrated may be needed. The DES piloting activity deals with these issues. Figure 6 illustrates the four subactivities of piloting.

The first activity is *Enhance DES Comprehensibility*. By themselves XML schemas can be difficult for the systems integrator to understand. Presenting the information in a variety of formats, such as graphical, tabular, and HTML is often quite useful for comprehension.

The process of integrating a DES into the broader semantic model can leave the DES too generic to fully support a given exchange. The *Augment DES* activity captures and codifies transaction-specific requirements on the DES. For example, the concept of a person in one application domain - like customer relationship management - may require more data elements than a person concept in a much simpler domain such as an address book. Consequently, when using the person data structure to exchange the address book, the augment DES activity codifies only the small number of elements that are used in the exchange transaction. These transaction-specific rules may be based on *Business Rules* and the overall *Integration Requirement Document* where DES use cases are documented. The *Validated DES Instance Data* from A2 verifies that the output of the augmentation is not over or under specified.

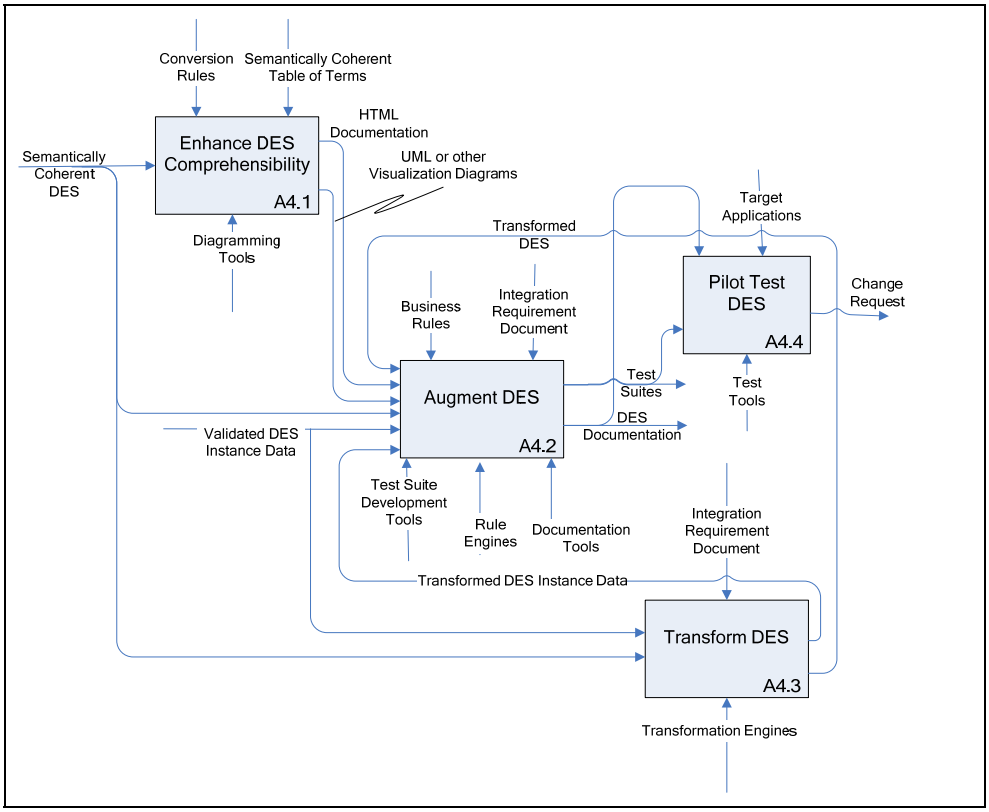

Figure 6: Activity A4 – Pilot DES

To support the specification of these transaction-based rules, NIST developed a tool called the Content Checker. This tool allows a user to write, store, and execute rules against which instance data is validated. This tool is especially useful in the scenario where standardized data exchange specifications are used but the transaction restricts the data in ways that are not specified. The restrictions can be codified using the tool and the data can be validated against those restrictions. For example, suppose an exchange schema is used for Purchase Orders (PO) which allows the user to specify a point of origin for the PO. In a particular transaction, one may wish to restrict the point of origin to a particular location. Other locations would be valid with respect to the schema but would be invalid for the transaction in question. This restriction could be codified and made available as a test of data participating in this type of transaction.

The outputs from the A4.1 activity are modeled as inputs to the A4.2 activity because we assume that they will be aggregated into the *DES Implementation Guide* along with the DES itself. The purpose of the *Implementation Guide* is to provide a single point of reference for systems integrators. The *Test Suites* output is an aggregation of the transaction-specific requirements, test scenarios derived from the integration requirements document, and DES instance data to be used as test data. The test suite has two main uses: the pilot test DES activity and conformance and interoperability testing of applications using the same transaction context. The information contained in the DES documentation and test suites is mostly the same. However, the former is tailored to human comprehension during the interface development while the latter is computer interpretable for run-time testing.

Often the *Transform DES* activity may be necessary in the deployment environments. This activity may include DES simplification, terminology transformation, or different DES representation forms. For example, we may need to simplify the XML schema by flattening its namespace to make it work with specific integration software or middleware tools. Or, we might need to use domain-specific terminology to maintain the DES semantic coherence. Finally, different DES representations may be required if the run-time data exchange is in EDI (Electronic Data Interchange) [22] but the broader semantic library uses XML syntax. These specific requirements should be documented in the *Integration Requirement Document*. The DES output from this activity is called the *Transformed DES*. The *Validated DES Instance Data* is also an input to this activity, because the instance data similarly should be transformed (into the *Transformed DES Instance Data*) and used in the DES documentation and test suites as well.

The other important subactivity of Pilot DES is to actually *Pilot Test DES* with *Target Applications*. In this activity, application developers follow the *DES Implementation Guide* to implement the data exchange interfaces. They then perform integration testing using the data in the test suites. Among the issues that may be discovered are (1) the DER was not documented correctly and (2) DES documentation is unclear or ambiguous. A *Change Request* document that summarizes the findings from the test is generated and fed back to earlier activities.

Register DES

The *Register DES* activity organizes the DES and related materials within a registry and stores them in a repository that

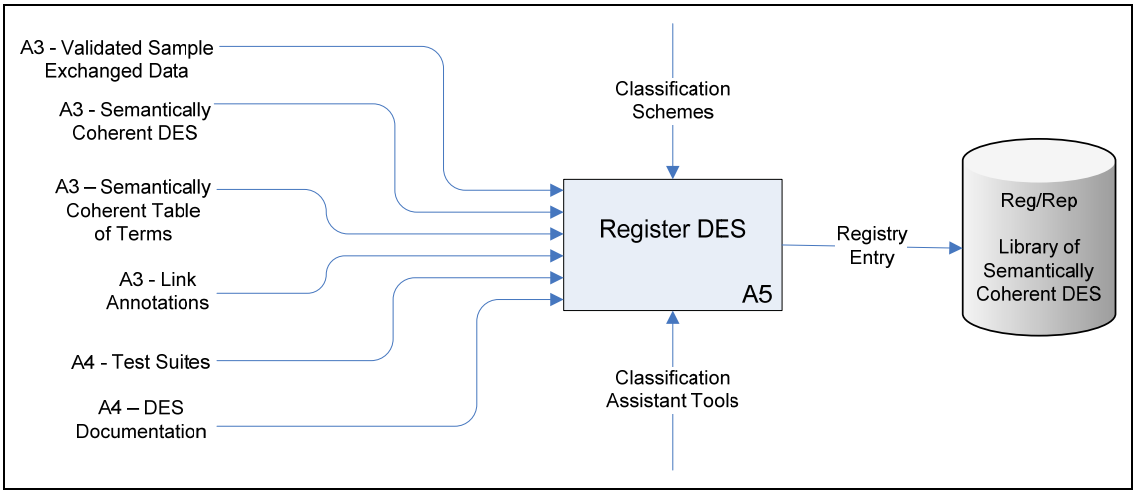

Figure 7: Activity A5 – Register DES

is accessible to other activities and users (see Figure 7). The inputs to this activity are those materials that are stored and maintained in the repository. Other supplemental information such as version, dependencies, associative semantics, and context information may be stored as well [24]. *Classification Schemes*, which are taxonomies, are typically used to categorize registered information. The taxonomies are typically domain specific. A piece of related information may be classified according to multiple schemes. This supports a multi-dimensional and structured search of the registry to make the discovery of DES more efficient.

Placing a DES and associated information into one or more classifications can be a tedious and error-prone task. Placing them in a wrong node in a classification not only makes them less accessible but poses the risk of misinterpretation by other users. In addition, placing a schema in a node that is too generic makes the DES discovery activity (A1) less efficient by inundating the user with too many options. Correct placement involves extensive understanding of the semantics of the classification scheme as well as the DES.

An envisioned tool to support the DES registration activity is the *Classification Assistant Tool*. This tool would use the semantic similarity measure described previously to suggest classification nodes to the user by matching the DES and associated information to a detailed definition in the classification scheme. This would narrow down the choices of classification nodes. The tool would support the user's decision-making process, which currently is based solely on node labels.

3 SUMMARY

We have developed many tools that support an XML Schema-based implementation of the activity model described here with particular emphasis on the validation activity [14]. Research to develop the semantic tools described above - such as the semantic lookup assistant and alignment tools - is beginning. This future work will allow the registry to be an active component in the enterprise data architecture rather than a static file store. These tools are summarized in Table 1.

Table 1 lists the tools along with their stage of development. The available tools can be found at http://www.mel.nist.gov/msid/XML_testbed. The stages of development, in order of increasing maturity, are envisioned, research, prototype, and released. Tools in the envisioned stage are those that we have identified here but have not initiated a project to address; those in the research stage are conceptualizations, include some understanding of a basic design, and are being researched further. A prototype tool has been demonstrated but further development at NIST has not been pursued. A released tool is one that NIST either has in production or would be able to make source code available.

Tool	**Stage**
XML Schema tools	Non-NIST
XML Validation page	Released
Schematron engines	Non-NIST
Schematron Editor	Prototype
Naming Assister	Released
Content Checker	Released
Quality of Design	Released
Model transformation tool	Prototype
Classification assistant	Envisioned
Semantic lookup assistant	Envisioned
Semantic similarity measure	Research
Semantic alignment algorithm	Research

Table 1: Tools supporting the XML Schema DES Life Cycle

This list is evidence of the claim that many of the tasks involved in the use of a DES for systems integration can be automated and more will be automatable in the future as technology matures. We are confident that if an enterprise data architecture is designed according to the proposed life-cycle model, long-term interoperability cost will be contained or reduced while integration activities grow.

PRODUCT DISCLAIMER

Certain commercial software products or services are identified in this paper. This use does not imply approval or endorsement by the National Institute of Standards and Technology, nor does it imply that these products or services are necessarily the best available for the purpose.

REFERENCES

1. Alspaugh, T.A.; Anton, A.I.; Barnes, T.; and Mott, B.W. An integrated scenario management strategy. In *Proceedings of the 4th IEEE Symposium on Requirements Engineering (RE99)*, Limerick, 1999, pp. 142-149.

2. Ambite, J.L. and Knoblock, C.A. Reconciling distributed information sources. In *Working Notes of the AAAI Spring Symposium on Information Gathering in Distributed Heterogeneous Environments*, Palo Alto, 1995.

3. Bhavsar, V.C.; Boley, H.; and Yang, L. A weighted-tree similarity algorithm for multi-agent systems in e-business environments. *Computational Intelligence Journal*, 20, 4 (2004).

4. Department of the Navy XML Naming and Design Rules. Final Version 2.0. Office of the DON, Chief Information Officer. January 2005.

5. Dong, X.; Halevy, A.; Madhavan, J.; Nemes, E.; and Zhang, J. Similarity search for web services. In *Proceedings of the 30th VLDB Conference*, Toronto, 2004, 827-383.

6. Evans, M.W.; Piazza, P.; and Dolkas, J.P. *Principles of Productive Software Management*. New York: John Wiley & Sons, 1983.

7. Gilb, T. *Principles of Software Engineering Management*. Reading: Addison Wesley, 1988.

8. Goyal, P. *An XML Schema Naming Assister for Elements and Types*, NISTIR 7143, October 2004.

9. International Revenue Service, *XML Standards and Guidelines (v1.0)*, September 2006.

10. Jelliffe, R. *The Schematron Assertion Language 1.5*.

11. Kulvatunyou, B.S.; Morris, K.C.; Buhwan, J.; and Goyal, P. Development life cycle and tools for XML content models. *XML Conference*, Washington DC, 2004.

12. Lubell, J., Kulvatunyou, B.S., Morris, K.C., Harvey, B. Implementing XML Schema Naming and Design Rules, *Proceedings of the Extreme Markup Languages 2006 Conference*, June 2006.

13. Morris, K.C.; Kulvatunyou, B.S.; Frechette, S; Lubell, J.; Goyal, P. *XML Schema Validation Process for CORE.GOV*, NISTIR 7187, 2004.

14. National Institute of Standards and Technology, Manufacturing Systems Integration Division. *XML Testbed Web Site*. http://www.nist.gov/msid/XML_testbed/.

15. National Institute of Standards and Technology. *Integration Definition for Function Modeling (IDEF0), Draft Federal Information Processing Standards Publication 183*. NIST 1993.

16. Open Application Group. *Open Application Group 9.0 Naming and Design Rules*. http://www.openapplications.org/downloads.

17. Peng, Y.; Zou, Y.; Luan, X.; Ivezic, N.; Gruninger, M.; and Jones, A. Towards semantic-based integration for e-business. In *International Symposium on Manufacturing and Applications*, Orlando, 2002.

18. Ryu, T.W., and Eick, C.F. Similarity measures for multi-valued attributes for database clustering. *In Proceedings of Smart Engineering System Design Neural Network, Fuzzy Logic, Evolutionary Programming, Data Mining and Rough Sets (ANNIE'98)*, St. Louis, 1998.

19. Schallehn, E., and Sattler, K.U. Using similarity-based operations for resolving data-level conflicts. In *Proceedings of BNCOD'o3*, Coventry, 2003, pp. 172-189.

20. Stuckenschmidt, H. and Visser, U. Semantic translation based on approximate re-classification. In *Proceedings of the Workshop Semantic Approximation, Granularity and Vagueness KR'00*, Breckenridge, 2000.

21. UN/CEFACT Applied Technology Group. *XML Naming and Design Rules Draft 1.1a*. UN/CEFACT, 2005.

22. UNECE United Nation. *UN/EDIFACT* Electronic Data Interchange Standard.

23. World Wide Web Consortium. *XML Schema 1.0*.

24. Xu, Z.; Karlsson, M.; Tang, C.; and Karamanolis, C. Towards a semantic-aware file store. In *Proceedings of HotOS IX: The 9th Workshop on Hot Topics in Operating Systems*, Lihue, 2003, 145-150.