STEP, XML, and UML: Complementary Technologies[1]

Russell S. Peak
ASME Mem.,
Georgia Institute of Technology,
Manufacturing Research Center, 813 Ferst Drive,
MARC 373, Atlanta, GA 30332-0560
e-mail: russell.peak@marc.gatech.edu

Joshua Lubell
National Institute of Standards and Technology, 100
Bureau Drive, Stop 8263, Gaithersburg, MD 20899-8263
e-mail: lubell@nist.gov

Vijay Srinivasan
ASME Mem.,
IBM Corporation, 1133 Westchester Avenue,
Mail Drop 160, White Plains, NY 10604
e-mail: vasan@us.ibm.com

Stephen C. Waterbury
NASA Goddard Space Flight Center, Code 562,
Greenbelt, MD 20771
e-mail: stephen.c.waterbury@nasa.gov

One important aspect of product lifecycle management (PLM) is the computer-sensible representation of product information. Over the past 15 years or so, several languages and technologies have emerged that vary in their emphasis and applicability for such usage. ISO 10303, informally known as the Standard for the Exchange of Product Model Data (STEP), contains the high-quality product information models needed for electronic business solutions. By using STEP, the aerospace, automotive, and shipbuilding industries are saving $150M/yr primarily in areas related to geometric modeling. However, traditional STEP-based model information is represented using languages that are unfamiliar to most application developers, thus impeding widespread usage in other areas.

This paper discusses efforts underway to make STEP information models available via mechanisms familiar to more business application developers, specifically XML and the Unified Modeling Language™ (UML®). We also present a vision and roadmap for STEP integration with XML, UML, and other technologies to enable enhanced PLM interoperability.

Our conclusion is that STEP, XML, and UML are complementary technologies, where STEP provides significant standardized content models, while XML and UML provide enhanced implementation methods. Together, they are a powerful force to enable pervasive digital representation and sharing of diverse technical information.

[DOI: 10.1115/1.1818683]

1 Introduction

Many businesses are turning to business-to-consumer and business-to-business solutions based on the Extensible Markup Language (XML) [1] to reduce transaction costs, open new markets, and better serve their customers. These solutions, which tend to emphasize messaging and business processes, require basic information about products. Many of these XML business vocabularies are adhoc and/or conflict with other XML applications. Often missing from these solutions is a rigorous definition of the business information concerning the design, manufacture, and support of these goods. To ensure the longevity of this business data, it should be represented using a language defined by an open standard and not exclusively dependent on a particular software vendor. In this paper we discuss three open standards we believe are useful for representing product information: the Standard for the Exchange of Product Model Data (STEP) [2,3], XML, and the Unified Modeling Language™ (UML®) [4].

1.1 Types of Standards. What do we mean by a ''standard'' in the context of technical information management? Such standards may be one of at least three types:

1. Open Standards relate to the general idea of interoperability and integration—an agreement that people make so that products and systems made by different parties can work together. Open standards are not software applications; they are only specifications explaining how information should look. Open standards are developed by consensus in an industry group. There is a tremendous variation in the membership rules and processes for these organizations, and they range from official organizations, such as the International Organization for Standardization (ISO, http://www.iso.ch), to small vertical industry groups.

STEP is an example of an open standard. It is developed by ISO, with the help of industrial consortia, such as PDES, Inc. (http://pdesinc.aticorp.org) and ProSTEP (http://www.prostep.de). XML and UML are also open standards, even though they are being developed outside ISO. XML is developed by the World Wide Web Consortium (W3C, http://www.w3.org), and UML is developed by the Object Management Group (OMG, http://www.omg.org). We note that, even though XML and UML standardization is taking place outside ISO, both of these technologies are closely tied to published ISO standards. The W3C's XML 1.0 standard is equivalent to ISO's ''Web SGML Adaptations'' [5] of the Standard Generalized Markup Language (SGML) [6] and is, in fact, a subset of SGML. Although UML standardization occurs primarily within the OMG, UML is also being standardized ''after the fact'' in ISO. The most recent version of ISO UML [7] is based on an earlier version (1.4) of OMG UML.

2. Industry Standards are technologies that are commonly used, but are not open or democratically managed by a group of users. The Java™ technology is a well-known example of an industry standard. There are a number of companies involved in the Java Community Process (http://jcp.org), but because one company wields a tremendous amount of control over the process, Java is classified as an industry standard, not an open standard.

3. De facto Standards are in wide use because of their value or association with other technologies, and not necessarily because they were produced by a standards organization. A commercial software product may be a *de facto* standard because of its wide adoption. The Microsoft Windows® operating system is a *de facto* standard for personal computers. The Simple Object Access Protocol (SOAP) [8] was initially a *de facto* standard because of its broad use in web services, though it has now been formalized as

[1]Commercial equipment and materials are identified in order to describe certain procedures. In no case does such identification imply recommendation or endorsement by the authors or their organizations nor does it imply that the materials or equipment identified are necessarily the best available for the purpose. Unified Modeling Language, UML, Object Management Group, OMG, and XMI are trademarks or registered trademarks of the Object Management Group, Inc., in the U.S. and other countries. Java is a trademark or registered trademark of Sun Microsystems, Inc. Other company, product, and service names may be trademarks or service marks of others.

Contributed by the Engineering Informatics (EIX) Committee for publication in the JOURNAL OF COMPUTING AND INFORMATION SCIENCE IN ENGINEERING. Manuscript received May 8, 2004; Revised September 18, 2004. Associate Editor: R. Rangan.

an open standard in the W3C. *De facto* standard status does not mean that there are no alternatives to a particular technology; such alternatives are just less commonly used.

In addition to the three types of standards mentioned above, there is open-source software. Open-source software is not necessarily an open standard. Open-source refers to software source code that is available to the general public and does not have licensing restrictions that limit use, modification, or redistribution under the same terms as the license of the original software.[2] The GNU/Linux operating system (http://www.linux.org) and Eclipse software development environment (http://www.eclipse.org) are examples of open-source technologies. Some companies frequently release software as open-source when they want to lower the barrier of entry for certain technologies. The XML4J (XML for Java) XML parser (http://www.alphaworks.ibm.com/tech/xml4j) and the Apache project's SOAP implementation (http://ws.apache.org/soap) are two such examples.

1.2 STEP: Powerful Content Models. ISO 10303, also informally known as the Standard for the Exchange of Product Model Data (STEP), is a family of standards defining a robust and time-tested methodology for describing product data throughout the life cycle of a product. STEP is widely used in computer-aided design (CAD) and product data/life-cycle management (PDM/PLM) systems. Major aerospace, automotive, and shipbuilding companies have proven the value of STEP through production implementations resulting in actual savings of $150M/yr in the U.S. (and potential savings of $928M/yr) [9,10]. STEP contains the high-quality and high-fidelity information models many XML business applications require.

An application protocol (AP) in STEP describes the information model of a particular engineering or technical domain. For example, AP203 is for configuration-controlled mechanical assembly design, AP209 represents finite element analysis models, and AP210 captures electronic/mechatronic design information. APs are developed using a rigorous process that promotes reuse of common ''building block'' models as well as the integration of industrial requirements with these common resources. APs are what software developers implement (most often as import/export interfaces to their CAX[3] and PLM applications). APs and the resources used to develop them contain formally specified information models written in a language created specially for STEP. Kemmerer [3], Pratt [11], and Peak [12] provide overviews of STEP, the structure of APs, and overviews of example APs and their usage.

The objects represented and exchanged using STEP, as well as the associations between these objects, are defined in schemas written in EXPRESS (ISO 10303-11) [13], an information modeling language combining ideas from the entity-attribute-relationship family of modeling languages with concepts from object-oriented modeling. This language existed long before UML, XML, and the Internet as we know it today. It was developed in the 1980s when few, if any, alternatives existed that had the combined representational capabilities needed for the ambitious scope of STEP. These desired integrated capabilities included computer- and human-sensible formulations (both lexical and graphical), constraint specifications, scalability and flexibility for large schema and instance models, and implementability [3].

[2]See http://www.opensource.org/docs/definition_plain.php for a more detailed definition of ''open-source.''

[3]CAX stands for computer-aided X, where X = design, engineering, manufacture, sustainment, etc.

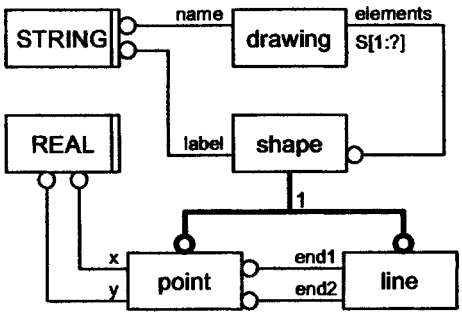

Fig. 1 EXPRESS-G diagram for the simple_drawings schema

The following *simple_drawings* EXPRESS schema represents two-dimensional drawings consisting of points and lines. Figure 1 is the EXPRESS-G graphical formulation of this schema. Although this example is very simple, it illustrates several EXPRESS language features. These include:

- Built-in basic types, such as *STRING* and *REAL*
- Constructs representing collections, such as *SET*
- Inheritance of types (*point* and *line* both inherit properties from *shape*)

```
SCHEMA simple_drawings;

ENTITY drawing;
  name : STRING;
  elements : SET [1:?] OF shape;
END_ENTITY;

ENTITY shape;
  label : STRING;
END_ENTITY;

ENTITY point SUBTYPE OF (shape);
  x : REAL;
  y : REAL;
END_ENTITY;

ENTITY line SUBTYPE OF (shape);
  end1 : point;
  end2 : point;
END_ENTITY;

END_SCHEMA;
```

EXPRESS has more capabilities than this simple example suggests. It can represent complex inheritance relationships and functions, and it includes a rich set of constructs for specifying constraints on populations of instances. In the interest of brevity we do not focus on advanced EXPRESS language features in this paper. However, as an example of a relatively basic EXPRESS *WHERE* constraint, consider the following definition for a point on a parabola represented by the equation $y = x^2$:

```
ENTITY point_on_parabola SUBTYPE OF (point);
WHERE
     parabola : y=x**2;
END_ENTITY;
```

WHERE expressions can be far more complicated than the parabolic equation above. For example, the following is a *WHERE* constraint on ordinal dates from an actual STEP standard [14]. The constraint enforces the rule that the date's *day_component* must be an integer from 1 to 365 or, if the year is a leap year, from 1 to 366. *leap_year* is an EXPRESS function (not shown in the example) defined to compute whether the date's *year_component* is a valid leap year.

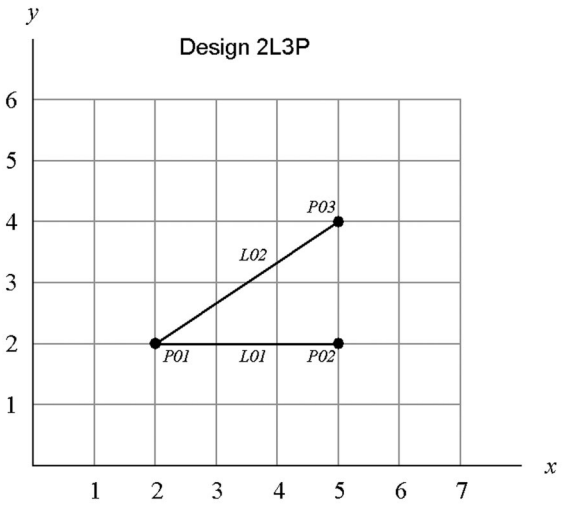

Fig. 2 A sample drawing consisting of two lines connecting three points: "Design 2L3P"

```
WHERE
  wr1: (((NOT leap_year(SELF.year_component))
       AND (1 <= day_component)
       AND (day_component <= 365))
    OR (leap_year(SELF.year_component)
       AND (1 <= day_component)
       AND (day_component <= 366)));
```

Although EXPRESS is a powerful language, it is relatively unknown to most programmers. Moreover, STEP data (i.e., an instance population of an EXPRESS schema) are typically exchanged as files using an ASCII character-based syntax defined in ISO 10303-21 (also known as STEP "Part 21") [15]. For example, consider the drawing shown in Fig. 2. A Part 21 instance population that represents this information based on the above *simple_drawings* EXPRESS schema is as follows:

```
#10 = point ('P01', 2.0, 2.0);
#20 = point ('P02', 5.0, 2.0);
#30 = point ('P03', 5.0, 4.0);
#110 = line ('L01', #10, #20);
#150 = line ('L02', #10, #30);
#200 = drawing ('Design 2L3P',
        (#10, #20, #30, #110, #150));
```

The type of instance is indicated first (e.g., *point*) followed by a list of attribute values (where the list is ordered based on the attributes sequence in the EXPRESS definition of the entity).

Hash numbers like #10 and #20 denote object instance identifiers so that instances can be easily referenced and used elsewhere in the population. For example, the #10 and #30 used in the *line* L02 instance indicate that its *end1* and *end2* properties correspond to *points* P01 and P03, respectively. Hash numbers are valid within the scope of a single STEP instance population (usually a text file ending in .stp, .step, or .p21), and thus their actual values are arbitrary as long as they are consistently used within the population.

Aggregate members are enclosed in parentheses as exemplified in the #200 *drawing* instance, where the hash numbers of all five drawing *elements* in Fig. 2 are contained.

The Part 21 syntax, although effective for the task at hand, lacks extensibility, can be hard for humans to read, and—perhaps most limiting—is computer interpretable by a relatively small number of software development toolkits that support STEP.

1.3 XML and UML: Ubiquitous Implementation Tools. Unlike the STEP Part 21 syntax, XML is easily extensible and is supported by numerous inexpensive and widely used software tools. Thus, from the perspective of a typical programmer, it is easier to render XML data into forms that are suitable for human perusal. Many applications developed today that import or export data utilize or support some form of XML format. Finally, XML is used in virtually all new work done on developing standard data formats for many domains, including some domains that are within the scope of STEP. So it is only natural that traditional STEP technology has been updated to accommodate itself to XML.

UML is a widely accepted and supported standard software modeling language. UML tools abound. By contrast, the tools that are used to develop and manage STEP schemas and instance populations have a relatively small user community. STEP modeling tools are adept at the EXPRESS language and can develop and validate very complex information. However, while EXPRESS is a powerful information modeling language, it has traditionally been relatively unknown in the world of general software modeling methods. For software developers who need to deal with STEP instances, it would be a tremendous boon to be able to capture, model, and visualize the relationships between STEP constructs and the other information types that they use in their development process. UML, through its class diagrams and through its profile extensibility mechanism, is already being used to visually represent XML schemas [16]. As we shall see in the next section, the capability to visualize STEP constructs in UML will be available soon, as methods for integrating EXPRESS schemas with UML models are now emerging.

2 Can STEP Work with XML and UML?

2.1 STEP and XML. In order to capitalize on XML's popularity and flexibility and to accelerate STEP's adoption and deployment, ISO is developing a standard for representing EXPRESS schemas and instance populations in XML. The expectation is that this emerging standard, ISO 10303-28, *Implementation methods: XML Schema governed representation of EXPRESS schema governed data* [17,18], will not only enable developers to use low-cost, ubiquitous XML software tools to implement file-based exchange and visualization of STEP instances, but will also potentially facilitate the use of STEP information in emerging areas, such as XML-based web services.

For our *simple_drawings* example in Section 1.2, the W3C XML Schema [19] generated from the EXPRESS schema using Part 28 might look like this:

```
⟨xs:schema xmlns:xs=‘‘http://www.w3.org/2001/XMLSchema’’⟩

 ⟨xs:attributeGroup name=‘‘OID’’⟩
  ⟨xs:attribute name=‘‘id’’ type=‘‘xs:ID’’ use =‘‘optional’’/⟩
 ⟨/xs:attributeGroup⟩

 ⟨xs:element name=‘‘p28data’’⟩
  ⟨xs:complexType⟩
   ⟨xs:choice minOccurs=‘‘1’’ maxOccurs=‘‘unbounded’’⟩
     ⟨xs:element name=‘‘drawing’’ type=‘‘Drawing’’/⟩
     ⟨xs:element name=‘‘point’’ type=‘‘Point’’/⟩
     ⟨xs:element name=‘‘line’’ type=‘‘Line’’/⟩
   ⟨/xs:choice⟩
  ⟨/xs:complexType⟩
 ⟨/xs:element⟩

 ⟨xs:complexType name=‘‘Drawing’’⟩
  ⟨xs:sequence⟩
   ⟨xs:element name=‘‘name’’ type=‘‘xs:string’’/⟩
   ⟨xs:element name=‘‘elements’’⟩
     ⟨xs:complexType⟩
      ⟨xs:choice maxOccurs=‘‘unbounded’’⟩
        ⟨xs:element name=‘‘line’’ type=‘‘Line-ref’’/⟩
        ⟨xs:element name=‘‘point’’ type=‘‘Point-ref’’/⟩
      ⟨/xs:choice⟩
     ⟨/xs:complexType⟩
   ⟨/xs:element⟩
  ⟨/xs:sequence⟩
  ⟨xs:attributeGroup ref=‘‘OID’’/⟩
 ⟨/xs:complexType⟩

  ⟨xs:complexType name=‘‘Shape’’⟩
   ⟨xs:sequence⟩
     ⟨xs:element name=‘‘label’’ type=‘‘xs:string’’/⟩
   ⟨/xs:sequence⟩
   ⟨xs:attributeGroup ref=‘‘OID’’/⟩
  ⟨/xs:complexType⟩

  ⟨xs:complexType name=‘‘Point’’⟩
   ⟨xs:complexContent⟩
    ⟨xs:extension base=‘‘Shape’’⟩
     ⟨xs:sequence⟩
       ⟨xs:element name=‘‘x’’ type=‘‘xs:decimal’’/⟩
       ⟨xs:element name=‘‘y’’ type=‘‘xs:decimal’’/⟩
     ⟨/xs:sequence⟩
    ⟨/xs:extension⟩
   ⟨/xs:complexContent⟩
  ⟨/xs:complexType⟩
  ⟨xs:complexType name=‘‘Point-ref’’⟩
   ⟨xs:attribute name=‘‘ref’’ type=‘‘xs:IDREF’’/⟩
  ⟨/xs:complexType⟩

  ⟨xs:complexType name=‘‘Line’’⟩
   ⟨xs:complexContent⟩
    ⟨xs:extension base=‘‘Shape’’⟩
     ⟨xs:sequence⟩
       ⟨xs:element name=‘‘end1’’ type=‘‘Point-ref’’/⟩
       ⟨xs:element name=‘‘end2’’ type=‘‘Point-ref’’/⟩
     ⟨/xs:sequence⟩
    ⟨/xs:extension⟩
   ⟨/xs:complexContent⟩
  ⟨/xs:complexType⟩
  ⟨xs:complexType name=‘‘Line-ref’’⟩
   ⟨xs:attribute name=‘‘ref’’ type=‘‘xs:IDREF’’/⟩
  ⟨/xs:complexType⟩

 ⟨/xs:schema⟩
```

Our ''Design 2L3P'' drawing instance (Fig. 2) can be represented as an XML document conforming to the above Part 28-generated XML schema. The resulting lexical representation is as follows:

```
⟨p28data⟩
 ⟨point id=''_10''⟩
  ⟨label⟩P01⟨/label⟩
  ⟨x⟩2.0⟨/x⟩
  ⟨y⟩2.0⟨/y⟩
 ⟨/point⟩
 ⟨point id=''_20''⟩
  ⟨label⟩P02⟨/label⟩
  ⟨x⟩5.0⟨/x⟩
  ⟨y⟩2.0⟨/y⟩
 ⟨/point⟩
 ⟨point id=''_30''⟩
  ⟨label⟩P03⟨/label⟩
  ⟨x⟩5.0⟨/x⟩
  ⟨y⟩4.0⟨/y⟩
 ⟨/point⟩
 ⟨line id=''_110''⟩
  ⟨label⟩L01⟨/label⟩
  ⟨end1 ref=''_10''/⟩
  ⟨end2 ref=''_20''/⟩
 ⟨/line⟩
 ⟨line id=''_150''⟩
  ⟨label⟩L02⟨/label⟩
  ⟨end1 ref=''_10''/⟩
  ⟨end2 ref=''_30''/⟩
 ⟨/line⟩
 ⟨drawing id=''_200''⟩
  ⟨name⟩Design 2L3P⟨/name⟩
  ⟨elements⟩
   ⟨point ref=''_10''/⟩
   ⟨point ref=''_20''/⟩
   ⟨point ref=''_30''/⟩
   ⟨line ref=''_110''/⟩
   ⟨line ref=''_150''/⟩
  ⟨/elements⟩
 ⟨/drawing⟩
⟨/p28data⟩
```

Even in this simple example one can see relative benefits and weaknesses in different approaches for representing the same or similar information. For example, the Part 21 instance omits the names of properties such as *x*, *y*, *end1*, and *end2*. The Part 28 instance, on the other hand, includes this information. Hence, Part 21 instances are more concise than Part 28 instances (resulting in smaller file sizes), whereas Part 28 instances are more human readable to a person who is not familiar with the schema(s) to which those instances conform. However, human readability is not necessarily among the most important requirements for data exchange and serialization formats, as direct visual inspection of the source is not the primary means for using, validating, or debugging such formats.

One advantage of both XML and STEP Part 21 formats is the ability to reference complete instances via object instance identifiers, as opposed to implied referencing using ''magic strings'' (e.g., using the ''P01'' *label* attribute as a magic string to indicate the *point* P01). Whereas such magic strings are convenient for human interpretation, explicit object instance identifier methods offer more robustness for computer interpretation. Part 21 achieves this capability using identifiers like #10, described above, to indicate that *point* P01 is used as *end1* in *line* L01.

While XML does not require this approach, it supports it through means including *ID* and *IDREF* types and the XPath [20] *id* function. In the XML example above, attributes, such as *id =_10*, are implemented in this way (where the convention is taken that Part 21 hash marks are mapped to underscores, such that #10 becomes *_10* and so on, because hash characters are not allowed in XML *ID* types). Thus, software developers and information modelers obtain this referencing capability explicitly with Part 21 and XML, whereas it is often implied in other formats and must be supported with additional programing.

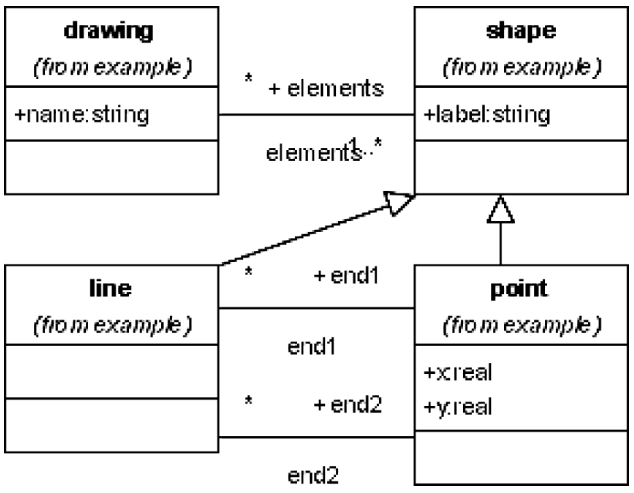

Fig. 3 A UML class diagram obtained from the *simple_drawings* EXPRESS schema via Part 25

2.2 STEP and UML. The ISO STEP committee (TC184/SC4) is also developing another standard, *ISO 10303-25, EXPRESS to OMG XMI binding* [21,22] (also known as Part 25), for transforming EXPRESS schemas into UML models. This will enable developers to use their familiar UML tools to see the contents of STEP (EXPRESS) schemas and eventually to specify relationships between STEP information models and the other UML models that they use. A Part 25 mapping from our EXPRESS schema to the XML Metadata Interchange (XMI®) format [23] would produce a UML class diagram like that shown in Fig. 3.

The current version of Part 25 (ISO/CD TS 10303-25) includes a mapping from EXPRESS to XMI that covers most of the basic EXPRESS productions, with the exception of several of its more technical features that are commonly used to implement constraints (such as *RULE*, *PROCEDURE*, and *FUNCTION* declarations and *UNIQUE* rules). The mapping is also one-way only (i.e., from EXPRESS to XMI, but not yet from XMI to EXPRESS).

There is still more work to do to make EXPRESS and UML interoperable, but a significant start has been made. Tentative plans for the next edition of Part 25 include

- A UML Profile of EXPRESS to accommodate some of the productions not directly mappable to core UML concepts
- Support for EXPRESS 2 (which is currently used in some STEP APs)
- An OMG Meta-Object Facility (MOF) definition of EXPRESS, which would have the benefit of bringing EXPRESS itself into the UML family of languages and offering the possibility of native support for EXPRESS in UML tools
- A UML-to-EXPRESS mapping, enabling round-trip metadata interchange

For more information about Part 25 and the relationship between EXPRESS and UML, see [22].

So the answer to the question ''Can STEP work with XML and UML?'' is *yes*. Capabilities for STEP to interoperate with and integrate with XML and UML are now emerging.

3 Software Tools for STEP and XML

Until recently, there were few software tools for using STEP schemas and instance populations in the XML and UML worlds.

Table 1 Primary information modeling technologies for standards-based PLM frameworks (after [27])

Schema language	Mapping language	Serializations (lexical) and interface methods (APIs, . . .)	Standardized content schemas
EXPRESS	Express-X	Part 21, Part 28 ed. 1 and 2 (XML), Part 25 (XMI), . . .	ISO 10303 series (STEP)—O(1000) man-years of effort and O(10,000) standardized technical concepts
XML Schema, DTD	XSLT	Part 28 ed.2 (XML Schema), . . .	ChemML,[a] GenCAM,[b] FemML,[c] MathML,[d] MatML[e] PLM XML, PDTnet,[f] SVG,[g] . . .
UML (XMI, . . .)	QVT[h]	Web (XSP),[i] SOAP, WSDL, CORBA, PB,[j] . . .	UML Profiles emerging (e.g., SysML[k] for systems engineering)

[a]http://www.xml-cml.org
[b]http://gencam.ipc.org
[c]http://www.istos.org/femML, see also [28].
[d]http://www.w3.org/Math
[e]http://www.matml.org
[f]http://www.pdtnet.org
[g]http://www.pdtnet.org
[h]MOF 2.0 Query/Views/Transformations RFP, OMG Document ad/02-04-10 (http://www.omg.org/egi-bin/doc?ad/02-04-01).
[i]Web (XSP)=''X'' Server Pages (where X can be A, J, P, etc.).
[j]PB=Perspective Broker (see: http://twistedmatrix.com).
[k]http://www.sysml.org

However, there are now several promising development efforts underway to create such software that capitalizes on the popularity of XML and UML.

• The STEP Module Repository (http://stepmod.sourceforge.net) is a collection of resources tagged in XML to serve as the core of a modular environment for developers of STEP and related standards. The Extensible Style Language Transformation (XSLT) standard [24] is used to produce both ISO-compliant as well as developer-friendly documentation.

• The ''EXPRESS For Free'' (*exff*) project (http://exff.sourceforge.net) is developing tools to convert between EXPRESS and UML. The initial goal is to be able to employ UML-based code generation tools to help implement STEP. Future plans include supporting the use of UML modeling tools to build EXPRESS schemas. The current *exff* release provides translators between XMI and EXPRESS marked up in XML using the STEP Module Repository Document Type Definition.

• The National Shipbuilding Research Program (http://www.nsrp.org) has implemented a translator for its Integrated Shipbuilding Environment from STEP Application Protocol AP218 (Ship structures) to AP209 (Composite and metal structural analysis and related design). The translator uses Part 28 to represent both APs and XSLT to convert from one to the other.

• Various STEP software vendors (http://pdesinc.aticorp.org/step_products.html) are developing Part 28 EXPRESS-to-XML translators and/or have products that import or export product model data in an XML format. For example, one STEP vendor uses XML in several ways in addition to supporting Part 28: managing converter configuration settings and capturing executable user interface specifications [25] in an EXPRESS-based application for AP210 (Electronic assembly interconnect and packaging design).

Meanwhile, other tools and formats are emerging that overlap with STEP in some respects:

• PLM XML (http://www.ugs.com/products/open/plmxml/) is a proprietary XML format for enabling product life-cycle interoperability. Based on W3C XML Schemas, PLM XML contains product information and geometric representation data.

• JT, a proprietary format based on the Jupiter technology [26], is aimed at being an efficient format for collaboratively visualizing and interacting with large 3D models. The scene graph can also contain different levels of detail to minimize memory footprints while viewing large assemblies. A JT file may also include assembly structure, text, and symbolic annotations (e.g., tolerances). A recently formed JT Consortium (http://www.jtopen.com/) is endeavoring to make JT an industry standard.

JT is one example of the trend in the 3D visualization arena where alternative neutral formats are being developed (e.g., by organizations such as Lattice-3D (XVL), Actify (.3d), and Autodesk (dwf), including some with XML-based formats). These alternatives are aimed at enhanced data compression, streaming capabilities, visualization optimization, and publishing that is authoring tool-independent. While a full treatment of this topic is beyond the scope of this paper, we feel there is opportunity for synergy and harmonization with STEP-based shape representations (which are targeted at robust representation for high-diversity exchange and long-term archiving, whereas these newer formats are focused towards lightweight visualization and real-time collaboration).

4 Suites of Standards: How the Pieces Fit

While it is useful to compare the expressiveness and richness of information modeling languages, it is perhaps even more important to understand the following:

- What the roles of each standard technology are
- The quantity, quality, scope, and interoperability of standard schemas that exist in a given technology, and the degree to which such schemas have been implemented and deployed
- How different technologies can be used to complement each other

The right-most column in Table 1 identifies example content schemas for the indicated information modeling technologies (each row), and it estimates the capabilities and investments in the STEP series of standards. It shows how STEP provides on the order of 10,000 standardized concepts[4] for engineering and technical domains. For example, AP210 [29] specifies over 900 concepts for electronics/mechatronics including requirements, configuration management, simulation, connectivity, libraries, and 3D geometry. On the order of 1000 person-years of effort over the past 15+ years has been required to create this integrated family of STEP standards and gain consensus. It would be cost prohibitive and unnecessary to start from scratch and try to reinvent this capability using just XML directly.

Instead, there are several ways to interface to EXPRESS-based schemas using XML-based schemas, thereby leveraging strengths

[4]See ''STEP-on-a-Page'' at http://www.nist.gov/sc5/soap/ and note that it is now actually two pages, since the advent of STEP Modules, which define small, reusable information models that are employed in combinations to support business processes.

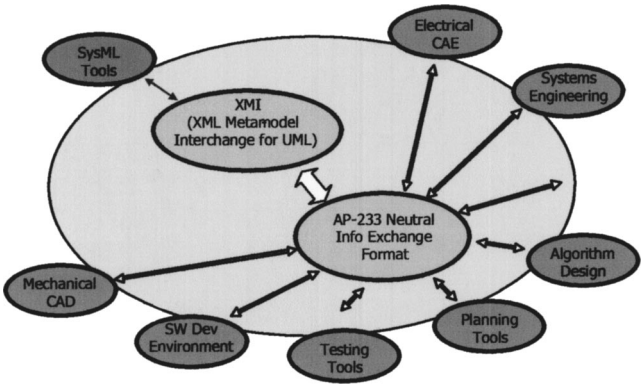

Fig. 4 Complementary usage of STEP, UML, and XML for Systems Engineering: Envisioned AP233-SysML relationship [32]

from both technologies. For example, a spectrum of XML-based interfaces can be created to get information into and out of more complete STEP-based rich product models. And now there are parts of STEP that are specifically designed to produce XML and UML models, as noted in Section 2.

A limitation of STEP is that there is no formal way to add user-defined extensions while maintaining interoperability with other STEP implementations. Although Part 21 allows user-defined instances in an exchange file, the STEP architecture lacks a well-defined AP-extension mechanism. Part 28, through its use of XML to represent STEP instances, enables STEP implementations to take advantage of XML's extensibility and flexibility. XML namespaces [30] make it possible for an XML document to contain both STEP and non-STEP data such that an application can process the STEP data while ignoring the non-STEP data. A remedy for the lack of AP extensibility in the STEP architecture is less obvious, but perhaps the STEP community can adapt approaches being used by some XML standards organizations such as the Open Applications Group [31]. STEP could also gain extensibility by using UML in place of EXPRESS as a description language, and making use of UML stereotyping and tagged values.

A survey of native XML content standards[5] reveals many areas that STEP does not deal with (e.g., legal applications), but relatively few engineering and technical topics that are the forte of STEP. This situation underscores these observations: (i) the XML syntax alone is not enough—agreements on content schemas are required, and (ii) technical domains are complex (hence, creating standardized information models is a challenge).

There appears to be an opportunity for the best of both worlds to work together and create more complete product and system models. As an example of this synergy, Fig. 4 envisions the complementary usage of STEP, UML, and XML for systems engineering. Tools that work with the UML-based SysML will be able to interoperate with AP233-based systems engineering models and a variety of domain-specific models facilitated by STEP (e.g., AP210 for electrical CAD/CAE and connections with mechanical CAD).

5 Usage Context: Standards-Based Product Life-cycle Management (PLM) Environments

A target usage context for combining and leveraging STEP/UML/XML-based models in product life-cycle management (PLM) environments is described next.

[5]For example, see the Cover Pages directory of XML applications at http://xml.coverpages.org/xmlApplications.html

5.1 Towards Fine-Grained Interoperability. Imagine this next-generation PLM environment (Fig. 5): Beginning in the product definition and conceptual design phases, customers or acquisitions staff define models that capture key aspects of the envisioned product (top horizontal bar). Over time these high-level models are iteratively refined and linked with detailed models that ultimately fully define the product. Throughout this process the PLM environment captures parametric relationships between all types of system and component models, including property-based requirements models (representations of requirements in terms of functional and physical parameters), functional and behavioral models, low-fidelity and high-fidelity analysis models, human simulation models, multiphysics models, and physical assembly models.

Figure 5 notionally illustrates such an environment, where the vertical bars depict specific domains that contain their own varieties of models. The second horizontal bar shows how system-level models employ fine-grained standards-based associativity to interconnect these domain models. These systems engineering models also link to models at the customer/acquisitions level.

All of these models and their relationships are captured in a comprehensive intelligent repository that cumulatively builds a collective product model that connects all the models developed throughout the life cycle. A tool-submodel view of a PLM framework supporting this approach is given in Fig. 7 and described later in this section. The models are further refined during the detailed design phases. They are used in design and systems review processes, providing a significant part of the knowledgebase on which computational tools for testing and system validation and verification are built.

Throughout this process various stakeholders must exchange information that affects models within the scope of their work. One can view these exchanges as parametric associativity relations among models. Standards-based interfaces are needed both between models in different domains and among diverse models within a domain.

A key capability for this type of advanced PLM environment is enhanced access to models and other knowledge about the products of interest. Today, however, such information is often not available in computer-sensible forms. Even when it is computer sensible, its usable life is often much less than that of the associated physical system (e.g., 5 yr versus 30+ yr) due to obsolescence of the native software tools that created the information. Moreover, gaps often exist in terms of information content coverage, content semantics, and associativity [34].

Figure 6 illustrates the current typical approach for CAD/X and product data management (PDM) systems. Level 1 PDM systems exist that are specialized for particular domains (e.g., mechanical PDM systems or electrical PDM systems). Level 3 enterprise PDM systems are also common that focus on major design releases and long-term archiving. Little exists, however, for Level 2 systems dealing with work-in-progress (WIP) multidisciplinary model development, which requires high-intensity interoperability. Even Level 1 and 3 systems typically manage monolithic native files only at the macrolevel, e.g., stating ''this finite element analysis (FEA) model is related to this part'' (without specifying the actual detailed relations). This situation creates knowledge gaps and relies on custom software and manual processes (''people-ware'') to manage microlevel associativity (e.g., taking parameter values and objects from several models, manually transforming them, and then using them in other models).

Research in standards-based PLM frameworks is underway to help overcome these barriers [35–37]. Such efforts are exploring how to achieve seamless interoperability among models implemented in different meta-modeling technologies. Thus, having STEP, XML, and UML work together as described in this paper is of keen interest in this context.

Figure 7 portrays an example standards-based PLM framework as a system of tools, models, and associated standards. The Venn

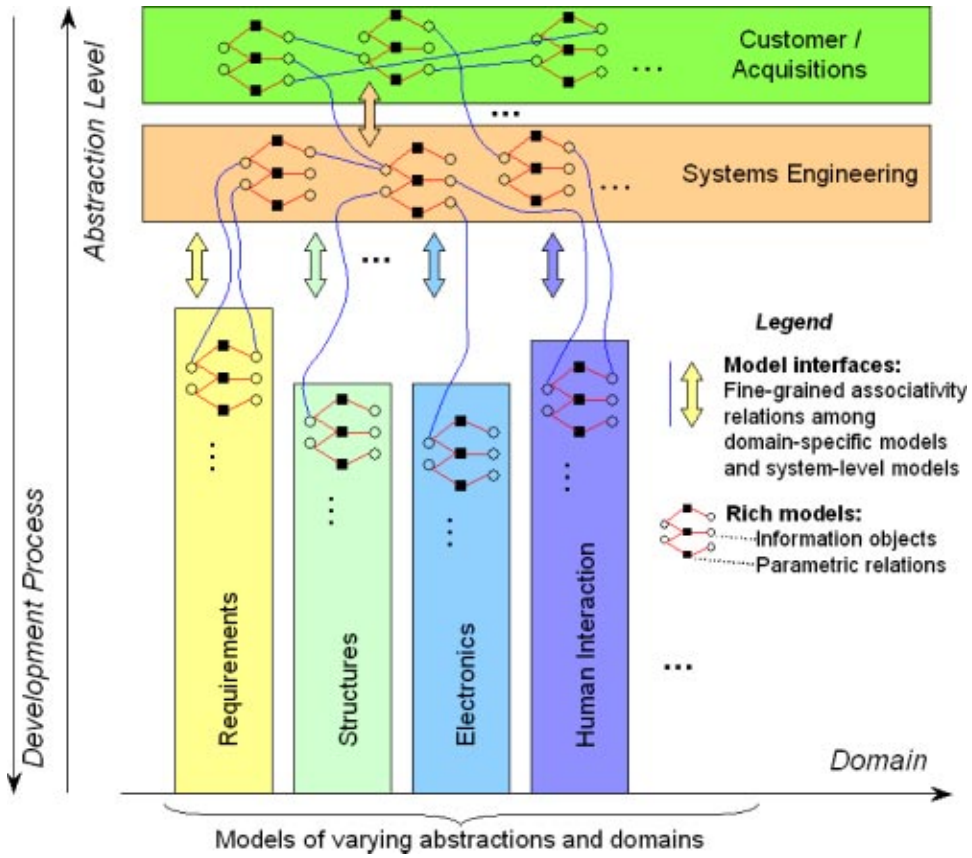

Fig. 5 Next-generation PLM framework with fine-grained interoperability (after [33])

diagram-like center notionally illustrates how a ''collective product model'' can be composed of diverse submodels. Each traditional tool in this figure typically focuses on viewing or editing a particular type of submodel within this overall product model (e.g., Mentor Graphics for circuit board electrical design and layout, and Pro/E for 3D enclosure and circuit board mechanical assembly). Portions of the collective product model may exist that no traditional tools address. This situation is where so-called gap-filling tools [25,34] are required to complete the product model.

Utilizing model formats that are compatible with commonly used CAX tools (e.g., traditional commercial off-the-shelf (COTS) tools for CAD and systems engineering, such as those illustrated in the figure) will help increase the sustainability of the PLM framework. However, these formats are often developed by different groups over various evolving time scales for diverse specific purposes. The end result is that a variety of modeling techniques (including STEP, XML, UML, and OWL [38]) will end up being employed over time to represent the different submodels in the Fig. 7 collective product model. These information models can be embodied in various forms ranging from international standards to internal specifications and customizations. We believe the multitechnology intermodeling method envisioned in this paper is one key to connecting these submodels and enabling fine-grained interoperability for a new generation of PLM environments. Note that such PLM frameworks are themselves systems whose evolving composition and configuration must be designed and managed (in conjunction with the products/systems that they are used to produce).

5.2 STEP and Web Services. Web services are a technology for providing access to applications via the Internet, either for human interaction or for automated system-to-system interaction. Some likely candidates for commercial web services useful to technical enterprises include on-line catalogs, directories of commercial services, data management services, interenterprise transaction services, and computing services.

Within an enterprise or a virtual enterprise (such as an original equipment manufacturer and their supply chain), web services could provide the next generation of trading partner business communications. These capabilities might very well include some types of interactions that are currently conducted either as paper transactions or by relatively primitive, human-mediated file-transfer interactions.

What role should STEP play in the world of web services? While STEP has traditionally been used primarily in file-transfer scenarios, it can also provide support for more agile and flexible web services applications.

Numerous inter- and intraenterprise communications involve product data that has been modeled in STEP, such as requests for quotes, requests for proposals, technical data package management, work orders, engineering change requests, and engineering change orders. These exchanges and others can be implemented as web services using STEP-defined standard data structures. STEP-based web services can enhance these processes by supporting whatever balance of machine automation/validation and human verification is optimal for such interactions.

One view of how STEP EXPRESS-based product models and web services can be architected is as follows [39]:

> The new [STEP] Part 28/XML standard will enable the definition of XML Schemas from the STEP Application Protocol mapping tables. This will make the definition XML data for STEP much more straightforward and easy to follow because these tables frequently restrict the large range of cases allowed in an EXPRESS model to one or two specific cases for a particular mapping.

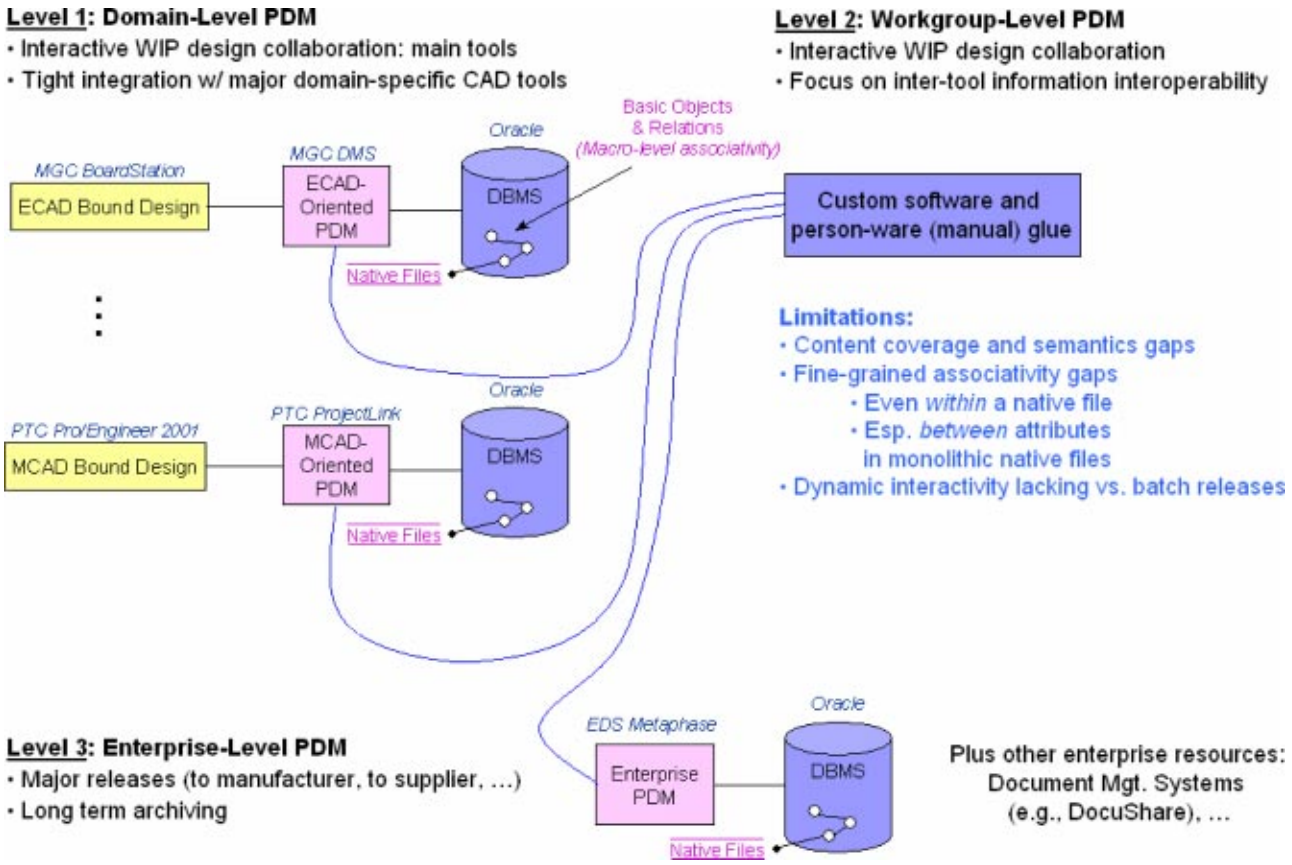

Fig. 6 Current typical levels of PDM system deployment and related limitations. [**This figure includes product names for example purposes only (i.e., to help clarify the concepts presented via specific instances.**)]

A three-level product model stack is emerging for representing STEP data on the web with EXPRESS defining the lowest level, XML Schema the second level and RDF the top level. Soon this architecture will enable distributed STEP databases where engineers use search and integration engines to identify compatible products and processes on the World Wide Web.

Further work is clearly needed in this area and several issues require more investigation. Some issues may be solvable near-term, while other aspects will require more time and resources. Efforts are already underway to add Part 28 functionality to the STEP Module Repository to produce ''equivalent'' XML schemas for portions of the Product Life Cycle Support (PLCS) family of STEP standards (http://www.oasis-open.org/committees/plcs) and to use the module repository to produce XML-based specifications based on these XML schemas. If this project is successful, then XML developers without any knowledge of EXPRESS will be able to build STEP implementations. Also, we would gain some practical examples of how the EXPRESS and XML worlds can coexist and benefit from each other's strengths.

Efforts are also ongoing to better integrate/combine STEP with the Internet [35,40], including ''Semantic Web'' technologies [41]. Specifically, a mapping from EXPRESS to the Web Ontology Language (OWL) [38] has been proposed. The *exff* software distribution (see Section 3) includes an XSLT implementation of this mapping.

6 Further Work: Bridging the Worlds of STEP, XML, UML, and Emerging Technologies

The following areas may merit further research.

6.1 Additional Modeling Languages Comparisons. Good analogies could help people understand the technologies and issues with various information and knowledge representation approaches. For example, older traditional programing languages can continue to thrive in the midst of newer languages due to specific advantageous features, as well as significant libraries of code and the volume of related legacy applications (e.g., FORTRAN is alive and well due to its large embedded base of useful numerical routines and its interfaces to other newer programing languages). Comparing modeling languages, such as EXPRESS, XML, and UML, could be analogous to comparing traditional languages like C++ and Java. Their modeling features could be compared as well as their usage and technology factors, such as available libraries, suitable application characteristics, developer base and popularity, and ease of use.

In other words, what are proper metrics and methods for comparing modeling languages such as EXPRESS, XML, and UML (and comparing their associated collections of standards), and recommending where best to use what technology? Earlier surveys, such as [42], could provide a methodological starting point. Comparisons would be helpful in terms of their expressive constructs, cost of modeling, available tools, developer base, etc. (i.e., various factors related to their ''total cost of ownership/usage,'' capabilities, and adoption within industry and government). It is likely that capabilities in one language are not supported as effectively in other languages. These situations need to be more completely identified and their work-arounds or recommended practices articulated.

Additionally, it would be useful to describe and compare the status of EXPRESS/XML/UML in these timeframes:

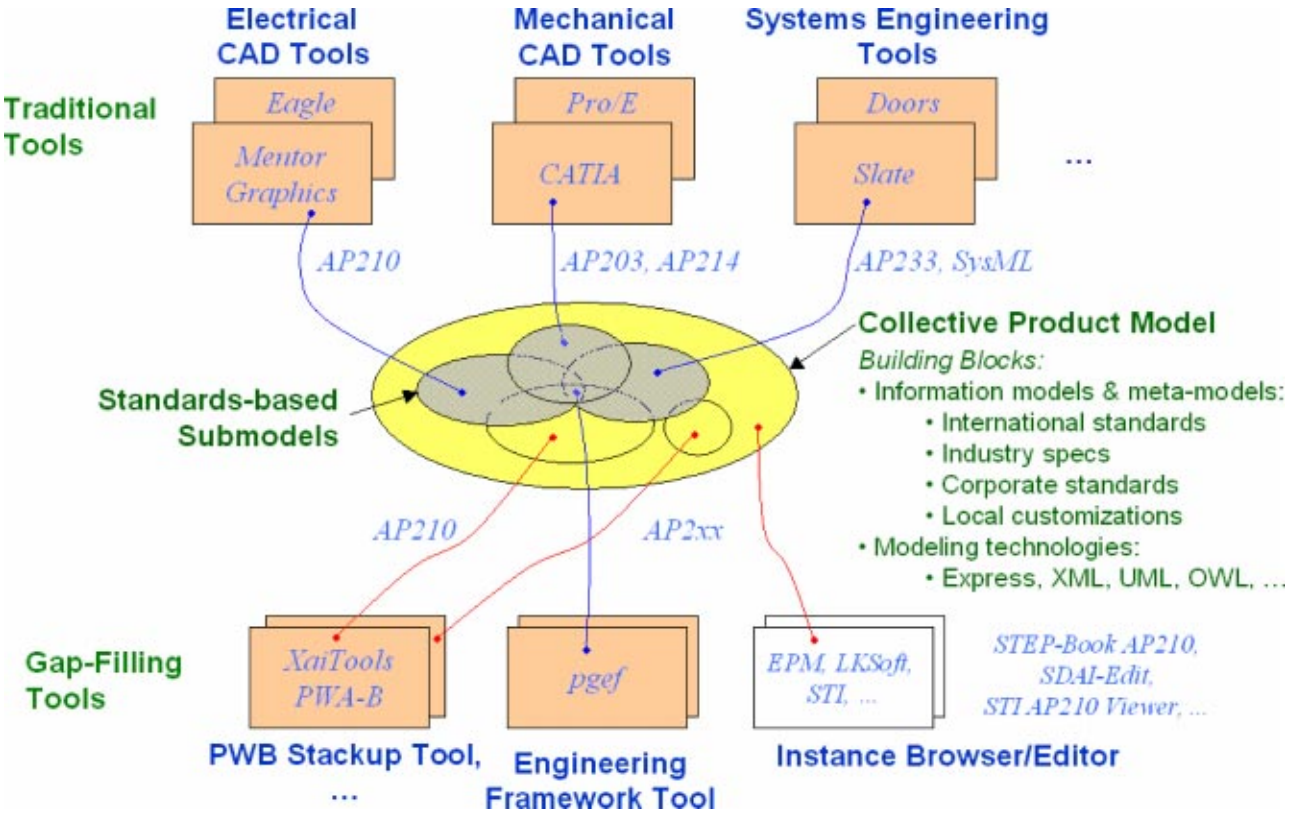

Fig. 7 Tool-submodel relationships in an example standards-based model-centric PLM framework [This figure includes product names for example purposes only (i.e., to help clarify the concepts presented via specific instances.)]

- current status (specifications, tools, interrelations between specifications, etc.)
- likely near-term status
- target status over the next few years and recommended actions to achieve those targets

6.2 Comparisons of Similar Content Standards. Further investigation of PLM XML (see Section 3) would be useful to better understand answers to questions like the following: How similar is it to related standards such as the STEP Product Data Management modules? How do they compare in terms of completeness, robustness, and capabilities?

It would be helpful to determine the degree to which EXPRESS-based standards, such as ISO 10303, are supersets of typically smaller scoped native XML-based standards. For example, we believe GenCAM is largely a subset of AP210 [29] in terms of its representation capabilities (i.e., there are design concepts and features that AP210 can represent which GenCAM cannot). Similarly, femML [28] may be a subset of AP209, and so on.

It would be useful to make such comparisons in a structured manner, give guidelines on how to do similar comparisons in other areas, and give recommended practices on how to handle such situations (e.g., show where and how such standards can then work together). Related questions are What are useful metrics and methodologies for determining the total cost of ownership/usage of such content-level standards? How can we judge their overall usability and value?

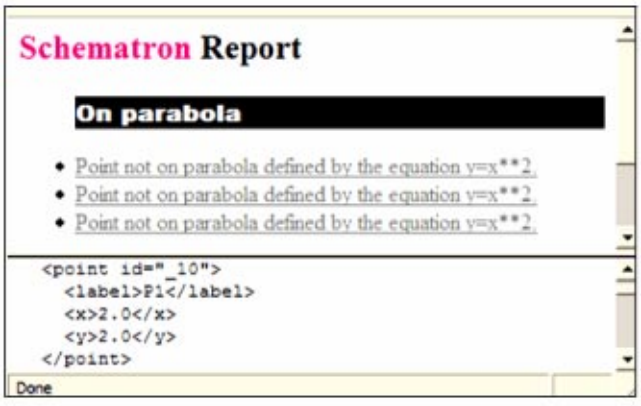

Fig. 8 Schematron diagnostics for the "Design 2L3P" XML lexical data

6.3 Further Investigation and Usage of XML Core Technologies. XML is already being used extensively in the STEP Module Repository (Section 3) to streamline the development and publication of modular STEP specifications. A description of how XML technology can help developers of standards, such as STEP, would be useful (including how it is doing so now in the case of the Module Repository).

Additionally, alternatives to the XML schema languages currently being used to represent EXPRESS should be investigated. RELAX NG [43], a powerful yet easy-to-use schema language for XML, could substitute for the W3C XML Schema language in STEP Part 28 mappings. Another alternative is Schematron [44], a language for making assertions about patterns in XML documents. EXPRESS *WHERE* constraints, which are currently outside the scope of Part 28—and are often impossible to specify as W3C XML Schema language definitions or RELAX NG patterns—could in some cases be represented as Schematron rules.

For example, the following Schematron schema represents the *WHERE* constraint from the *point_on_parabola* EXPRESS definition from Section 1.2:

```
⟨schema xmlns=‘‘http://www.ascc.net/xml/schematron’’
  xmlns:xsl=‘‘http://www.w3.org/1999/XSL/Transform’’⟩
   ⟨pattern name=‘‘On parabola’’⟩
    ⟨rule context=‘‘p28data/point’’⟩
     ⟨assert
      test=‘‘number(y)= number(x)* number(x)’’⟩
       Point not on parabola defined by the equation y=x**2.
    ⟨/assert⟩
   ⟨/rule⟩
  ⟨/pattern⟩
⟨/schema⟩
```

None of the three points in Fig. 2 satisfy this schema's rule. Therefore, when this Schematron schema is applied to the XML lexical representation of our ''Design 2L3P'' example, a Schematron validator generates an error message for each of the three points. Figure 8 shows a screen shot of Schematron validator-created browser output.

7 Summary

Today the aerospace, automotive, and shipbuilding industries are saving $150M/yr using STEP technology. STEP provides a large body of standardized, rigorously defined, high-fidelity technical concepts. The quality and scope of these STEP information models compares favorably with, and often exceeds, that of other data exchange standards.

However, traditional description and implementation methods for STEP (EXPRESS and Part 21) are not as popular or well known as web-oriented technologies, such as XML and UML. Thus, STEP, XML, and UML are complementary technologies, with STEP providing significant standardized content models, and XML and UML providing enhanced implementation methods. Combined, they are a powerful force for lowering the barriers to the widespread digital exchange and sharing of technical information. Emerging XML and UML-based STEP implementation technologies and current projects bridging these worlds show great promise to enable greater PLM interoperability.

Acknowledgments

The authors wish to thank the PDES, Inc. team, David Price, Allison Barnard Feeney, Matthew Aronoff, as well as the *2004 ASME Computers and Information in Engineering Conference* reviewers for their helpful comments on earlier drafts of this paper.

References

[1] W3C, 2004, Extensible Markup Language (XML) v1.0, http://www.w3.org/TR/REC-xml.
[2] ISO 10303-1:1994, Industrial Automation Systems and Integration—Product Data Representation and Exchange-Part 1: Description Methods: Overview and Fundamental Principles.
[3] Kemmerer, S., ed., 1999, STEP: The Grand Experience, National Institute of Standards and Technology, NIST SP 939, http://www.mel.nist.gov/msidlibrary/publications.html.
[4] Object Management Group, 2003, OMG Unified Modeling Language Specification v1.5, http://www.omg.org/technology/documents/formal/uml.htm.
[5] ISO/IEC JTC1/SC34 N0029, 1998, Document Description and Processing Languages–Final Text of Revised TC2 to ISO 8879:1986, http://www.y12.doe.gov/sgml/sc34/document/0029.htm.
[6] ISO 8879:1986, Information Processing—Text and Office Systems—Standard Generalized Markup Language (SGML).
[7] ISO/IEC PRF 19501, 2004, Information Technology—Open Distributed Processing—Unified Modeling Language (UML) v1.4.2.
[8] W3C, 2003, SOAP v1.2 Part 0: Primer, http://www.w3.org/TR/soap12-part0.
[9] Gallaher, M. P., O'Connor, A. C., Phelps, T., 2002, ''Economic Impact Assessment of International Standard for the Exchange of Product Model Data (STEP) in Transportation Equipment Industries,'' NIST Planning Report 02-5, http://www.mel.nist.gov/msid/sima/step_economic_impact.pdf.
[10] PDES, Inc., ''STEP Success Stories,'' http://pdesinc.aticorp.org/success_stories.html.
[11] Pratt, M. J., 2001, ''Introduction to ISO 10303–the STEP Standard for Product Data Exchange,'' J. Comput. Inf. Sci. Eng., **1**(1), pp. 102–103.
[12] Peak, R. S., 2002, ''Using Standards-Based Approaches for Electronics Product Design and Life Cycle Support. Extended Lecture,'' Shinshu University, Nagano City, Japan, http://eislab.gatech.edu/pubs/seminars-etc/2002-04-shinshu-peak.
[13] ISO 10303-11:1994, Industrial Automation Systems and Integration—Product Data Representation and Exchange—Part 11: Description Methods: The EXPRESS Language Reference Manual.
[14] ISO 10303-41:2000, Industrial Automation Systems and Integration—Product Data Representation and Exchange—Part 41: Integrated Generic Resource: Fundamentals of Product Description and Support.
[15] ISO 10303-21:2002, Industrial Automation Systems and Integration—Product Data Representation and Exchange—Part 21: Implementation Methods: Clear Text Encoding of the Exchange Structure.
[16] Carlson, D., 2001, *Modeling XML Applications with UML: Practical e-Business Applications*, Addison-Wesley, Reading, MA.
[17] ISO TC184/SC4/WG11 N223, ISO/WD 10303-28, 2004, Product Data Representation and Exchange: Implementation Methods: XML Schema Governed Representation of EXPRESS Schema Governed Data.
[18] Lubell, J., 2002, ''From Model to Markup: XML Representation of Product Data,'' XML Conference, Baltimore, MD, http://www.mel.nist.gov/msidlibrary/publications.html.
[19] W3C, 2001, XML Schema Part 0: Primer, http://www.w3.org/TR/xmlschema-0.
[20] W3C, 1999, XML Path Language, http://www.w3.org/TR/xpath.
[21] ISO TC184/SC4/WG11 N204, ISO/CD TS 10303-25, 2003, Product Data Representation and Exchange: Implementation Methods: EXPRESS to XMI Binding.
[22] Price, D., 2004, ''An Introduction to ISO STEP Part 25,'' http://www.exff.org.
[23] Object Management Group, 2002, OMG XML Metadata Interchange Specification v1.2, http://www.omg.org/technology/documents/formal/xmi.htm.
[24] W3C, 1999, XSL Transformations (XSLT) v1.0, http://www.w3.org/TR/xslt.
[25] Peak, R. S., Wilson, M. W., Kim, I., Udoyen, N., Bajaj, M., Mocko, G., Liutkus, G., Klein, L., Dickerson, M., 2002, ''Creating Gap-Filling Applications Using STEP Express, XML, and SVG-based Smart Figures—An Avionics Example,'' NASA-ESA Workshop on Aerospace Product Data Exchange, The Netherlands.
[26] Bartz, D., Staneker, D., Straßer, W., Cripe, B., Gaskins, T., Orton, K., Carter, M., Johannsen, A., and Trom, J., 2001, ''Jupiter: A Toolkit for Interactive Large Model Visualization,'' *Proc. IEEE Symp. Parallel and Large Data Visualization and Graphics*, San Diego, pp. 129–134.
[27] Engineering Framework Interest Group, emails from Stephen Waterbury, July 13, 2002, and David Leal, Nov. 26, 2002, http://eislab.gatech.edu/efwig.
[28] Michopoulos, J. G., 2002, ''Development of the Finite Element Modeling Markup Language,'' *Proc. ASME DETC & CIE Conf.*, Montreal, Canada, ASME, New York, DETC2002/CIE-34406.
[29] ISO 10303-210:2001, Industrial Automation Systems and Integration—Product Data Representation and Exchange—Part 210: Electronic Assembly, Interconnection, and Packaging Design, http://www.ap210.org.
[30] W3C, 1999, Namespaces in XML, http://www.w3.org/TR/REC-xml-names.
[31] Open Applications Group, User's Guide for Extending OAGIS 8.0, http://www.openapplications.org/downloads/memberdocuments.htm.
[32] SysML Partners, 2003, ''SysML Overview,'' Presentation to OMG Analysis and Design Task Force, http://www.sysml.org.
[33] Bajaj, M., Peak, R. S., Waterbury, S. C., 2003, ''The Constrained Object Knowledge Representation: Enhancing Interoperability and Visualization in Complex Systems,'' Tech. Report, http://eislab.gatech.edu.
[34] Peak, R. S., 2003, ''Characterizing Fine-Grained Associativity Gaps: A Preliminary Study of CAD-E Model Interoperability,'' *Proc. ASME DETC & CIE Conf.*, Chicago, ASME, New York, DETC2003/CIE-48232.
[35] Industrial Data on the Web (IDW) Working Group, 2004, http://step.jpl.nasa.gov/IDW.
[36] Waterbury, S. C., 2003, ''The Pan Galactic Engineering Framework,'' Aerospace Product Data Exchange (APDE) Workshop, NIST, Gaithersburg MD, http://step.nasa.gov.
[37] Peak, R. S., 2002, ''Standards-Based Engineering Frameworks as an Enabling Infrastructure,'' Invited Presentation, Next Generation Systems Engineering Panel, 36th Eng. & Tech. Mgt. Conf., Gov. Electronics & Info. Tech. Assoc. (GEIA), Snowbird, UT.
[38] W3C, 2004, OWL Web Ontology Language Overview, http://www.w3.org/TR/owl-features.
[39] Hardwick, M., 2003, ''XML and STEP,'' STEP Tools, Inc. Newsletter, http://www.steptools.com.

[40] STEPml, 2003, http://www.stepml.org.
[41] Price, D., 2003, ''A Brief Foray Into Semantic Web Technology and STEP,'' http://www.exff.org.
[42] Eastman, C. M., and Fereshetian, N., 1994, ''Information Models for Use in Product Design: A Comparison,'' Comput.-Aided Des., **26**(7), pp 551–572.
[43] ISO/IEC FDIS 19757-2:2002, Document Schema Definitions Languages (DSDL—Part 2: Regular-Grammar-based Validation—RELAX NG, http://relaxng.org.
[44] Jelliffe, R., 2002, The Schematron Assertion Language Specification v1.5, http://www.schematron.com.