PDML

Product Data Markup Language

A New Paradigm for Product Data Exchange and Integration

A White Paper

William C. Burkett
April 30, 1999

A B S T R A C T

Product Data Markup Language (PDML) is a set of XML vocabularies and a usage structure for deploying product data on the Internet and making it visible to DoD weapon system support personnel. PDML offers a new paradigm for product data exchange based on existing technology that facilitates the integration and interoperability of business processes across the DoD and contracting organizations, particularly among those using PDM systems for process control and product data management. The Internet provides a ubiquitous platform for connectivity; XML provides a web-friendly and well-understood syntax for the exchange of data; and STEP/ISO 10303 provides a methodology for satisfying and integrating the information needs of the diverse collection of data-usage communities that comprise the DoD weapon system support personnel.

PDML defines a set of Application Transaction Sets (ATS) that define the data requirements for communities defined by the users of particular legacy systems and standards. The Integration Schema is an encompassing generalization of the ATSs that provides an integrated view across the ATSs. Mapping specifications define the relationships between the specialized vocabularies in the ATSs and the generic vocabulary in the Integration Schema.

1 The Visibility of Product Data

The Challenge

Information system technology has long promised to integrate business processes by providing communication channels that seamlessly enable members of a team to exchange data and information across physical and temporal boundaries. The reality, however, has been very different. Data is stored and duplicated in so many different places that one often doesn't know "good" data from bad – even if one can *find* the data that one is interested in. And these aren't the biggest problems that integration efforts must overcome. A bigger obstacle is the fact that data is tightly bound to the applications that create/use the data, which means that moving data *between* applications requires a conversion or translations process – and anyone that has opened a WordPerfect document in Microsoft Word knows how well this translation typically works.

The support of weapon systems in the U.S. Department of Defense is a huge undertaking relying on data and information submitted/delivered from the countless manufacturers, suppliers, and support personal. The DoD envisions transitioning from a primarily government-operated product data repository environment to an environment in which the responsibility for maintaining current and accurate data describing defense material is partitioned and distributed among government entities and contractors; the product data repositories are operated and maintained by the party most responsible for the data at that point in the product's life cycle. In the current environment, for example, there is considerable duplication and redundancy between data repositories (e.g., PDM systems) managed by a weapon system's engineers and repositories managed by the personnel who buy spare parts for the weapon system. The duplication burdens personnel from both sites with non-valued-added work to synchronize the versions and indexes of the repositories so that changes initiated at either site are incorporated in all sites. Often, the weapon system prime contractor may maintain an additional data repository for internal use, thus exacerbating the synchronization problem.

Thus, the problem consists of (1) countless data repositories that are bound to and designed to support specific applications, and (2) the need to provide a integrated, accessible, consistent data in a form useful and meaningful to the personnel supporting a weapon system (i.e., in a form tailored/customized to the particular users of the data.) The problem is that meaningful product data is not *visible* to the personnel that need it.

The PDML Vision

The PDML vision is to enable any authorized user to access product data for any item from any site without manual intervention or coordination by other personnel. As PDML technology is put into place, duplicative government repository sites will be downscoped and eventually shut down. For example, a possible migration scenario to technology supporting this vision would be:

- Reference (master) copies of product data will reside and be maintained at the site of engineering authority for the product item. All other copies and variants of the data will be considered non-authoritative.
- In the product development phase of the life cycle, engineering authority will most often reside at the site of the development agent (usually the weapon system prime contractor, but sometimes a lower tier supplier or government activity for common items or Government Furnished Equipment (GFE)).
- Later in a product life cycle, the authority rests at the site of the Change Control Board (CCB) or the sustaining engineering activity. A sustaining engineering activity usually assumes full engineering authority from the CCB once the product item evolves to the late stages of entirely government based logistic support.

The vision is not a single, integrated virtual product data repository, but rather an environment in which users can:

- easily locate and obtain product data that is authoritative, timely, and accurate;
- view and understand the data without resorting to thesauri or indices, or dictionaries; and
- import the data into their own application systems with no more difficultly than simply downloading the file and opening it with the application.

If a user knows a product identification number, such as an NSN, part number, or serial number, that should be the only entry-point needed to obtain and use the product data for his job function.

The Current Solutions

Prior to 1999, realizing this vision was focused on the development of integration technologies that enabled interoperability of application systems. The integration disparate and heterogeneous information systems has been pursued since the recognition that two different systems are using some of the same information and someone said "hey – let's share data and save time". Approaches that have been pursued include:

- Point-to-point translators that convert data bound to one system into the data format of a target system;
- A shared database that is used by multiple applications;
- Product Data Exchange (PDE) standards that specify a neutral, application-independent data structure used to convey data between applications (and translators written to/from the PDE standard);
- Database federations in which each repository makes (some part of) its data visible to other databases/applications in the federation and accessible via an API;
- Product Data Management (PDM) and Enterprise Resource Planning (ERP) applications that "throw a net" over the applications within an enterprise and route, control, and constrain data that moves between individuals and/or applications.

The Internet and World Wide Web provide an even newer avenue to combat this problem. The ubiquity of the Internet breaks spatial and temporal barriers and provides the opportunity for users to access data anywhere, anytime. Furthermore, the rapid rise in popularity of the eXtensible Markup Language (XML) provides a platform independent and web-friendly data structuring syntax for the representation and exchange of data.

So – why don't these solutions provide the product data visibility needed for the system integration and weapon system support? There are three reasons:

- The complexity of application interoperability coupled with the huge variety of platforms and implementations make these solutions (1) point solutions, and (2) expensive solutions;
- The exchange of bits does not equate to the exchange of meaning (i.e., *communication*);
- The ever-changing requirements on system use makes point solutions brittle and short-lived.

The PDML Solution

PDML provides a technology solution that moves closer to the vision described above not by introducing new technology, but introducing a new data exchange paradigm that *leverages* existing integration technology and addresses the reasons existing technology doesn't meet the PDML vision. By drawing upon the best features of the current technology solutions, PDML provides a data exchange solution that

- doesn't require special applications,
- provides meaningful and usable data to users, and
- provides a means of exchanging data between application systems.

PDML defines a data usage architecture that starts with the assumption of data access and exchange over the Internet using existing Internet protocols and languages. This provides a general and ubiquitous platform that will be more stable and have a longer lifespan than point-specific solutions. In addition, the volume of development on the Internet will result in tools and solutions that are more widely applicable, more functional, and far less expensive than those applicable to integration point solutions. Tools for processing XML data, for example, are freely available on the Internet.

This keeps the PDML solution general, uniform, and widely applicable.

The elements of the architecture dealing with data "content" (i.e., the meaning of the data) rather than data "conveyance" (i.e., "moving bits around") provide a two-edge approach for the definition and exchange of "meaning" between people. The first "edge" is a usage-specific view of data that defines the meaning of data with respect to a specific "usage community". By using the terminology/vocabulary and structure of a particular community, the view provides unambiguous and easily understood representation of the data; the representation of this view in XML further provides a widely accepted and processable syntax for the exchange of data.

This keeps the PDML solution simple.

The definition of data (i.e., the vocabulary and structure) with respect to the viewpoint of a particular usage community is important because "meaning" is a feature of human cognition and language use. Data has no inherent "meaning; the "meaning" of data is ascribed to it by a community of users, application

programmers, and database designers. This usage community establishes a *context* for the data use. Current integration solutions described above work only when the users share a stable *context* for the usage of the data; by making the context of the data explicit, PDML enables unambiguous data exchange within a context.

The second "edge" of the PDML approach for the specification of meaning is also part of the PDML solution for adaption to changing requirements. It is integrated view of the data that encompasses all of the individual usage-specific views. Allied usage communities overlap with respect to the data needed within/by the community, though they may look at the data differently. The integrated view not only provides a mechanism for mapping the meaning of data between usage communities, but is also isolated from the changing usage requirements. New usage-specific views can be added or old views changed without affecting the integrated view.

This keeps the PDML solution integrated.

2 PDML Design Philosophy

The "philosophy" behind the design of PDML consists of two primary design principles:

- Leverage the best features of existing technology; and
- Maintain the data usage semantics.

Leverage Existing Technologies

The development of new technology is fraught with unavoidable obstacles. PDML seeks to avoid as many obstacles as possible by applying the best features of existing technologies to meeting the PDML vision. These technologies represent a range of system integration, PDE, and network technologies, including:

- STEP (ISO 10303) as a PDE technology;
- The Internet as an integration platform;
- XML as data structuring and encoding language;
- PDM systems as an enterprise data management tool.

Maintain Emphasis on Data Semantics

Information technologies seem to overwhelmingly focus on transport and processing of data with little, if any, regard for the principle value of data: conveying meaning to a user. Thus, an important design principle of PDML is the emphasis on the meaning of the data to the users of the data (i.e., the data semantics). This principle manifests itself in the usage-specific views of data defined in PDML and in the semantic mappings between the views through the mediating integrated view. This emphasis on semantic contrasts is most clearly illustrated when comparing PDML to HTML as a data presentation format or the general use of SGML as a document structure representation.

3 Application of Existing Technology to PDML

Leveraging existing technologies in the development of PDML requires the identification of the desirable features of the technology and the adaptation of those features to PDML requirements. Two technologies in particular – XML and STEP – are central to the structure and use of PDML.

The Web and XML

The use of the Internet, World Wide Web and XML is almost a given in new system integration projects, application interoperability, and data exchange. The reason for this is obvious: ubiquity and acceptance of the Internet and World Wide Web make it furthest-reachng and lowest cost integration platform in the world. The Internet is everywhere and is the most obvious solution to provide product data visibility to weapon system support personnel.

XML is the syntax of choice for exchange of web data, leveraging a middle ground between full-blown document structure and content management offered by SGML and the presentation mechanism provided by HTML. SGML originated in the field of text processing and the developers of the SGML language made an important realization that there is a distinction between the content of the document and the manner or style in which it is presented. HTML is a simple *application* of SGML that designed to present content on the World Wide Web, thus bringing SGML onto the Internet as a data structuring and exchange syntax.

XML is "… a simplified subset of SGML …optimized for the web environment, which implies data-processing-oriented (rather than publishing-oriented), and short life-span (in fact, usually dynamically generated) information". [5] XML is the ideal approach for deploying structured information on the web because it marries the presentation-free content management view of SGML (without many of the publication biases) to

the *de facto* language syntax of web established by HTML.

Data view versus document view

An important aspect of the PDML philosophy is the question of "data or document". As noted by Charles Goldfarb (one of the original developers of SGML and XML) "…many people have noticed that XML documents resemble traditional relational and object database data in many ways. Once you have a language for rigorously representing documents, those documents can be treated more like other forms of data." [5]

Since it originated in the SGML world of document publishing, XML is often viewed with a document-biased perspective inherited from SGML. This bias induces particular DTD design principles – chief among them a propensity for hierarchically-structured data – that is at odds with good data management practice. Individuals with database/data modelling experience view the XML syntax as another data structuring/encoding scheme and apply their data structure design principles to DTD design, resulting in DTD's noticeably different form the "document-biased" DTD's. Because Product Data Management is more about databases and data exchange, the design of PDML has adopted the "XML as data" philosophy in the design of DTDs.

STEP Technology

STEP (ISO 10303) is the principal product data exchange standard in the world. STEP pioneered several significant and revolutionary innovations in the use and exchange of data, notably the interpretation and use of generic data structures in different application domains. PDML is based upon the same basic structure of STEP, so a brief introduction to three main components of STEP as a PDE standard would be instructive.

The structure of STEP

The design of STEP attempts to reconcile two diametrically-opposed objectives:

- Define a set of data elements that are unambiguous;
- Define a set of data elements that are manageable (i.e., few in number), robust (i.e., stable over time), and flexible (i.e., can be used in many ways).

It meets the second of these objectives through the definition of a set of data elements known as the Integrated Resources. The Integrated Resources is a collection of schemas written in the EXPRESS information modelling language [6] that were designed to be applicable in all usage communities that deal with product data. As a result, the schemas are very generic and flexible – the schemas are independent of any particular industrial domain.

The generic design of the Integrated Resources, however, is directly at odds with the first objective: an unambiguous set of data elements would require not generic entities, but very specific entities – which, consequently, would result in a very, very large number of entities.

STEP solved this problem by introducing an innovative technique called "Interpretation" [2]. Interpretation, just as the normal English usage of the term implies, explains/specifies how a generic Integrated Resource construct, like `product`, is to be understood within a particular usage domain (i.e., by a particular usage community.)

Application of STEP Technology in PDML

There are two parts of STEP that have been leveraged in PDML

- The generic, integrated view (Integrated Resources);
- EXPRESS as a data specification language.

Within PDML, component corresponding to the domain-specific vocabulary for a particular usage community is called an Application Transaction Set (ATS). The PDML component corresponding to the generic vocabulary that is applicable across communities is called the Integration Schema; the design of the Integration Schema is based on the STEP Integrated Resources.

EXPRESS, as a data modelling language, is far richer in features than SGML/XML DTDs with respect to semantics and constraints and will therefore be used to specify the PDML schemas. The EXPRESS PDML schemas will be the *master* schemas with respect to the meaning and constraints of PDML data elements. Conversion algorithms have been defined that specify the transformation of the EXPRESS schema to an XML DTD and are described below.

UML (Unified Modeling Language [4]) is a more widely known and popular language than EXPRESS, and has richer, more expressive features than

Table 1 - Example of EXPRESS -> XML DTD Conversion

ENTITY product;
  name : STRING;
  description : STRING;
  id : STRING;
  frame_of_reference : SET [1:?]
OF product_context;
END_ENTITY;
	<!ELEMENT product (product.name,
	product.description, product.id,
	product.frame_of_reference, material?)>
	<!ATTLIST product
	  id ID #IMPLIED>
	
	<!ELEMENT product.name (#PCDATA)>
	<!ATTLIST product.name
	  datatype CDATA #FIXED "STRING">
	<!ELEMENT product.description (#PCDATA)>
	<!ATTLIST product.description
	  datatype CDATA #FIXED "STRING">
	<!ELEMENT product.id (#PCDATA)>
	<!ATTLIST product.id
	  datatype CDATA #FIXED "STRING">
	<!ELEMENT product.frame_of_reference
	(product_context_ref+)>
	<!ATTLIST product.frame_of_reference
	  aggregatetype CDATA #FIXED "SET">
	
	<!ELEMENT product_ref EMPTY>
	<!ATTLIST product_ref
	  refid IDREF #REQUIRED>

EXPRESS. However, UML, as an object-modelling language, has a different *purpose* than EXPRESS. Objects in UML "do" something - they have functionality and capabilities and lend themselves to the development of application systems. Entities in EXPRESS, on the other hand, don't "do" anything other than represent a real-world concept and don't lead to application system designs or functionality. It was felt that UML is *over-featured* with respect to the requirements of PDML.

There is a very important point to keep in mind with respect to the use of STEP in PDML:

> PDML IS AN ***APPLICATION*** OF STEP TECHNOLOGY; IS NOT A COMPETITOR OF OR REPLACEMENT FOR STEP OR THE STEP STANDARDIZATION EFFORTS.

As stated above, PDML is leveraging a combination the best features of existing technology – it is not intended to replace those technologies.

EXPRESS -> XML DTD conversion

Since there was a desire to leverage the semantic power of the EXPRESS language in the definition of the Application Transaction Sets and the Integration Schema, a conversion process was needed that would turn the EXPRESS schema into an XML DTD. We defined an algorithm that (as faithfully as possible) maintains the semantics and structure of the EXPRESS schema. This conversion is what is called an *early binding* approach because there is a close relationship between the EXPRESS schema and the resulting DTD. Table 1 illustrates the relationship between the EXPRESS declaration and the equivalent declaration in the XML DTD.

From the document-biased perspective of DTD development, the result of the conversion is somewhat unpleasing: it is bulky, verbose, and inflexible. However, from the data-biased perspective, the resulting DTD is robust and complete with respect to the product data semantics and constraints in the EXPRESS schema.

PDML has chosen this literal, data-centric approach to XML for a number of reasons:

- The DTD captures both the semantics and structure of the EXPRESS schema, thus explicitly encoding the semantics and structure in XML instances using the DTD;
- Verbosity and file size are not as important a concern as they once were due to faster channels, higher bandwidth, and common file compression tools and techniques;
- The "early binding" of the DTD results in XML that, to a large degree, encodes the schema into

the data (i.e., the schema can be largely reconstructed from the data itself); the explicitness of the schema in the data has two valuable side effects:

- The data is more resistant to degradation over time, thus making it a better archiving format;
- The data is more resistant to problems with compatibility between versions of the schema (e.g., "upward compatibility" problems) because the data fields are explicitly named and delimited.

The DTD's that comprise PDML adhere to the XML 1.0 specification [1] syntax. Therefore, any XML compliant tool kit will be able to process PDML.

4 Product Data Markup Language

PDML is not a single data specification, but rather a structure of related specifications and tools to deploy and use integrated product data on the World Wide Web. PDML Version 0.5 is composed of the following components:

- Seven Application Transaction Sets:
- The Integration Schema;
- Mapping specification between the Application Transaction Sets and the Integration Schema;
- The PDML Toolkit.

The relationship between these components is illustrated in Figure 1. As PDML grows, additional transaction sets will be added to the specification.

PDML defines views called Application Transaction Sets (ATS) that name and structure data in a way already familiar to a particular usage community – they establish a *vocabulary* for the community. The ATS is an XML DTD that specifies the elements and the structure needed to exchange data between current users of a particular application (i.e., users within the same context). Exchange of the data and a presentation style sheet would enable a user to receive and view the product data with *any* mainstream (and XML-savvy) web browser.

These view-specific vocabularies don't solve the integration problem, however.

The usage community (i.e., context) represented by the weapon system design and support personnel actually consists of many component usage communities. The Application Transaction Sets were designed specifically to support a few of the most important of these communities. The data required in these contexts overlaps with data in other contexts; furthermore, the data used in different contexts often has different names, or is stored in a different structure. Therefore, PDML provides an Integration Schema that serves as an intermediary between views, a neutral representation that is designed to service the information needs of *all* the contexts within weapon system support in a uniform and integrated way.

The relationship between the Application Transaction Sets and the Integration Schema is specified through *mappings*, or conversion rules, between the Application Transaction Set and the Integration Schema. The mappings, thus, provide an approach for taking view-specific data, converting it to a neutral *integrated* representation, and producing a new view-specific representation that presents the reconciled and integrated data to a user or application.

Application Transaction Sets

Jargon, lexicons, vocabularies, and languages all develop and grow within "meaning communities" – collections of individuals to whom certain words and phrases have a specific meaning and within which the meaning is reinforced and evolves through usage over time. The PDML Application Transaction Sets are exactly that: vocabularies meaningful within a well-defined community - except that the community is defined as the users of a particular legacy system like JEDMICS or standard like MIL-STD-2549.

Application Transaction Sets permit legacy data to be converted to/conveyed by XML in a simple form with a virtually one-to-one correspondence between the XML instances and the legacy data elements. This data can be exchanged with other systems that share the view using the Application Transaction Set directly, or it can be converted to the neutral format (i.e., the Integration Schema), integrated with data from other views, and then converted "back out" into another view.

There are seven Application Transaction Sets defined in PDML (version 0.5, April, 1999):

- Product Structure;
- Product Description Document;
- Technical Order –4;
- JEDMICS;
- MIL-STD-2549 DIP1;
- MIL-STD-2549 DIP3;
- MIL-STD-2549 DIP7.

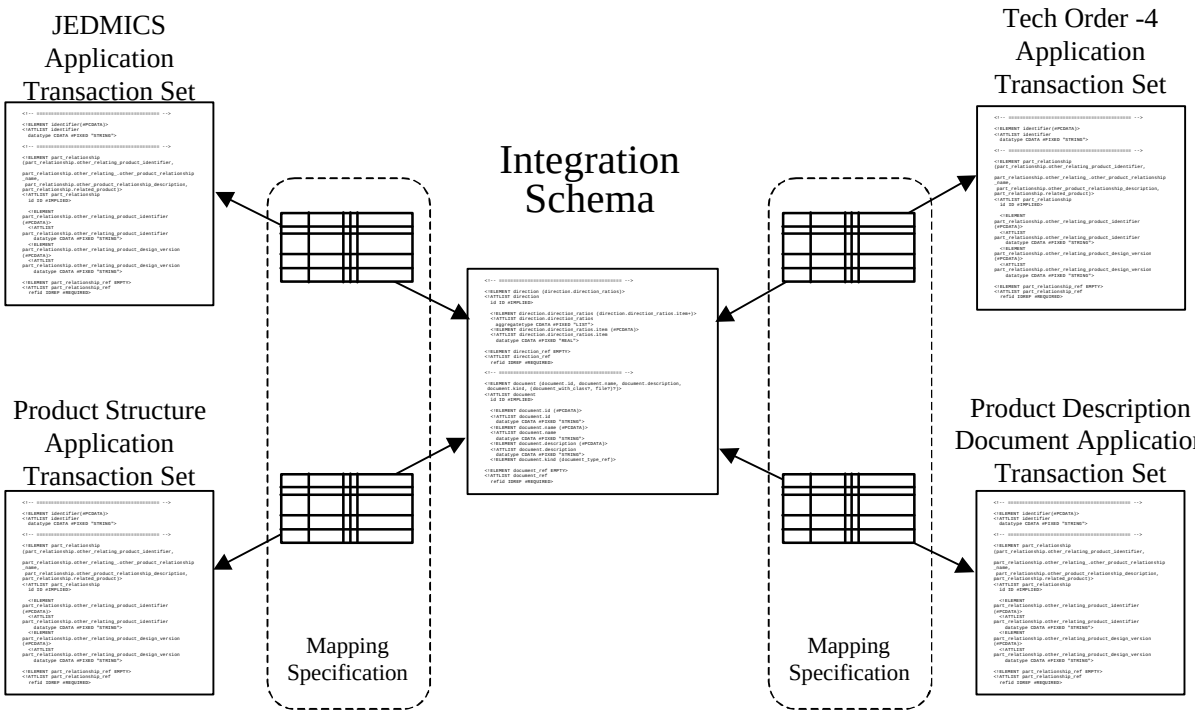


Figure 1 - Relationship of PDML Components

The scope of the Application Transaction Sets as a whole and the definition of individual transaction sets originated from a number of sources. The PDM Enablers effort within the Manufacturing Domain Task Force of OMG [7] defines an interface enabling the interoperabilty of PDM systems. Of this work, the areas of particular importance were the Product Structure and the Product Description Documents associated with the products.

Another source was the scenario to be used in the demonstration of PDML. The scenario is based on the competitive procurement of B52 Parts, which led to the other four transaction sets.

Product Structure and Product Description Document

The Product Structure and (associated) Product Description Document Application Transaction Sets originates in the requirements to support PDM interoperability. From the perspective of a single product, it calls out related design documentation, relationships to other parts, and usages of the product within higher assemblies/end items. The primary elements that comprise the Product Structure view are presented in Table 2.

The Product Description Document Application Transaction Set is a very simple view intended for document identification and management. Table 3 presents the primary elements in the Design Document view.

JEDMICS and Technical Order –4

JEDMICS is the primary system used by DoD for managing technical data. Thus, it provides a natural context for use and management of product data within the DoD. It is also the Application Transaction Set most closely tied to a particular legacy system.

Because it is such a well-defined and simple system view, there is only a single element defined in the JEDMICS transaction set, as shown in Table 4.

The Technical Order –4 is a standard form used by the Engineering Support Authority (ESA) at Tinker Air Force Base for maintenance and procurement of B52 parts. Thus, the TO –4 is another natural context for the definition and use of product data. The elements of the TO –4 transaction set are shown in Table 5.

Table 2 - Product Structure Elements

```
<!ELEMENT product_structure ((part_relationship |
      part_usage_relationship | product | product_relationship |
      related_design_document)*)>
<!ELEMENT identifier(#PCDATA)>
<!ELEMENT part_relationship (other_relating_product_identifier,
      other_relating_product_design_version,
      other_product_relationship_name,
      other_product_relationship_description, related_product)>
<!ELEMENT part_usage_relationship (product_structure_context_name,
      assembly_product_identifier, assembly_product_design_version,
      usage_description, interface_product_identifier,
      interface_product_design_version, interface_description,
      substitute_product_identifier, substitute_product_design_version,
      substitution_requirements, substitution_rank,
      alternate_product_identifier, alternate_product_design_version,
      alternate_description)>
<!ELEMENT product (product_identifier, product_name,
      product_description, product_design_version, design_document)>
<!ELEMENT product_relationship (related_product, relating_product)>
<!ELEMENT related_design_document (design_version_description,
      design_drawing_number, drawing_type, find_number,
      design_document_number, document_type, document_subject_name,
      document_subject_description)>
```

Table 3 – Product Description Document Elements

```
<!ELEMENT product_description_document ((associated_design_document |
      drawing_document)*)>
<!ELEMENT associated_design_document (design_document_number,
      document_title, document_release_date, document_author_name,
      author_organization_name, author_organization_address)>
 <!ELEMENT drawing_document (drawing_number, drawing_revision,
      designer_name, designer_organization_name,
      designer_organization_address, drawing_size, drawing_sheet,
      drawing_sheet_revision)>
```

MIL-STD-2549 DIP1, DIP3, and DIP7

MIL-STD-2549 [3] is new DoD standard for a Configuration Management Data Interface. This standard defines the data elements, definitions, and relationships necessary for the delivery of and access to electronic configuration management data.

"DIPs" are Data Information Packets that specify a predefined subset or usage of the overall 2549 standard for a particular purpose. The DIPs defined as Application Transaction Sets in PDML are:

- DIP1: Drawings, Specification, Standards, Software and Software Support Documents

  Data concerning documents defining the configuration design of hardware or software, or establishing standards for material, products, or processes.

- DIP3: Product/Asset Configuration

  Data concerning: part/material identification, serialization/lot tracking, actual current configuration of fielded products, update information concerning changes.

- DIP7: Engineering Parts List

  Data concerning the content of an engineering parts list.

All the definitions and conformance rules for these DIPs have also been adopted in PDML

Integration Schema

All of the Application Transaction Sets are views (or subsets) of product data necessary for weapon system support. They also overlap with respect to the data they include – a quick glance at the elements presented above shows values such as `part_number` and `drawing_number` are

Table 4 – JEDMICS Element

```
<!ELEMENT jedmics (drawing_number, cage_code, sheet, frame, revision,
      high_dwg, drawing_size, contl_code, revision_date, number_frames,
      file_type, file_size, security, rights, foreign, nuclear, safety,
      dist, physical_location, ODMS_date, in_date, last_access_date,
      last_change_date, last_change_user_id, file_type_source_flavor,
      file_type_dest_flavor, file_type_format, extension, num_acc_docs)>
```

Table 5 – Technical Order -4 Elements

```
<!ELEMENT tech_order_4 ((ipb | figure | major_assembly_breakdown |
      use_on_code)*)>
<!ELEMENT ipb (part_number, nomenclature, technical_order_number, CAGE,
      Name, Address, TCTO_number, Issue_date)>
<!ELEMENT figure (figure_index, section, sheet, assembly_index_number,
      part_number, units, use_on_code, parts_list_id)>
<!ELEMENT major_assembly_breakdown (index_number, section, description,
      figure_number)>
<!ELEMENT use_on_code (code, definition)>
```

common to two or more of the views. The relationships between the data defined in these views are established through the mapping to the Integration Schema.

The Integration Schema is a large DTD that is based on the STEP Integrated Resources. Like the Integrated Resources, it serves as an integrating mechanism – an integrated view – of the product data used within the communities/applications represented by the Application Transaction Sets. Unlike the STEP, data is not exchanged using this neutral view, but rather using the external views. When an integrated, cross-application view of product data is needed, data is extracted from the appropriate systems using their Application Transaction Sets, integrated via the Integration Schema, and then converted back to a specific Application Transaction Set view. The PDML Toolkit provides the mapping and conversion capabilities that insulate the users of the individual views from the complexity of the mapping process.

The relationship between each Application Transaction Set and the Integration Schema is specified in Mapping Specification, explained in the following section.

The Integration Schema is not intended to be directly used for product data exchange. Rather, it is more appropriate to consider it a temporary neutral form for integration and view translation.

Table 6 presents a small subset of the elements defined in the Integration Schema.

Mapping Specifications and View Integration

The Application Transaction Sets are application-specific *views* of product data and define a narrow context of data usage. The Integration Schema is an application *independent* view of product data and establishes a context of product data usage that encompasses the contexts of the application views. As a view, the Application Transaction Sets can be considered as a particular *interpretation* of the Integration Schema. This interpretation is formally specified by a Mapping Specification (just as "interpretations" are specified by Mapping Tables in STEP).

Mapping is more than conversion of between data structures. It encompasses the interpretation of data based on contextual values – a value from a single field doesn't *always* mean exactly the same thing (though it always *generally* means the same thing.) Based on contextual value that indicates the use, a field such as `document.id` could be drawing number, a tech order number, the designation of a standard or specification, or the identification of a digital file.

The PDML Toolkit "internalizes" and uses the Mapping Specification to drive the conversion of XML data to/from the Integration Schema format.

Table 6 – Integration Schema Elements

```
<!ELEMENT address (internal_location?, street_number?, street?,
       postal_box?, addtown?, region?, postal_code?, country?,
       facsimile_number?, telephone_number?, electronic_mail_address?,
       telex_number?, organizational_address?)>
<!ELEMENT calendar_date (month_component, day_component)>
<!ELEMENT date (year_component, calendar_date?)>
<!ELEMENT dated_effectivity (effectivity_start_date,
       effectivity_end_date)>
<!ELEMENT document (id, name, description, kind, (document_with_class?,
       file?)?)>
<!ELEMENT document_type (product_data_type)>
<!ELEMENT drawing_definition (drawing_number, drawing_type?)>
<!ELEMENT drawing_document (drawing_revision, document)>
<!ELEMENT drawing_revision (revision_identifier, drawing_identifier,
       intended_scale?, total_sheet_number)>
<!ELEMENT drawing_sheet_revision (revision_identifier)>
<!ELEMENT effectivity (effectivity.id, effectivity.name,
       effectivity.description, (dated_effectivity |
       serial_numbered_effectivity | lot_effectivity)?)>
<!ELEMENT file (size, file_format, source)>
<!ELEMENT group (group.id, group.name, group.description)>
<!ELEMENT local_time (hour_component, minute_component,
       second_component)>
<!ELEMENT organization (organization.identifier?, organization.name,
       organization.description, cage?)>
<!ELEMENT person (person.identifier, first_name?, middle_names?,
       last_name?, prefix_titles?, suffix_titles?)>
<!ELEMENT product (product.name, product.description, product.id,
       product.frame_of_reference, material?)>
<!ELEMENT product_composition (produrelating_product_definition,
       related_product_definitions, description,
       product_composition.name)>
<!ELEMENT product_context (discipline_type)>
<!ELEMENT product_definition (product_definition.id,
       product_definition.description, of_product, version_id,
       frame_of_reference,
       product_definition_with_associated_documents?)>
<!ELEMENT product_definition_context (life_cycle_stage)>
<!ELEMENT product_usage_relationship
       (specified_usage_occurrence_relationship?)>
<!ELEMENT security_classification (security_classification.name,
       purpose, security_level)>
<!ELEMENT security_classification_level
       (security_classification_level.name)>
```

PDML Toolkit

The PDML Application Transaction Sets can be used directly by application system for viewing and exchanging product data. It allows a user to pull XML files from local or remote locations, browse multiple XML documents in selected styles, integrate multiple XML documents of the same DTD, and publish XML files. It also provides API's allowing applications to exchange data through XML technology.

The integration performed in the Toolkit is driven by the mapping specifications. Using the mapping specification, the toolkit can convert data encoded according to an Application Transaction Set to the format specified by the Integration Schema. Data encoded according to other Application Transaction Sets can then also be converted and integrated into the neutral form. This integrated data set can then be converted "back out" to data conforming to an Application Transaction Set and view, exchanged, or saved.

The intent of PDML is for users to interact only with the Application Transaction Set views of the data; the

neutral model is hidden by the mapping specification and conversion software.

5 Why is PDML different?

PDML introduces to the web a new paradigm for data use, integration, and exchange. Mainstream pursuits of standardization of XML vocabularies and DTDs focus their efforts on the definition of the elements of their vocabulary – ***within their own particular usage community!*** There is nothing inherently wrong with these efforts, and it is extremely important that data usage communities identify and define the principle elements that they use to talk among themselves and exchange information. However, experience with data exchange standards such as STEP show that such solutions are not

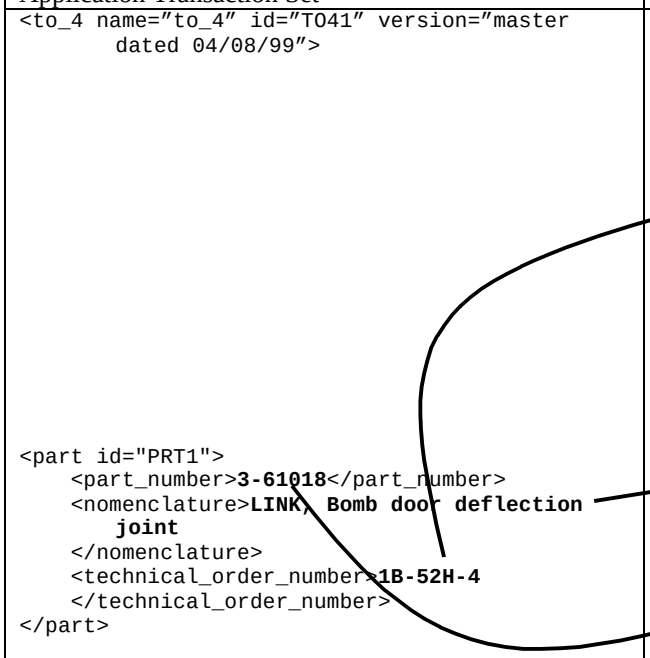
- portable;
- usable or useful (or not very usable/useful) outside the usage community that defined it;
- integrated with other vocabularies (by the single fact that they were developed within and for a particular community and without consideration of other vocabularies.

PDML provides a mechanism and paradigm in which data exchange is no longer bound to semantically "flat" schemas, but leverages the abstraction and context-sensitivity that people use unconsciously every day in human language use. In this way, individual usage communities can "have their cake and eat it, too" – they can use/exchange data in accordance with the own vocabulary and definitions, but still have an unambiguous path for exchanging data with users from other communities.

Unlike many XML DTD development efforts, PDML is not "thrown together" to meet a few short-sighted needs. Rather, PDML is based on database schema modelling principles and adopts a long-term view of data use and integration. Short-term solutions are brittle and quickly become obsolete. The flexible, stable, and reusable solution offered by PDML in the form of the Integration Schema offers a long-term solution can be applied (through interpretation) to any number of usage communities. The usage communities may come and go, but the Integration Schema will remain and provide an integrated point for application views both past and present.

The independent development of XML DTDs as vocabularies for individual and disjoint communities of users will result in a cacophony of incompatible – or worse: partially compatible – data specifications. If there is to be any hope that web resources can ever truly be integrated, some form of relationship or structure that provides a method for mapping between distinct communities is essential. PDML provides a valuable step in that direction.

Appendix A - Example XML in different views

XML data for the Tech Order –4 Application Transaction Set	XML data for same information in Integration Schema format
 <to_4 name="to_4" id="TO41" version="master dated 04/08/99"> <part id="PRT1"> <part_number>**3-61018**</part_number> <nomenclature>**LINK, Bomb door deflection joint** </nomenclature> <technical_order_number>**1B-52H-4** </technical_order_number> </part> <part id="PRT2"> <part_number>**8-6515-3**</part_number> <nomenclature>**SPAR INSTL, Center bomb door inboard (LH only) (see fig. 570)** </nomenclature> <technical_order_number> </technical_order_number> </part> <major_assembly_breakdown_figure id="MABF1"> <index_number>**17**</index_number> <section_number>**43**</section_number> <section_description> **Wheel and bomb bay area** </section_description> <figure_number>**570**</figure_number> <figure_description> **Center Bomb Door Spar Installations** </figure_description> </major_assembly_breakdown_figure> </to_4>	<pdml name="pdml" id="pdml1" version="master dated 04/20/99"> <document_type id="DT1"> <product_data_type>**technical order** </product_data_type> </document_type> <document id="D1"> <document.id>**1B-52H-4**</document.id> <document.name></document.name> <kind> <document_type_ref refid="DT1"/> </kind> </document> <product_context id="PC1"> <discipline_type>**B-52**</discipline_type> </product_context> <product id="P1"> <product.name>**LINK, Bomb door deflection joint**</product.name> <product.description></product.description> <product.id >**3-61018**</product.id> <product.frame_of_reference> <product_context_ref refid="PC1"/> </product.frame_of_reference> </product> <product id="P2"> <product.name>**SPAR INSTL, Center bomb door inboard (LH only) (see fig. 570)**</product.name> <product.description></product.description> <product.id >**8-6515-3**</product.id> <product.frame_of_reference> <product_context_ref refid="PC1"/> </product.frame_of_reference> </product> <product_definition id="PD1"> <product_definition.id>**43**</product_definition.id> <product_definition.description> **Wheel and bomb bay area** </product_definition.description> <product_definition.of_product> <product_ref refid="P2"/> </product_definition.of_product> <product_definition.version_id> </product_definition.version_id> <product_definition.frame_of_reference> <product_definition_context_ref refid="PDC1"/> </product_definition.frame_of_reference> <product_definition_with_associated_documents> <document_ref refid="D2"/> </product_definition_with_associated_documents> </product_definition> <document_type id="DT2"> <product_data_type>**-4 IPB figure** </product_data_type> </document_type> <document id="D2">

```
    <document.id>570</document.id>
    <document.name>
     Center Bomb Door Spar Installations
    </document.name>
    <kind>
        <document_type_ref refid="DT2"/>
    </kind>
</document>

<document_relationship id=”DR1”>
    <document_relationship.name>IPB figure
    </document_relationship.name>
    <document_relationship.description>
    </document_relationship.description>
    <document_relationship.relating_document>
      <document_ref refid="D1"/>
    </document_relationship.relating_document>
    <document_relationship.related_document>
      <document_ref refid="D2"/>
    </document_relationship.related_document>
</document_relationship>

<assignment_role id=”AR1”>
  <assignment_role.name>part illustration
  </assignment_role.name>
  <assignment_role.description>
  </assignment_role.description>
</assignment_role>

<product_document_reference id=”PDR1”>
    <pdr.documented_products>
      <product_ref refid="P1"/>
    </pdr.documented_products>
    <pdr.assigned_document>
      <document_ref refid="D2">
    </pdr.assigned_document>
    <pdr.role>
      <assignment_role_ref refid=”AR1”>
    </pdr.role>
    <product_document_reference.source>
    </pdr.source>
</product_document_reference>

<assignment_role id=”AR2”>
  <assignment_role.name>major assembly breakdown index
       number</assignment_role.name>
  <assignment_role.description>
  </assignment_role.description>
</assignment_role>

<item_location_identification_assignment id=”ILIA1”>
  <ilia.identified_product_documents>
    <product_document_reference_ref refid=”PDR1”/>
  </ilia.identified_product_documents>
  <ilia.assigned_id>17</ilia.assigned_id>
  <ilia.role>
    <assignment_role_ref refid="AR2"/>
  </ilia.role>
</item_location_identification_assignment>

</pdml>
```

Bibliography

1. Bray, T., Paoli, J., and Sperberg-McQueen, C.M. Extensible Markup Language (XML) 1.0 W3C Recommendation 10-February-1998. (1998) http://www.w3.org/. Date of page: 1998-02-10. Found c. 199-04-24.

2. Danner, W.F. Developing Application Protocols (APs) Using the Architecture and Methods of STEP (STandard for the Exchange of Product data). Fundamentals of the STEP Methodology. National Institute of Standards and Technology. NISTIR 5972. 1997.

3. Department of Defense. *Department of Defense Interface Standard Configuration Management Data Interface*. MIL-STD 2549, standard, 1997.

4. Fowler, M. and Scott, K., *UML Distilled Applying the Standard Object Modeling Language*. Addison-Wesley Object Technology Series, G. Booch, I. Jacobson, and J. Rumbaugh, ed. Addison-Wesley, Reading, Mass, 1997. ISBN 0-201-32563-2.

5. Goldfarb, C. and Prescod, P., *The XML Handbook*. Open Information Management, C. Goldfarb, ed. Prentice Hall, Upper Saddle River, NJ, 1998. ISBN 0-13-081152-1.

6. ISO. *Industrial automation systems and integration - Product data representation and exchange - Part 11: EXPRESS Language Reference Manual*. ISO 10303-11:1994, International standard, Geneva, 1994.

7. OMG. *PDM Enablers Joint Proposal to the OMG in Response to OMG Manufacturing Domain Task Force RFP 1*. mfg/98-02-02, Proposal standard, 1998

Acronyms

JEDMICS	Joint Engineering Data Management Information Control System
HTML	HyperText Markup Language
PDE	Product Data Exchange
PDM	Product Data Management
PDML	Product Data Markup Language
SGML	Standard Generalized Markup Language
STEP	STandard for the Exchange of Product model data
XML	eXtensible Markup Language