Teamcenter Engineering Data Exchange

Bronek Grzegorek
bronislaw.grzegorek@eds.com

EXPERTISE. ANSWERS. RESULTS.

EDS Corporate Overview

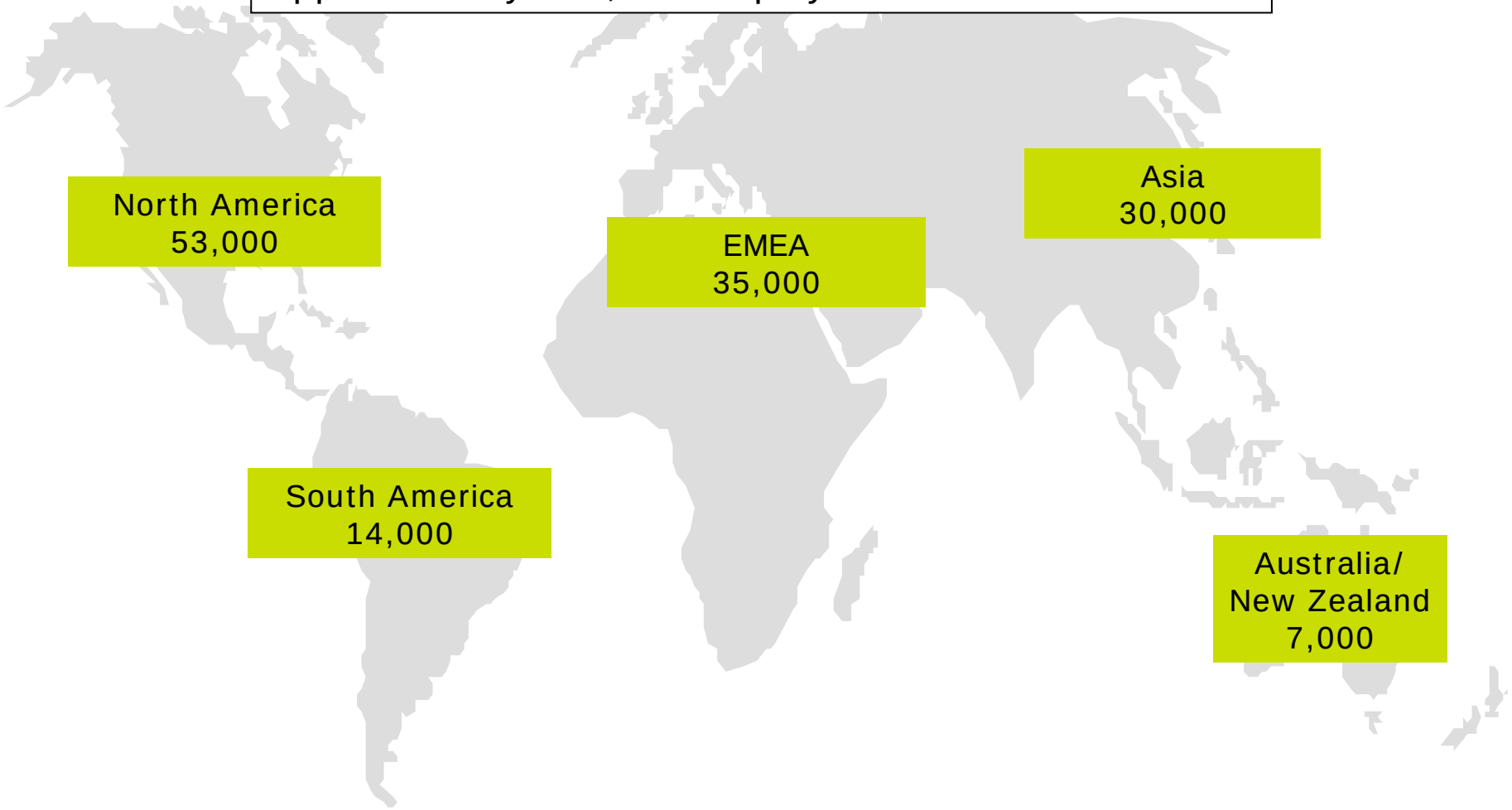
EDS Campus, Plano, Texas

- 139,000 employees
- 20,000 support Manufacturing clients
- 2007 revenues of $22.1 billion
- Ranked 115 on the 2008 Fortune 500
- 2 in the IT Services category
- 2000 clients globally, in 65 countries

- EDS provides a broad portfolio of business and technology solutions to help its clients worldwide improve their business performance
- EDS serves 75 of the top 100 Manufacturing clients globally
- 21 of the top 25 Global 1000 Automotive companies
- Siemens PLM Software is an EDS Alliance Partner
- EDS has a dedicated PLM services practice

EDS has a global footprint

Approximately 139,000 employees in over 60 countries



EDS Understands the Importance of Commitments

- **We manage globally:**
  - CRM services from 149 locations in 47 languages
  - 300,000 servers, over 100 data centers and just over 3 million client desktops
  - 2,500 firewalls and 3,000 systems to detect intrusions, threats and vulnerabilities
- **We process annually:**
  - Over 5 billion credit card transactions
  - 500 million travel reservation transactions
  - $95 billion in U.S. Medicaid benefits for more than 20 million recipients
- **We prevent monthly:**
  - 550 million junk e-mails from reaching our employees and clients
- **We serve clients in every major industry and geography, including:**
  - 75 of the top 100 manufacturing companies
  - 109 domestic and international air carriers - more than any other IT services provider
  - 200+ government clients in 24 countries
  - more than 200 clients in 21 countries worldwide through our BPO Administrative and Transaction business units.

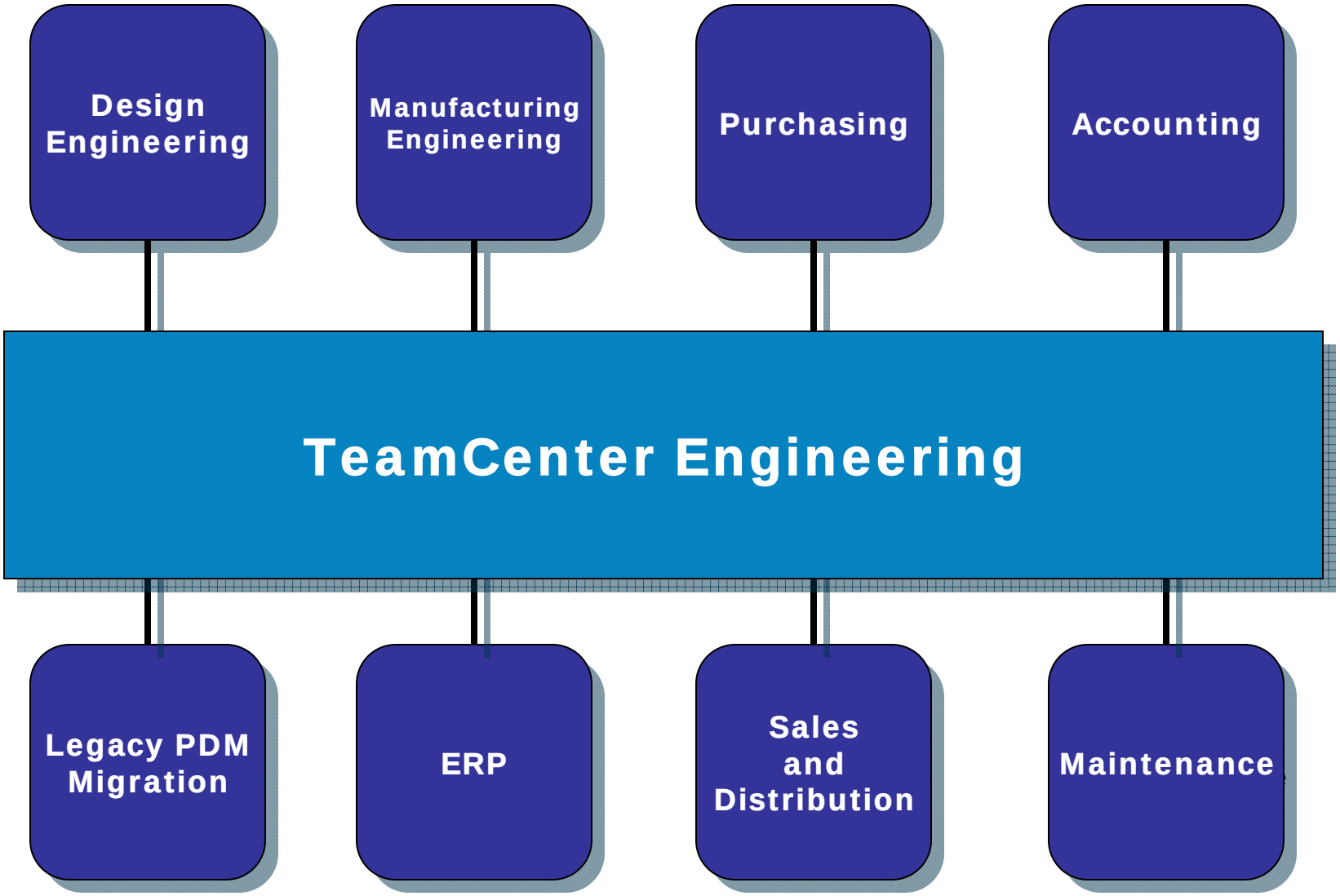
Agenda

- Needs for Data Exchange
- Existing Standards
- XLM technology overview
- PLMXML data schema overview
- MSXML Parser with Internet Explorer
- How to read and manage PLMXML files
- Demo of practical, working examples

Product Lifecycle Management

Is an **integrated**, information-driven approach **integrating** people, processes, business systems and technology to all aspects of a product's life including design, validation, manufacture, maintenance and disposal.

Needs for Data Interoperability



Collaborative Product Development

It strategy, work process and collection of software applications that facilitates different organizations to work together.

CPD Needs

- Perform version control of files.
- Release status of products and assemblies.
- Support change management .
- Work with dispersed organizations.
- Process different geometry and metadata formats of used by partners' systems.
- Provide secure exchange mechanism.

Asynchronous vs. synchronous CPD

An **asynchronously** a chief designer outlines the assembly configuration and the detailed component design tasks are assigned to individual designers to carry on separately.

A **synchronous** collaboration activity is conducted in a way such that a group of designers are dedicated to the same task actively.

Export for Change

OEM transfers files to a supplier that is allowed to those files, and/or may be depending of the authorized way of working.

OEM must protect these files against any modification, and is not allowed to create new revisions.

Export for Revise

OEM transfers files to a supplier that is not allowed to modify those files, but the supplier may only create new revisions.

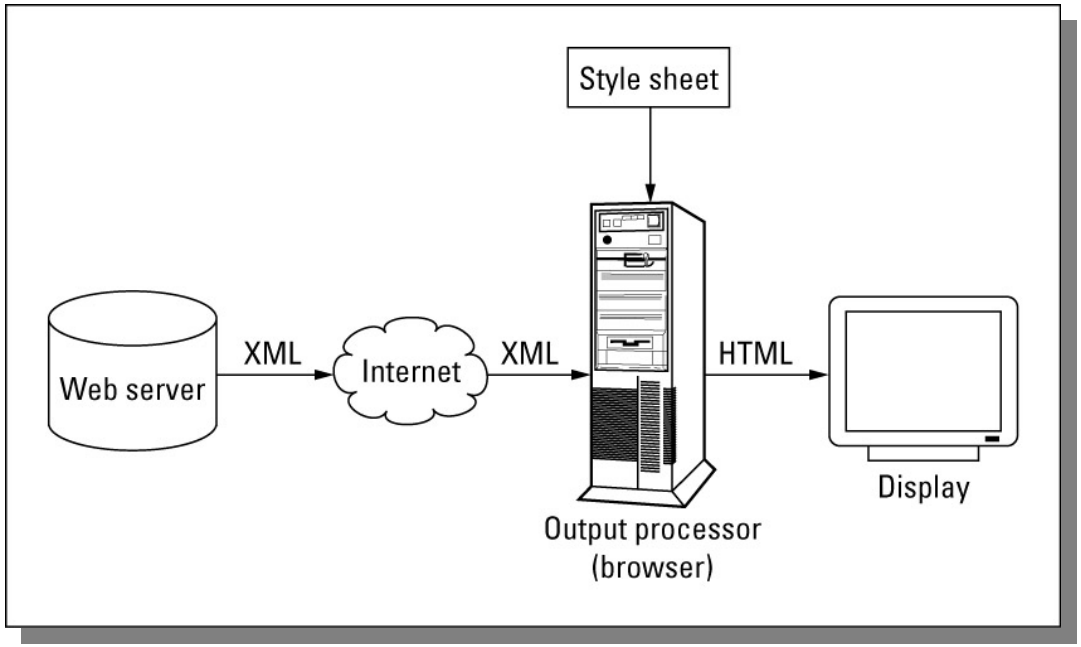
OEM must protect these files against change and revising.

Export for Reference

OEM transfers files to a supplier who is not allowed to modify the files, or to create new revisions;

OEM may make modifications or revisions, and has to inform the supplier, when these revisions have impact on the work assigned to that party.

XML (E**x**tensible **M**arkup **L**anguage)



- Technology for WWW applications
- Sponsored by World Wide Web Consortium
- Design to simplifies business-to-business transactions

XML Benefits

- Applications can easy access and extract information.
- Enables robust database contents exchange
- Data can be used in HTML pages.
- Documents must follow certain rules.
- Data is structure is self-describing and extendable.

Simple XML Example

Root element

Attribute

Empty tag

Attribute value

XML declaration

Closing tag

Untitled - Notepad

File Edit Format Help

```
<?xml version="1.0" standalone="yes"?>
<Recipes>
 <Recipe cook="The XML Gourmet">
  <Title>Bean Burrito</Title>
  <Category name="tex-mex"/>
  <Ingredients>
   <Item>1 can refried beans</Item>
   <Item>1 cup longhorn colby cheese, shredded</Item>
   <Item>1 small onion, finely chopped</Item>
   <Item>3 flour tortillas</Item>
  </Ingredients>
  <CookingInstructions>
   <Cinst>Empty can of refried beans into medium saucepan.</Cinst>
   <Cinst>Heat over medium-high heat until beans are smooth and bubbly.</Cinst>
   <Cinst>Warm tortillas in microwave for 30 seconds.</Cinst>
  </CookingInstructions>
  <ServingInstructions>
   <Sinst>Spread 1/3 of warm beans on each tortilla.</Sinst>
   <Sinst>Sprinkle with cheese and onions.</Sinst>
   <Sinst>Roll tortillas and serve.</Sinst>
  </ServingInstructions>
 </Recipe>
</Recipes>
```

Root element

Opening tag

XML Schema

XML Schema Examples - Microsoft Internet Explorer

File Edit View Favorites Tools Help

Back Search Favorites

Address http://msdn.microsoft.com/library/default.asp?url=/library/en-us/xmlsdk/html/278f540d-b2c1-4907-bb49-2a659b17451f.asp

msdn

MSDN Home | Developer Centers | Library | Downloads | How to Buy | Subscribers | Worldwide

Search for

MSDN Library Search

Advanced Search

sync toc

- Windows Versions
- XML
  - Soap Toolkit
  - MSXML
    - MSXML SDK
      - MSXML
      - XML Standards Reference
        - Document Map
          - Elements [XML Standards]
          - Prolog [XML Standards]
          - XML Declaration [XML Standards]
          - Processing Instructions [XML Standards]
          - DOCTYPE Declaration [XML Standards]
          - XML Comments [XML Standards]
          - Textual Content [XML Standards]
          - Character and Entity References [XML Standards]
          - CDATA Sections [XML Standards]
          - Attributes [XML Standards]
          - White Space [XML Standards]
        - DTD Statements
          - ELEMENT
          - ATTLIST (Attribute List)
          - ENTITY
          - NOTATION
        - XML Schemas (XSD) Reference
          - XML Schema Examples
          - XML Schema Elements
            - <xsd:all> Element
            - <xsd:annotation> Element
            - <xsd:any> Element

This topic contains the World Wide Web Consortium (W3C) purchase order examples. The first example is the schema for the purchase order. The schema example.

Example: Purchase Order Schema

The following example shows a schema, po.xsd, that defines a purchase order. This example shows the use of **element**, and **attribute** declara **complexType** definitions.

Xml

```
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema" targetNamespace="http://tempuri.org/po.xsd"
xmlns="http://tempuri.org/po.xsd" elementFormDefault="qualified">
 <xs:annotation>
  <xs:documentation xml:lang="en">
   Purchase order schema for Example.com.
   Copyright 2000 Example.com. All rights reserved.
  </xs:documentation>
 </xs:annotation>

 <xs:element name="purchaseOrder" type="PurchaseOrderType"/>

 <xs:element name="comment" type="xs:string"/>

 <xs:complexType name="PurchaseOrderType">
  <xs:sequence>
   <xs:element name="shipTo" type="USAddress"/>
   <xs:element name="billTo" type="USAddress"/>
   <xs:element ref="comment" minOccurs="0"/>
   <xs:element name="items"  type="Items"/>
  </xs:sequence>
  <xs:attribute name="orderDate" type="xs:date"/>
 </xs:complexType>

 <xs:complexType name="USAddress">
  <xs:sequence>
   <xs:element name="name"   type="xs:string"/>
   <xs:element name="street" type="xs:string"/>
   <xs:element name="city"   type="xs:string"/>
   <xs:element name="state"  type="xs:string"/>
   <xs:element name="zip"    type="xs:decimal"/>
  </xs:sequence>
  <xs:attribute name="country" type="xs:NMTOKEN"
     fixed="US"/>
```

XML Parsers

Validating Parsers

Xerces

The Apache XML Project is maintaining XML parsers in Java, C++, and Perl [free product from Apache.org; all Java, C++, and perl platforms]

IBM's XML Parser for Java

Also known as XML4J. Version 1 of IBM's XML Parser for Java was the highest rated Java XML parser in Java Report's February 1999 review of XML parsers. **Version 2** adds these exciting new features: Configurable, Modular Architecture; High Performance; Revalidation; and XCatalog Support. Support for XML 1.0, DOM 1.0 and SAX 1.0 is also included. **XML4J 3.0.1** is based on the Apache Xerces XML Parser Version 1.0.3. New features include experimental versions of DOM Level 2, SAX2 (beta 2), and parts of W3C Schema. *See also the numerous IBM alphaWorks freeware in our "Specialized XML Software" section.* [free product from IBM; all Java platforms]

JavaSoft's XML Parser

See **Java Project X** in the API section of this article for Sun's entry into the world of XML parsers. [free product from JavaSoft; all Java platforms]

Oracle XML Parser

Oracle released its XML Parser for Java, a standalone XML component that enables parsing of XML documents through either SAX or DOM interfaces using validating or non-validating modes. See also the Oracle XML site. [free product from Oracle; all Java platforms]

XMLBooster

XMLBooster generates XML parsers for COBOL, C, Java, etc. According to the company, XMLBooster is said to "achieve performance comparable with message-specific hand-written parsers by skipping the intermediate step where the message is turned into a generic DOM tree using a generic parser which must take the entire generality of XML into account and support every feature, no matter how obscure. The parsers generated by XMLBooster only recognize the XML features required to parse the message at hand, and produces directly a parser that initializes application-level data structures without going through any time-consuming intermediate representation. Tool features: (1) Generates parsers, which are between 5 and 45 times faster than generic parsers (2) Produce parsers in C, COBOL, Delphi and Java (3) Produces working data structures in the host language, rather than a dynamic and poorly typed generic tree (4) The XML message to parse can come from a file, a message, a socket, a data structure, etc. (5) Produce naturally validating parsers, far beyond the validation possibilities of DTDs." [commerical product for C, COBOL, Delphi, Java]

SXP, the Silfide XML Parser

The Silfide XML Parser (SXP) is a parser and a complete XML API in Java. It is part of XSilfide, a client/server based environment. XSilfide includes SIL, the Silfide Interface Language, among other things. "The SIL DTD is organized using modules, gathering (1) the encoding of the user workspace (2) the encoding of the user informations (3) the extended query language and (4) the encoding of the queries result set." [free product from Silfide; all Java platforms]

MSXML

Microsoft's XML parser in Java is included in IE4. The version presently available predates the final XML 1.0 spec by one month. "The parser checks for well-formed documents and optionally permits checking of the documents' validity. Once parsed, the XML document is exposed as a tree through a simple set of Java methods, which [Microsoft is] working with the World Wide Web Consortium (W3C) to standardize. These methods support reading and/or writing XML structures..." See sample parsing of an XML file using JScript. (Microsoft also includes an XML parser in C++ in IE4 which is "a high-performance, non-validating parser, [that] supports most of the W3C XML specification".) [free product from Microsoft; all Java platforms; all IE4 platforms]

DXP

Note: DXP is now a team development effort with Microsoft and will be included in Internet Explorer 5.0. "The DataChannel XML Parser (DXP) is a validating XML parser written in Java. DXP is specifically aimed at providing a utility for server-side applications that need to integrate XML based software. DXP provides the highly sophisticated error-checking mechanisms required for XML-based data interchange. DXP has not been ar DXP is based on NXP (Norbert's XML Parser), one of the earliest XML parsers. See also our DataChannel entry in the "Specialized XML Softwar and Microsoft; all Java platforms]

Larval

Larval is Tim Bray's validating XML processor built on the same code base as Lark (below). "Larval is a full validating XML processor; it reports vi

Near & Far Designer

According to Microstar, "Near & Far Designer is the ideal tool for those who are new to structured information as well as those who are already achieving the benefits of structured information. DTDs can be created and modified graphically without prior knowledge of XML/SGML language syntax. With the intuitive tree representation a DTD can be created from scratch or imported, reworked and exported as a revised DTD. Structures can be explored to any level of detail. The drag and drop interface makes working with DTDs easy."Read review from XMLXperts, Ltd of converting SGML to XML using Near & Far Designer . [commercial product from Microstar Software Ltd.; Windows only]

XML::Parser

This perl-based XML parser is from Larry Wall, the creator of perl. Some of the parsing code is based on James Clark's expat (below). At this time, there is no documentation or description; the link is for downloading. [freeware from Larry Wall; Perl]

xmlproc

"xmlproc is an XML parser written in Python. It is a fairly complete validating parser, but does not do everything required of a validating parser, or even a well-formedness parser. The average user should not run into any omissions, though. Later releases will be more complete." freeware by Lars Marius Garshol; Python]

TclXML

This XML parsing package requires Tcl 8.0b1 or a later version. Last updated 19th June 1997. It is possible that something newer will come from Zveno. [freeware; Tcl]

Non-validating Parsers

Lark

Lark is a non-validating Java XML processor by Tim Bray, one of the authors of the W3C XML spec. It implements all of the XML 1.0 Recommendation and reports violations of well-formedness. [freeware by Tim Bray (Textuality); all Java platforms; see also Larval above]

XP

James Clark's XML Parser in Java, complete with javadoc documentation. "XP is an XML 1.0 parser written in Java. It is fully conforming: it detects all non well-formed documents. It is currently not a validating XML processor. However it can parse all external entities: external DTD subsets, external parameter entities and external general entities. " XP is a high performance parser intended for use with Java applications, rather than applets. It includes a SAX driver implementation. (In addition to expat [below] and XP, James Clark also has developed SP, a free, object-oriented toolkit for SGML parsing and entity management; SP can parse XML and can convert SGML to XML.) [freeware from James Clark; all Java application platforms]

Ælfred

According to Microstar, Ælfred is "a small, fast, DTD-aware Java-based XML parser, especially suitable for use in Java applets. We've designed Ælfred for Java programmers who want to add XML support to their applets and applications without doubling their size: Ælfred consists of only two core class files, with a total size of about 26K, and requires very little memory to run. There is also a complete SAX (Simple API for XML) driver available in this distribution for interoperability." Note that Microstar also has a commercial XML authoring tool called Near and Far Designer (above). [freeware from Microstar Software Ltd.; all Java applet platforms]

HEX

HEX is the HTML Enabled XML Parser. It is "simple, 100% Java, non-validating XML parser with some hooks for more-or-less correct parsing of most HTML pages. It doesn't understand either SGML or XML DTD's but the parser API allows the application to control its operation in ways that facilitate HTML parsing. " HEX includes an implementation of SAX. HEX also implements the Java binding for the DOM core level one as per the March 1998 Working Draft. [freeware by Anders Kristen, HP Labs; all Java platforms]

expat

XML Parser Toolkit is James Clark's library for XML parsing in C. Expat (formerly called xmltok) is being used to add support for XML to Netscape Navigator 5 and Perl. Expat aims to be a fully conforming XML 1.0 parser and is written in C.

HXA (Hubrick's XML Analyzer)

Hubick's XML Analyzer "is a pure Java tool built upon a low level XML parser (HXP) which breaks an XML file down into it's constituent productions for analysis. HXA allows one to examine the production hierarchy for any character in an XML document or document fragment. For easy reference HXA also provides links from each production in the analysis to its corresponding section in the XML specification." [freeware for all Java platforms; may require Microsoft Internet Explorer]

LT XML

"LT XML is an integrated set of XML tools and a developers' tool-kit, including a C-based API...The LT XML tool-kit includes stand-alone tools for a wide range of processing of well-formed XML documents, including searching and extracting, down-translation (e.g. report generation, formatting), tokenising and sorting. Sequences of tool applications can be pipelined together to achieve complex results.... It also includes a powerful, yet simple, querying language, which allows the user to quickly and easily select those parts of an XML document which are of interest." The parser produces either a textual view or a tree view of an XML document. [freeware from the Language Technologies Group; C language; Unix and Win32 platforms]

xmlib

Internet

http://wdvl.com/Software/XML/parsers.html

XML Schema

XPath (XML Path Language) is a language for selecting nodes from an **XML** document

Examples:

PLMXML/ProductRevision

PLMXML/ProductRevision/AssociatedDataSet

PLMXML/Product[@id='id1245']

PLMXML/Form[@role='IMAN_MASTER_FORM']/UserData/UserValue

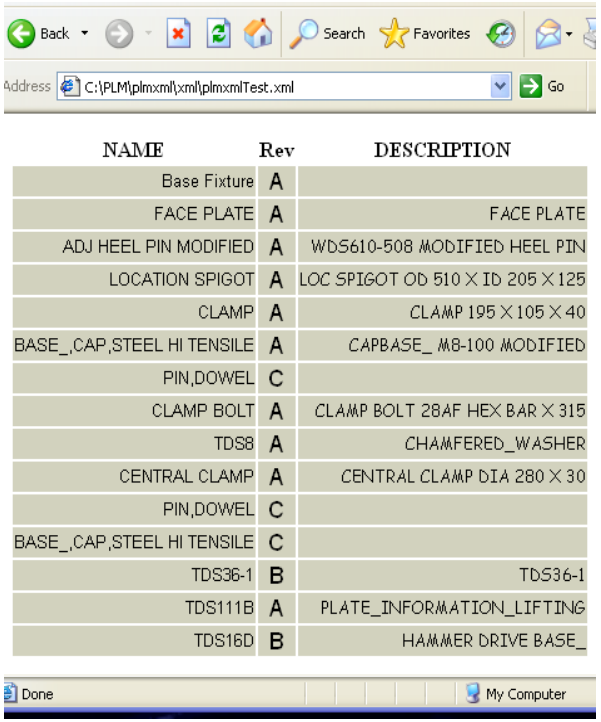
PLMXML/DataSet[@type='DirectMode']

PLMXML/ExternalFile[@id='id40']/locationRef

XSL Transformation

+

=

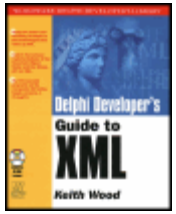


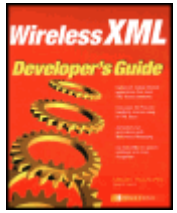
NAME	Rev	DESCRIPTION
Base Fixture	A	
FACE PLATE	A	FACE PLATE
ADJ HEEL PIN MODIFIED	A	WDS610-508 MODIFIED HEEL PIN
LOCATION SPIGOT	A	LOC SPIGOT OD 510 X ID 205 X 125
CLAMP	A	CLAMP 195 X 105 X 40
BASE_,CAP,STEEL HI TENSILE	A	CAPBASE_ M8-100 MODIFIED
PIN,DOWEL	C	
CLAMP BOLT	A	CLAMP BOLT 28AF HEX BAR X 315
TDS8	A	CHAMFERED_WASHER
CENTRAL CLAMP	A	CENTRAL CLAMP DIA 280 X 30
PIN,DOWEL	C	
BASE_,CAP,STEEL HI TENSILE	C	
TDS36-1	B	TDS36-1
TDS111B	A	PLATE_INFORMATION_LIFTING
TDS16D	B	HAMMER DRIVE BASE_

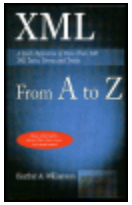
An XSL stylesheet templates that define how the elements in an XML document should be transformed

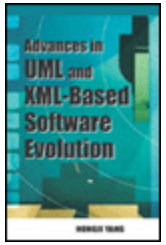
XML Resources

Amazon.com over 980 books about XML !

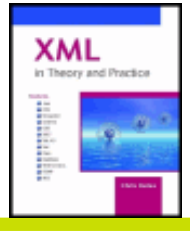




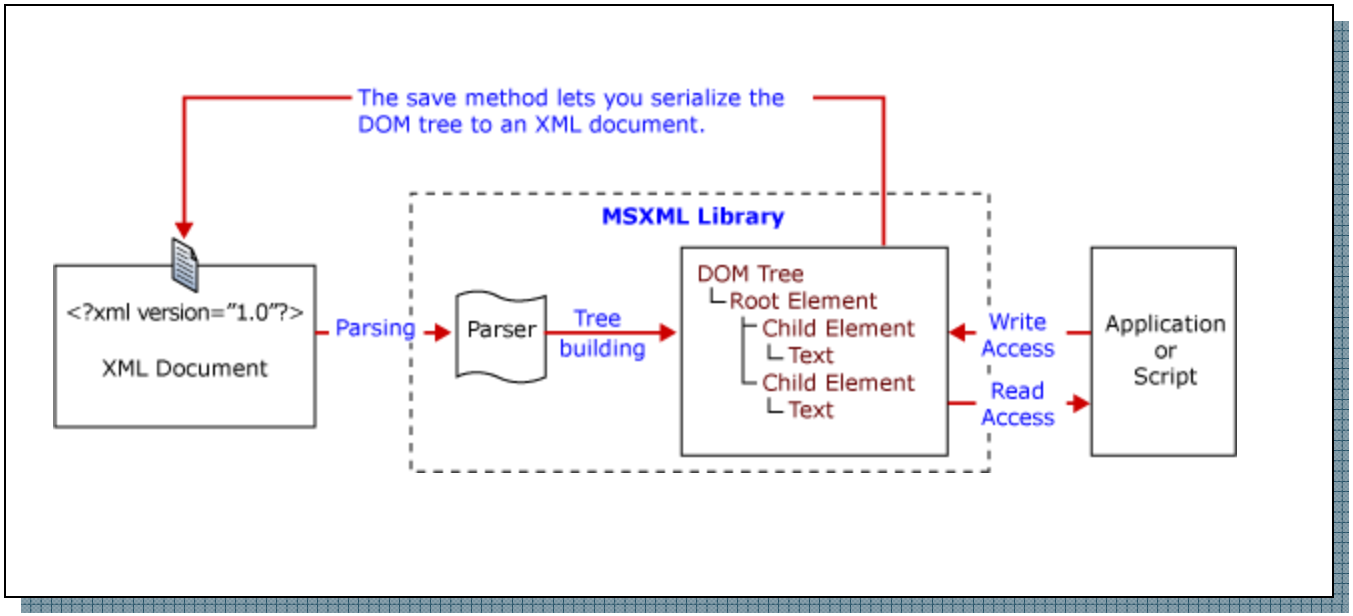




Reference point XML



DOM and XML



The Document Object Model (DOM) as implemented in MSXML provides a programmatic representation of XML documents.

PLMXML

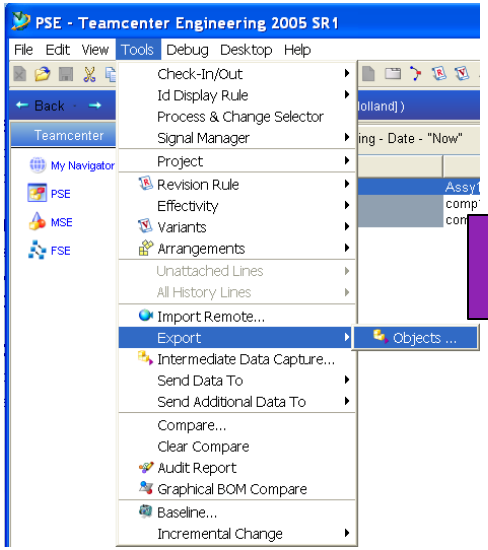
- Similar to STEP product data definition (schema)
- It is becoming industry “de facto” standard
- Implements XML format for data exchange interoperability

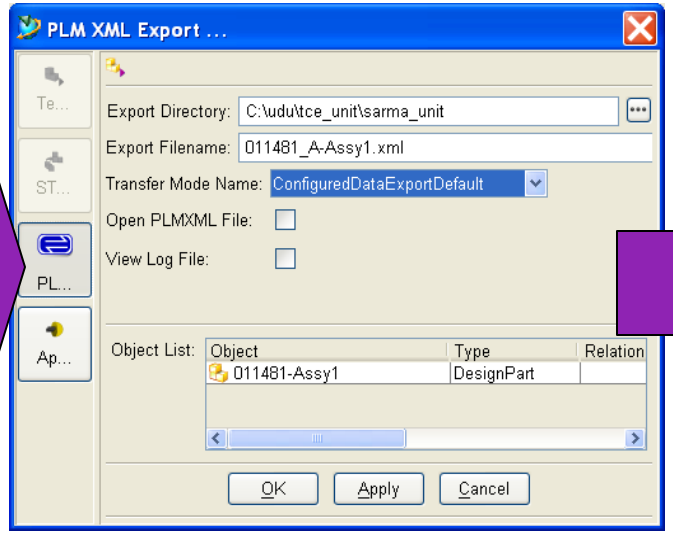
PLMXML Schema

```
<?xml version="1.0" standalone="no"?>
<PLMXML xmlns="http://www.plmxml.org/Schemas/PLMXMLSchema" schema
 xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
time="14:49:12" language="en" date="2001-12-19">
  <ProductDef id="id1" name="assembly" revision="1">
    <InstanceGraph id="id2" rootInstanceRef="id3">
      <Description>
        Product hierachy for sample car, distributed over two fil
      </Description>
      <Instance id="id3" partRef="#id4"/>
      <Instance id="id6" partRef="#id7"/>
      <Instance id="id10" partRef="car_axle_assembly.plmxml#id11"
      <Instance id="id24" partRef="car_axle_assembly.plmxml#id11"
      <Part id="id4" type="assembly" instanceRefs="id6 id10 id24"
      </Part>
      <Part id="id7" type="solid" representationRefs="id9">
      </Part>
      <Transform id="id23">1 0 0 0 0 1 0 0 0 0 1 0 0 1 0 1</Trans
      <Transform id="id25">1 0 0 0 0 1 0 0 0 0 1 0 0 -1 0 1</Tran
    </InstanceGraph>
    <Representation id="id9" name="car_id7" format="XT" revision=
      <Description>
        Chassis of Car assembly
      </Description>
    </Representation>
    <ProductView id="id26" primaryOccurrenceRef="id27"
```

PLM XML is defined by a set of XML schemas that are compliant with the W3C XML schema recommendation.

Exporting PLMXML Interactively



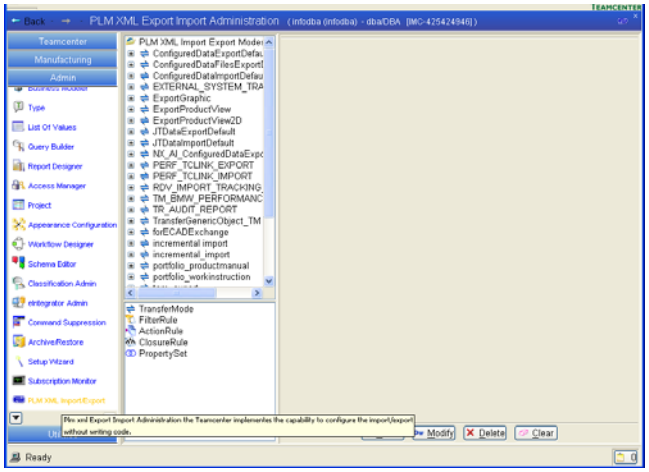


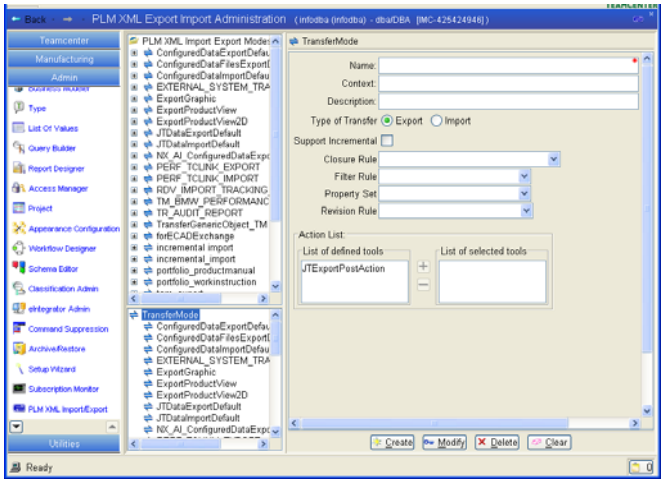
Entry Field	Description
Export Directory	Select the directory into which the PLM XML file along with associated files should be written to.
Export Filename	Specify the name of PLM XML file.
Transfer Mode Name	The name of Transfer Mode to be used for this translation. Based on the scope defined in the Transfer Mode, the contents of the xml file are appropriately generated.
Open PLM XML File	Allows you to view the XML file generated in an editor after completion of export, if the check box is on.
View Log file	Allows you to view the log file if this check box is on. It allows checking for any errors due to translation.

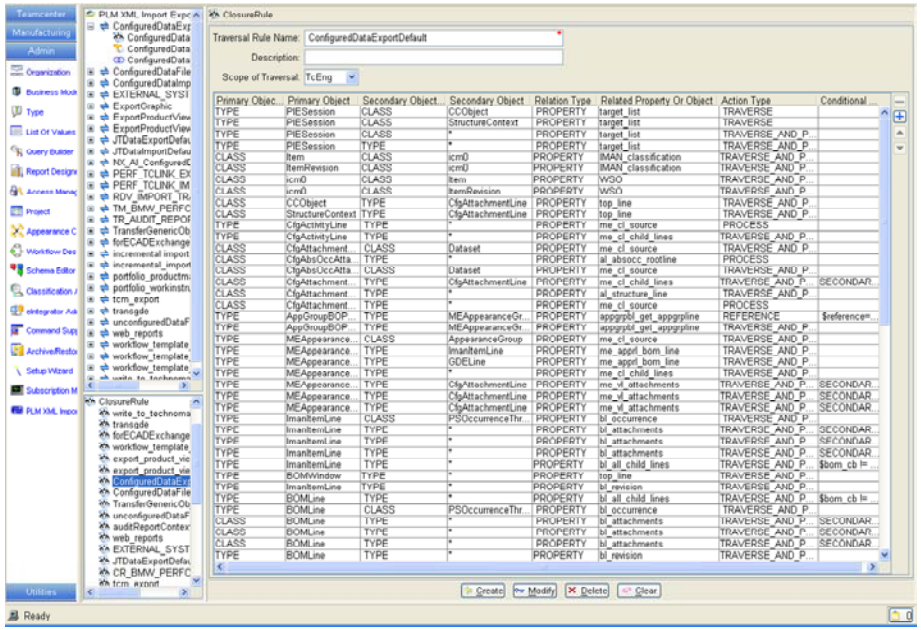
Command line export

```
plmxml_export –u=user –p=pw –g=grp –
xml_file=myAssembly1.xml –item=012345
– rev=A –export_bom=yes –
transfermode=ConfiguredDataExportDefault
```

Creating Transfer Mode Objects








```
CLASS.Document:CLASS.Form:CONTENT.*:SKIP
CLASS.*:CLASS.ProductView:CONTENT.*:PROCESS
CLASS.ProductDef:CLASS.InstanceGraph:CONTENT.*:SKIP
CLASS.*:CLASS.*:CONTENT.*:DO
CLASS.*:CLASS.*:Attribute.*:DO
```

Loading PLMXML

```
var root = new ActiveXObject("Msxml2.DOMDocument.5.0");

  root.async=false;
  root.load(“myNEW.plmxml);
  root.setProperty("SelectionLanguage", "XPath");
 if (root.parseError.errorCode != 0)
  {
    result = reportParseError(source.parseError);
    alert(result);
  }
```

Transforming with XSLT

```
stylesheet = new ActiveXObject("Msxml2.DOMDocument.5.0");
stylesheet.async = false;
stylesheet.load("hilite-xml.xsl");

if (root.parseError.errorCode != 0) {
    alert( reportParseError(source.parseError));
  }

 try {
  result = root.transformNode(stylesheet);
 }
 catch (e) {
  alert( reportRuntimeError(e));
 }
```

Query with XPath

```
function doQuery(theQuery) {
   var q = source.selectNodes(theQuery);
   alert(q.length + " Objects Found")
   for (var e = q.nextNode(); e != null; e = q.nextNode() {
     try {
            alert( "Node "   + e.xml )
     }
     catch (exception) {
        alert reportRuntimeError(exception);
    }
   }
  }
```

theQuery="PLMXML/ProductRevision "
theQuery="PLMXML/ProductRevision/AssociatedDataSet
theQuery="PLMXML/Product[@id='id1245'] "
theQuery="PLMXML/Form[@role='IMAN_MASTER_FORM']/UserData/UserValue"
theQuery="PLMXML/DataSet[@type='DirectMode']"
theQuery="PLMXML/ExternalFile[@id='id40']/locationRef

Evaluting ProductRevision Nodes


```
<?xml version="1.0" encoding="utf-8" ?>
- <PLMXML schemaVersion="5.1" date="2008-04-04" time="10:17:53" author="TcEng - infodba@CXDF(475796077)">
    <Header id="id1" traverseRootRefs="#id6" transferContext="JTDataExportDefault" />
  - <ProductRevision id="id12" name="Base Fixture" accessRefs="#id8" subType="CORPORATE_Part Revision" releaseStatusRefs="#id15" masterRef="#id14" revision="A">
      <ApplicationRef version="RbGdMRc9BTZ8mD" application="TcEng" label="xBCdMRc9BTZ8mD" />
    - <UserData id="id13">
        <UserValue value="XTY99912/A-B BASE FIXTURE" title="object_string" />
      </UserData>
      <AssociatedForm id="id18" role="IMAN_specification" formRef="#id16" />
      <AssociatedForm id="id23" role="IMAN_master_form" formRef="#id20" />
    </ProductRevision>
  - <ProductRevision id="id34" name="FACE PLATE" accessRefs="#id8" subType="CORPORATE_Part Revision" releaseStatusRefs="#id43" masterRef="#id36" revision="A">
      <Description>FACE PLATE</Description>
      <ApplicationRef version="xnKdMRc9BTZ8mD" application="TcEng" label="xZDdMRc9BTZ8mD" />
    - <UserData id="id35">
        <UserValue value="RGH01000235127/A-FACE PLATE" title="object_string" />
      </UserData>
      <AssociatedDataSet id="id39" dataSetRef="#id37" role="IMAN_Rendering" />
      <AssociatedForm id="id46" role="IMAN_specification" formRef="#id44" />
      <AssociatedForm id="id51" role="IMAN_master_form" formRef="#id48" />
```


```
nodes=source.getElementsByTagName("ProductRevision");
//or nodes = root.selectNodes("PLMXML/ProductRevision");

for ( var i=0; I <parts.length; i++) {
    var productRevId=nodes[i].getAttribute("id");
    var descriptionNode=nodes[i].getElementsByTagName("Description");
    var item_rev=nodes[i].getAttribute("revision");
    var item_name = nodes[i].getAttribute("name");
    var masterRef=nodes[i].getAttribute("masterRef").substr(1);

}
```

Evaluating Form Nodes

```
- <Form id="id201" name="CAPBASE_ M8-100 MODIFIED" accessRefs="#id8" subType="RR PDM Form" releaseStatusRefs="#id204" subClass="RR PDM Form">
    <ApplicationRef version="RvPdMRc9BTZ8mD" application="TcEng" label="RvPdMRc9BTZ8mD" />
  - <UserData id="id202" type="FormAttributes">
      <UserValue value="RGH01000235130_AssyType" title="CORPORATE_AssyType" />
      <UserValue value="RGH01000235130_CoProdInd" title="CORPORATE_CoProdInd" />
      <UserValue value="RGH01000235130_ConClass" title="CORPORATE_ConClass" />
      <UserValue value="RGH01000235130_Description" title="CORPORATE_Description" />
      <UserValue value="RGH01000235130_MRPCat" title="CORPORATE_MRPCat" />
      <UserValue value="RGH01000235130_MetaRev" title="CORPORATE_MetaRev" />
      <UserValue value="RGH01000235130_PlanQty" title="CORPORATE_PlanQty" />
      <UserValue value="RGH01000235130_SAPDefUnit" title="CORPORATE_SAPDefUnit" />
      <UserValue value="RGH01000235130_SSA" title="CORPORATE_SSA" />
    </UserData>
  </Form>
```

```
var formNode = root.selectNodes("PLMXML/Form[@id='" + form_id  +
                    "']/UserData[@type='FormAttributes']/UserValue" );

var productName = root.selectNodes("PLMXML/Form[@id='" + form_id  +
                    "']")[0].getAttribute("name");

for (var ii = 0; ii < formNode.length; ii++) {

        title=formNode[ii].getAttribute("title");

        value=formNode[ii].getAttribute("value");
}
```

Evaluating Form Nodes


```
  </Product>
- <Product id="id63" name="ADJ HEEL PIN MODIFIED" accessRefs="#id8" subType="CORPORATE_Part" productId="RGH01000235151">
    <Description>WDS610-508 MODIFIED HEEL PIN</Description>
    <ApplicationRef version="xJMdMRc9BTZ8mD" application="TcEng" label="xJMdMRc9BTZ8mD" />
  </Product>
- <Product id="id90" name="LOCATION SPIGOT" accessRefs="#id8" subType="CORPORATE_Part" productId="RGH01000235135">
    <Description>LOC SPIGOT OD 510 X ID 205 X 125</Description>
    <ApplicationRef version="xVIdMRc9BTZ8mD" application="TcEng" label="xVIdMRc9BTZ8mD" />
  </Product>
- <Product id="id162" name="CLAMP" accessRefs="#id8" subType="CORPORATE_Part" productId="RGH01000235141">
    <Description>CLAMP 195 X 105 X 40</Description>
    <ApplicationRef version="xRNdMRc9BTZ8mD" application="TcEng" label="xRNdMRc9BTZ8mD" />
  </Product>
- <Product id="id189" name="CAPBASE_ M8-100 MODIFIED" accessRefs="#id8" subType="CORPORATE_Part" productId="RGH01000235130">
```


```
var productNode = root.selectNodes("PLMXML/Product[@id='" + masterRef + "']");

var productID=productNode[0].getAttribute("productId");

var master_formID=AssociatedDataSet[0].getAttribute("formRef").substr(1);

var AssociDataSet=nodes[i].selectNodes("AssociatedForm[@role='IMAN_specification']");

var pdm_formID=AssociatedDataSet[0].getAttribute("formRef").substr(1);
```

Setting new attribute value

```
var formNodes = root.selectNodes("PLMXML/Form[@id='" +
                object.id + "']/UserData/UserValue[@title='" + object.name + "']" );
```

```
formNodes[0].setAttribute( "value" , object.value);
```

Saving Nodes

```
function SaveXML (filename){

  if ( filename.length == 0 ) {
            alert ("Destination file name required !");
            return (0);
     }

            alert("Saving " + filename + ".xml");

            root.save( filename + ".xml");
}
```

Summary

- Growing needs for robust product data exchange.
- Why an existing standards are not adequate?
- PLMXML as vertical implantation open IT standard could become “de facto” standard of PLM .
- Overview of MSXML as one of many available XLM toolboxes.
- Example of code bites for managing PLMXML files.
- Demo of practical examples

Questions ?