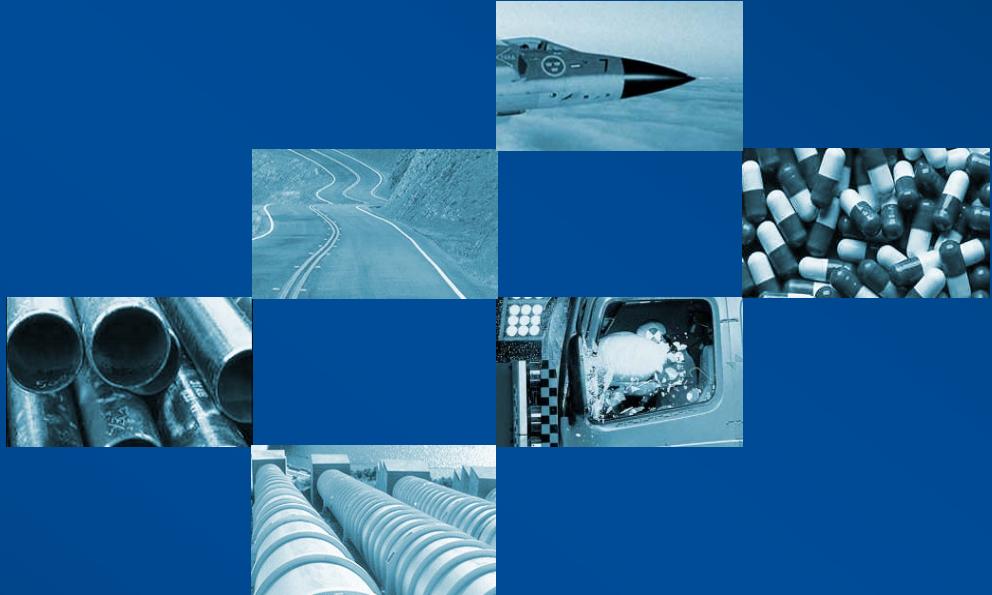


STEPmod Tutorial



An Introduction to the STEP Modules
Repository



Introductions

Rob Bodington

- Eurostep
<http://www.eurostep.com>
- Funded by NIST to provide training
- Technical lead on module repository development
- Modeller on PLCS

Aims of course

- Get you started as a Module developer
- Will not cover the modules
- Will show how to build a module

Before we start

- Module developer ??
- General interest ??
- XML ??
- Used the stepmod XSL

Overall Agenda

- A Reminder about STEP Modularization
- Introducing the STEP Modules Repository
- Building the STEP Module Content
- Repository Details for Module Developers

Intro XML
Build a module

CVS, HTML,
Ballots,

A Reminder about STEP Modularization

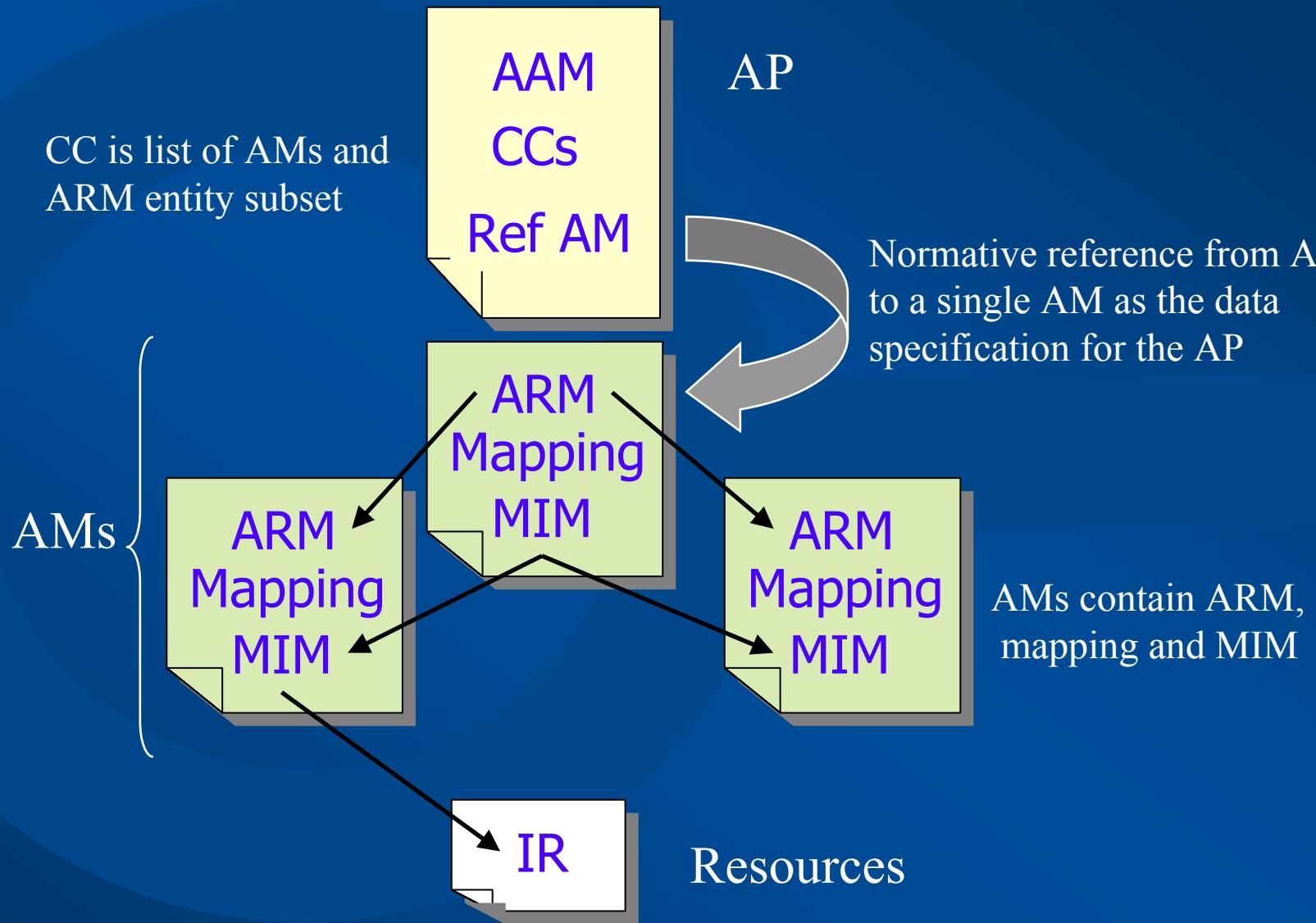
Section Agenda

1. STEP Modular Architecture
2. Content of a STEP Application Module

Modularization Requirements

- Eliminate duplication of the same requirements in different APs
 - Harmonization
- Reduce AP development cost and time to standardization
 - Facilitate reuse of common resources
- Reduce AP implementation cost and time-to-market
 - Enable application software reuse

Architecture Diagram



Modular Facts

- Important Application Module (AM) facts
 - An AM must always use other AMs in their entirety
 - The ARM and MIM are in EXPRESS
 - The AM size, scope and domain are not prescribed
 - Not all AMs are implementable
- Important Application Protocol (AP) facts
 - The AP contains no ARM-mapping-AIM, it references a single AM as its data specification
 - The AP itself, is therefore reusable
 - Conformance Classes are against an implementation module allowing a subset of ARM entities as well

Effects of the Modular Approach

- In order to support reusability, the modular approach has led to numerous small modules
 - Managing these, is the main reason for the development of the STEP Modules Repository
- Need for separation of technical content from presentation led to use of XML for the standards documents themselves
- No formal "Harmonization team" has materialized, so AP teams are working together individually
 - The use of the repository facilitates this

A Reminder about STEP Modularization

Section Agenda

1. STEP Modular Architecture
2. Content of a STEP Application Module

Content of a STEP Module

A STEP module is a "baby AP" containing

- Foreword, Introduction, Scope, Normative References, Terms and abbreviations
- Information requirements
 - Application Reference Model (ARM) schema
- Module Interpreted Model
 - Mapping specification
 - MIM schema
- Annex for each of:
 - MIM short names
 - Information object registration
 - ARM and MIM EXPRESS-G
 - Computer interpretable listings
 - Usage guide
 - Bibliography

Overall Agenda

- A Reminder about STEP Modularization
- Introducing the STEP Modules Repository
- Building the STEP Module Content
- Repository Details for Module Developers

Introducing the STEP Modules Repository

Section Agenda

1. What is the repository?
2. Where is the repository?
3. Navigating the repository
4. Repository structure

The STEP Modules Repository

The repository ...

- is really the use of a comprehensive set of software development tools to manage the development of STEP Application Modules, STEP Common Resource and STEP Application Protocols;
- is hosted on the Web where these services are freely available and where many open-source projects are hosted.

- STEPmod is XML based!
- STEPmod is an open-source project!

The STEP Modules Repository(2)

The repository consists of two parts.

1. Software Framework

The software developed to support the creation of AM and AP technical content and their ISO ballot and publication.

- Developed by Eurostep under contract to NIST and PDES
- Contributions from Boeing for PLCS

2. Technical Content

The schemas and other technical content that make up the AMs and APs created by the AP teams themselves

- PDES, Inc. drove this with the PDM modules
- Now has PLCS (AP239), AP236, AP221, AP203e2 modules
- And others

Why use a repository?

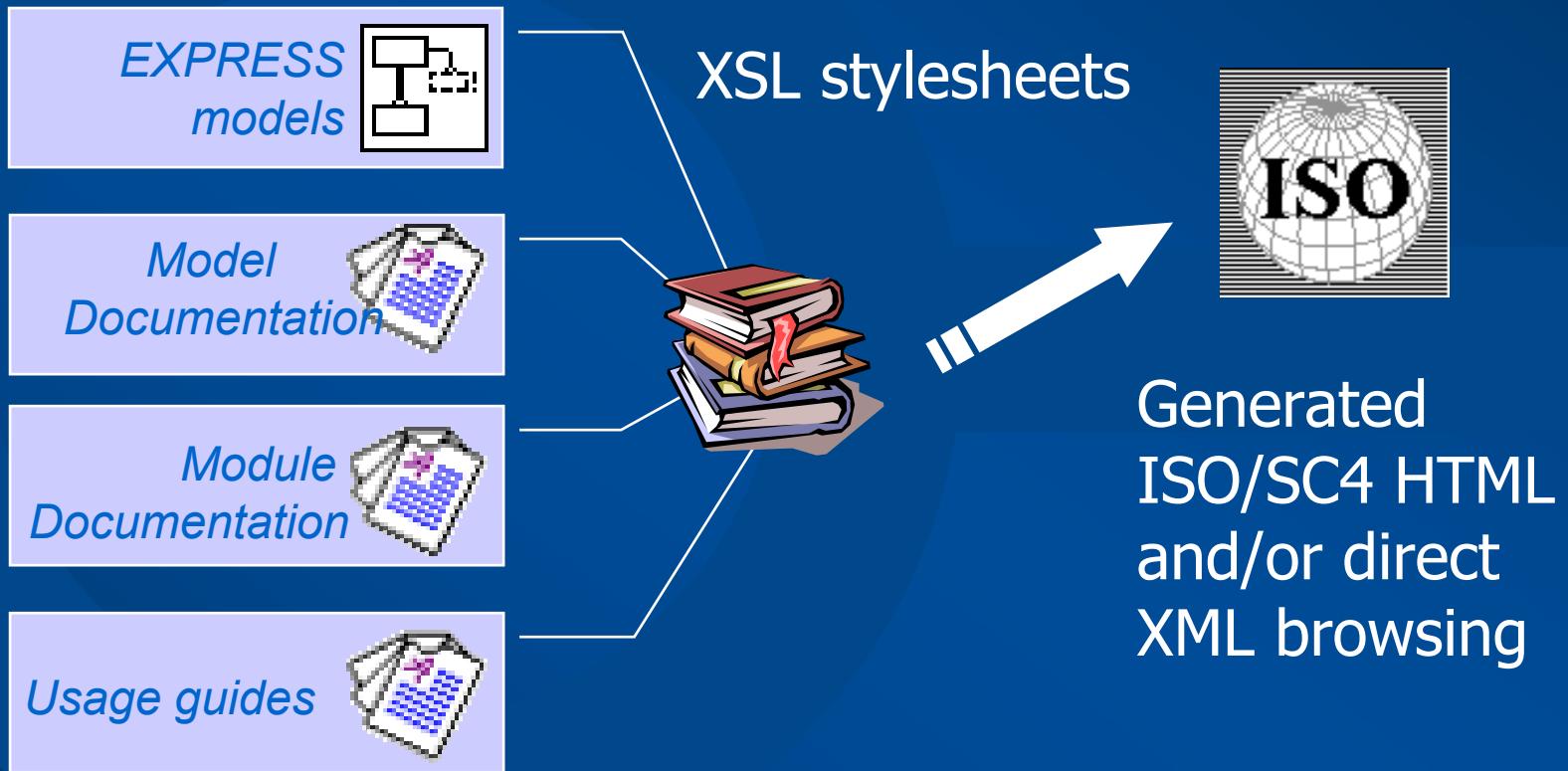
- Centralized location on the Web for access by all STEP developers
 - Single source for all modules and APs
- Formalize modules change management during development
- Reduces costs and wasted time
- Consistency of approach and resulting ISO documents

Why use XML?

- Reduce cost of development by allowing STEP expert to focus on technical content, not presentation
 - Presentation generated using XML Stylesheets
 - e.g. ISO and SC4 guidelines and boilerplate implemented once in XML stylesheets rather than repeatedly Word/HTML for each module
- Allow STEP developers to build in quality through the use of good tools rather than word processors
 - Tools can check for invalid references within and between modules, check mappings, module completeness

Repository Diagram

XML documents



XML enables other uses too!

To allow publication on the web for multiple purposes

Published
standards



Hyperlinked
web pages



Domain specific
view



How does presentation work?

XML

```
<express>
  <schema
    name='product'
  </schema>
</express>
```

XSL

```
<?xml version='1.0'?>
<xsl:stylesheet version="1.0"
  xmlns:xsl="http://www.w3.org/1999/XSL/Transform">
  <xsl:output method="html"/>
  <xsl:template match="/">
    <HTML>
      <HEAD>
        <TITLE><xsl:value-of
          select="/express/schema/@name"/></TITLE>
      </HEAD>
      <BODY>
        EXPRESS specification:
        *)
        <H1><xsl:value-of
          select="/express/schema/@name"/></H1>
        (*
        </BODY>
      </HTML>
    </xsl:template>
  </xsl:stylesheet>
```

HTML

```
<HTML>
  <HEAD>
    <TITLE>product_schema_mim</TITLE>
  </HEAD>
  <BODY>
    EXPRESS specification:
    *)
    <H1>product_schema_mim</H1>
    (*
    </BODY>
  </HTML>
```



Who will use the XML repository?

- Standards developers
 - Repository will reduce AP development costs through eliminating unnecessary repetition and better reuse of common resources
 - This will lead to improved AP interoperability
- Implementors
 - Repository will make it easier to build implementations containing multiple AP subsets or extensions

What do AP teams do?

- Use a script that automatically generates the folders and template files for your module
- Edit the generated XML files to provide the modules technical content
- Create your ARM and MIM EXPRESS/EXPRESS-G
 - Use scripts or tools to convert to XML
- Re-use other teams modules
- Check and view the module results to make sure you see what you want

What does it cost?

- The repository is free to use
- The real costs of using the repository are
 - time associated with setup as a developer
 - including software installation on your computer
 - time associated with learning and understanding how the necessary software works and how the repository is structured
 - if you've never participated in software projects, things will seem a bit strange at first

Introducing the STEP Modules Repository

Section Agenda

1. What is the repository?
2. Where is the repository?
3. Navigating the repository
4. Repository structure

The repository is at SourceForge

- SourceForge
 - is an open-source software project management Web site
 - hosts projects for free!
 - provides a comprehensive set of software management tools
 - Version control
 - Bug and issue tracking (not used by STEPmod)
 - Backups and archives
 - Communication and collaboration services
 - can be browsed by anyone on the Web
 - only provides write access to registered developers

OSDN: Webservices - Newsletters - Shop

SEARCH: All OSDN Sites

Go

NEWSFORGE**All Technology
News, All the time.**

my sf.net

software map

foundries

about sf.net

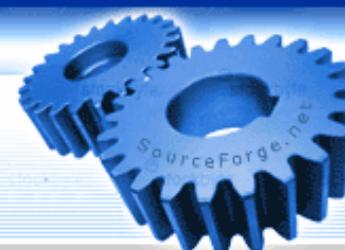
My Favor

[Login via SSL](#)
[New User via SSL](#)**Search****SF.net Resources**

- [· Site Docs](#)
- [· Site Status](#)
- [· Site Map](#)
- [· Compile Farm](#)
- [· Project Help Wanted](#)
- [· New Releases](#)
- [· Contact SF.net](#)

Support

SourceForge.net is the world's largest Open Source software development website, with the largest repository of Open Source code and applications available on the Internet. SourceForge.net provides free services to Open Source developers.

**Project of the Month**

Every month the SF.net team picks one outstanding project to highlight the software and personality that drive the Open Source Community!

Project of the Month for June 2003
[MegaMek](#)

SourceForge.net Statistics

Hosted Projects: **63,724**
Registered Users: **643,947**

SourceForge.net Newsletter

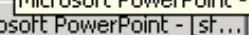
Email Address:



SourceForge.net: Wel...



Microsoft PowerPoint - [sf.ppt]



Microsoft

PowerPoint

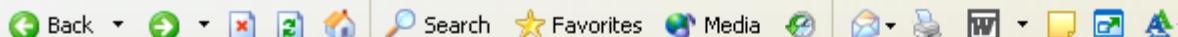
[sf...



5:04 PM

stepmod

- The repository is known as "stepmod project on SourceForge
 - Web address is <http://stepmod.sourceforge.net>
 - Project summary is
<http://sourceforge.net/projects/stepmod>
- For developers, there is an email exploder
 - <http://lists.sourceforge.net/lists/listinfo/stepmod-devel>



STEP Module Repository Project

The STEP Module Repository is a collection of resources tagged in XML to serve as the core of a modular environment for developers of STEP, a family of product data exchange standards.

SourceForge Resources

- [Project Summary](#)
- [Downloads](#)
- [Browse CVS Repository](#)
- [CVS Access Instructions](#)
- stepmod-devel mailing list
 - [Archives](#)
 - [Subscribe/Unsubscribe/Preferences](#)

CVS Help for Developers

- [NIST CVS/SSH help page](#)
- [Rob Bodington's notes](#)

For More Information

See the National Institute of Standards and Technology's [STEP Modularization home page](#).

NEWSFORGE

**All Technology
News, All the time.**

my sf.net

software map

foundries

about sf.net

My Favo

Login via SSL

New User via SSL

Search

Software/Group ▾

Search

SF.net Resources

- Site Docs
- Site Status
- Site Map
- Compile Farm
- Project Help Wanted
- New Releases
- Contact SF.net
- Support

Project: STEP Module Repository: Summary

[Summary](#) | [Admin](#) | [Home Page](#) | [Forums](#) | [Tracker](#) | [Bugs](#) |
[Support](#) | [Patches](#) | [RFE](#) | [Lists](#) | [Tasks](#) | [Docs](#) | [News](#) |
[CVS](#) | [Files](#) |

A collection of resources tagged in XML to serve as the core of a modular environment for developers of STEP, a family of product data exchange standards, and PLCS, a family of standards related to supporting the complete product life cycle.

This project has not yet categorized itself in the

Developer Info

Project
Admins:
[buchanan](#)
[goset1](#)

**Online Bug
Tracking Tool**

bug tracking at [adminitrack.com](#)
no-risk 30-day trial now available
[www.adminitrack.com](#)

**Tenrox Incident
Tracking**

Web based Bug & Issue Workflows
Highly Rated & Great integrations
[www.inchange.com](#)



5:04 PM

stepmod

- Download using CVS
 - More about that later
- There is an HTML mirror on SC4Online:
 - <http://www.tc184-sc4.org/private/>
 - http://www.tc184-sc4.org/private/STEP_Modules_Mirror/

SC4 Private > STEP Modules Mirror



STEP Modules Repository

STEP information modules are developed in a STEP modules repository, hosted as SourceForge project, with controlled access for developers. Modules in the repository are under development, and are subject to continuous change and refinement, without notice, until they are released for review and ballot by the ISO community.

The modules that have been released for ballot are available as SC4 documents, and approved modules are available as ISO parts.

This site provides a mirror of the STEP modules repository, which is updated from the [SourceForge](#) site on a daily basis, and should be used for informative purposes only does not reflect either the latest development or publication status of the STEP modules.

Repository last updated (latest tarfile received at): **2003-06-13 22:26:06**

Local repository last updated (last successful HTML build) at: **2003-06-05 00:44:47**

- [Module repository online](#)
- [Module Repository \(HTML\) Zipped \(45MB\)](#)

- SC4 Public
- SC4 Open**
- Welcome
- SC4 Members
- Exploder Lists
- FTP
- SC4 QC and Working Groups
- Meetings
- Projects
- STEP Modules Mirror
- Technical Support
- Manage My Subscription

Introducing the STEP Modules Repository

Section Agenda

1. What is the repository?
2. Where is the repository?
3. Navigating the repository
4. Repository structure

Browsing the modules

- The next few slides show examples of browsing the repository
 - Main index of all modules is stepmod\nav\index.xml
- You can get:
 - an index of all modules
 - an index of all entities in all module ARMs
 - a table of dependencies between modules
- There's no end to the list of indices that can be produced for the repository

Tools to browse

- You can browse the XML or HTML version.
- HTML
 - is a snap shot.
 - should be able to use any browser
- XML
 - is dynamic – any change to the module is immediately visible.
 - You need:
 - Internet Explorer v6
 - XSL capable
 - Internet Explorer v5
 - Install MSXML3 (xmlinst)

STEPmod

Modules

Express

Application Protocols

Resource parts

Resource schemas

Balloons

[Alphabetical, Project Parts, Leader, Status, Keywords, Definitions, Dependencies](#)

[ARM objects](#)

[Alphabetical, Project, Parts, Leader](#)

[MIM objects](#)

[Alphabetical, Mappings](#)

[Resource objects](#)

[xml \(Date: 2003/01/01\)](#)

ISO/CD-TS 10303-1134:- Product structure

N

+[name assignment](#)

+[numeric function](#)

+[numerical interface](#)

+[nut and bolt](#)

O

+[observation](#)

+[organization type](#)

P

+[parametric catalog data](#)

+[parametric catalog data](#)

+[part and version identific](#)

+[part collection](#)

+[part definition relationshi](#)

+[part view definition](#)

+[person organization](#)

+[person organization assi](#)

+[physical breakdown](#)

+[plib class reference](#)

+[position in organization](#)

+[possession of property](#)

+[probability](#)

+[probability distribution](#)

+[process property assignm](#)

+[product as individual](#)

+[product behaviour](#)

+[product breakdown](#)

+[product categorizatio](#)

+[product class](#)

+[product concept identifica](#)

+[product group](#)

+[product identification](#)

+[product occurrence](#)

+[product placement](#)

+[product relationship](#)

+[product replacement](#)

+[product structure](#)

+[product structure and cla](#)

+[product version](#)

+[product version relations](#)

+[product view definition](#)

+[product view definition pi](#)

+[product view definition re](#)

+[project](#)

+[property and property as](#)

+[property assignment](#)

+[property condition](#)

+[property identification](#)

+[property space](#)

Q

+[qualifications](#)

R

+[reference data library](#)

Alphabetized index of module names

Other indices you can use

ISO view of selected module

Introduction

ISO 10303 is an International Standard for the computer-interpretable representation of product information and for the exchange of product data. The objective is to provide a neutral mechanism capable of managing products throughout their life cycle. This mechanism is suitable not only for neutral file exchange, but also as a basis for implementing and sharing product databases, and as a basis for archiving.

This part of ISO 10303 specifies an application module for the representation of assemblies of parts, of their properties, of their shape and of their associated documentation.

Clause 1 defines the scope of the application module and summarizes the functionality and data covered. Clause 3 lists the words defined in this part of ISO 10303 and gives pointers to words defined elsewhere. The information requirements of the application are specified in clause 4 using terminology appropriate to the application. A graphical representation of the information requirements, referred to as the application reference model, is given in annex C. Resource constructs are interpreted to meet the information requirements. This interpretation produces the module interpreted model (MIM). This interpretation, given in 5.1, shows the correspondence between the information requirements and the MIM. The short listing of the MIM specifies the interface to the resources and is given in 5.2. A graphical representation of the short listing of the MIM is given in annex D.

STEPmod **Modules** **Express** **Application Protocols** **Resource parts** **Resource schemas** **Balloons**

Alphabetical, Project
Parts, Leader, Status, Keywords,
Definitions, Dependencies

ARM objects
MTM objects
Resource objects

Alphabetical, Project,
Parts, Leader

Alphabetical,
Mappings

Balloons

HTML Help
Module menu □

ARM Objects

schemas constants
types entities rules
functions

Schemas [top]

Activity arm
Activity as realized arm
Activity characterized arm
Activity method arm
Activity method assignmen
Activity method characteriz
Activity method implement
Activity structure and classi
Advanced boundary repres
Alias identification arm
Alternative solution arm
Ap203e2 configuration man
Ap236 furniture catalog an
Ap239 activity recording ar
Ap239 configuration manag
Ap239 document managen
Ap239 management resou
Ap239 part definition infor
Ap239 product definition ir
Ap239 product life cycle su
Ap239 product status recor
Ap239 properties arm
Ap239 task specification re
Ap239 work definition arm
Appearance assignment ar
Approval arm
Assembly structure arm
Associative text arm
Attachment slot arm
Attribute classification arm
Building component arm
Building item arm
Building structure arm
Cardinality of relationship
Catalog data information s
Catalog data information s
Certification arm
Class arm
Class of activity arm
Class of activity library arm
Class of activity structure i
Class of composition of ac

ISO/TS 10303-1047:2003 Activity

entity definitions
selected model
definition
3 short listing
definition
definition

D MIM EXPRESS-G
E Computer interpretable listings
Bibliography

Alphabetized index of ARMs

4 Information requirements

The clause specifies the information requirements for the **Activity** application module of the Application Reference Model (ARM) of this application module.

NOTE 1 A graphical representation of the information requirements is given in [Annex A](#).

NOTE 2 The mapping specification is specified in [5.1](#). It shows how the information requirements are defined or imported in the MIM schema of this application module.

The following EXPRESS specification begins the **Activity_arm** schema and identifies the necessary external references.

EXPRESS specification:

```
*)  
SCHEMA Activity_arm;  
(*
```

4.1 Required AM ARM

The following EXPRESS interface statement specifies the elements imported from the ARM of another application module.

EXPRESS specification:

```
*)  
USE FROM Activity\_method\_arm; -- ISO/TS 10303-1049  
(*
```

ISO view of selected ARM

STEPmod **Modules** **Express** **Application Protocols** **Resource parts** **Resource schemas** **Balloons**

HTML Help Module menu

Alphabetical, Project Parts, Leader, Status, Keywords, Definitions, Dependencies

ARM objects MIM objects Resource objects

Alphabetical, Project Parts, Leader

Alphabetical, Alphabetical, Mappings

Activity method
Activity method arm
Activity method assignment
Work request arm
Activity method realization
Activity method imp.
Activity method realization
Activity method imp.
Activity method relationship
Activity method assi.
Activity property
Process property assi
Activity property represents
Process property assi
Activity relationship
Activity arm
Activity status
Activity arm
Address
Person organization a.
Address assignment
Person organization a.
Address based location ref
Location arm
Advanced_brep_shape repr
Advanced boundary re
Advanced face
Topologically bounded
Advisory_task_step
Task specification arm
Affected_items_assignment
Work request arm
Alias identification
Alias identification arm
Alternate_part_relationship
Product structure arm
Alternate_product_relationship
Product replacement s
Alternative_solution
Alternative solution a
Amount_of_substance_unit
Value with unit arm
And_expression

ISO/TS 10303-1047:2003 Activity

Alphabetized index of ARM entities

Open two instances of **Activity**.

be decomposed into a series of activities. Their corresponding instances would be

To define the **Activity** specification:

```
*)  
ENTITY Activity_relationship;  
description : OPTIONAL STRING;  
name : STRING;  
related_activity : Activity;  
relating_activity : Activity;  
END_ENTITY;  
(*
```

Attribute definitions:

description: a text that provides further information about the Activity.

name: the text by which the **Activity_relationship** is known.

related_activity: the second instance of **Activity** that is part of the relationship.

NOTE 3 The **related_activity** usually identifies the Activity, which is based on the definition of the **relating_activity**.

NOTE 4 The meaning of this attribute is defined by the **name** attribute.

relating_activity: the first of the instances of **Activity** that is part of the relationship.

NOTE 1 The **relating_activity** usually identifies the activity the definition of the **related_activity** is based on, for example, derived from or dependent on.

ISO view of selected ARM entity

STEPmod **Modules** **Express** **Application Protocols** **Resource parts** **Resource schemas** **Balloons**

[Alphabetical, Project Parts, Leader, Status, Keywords, Definitions, Dependencies](#)

[ARM objects](#) [MIM objects](#) [Resource objects](#)

[Alphabetical, Parts, Leader](#) [Alphabetical, Mappings](#)

[Balloons](#)

A

- + activity
- + activity as realized
- + activity characterized
- + activity method
- + activity method assignr
- + activity method charact
- + activity method implr
- + activity structure_and_cl
- + advanced_boundary_rep
- + alias_identification
- + alternative_solution
- + ap23e2_configuration
- + ap236_furniture_catalog
- + ap239_activity_recording
- + ap239_configuration_ma
- + ap239_document mana
- + ap239_management_re
- + ap239_part_definition_ir
- + ap239_product_definitio
- + ap239_product_life_cycle
- + ap239_product_status_r
- + ap239_properties
- + ap239_task_specificatio
- + ap239_work_definition
- + appearance_assignment
- + approval
- + assembly_structure
- + associative_text
- + attachment_slot
- + attribute_classification

B

- + building_component
- + building_item
- + building_structure

C

- + cardinality_of_relationship
- + catalog_data_informatio
- + catalog_data_informatio
- + certification
- + class
- + class_of_activity
- + class_of_activity_library
- + class_of_activity_structur
- + class_of_composition_of
- + class_of_composition_of
- + class_of_connection_of
- + class_of_connection_of
- + class_of_containment_o
- + class_of_description_by
- + class_of_document
- + class_of_document_libr
- + class_of_involvement_in
- + class_of_involvement_of
- + class_of_product

Module repository dependencies

Modules and their dependencies

Shows which modules use and are used by each module's ARM. Links in the Depends on and Used by columns take you to the entry in this table for the named module.

Each cell shows the modules used directly above the line and the full set of dependent/using modules below the line.

Module	Depends on	Used by
activity (1047)	Activity_method_arm Activity_method_arm	Activity_characterized_arm Activity_as_realized_arm Ap239_activity_recording_arm Ap239_work_definition_arm Class_of_activity_arm Class_of_composition_of_activity Class_of_connection_of_activity Class_of_involvement_in_activity Composition_of_individual_activit Configuration_control_3d_design Connection_of_individual_activity Individual_activity_arm Individual_involvement_in_activity Process_property_assignment_arm Work_order_arm

Module

Depends on these modules

Is used by these modules

Introducing the STEP Modules Repository

Section Agenda

1. What is the repository?
2. Where is the repository?
3. Navigating the repository
4. Repository structure

On the structure

- The repository is structured as a set of folders/directories on the Web site
 - If you download stepmod, the same structure is duplicated on your computer
- The content of a module is actually created in more than one file
- All files for a module are in the same folder
 - The folder has the same name as the module
- As a module developer, you can ignore most of the other folders – see next few screenshots

The directory/folder structure

Cascading stylesheets
– not used for ISO

Application Protocols

Normative references
contacts, bibliography

The modules

Integrated resource
EXPRESS schemas

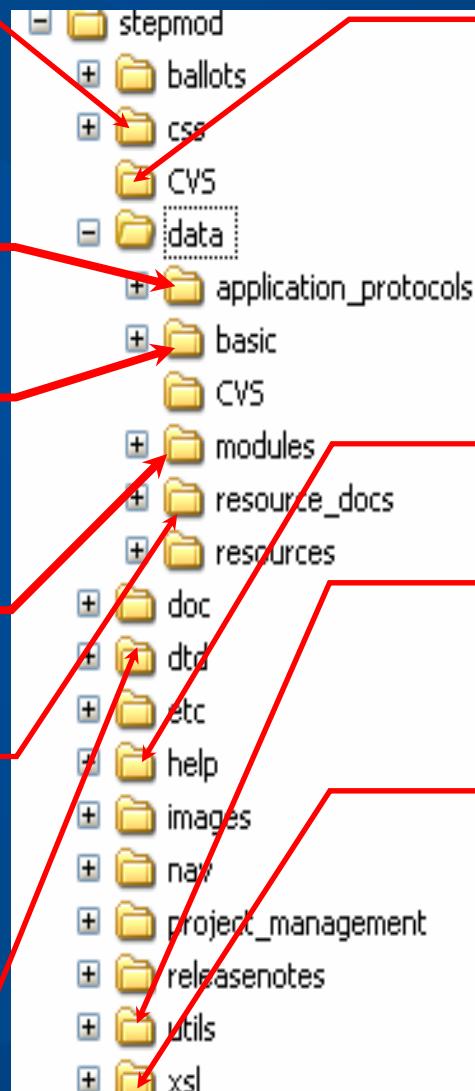
DTDs

CVS directory
⚠ Do not delete

Authoring help

Module authoring
tools

XSL stylesheets





File Edit View Favorites Tools Help

Back Forward Stop Home

Search Favorites History

> Go

Address http://cvs.sourceforge.net/cgi-bin/viewcvs.cgi/stepmod/stepmod/

Current directory: [SourceForge] / stepmod / stepmod

File

[Attic/](#) [\[Don'](#)[ballots/](#)[css/](#)[data/](#)[doc/](#)[dtd/](#)[etc/](#)[help/](#)[images/](#)[nav/](#)[project management/](#)[project managment/](#)[releasenotes/](#)[utils/](#)[xsl/](#)[.cvsignore](#)

1.1

19 months

robbo

Initial revision

or

[Last log entry](#)Data you create
goes here

5:06 PM

Address http://cvs.sourceforge.net/cgi-bin/viewcvs.cgi/stepmod/stepmod/data/modules/ Go

Current directory: [SourceForge] / stepmod / stepmod / data / modules

File

- [Attic/ \[Don't hide\]](#)
- [activity/](#)
- [activity as directed/](#)
- [activity as realized/](#)
- [activity characterized/](#)
- [activity directive/](#)
- [activity method/](#)
- [activity method assignment/](#)
- [activity method characterized/](#)
- [activity method implementation/](#)
- [activity output/](#)
- [activity output characterized/](#)
- [activity structure and classification/](#)
- [activity trigger definition/](#)
- [activiy directive/](#)
- [advanced boundary representation/](#)
- [index.html](#)

Each module has a
folder under
data\modules



[Attic/](#) [Don't hide][dvlp/](#)[nav/](#)[sys/](#)[arm.exp](#)

1.14

3

[arm.xml](#)

1.18

5

[arm_descriptions.xml](#)

1.8

7

[armexpg1.gif](#)

1.3

2

5 months

robbod

Added roles to assignments

[armexpg1.xml](#)

1.9

2

5 months

robbod

Added roles to assignments

[armexpg2.gif](#)

1.6

3

months

robbod

Added roles to assignments

[armexpg2.xml](#)

1.7

4

5 months

robbod

Added roles to assignments

[index.xml](#)

1.1

5

months

robbod

Initial revision

[mim.exp](#)

1.10

3

months

robbod

Initial revision

[mim.xml](#)

1.14

6

months

robbod

Initial revision

[mim_descriptions.xml](#)

1.5

1

months

robbod

Initial revision

[mimexpg1.gif](#)

1.3

2

months

robbod

Initial revision

[mimexpg1.xml](#)

1.8

3

months

robbod

Initial revision

[mimexpg2.gif](#)

1.4

4

months

robbod

Initial revision

[mimexpg2.xml](#)

1.4

5

months

robbod

Initial revision

[module.xml](#)

1.30

6

months

darla

corrected broken links

[module.xsd](#)

1.30

7

months

darla

updated checklist numbers

A module is made up of several files in a single folder

The files have the same name regardless of the module



stepmod/stepmod/da...

Microsoft PowerPoint - [sf...



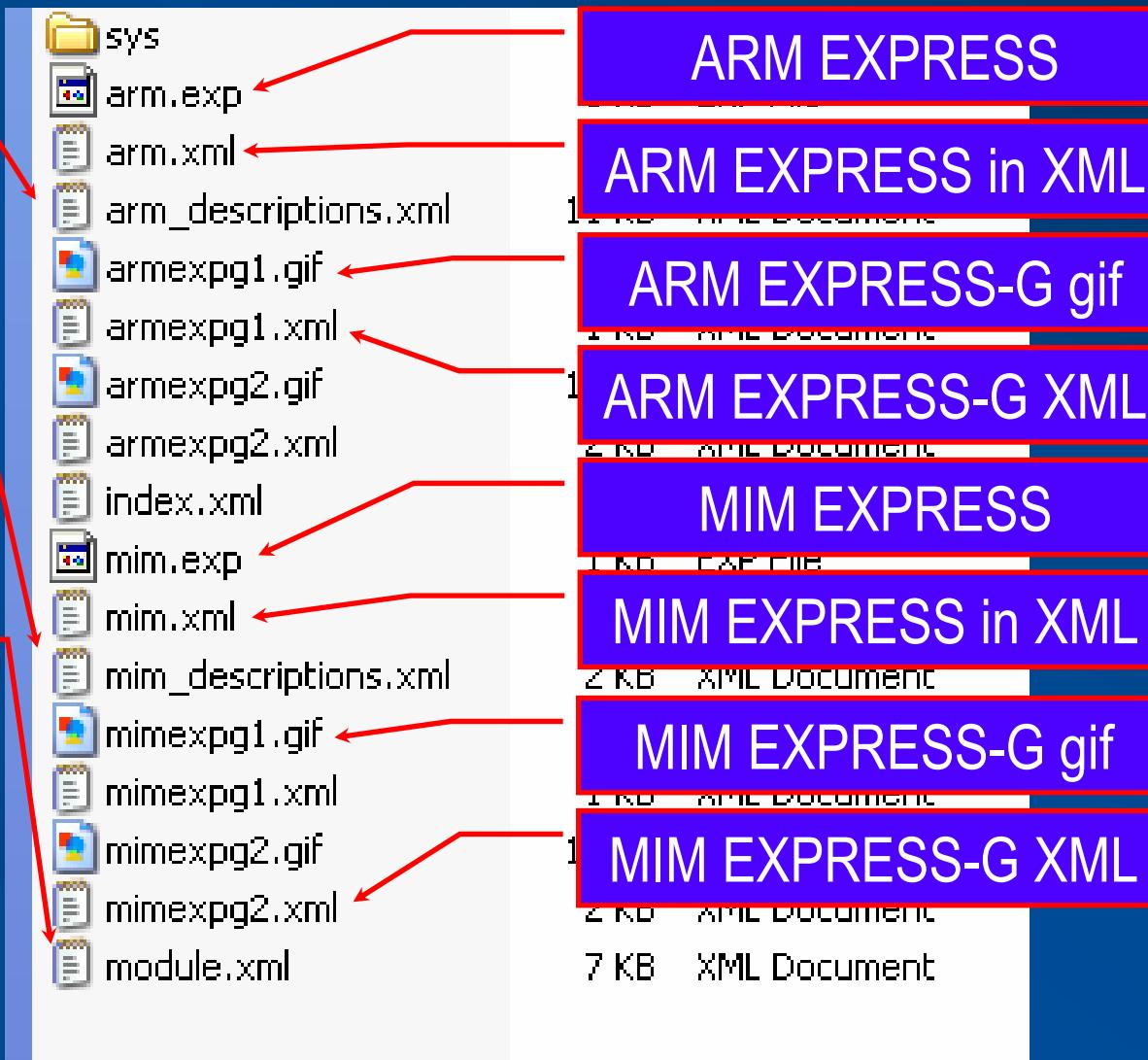
5:07 PM

Module files

ARM schema documentation

MIM schema documentation

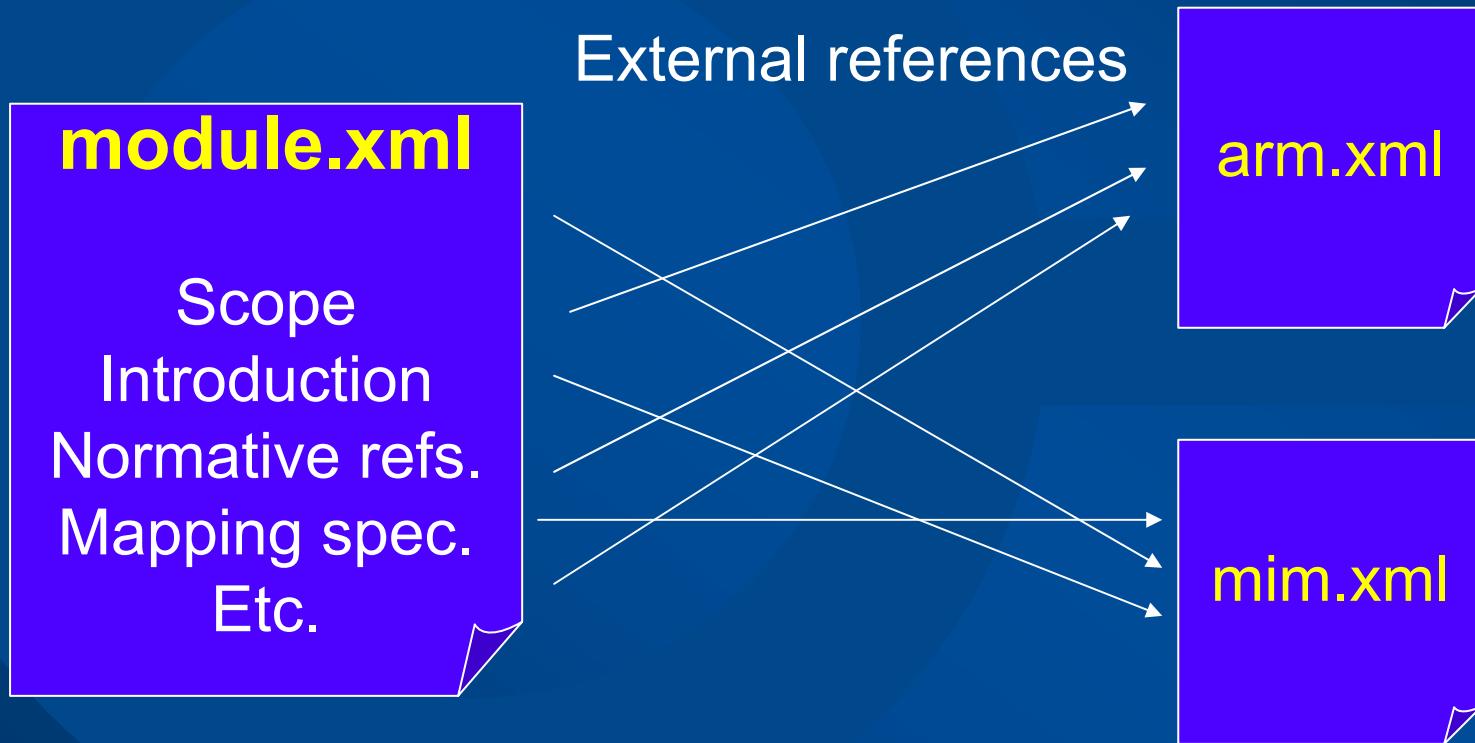
Module documentation
Scope
Introduction
Mapping



Repository logical structure

- All these folders and files have a logical structure that is not exactly the same as the physical folder/file structure
- The logical structure is implemented in the XSL that generates the ISO view of the repository
- Within each file are references to elements in other files
 - These references are made "by name" using a defined naming convention
 - e.g. "Schema_name.Entity_name" for an ARM entity

Logical structure diagram





Overall Agenda

- A Reminder about STEP Modularization
- Introducing the STEP Modules Repository
- Building the STEP Module Content
- Repository Details for Module Developers

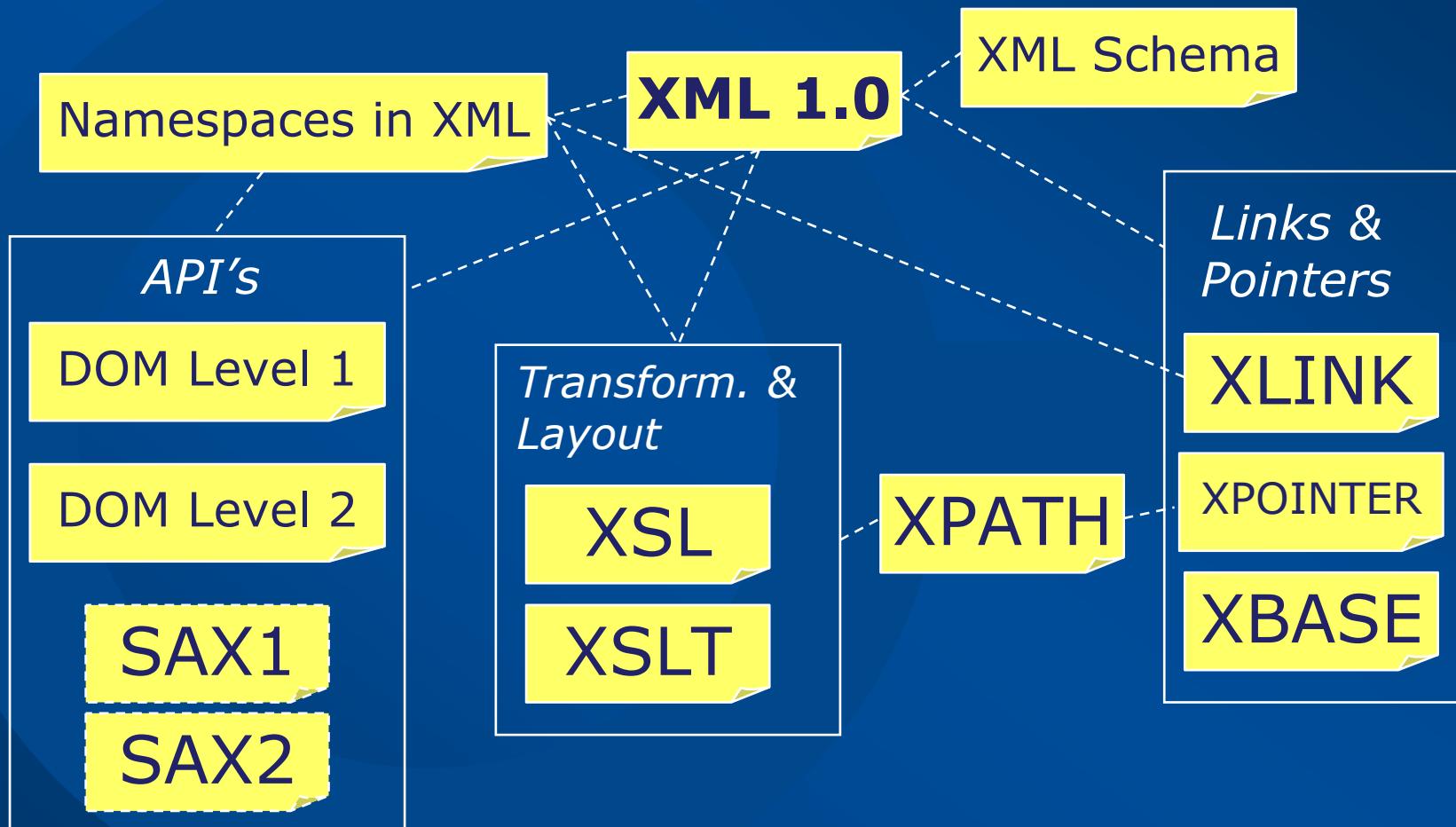
Building the STEP Module Content Section Agenda

1. XML Basics
2. Module document structures
3. Building a module
4. Repository tools used to build a module

XML is familiar

- Before worrying about the details of XML, remember that for our purposes
 - an XML document is like a Part 21 file
 - an XML document contains data
 - there is usually a "schema" for the data
 - this is called a Document Type Declaration or DTD
 - there are also other languages that do the same thing for XML documents (e.g. XML Schema)
 - the DTD declares classes of objects, attributes, relationships and a few data types
 - The only new concept is really just "containment"
 - e.g. "chapters" are contained in "books"

XML is a family of technologies



XML Basics

- eXtensible Markup Language (XML) is standardized by the World Wide Web Consortium (W3C)
 - They standardized HTML too
- In simple terms, XML is a method for tagging text in order to say what the text is
- You may define the tags in a Document Type Declaration (DTD)
 - Like an EXPRESS schema where the XML document is like the Part 21 file
 - Not nearly as powerful as EXPRESS though

XML Basics(2)

- In addition to what your text is, XML allows you to relate things in two ways
 - assigning something an identifier and referencing that identifier
 - using containment ("a" is within "b" so "a" is part of "b" in some sense)
- XML does not state when to use which relationship or what the relationship means

A recipe example



```
<?xml version="1.0"?>
<recipe>
  <ingredients>
    <ingredient>
      <amount unit="cup">3</amount>
      <description>Flour</description>
    </ingredient>
    <ingredient>
      <amount unit="spoon">1</amount>
      <description>Baking powder
      </description>
    </ingredient>
  </ingredients>
  <procedure>Mix flour and baking
  powder...
  </procedure>
</recipe>
```

XML declaration

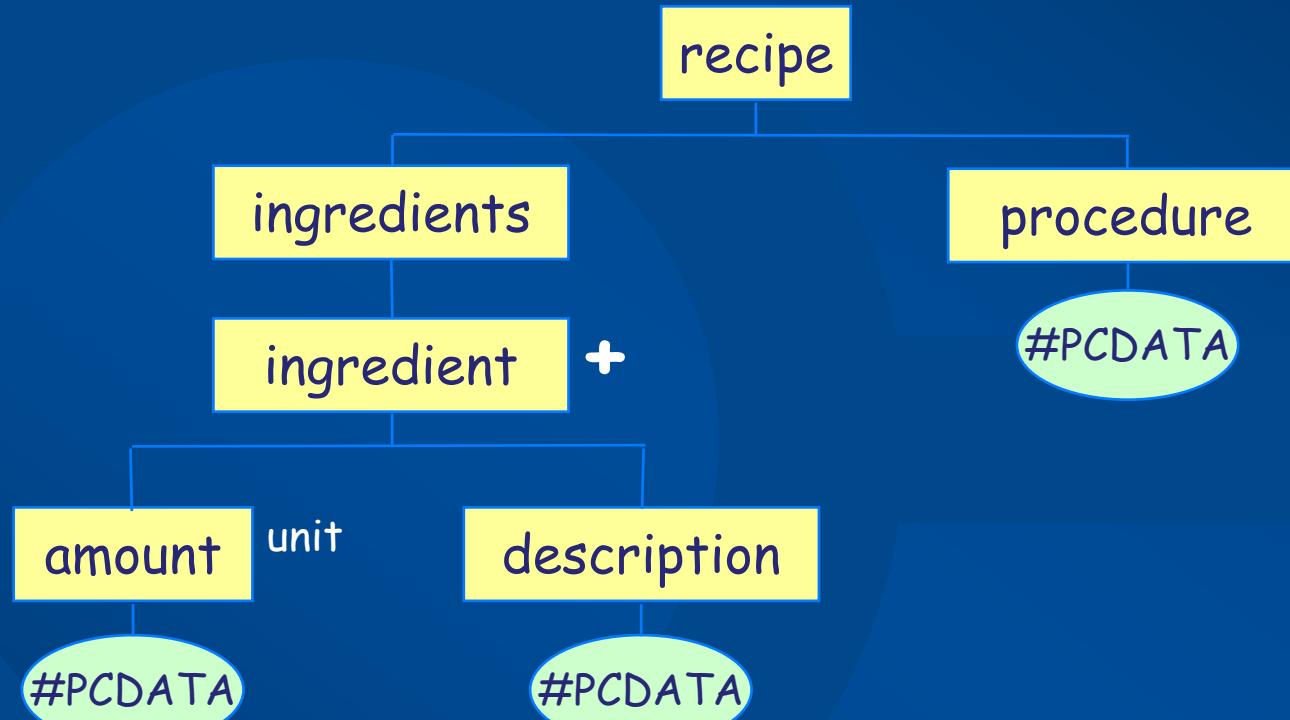
Root element start tag

Attribute

Content

Root element end tag

DTD the “built-in” schema



Main DTD Concepts

- ELEMENT
 - declares a tag and any containment
 - think EXPRESS entity type
- ATTLIST
 - declares attributes of a tag
 - think EXPRESS explicit attribute
- ENTITY
 - reuse of declarations through string replacement
 - think EXPRESS USE FROM but simpler, more like a programming language "include"
- Root
 - every XML document must have a single outermost tag that contains all others

Some EXPRESS

SCHEMA part41;

(* A product is almost anything. *)

```
ENTITY product;  
END_ENTITY;
```

(* A product_definition is about a product. *)

```
ENTITY product_definition;  
END_ENTITY;
```

```
END_SCHEMA;
```

A DTD Fragment

```
<!ELEMENT schema (description?, interface*,  
constant*, (entity | function | procedure |  
rule | subtype.constraint | type)*,  
graphic.element?)>
```

A "schema" contains
"description", "entity",
etc.

```
<!ELEMENT entity (description?, explicit*,  
derived*, inverse*, unique*, where*,  
graphic.element?)>
```

An "entity" contains
"explicit" and other
elements

```
<!ATTLIST entity  
name NMTOKEN #REQUIRED  
abstract.entity (YES | NO) "NO"  
abstract.supertype (YES | NO) "NO"  
supertypes NMTOKENS #IMPLIED  
super.expression CDATA #IMPLIED
```

An "entity" has a "name"
and several other
attributes

Simple XML document

```
<?xml version="1.0"?>
<schema name="part41">
  <entity name="product">
    <description>A product is almost anything.</description>
  </entity>
  <entity name="product_definition">
    <description>A product_definition is about a
      product.</description>
  </entity>
</schema>
```

Building the STEP Module Content Section Agenda

1. XML Basics
2. Module document structures
3. Building your first module
4. Repository tools used to build a module

Approach

- Add an ISO publication capability over a capability to document an EXPRESS schema
 - The EXPRESS schema documentation can be separated from the module documentation as a whole
 - It can even be generated from an EXPRESS toolkit
 - Allows separate creation of the ARM and MIM as documented schemas
 - Allows reuse of EXPRESS-driven capability for other than STEP modules (e.g. ISO 15926 potentially)

A module is more than one file

- The "source code" for a module is created in several XML documents each based on a DTD
- Some DTDs include other files
- Major DTDs are:
 - express_model.dtd supports the EXPRESS language
 - And potentially embedded descriptions
 - description.dtd supports documentation of a schema in a separate XML document
 - module.dtd supports the clauses without EXPRESS

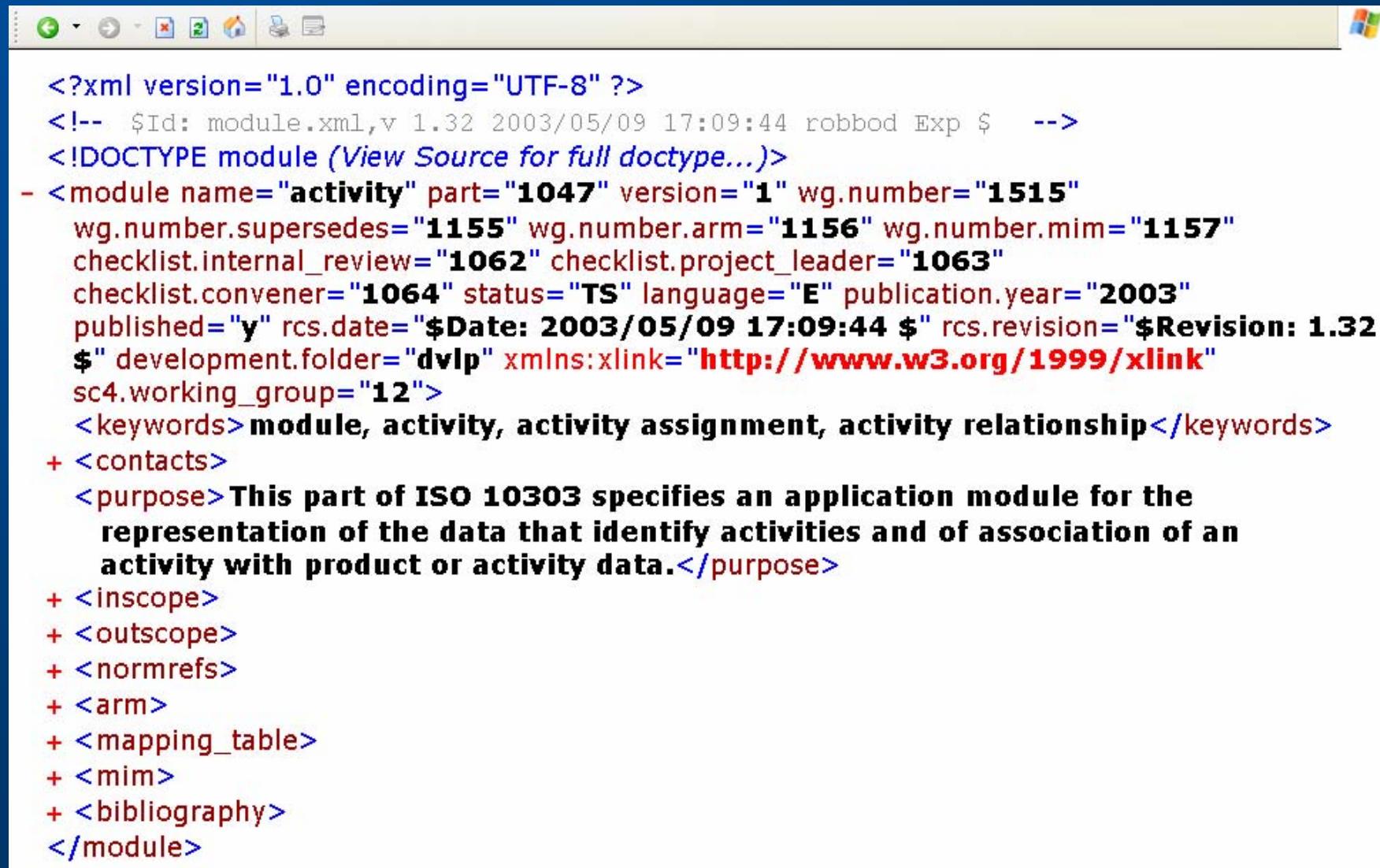
The Module Files

- ARM and MIM EXPRESS
 - arm.exp and mim.exp – The EXPRESS text itself
 - arm.xml and mim.xml – The marked up EXPRESS schema
 - This is generated from the EXPRESS schema
- Documenting the EXPRESS
 - arm_descriptions.xml and mim_descriptions.xml – The text describing the EXPRESS declarations
 - armexpn.xml and mimexpn.xml – One for each EXPRESS-G diagram (n=1,2,...)
 - Define the image map for hyperlinks here
- Documenting the remaining clauses and mapping
 - module.xml – Basically everything not driven from EXPRESS

module.xml

- Used for everything in a module that's not about a schema
- Contains technical content of
 - Cover page (contacts, keywords, WG N numbers, etc.)
 - Introduction
 - Scope
 - New normative references
 - New terms and abbreviations
 - ARM Application Object in arm.xml
 - Mapping specification
 - Reference to EXPRESS-G diagram xml documents
 - Short names
 - Bibliography

Browsing a module.xml



The screenshot shows a Windows desktop environment with a browser window open. The browser has a standard toolbar at the top. The main content area displays an XML document titled 'module.xml'. The XML code is color-coded for readability, with tags in blue and attribute values in red. The document describes a module named 'activity' with various attributes and nested elements.

```
<?xml version="1.0" encoding="UTF-8"?>
<!-- $Id: module.xml,v 1.32 2003/05/09 17:09:44 robbod Exp $ -->
<!DOCTYPE module (View Source for full doctype...)>
- <module name="activity" part="1047" version="1" wg.number="1515"
  wg.number.supersedes="1155" wg.number.arm="1156" wg.number.mim="1157"
  checklist.internal_review="1062" checklist.project_leader="1063"
  checklist.convener="1064" status="TS" language="E" publication.year="2003"
  published="y" rcs.date="$Date: 2003/05/09 17:09:44 $" rcs.revision="$Revision: 1.32
$ development.folder="dvlp" xmlns:xlink="http://www.w3.org/1999/xlink"
  sc4.working_group="12">
  <keywords>module, activity, activity assignment, activity relationship</keywords>
+ <contacts>
  <purpose>This part of ISO 10303 specifies an application module for the
    representation of the data that identify activities and of association of an
    activity with product or activity data.</purpose>
+ <inscope>
+ <outscope>
+ <normrefs>
+ <arm>
+ <mapping_table>
+ <mim>
+ <bibliography>
</module>
```

module.xml elements (imported)

keywords

– comma separated list of keywords

abstract?

– optional text

contacts

– project leader and editor

purpose

– text added to Introduction

inScope

– list items that are in-scope

outScope?

– optional list items that are out-of-scope

normrefs?

– optional new normative references

definition*

– optional definition set

abbreviations?

– optional abbreviations

arm

– Application Object refs, EXPRESS-G

arm_if?

– optional arm long form EXPRESS-G

mapping_table?

– optional mapping specification (see later slide)

mim

– MIM EXPRESS-G and shortnames

mim_if?

– optional mim long form EXPRESS-G

usage_guide?

– optional usage guide informative annex

bibliography?

– optional additional bibliography entries

text.ent markup elements

table and th – table and table header

tr and td – table row and table data

note and example – labelled note and labelled example

screen – computer data screen (as-is)

figure – labelled figure

title – title of figure

img – image for figure

img.area – hyperlinked area of image

imgfile – image for new browser window

imgfile.content – root of file referenced by imgfile

text.ent markup elements (2)

p - paragraph

b and i – bold and italics

sub and sup – subscript and superscript

ul and ol - – unordered and ordered list

li – list item

dl – definition list

dt and dd – definition term and definition

express_ref – reference from text to EXPRESS declaration

module_ref – reference from text to an AM

Address XML D:\rbn\projects\nist_module_repo\stepmod\data\modules\interface\module.xml

```
<project owner="pros.project" />
<editor ref="timking" />
</contacts>
- <purpose>
- <p>
```

This part of ISO 10303 specifies an application model for defining interfaces between products (see [Figure 1](#)). An interface is the interaction between co-functioning products, or a product and its environment, and, in order to ensure repeatability, is the subject of an EXPRESS specification. The specification defines the product attributes that are necessary to exist at a common boundary.

```
</p>
- <example number="1">
  Some product attributes:
  - <ul>
    <li>fuel flow rate;</li>
    <li>required speed;</li>
    <li>operating temperature;</li>
    <li>physical shape;</li>
    <li>dimensional tolerance.</li>
  </ul>
</example>
- <p>
```

In some industry sectors, an EXPRESS specification is referred to as an Interface Control Document or Drawing.

Link to Figure

Link to EXPRESS

Examples need numbers

```
<li>optical interaction, such as an infrared connection between a remote control  
and a television;</li>  
<li>electrical interaction, such as the current at 240 volts, 50 hertz from a domestic  
supply as required by a refrigerator.</li>
```

```
</ul>
```

```
</example>
```

```
- <p>
```

The interaction, as defined by the

```
<express_ref linkend="interface:arm:Interface_arm.Interface_specification" />
```

, is through at least one

```
<express_ref linkend="interface:arm:Interface_arm.Interface_connector" />
```

```
.
```

```
</p>
```

```
- <p>
```

An

```
<express_ref linkend="interface:arm:Interface_arm.Interface_connector" />
```

is a

```
<express_ref linkend="product_identification:arm:Product_identification_arm.Product" />
```

and, thus, may have the attributes that are necessary to ensure the viability of the
interface.

```
</p>
```

```
- <figure id="f1">
```

```
<title>A typical interface</title>
```

```

```

```
</figure>
```

```
</purpose>
```

```
- <inscope>
```

```
<li>the identification of the interface specification:</li>
```

Figure

Image

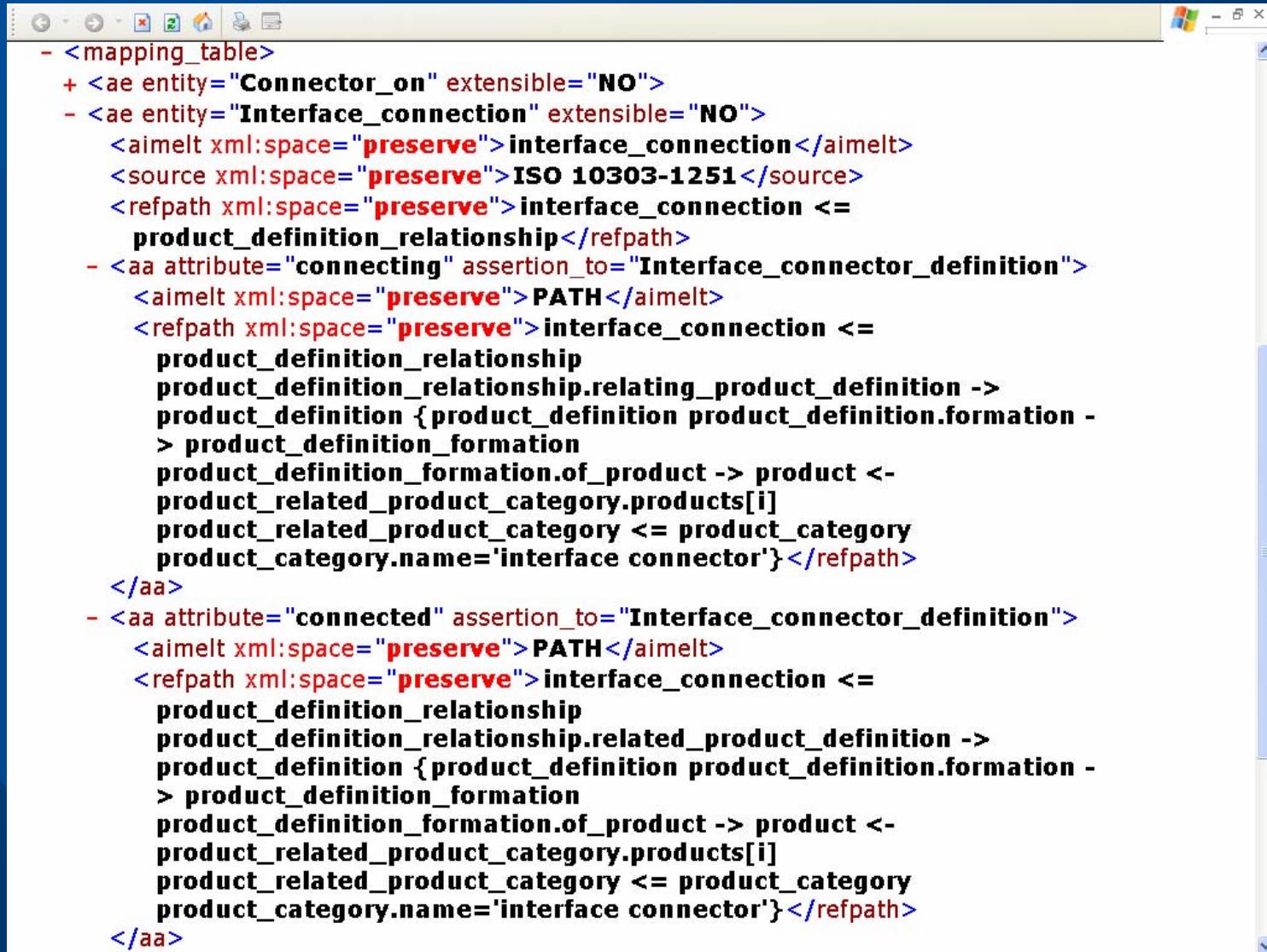
mapping.ent elements

- mapping_table (ae+,sc*)
- ae = application entity
 - contents (alt?, description?, aimelt?, source?, rules?, express_ref*, refpath?, refpath_extend?, alt_map*, aa*)
 - attributes entity, extensible, original_module
- aa = application attribute
 - contents (alt?, description?, aimelt?, source?, rules?, express_ref*, refpath?, refpath_extend?, alt_map*)
 - attributes attribute, assertion_to, inherited_from_module, inherited_from_entity
- sc = subtype constraint
 - contents (description?,(constraint|rules),source) | alt_scmap+)
 - attributes constraint, entity and original_module

mapping element details

- alt? – don't use, use alt_map
- description? – it's a description
- aimelt? – AIM element
- source? – ISO part number
- rules? – rules for mapping subtype_constraint
- ~~express_ref* - reference to an EXPRESS declaration~~
- refpath? – reference path as text, not broken down
- refpath_extend? – reference path copied and extended from other module
- alt_scmap – alternative mapping for subtype_constraint
- constraint – constraint for mapping subtype_constraint

Browsing a mapping table



The screenshot shows a Windows application window with a title bar and a scroll bar on the right. The main content area displays an XML document structure. The XML code is as follows:

```
- <mapping_table>
  + <ae entity="Connector_on" extensible="NO">
  - <ae entity="Interface_connection" extensible="NO">
    <aimelt xml:space="preserve">interface_connection</aimelt>
    <source xml:space="preserve">ISO 10303-1251</source>
    <refpath xml:space="preserve">interface_connection <=
      product_definition_relationship</refpath>
  - <aa attribute="connecting" assertion_to="Interface_connector_definition">
    <aimelt xml:space="preserve">PATH</aimelt>
    <refpath xml:space="preserve">interface_connection <=
      product_definition_relationship
      product_definition_relationship.relating_product_definition ->
      product_definition {product_definition product_definition.formation -
        > product_definition_formation
        product_definition_formation.of_product -> product <-
          product_related_product_category.products[i]
          product_related_product_category <= product_category
          product_category.name='interface connector'}</refpath>
    </aa>
  - <aa attribute="connected" assertion_to="Interface_connector_definition">
    <aimelt xml:space="preserve">PATH</aimelt>
    <refpath xml:space="preserve">interface_connection <=
      product_definition_relationship
      product_definition_relationship.related_product_definition ->
      product_definition {product_definition product_definition.formation -
        > product_definition_formation
        product_definition_formation.of_product -> product <-
          product_related_product_category.products[i]
          product_related_product_category <= product_category
          product_category.name='interface connector'}</refpath>
    </aa>
```

contacts

- contact – container for a person
 - attributes: id
 - content: firstname, lastname, affiliation?, pobox?, street?, city, state?, postcode, country, phone, fax, email
- contacts (projlead, editor+)
- projlead – project leader
 - attributes : ref which points to a contact.id
 - no content
- editor – project editor
 - attributes : ref which points to a contact.id
 - no content

Contacts

```
stepmod\data\basic\contacts.xml
```

```
<contact.list>
  <contact id="pdmmmodules.editor">
    <firstname>Darla</firstname>
    <lastname>Nettles</lastname>
```

```
stepmod\data\modules\[module]\module.xml
<contacts>
  <projlead ref="pdmmmodules.projlead"/>
  <editor ref="pdmmmodules.editor"/>
</contacts>
```

Normative References

- normrefs – the set of normative references in a module
 - content: (normref*, normref.inc*)
- normref – defining a new normative reference in your module
 - content: (stdref, term*)
 - attributes: url, id
- <!-- normative reference inclusion -->
- normref.inc – including an already defined normative reference
 - content: term.ref*
 - attributes: normref (the id of a normative reference to be included),
 - module.name (the name of a module to be included as a normative reference), resource.name (the name of an integrated resource to be included as a normative reference)

A standard reference

- stdref
 - attribute: published (y | n) – has it been published?
 - content: (#PCDATA | orgname | pubdate | stdnumber | subtitle | stdtitle | ulink)*

Normrefs

stepmod\data\basic\normrefs.xml

```
<normref id="ref10303-41.2000">
  <stdref>
    <orgname>ISO</orgname>
    <pubdate>2000</pubdate>
    <stdnumber>10303-41</stdnumber>
    <stdtitle>Industrial automation sys.....</stdtitle>
    <subtitle>&mdash; Part 41: ..... </subtitle>
  </stdref>
</normref>
```

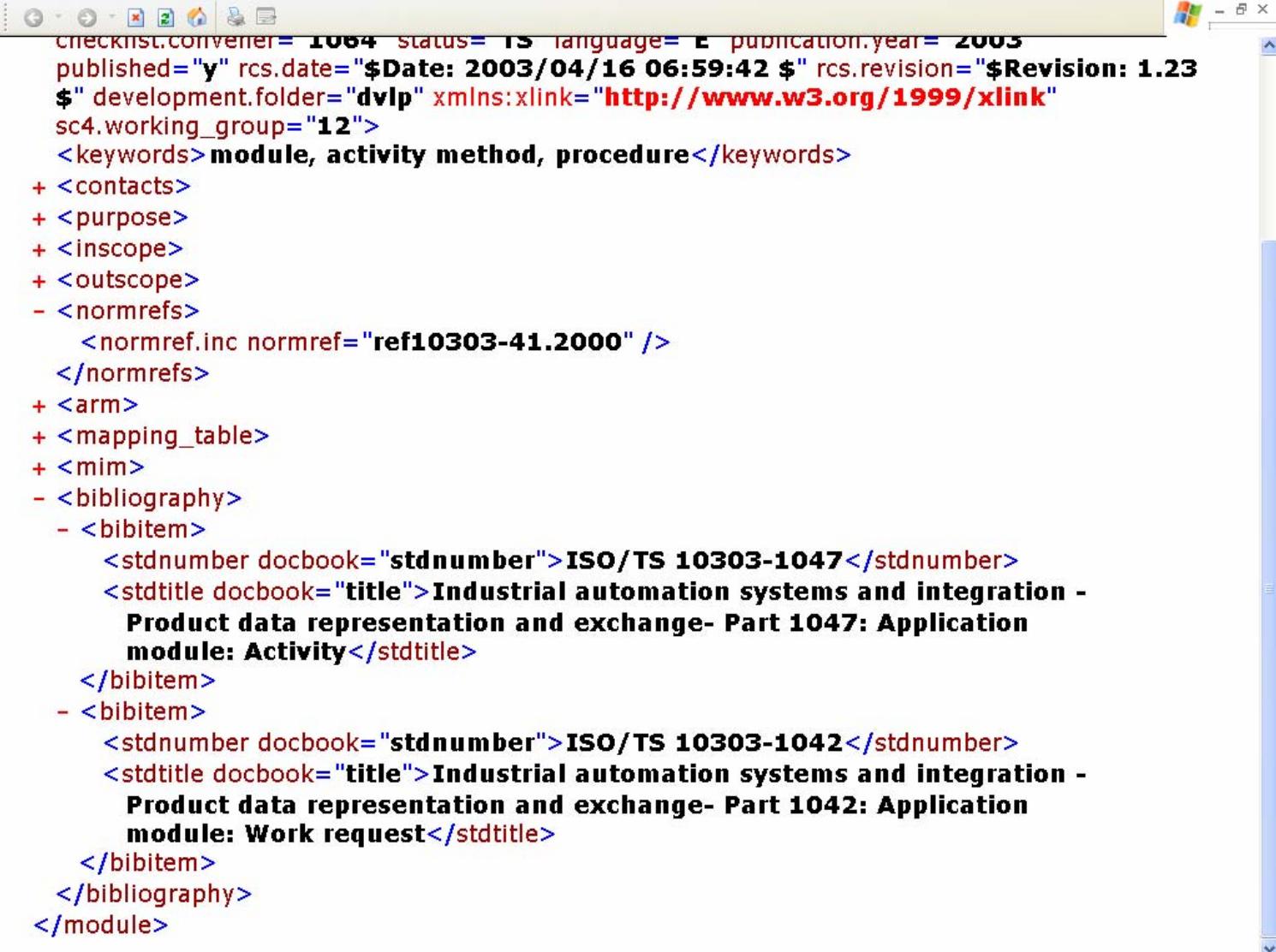
stepmod\data\modules\[module]\module.xml

```
<normrefs>
  <normref.inc normref="ref10303-41.2000"/>
</normrefs>
```

Bibliography

- **bibliography**
 - content: (bibitem.inc | bibitem)+
- **bibitem** – an item in the bibliography
 - same content as stdref
 - attribute: id (an identifier for the bibliographic item. It should be unique to the repository)
- **bibitem.list** - A list of bibitems that can be included in module.xml
 - content: (bibitem+)
 - no attributes
- **bibitem.inc** – included bibliographic item
 - no content
 - attribute: ref

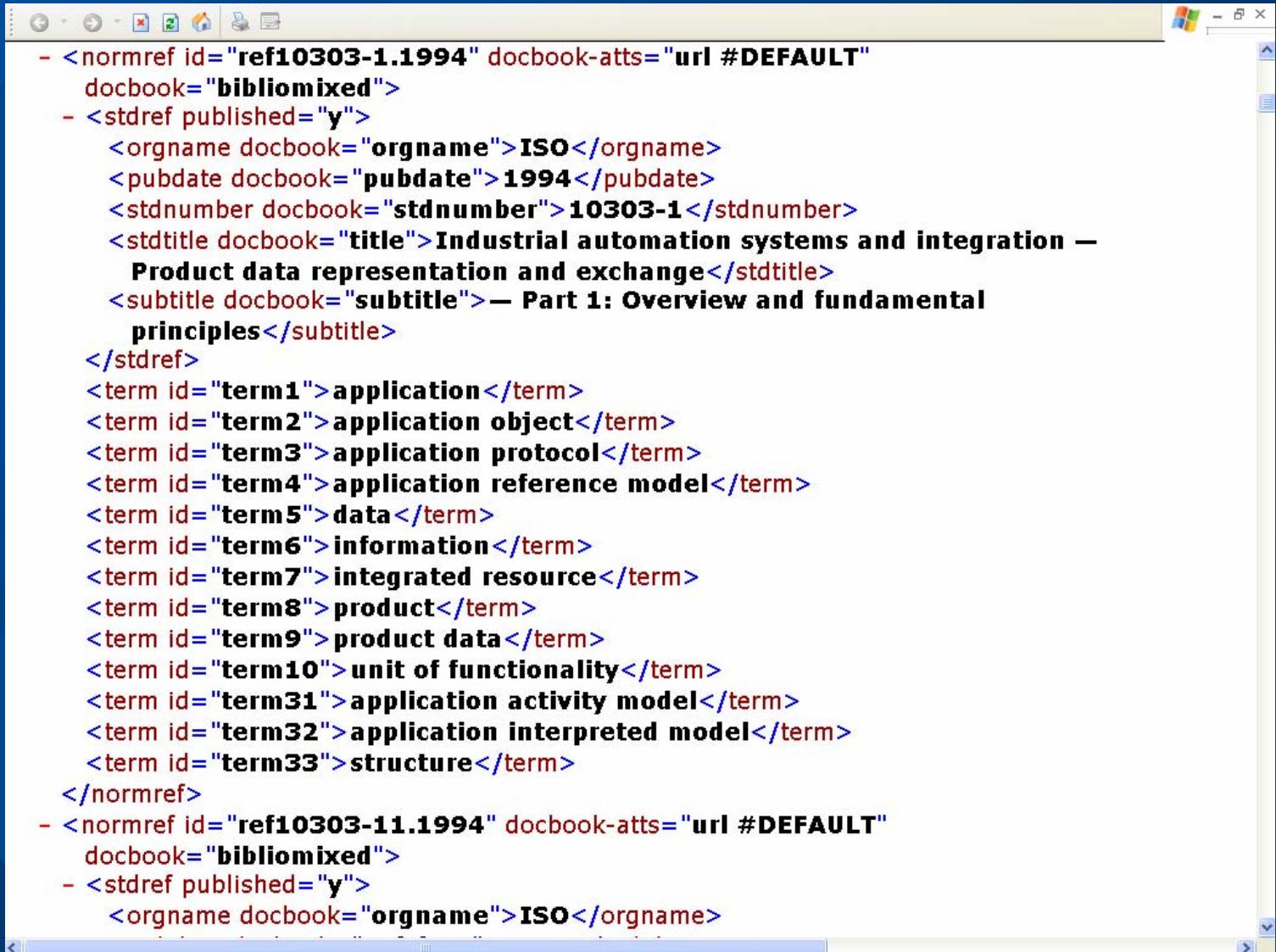
Example normref and bibliography



The screenshot shows a Windows Notepad window displaying XML code. The code includes metadata at the top and a hierarchical structure of elements. It features several normative references (normrefs) pointing to ISO/TS 10303-1047 and ISO/TS 10303-1042, and a bibliography section listing these standards with their titles and subtitles.

```
checklist.converter= 1084 status= TS language= E publication.year= 2003
published="y" rcs.date="$Date: 2003/04/16 06:59:42 $" rcs.revision="$Revision: 1.23 "
$" development.folder="dvlp" xmlns:xlink="http://www.w3.org/1999/xlink"
sc4.working_group="12">
<keywords>module, activity method, procedure</keywords>
+ <contacts>
+ <purpose>
+ <inscope>
+ <outscope>
- <normrefs>
    <normref.inc normref="ref10303-41.2000" />
</normrefs>
+ <arm>
+ <mapping_table>
+ <mim>
- <bibliography>
- <bibitem>
    <stdnumber docbook="stdnumber">ISO/TS 10303-1047</stdnumber>
    <stdtitle docbook="title">Industrial automation systems and integration -
        Product data representation and exchange- Part 1047: Application
        module: Activity</stdtitle>
</bibitem>
- <bibitem>
    <stdnumber docbook="stdnumber">ISO/TS 10303-1042</stdnumber>
    <stdtitle docbook="title">Industrial automation systems and integration -
        Product data representation and exchange- Part 1042: Application
        module: Work request</stdtitle>
</bibitem>
</bibliography>
</module>
```

Example stdref – Part 1



The screenshot shows a Windows Notepad window displaying XML code. The code defines two normative references, each with a normref element. The first normref (ref10303-1.1994) describes a standard document from ISO published in 1994, titled 'Industrial automation systems and integration — Product data representation and exchange', with a subtitle '— Part 1: Overview and fundamental principles'. It lists various terms such as application, application object, application protocol, etc. The second normref (ref10303-11.1994) also describes a standard document from ISO published in 1994, with a similar structure.

```
- <normref id="ref10303-1.1994" docbook-atts="url #DEFAULT"
  docbook="bibliomixed">
  - <stdref published="y">
    <orgname docbook="orgname">ISO</orgname>
    <pubdate docbook="pubdate">1994</pubdate>
    <stdnumber docbook="stdnumber">10303-1</stdnumber>
    <stdtitle docbook="title">Industrial automation systems and integration —
      Product data representation and exchange</stdtitle>
    <subtitle docbook="subtitle">— Part 1: Overview and fundamental
      principles</subtitle>
  </stdref>
  <term id="term1">application</term>
  <term id="term2">application object</term>
  <term id="term3">application protocol</term>
  <term id="term4">application reference model</term>
  <term id="term5">data</term>
  <term id="term6">information</term>
  <term id="term7">integrated resource</term>
  <term id="term8">product</term>
  <term id="term9">product data</term>
  <term id="term10">unit of functionality</term>
  <term id="term31">application activity model</term>
  <term id="term32">application interpreted model</term>
  <term id="term33">structure</term>
</normref>
- <normref id="ref10303-11.1994" docbook-atts="url #DEFAULT"
  docbook="bibliomixed">
  - <stdref published="y">
    <orgname docbook="orgname">ISO</orgname>
```

Bibliography

stepmod\data\basic\bibliography.xml

```
<bibitem id="ISO10303-1070">
<pubdate>2003</pubdate>
<stdnumber>ISO/CD-TS 10303-1070</stdnumber>
<stdtitle>
    Industrial automation systems and integration &mdash;
    Product data representation and exchange &mdash;
    Part 1070: Application module: Class
</stdtitle>
</bibitem>
```

stepmod\data\modules\[module]\module.xml

```
<bibliography>
    <bibitem.inc ref="ISO10303-1070"/>
</bibliography>
```

Abbreviations

- abbreviations - the abbreviations used in a module
 - content: (abbreviation*, abbreviation.inc*)+
 - no attributes
- abbreviation.inc - abbreviation inclusion
 - attribute abbreviation.inc
 - no content
- abbreviation.list - A list of abbreviations that can be included
 - content: (abbreviation+)
- abbreviation
 - content: (acronym, (term | term.ref))
 - attribute: id
- acronym – content is text
- term – content is text
 - attributes: id, linkend, url
- term.ref - a reference to the term that has been defined elsewhere: either in a normative reference, a module or an integrated resource
 - attributes: linkend, normref



```
<term.ref linkend="NCM" />
<term.ref linkend="part" />
</normref.inc>
</normrefs>
<!-- <definition/>
-->
- <abbreviations>
  - <abbreviation id="PADI">
    <acronym>PADI</acronym>
    <term>Part Definition Information</term>
  </abbreviation>
</abbreviations>
<!-- Clause 4 ARM -->
- <arm>
  <!-- Note ARM short form EXPRESS is in arm.xml -->
  <!-- Units of functionality -->
  - <!--
    <uof name="Part_Definition_Information">
      <description>
        <p>
          This UoF specifies the definitional information for the
          classification and representation of part information, documents,
          document information, and management information.
        </p>
        <p>
          No additional entities are defined in this UoF.
        </p>
      </description>
    </uof>
  </!-->
```

The Module Files

- ARM and MIM EXPRESS
 - arm.exp and mim.exp – The EXPRESS text itself
 - arm.xml and mim.xml – The marked up EXPRESS schema
 - This is generated from the EXPRESS schema
- Documenting the EXPRESS
 - arm_descriptions.xml and mim_descriptions.xml – The text describing the EXPRESS declarations
 - armexpn.xml and mimexpn.xml – One for each EXPRESS-G diagram (n=1,2,...)
 - Define the image map for hyperlinks here
- Documenting the remaining clauses and mapping
 - module.xml – Basically everything not driven from EXPRESS

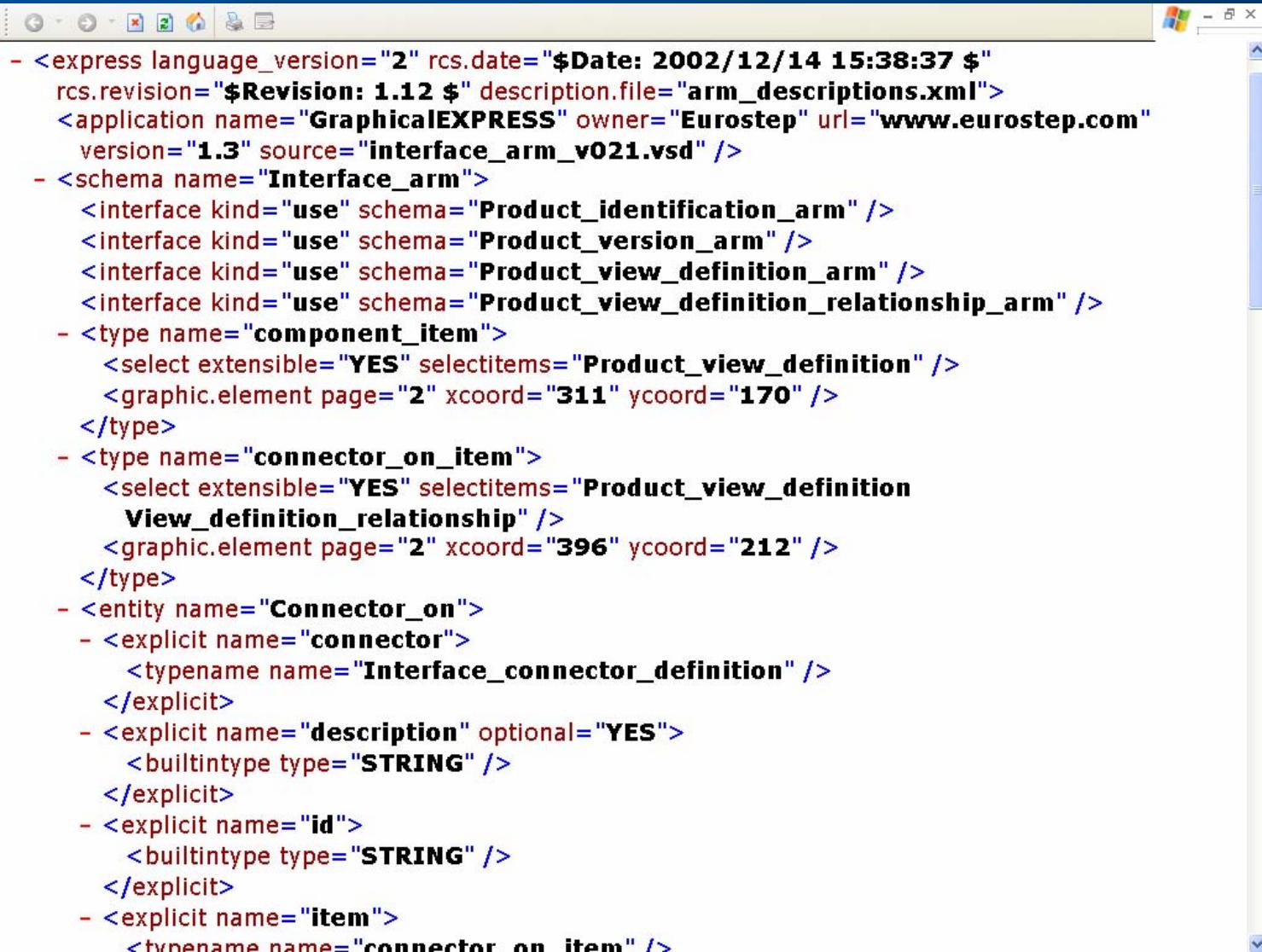
express_model.dtd

- Main elements (remember, these get generated for you)
 - express – root that contains one or more schema
 - schema – contains the other elements
 - interface – use from and reference from
 - constant – just what it says
 - entity – entity data type, contains attributes
 - type – defined type (including select and enum)
 - subtype.constraint – new in EXPRESS 2
 - function – just what it says (algorithm not exploded!)
 - procedure – just what it says (algorithm not exploded!)
 - rule – global rule (algorithm not exploded!)

Repository DTD is not Part 28

- For reasons of simplicity, the express_model.dtd does not use the Part 28 EXPRESS schema DTD directly
- Many of the elements are the same or similar
- The key difference is that the algorithms for expressions are text in the repository, they are broken down into bits in Part 28
 - This greatly simplifies writing stylesheets to present EXPRESS as text

EXPRESS as XML

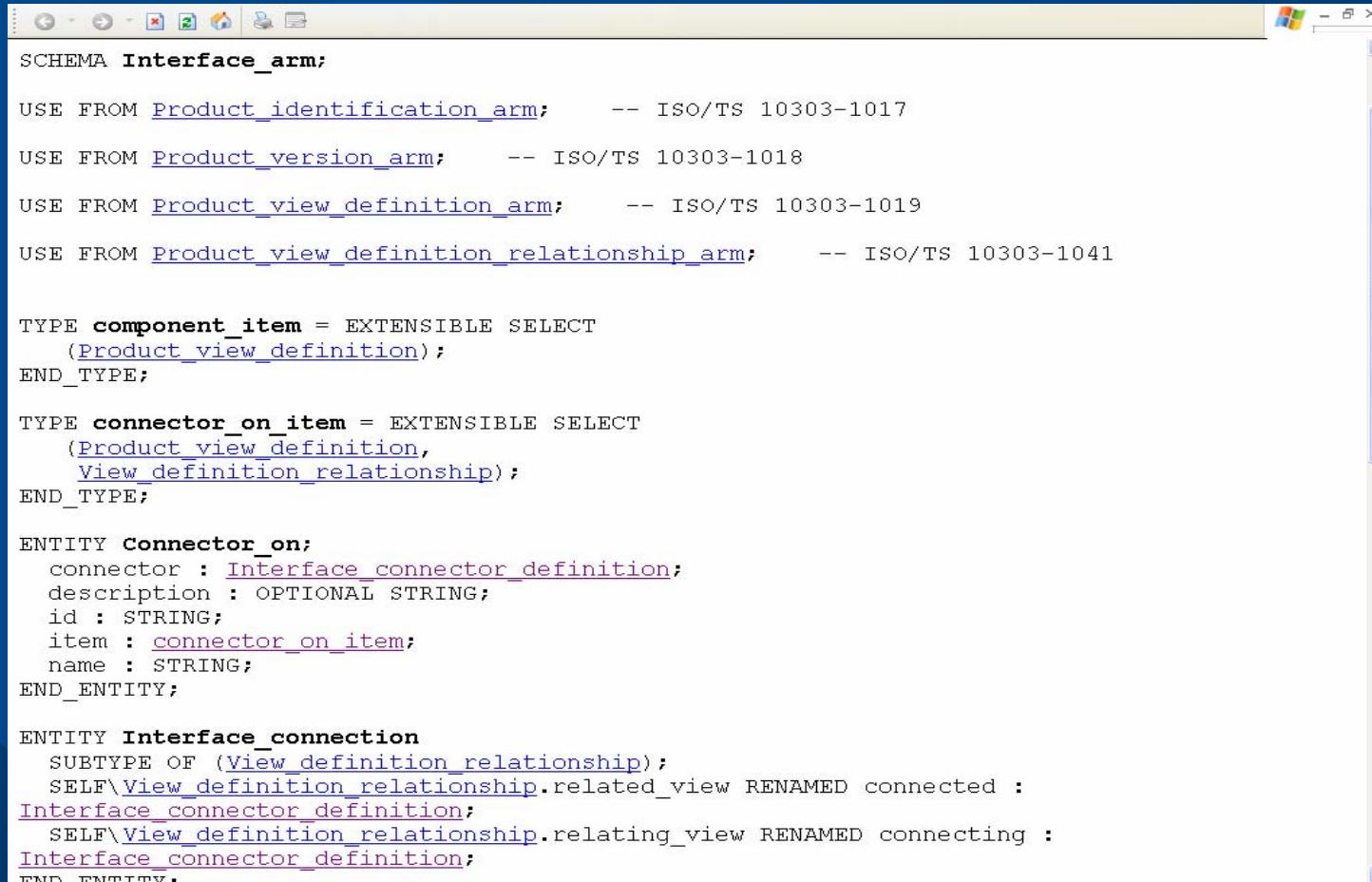


The screenshot shows a Windows application window with a title bar containing standard icons. The main area displays an XML document with syntax highlighting. The XML code defines EXPRESS language elements such as schema, type, and entity definitions.

```
<express language_version="2" rcs.date="$Date: 2002/12/14 15:38:37 $" rcs.revision="$Revision: 1.12 $" description.file="arm_descriptions.xml">
<application name="GraphicalEXPRESS" owner="Eurostep" url="www.eurostep.com" version="1.3" source="interface_arm_v021.vsd" />
- <schema name="Interface_arm">
  <interface kind="use" schema="Product_identification_arm" />
  <interface kind="use" schema="Product_version_arm" />
  <interface kind="use" schema="Product_view_definition_arm" />
  <interface kind="use" schema="Product_view_definition_relationship_arm" />
- <type name="component_item">
  <select extensible="YES" selectitems="Product_view_definition" />
  <graphic.element page="2" xcoord="311" ycoord="170" />
</type>
- <type name="connector_on_item">
  <select extensible="YES" selectitems="Product_view_definition View_definition_relationship" />
  <graphic.element page="2" xcoord="396" ycoord="212" />
</type>
- <entity name="Connector_on">
  - <explicit name="connector">
    <typename name="Interface_connector_definition" />
  </explicit>
  - <explicit name="description" optional="YES">
    <builtintype type="STRING" />
  </explicit>
  - <explicit name="id">
    <builtintype type="STRING" />
  </explicit>
  - <explicit name="item">
    <typename name="connector_on_item" />
```

EXPRESS in the repository

- The EXPRESS XML is presented as text by a stylesheet



A screenshot of a Windows application window displaying EXPRESS schema code. The window has a standard Windows title bar with icons for minimize, maximize, and close. The main area contains the following EXPRESS code:

```
SCHEMA Interface_arm;

USE FROM Product_identification_arm;      -- ISO/TS 10303-1017
USE FROM Product_version_arm;      -- ISO/TS 10303-1018
USE FROM Product_view_definition_arm;      -- ISO/TS 10303-1019
USE FROM Product_view_definition_relationship_arm;      -- ISO/TS 10303-1041

TYPE component_item = EXTENSIBLE SELECT
  (Product_view_definition);
END_TYPE;

TYPE connector_on_item = EXTENSIBLE SELECT
  (Product_view_definition,
   View_definition_relationship);
END_TYPE;

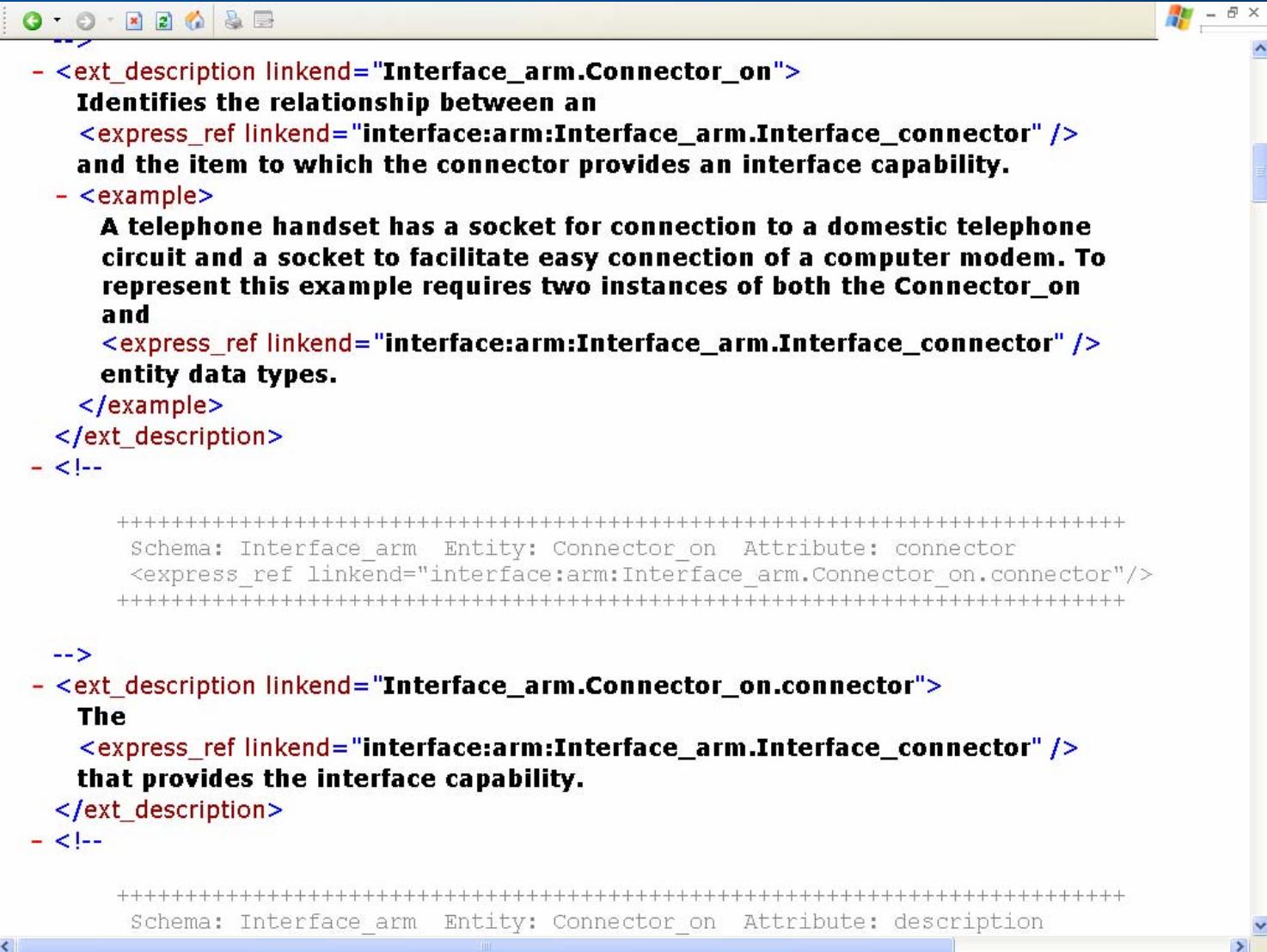
ENTITY Connector_on;
  connector : Interface_connector_definition;
  description : OPTIONAL STRING;
  id : STRING;
  item : connector_on_item;
  name : STRING;
END_ENTITY;

ENTITY Interface_connection
  SUBTYPE OF (View_definition_relationship);
  SELF\View_definition_relationship.related_view RENAMED connected :
  Interface_connector_definition;
  SELF\View_definition_relationship.relating_view RENAMED connecting :
  Interface_connector_definition;
END_ENTITY.
```

external descriptions

- description in one file references EXPRESS in another
 - arm_description.xml and mim_description.xml
- ext_descriptions
 - attributes: module_directory, schema_file, rcs.date, rcs.revision, describe.selects, describe.subtype_constraints
 - content: ext_description*
- ext_description
 - attributes: linkend
 - content: all the text-related elements from earlier

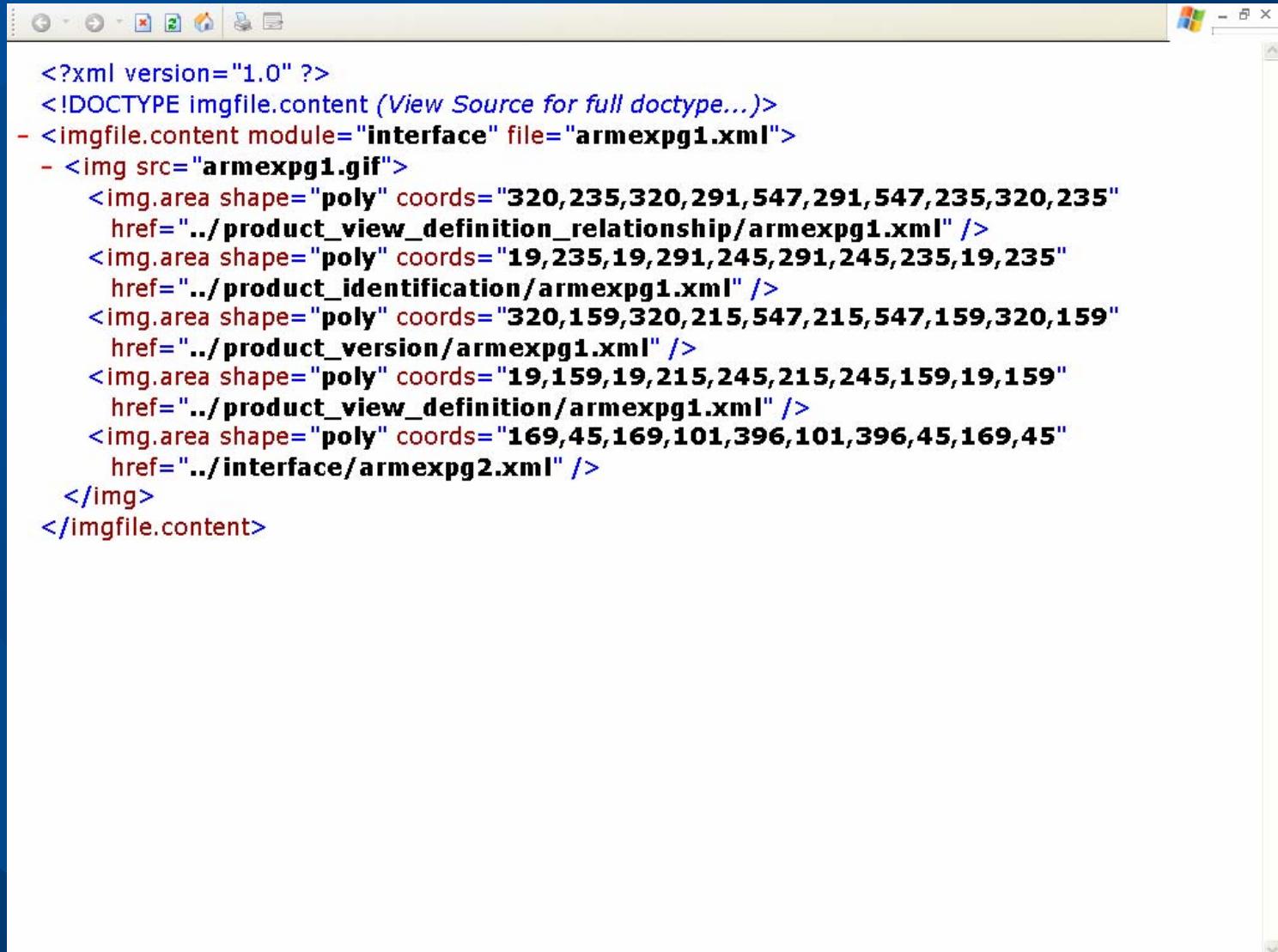
An example external description



The screenshot shows a Windows application window with a title bar and various icons. The main content area displays XML code with some text annotations in bold black font. The XML code includes several comments and examples related to interface connectors.

```
- <ext_description linkend="Interface_arm.Connector_on">
  Identifies the relationship between an
  <express_ref linkend="interface:arm:Interface_arm.Interface_connector" />
  and the item to which the connector provides an interface capability.
- <example>
  A telephone handset has a socket for connection to a domestic telephone
  circuit and a socket to facilitate easy connection of a computer modem. To
  represent this example requires two instances of both the Connector_on
  and
  <express_ref linkend="interface:arm:Interface_arm.Interface_connector" />
  entity data types.
</example>
</ext_description>
- <!--
+++++
Schema: Interface_arm Entity: Connector_on Attribute: connector
<express_ref linkend="interface:arm:Interface_arm.Connector_on.connector"/>
+++++
-->
- <ext_description linkend="Interface_arm.Connector_on.connector">
  The
  <express_ref linkend="interface:arm:Interface_arm.Interface_connector" />
  that provides the interface capability.
</ext_description>
- <!--
+++++
Schema: Interface_arm Entity: Connector_on Attribute: description
-->
```

EXPRESS-G in XML



```
<?xml version="1.0" ?>
<!DOCTYPE imgfile.content (View Source for full doctype...)>
- <imgfile.content module="interface" file="armexpg1.xml">
- 
  <img.area shape="poly" coords="320,235,320,291,547,291,547,235,320,235"
    href="../product_view_definition_relationship/armexpg1.xml" />
  <img.area shape="poly" coords="19,235,19,291,245,291,245,235,19,235"
    href="../product_identification/armexpg1.xml" />
  <img.area shape="poly" coords="320,159,320,215,547,215,547,159,320,159"
    href="../product_version/armexpg1.xml" />
  <img.area shape="poly" coords="19,159,19,215,245,215,245,159,19,159"
    href="../product_view_definition/armexpg1.xml" />
  <img.area shape="poly" coords="169,45,169,101,396,101,396,45,169,45"
    href="../interface/armexpg2.xml" />
</img>
</imgfile.content>
```



Building the STEP Module Content Section Agenda

1. XML Basics
2. Module document structures
3. Building your first module
4. Repository tools used to build a module

Simple few steps

1. Create a module template
 1. Add purpose, scope,
2. Create the ARM EXPRESS/EXPRESS-G
 1. Create the EXPRESS
 2. Convert to XML
 3. Place in module
3. Create the MIM EXPRESS/EXPRESS-G
4. Add the mapping
5. Add contacts
6. Add normrefs
7. Add bibliography
8. Add WG numbers

First some rules

Rule: Modules

- Use all of a module
- Avoid defining EXPRESS rules in lower level modules
 - it prevents reuse
- Try to separate assignment from the entity being assigned
 - other users may not want to use the assignment
- Avoid defining EXPRESS entities in implementation level modules

Rule: Naming

- All files in lower case
 - ☺ – arm.xml OK
 - ☹ – Arm.XML Not OK
 - as SourceForge is a UNIX system, upper case filenames will cause problems
- All module names in lower case with no spaces
 - ☺ – product_as_individual OK
 - ☹ – Product as individual Not OK

More rules: Schemas

- ARM schemas end in "_arm".
- MIM schemas end in "_mim".
- Schema name shall start with an uppercase letter.
 - activity module ARM schema is: Activity_arm
 - activity module MIM is: Activity_mim
- If you use a module – you use all of it
 - 😊 – USE FROM Activity;
 - 😢 – USE FROM Activity
(Activity);

More rules: EXPRESS Objects

- ARM entities start with Upper case.
 - E.g Activity
- MIM entities are lower case
 - E.g. action
- All types, in ARMs and MIMs, are lower case.
 - E.g. activity_item

More rules

- Avoid using the same name for an entity as someone else as it prevents reuse
- Use the index of ARM objects to check.
- Make sure your XML is well formed and valid

Now build a module



Simple few steps

1. Create a module template
 1. Add purpose, scope,
2. Create the ARM EXPRESS/EXPRESS-G
 1. Create the EXPRESS
 2. Convert to XML
 3. Place in module
3. Create the MIM EXPRESS/EXPRESS-G
4. Add the mapping
5. Add contacts
6. Add normrefs
7. Add bibliography
8. Add WG numbers

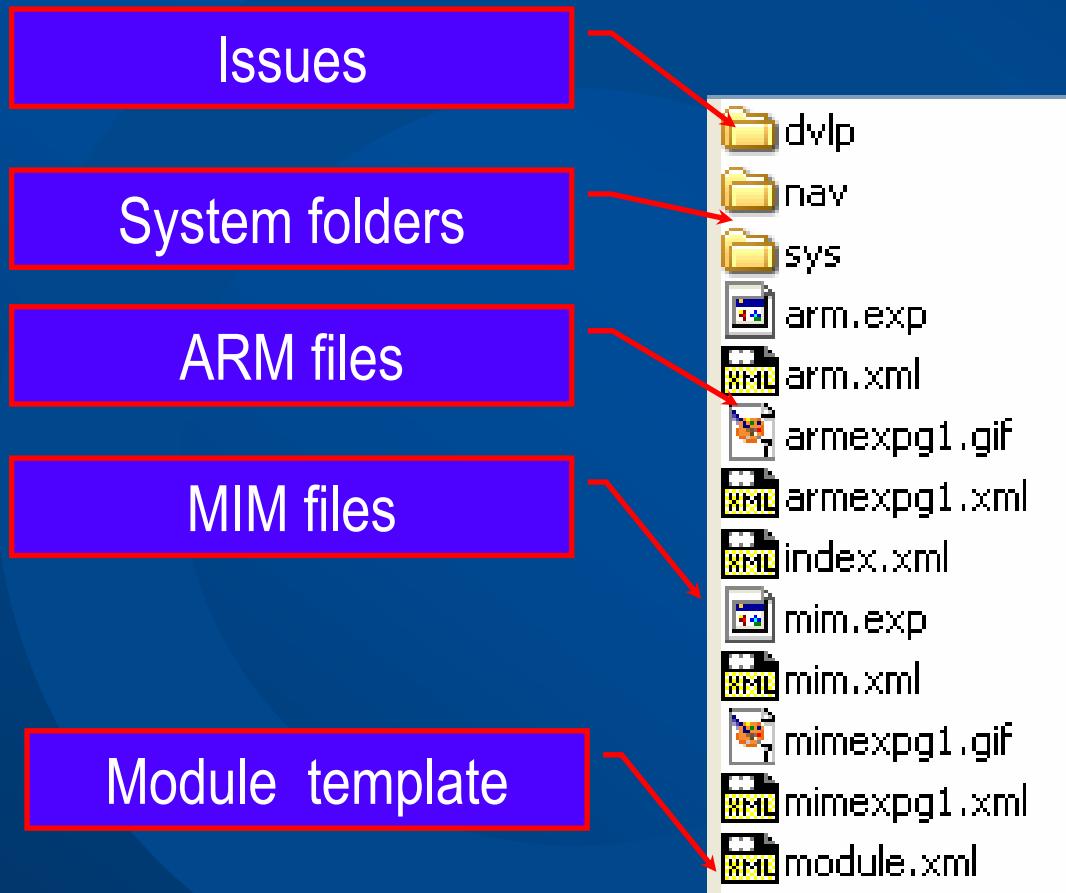
Prequisites

- The utility functions are JScrips and the assumption is made that JScript is installed on the machine. The later versions of windows have this installed be default. To test run a DOS command (Start->run>command) and then cscript.
- If CScript is not present on the machine, it can be down loaded from Microsoft:
<http://msdn.microsoft.com/downloads/default.aspx?url=/downloads/topic.asp?url=/msdn-files/028/001/175/topic.xml>

Make your module template

- Create the module
 - Run: *stepmod|utils|mkmodule.wsf*
 - creates a new module. It will generate:
 - the module directory,
 - skeleton XML files for the module,
 - the SYS files.
 - Edit: *stepmod|data|modules|[module]|module.xml*

Module



Add the module to repository

- Add the module to the index:
 - Edit: *stepmod|repository_index.xml*
<module name="activity" part="1047" project="PDM modules" status="ballot-resolution"/>

Demo mkmodule



- Run mkmodule
- Update repository_index.xml
- Add purpose, scope

Simple few steps

1. Create a module template
 1. Add purpose, scope,
2. Create the ARM EXPRESS/EXPRESS-G
 1. Create the EXPRESS
 2. Convert to XML
 3. Place in module
3. Create the MIM EXPRESS/EXPRESS-G
4. Add the mapping
5. Add contacts
6. Add normrefs
7. Add bibliography
8. Add WG numbers
9. Add short names

Create your ARM and MIM EXPRESS

- EXPRESS/EXPRESS-G is all hyperlinked in modules
- Module EXPRESS files:
 - arm.exp, arm.xml
 - armexpg1.xml, armexpg1.gif
 - mim.exp, mim.xml
 - mimexpg1.xml, mimexpg1.gif
- A number of approaches:
 1. Hand edit the EXPRESS and XML
 - Error prone
 2. Use a tool to convert the EXPRESS to XML
 - Some tools available
 - stepmod\utils\express2xml.js (HAS BUGS!)
 - Eurostep EXPRESS Parser (eep)
 - Free from: <http://www.eurostep.com>
 - PDTEC ECCO outputs XML for modules XML

Create EXPRESS-G

Now create the EXPRESS-G

1. Draw the diagram
2. Draw an HTML image map
3. Add URL's to image map
4. Convert HTML image map to XML
 - Run: *stepmod\utils\imagemap2xml
cscript imagemap2xml.js <module>
<imagemap.html> <file.xml> <optional title>*

EXPRESS/EXPRESS-G Problems

- Hand editing will lead to errors!
- Is the EXPRESS-G the same as the EXPRESS?
- Is EXPRESS the same as the XML?
- Are the URL's in the EXPRESS correct?
- The real problems come when you start changing the EXPRESS. Everything must be in sync

GraphicalEXPRESS

- A better approach is to use a tool that creates the EXPRESS-G and derives:
 - The EXPRESS
 - The XML
- GraphicalEXPRESS
 - A VISIO plug in
 - Freely available from Eurostep
 - <http://www.eurostep.com>
 - Best with VISIO 2002

Using GraphicalEXPRESS

1. Draw your EXPRESS-G
2. Publish as XML
3. Copy to XML to module
Run: *stepmod\etc\graphicalexpress\utils\ge2module.wsf*
4. Update module.xml to reference the EXPRESS-G

```
<mim>
  <express-g>
    <imgfile file="mimexpg1.xml"/>
    <imgfile file="mimexpg2.xml"/>
  </express-g>
```

Demo GraphicalEXPRESS



- Run GraphicalEXPRESS
- Create Test_arm
- Add Course entity
- Add Approval_arm
- Extend approval_item
- Export as XML

Add EXPRESS descriptions

- Descriptions can be embedded in arm.xml

```
<entity name="Classification">
<description><p>A <b>classification</b> is a
relationship between a class and a thing, that indicates
the thing is a member of the class.</p>
<note>The meaning of this entity is identical to the
entity
<b>classification</b> defined in ISO 15926-2.</note>
</description>
```

Use a separate description file

- Can separate the descriptions from the EXPRESS
- Has the advantage that you can modify the EXPRESS and not affect the descriptions independently of the EXPRESS.
- Run: *stepmod\utils\extractDescriptions.wsf*

Use a separate description file

arm.xml

```
<express  
language_version="2"  
description.file="arm_descriptions.xml"  
rcs.date="$Date: 2002/07/30 10:44:56 $"  
rcs.revision="$Revision: 1.13 $">
```

arm_descriptions.xml

```
<ext_descriptions  
module_directory="activity" schema_file="arm.xml"  
rcs.date="$Date: 2003/06/06 20:29:47 $"  
rcs.revision="$Revision: 1.12 $"  
describe.selects="YES">  
  
<ext_description linkend="Activity_arm.Activity">  
An Activity is the identification of the occurrence  
of an activity.  
</ext_description>
```

Automated select descriptions

```
<ext_descriptions  
    module_directory="activity" schema_file="arm.xml"  
    rcs.date="$Date: 2003/06/06 20:29:47 $"  
    rcs.revision="$Revision: 1.12 $"  
describe.selects="YES">
```

- No description necessary
- XSL does the work

E.g. Empty extensible select

The screenshot shows a Microsoft Internet Explorer window with the title "ISO/TS 10303-1012:2003 Approval - Microsoft Internet Explorer". The address bar displays the URL "file:///D:/rbn/projects/nist_module_repo/stepmod/data/modules/approval/sys/4_info_reqs.xml#approval_arm". The main content area contains the following text:

4.2.1 approval_item

The **approval_item** type is an extensible list of alternate data types. Additional alternate data types are specified in select data types that extend the **approval_item** type.

NOTE This empty extensible select requires extension in a further module to ensure that entities that refer to it have at least one valid instantiation.

EXPRESS specification:

```
*)  
TYPE approval_item = EXTENSIBLE GENERIC_ENTITY SELECT;  
END_TYPE;  
(*
```

4.3 ARM entity definitions

E.G populated extensible select

The screenshot shows a Microsoft Internet Explorer window with the title "ISO/CD-TS 10303-1288:- Management resource information - Microsoft Internet Explorer". The address bar contains the URL "file:///D:/rnb/projects/nist_module_repo/stepmod/data/modules/management_resource_information/sys/4_info_re". The main content area displays the following text:

4.2.2 mri_approval_item

The **mri_approval_item** type is an extension of the [**approval_item**](#) type. It adds the data types [Certification](#) and [Contract](#) to the list of alternate data types.

NOTE The list of entity data types may be extended in application modules that use the constructs of this module.

Using [**Approval_assignment**](#), an [**Approval**](#) can be assigned to the entity data types listed in **mri_approval_item**.

EXPRESS specification:

```
*)  
TYPE mri_approval_item = EXTENSIBLE GENERIC_ENTITY SELECT BASED_ON  
approval_item WITH  
  (Certification,  
   Contract);  
END_TYPE;  
(*
```

Demo extractDescriptions



- Run utils\extractDescriptions.wsf
- Describe Course entity

Simple few steps

1. Create a module template
 1. Add purpose, scope,
2. Create the ARM EXPRESS/EXPRESS-G
 1. Create the EXPRESS
 2. Convert to XML
 3. Place in module
3. Create the MIM EXPRESS/EXPRESS-G
4. Add the mapping
5. Add contacts
6. Add normrefs, terms, abbreviations
7. Add bibliography
8. Add WG numbers
9. Add short names

Mapping

Mapping defined in module.xml

<mapping_table>

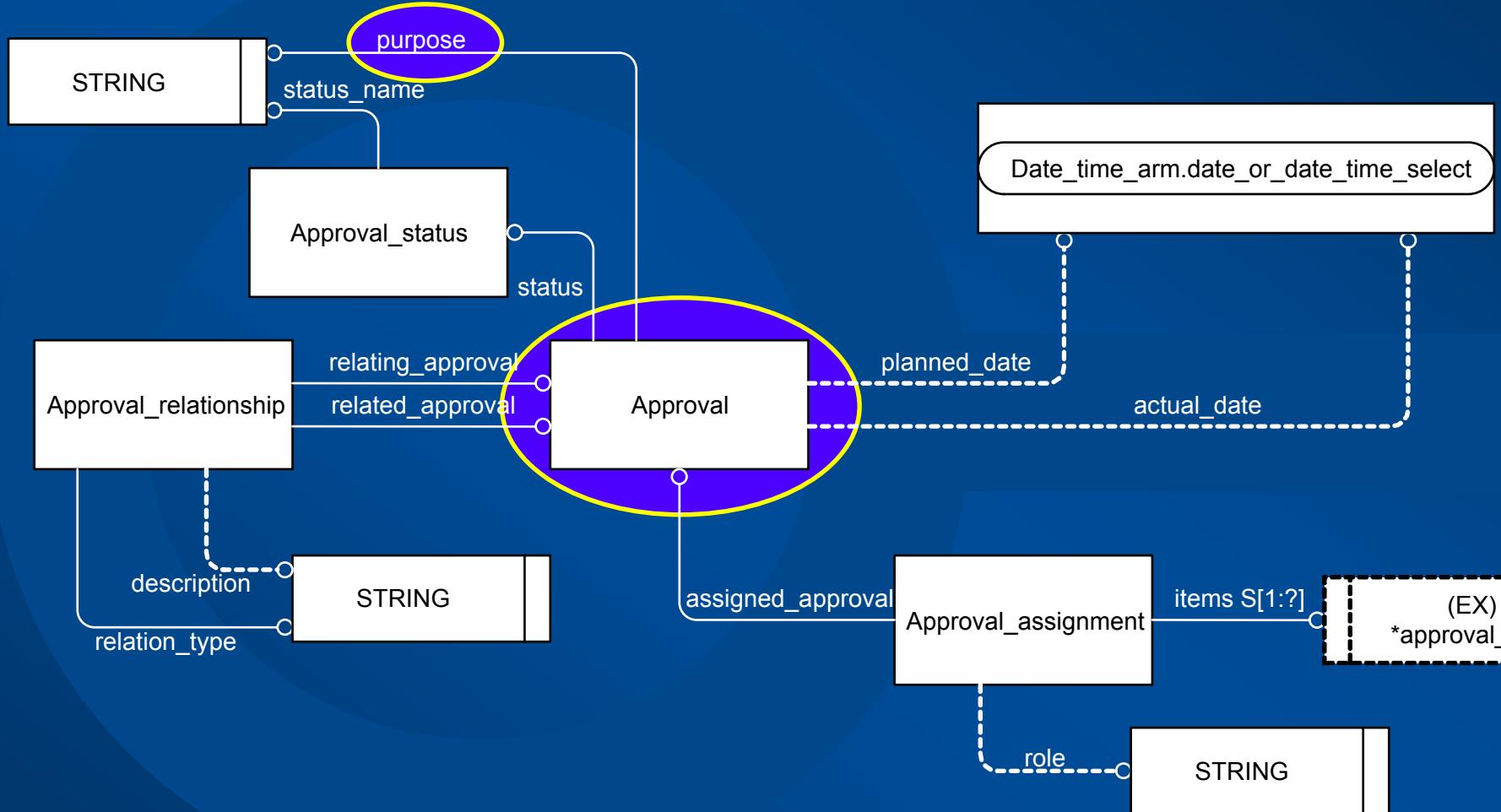
<ae entity="Course">
 <aimelt>-</aimelt>
 <source>-</source>
 <refpath>-</refpath>

ARM entity being mapped
MIM entity mapped to
Document in which MIM entity defined
Mapping reference path

<aa attribute="name">
 <aimelt>-</aimelt>
 <source>-</source>
 <refpath>-</refpath>
 </aa>
</ae>
</mapping_table>

ARM attribute (simple type being mapped)

Mapping



Mapping of Entities + attributes of simple type

```
<mapping_table>
  <ae entity="Approval">
    <aimelt>approval</aimelt>
    <source>ISO 10303-41</source>

  <aa attribute="purpose">
    <aimelt>approval.level</aimelt>
    <source>ISO 10303-41</source>
  </aa>
```

The definition and use of mapping templates is not supported in the present version of the application modules. However, use of predefined templates /SUBTYPE/ and /SUPERTYPE/ is supported.

5.1.1 Approval

MIM element: approval

Source: ISO 10303-41

5.1.1.1 Approval to Approval_status (as status)

MIM element: approval_status.name

Source: PATH

Reference path: approval.status ->
 approval_status
 approval_status.name

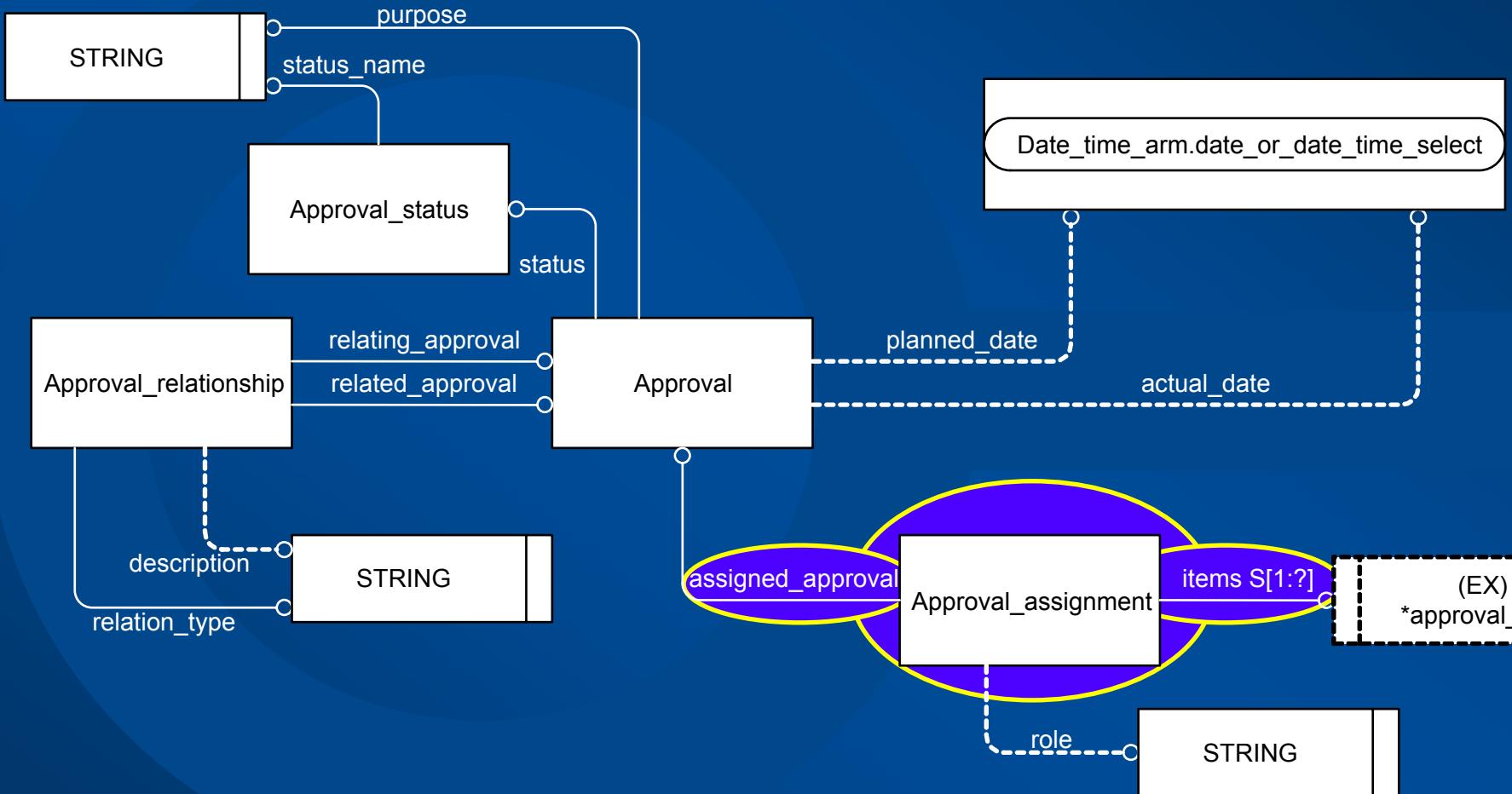
5.1.1.2 purpose

MIM element: approval.level

Source: ISO 10303-41



Mapping



Mapping of assertions

```
<ae entity="Approval_assignment">  
  <aimelt>applied_approval_assignment</aimelt>  
  <source>ISO 10303-1012</source>  
  <refpath>  
    applied_approval_assignment &lt;= approval_assignment  
  </refpath>
```

Entity mapping

< is <
> is >

```
<aa attribute="assigned_approval" assertion_to="Approval">  
  <aimelt>PATH</aimelt>  
  <refpath>  
    approval_assignment.assigned_approval -&gt; approval  
  </refpath>  
</aa>
```

Assertion to an Entity

```
<aa attribute="items" assertion_to="approval_item">  
  <aimelt>PATH</aimelt>  
  <refpath>  
    applied_approval_assignment.items[i] -&gt; approval_item  
  </refpath>  
</aa>
```

Assertion to a Select

ISO/TS 10303-1012:2003 Approval

File Edit View Favorites Tools Help

Back Favorites Media Go

D:\rbn\projects\nist_module_repo\stepmod\data\modules\approval\sys\5_mapping.htm

```
(date_time_select = date  
date => calendar_date))
```

5.1.2 [Approval_assignment](#)

MIM element: applied_approval_assignment
Source: ISO 10303-1012
Reference path: applied_approval_assignment <= approval_assignment

5.1.2.1 [Approval_assignment](#) to [Approval](#) (as [assigned_approval](#))

MIM element: PATH
Reference path: approval_assignment.assigned_approval -> approval

5.1.2.2 [Approval_assignment](#) to [approval_item](#) (as [items](#))

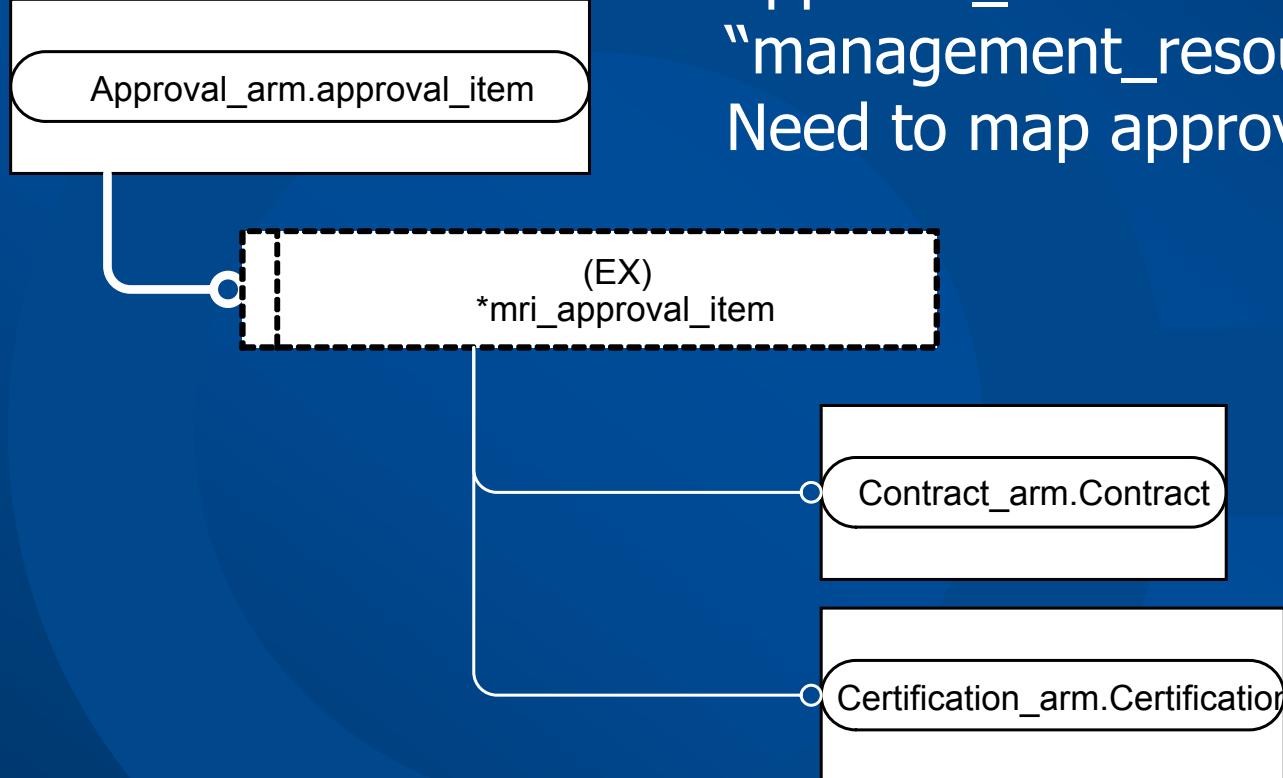
MIM element: PATH
Reference path: applied_approval_assignment.items[i] -> approval_item

5.1.2.3 [role](#)

Mapping of extensible selects

- When you extend a select, need to map any attribute whose type is the base select to all entities in select
 - E.g approval_item

Extended Select



Approval_item extended in
“management_resource_information”
Need to map approval_assignment.item

Mapping of extensible selects

```
<ae entity="Approval_assignment" original_module="approval">
  <aa attribute="items" assertion_to="Certification">
    <refpath_extend extended_select="approval_item">
      approval_item *> mri_approval_item
      mri_approval_item = certification
    </refpath_extend>
  </aa>
```

Module in which initial mapping to select defined

ARM select being extended

```
<aa attribute="items" assertion_to="Contract">
  <refpath_extend extended_select="approval_item">
    approval_item *> mri_approval_item
    mri_approval_item = contract
  </refpath_extend>
</aa>
</ae>
```

MIM select being extended

STEPmodHTML Help
Module menu

- [+] individual_product
- [+] inertia_characteris
- [+] information_rights
- [+] interface
- [+] interface_lifecycle
- [+] involvement_of_in
- [+] item_definition_st

J

- [+] justification

L

- [+] layer_assignment
- [+] location
- [+] location_assignme
- [+] location_assignme
- [+] location_in_buildin

M

- [+] management_res
- [+] manifold_surface
- [+] manufacturing_co
- [+] material_aspects
- [+] maths_space
- [+] maths_value
- [+] measure_represer
- [+] message
- [+] multi_linguism

N

- [+] name_assignmen
- [+] numeric_function
- [+] numerical_interfac
- [+] nut_and_bolt

O

- [+] observation

Modules[Alphabetical](#),
[Project](#)
[Parts, Leader,](#)
[Starting Commands](#)**Express**[ARM objects](#)
[MIM objects](#)**Application Protocols**[Alphabetical](#),
[Project](#)**Resource parts**[Alphabetical](#),
[Project](#)**Resource schemas**[Alphabetical](#),
[Mappings](#)**Ballots**[Ballots](#)**ISO/CD-TS 10303-1288:- Management resource information**

5.1.1 Approval_assignment

This application object, Approval_assignment, is defined in the module approval. This mapping section extends the mapping of Approval_assignment, to include assertions defined in this module.

5.1.1.1 Approval_assignment to Certification (as items)

Reference path: applied_approval_assignment.items[i] -> approval_item
approval_item *-> mri_approval_item
mri_approval_item = certification

5.1.1.2 Approval_assignment to Contract (as items)

Reference path: applied_approval_assignment.items[i] -> approval_item
approval_item *-> mri_approval_item
mri_approval_item = contract

My Computer

Alternate mappings

```
<ae entity="Approving_person_organization">
  <aimelt>approval_person_organization</aimelt>
  <source>ISO 10303-41</source>

  <aa attribute="approval_date" assertion_to="date_or_date_time_select">
    <alt_map id="1">
      <description>if a calendar date is specified for the date of approval sign-off</description>
      <aimelt>PATH</aimelt>
      <refpath>
        (date_item = approval_person_organization
         etc .....
      </refpath>
      </alt_map>
    <alt_map id="2">
      <description>if a date and time is specified for the date of approval sign-off</description>
      <aimelt>PATH</aimelt>
      <refpath>
        (date_and_time_item = approval_person_organization
         etc ...
      </refpath>
    </alt_map>
  </aa>
```



5.1.5.2 Approving_person_organization to date_or_date_time_select (as approval_date)

#1: if a calendar date is specified for the date of approval sign-off

MIM element: PATH

Reference path: (date_item = approval_person_organization
date_item <- applied_date_assignment.items[i]
applied_date_assignment <= date_assignment
(date_assignment.role -> date_role
date_role.name = 'sign off')
(date_assignment.assigned_date -> date
date => calendar_date))

#2: if a date and time is specified for the date of approval sign-off

MIM element: PATH

Reference path: (date_and_time_item = approval_person_organization
date_and_time_item <-
applied_date_and_time_assignment.items[i]
applied_date_and_time_assignment <=
date_and_time_assignment
(date_and_time_assignment.role ->
date_time_role
date_time_role.name = 'sign off'))

Mapping

Module **property_assignment**

```
ENTITY Assigned_property;
  described_element : property_assignment_select;
  description : OPTIONAL STRING;
  id : OPTIONAL STRING;
  name : STRING;
END_ENTITY;
```

Module document_properties in which the extensible select property_assignment_select is extended:

```
ENTITY Assigned_document_property
  SUBTYPE OF (Assigned_property) ;
  DERIVE
    SELF\Assigned_property.name : STRING := 'document property';
  UNIQUE
    UR1: SELF\Assigned_property.described_element;
  WHERE
    WR1: SIZEOF(['DOCUMENT_PROPERTIES_ARM.DOCUMENT_DEFINITION',
      'DOCUMENT_PROPERTIES_ARM.FILE']*TYPEOF(SELF\Assigned_property.described_element)) = 1;
END_ENTITY;
```

Mapping

```
<mapping_table>
  <ae entity="Assigned_document_property">
    <aimelt>property_definition</aimelt>
    <source>ISO 10303-41</source>
    <refpath> property_definition
      {property_definition.name = 'document property'}
    </refpath>
  <aa attribute="described_element"
    assertion_to="Document_definition"
    inherited_from_module="property_assignment"
    inherited_from_entity="Assigned_property">
  .....

```

Mapping creation

- Create a mapping template:
 - Run: stepmod\utils\mkmapping\mkmapping.wsf
 - Copy to module.xml
 - Populate:
 - aimelt – the MIM entity the ARM entity maps to
 - source – the source of the MIM entity (e.g. ISO 10303-41)
 - refpath – the mapping reference path

Developer views

- Use developer ARM select view to show unmapped select extensions
- Use mapping view and mapping view with test to show mapping errors

Demo Mapping



- Run
`utils\mkmapping\mkmapping.wsf`
- Add mapping to `module.xml`
- Show developer ARM select view
- Copy template to `module.xml`

Simple few steps

1. Create a module template
 1. Add purpose, scope,
2. Create the ARM EXPRESS/EXPRESS-G
 1. Create the EXPRESS
 2. Convert to XML
 3. Place in module
3. Create the MIM EXPRESS/EXPRESS-G
4. Add the mapping
5. Add contacts
6. Add normrefs, terms, abbreviations
7. Add bibliography
8. Add WG numbers

Contacts

- Contacts stored in:
stepmod\data\basic\contacts.xml
- Referenced from module.xml

```
<contacts>
    <projlead ref="pdmmmodules.projlead"/>
    <editor ref="pdmmmodules.editor"/>
</contacts>
```

stepmod\data\basic\contacts.xml

```
<contact.list>
    <contact id="pdmmmodules.editor">
        <firstname>Darla</firstname>
        <lastname>Nettles</lastname>
```

module.xml

```
<contacts>
    <projlead ref="pdmmmodules.projlead"/>
    <editor ref="pdmmmodules.editor"/>
</contacts>
```



```
-->
<!DOCTYPE contact.list (View Source for full doctype...)>
- <contact.list>
  - <contact id="pdmmmodules.editor">
    <firstname>Darla</firstname>
    <lastname>Nettles</lastname>
    <affiliation>Advanced Technology Institute / PDES, Inc.</affiliation>
    <street>5300 International Boulevard</street>
    <city>North Charleston</city>
    <state>SC</state>
    <postcode>29418</postcode>
    <country>USA</country>
    <phone>+1-843-760-3232</phone>
    <fax>+1-843-760-3349</fax>
    <email>nettles@aticorp.org</email>
  </contact>
  - <contact id="pdmmmodules.projlead">
    <firstname>Simon</firstname>
    <lastname>Frechette</lastname>
    <affiliation>NIST</affiliation>
    <street>I-270 and Quince Orchard Rd. Bldg 220 Room A127</street>
    <city>Gaithersburg</city>
    <state>MD</state>
    <postcode>20899</postcode>
    <country>USA</country>
    <phone>(301) 975-3335</phone>
    <fax>(301) 258-9749</fax>
    <email>simon.frechette@nist.gov</email>
  </contact>
```

```
<?xml version="1.0" encoding="UTF-8" ?>
<!-- $Id: module.xml,v 1.33 2003/05/22 19:14:34 darla Exp $ -->
<!DOCTYPE module (View Source for full doctype...)>
- <module name="activity" part="1047" version="1" wg.number="1515"
  wg.number.supersedes="1155" wg.number.arm="1156" wg.number.mim="1157"
  checklist.internal_review="2141" checklist.project_leader="2142" checklist.convener="2143"
  status="TS" language="E" publication.year="2003" published="y" rcs.date="$Date:
  2003/05/22 19:14:34 $" rcs.revision="$Revision: 1.33 $" development.folder="dvlp"
  xmlns:xlink="http://www.w3.org/1999/xlink" sc4.working_group="12">
  <keywords>module, activity, activity assignment, activity relationship</keywords>
- <contacts>
  <projlead ref="pdmmmodules.projlead" />
  <editor ref="pdmmmodules.editor" />
</contacts>
<purpose>This part of ISO 10303 specifies an application module for the representation
  of the data that identify activities and of association of an activity with product or
  activity data.</purpose>
+ <inscope>
+ <outscope>
- <normrefs>
  <normref.inc normref="ref10303-41.2000" />
</normrefs>
+ <arm>
+ <mapping_table>
+ <mim>
+ <bibliography>
</module>
```



ABSTRACT:

This part of ISO 10303 specifies the application module Activity.

The following are within the scope of this part of ISO 10303:

- definition of an activity;
- relationship between two activities;
- relationship between an activity and the product or activity data that it affects or uses.

KEYWORDS:

STEP, ISO 10303, module, activity, activity assignment, activity relationship

COMMENTS TO READER:

This document has been reviewed using the internal review checklist (see WG12 N2141), the project leader checklist (see WG12 N2142), and the convener checklist (see WG12 N2143), and is ready for publication.

Project leader: Simon Frechette

Address: NIST

I-270 and Quince Orchard Rd. Bldg 220 Room
A127
Gaithersburg
MD
20899
USA

Telephone: (301) 975-3335

Telefacsimile: (301) 258-9749

Electronic mail: simon.frechette@nist.gov

Project editor: Darla Nettles

Address: Advanced Technology Institute /

PDES, Inc.
5300 International Boulevard
North Charleston
SC
29418
USA

Telephone: +1-843-760-3232

Telefacsimile: +1-843-760-3349

Electronic mail: nettles@aticorp.org

Simple few steps

1. Create a module template
 1. Add purpose, scope,
2. Create the ARM EXPRESS/EXPRESS-G
 1. Create the EXPRESS
 2. Convert to XML
 3. Place in module
3. Create the MIM EXPRESS/EXPRESS-G
4. Add the mapping
5. Add contacts
6. Add normrefs, terms, abbreviations
7. Add bibliography
8. Add WG numbers
9. Add short names

Normrefs

- Normative refs for all modules used in EXPRESS added automatically
- Manually add any Resource parts used to module.xml
- Manually add any module referenced in scope
- Normative refs stored in:
stepmod\data\basic\normrefs.xml
- Referenced from module.xml

```
<normrefs>
  <normref.inc
    module.name="product_identification"/>
  <normref.inc normref="ref10303-41.2000"/>
</normrefs>
```

Reference to
module

Reference to resource
(from normrefs.xml)

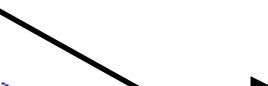


Address D:\rbn\projects\nist_module_repo\stepmod\data\basic\normrefs.xml



```
</normrefs>
+ <normref id="ref10303-1.1994" docbook-atts="url #DEFAULT"
  docbook="bibliomixed">
+ <normref id="ref10303-11.1994" docbook-atts="url #DEFAULT"
  docbook="bibliomixed">
+ <normref id="ref10303-11.-" docbook-atts="url #DEFAULT"
  docbook="bibliomixed">
+ <normref id="ref10303-21.1994" docbook-atts="url #DEFAULT"
  docbook="bibliomixed">
+ <normref id="ref10303-21.2001" docbook-atts="url #DEFAULT"
  docbook="bibliomixed">
+ <normref id="ref10303-31.1994" docbook-atts="url #DEFAULT"
  docbook="bibliomixed">
+ <normref id="ref10303-41.2000" docbook-atts="url #DEFAULT"
  docbook="bibliomixed">
- <stdref published="y">
  <orgname docbook="orgname">ISO</orgname>
  <pubdate docbook="pubdate">2000</pubdate>
  <stdnumber docbook="stdnumber">10303-41</stdnumber>
  <stdtitle docbook="title">Industrial automation systems and
    integration — Product data representation and
    exchange</stdtitle>
  <subtitle docbook="subtitle">— Part 41: Integrated generic resource:
    Fundamentals of product description and support.</subtitle>
</stdref>
</normref>
- <normref id="ref10303-42.2000" docbook-atts="url #DEFAULT"
  docbook="bibliomixed">
- <stdref published="y">
  <orgname docbook="orgname">ISO</orgname>
```

```
<?xml version="1.0" encoding="UTF-8" ?>
<!-- $Id: module.xml,v 1.33 2003/05/22 19:14:34 darla Exp $ -->
<!DOCTYPE module (View Source for full doctype...)>
- <module name="activity" part="1047" version="1" wg.number="1515"
  wg.number.supersedes="1155" wg.number.arm="1156" wg.number.mim="1157"
  checklist.internal_review="2141" checklist.project_leader="2142" checklist.convener="2143"
  status="TS" language="E" publication.year="2003" published="y" rcs.date="$Date:
  2003/05/22 19:14:34 $" rcs.revision="$Revision: 1.33 $" development.folder="dvlp"
  xmlns:xlink="http://www.w3.org/1999/xlink" sc4.working_group="12">
  <keywords>module, activity, activity assignment, activity relationship</keywords>
- <contacts>
  <projlead ref="pdmmmodules.projlead" />
  <editor ref="pdmmmodules.editor" />
</contacts>
<purpose>This part of ISO 10303 specifies an application module for the representation
  of the data that identify activities and of association of an activity with product or
  activity data.</purpose>
+ <inscope>
+ <outscope>
- <normrefs>
  <normref.inc normref="ref10303-41.2000" />
</normrefs>
+ <arm>
+ <mapping_table>
+ <mim>
+ <bibliography>
</module>
```



Reference to resource
(from normrefs.xml)



2 Normative references

The following normative documents contain provisions which, through reference in this text, constitute provisions of this International Standard. For dated references, subsequent amendments to, or revisions of, any of these publications do not apply. However, parties to agreements based on this International Standard are encouraged to investigate the possibility of applying the most recent editions of the normative documents indicated below. For undated references, the latest edition of the normative document referred to applies. Members of ISO and IEC maintain registers of currently valid International Standards.

ISO/IEC 8824-1:1998, *Information technology — Abstract Syntax Notation One (ASN.1) : Specification of basic notation.*

ISO 10303-1:1994, *Industrial automation systems and integration — Product data representation and exchange — Part 1: Overview and fundamental principles.*

ISO 10303-11:[11](#), *Industrial automation systems and integration — Product data representation and exchange — Part 11: Description methods: The EXPRESS language reference manual.*

ISO 10303-21:1994, *Industrial automation systems and integration — Product data representation and exchange — Part 21: Clear text encoding of the exchange structure.*

ISO 10303-202:1997, *Industrial automation systems and integration — Product data representation and exchange — Part 202: Application protocol: Associative draughting.*

ISO/TS 10303-1001:2001, *Industrial automation systems and integration — Part 1001: Application module: Appearance assignment.*

ISO/TS 10303-1017:2003, *Industrial automation systems and integration — Part 1017: Application module: Product identification.*

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.*

ISO/TS 10303-1049:2003, *Industrial automation systems and integration — Product data representation and exchange — Part 1049: Application module: Activity method.*

Reference to resource
(from normrefs.xml)

Terms

- Terms from outside the repository are stored in:
stepmod\data\basic\normrefs.xml
- Or defined with in a module.
- Referenced from module.xml



```
<?xml version="1.0" encoding="UTF-8" ?>
<!-- $Id: module.xml,v 1.25 2003/06/06 20:22:15 darla Exp $ -->
<!DOCTYPE module (View Source for full doctype...)>
<!-- Generated by mkmodule.js, Eurostep Limited, http://www.eurostep.com -->
- <module name="part_and_version_identification" part="1022" version="1" wg.number="1113"
  wg.number.supersedes="738" wg.number.arm="1114" wg.number.mim="1115" status="TS" language="E"
  publication.year="2003" published="y" checklist.internal_review="2133" checklist.project_leader="2134"
  checklist.convener="2135" rcs.date="$Date: 2003/06/06 20:22:15 $" rcs.revision="$Revision: 1.25 $"
  development.folder="dvlp" xmlns:xlink="http://www.w3.org/1999/xlink" sc4.working_group="12">
  <keywords>module, part, version, identification</keywords>
+ <contacts>
  <purpose>This part of ISO 10303 specifies an application module for the representation of the data
    that identify parts and their versions.</purpose>
+ <inscope>
+ <normrefs>
- <definition>
  <term id="NCM">non-countable material</term>
- <def>
  A non-countable material is a material the components of which cannot be counted or do not
  need to be counted.
  <example>Oil, Helium and sand are examples of non-countable material.</example>
</def>
</definition>
- <definition>
  <term id="part">part</term>
  <def>A part is a discrete object that may come into existence as a consequence of a
    manufacturing process. A part may contain components, which may be parts or non-countable
    materials.</def>
</definition>
```



3.5 Other terms and definitions

For the purposes of this part of ISO 10303, the following definitions apply:

3.5.1

non-countable material

A non-countable material is a material the components of which cannot be counted or do not need to be counted.

EXAMPLE Oil, Helium and sand are examples of non-countable material.

3.5.2

part

A part is a discrete object that may come into existence as a consequence of a manufacturing process. A part may contain components, which may be parts or non-countable materials.

3.5.3

raw material



Abbreviations

- Abbreviations from outside the repository are stored in: stepmod\data\basic\normrefs.xml
- Or defined with in a module.
- Referenced from module.xml

D:\rbn\projects\nist_module_repo\stepmod\data\basic\abbreviations.xml

```
<acronym>B-rep</acronym>
<term>boundary representation solid model</term>
</abbreviation>
- <abbreviation id="MIM">
  <acronym>MIM</acronym>
  <term.ref linkend="term13" normref="ref10303-1001.2001" />
</abbreviation>
- <abbreviation id="RDL">
  <acronym>RDL</acronym>
  <term.ref linkend="rdl" normref="ref10303-1001.2001" />
</abbreviation>
- <abbreviation id="UoF">
  <acronym>UoF</acronym>
  <term.ref linkend="term10" normref="ref10303-1.1994" />
</abbreviation>
- <abbreviation id="URL">
  <acronym>URL</acronym>
  <term>universal resource locator</term>
</abbreviation>
- <abbreviation id="AAM">
  <acronym>AAM</acronym>
  <term.ref linkend="term31" normref="ref10303-1.1994" />
</abbreviation>
```



My Computer

D:\rbn\projects\nist_module_repo\stepmod\data\modules\ap239_part_definition_information\module.

File Edit View Favorites Tools Help

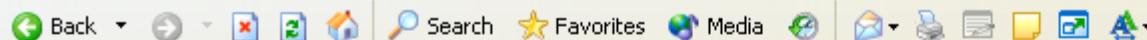
Back Search Favorites Media A

Address D:\rbn\projects\nist_module_repo\stepmod\data\modules\ap239_part_definition_information\module.xml Go

```
<!-- ++++++-->
-->
+ <outscope>
+ <normrefs>
<!-- <definition/>
-->
- <abbreviations>
- <abbreviation id="PADI">
    <acronym>PADI</acronym>
    <term>Part Definition Information</term>
</abbreviation>
</abbreviations>
<!-- Clause 4 ARM -->
- <arm>
    <!-- Note ARM short form EXPRESS is in arm.xml -->
    <!-- Units of functionality -->
- <!--
        <uof name="Part_Definition_Information">
            <description>
                <p>
                    This UoF specifies the definitional information for the
                    classification and representation of part information, document
                    information, and management information.
                </p>
            </description>
        </uof>
    </arm>
<!--
    <!-- Note ARM short form EXPRESS is in arm.xml -->
    <!-- Units of functionality -->
-->
```

ISO/CD-TS 10303-1293:- Ap239 part definition information - Microsoft Internet Explorer

File Edit View Favorites Tools Help



Address D:\rbn\projects\nist_module_repo\stepmod\data\modules\ap239_part_definition_information\sys\3_defs.xml Go Links >

- non-countable material;
- part.

3.11 Abbreviations

For the purposes of this part of ISO 10303, the following abbreviations apply:

AM	application module
ARM	application reference model
MIM	module interpreted module
URL	universal resource locator
PADI	Part Definition Information

© ISO 2003 — All rights reserved

Done

My Computer

Simple few steps

1. Create a module template
 1. Add purpose, scope,
2. Create the ARM EXPRESS/EXPRESS-G
 1. Create the EXPRESS
 2. Convert to XML
 3. Place in module
3. Create the MIM EXPRESS/EXPRESS-G
4. Add the mapping
5. Add contacts
6. Add normrefs, terms, abbreviations
7. Add bibliography
8. Add WG numbers
9. Add short names

Bibliography

- Bibliography defined in stepmod\data\basic\bibliography.xml
- Referenced from module
- OR defined in module.xml

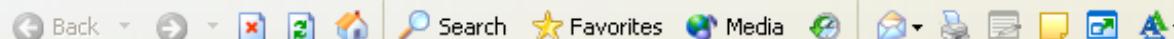
D:\rbn\projects\nist_module_repo\stepmod\data\modules\ap239_part_definition_information\module.

File Edit View Favorites Tools Help

Back Search Favorites Media A

Address D:\rbn\projects\nist_module_repo\stepmod\data\modules\ap239_part_definition_information\module.xml Go

```
+ <mim>
  <!-- MIM long form (optional) -->
+ <!-- -->
+ <mim>If>
- <bibliography>
- <!--
    Product identification
    in normrefs
    <bibitem.inc ref="ISO10303-1017"/>
-->
<bibitem.inc ref="ISO10303-1289" />
<!-- Ap239 management resource information -->
<bibitem.inc ref="ISO10303-1127" />
<!-- File identification -->
<bibitem.inc ref="ISO10303-1124" />
<!-- Document structure -->
<bibitem.inc ref="ISO10303-1126" />
<!-- Document properties -->
<bibitem.inc ref="ISO10303-1288" />
<!-- Management resource information -->
<bibitem.inc ref="ISO10303-1070" />
<!-- class -->
</bibliography>
'
```



module.xml by using the XML element: bibitem.

```
-->
<!DOCTYPE bibitem.list (View Source for full doctype...)>
- <bibitem.list>
- <bibitem id="AMConGde06">
  <!-- <orgname>ISO TC184/SC4</orgname> -->
  <pubdate docbook="pubdate">2001-05-02</pubdate>
  <stdtitle docbook="title">Draft standing document — Guidelines for the content of
    application modules</stdtitle>
  <subtitle docbook="subtitle">ISO TC184/SC4/N1161</subtitle>
</bibitem>
- <bibitem id="ISO15926-1">
  <pubdate docbook="pubdate">2000</pubdate>
  <stdnumber docbook="stdnumber">ISO 15926-1:2000</stdnumber>
  <stdtitle docbook="title">Industrial automation systems and integration — Integration of
    life-cycle data for process plants including oil and gas production facilities — Part 1:
    Overview and fundamental principles</stdtitle>
</bibitem>
- <bibitem id="ISO10303-41.2000">
  <orgname docbook="orgname">ISO</orgname>
  <pubdate docbook="pubdate">2000</pubdate>
  <stdnumber docbook="stdnumber">ISO 10303-41</stdnumber>
  <stdtitle docbook="title">Industrial automation systems and integration — Product data
    representation and exchange — Part 41: Integrated generic resource: Fundamentals of
    product description and support.</stdtitle>
</bibitem>
- <bibitem id="ISO10303-1012">
  <orgname docbook="orgname">ISO/TS</orgname>
```

ISO/CD-TS 10303-1293: Ap239 part definition information - Microsoft Internet Explorer

File Edit View Favorites Tools Help

Back Search Favorites Media

Address D:\rbn\projects\nist_module_repo\stepmod\data\modules\ap239_part_definition_information\sys\biblio.xml

[Index](#) | [Bibliography](#)



Application module: Ap239 part definition information

ISO/CD-TS 10303-1293

Cover page
Table of contents
Copyright
Foreword
Introduction
1 Scope
2 Normative references
3 Terms and abbreviations

4 Information requirements
4.1 Required AM ARMs
4.2 ARM type definitions
5 Module interpreted model
5.1 Mapping specification
5.2 MIM EXPRESS short listing
5.2.1 MIM type definitions

A MIM short names
B Information object registration
C ARM EXPRESS-G
D MIM EXPRESS-G
E Computer interpretable listings
Bibliography

Bibliography

[1] *Draft standing document — Guidelines for the content of application modules*, ISO TC184/SC4/N1161, 2001-05-02.

[2] ISO/CD-TS 10303-1289, *Industrial automation systems and integration — Product data representation and exchange — Part 1289: Application module: Ap239 management resource information*, 2003.

[3] ISO/CD-TS 10303-1127, *Industrial automation systems and integration — Product data representation and exchange — Part 1127: Application module: File identification*, 2003.



My Computer

Simple few steps

1. Create a module template
 1. Add purpose, scope,
2. Create the ARM EXPRESS/EXPRESS-G
 1. Create the EXPRESS
 2. Convert to XML
 3. Place in module
3. Create the MIM EXPRESS/EXPRESS-G
4. Add the mapping
5. Add contacts
6. Add normrefs
7. Add bibliography
8. Add WG numbers
9. Add short names

WG numbers

```
<module  
  name="activity"  
  part="1047"  
  version="1"  
  wg.number="1515"  
  wg.number.supersedes="1155"  
  wg.number.arm="1156"  
  wg.number.mim="1157"  
  checklist.internal_review="2141"  
  checklist.project_leader="2142"  
  checklist.convener="2143"
```

WG numbers to arm.exp and mim.exp

(*

\$Id: arm.exp,v 1.14 2002/08/02 13:53:41 darla Exp \$
ISO/TC184/SC4 WG12N1156 - ISO/TS 10303-1047 Activity
- EXPRESS ARM

*)

SCHEMA Activity_arm;

Easier to copy from EXPRESS in Annex E



Annex E (informative)

Computer interpretable listings

This annex references a listing of the EXPRESS entity names and corresponding short names as specified or referenced in this part of ISO 10303. It also provides a listing of each EXPRESS schema specified in this part of ISO 10303 without comments nor other explanatory text. These listings are available in computer-interpretable form in Table E.1 and can be found at the following URLs:

Short names: <http://www.tc184-sc4.org/Short_Names/>

EXPRESS: <<http://www.tc184-sc4.org/EXPRESS/>>

Table E.1 — ARM and MIM EXPRESS listings

Display
EXPRESS

Description	XML file	ASCII file	Identifier
ARM short form EXPRESS	XML	EXPRESS	ISO TC184/SC4/WG12 N1156
MIM short form EXPRESS	XML	EXPRESS	ISO TC184/SC4/WG12 N1157

If there is difficulty accessing these sites, contact ISO Central Secretariat or contact the ISO TC184/SC4 Secretariat directly at: sc4sec@tc184-sc4.org.

NOTE The information provided in computer-interpretable form at the above URLs is informative. The information that is contained in the body of this part of ISO 10303 is normative.



module.xml (Date: 2003/05/22 19:14:34 Revision: 1.33) arm.xml (Date: 2002/07/30 10:44:56 Revision: 1.13) mim.xml (Date: 2002/07/09 11:06:43 Revision: 1.11)

[Modules Index](#) | [Ballot home](#)

Application module: Activity

ISO/TS 10303-1047:2003(E)

© ISO

[Cover page](#)

[Table of contents](#)

[Copyright](#)

[Foreword](#)

[Introduction](#)

[1 Scope](#)

[2 Normative references](#)

[3 Terms and abbreviations](#)

[4 Information requirements](#)

[4.1 Required AM ARM](#)

[4.2 ARM type definition](#)

[4.3 ARM entity definitions](#)

[5 Module interpreted model](#)

[5.1 Mapping specification](#)

[5.2 MIM EXPRESS short listing](#)

[5.2.1 MIM type definition](#)

[5.2.2 MIM entity definition](#)

[A MIM short names](#)

[B Information object registration](#)

[C ARM EXPRESS-G](#)

[D MIM EXPRESS-G](#)

[E Computer interpretable listings](#)

[Bibliography](#)

(*

ISO TC184/SC4/WG12 N1156 - ISO/TS 10303-1047 Activity - EXPRESS ARM
*)

SCHEMA Activity_arm;

USE FROM [Activity_method_arm](#); -- ISO/TS 10303-1049

```
TYPE activity_item = EXTENSIBLE GENERIC_ENTITY SELECT;
END_TYPE;
```

```
ENTITY Activity;
  chosen_method : Activity\_method;
  description : OPTIONAL STRING.
```



Simple few steps

1. Create a module template
 1. Add purpose, scope,
2. Create the ARM EXPRESS/EXPRESS-G
 1. Create the EXPRESS
 2. Convert to XML
 3. Place in module
3. Create the MIM EXPRESS/EXPRESS-G
4. Add the mapping
5. Add contacts
6. Add normrefs
7. Add bibliography
8. Add WG numbers
9. Add short names

Short names

- Generate and register them at:

<http://www.steptools.com/sc4/services/index.html>

- Register by:
 - “Submit report” button on the “Output not wanted pane” (as below).
- Fill in the comment field requesting that the names are registered.
- You should indicate what the module is and what ballot stage it is at.

Short names

- Add them to module.xml:

```
<mim>
  <express-g>
    <imgfile file="mimexpg1.xml"/>
    <imgfile file="mimexpg2.xml"/>
  </express-g>

  <shortnames>
    <shortname name="APACAS" entity="applied_action_assignment"/>
  </shortnames>

</mim>
```

[1 Scope](#)[2 Normative references](#)[3 Terms and abbreviations](#)[5.2 MIM EXPRESS short listing](#)[5.2.1 MIM type definition](#)[5.2.2 MIM entity definition](#)[Bibliography](#)

Annex A (normative)

MIM short names

Table A.1 provides the short names for entities defined in the MIM of this part of ISO 10303.

Entity names in this part of ISO 10303 have been defined in clause 5.2 and in other ISO standards identified in clause 2.

Requirements on the use of the short names are found in the implementation methods included in ISO 10303.

NOTE The EXPRESS entity names are available from Internet:

< http://www.tc184-sc4.org/Short_Names/ >

Table A.1— MIM short names of entities

Entity data types names	Short names
applied_action_assignment	APACAS



The module is complete!



Issues and SEDS

- Add issues to:
stepmod\data\modules\[module]\dvlp\issues.xml
- Will be displayed in line
- If marked as SEDS will be displayed on cover page
 - See event, event_occurrence_date_and_time_item

STEP modules

File Edit View Favorites Tools Help

Back Search Favorites Media

Address: D:\rbn\projects\nist_module_repo\stepmod\nav\index.xml

STEPmod

Modules Express Application Protocols Resource parts Resource schemas Ballots

[Alphabetical](#), [ARM objects](#), [Alphabetical](#), [Alphabetical](#), [Alphabetical](#), [Ballots](#)

[Project](#), [MIM objects](#), [Project](#), [Mappings](#)

[Parts, Leader](#), [Keywords](#)

[event](#)

ISO/TS 10303-1064:2003 Event

This subclause specifies the MIM types for this application module. The MIM types and definitions are specified below.

5.2.1.1 event_occurrence_date_and_time_item

The **event_occurrence_date_and_time_item** type is an extension of the **date_and_time_item** type. It adds the entity data type **event_occurrence** to the list of alternate data types.

NOTE This select type provides the capability to associate date and time information to an **event_occurrence**. In the context of this application module, this enables to represent the start dates of an event.

G

[general_surface](#), [generic_expressions](#), [geometric_tolerances](#), [geometric_validations](#), [geometrically_bounded_group](#), [group](#)

Issue: 850 by Rob Bodington (03-04-15) [editorial, closed]

Registered in the *SC4 database* as SEDS: 850

Incorrect use of GENERIC_ENTITY. Should either be a SELECT or
GENERIC_ENTITY EXTENSIBLE SELECT

My Computer

STEP modules

File Edit View Favorites Tools Help

Back Search Favorites Media

Address XML D:\rbn\projects\nist_module_repo\stepmod\nav\index.xml Go

STEPmod

HTML Help
Module menu □

+ elemental_geometry
+ elemental_topology
+ envelope
+ event
+ event_assignment
+ experience
+ expression
+ extended_measurement
+ external_class
+ external_item_id
+ external_model
+ external_property

F

+ faceted_boundary
+ file_identification
+ foundation_representation
+ functional_breakout
+ functional_data
+ functional_data_annotation
+ furniture_catalog
+ furniture_catalog
+ furniture_catalog
+ furniture_interior

G

+ general_surface
+ generic_expressions
+ geometric_tolerances
+ geometric_validation
+ geometrically_bounded
+ geometrically_bounded
+ group

Modules

Alphabetical, Project Parts, Leader, Status, Keywords

Express

ARM objects MIM objects

Application Protocols

Alphabetical, Project,

Resource parts

Alphabetical,

Resource schemas

Alphabetical, Mappings

Balloons

Ballots

ISO/TS 10303-1064:2003 Event

- characterization of an event with respect to another event.

KEYWORDS:

STEP, ISO 10303, module, event, event relationship

COMMENTS TO READER:

This document has been reviewed using the internal review checklist (see WG12 N2145), the project leader checklist (see WG12 N2146), and the convener checklist (see WG12 N2147), and is ready for publication.

The following SEDS have been addressed by this edition: 830, 850.

Project leader: Simon Frechette
Address: NIST

Project editor: Darla Nettles
Address: Advanced Technology Institute /

Issues and SEDS

- XML can be copied from the developer view of repository

**STEPmod**HTML Help
Module menu □**Modules**[Alphabetical](#),
[Project](#),
[Parts, Leader,](#)
[Status, Keywords](#)**Express**[ARM objects](#)
[MIM objects](#)**Application Protocols**[Alphabetical](#),
[Project](#),**Resource parts**[Alphabetical](#),**Resource schemas**[Alphabetical](#),
[Mappings](#)**Ballots**[Ballots](#)**A**

- activity
- Module
- index
- + ISO view
- + Summary
- view
- + Developer
- View
- XML
- references
- Issue templates
- ARM
- SELECTs
- view
- Mapping view
- Mapping view with
- test
- Select matrix

XML templates for issues against activity

```
category="editorial"
by=""
date=""
status="open"
seds="no">
Issue description goes here.
</issue>
```

attribute: Activity.description

```
<!-- ++++++-->
<issue id="" type="arm" linkend="Activity_arm.Activity.description" category="editorial" by="" date="" status="open" seds="no">
Issue description goes here.
</issue>
```

attribute: Activity.id

```
<!-- ++++++-->
<issue id="" +--+="" -->
```

My Computer

In addition to modules

The repository will support:

- Application Protocol documents
- Common Resource documents

Summary slide

- Covered the “rules” for modules
- Created a module using mkmodule.wsf
 - module.xml
- Added new module to repository_index.xml
- Created the EXPRESS/EXPRESS-G
 - Using GraphicalEXPRESS
- Generated mappings using mkmapping and “developer view”
- Populated rest of module.xml
- Raised issues



Overall Agenda

- A Reminder about STEP Modularization
- Introducing the STEP Modules Repository
- Building the STEP Module Content
- Repository Details for Module Developers

Section Agenda

1. Using the repository

- Tools for accessing, adding modules, revision control
SourceForge / CVS

2. Repository tools

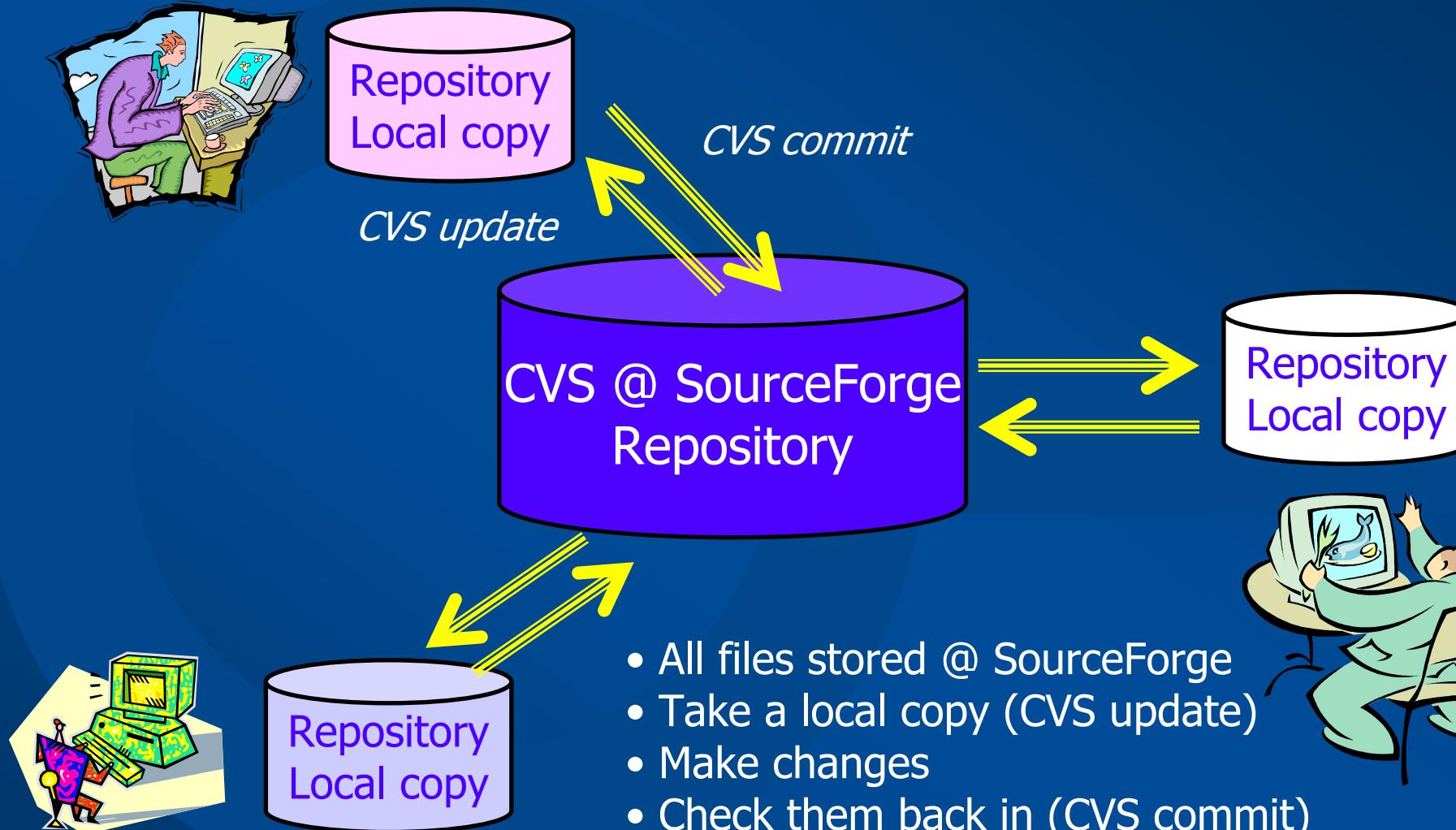
- Build tools, validation, generating HTML

3. Ballot packages

Updating SourceForge repository

- Stepmod is hosted at SourceForge
- Use CVS to manage collaboration
- CVS is a Software configuration management system or version control system
 - Files are managed
 - Can retrieve previous revisions
- It supports distributed development teams
 - Can share files with all developers (no more sending zips with associated synchronization problems)

CVS supports distributed teams



View history

- <http://cvs.sourceforge.net/cgi-bin/viewcvs.cgi/stepmod/stepmod>

stepmod/stepmod/data/modules/activity

Current directory: [\[SourceForge\]](#) / [stepmod](#) / [stepmod](#) / [data](#) / [modules](#) / activity

File	Rev.	Age	Author	Last log entry
Attic/ [Don't hide]				
dvlp/				
nav/				
sys/				
arm.exp	1.14	10 months	darla	added ISO/TC184/SC4 WG12 to N number
arm.xml	1.13	10 months	robbod	now outputs GENERIC_ENTITY
arm_descriptions.xml	1.12	2 weeks	darla	corrected note numbering
armexpg1.gif	1.7	12 months	robbod	Removed all person, date, document approval assignments
armexpg1.xml	1.10	12 months	robbod	Removed all person, date, document approval assignments
armexpg2.gif	1.5	12 months	robbod	Removed all person, date, document approval assignments

CVS log for stepmod/stepmod/data/modules/activity/arm.exp



Up to [\[SourceForge\]](#) / [stepmod](#) / [stepmod](#) / [data](#) / [modules](#) / [activity](#)

[Request diff between arbitrary revisions](#)

Default branch: [MAIN](#)

Bookmark a link to: [HEAD](#) / ([download](#))

Revision [1.14](#) / ([view](#)) - [annotate](#) - [[select for diffs](#)] , Fri Aug 2 13:53:41 2002 UTC (10 months, 2 weeks ago) by *darla*

Branch: [MAIN](#)

CVS Tags: [wg3n1188 20030410](#), [test1](#), [release 2003-01-22 pdm modules](#), [release-20021125-sc4sec](#), [release-20020912-pdm-modules-signoff](#), [release-20020812-pdm-modules-wg12-convener-review](#), [preplcsqcreview 20020926](#), [plcsqcreview plcs bp1-20021002](#), [plcs bp3 20030607](#), [pdm ballot 072002-20020809](#), [pdm ballot 072002-20020807](#), [pdm 20030206](#), [pdm 20030130](#), [partinfoballot](#), [part info pkg](#), [part 56 cd ballot](#), [gdt 20030116](#), [apdoc 20030604a](#), [antxtgdtpartinfo](#), [annotatedtxt pkg](#), [PLC 20030103 bcl](#), [PLCS logical model of expressions-20030224](#), [PLCS 20030123 bcl](#), [PLCS 20030121 bcl](#), [PLCS 20030103 bcl](#), [PLCS 20021223 bcl](#), [HEAD](#)

Changes since 1.13: +2 -2 lines

Diff to [previous 1.13](#)

added ISO/TC184/SC4 WG12 to N number

Revision [1.13](#) / ([view](#)) - [annotate](#) - [[select for diffs](#)] , Thu Aug 1 10:20:09 2002 UTC (10 months, 3 weeks ago) by *gosetl*

Branch: [MAIN](#)

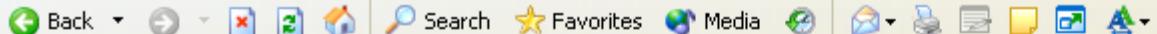
Changes since 1.12: +2 -2 lines

Diff to [previous 1.12](#)

no message



File Edit View Favorites Tools Help



Address

<http://cvs.sourceforge.net/cgi-bin/viewcvs.cgi/stepmod/stepmod/data/modules/activity/arm.exp.diff?r1=1.13&r2=1.14>

Go

[Return to arm.exp CVS log](#)[Up to \[SourceForge\] / stepmod / stepmod / data / modules / activity](#)

Diff for /stepmod/stepmod/data/modules/activity/arm.exp between version 1.13 and 1.14

version 1.13, 2002/08/01 10:20:09

version 1.14, 2002/08/02 13:53:41

Line 1

```
(*  
$Id$  
N1156 - ISO/TS 10303-1047 Activity - EXPRESS ARM  
*)  
SCHEMA Activity_arm;
```

Line 1

```
(*  
$Id$  
ISO/TC184/SC4 WG12N1156 - ISO/TS 10303-1047 Activity -  
EXPRESS ARM  
*)  
SCHEMA Activity_arm;
```

Legend:

Removed from v. 1.13

changed lines

Added in v. 1.14

Colored Diff

Show

[Back to SourceForge](#)Powered by
[ViewCVS 0.8](#)

Internet

Requirements

SourceForge

1. Create a SourceForge account

Goto <http://sourceforge.net/projects/stepmod> and use the menu option on the left "New User via SSL"

2. Become a Stepmod developer

Email your project leader to be added to the Stepmod development team

Tools

- CVS client (WinCVS)
- SSH communications software
- Java
- ANT
- SAXON (XSL engine)
- Internet Explorer (IE6 or IE5 wth MSXML3.0)
- Cscript

CVS tools

1. Secure Shell communication tools

- Putty / Plink / Pageant
 - <http://www.chiark.greenend.org.uk/~sgtatham/putty/>

2. CVS client

- WinCVS
 - <http://www.wincvs.org/>

Detailed instructions for getting setup as a SourceForge developer

1. Install WinCVS - Can be down loaded from: <http://www.wincvs.org/>
2. Install the SSH software PuTTY - You need putty.exe, plink.exe, pageant.exe, puttygen.exe. The putty help pages would be a useful download as well. These can be down loaded from <http://www.chiark.greenend.org.uk/~sgtatham/putty/>. Useful help pages: http://sourceforge.net/docman/display_doc.php?docid=6841&group_id=1
3. Set up and test your SSH to connect to SourceForge. First generate and post your SSH key to SourceForge. Details on how to do this are at http://sourceforge.net/docman/display_doc.php?docid=761&group_id=1. Note - you will have to wait 6 hours before your key has been circulated.
4. Create and save a putty session that logs on to SourceForge using SSH. On the session window:
 5. Host Name: cvs.stepmod.sourceforge.net
SSH selected On the SSH window (the SSH window is obtained by clicking Auth on the left side of the Putty window):
 6. SSH 1 selected Specify your authentication key.
 7. Set up PLINK and CVS environment variables
 8. set CVS_RSH=\path\to\plink.exe (that means you have to write:
CVS_RSH=C:\.....i.e. the path to plink.exe in your computer) set PLINK_PROTOCOL=ssh
 9. Run pageant adding your key (that should be done any time you will switch on or reboot your computer, including the adding of your key)
 10. Configure WinCVS
Admin->Preferences->General
CVSROOT box
:ext:putty_session_name:/cvsroot/stepmod
where putty_session_name is the name of the putty session you just created.
Admin->Preferences->Ports
Alternative rsh name
C:\..... (i.e. the path to plink.exe in your computer)
 11. Check out the repository - use the Admin->Command line
cvs -d:ext:username@cvs.stepmod.sourceforge.net:/cvsroot/stepmod co stepmod



Name	Rev.	Option	Status
ballots			Folder
browser			NonCvs Folder
css			Folder
data			Folder
doc			Folder
dtd			Folder
etc			Folder
express			NonCvs Folder
help			Folder
ieee			NonCvs Folder
images			Folder
nav			Folder

Modules Explore

```
Python is not available !
CVSROOT: robbod@cvs.stepmod.sourceforge.net:/cvsroot/stepmod (ssh authentication)
TCL is available, shell is enabled : help (select and press enter)
TCL or Python are *not* available, shell is disabled
```

For Help, press F1

Way of working

- Works on trust
 - Everyone can see and edit everyone's files
- Only edit your own modules
 - Especially don't edit XSL and DTDs
- Do frequent CVS updates
 - i.e. keep your local copies up to date
- Make sure all files are in lower case
- Commit all files including files in sys, nav, & dvlp directories
- Only commit valid XML

Otherwise you stop the repository from working for everyone else!
- Do commit your files – that encourages harmonization
- No locking of files

Build tools

- ANT
 - Applies the XSL to the XML to generate the HTML
 - Uses SAXON and Java
 - Note – there are SAXON specific extensions used, so other XSL tools will not work.
 - A good way of checking for errors.
 - See stepmod\help\util.htm#xmlhtmlant for setup details

Section Agenda

1. Using the repository

- Tools for accessing, adding modules, revision control
SourceForge / CVS

2. Repository tools

- Build tools, validation, generating HTML

3. Ballot packages

ANT build

- See stepmod\utils\build.xml for ANT targets command

cd stepmod\utils

ant

generates HTML for everything in the module repository
ant modules

generates HTML for all the modules

ant wf

checks all XML documents for well-formedness

ant valid_modules

checks module XML for well-formedness and conformance
to DTD

Run ANT on single modules

`ant -DMODULES=data/modules/[module-name] modules`
generates HTML for a single module

`ant -DMODULES=data/modules/[module-name] clean_module`
deletes all HTML generated for a single module

`ant -DMODULESDIR=data/modules/[module-name] wf_module`
checks a module's XML files for well-formedness

Ballot packages

- Create a ballot package
 - Requires ANT and Java
 - Run: stepmod\utils\mkballot.wsf
- Add the modules to the ballot
 - Edit: stepmod\ballots\ballots\[ballot]\ballot_index.xml
- Build the ANT file
 - Run: ant –buildfile buildbuild
- Generate the HTML
 - Run: ant all
 - Generates: stepmod\ballots\isohtml\[ballot]
- Get Ballot EXPRESS
 - stepmod\utils\getBallotExpress.wsf
- Create a zip file of the ballot package.
 - Run: ant zip
 - Creates a zip file: stepmod\ballots\isohtml\plcs_bp3\plcs_bp3xxxx.zip

Other Useful Utilities

getModuleExpress

- Concatenates all the EXPRESS used by a module
- Need to provide a list of:
 - ARM modules
 - MIM modules
 - IR schemas

Conclusions

- Introduction to modules
- Shown the contents of a module
- Demonstrated how to build a module
- Shown how to work with SourceForge to share your module
- Shown how to send your modules ballot



• **eurostep**[®]

All Presentation Material Copyright Eurostep Limited

BACKUP

Utilities

express2xml

- converts EXPRESS into XML
- ignores Express comments

splitresource

- splits an EXPRESS file into its constituent schemas.

Tools for XML

- To view rendered XML
 - Internet Explorer v6
 - XSL capable
 - Internet Explorer v5
 - Install MSXML3 (xmlinst)
- XML / XSLT tools
 - Many available
 - Using SAXON
<http://sourceforge.net/projects/saxon>
- To generate EXPRESS-G, EXPRESS, XML
 - GraphicalExpress (free VISIO plug-in)
 - <http://www.eurostep.com>