Validating STEP Application Models at the National PDES Testbed

KC Morris, Mary J. Mitchell, Allison Barnard

kmorris@nist.gov, mmitchell@nist.gov, abarnard@nist.gov

National Institute of Standards and Technology, Building 220, Room A127, Gaithersburg, MD, 20899

Abstract

The problem of sharing data has many facets. The need to share data across multiple enterprises, different hardware platforms, different data storage paradigms and systems, and a variety of network architectures is growing. The emerging Standard for the Exchange of Product Model Data (STEP), a project of the International Organization for Standardization (ISO), addresses this need by providing information models which clearly and unambiguously describe data. These models are organized into a*pplication protocols.* An application protocol addresses the data sharing needs for a particular application area. STEP integrates the information requirements from all the application protocols. The validity of these information models is essential for success in sharing data in a highly automated environment.

This paper describes how information models will be validated in the National PDES Testbed at the National Institute of Standards and Technology. (PDES, Product Data Exchange using STEP, is the U.S. effort in support of the international standard.) Developing and testing information models for STEP is a complex process which involves synthesizing, analyzing, and manipulating large amounts of diverse information. Most of the process relies exclusively on human capabilities for analysis, judgment, and interaction; however, part of this process can and should be automated. A strategy for automation is based on an analysis of the flow of information in the model development and testing process and initial experiences with automation for validation testing at the National PDES Testbed.

Keywords: application protocols; conceptual schemas; information modeling; data sharing; Standard for the Exchange of Product Model Data (STEP); Product Data Exchange using STEP (PDES); validation testing; EXPRESS; STEP exchange file format; Initial Graphics Exchange Specification (IGES).

Table of Contents

Abstract ..... *i*

Table of Contents iii

List of Figures, Tables, and Examples v

1 Introduction 1

2 Application Protocols within STEP 2

2.1 Definition of an Application Protocol 3

2.2 Components of an Application Protocol 3

2.3 Application Protocol 202: Associative Draughting 4

3 Overview of Validation Testing 4

4 Activities of the Validation Testing Methodology 6

4.1 Scoping the Application Context 7

4.2 Model Construction 9

4.3 Test Definition 13

4.4 Test Case Data Generation 15

4.5 Test Execution and Analysis 16

4.6 Model Refinement 16

4.7 Summary 17

5 Automation for the Validation Testing Methodology 18

6 Validation Testing at the National PDES Testbed 24

6.1 Experiences with the Interim System 24

6.2 Directions for the Validation Software 26

7 Summary 27

8 References 29

List of Figures, Tables, and Examples

EXAMPLE 1	Representative Data for AP 202	4
TABLE 1	Activities and Toolkits of the Validation Testing System	5
FIGURE 1	Process Model of the Validation Testing Activities	8
EXAMPLE 2	High-Level Scope Statement for AP 202	9
EXAMPLE 3	Detailed Requirements of the AP 202 Application Object Drawing	10
EXAMPLE 4	Drawing_structure_and_administration Unit of Functionality for AP 202	10
EXAMPLE 5	Partial Application Reference Model for AP 202	11
EXAMPLE 6	Partial EXPRESS-G Application Interpreted Model for AP 202	12
EXAMPLE 7	Partial EXPRESS Application Interpreted Model for AP 202	13
EXAMPLE 8	Product Profile for AP 202	14
EXAMPLE 9	Test Purposes for AP 202 AIM	15
EXAMPLE 10	Example Issue Raised by Validation of AP 202	17
TABLE 2	VTS Tools	19
TABLE 3	Composition of the VTS Software Toolkits	20
TABLE 4	Information Flow Between Toolkits	21
FIGURE 2	Data Flow Between Toolkits	22

1 Introduction

Confidence in a standard by its user community is absolutely essential for a standard to gain acceptance. Proof that the standard is properly defined and that it can be used reliably will help achieve user confidence. A rigorous testing program is the foundation for any useful standard. Appropriate testing before standardization can ensure that a draft specification indeed meets the functional requirements for the standard.

The Standard for the Exchange of Product Model Data, commonly referred to as STEP[1], is an emerging international standard designed to provide a complete, unambiguous, computer-readable definition of the physical and functional characteristics of a product throughout its life cycle. The specification of these characteristics is in the form of a conceptual information model. While this specification is intended to be independent of any implementation language or technology, distinct implementation interface techniques are also specified in the standard. For instance, a file exchange format [ISO21] is specified for data exchange. The format is derived from an information model [ISO11] formally specified in EXPRESS. Other interfaces in development [ISO22-CD] will allow applications to share data using database technology.

An *Application Protocol* (AP) is a specification for a portion of the data described in STEP. This specification describes the information structures and their relationships in the context of a particular type of application [Palmer93]. The AP also specifies constraints on how the information is to be used based the product data requirements for the application area. The STEP AP specifications are intended to permit product information to be unambiguously exchanged or shared between implementations on dissimilar systems.

In its entirety, the conceptual information model in STEP is a logically integrated union of all the STEP APs. Each AP consists of two types of information models: the Application Reference Model (ARM) and the Application Integrated Model (AIM). The ARM specifies the information requirements of the AP in terms familiar to practitioners and experts in the application area. The AIM also specifies the requirements of the application but is integrated with other APs in STEP. In addition to the APs, STEP includes more general information models called the *Integrated Resources* [ISO1,ISO41], which act as libraries for the AIM specifications and are general enough to support diverse applications.

The National PDES Testbed[2] is used to test the validity of the information models proposed as standards within STEP. In this paper the term *application model* refers to the component information models of an AP or a similar information model.[3] The term is used since the ARM and AIM have many common characteristics. In 1991, Mitchell

1. The Standard for the Exchange of Product Model Data (STEP) is a project of the International Organization for Standardization (ISO) Technical Committee on Industrial Automation Systems (TC184) Subcommittee on Industrial Data and Global Manufacturing Programming Languages (SC4). For an overview of the standard refer to *Part 1: Overview and Fundamental Principles* [ISO1].

2. The National PDES Testbed is located at the National Institute of Standards and Technology. Funding for the Testbed Project has been provided by the Department of Defense's Computer-Aided Acquisition and Logistic Support (CALS) Office. The work described in this paper is funded by the United States Government and is not subject to copyright.

[Mitch91][Mitch92] proposed a methodology for validating STEP APs. The Validation Testing System (VTS) at the National PDES Testbed provides automation to support that methodology.

This paper discusses strategies for using software to support the proposed methodology for application model validation. The methodology and software build on previous experience with testing of application models for STEP. Section 2 provides background information on STEP and application protocols. Section 3 of the paper gives an overview of the complete validation testing process for those readers who are unfamiliar with AP validation[4]. Section 4 describes the specific activities which comprise the validation testing methodology. Section 5 discusses automation to support validation testing. Section 6 forecasts future directions for the Validation Testing System software based on experiences with the interim software being used at the National PDES Testbed.

2 Application Protocols within STEP

The Standard for the Exchange of Product Model Data is being developed in order to provide a comprehensive and reliable data exchange mechanism for industry. STEP will support the unambiguous definition of product definition data throughout the product's life cycle. It will supply a neutral representation for integrating computer-aided software systems allowing them to represent and transfer product data for the full life cycle of a product.

STEP consists of a set of information models that describe product definition data which could be used by multiple industries and application systems. The development of STEP uses information modeling methods to define the types of data within STEP and the rules which govern the use of the data. The standard is organized such that the application requirements are defined in separate parts of the ISO standard called *Application Protocols* (APs). Other parts of the standard, referred to as the *Integrated Resources*, are used as a basis for the Application Interpreted Model in the AP. The Integrated Resources contain information that is used across application areas. For example, the information model containing geometry [ISO42] is used by both AP 202 (Associative Draughting) and AP 203 (Configuration Controlled Design) [ISO203].

STEP also includes the definition of mechanisms needed to share information using the standard. These mechanisms include the specification language called EXPRESS [ISO11] and an exchange file format [ISO21]. The exchange file format abides by the definitions in the information models specified in EXPRESS.

3. The term *application model* will be used throughout this paper to refer to the domain specific information model which is under evaluation. The information model can be the component models of an AP, e.g., an Application Reference Model (ARM) and an Application Interpreted Model, (AIM) or a precursor to an AP such as a Context Driven Integrated Model (CDIM). While the validation methodology is applicable to this general class of application models, the first priority at the National PDES Testbed is to support the requirements for validation of AIMs. The software toolkit developed by the NPT will support any information model specified in EXPRESS.

4. The National PDES Testbed Report Series [Mitch90, Mitch91, Morris91a, Morris91b, Morris92] expands on different aspects of the system described in this paper.

2.1 Definition of an Application Protocol

APs are an important concept in the development of STEP since it is unlikely that there will be a single practical implementation of the standard in its entirety due to the size of its scope. APs are designed to permit practical implementations of STEP. An AP is a specification for applying STEP in a standard way to meet the specific needs for a particular application domain (such as two-dimensional drafting.) The AP is the part of the standard that will directly affect industry. Industrial users of the standard will either buy systems that implement an AP or build application systems that meet the AP specification. Vendors of computer-aided systems for design, analysis, manufacturing and support will be asked by purchasers of such systems to comply with specific APs. Validation of the APs is needed to ensure that the specifications they contain are correct and will work in a practical sense.

In order for STEP to support a wide variety of applications and industries, each AP selects the elements from STEP that provide the semantics to support the information requirements of the set of applications within the scope of the AP. The use of these elements is then restricted where necessary to enforce the rules of the discipline. For example, in general a cartesian point contains three coordinate attributes. However, for two-dimensional drafting applications to meaningfully share information, both applications should not use the third coordinate. The AP for two-dimensional applications would specify that to exchange data between applications the third coordinate of a cartesian point must not have data associated with it.

2.2 Components of an Application Protocol

The specification of an AP includes a number of components. Two of the components are computer processible information models; the remainder of the AP is descriptive. The specification of an AP will include the following:

- Scope: A clear definition of what type of application is covered by the AP.
- Application Activity Model: A formal graphical description of the activities contained in the application area and their relationships. The activity model formalizes the definition of the scope for the AP.
- Application Reference Model (ARM): A definition of the information requirements in terms familiar to an expert in the discipline.
- Application Interpreted Model (AIM): A specification of the information requirements in terms of the information models within STEP Integrated Resources. This specification functionally matches the requirements described in the ARM, but uses different terminology and organization.
- Conformance Clause: Statements which define explicitly how complete an implementation must be to conform to the standard. The statement defines any options for an implementation.

In addition, other material that supports the AP is documented separately. The supporting material includes the following:

- Validation Report: A report describing the methods used to validate the components of an AP, the results of the validation tests, and an analysis of those results.
- Abstract Test Suites: A standardized set of abstract test cases necessary for testing an implementation for conformance to the standard defined by the AP.

2.3 Application Protocol 202: Associative Draughting

To more clearly present the concepts described in this paper, examples taken from an Application Protocol being proposed within STEP have been included in this text. Application Protocol 202: Associative Draughting [ISO202] is intended for the exchange of computer-interpretable drawing information and the associated product definition data.[5] This AP satisfies the need for the exchange of drawings that maintain an association between the annotations on the illustration of a product and the geometric shape of the product. The representations for both the annotation and the shape are independently defined in a two-dimensional or three-dimensional coordinate system. This information is then mathematically transformed to two-dimensional sheet space and placed on the drawing.

FIGURE 1 depicts a drawing which would be described by this AP.[6] The AP specifies a standard format for the information needed to represent this drawing in a computer.

EXAMPLE 1 **Representative Data for AP 202**

The figure below illustrates a simple connector support bracket made from sheet stainless steel. This part is a component of an HP 83711 Signal Generator. Using AP 202 the information presented as annotation on the drawing as well as the geometric model and the transformation used to present the geometric model on the drawing could be exchanged between different commercial or proprietary applications for drafting.

See attached Figure.

3 Overview of Validation Testing

Application protocol development and testing is a complex process; it involves synthesizing, analyzing, and manipulating large amounts of diverse information. Most of the process relies exclusively on human capabilities for analysis, judgment, and interaction; however, part of this process can and should be automated. A strategy for automation is based on an analysis of the information flow needed for the development and testing process and initial experiences with automation for validation testing at the National PDES Testbed. This strategy is discussed throughout this paper. This section describes the validation process in general.

5. This application protocol is being developed in the U.S. by PDES, Inc. an industrial consortium formed to accelerate the development, validation and implementation of the STEP standard.

6. This drawing appears courtesy of the Hewlett Packard Corporation.

Validation testing of a developing application protocol determines whether the AP does what it is intended to do, i.e., meets the requirements that led to its development. The proposed approach is to validate APs by simulating the behavior of the application.[7] Since APs will be used for sharing data, the specific behavior that must be verified is the ability of the data structures to support the data access requirements for the application areas.

Validation tests are identified by examining the functions of the applications in a particular area. These functions are documented in the AP component called an Application Activity Model. The data required to perform each function in the application processes is specified in detail as a validation test. Realistic data is associated with each of the validation tests. The data needed to perform a specific activity is then mapped into the structures defined by the application model. The data and data structures are examined to determine if input requirements for each activity are satisfied. The application model (the ARM or AIM depending on the stage of AP development) with its associated data is examined to determine if it can support the generation of the outputs needed by the applications. This approach to validation essentially simulates the behavior of an application system interacting with a user of the system.

This methodology for validating an application model can be decomposed into the six high-level activities shown in TABLE 1: Scoping the Application Context, Model Construction, Test Definition, Test Case Data Generation, Test Execution and Analysis, and Model Refinement and Improvement [Mitch90]. Four logical software toolkits have been identified to support these activities. A toolkit consists of the set of software which automates or supports the particular activities for a phase of the AP development and testing process. Separate toolkits for each activity are not needed due to overlaps in the automation needs for the activities. TABLE 1 identifies how the software toolkits correspond to the activities.

TABLE 1 **Activities and Toolkits of the Validation Testing System**

Activity	Toolkit
Scoping the Application Context	Model Scoping and Construction
Model Construction	Model Scoping and Construction
Test Definition	Test Definition
Test Case Data Generation	Test Case Data Generation
Test Execution and Analysis	Test Execution and Analysis
Model Refinement and Improvement	Model Scoping and Construction

The validation process can be compared to the activities of a building code inspector. An AP developer or tester may use the toolkits to examine application models, just as a building inspector uses a blueprint to examine aspects of a new building at its site.

7. Alternative approaches to information model validation, such as prototype implementation, may yield equally convincing results. A simulation technique was chosen because sufficiently capable and flexible prototype environments were not available.

Constructing an application model is similar to an inspector's analysis of a new building to determine what should be tested. The AP tester defines tests to perform against the new application model, just as the inspector must formulate a plan for testing the building's features. The inspector identifies the appropriate building codes needed to test different features of the building for compliance. For example, the building inspector would test the chimney flue on a house with a fireplace. Likewise, product data for testing features of the application model is collected. Features are tested against known cases, just as a building inspector uses an instrument to check for correct voltages in electrical outlets. The VTS software is similar to the inspector's voltage meter — it allows the tests to be conducted in a consistent manner, but the meaning of the tests is dependent on human analysis of what the instrument reports.

Together the activities and the toolkits which support them comprise the Validation Testing System, just as inspecting a building requires the process the inspector uses, as well as his instruments. The activities and toolkits of the VTS are discussed in the following sections.

4 Activities of the Validation Testing Methodology

Application models are validated by simulating the data access patterns of the relevant applications. This process was originally decomposed into the six activities shown in TABLE 1. As the methodology for validation testing developed, the decomposition was refined to the seven steps shown in FIGURE 1. [Mitch91] The six original activities directly reflected needs for automation of information model development and validation testing. These are used as a basis for discussion in this paper.

The IDEF0 [ICAM82] activity model in FIGURE 1 depicts the seven steps in the development and validation testing methodologies and it shows the flow of information in the validation process. The first step corresponds to activities 1 and 2 below. In this step the application scope and model are constructed and released for testing. The scope controls what is in the application model and, therefore, guides what needs to be tested.[8] Once the application model has been constructed, it is typically turned over to a different set of people: the testers.

The testing activities, which fall into the grey area in FIGURE 1, verify the correctness of the model that has been developed during the previous activities. The next three steps in FIGURE 1 are discussed below as part of the Test Definition activity from Table 1. In the second step, *test planning*, the testers decide what will be tested. In the third and fourth steps, *create cross reference map* and *coverage analysis*, further refinement and additional detail are provided to the set of tests previously defined. In these two steps, the testers acquire a detailed understanding of the application model and assess whether the tests defined cover all aspects of the model. They then gather and select industry contributed product data. Separate activities were not defined for these steps because they did not generate new requirements for software tools to automate the process. These steps expand the testers' understanding of the problem and include interaction

8. Stark and Mitchell discuss these activities in detail in the paper "Development Plan: Application Protocol for Mechanical Parts Production" [Stark91].

with the potential contributors of product data. The remaining three steps in FIGURE 1 correspond one-to-one with the activity descriptions given below.

4.1 Scoping the Application Context

The activity of Scoping the Application Context identifies a formal technical boundary for the application model by examining the accepted practices and functions of the applications domain. The boundary is defined by analyzing the general processes the application performs. The manner in which the application model will be used to share data (e.g. exchange files or shared use) may constrain these processes. A decision is made on whether each process input or output will be supported by the AP. A study of the information needs of the application processes establishes categories for the information. The categorization clarifies what capabilities the AP should support. An activity model is developed to illustrate the scope of the application model. The activity model is reviewed by experts in the application area to ensure that it reflects common business practices. The results of Scoping the Application Context are the following:

- an activity model which defines the application processes and their interfaces, and
- a statement of the scope and requirements (EXAMPLE 2).

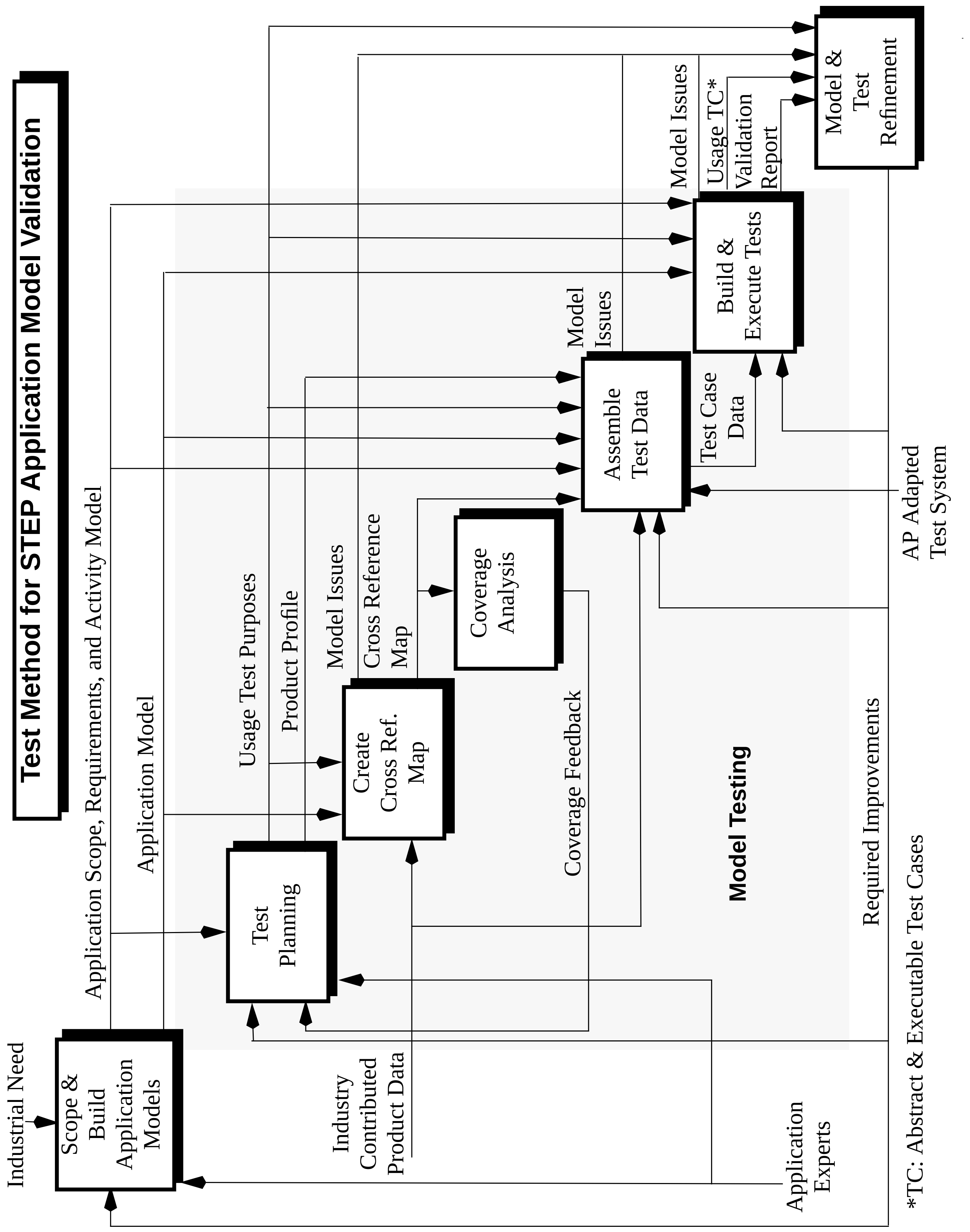


FIGURE 1 **Process Model of the Validation Testing Activities**

EXAMPLE 2 **High-Level Scope Statement for AP 202**

Below is a partial list of items that are in and out of scope for AP 202. The list contains the high-level scoping statements that guided the development of the Application Reference Model (ARM). The ARM is the information model which contains the detailed information requirements for the application area covered by the AP.

In Scope	Out of Scope
Mechanical product drawings	Representation of drawings not associated with a product
Drawing composition consisting of sheets, views of the geometric model depicting the product shape, and/or annotation	Representation of the shape of a product that is not associated with a drawing
Computer-interpretable 2D or 3D geometric model depicting the product shape and transformations used for the generation of the drawing views	Computer-interpretable bill of material structure except as conveyed by annotation on the drawing
Computer-interpretable associations between dimensions or draughting callouts and their respective target product shape geometry or annotation	Exchange of drawing history
Representation of a drawing that depicts any phase of design, approval or release Exchange of individual drawing revisions	Use of constructive solid geometry for the representation of the shape of the product
Several types of geometric models used to represent product shape	Non-planar 3D annotation
Presentation of non-shape product definition data depicted in a drawing by planar annotation	Strict enforcement of draughting standards
Administrative information used for the purpose of drawing management	
Administrative information identifying the product versions being documented in the drawing	
Mechanisms for the grouping of elements depicted on a drawing	

4.2 Model Construction

The Model Construction activity results in the creation of two separate, detailed application models, the ARM and the AIM. Interviews with experts in the application area provide detailed information requirements and an understanding of any constraints on the use of the information. These requirements are captured in the information model called the ARM. The elements in the ARM are called application objects. EXAMPLE 3 is a description of the ARM application object Drawing and its relationships to two other application objects.

The objects and relationships in an ARM are organized into Units of Functionality (UoF). The collection of objects in a UoF defines a distinct application concept. UoFs provide a summary of the functionality of an ARM. EXAMPLE 4 describes the Drawing_structure_and_administration UoF from the AP 202, while EXAMPLE 5 shows the graphical representation of the information model for the portion of the ARM containing the Product_relation and Drawing_structure_and_administration UoFs.

EXAMPLE 3

Detailed Requirements of the AP 202 Application Object Drawing

The data requirements from the Application Reference Model are defined as text within an AP. This example defines the requirements for the application object Drawing and the relationships from Drawing to Drawing_approval and Drawing_sheet.

Application Object Definition	Application Object Relationships
A **Drawing** is the presentation of product data in a human interpretable form wherein the physical and functional requirements for that product are presented pictorially and/or textually. The data associated with a Drawing are the following: Contract_reference, Drawing_number, Drawing_revision_id, Drawing_specification, Drawing_title, Drawing_type, Security_classification.	**Drawing** to **Drawing_approval** relationship: Each Drawing is governed by zero, one or many Drawing_approval_objects. Each Drawing_approval governs exactly one Drawing. **Drawing** to **Drawing_sheet** relationship: Each Drawing consists of one or more Drawing_sheet objects. Each Drawing_sheet belongs to exactly one Drawing.

EXAMPLE 4

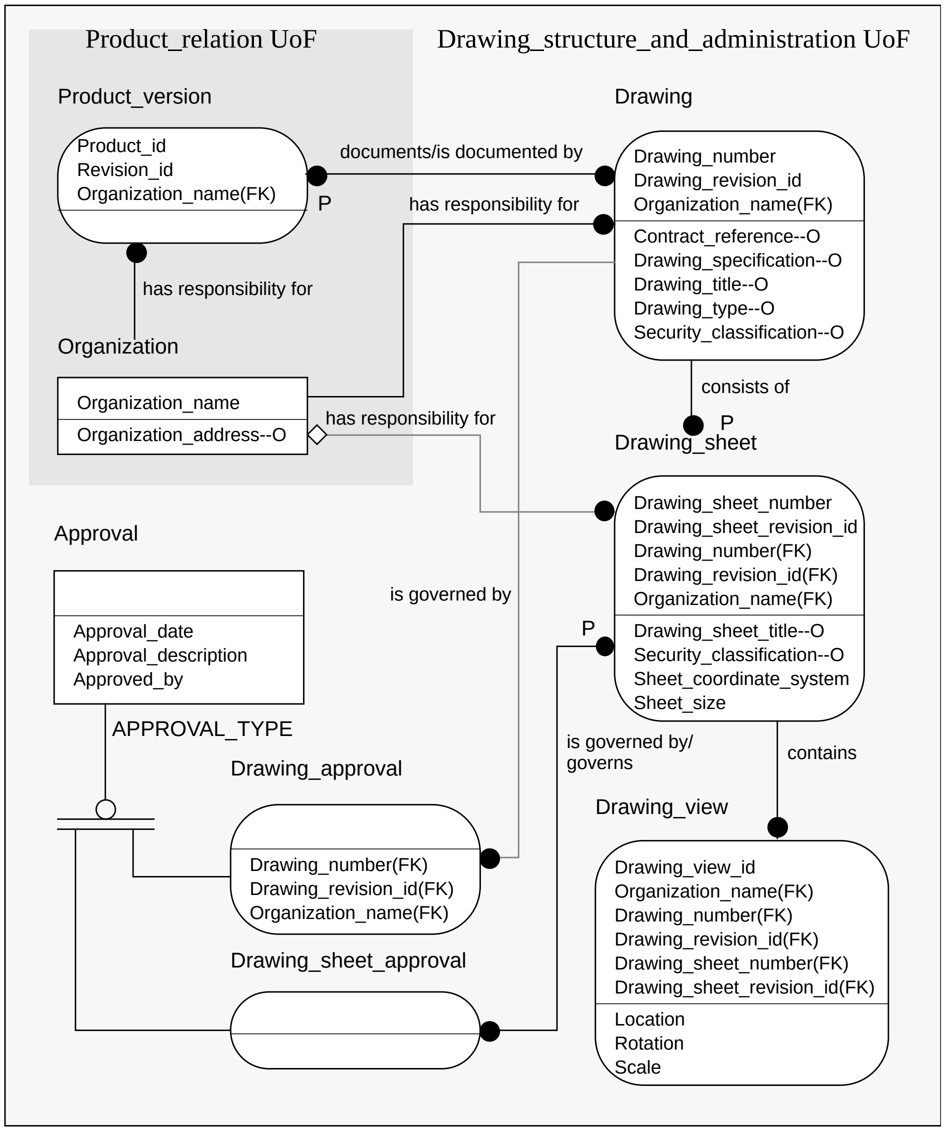
Drawing_structure_and_administration Unit of Functionality for AP 202

Units of Functionality are used to organize the data in the Application Reference Model. The objects in a UoF represent one distinct concept. This example describes one UoF for AP 202 and lists its application objects.

Description	Application Objects
The **Drawing_structure_and_administration** UoF provides information about the composition of drawings and drawing sheets and the required administrative information necessary to manage drawings. The application objects in the adjacent box are used by the **Drawing_structure_and_administration** UoF.	Approval Drawing Drawing_approval Drawing_sheet Drawing_sheet_approval Drawing_view

EXAMPLE 5 **Partial Application Reference Model for AP 202**

The diagram below illustrates a section of the Application Reference Model for AP 202. This part of the ARM contains two Units of Functionality, Product_relation and Drawing_structure_and_administration. The diagram shows the relationships between the two UoFs. The example is drawn using the IDEF1X technique for information modeling [ICA85,LOO86].

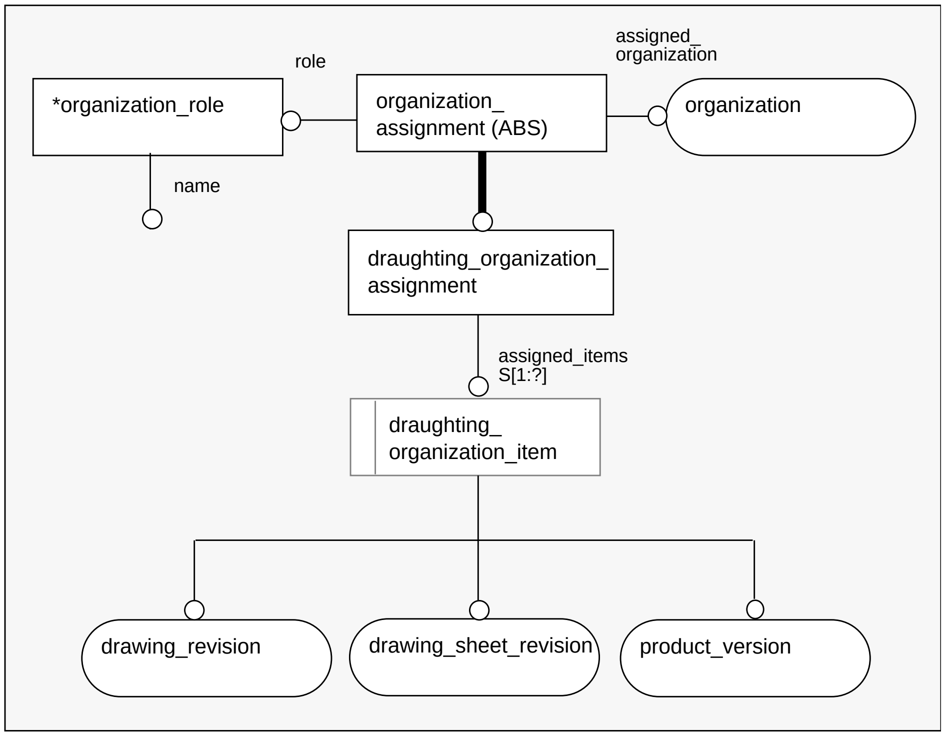


Next, segments of the Integrated Resources existing within STEP that support the semantics of the ARM are identified. Any application specific restrictions and preferences are also specified at this time. An Application Interpreted Model (AIM), one of the application models that is to be tested, is produced. The AIM supports the requirements specified in the ARM but is based on the information structures specified within STEP Integrated Resources. Within an AP the AIM is provided in three formats:

- a graphical representation, such as EXPRESS-G [ISO11] (EXAMPLE 6),
- a formal specification in EXPRESS [ISO11] (EXAMPLE 7), and
- a completely documented description including definitions and illustrations.

EXAMPLE 6 **Partial EXPRESS-G Application Interpreted Model for AP 202**

The model below, illustrated in EXPRESS-G, is a section of the Application Interpreted Model for AP 202. This section corresponds to the highlighted section of the ARM in EXAMPLE 5 that contains Organization. The model below shows how the ARM entity Organization and it's relationships to Drawing, Drawing_sheet, and Product_version are represented in the AIM.



The result of activities 1 and 2 is a complete description and understanding of the problem that the application model is addressing. The result of collecting together and analyzing the information requirements into a single information model, the ARM, is the technical basis for the AIM. The interpretation of this model with other information models within STEP produces an information model which is specific to an application area and also compatible with other views in a product life cycle.

These results of Step 1 and Step 2— the understanding of the domain and the integration with the existing information models — are represented in FIGURE 1 as the output of the first box, *Application Scope, Requirements, and Activity Models*. This result involves intensive analysis and judgment which could benefit from computer-aided support. The other output in FIGURE 1, *Application Models* (both the ARM and the AIM), is the computer-interpretable representation of that information model. These computer-interpretable results are the portion of the output which is used to automate the testing process.

EXAMPLE 7 **Partial EXPRESS Application Interpreted Model for AP 202**

The model below, documented in EXPRESS, is a section of the Application Interpreted Model for AP 202. This EXPRESS corresponds to the EXPRESS-G AIM entities shown in EXAMPLE 6.

```
TYPE draughting_organization_item = SELECT
    (drawing_revision, drawing_sheet_revision, product_version);
END_TYPE;

ENTITY organization_assignment
    ABSTRACT SUPERTYPE;
    assigned_organization: organization;
    role: organization_role;
END_ENTITY;

ENTITY draughting_organization_assignment
    SUBTYPE OF (organization_assignment);
    assigned_items: SET [1:?] OF draughting_organization_item;
END_ENTITY;
```

4.3 Test Definition

The primary result of the Test Definition activity is a plan for validating the application model. The information produced by this activity is informative and guides the rest of the testing process but is not computer-processible at this time. The test plan describes how typical application data access requests will be satisfied using representative data. The test plan provides the organization to manage the complexity of the required tests. The plan consists of tests based on the usage requirements and includes an understanding of the types of and sources for data needed to conduct the tests. These activities are depicted in three steps of FIGURE 1: *Test Planning*, *Create Cross Reference Map*, and *Coverage Analysis*.

During the Test Definition activity the testers define testing requirements, develop a test plan to be used to evaluate the functionality of the application model, and identify and gather test data. The testers consolidate the results of interviews with experts into realistic usage scenarios which focus on the exchange or shared use of data to describe the data access needs for the application area. The usage scenarios identify logical groupings of information. Each grouping that identifies a non-redundant need to access information is called a test purpose (see EXAMPLE 9). Test purposes may be organized into test groups.

In addition to the test plan, the Test Definition activity includes three outputs for analyzing the characteristics of the test data: a product profile, a cross reference map, and coverage feedback. A product profile describes characteristics of the information and is used for gathering representative test data. A portion of the product profile for AP 202 is provided in EXAMPLE 8. The cross reference map indicates the correspondence between the application model and the test data. The creation of the cross reference map may uncover major structural flaws in the application model i.e., test data that has no logical place in the application model. The coverage analysis of the representative test data reveals unused segments of the application model. If the data cannot be identified that corresponds to these segments, then the application model needs to change. Either the model was misunderstood and better documentation is needed, the original requirement is invalid or appropriate data needs to be found. Unused segments of the application model are ultimately removed.

EXAMPLE 8 **Product Profile for AP 202**

The example product profiles below describe characteristics of drawings that are within scope of AP 202 and would qualify as representative test data.

1. Drawings with associated computer-interpretable 2D product geometric model and transformations for generating views
   a. depicting mechanical products or possibly architectural, engineering and/or construction products
   b. with presentation of 2D non-shape information depicted through the use of annotation
   c. with administrative information for the purpose of drawing management
   d. in any phase of design or release

2. Drawings with associated computer-interpretable 2D product geometric model and transformations for generating views
   a. depicting mechanical products or possibly architectural, engineering and/or construction products
   b. with computer-interpretable associations between dimensions or draughting callouts
   c. with presentation of 2D non-shape information depicted through the use of annotation
   d. with administrative information for the purpose of drawing management
   e. in any phase of design or release

3. Drawings with associated computer-interpretable 3D product geometric model and transformations for generating views
   a. depicting mechanical products or possibly architectural, engineering and/or construction products
   b. with computer-interpretable associations between dimensions or draughting callouts
   c. with presentation of 2D or 3D non-shape information depicted through the use of annotation
   d. with administrative information for the purpose of drawing management
   e. in any phase of design or release

The Test Definition activity produces six outputs:

- a product profile (EXAMPLE 8),
- an overall test plan with test groups,
- test purposes with usage constraints (EXAMPLE 9),
- the identification of representative test data,
- a cross-reference map to correlate the test data with the application model, and
- a report which describes model issues and needed improvements to the application model.

EXAMPLE 9 **Test Purposes for AP 202 AIM**

The following test purposes correspond to the entities from the portion of AP 202's EXPRESS AIM that was provided in EXAMPLE 7.

Test Purposes for organization_assignment:

- **organization_assignment** as draughting_organization_assignment;
- **organization_assignment** with assigned_item as organization;
- **organization_assignment** with role as organization_role;

Test Purposes for draughting_organization_assignment:

- **draughting_organization_assignment** with assigned_item as drawing_revision
- **draughting_organization_assignment** with assigned_item as drawing_sheet_-revision
- **draughting_organization_assignment** with assigned_item as product_version

4.4 Test Case Data Generation

During the Test Case Data Generation activity, represented in FIGURE 1 as the *Assemble Test Data* step, the testers assemble and build the product data based on the product profile for a given usage scenario. The objective is to identify where in the application model the representative product data will reside. Each piece of industry-contributed data should have a single, logical place in the model.[9]

9. In testing AP203 [PDES90] less than half the data needed for simulating the application testing was available in electronic form. The rest was generated with simple text editors. This process was labor-intensive and error-prone.

The computer-processible output of this activity is the test case data. Another important output of this activity is the detailed description of the usage test purposes, which are fully documented as abstract test cases[10]. This activity results in the following output:

- detailed test data in a format suitable for processing by the VTS software, e.g. STEP exchange file format [ISO21],
- abstract test cases, and
- a report which describes model issues and needed improvements to the application model.

4.5 Test Execution and Analysis

The Test Execution and Analysis activity involves the execution and analysis of test cases against the application model. In order to execute the test cases, a computerized testing environment is established and the test cases are formally specified with respect to this testing environment. Analysis of the test cases involves comparing the test results to the expected results to determine the validity of the application model. In addition, general statements that describe criteria for conformance to the AP are developed and documented. This activity produces the following results:

- test reports,
- additional abstract test cases, if needed,
- improved test product data for re-testing and potential use in conformance testing test results,
- executable test cases for re-testing and potential use in conformance testing,
- test results, and
- a report which describes model issues and needed improvements to the application model.

4.6 Model Refinement

The Model Refinement activity resolves issues that were uncovered during the testing process. Alternative solutions are proposed and the best solution is selected. Once there is agreement on how to resolve an issue, the model is modified and a new model is released for validation testing. This activity results in the following information:

- the refined application model,
- an issue resolution statement describing the solution selected and supporting rationale (EXAMPLE 10), and
- refined test purposes and abstract test cases.

10. An abstract test case is the complete, implementation-independent specification of the actions required to achieve a specific test purpose. This includes the plan of what aspect of the model to test, the data to be used to test this aspect, and the expected results of the tests. These test cases may ultimately contribute to the abstract test cases needed for conformance testing. See *Part 31: Conformance Testing Methodology and Framework: General Concepts* [ISO31] for more information about conformance testing.

EXAMPLE 10 **Example Issue Raised by Validation of AP 202**

This is an example of an issue raised during the validation of the AP 202 ARM. The issue pertains to the Contract_reference attribute of the entity Drawing.

Issue Number:	74.WD.202	**Status:**	Resolved
AP 202 Version:	N105 + (v0.61)	**Date:**	920626
Author:	AP 202 Testing	**Area of Document:**	Clause 4, ARM
Issue Text:	A requirement exists for more than one contract number per drawing (therefore, 0, 1, and at least 2). For example, these can be the contract numbers of the original design and the current design activity, or prepared under and furnished under contract numbers. If this is allowed by the current model, a more explicit definition should be given explaining the use of the contract number attribute.		
Discussion:			
Solution:	The Contract_reference attribute of Drawing has been changed to be optional and an aggregate.		

4.7 Summary

Model validation testing is an iterative process. The end result of the process is an application model with a high probability of supporting the requirements that drove its development. The model must be both useful and usable to be part of the standard. The involvement of a variety of application experts in the validation process ensures that the model is useful. There should also be reviews by application experts independent of the AP project. To ensure that the model is usable, validation testing should be repeated until the model satisfactorily supports the information needs identified in the test plan for the AP. Once the utility of the application model has been demonstrated, the other components of the AP document can be developed, such as the specification of conformance requirements.[11]

The validation of an application model is dependent on the application area under consideration, but the validation process itself is constant and many aspects of it can be automated or aided by automation. Due to the complexity of the standard being developed, some automation of the process is mandatory. The standard will enable the automatic exchange and sharing of data. Therefore, the ability to automatically access data using the application model needs to be validated. The following section discusses how this is accomplished.

11. The form for specification of AP conformance requirements is evolving. The relationship between AP validation and conformance testing is still being defined within STEP.

5 Automation for the Validation Testing Methodology

This section describes the four software toolkits which support the validation testing methodology and the data flow between them. The software simulates the information access requirements for the application area being tested. The VTS software will provide a controlled environment for validation testing, thereby reducing the potential for introducing errors into the process. The VTS toolkits and the control of the supporting environment will also reduce the level of computer sophistication needed so that the testers will be able to concentrate on validating the application models.

TABLE 1 from the introduction identifies the four toolkits and their correspondence with the validation testing activities. The four toolkits are 1) *Model Scoping and Construction*, 2) *Test Definition*, 3) *Test Case Data Generation,* and 4) *Test Execution and Analysis*. All the toolkits support the validation testing process. The primary requirement of this process is the capability to manipulate and represent a particular application model in a variety of ways. Therefore, many functional requirements are common among the toolkits [Morris91]. TABLE 2 summarizes and briefly describes the software requirements of the validation process. Some of these tools — such as the CAx[12], database, and word processing systems — are available as commercial systems. For these tools commercial systems will be used in the VTS software. Other tools are either available from related projects or will be developed for the VTS.

12. CAx is any Computer-Aided operations/processes, including MCAD (Mechanical Computer-Aided Design) e.g. drawing/drafting, ECAD (Electrical Computer-Aided Design), e.g. PCB layout, MCAE (Mechanical Computer-Aided Engineering), e.g. solids modeling, ECAE (Electrical Computer-Aided Engineering), e.g. logic design, CAM and CIM (Computer-Aided Manufacturing and Computer-Integrated Manufacturing), e.g. NC processing.

TABLE 2 **VTS Tools**

Tool	Description
CAx Systems	Provides capability for producing product data.
Configuration System	Supports the management of documents and other files, including programs, used by the VTS system.
Cross Referencing System	Supports management of the relationships between specific tests and relevant sections of the application model.
Data Converter	Translates a data file corresponding to a particular application schema to a revised format based on changes made to the schema.
Database System	Provides for persistent storage of and shared access to data.
Diagramming Tool	Supports display and creation of diagrams which illustrate the concepts represented in an application model.
EXPRESS Browser	Displays the contents of an information model written in EXPRESS in a “user friendly” format (e.g. EXPRESS-G.) Provides minimal hypertext-like capabilities.
EXPRESS Constructor	Assists in the construction of EXPRESS descriptions of an application model. May be graphical, such as EXPRESS-G or context sensitive textual editing for EXPRESS.
EXPRESS Parser	Parses an application model specified in EXPRESS to verify syntactic correctness.
EXPRESS Translator	Translates an application model written in EXPRESS into a program work space.
IGES to STEP Translator	Reads an IGES file and outputs a corresponding STEP file.
Logging Mechanism	Records results of a session during which the tests on the data were conducted. Provides extensive error reporting.
Other Model Browsers	Displays the application model in a variety of different formats for reference. Does not provide the capability to modify model. Could be a simple drawing package.
Other Translators	Translate data from a CAx system’s internal format to STEP format.
STEP Data Editor	Provides an interactive environment for the display, manipulation, and editing of data that corresponds to an application model. Reads and writes data in STEP exchange file format.
STEP Data Browser	Provides an interactive environment for the display and manipulation of data that corresponds to the application model. Does not allow the user to change the data.
STEP Exchange File Parser	Parses a STEP file into a working format and/or database system.
Query Language	Provides capability to represent data access requests in a format that can be executed during testing. Dependent on the database system.
Word Processing System	Supports editing functions and provides context sensitivity for standard formats.

The decomposition of tools depicted in TABLE 3 represents the automation needed to support the activities covered by the individual toolkits. Several of these tools are shared by all four toolkits.

TABLE 3 Composition of the VTS Software Toolkits

Model Scoping and Construction	Test Definition
Diagramming Tool	Diagramming Tool
EXPRESS Browser	EXPRESS Browser
Word Processing System	Word Processing System
Other Model Browsers	Other Model Browsers
EXPRESS Parser	
EXPRESS Constructor	
Test Case Data Generation	**Test Execution and Analysis**
EXPRESS Browser	EXPRESS Browser
Word Processing System	Word Processing System
EXPRESS Translator	EXPRESS Translator
STEP Data Editor	STEP Data Browser
STEP Exchange File Parser	STEP Exchange File Parser
Logging Mechanism	Logging Mechanism
IGES Translator	Database System
Other Translators	Query Language
Data Converter	Cross Referencing System
CAx systems	

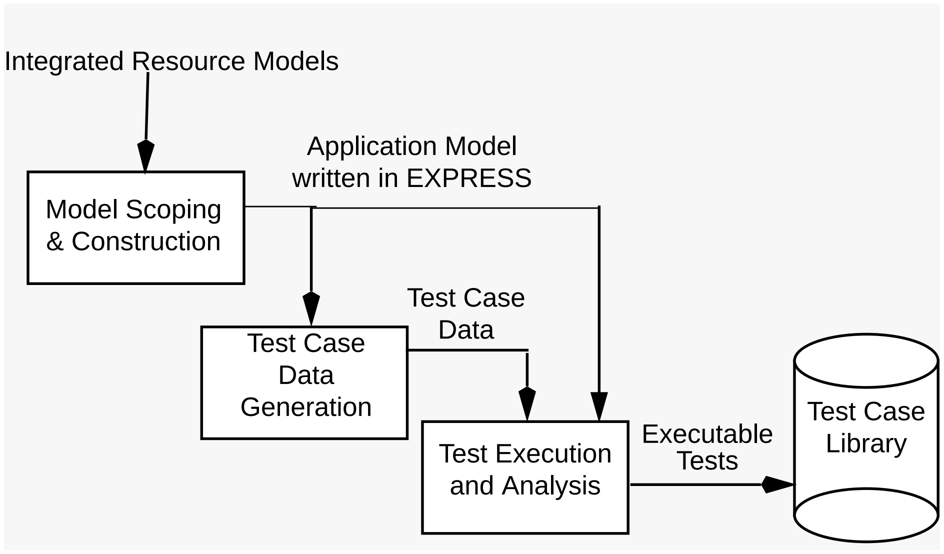
Each of the activities of the VTS consumes and produces data. The data produced by one activity is required in the subsequent activities. A subset of the information produced is directly processible and is used to automate the activities. This automation parallels the flow of information between steps illustrated earlier in FIGURE 1. TABLE 4 illustrates information inputs and outputs of the four toolkits; the entries in **bold** represent the computer-processible portion of the information which flows between the toolkits. The remainder of this section focuses on that information.

TABLE 4 Information Flow Between Toolkits

	INPUT	OUTPUT
Model Scoping and Construction	Application Requirements from Experts	Scope & Requirements Statement
	STEP Integrated Resource Models	Activity Model
		Application Models Written in EXPRESS
		Graphical Representations (i.e. EXPRESS-G)
Test Definition	Application Requirements from Experts	Test Plan with Test Purposes
	Scope & Requirements Statement	Product Profile
	Activity Model	Cross-Reference Map
	Application Model (for reference)	Sources for Product Data
		Model Issues
Test Case Data Generation	Scope & Requirements Statement	**Test Case Data**
	Activity Model	Abstract Test Cases
	Application Models Written in EXPRESS	Model Issues
	Test Plan with Test Purposes	
	Product Profile	
	Contributed Product Data (e.g., **IGES files**)	
Test Execution and Analysis	Scope & Requirements Statement	**Executable Test Cases**
	Activity Model	Validation Reports
	Application Models written in EXPRESS	Testing Software Enhancements
	Test Plan with Test Purposes	Refined Abstract Test Cases
	Abstract Test Cases	Model Issues
	Test Case Data	

As shown earlier in TABLE 1, the *Model Scoping and Construction* toolkit supports three activities of the validation testing process: *Scoping the Application Context*, *Model Construction*, and *Model Refinement and Improvement*. These activities involve intensive analysis and judgment and, therefore do not allow for direct automation. The *Model Scoping and Construction* toolkit assists in referencing and constructing application models and preparing documentation but does not provide more complex automation. FIGURE 2 illustrates the flow of data between toolkits, as represented by the items shown in bold in TABLE 4.

FIGURE 2 **Data Flow Between Toolkits**


Integrated Resource Models
Application Model written in EXPRESS
Model Scoping & Construction
Test Case Data
Test Case Data Generation
Test Execution and Analysis
Executable Tests
Test Case Library

The output of the *Model Scoping and Construction* toolkit is the application model in both human- and computer-interpretable formats. These formats include documentation describing the application model, a graphical representation of the model (EXPRESS-G), and the definition of the model in EXPRESS. While these different views of the application model are used in the other toolkits, only the EXPRESS version of the application model is directly used as a basis for the other toolkits. The data consumed by the Model Scoping and Construction toolkit is contained in the existing STEP information models. These models provide a basis for the application model being developed. The input information models are a reference source and are not modified.

The *Test Definition* activity, which is supported by the *Test Definition* toolkit, also involves a great deal of human interaction and analysis and is the least automatable activity. As with the earlier activities, the automation is limited to assistance in referencing the application and activity models and in the preparation of documentation. None of the outputs of this toolkit are directly used by the other toolkits.

The primary automation for the *Test Case Data Generation* activity, which is supported by the *Test Case Data Generation* toolkit, is in preparing test data. The toolkit is initialized by configuring the tools to support the application model produced by the *Model Scoping and Construction* toolkit. This toolkit consumes product data from external sources and produces data which corresponds to the application model. The data consumed is represented in many formats, while the data produced is available in STEP exchange file format and may also reside in a database system.

For many applications, a reliable and efficient way to obtain a subset of the test data is in the form of an Initial Graphics Exchange Specification (IGES) [IPO91] file extracted from a CAD system. The IGES file can be translated into the format of the application model in the *Test Case Data Generation* toolkit. This IGES file represents a geometric model of a product and may provide 25% or more of the data needed depending on the application requirements; however, it may cover only about 10% of geometric representation entities from STEP. Additional data must be manually prepared to complete the information required for the application model.

The *Test Execution and Analysis* toolkit assists in generating executable test cases, executing the test cases, and analyzing the results. These activities allow for a great deal of automation. The tests can not be effectively conducted manually — they must be automated to validate the application models. In order to execute the test cases, a software environment for testing must be set up, and the test cases need to be formally specified in that environment. A database management system is used for the execution of the test cases.

The typical testing scenario is as follows:

1. the application model, provided in EXPRESS by the *Model Scoping and Construction* toolkit, is represented in the database system;
2. data, prepared using the *Test Case Data Generation* toolkit, is loaded into the database;
3. the executable test cases are specified in the system's query language;
4. these queries are executed to simulate the typical application data access requests against the data in the database; and
5. the results are recorded and analyzed.

Currently the primary means of transferring data into the database is via files in the STEP exchange file format [ISO21]. However, the VTS software is designed so that data can be directly stored in the database by the *Test Case Data Generation* toolkit. When fully implemented the *Test Execution and Analysis* toolkit will use that same database, thereby avoiding the transfer of data between the two toolkits. In this case the *Test Execution and Analysis* toolkit will appear as enhancements to the *Test Case Data Generation* toolkit.

The *Test Execution and Analysis* toolkit is used to generate executable test cases based on the test plan and abstract test cases developed earlier. These test cases are then executed using the data which resides in the toolkit. This toolkit produces the results of executing the tests and the computer-interpretable test cases. Both of these products are available for re-testing of the application model after it has been improved. The results are also used in the analysis and refinement activities.

In summary, the *Model Scoping and Construction* toolkit is used to generate the application model that is the basis for the *Test Case Data Generation* and the *Test Execution and Analysis* toolkits. The *Test Definition* toolkit is used to formulate the plans for testing the application model; this plan guides the testers, when they use the remaining two toolkits. The *Test Case Data Generation* toolkit is used to assemble product data to be used for testing. The *Test Execution and Analysis* toolkit is used to prepare executable test cases and to execute these test cases against the product data. The results of the tests are used for model analysis and refinement. The test cases themselves are saved for future testing of the evolving application model and for conformance testing of implementations of the AP.

The model refinement activity leads to a revised application model and may contribute to refinements in the other information models within STEP. Throughout the testing process, any deficiencies in the application model, the test cases, or the test environment are documented, and appropriate enhancements are made. The entire validation process is repeated on changed aspects of the model using the refined application model, test data, and testing environment.

The information flow between the toolkits reflects the information flow in the application area outside of the testing environment. For example, consider the application area of exchange of drawings based on three-dimensional computer-aided drafting models:

> The application model represents the data structures used for exchanging the data associated with drawings. In the testing environment the application model is developed by interviewing experts in the application area. A software designer would interview drafting experts to develop a information model for exchanging drawings within an organization.
>
> The data and its usage in the validation testing process directly reflects the needs of the application area. The application model is populated with data, using the same access paths that an organization would need to enter new drawing data into their system. Much of the data used for validation testing is contributed from industrial sources involved in the application area. The tests performed on the data ensure that it is accessible as required based on the usage scenarios developed in the test definition activity. The usage scenarios describe the information requirements for exchanging drawings. The library of executable test cases is based on reports or queries that are commonly used in drafting applications.

6 Validation Testing at the National PDES Testbed

The National PDES Testbed has been used for STEP validation testing since 1989. Initially, the software in the National PDES Testbed was collected from a variety of sources to provide some of the automation desired for the Model Construction, Test Case Data Generation, and Test Execution activities. This software, referred to as the *interim* system is described in detail in [Breese91]. This section summarizes the experiences with the interim system, improvements that have been made to that system, and direction for future additions to the validation testing software at the Testbed. Experiences with the interim system were used as a basis for planning and prioritizing future work.

6.1 Experiences with the Interim System

The interim software that supported the testing process consisted of a set of independent tools collected from a variety of sources. As a result of the diversity of their origins the tools operated in a variety of hardware and software environments. Using this software the method of sharing data throughout the testing process was to exchange data files between these tools. The environment required data translation, which introduced the potential for errors or inconsistencies every time data was moved between activities in the testing process. Moreover, the translation process and the associated, manually-activated process of importing and exporting data was time-consuming.

Automation for the validation testing process was provided by a set of software tools which translated both the application model and the test data among a variety of formats [Clark90a]. The interim software included: a data editor; a relational database management system; an EXPRESS parser and translators for representing the application model in the data editor and database system; an exchange file parser and loaders for populating the data editor and database; data export facilities for extracting data from the editor, database, and a commercial CAD system; an IGES to STEP translator for providing some testing data; and a visualizer and geometric modeler for displaying and manipulating a limited set of data.

The interim system lacked some necessary functionality and provided unacceptable performance. Significant improvements were to be made in the following areas:

- performance, in terms of both computation time and reliability;
- workflow automation to minimize the need for manual intervention between automated parts of the testing process;
- adoption of a data representation paradigm which more closely resembles that of EXPRESS; and
- expansion of the functionality to cover the requirements of model scoping and construction.

The data editor and database management system were targeted as the systems most needing improvement. Both of these tools suffered from significant performance problems with large sets of data [PDES90]. In addition, the integration of the two systems will help in automating the workflow and align the representation paradigm with that of EXPRESS.

The data editor in the interim system [Clark90b] was developed as a prototype; it was not designed to be used as a production system. Performance problems with the database loader are magnified by the fact that the mapping for each application model was not optimized for the relational database management system being used. Instead, a general-purpose mapping was applied so that new database schemas could be automatically generated.

Another significant improvement to the interim system was a more integrated software environment. The objective of integrating the software was to eliminate the need for manual intervention at a number of points in the validation process. The interim system required that the test case data developed on a CAx system go through multiple translations or processing steps before it was usable for executing a test. Each translation or processing step required manual intervention that created an opportunity for introducing new errors or using the wrong version of a file.

In addition, in the interim system many necessary functions were supported by separate utilities. The use of some of these utilities required that the data be exported from the data editor or database, run through the utility, and imported back into the system; other utilities needed to be run before data could be loaded into a system.

The data editor and database system in the interim system used different data representation paradigms for representing the information involved in the testing process. This placed a burden on the testers, the users of the validation software, to understand the different representational formats for the information model and corresponding data and the relationship between these formats. This complexity was one of the more difficult problems facing the testers.

EXPRESS uses a representation paradigm which allows a hierarchical decomposition of information as well as the representation of a semantic network of information. This representation is much richer than that provided by the relational paradigm [Date90] used in the interim database implementation. The relational paradigm reduces all information to flat table structures and does not support any of the semantics of the associations between the tables. The data editor used in the interim software used an object-oriented paradigm [Gold85] that more closely resembled EXPRESS.

The testers were faced with learning not only the method of representing the application model in EXPRESS but also the format for representing the information in the various tools used in the testing process. The relational database system in the interim system used a different paradigm for representing data. The meaning of the representation in the relational system had to be inferred from the individual tester's interpretation of EXPRESS and the tester's understanding of the mapping of the application model into the relational database system. As with any interpretation, this left room for ambiguity and misunderstanding.

6.2 Directions for the Validation Software

The software needs for the STEP AP validation process were divided into two categories:

- automation *of* the validation process, by simulating the data access requirements of the application area; and
- automation *to support* the validation process, through assistance for preparing documentation and for referencing and browsing the application model.

The first category — automation of the process through simulation of the data access requirements for the application area — is mandatory for validation testing. The testing process is not complete if the tests are not computer-processible and repeatable. These tests reflect the intended usage of the application model. The interim system addressed the need to automate the validation process, and improvement of this software has been the first priority for further work. Two tools are most important for supporting the first category of automation:

1. a structured data editor and
2. a database system with query language.

The tools for these functions in the interim system did not satisfy performance and reliability requirements. Furthermore, since the tools were not integrated, problems arose related to data translation, application model and data configuration, and clarity due to the use of different data representation paradigms.

Initial efforts for the VTS software focused on developing a replacement for the data editor [Morris93][Morris92a][Morris92b][Sauder93] which was the weakest component of the interim system. The new editor provides better performance, is more reliable, and supports a broader range of functions. Many of these functions were supported by separate utilities in the interim system.

An effort is now underway to integrate the new data editor with a database system. However, use of the editor does not depend on having a database system. Furthermore, the integration of the database into the system will not cause the end-user interface to change, although additional operations may be available for the user as a result of adding the database to the system.

The new VTS software provides an integrated set of functions for addressing the four toolkits. The integration of the functions provides a more efficient environment that can better simulate the data access needs of the application areas being tested. The VTS software:

- reduces the frequency of data translation and human intervention which should reduce the number of errors;
- provides better and more extensive error checking which reduces the potential for errors;
- improves performance and automation of the workflow, which reduces the amount of time needed for the testing process;
- provides more sophisticated support for data editing and creation, which should result in fewer inconsistencies in the data;
- provides a single interface to the software, which will reduce the time needed for learning to use the different tools.

A complete VTS environment will allow the users to operate on the same data set throughout the testing process without the need for translations to share data between toolkits.

The second category of software needs — automation to support the validation process — includes assistance in document preparation and with referencing and browsing the application model in any of the various formats (i.e. English, EXPRESS, EXPRESS-G, etc.). In the interim system this need has been met by commercial word-processing and drawing packages; however. these packages provide only limited support for the desired functions. An effort is underway within the National PDES Testbed to establish an Application Protocol Development Environment (APDE) which more thoroughly addresses these needs. The APDE will contribute to all of the activities of the validation process. In addition, the APDE will include a repository of the existing information models included in STEP. Such a repository will ensure that the information models are available in a consistent format that is useful to the users and is usable by the other software components of the system.

7 Summary

This paper has described the use of Application Protocols within STEP. APs provide a mechanism for applications to share data within a specific discipline. The Associative Draughting AP (AP 202) has been used as an example. As of this writing, 19 AP projects have been approved as work items with ISO and many more are emerging. The content of these APs is quiet varied. The success of the standard relies on a firm methodology for developing and integrating the APs. Validation testing is an important part of that methodology and insures that the application models are meeting their requirements. Software to support validation is necessary and integral to the testing process.

The approach to validation described here has been developed at the National PDES Testbed. Other approaches may also satisfy this objective. The software developed at NIST to support validation is available in the public domain. Contact the Factory Automation Systems Division at NIST to obtain the software or for any information related to NIST work on the National PDES Testbed project.

8 References

[Breese91] Breese, J. Newton, Michael McLay, and Gerard Silvernale, Validation Testing Laboratory User's Guide, NISTIR 4683, National Institute of Standards and Technology, Gaithersburg, MD, October 1991.

[Clark90a] Clark, S. N., An Introduction to The NIST PDES Toolkit, NISTIR 4336, National Institute of Standards and Technology, Gaithersburg, MD, May 1990.

[Clark90b] Clark, S.N., QDES User's Guide, NISTIR 4361, National Institute of Standards and Technology, Gaithersburg, MD, June 1990.

[Date90] Date, C. J., An Introduction to Database Systems: Volume 1, Fifth Edition, Addison-Wesley, 1990.

[Gold85] Goldberg, A. and D. Robson, Smalltalk-80: The Language and Its Implementation, Addison-Wesley, Reading, MA, July 1985.

[ICA85] Information Modeling Manual IDEF Extended (IDEF1X), ICAM Project Report (Priority 6201), Air Force Systems Command, Wright-Patterson Air Force Base, OH, USA, December 1985.

[IPO91] The Initial Graphics Exchange Specification (IGES), Version 5.1, IGES/PDES Organization, NCGA, Fairfax, VA, September 1991.

[ISO1] International Organization for Standardization, *ISO 10303 Industrial Automation Systems and Integration— Product Data Representation and Exchange — Overview and Fundamental Principles*, Draft International Standard, ISO TC184/SC4, 1992.

[ISO11] International Organization for Standardization, *ISO 10303 Industrial Automation Systems and Integration — Product Data Representation and Exchange — Description Methods: The EXPRESS Language Reference Manual*, Draft International Standard, ISO TC184/SC4, 1992.

[ISO21] International Organization for Standardization, *ISO 10303 Industrial Automation Systems and Integration — Product Data Representation and Exchange — Clear Text Encoding of the Exchange Structure*, Draft International Standard, ISO TC184/SC4, 1993.

[ISO22-CD] International Organization for Standardization, *ISO 10303 Industrial Automation Systems and Integration — Product Data Representation and Exchange — Standard Data Access Interface Specification*, Committee Draft, ISO TC184/SC4, 1993.

[ISO31] International Organization for Standardization, *ISO 10303 Industrial Automation Systems and Integration — Product Data Representation and Exchange — Part 31: Conformance Testing Methodology and Framework: General Concepts*, Draft International Standard, ISO TC184/SC4, 1992.

[ISO41] International Organization for Standardization, *ISO 10303 Industrial Automation Systems and Integration — Product Data Representation and Exchange — Part 41: Fundamentals of Product Description and Support*, Drafit International Standard, ISO TC184/SC4/WG4, 1993.

[ISO42] International Organization for Standardization, *ISO 10303 Industrial Automation Systems and Integration — Product Data Representation and Exchange — Geometric and Topological Representation,* Draft International Standard, ISO TC184/SC4, 1993.

[ISO202-CD] International Organization for Standardization, *ISO 10303 Industrial Automation Systems and Integration — Product Data Representation and Exchange — Part 202: Application Protocol: Associative Draughting,* Committee Draft, ISO TC184/SC4/WG3, 1993.

[ISO203] International Organization for Standardization, *ISO 10303 Industrial Automation Systems and Integration — Product Data Representation and Exchange — Part 203: Configuration Controlled Design,* Draft International Standard, ISO TC184/SC4, 1993.

[KATZ91] Katz, Susan, Step On-Line Information Service User's Guide, NISTIR 4491, National Institute of Standards and Technology, Gaithersburg, MD, January, 1991.

[LOO86] Loomis, M., "Data Modeling - the IDEF1X Technique", *IEEE Conference on Computers and Communications*, Pheonix, Arizona, March, 1986, pp. 146-151.

[Mitch90] Mitchell, M., Validation Testing Systems Plan, NISTIR 4417, National Institute of Standards and Technology, Gaithersburg, MD, October 1990.

[Mitch91] Mitchell, M., A Proposed Testing Methodology for STEP Application Protocol Validation, NISTIR 4684, National Institute of Standards and Technology, Gaithersburg, MD, September 1991.

[Mitch92] Mitchell, M. J., Morris, K. C. The Use of Application Model Validation in Testing a Proposed Standard, *Proceedings of the Sixth Annual ASME Database Symposium - Engineering Data Management: Key to Integrated Product Development,* American Society of Mechanical Engineers, New York, August 1992.

[Morris91]	Morris, K.C., McLay, M., and Carr, P. J., Validation Testing System Requirements, NISTIR 4676, National Institute of Standards and Technology, Gaithersburg, MD, September 1991.
[Morris92a]	Morris, K.C., Architecture for the Validation Testing System Software, NISTIR 4742, National Institute of Standards and Technology, Gaithersburg, MD, January 1992.
[Morris92b]	Morris, K. C., Sauder, D., Ressler, S., Validation Testing System: Reusable Software Component Design, NISTIR 4937, National Institute of Standards and Technology, Gaithersburg, MD, October 1992.
[Morris93]	KC Morris, Data Probe: A Tool for EXPRESS—based Data, *Proceedings of the Seventh Annual ASME Database Symposium - Engineering Data Management: Key to Success in a Global Market,* American Society of Mechanical Engineers, New York, August 1992.
[Palmer93]	Mark Palmer and Mitch Gilbert, *Guidelines For The Development and Approval of STEP Application Protocols*, Version 1.1, working draft of ISO TC184/SC4/WG4, January 1993.
[PDES90]	Test Report for Context-Driven Integrated Model (CDIM) Application A1, Skeels, J., ed., PDES, Inc. internal report PMG012.01.00, SCRA, Charleston, SC, April 1990.
[Sauder93]	David Sauder, Data Probe User's Guide, NISTIR 5141, National Institute of Standards and Technology, Gaithersburg, MD, March 1993.
[Stark91]	Stark, C., and Mitchell, M., Development Plan: Application Protocol for Mechanical Parts Production, NISTIR 4628, National Institute of Standards and Technology, Gaithersburg, MD, July 1991.