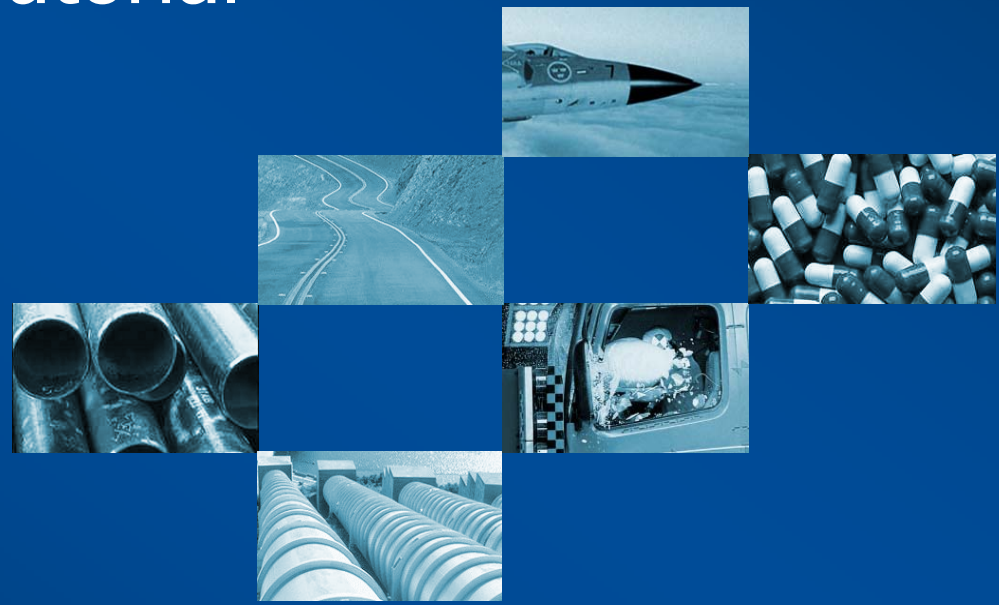
STEPmod Tutorial



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

eurostep®

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,

eurostep®

A Reminder about STEP Modularization

Section Agenda

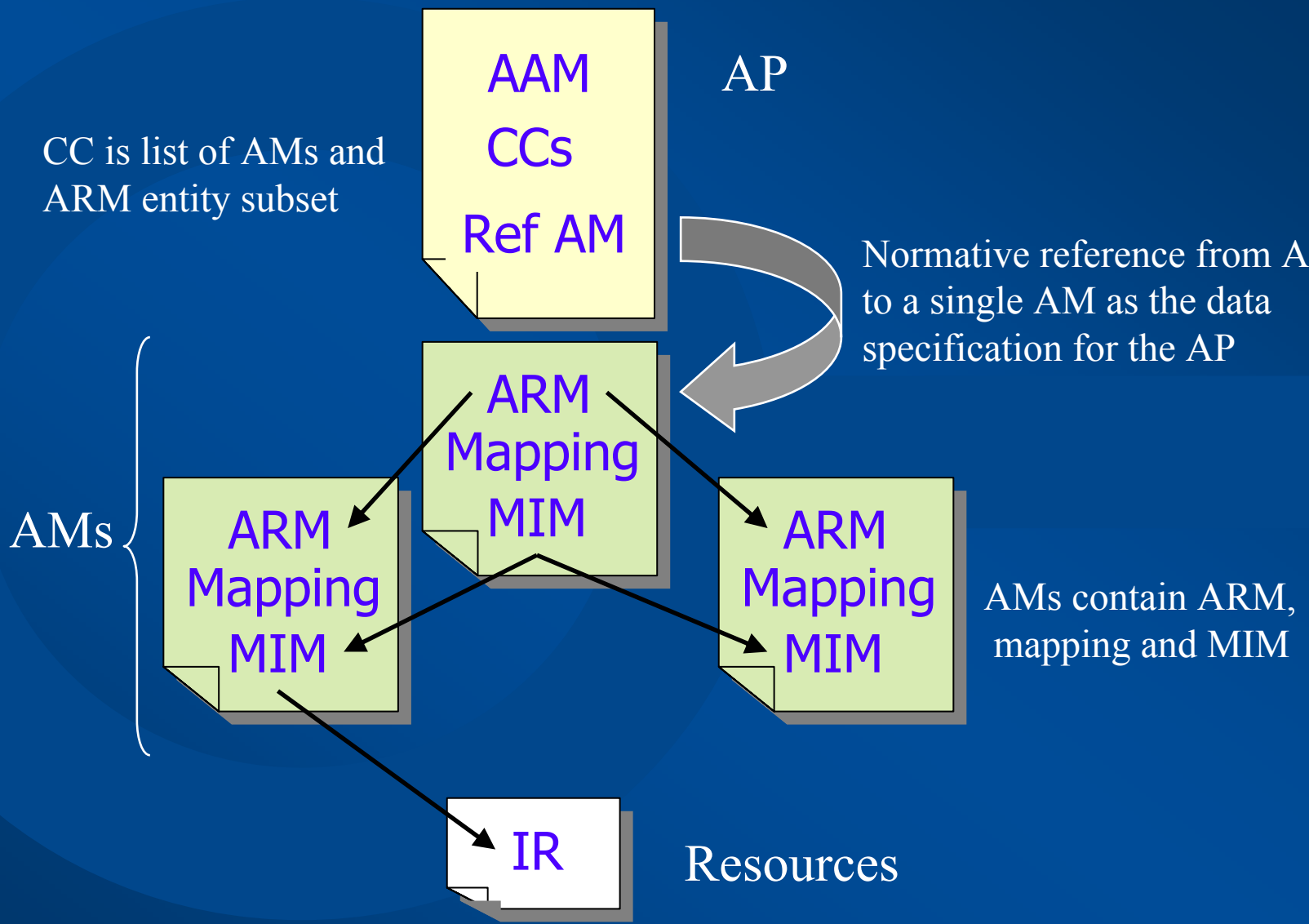
1. STEP Modular Architecture
2. Content of a STEP Application Module

eurostep®

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

eurostep®

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

eurostep®

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!

eurostep®

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

eurostep®

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

eurostep®

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

EXPRESS models

Model Documentation

Module Documentation

Usage guides

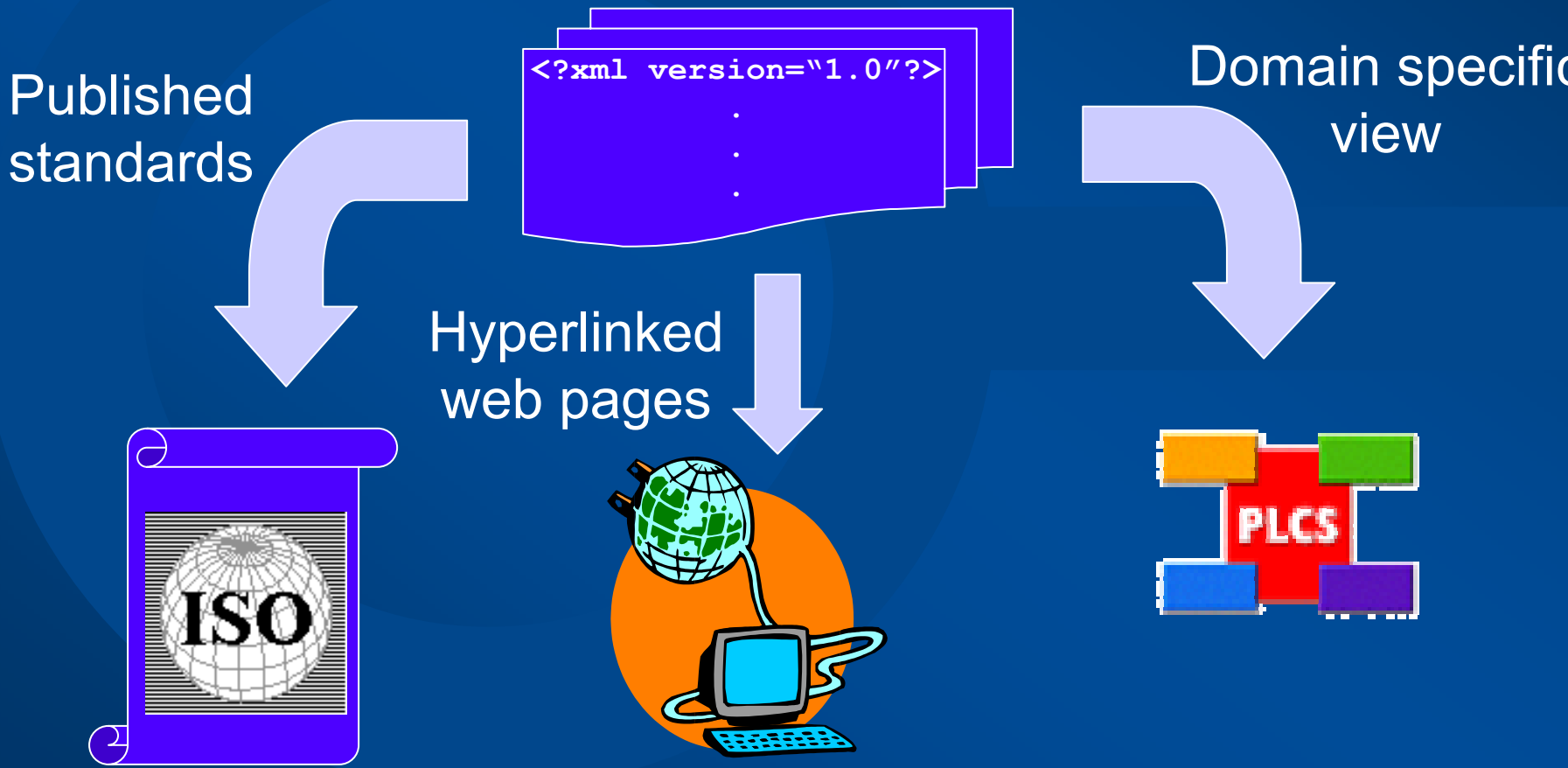
XSL stylesheets

Generated ISO/SC4 HTML and/or direct XML browsing

eurostep®

XML enables other uses too!

To allow publication on the web for multiple purposes



eurostep®

How does presentation work?

XML

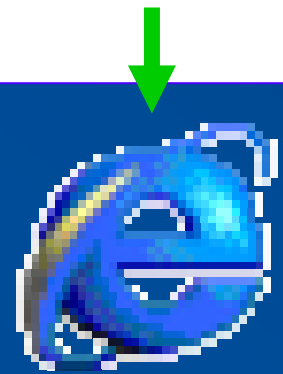
```
<express>
  <schema
     name='product_
  </schema>
</express>
```

XSL

```
<?xml version='1.0'?>
<xsl:stylesheet version="1.0"
     xmlns:xsl="http://ww
<xsl:output method="htr

<xsl:template match="/"
  <HTML>
    <HEAD>
      <TITLE><xsl:value-
      select="/express/sch
    </HEAD>
    <BODY>
      EXPRESS specificatio
      *)
      <H1><xsl:value-of
      select="/express/sch
      (*
    </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>
```



eurostep®

Who will use the XML repository?

- Standards developers
  - Repository will reduce AP development costs though 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

eurostep®

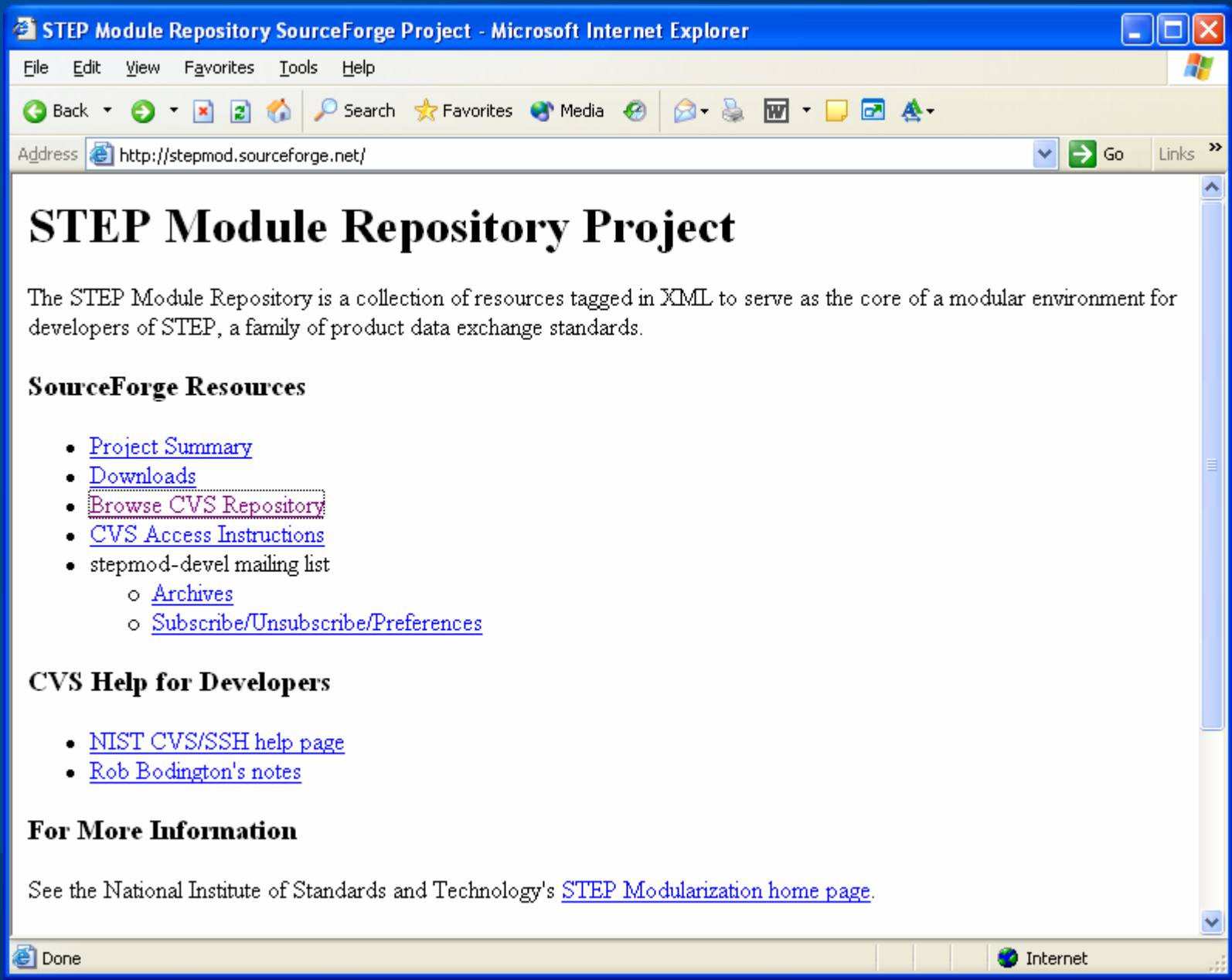
The repository is at SourceForge

- 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

eurostep®

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

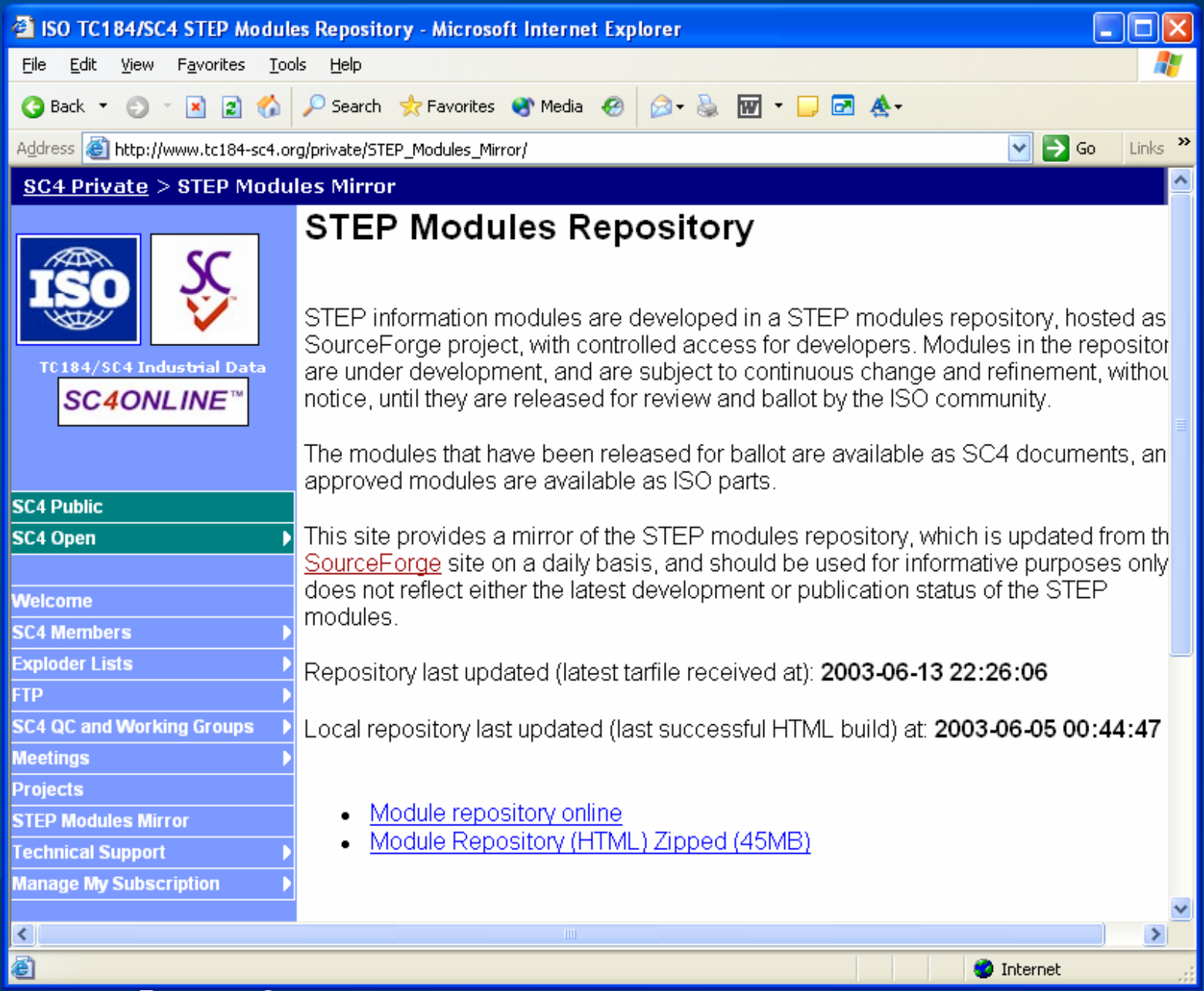


eurostep®

eurostep®

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/



eurostep®

Introducing the STEP Modules Repository
Section Agenda

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

eurostep®

Browsing the modules

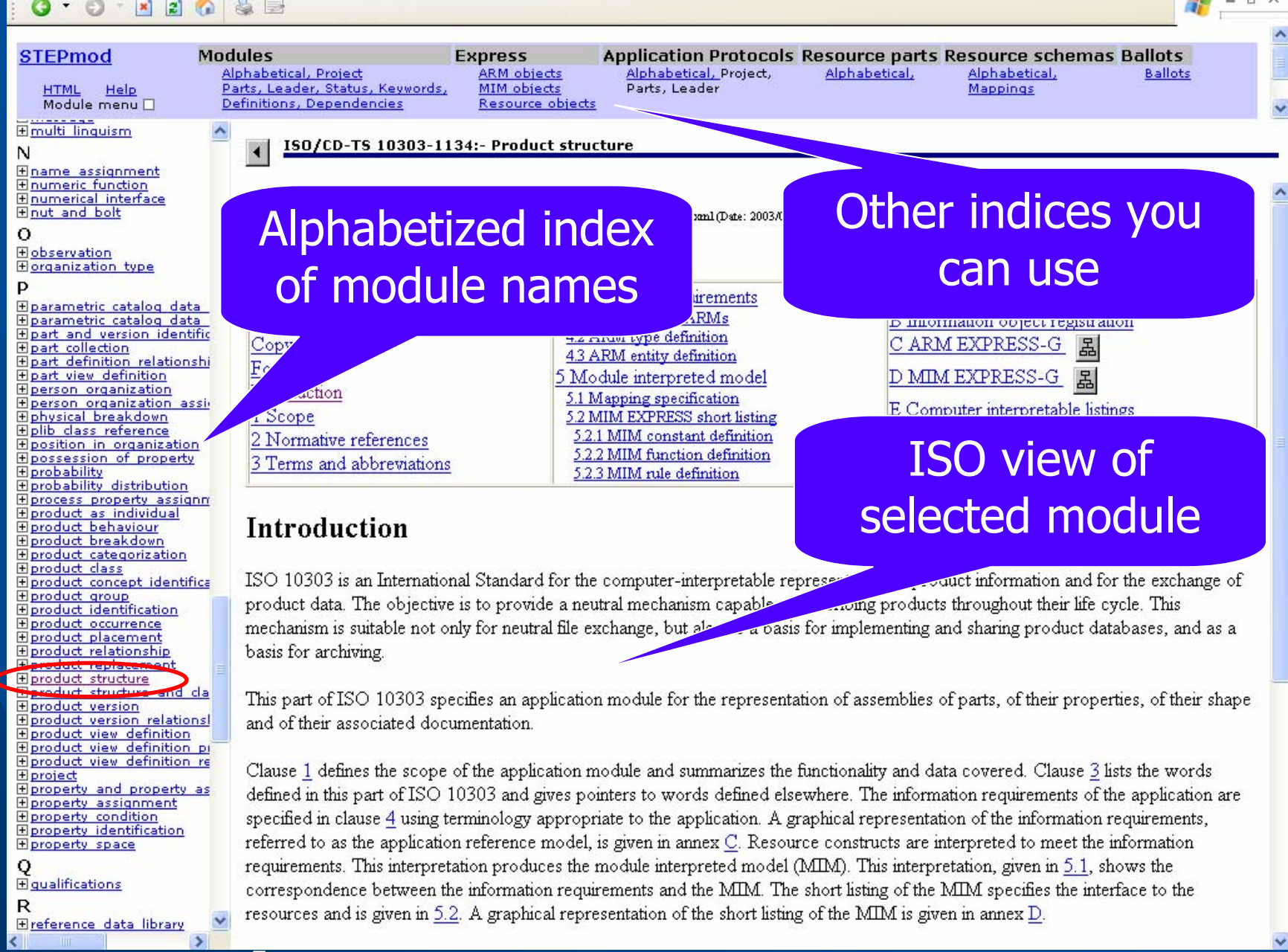
- The next few slides show examples of browsing the repository
  - Main index of all modules is stepmod\nav\index.xml
- You can get:
  - an index of all modules
  - an index of all entities in all module ARMs
  - a table of dependencies between modules
- There's no end to the list of indices that can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

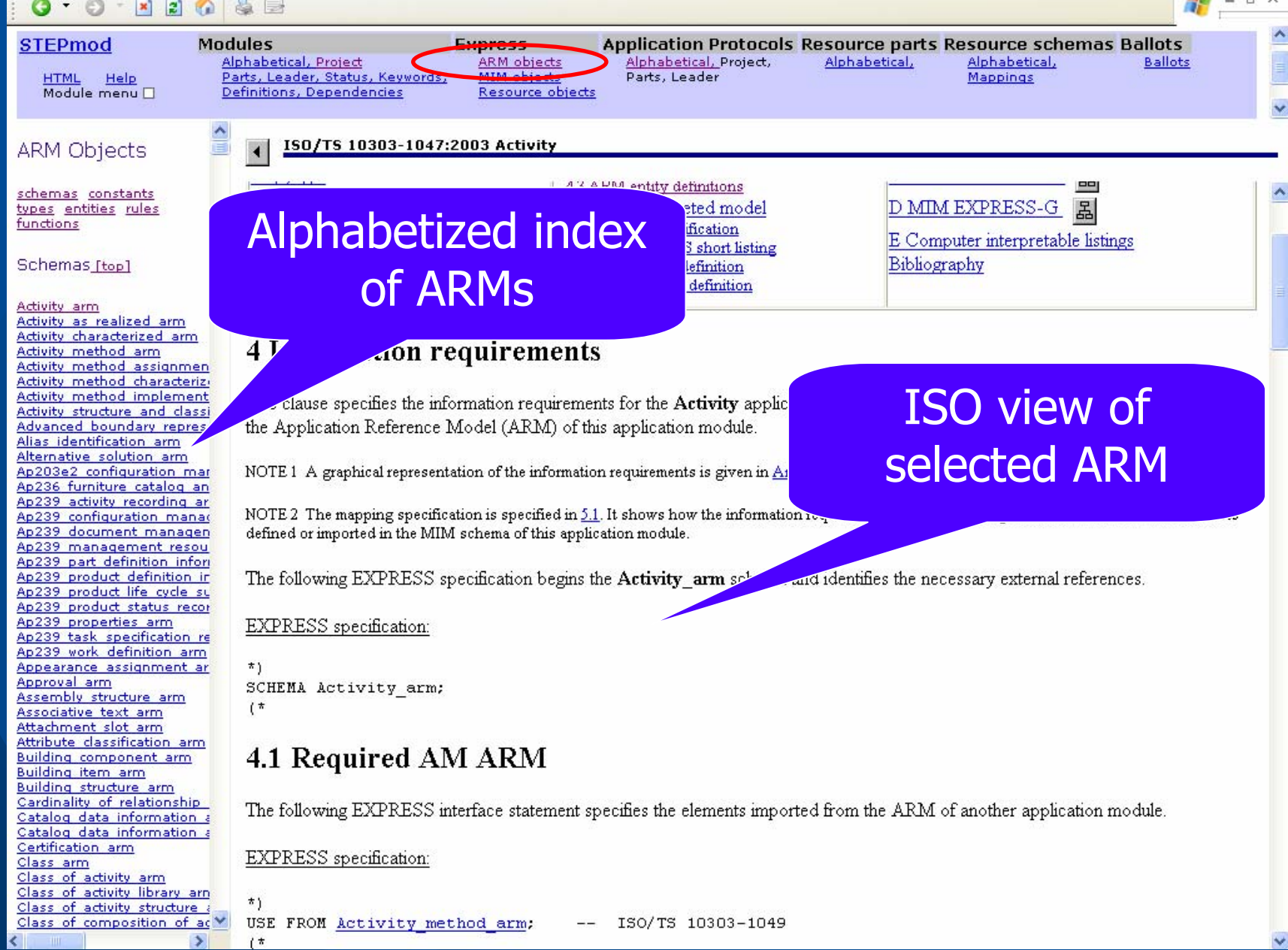
Tools to browse

- You can browse the XML or HTML version.

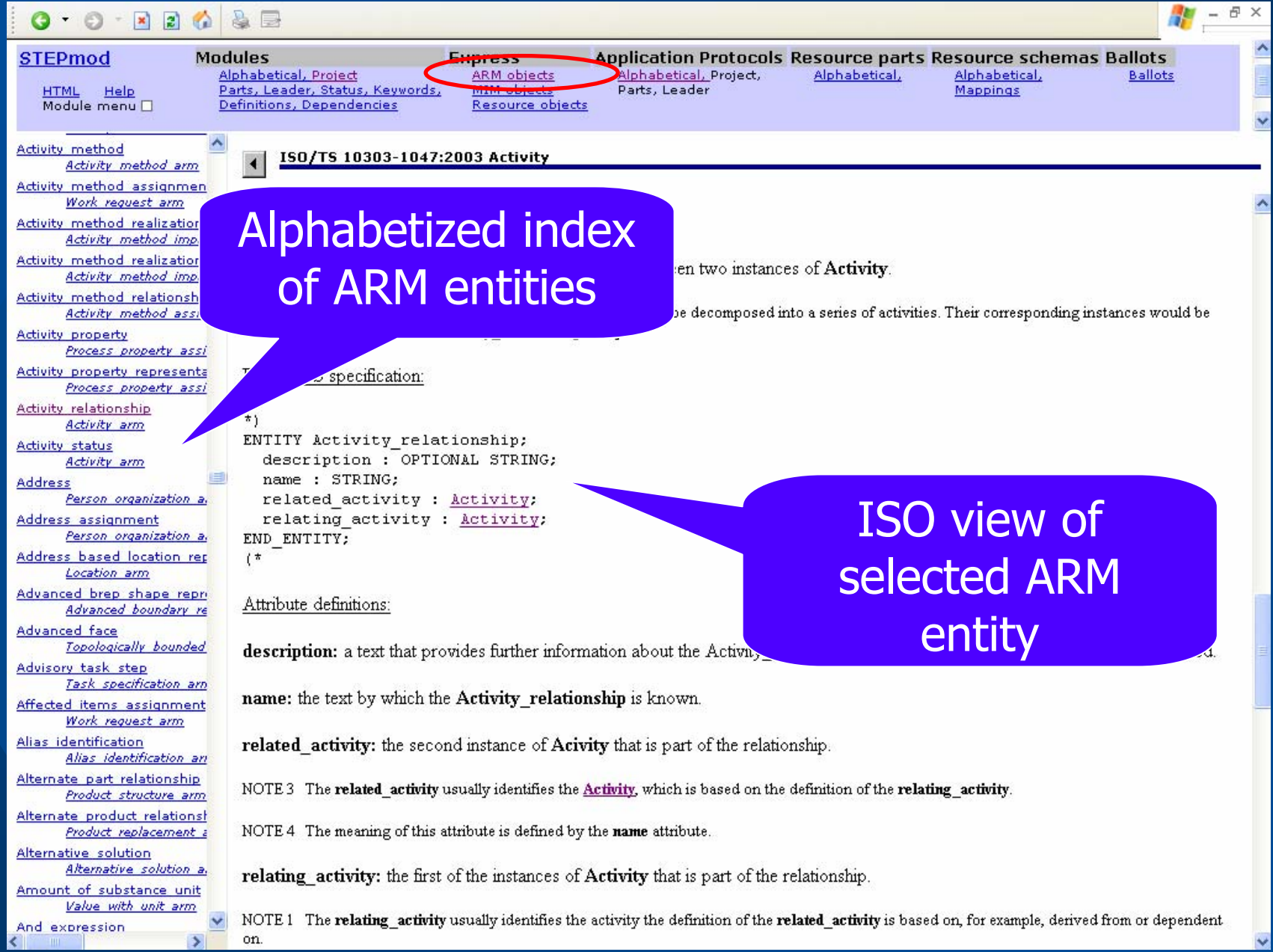
- HTML
  - is a snap shot.
  - should be able to use any browser

- XML
  - is dynamic – any change to the module is immediately visible.
  - You need:
    - Internet Explorer v6
      - XSL capable
    - Internet Explorer v5
      - Install MSXML3 (xmlinst)

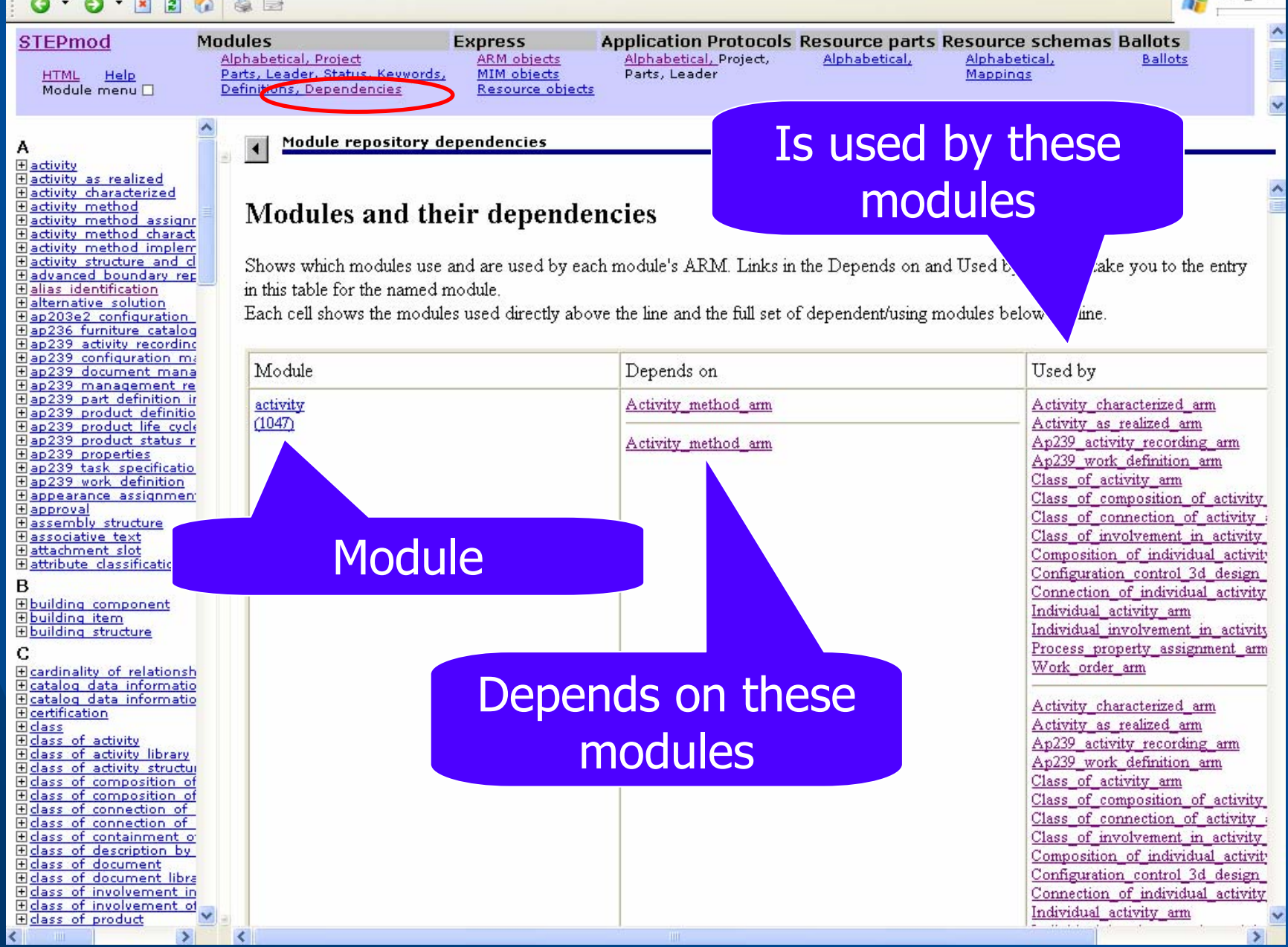




eurostep®



eurostep®



Introducing the STEP Modules Repository
Section Agenda

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

eurostep®

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

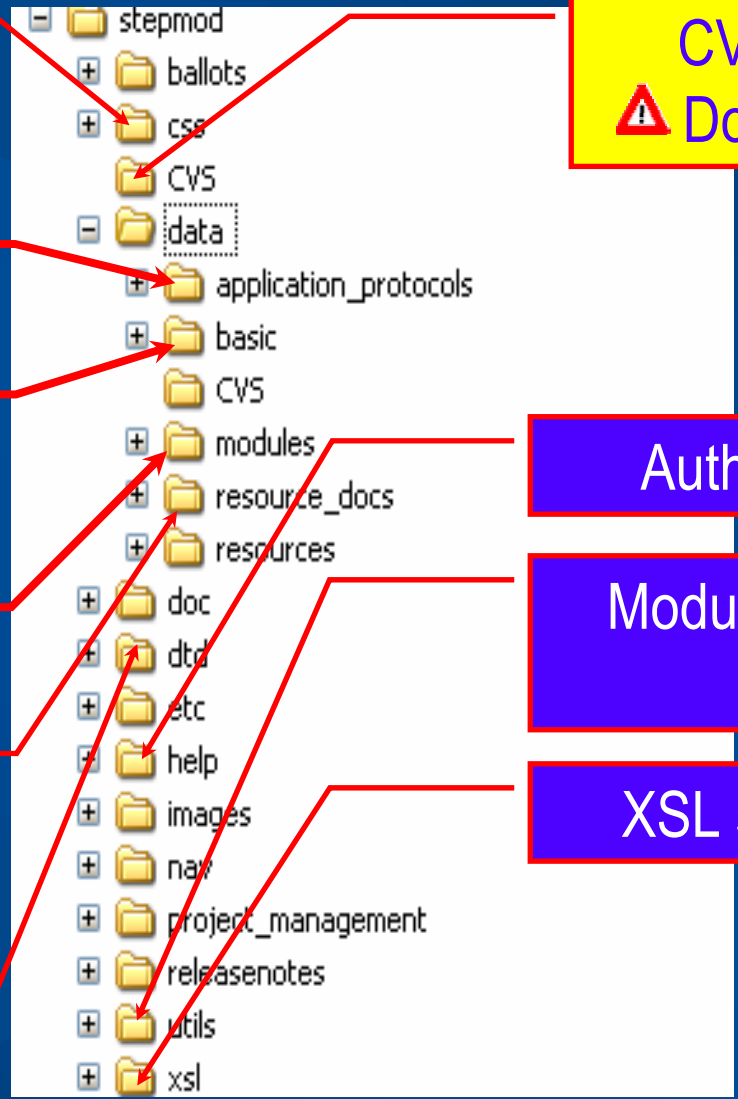
Application Protocols

Normative references contacts, bibliography

The modules

Integrated resource EXPRESS schemas

DTDs



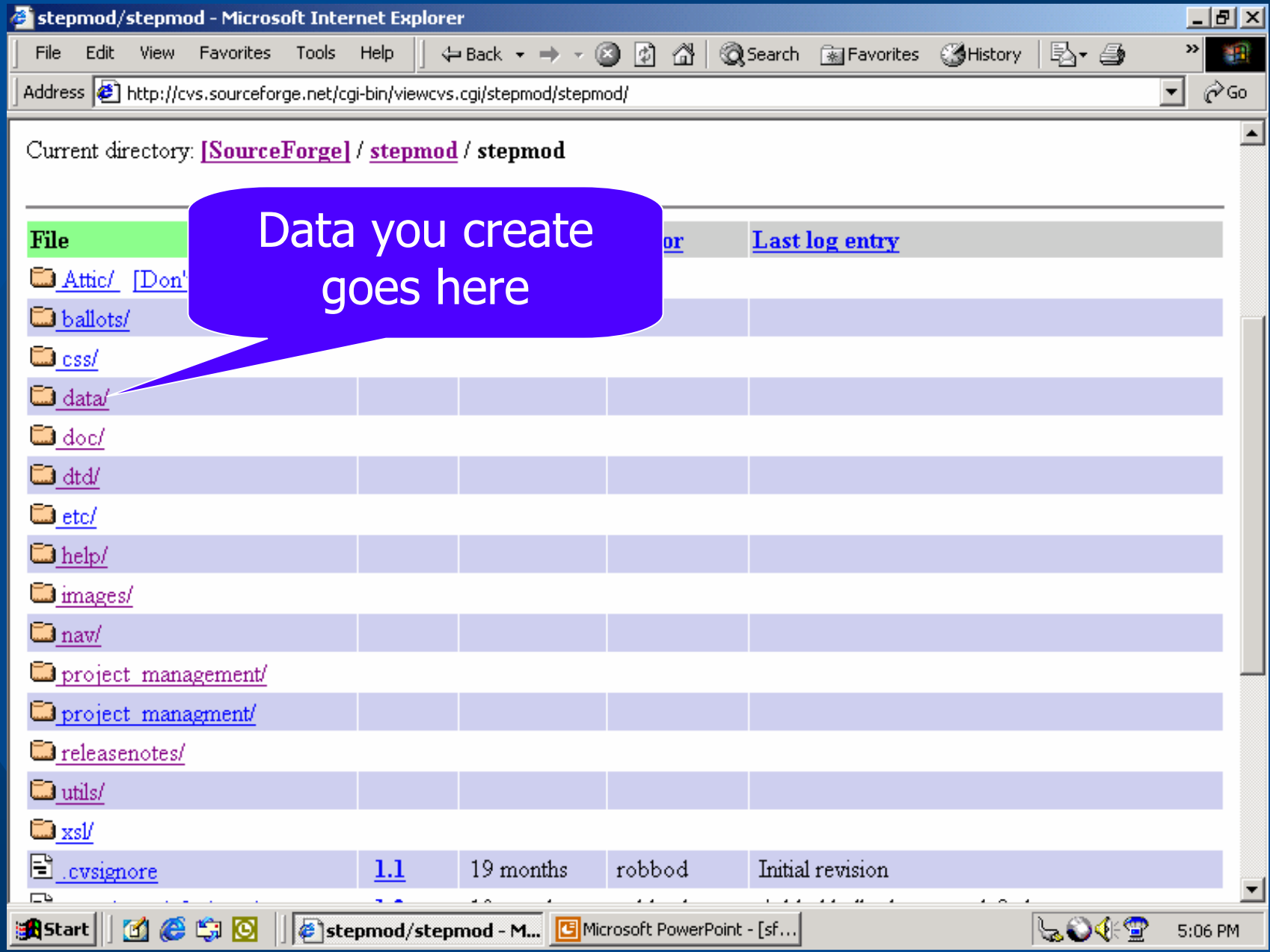
CVS directory
Do not delete

Authoring help

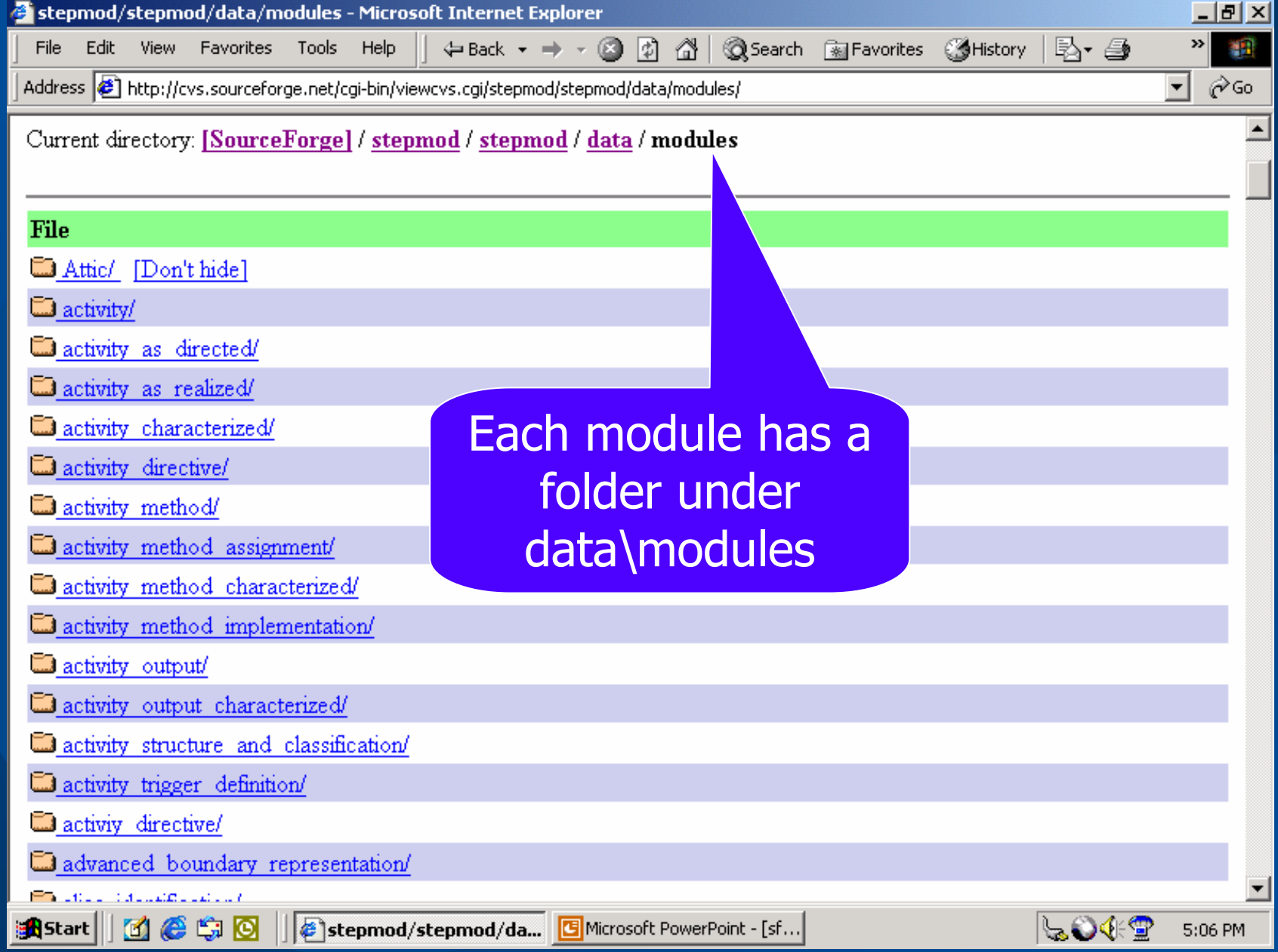
Module authoring tools

XSL stylesheets

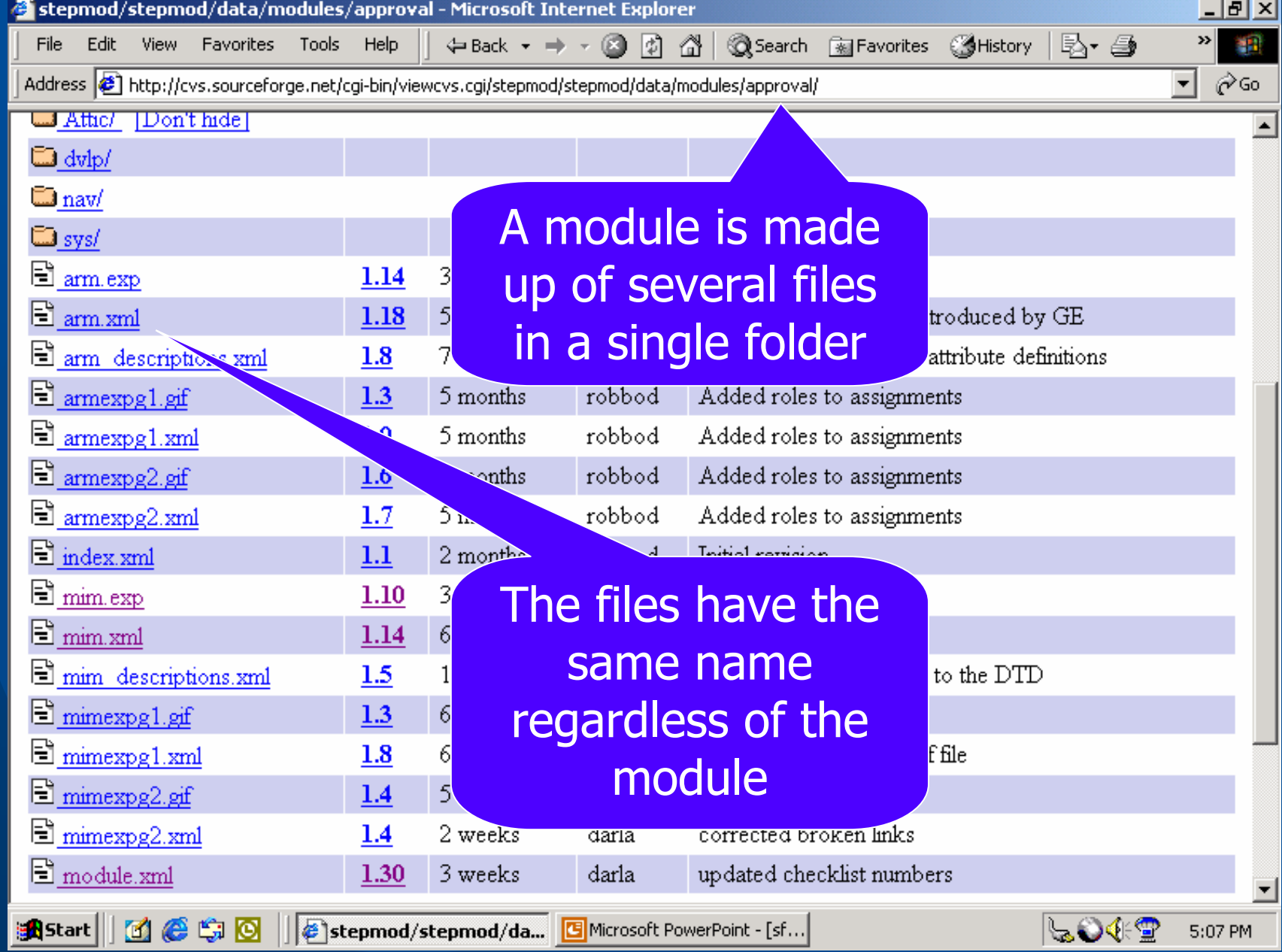
eurostep®



eurostep®



eurostep®



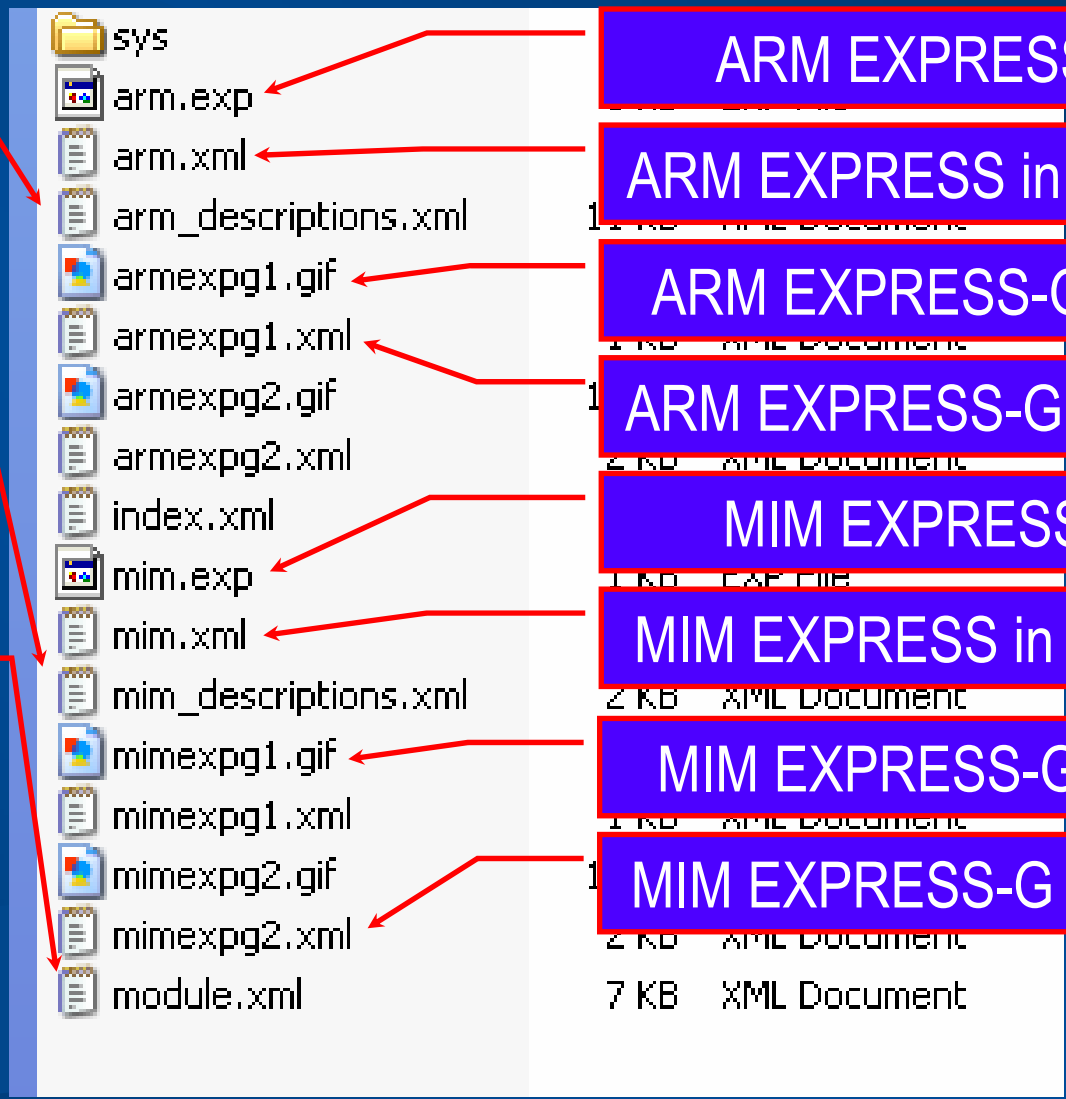
eurostep®

Module files

ARM schema documentation

MIM schema documentation

Module documentation
Scope
Introduction
Mapping

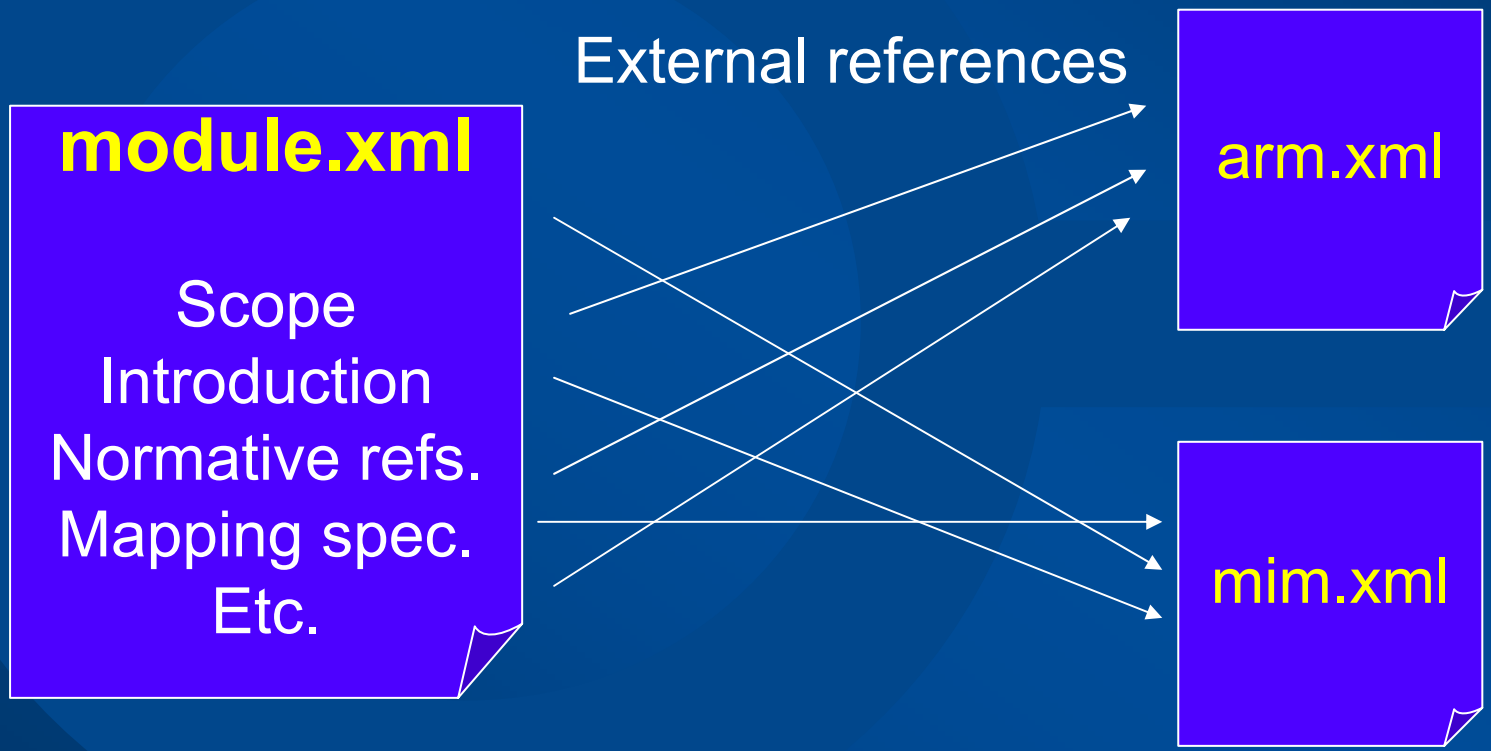


eurostep®

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



eurostep®

eurostep®

Overall Agenda

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

eurostep®

Building the STEP Module Content Section Agenda

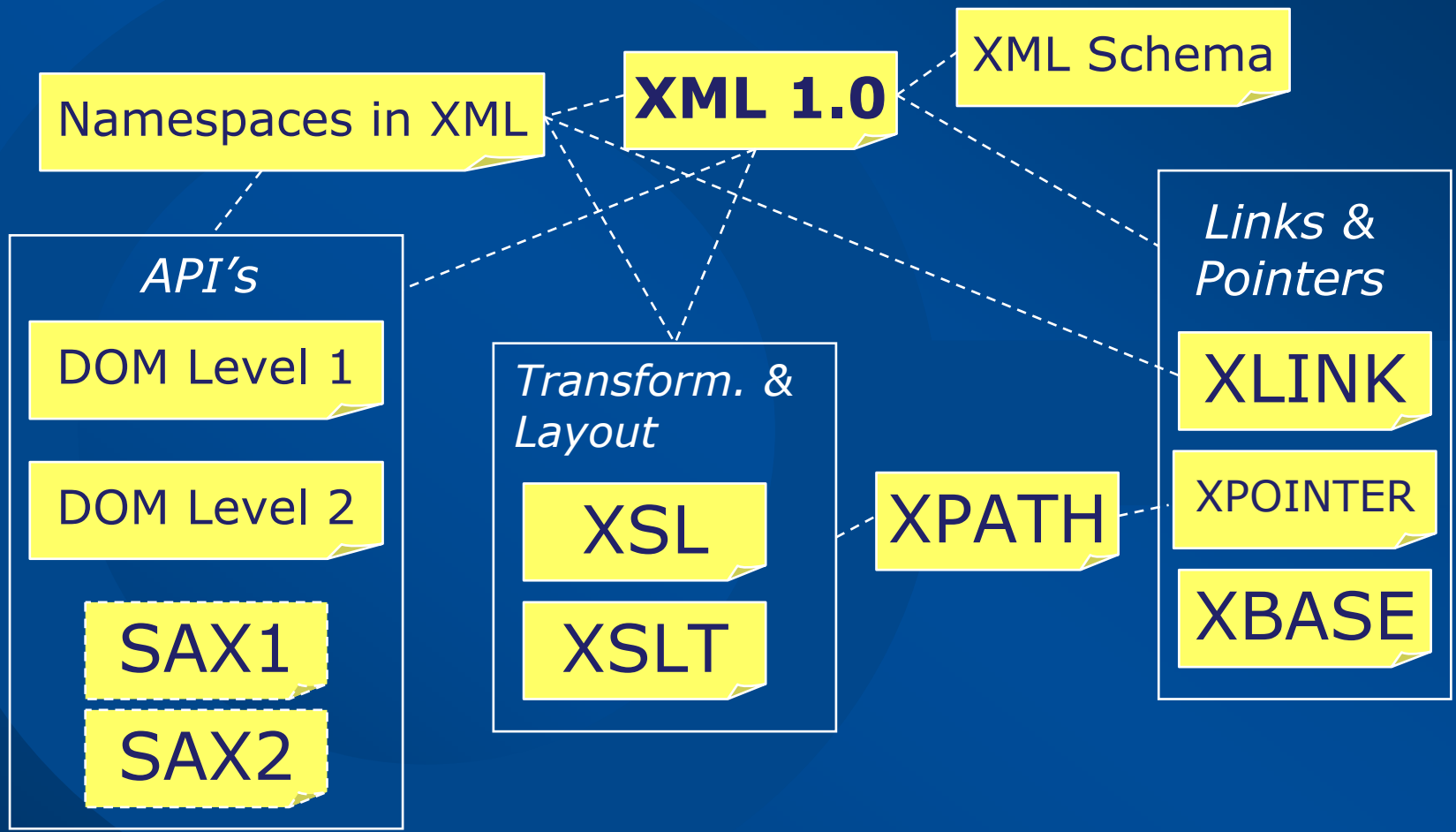
1. XML Basics
2. Module document structures
3. Building a module
4. Repository tools used to build a module

eurostep®

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



eurostep®

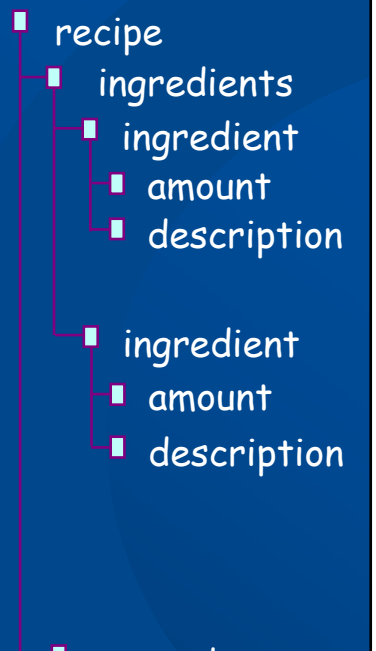
XML Basics

- 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 of 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

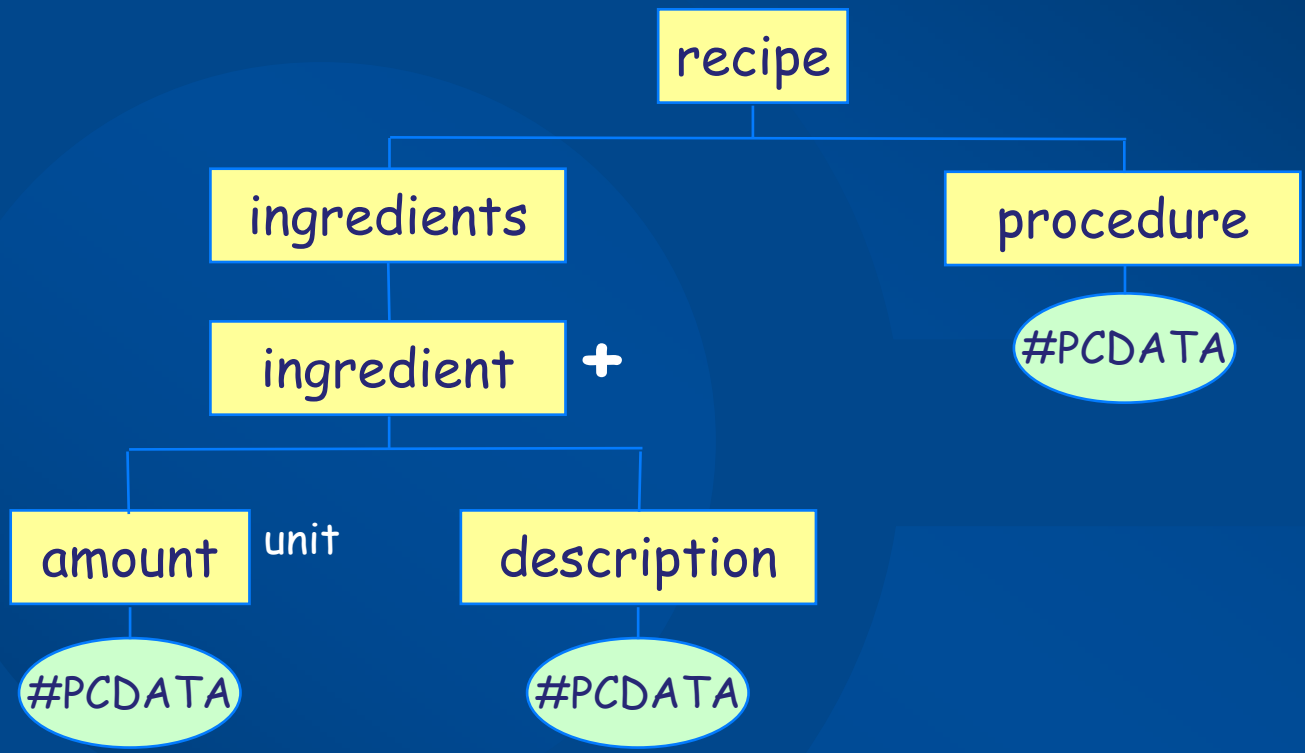
Attribute

Content

Root element end tag

eurostep®

DTD the "built-in" schema



eurostep®

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?)>

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

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

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

An "entity" contains "explicit" and other elements

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

eurostep®

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>
```

eurostep®

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

eurostep®

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 itsef
  - 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
  - armexpgn.xml and mimexpgn.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

```
<?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_lf?	– optional arm long form EXPRESS-G
mapping_table?	– optional mapping specification (see later slide)
mim	- MIM EXPRESS-G and shortnames
mim_lf?	– 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

eurostep®

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

File Edit View Favorites Tools Help

Back Search Favorites Media

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

```
<projlead ref="pics.projlead" />
<editor ref="timking" />
</contacts>
- <purpose>
- <p>
This part of ISO 10303 specifies an application mo
interfaces between products (see
<module_ref linkend="interface:introduction:figure:f
). 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_ref linkend="interface:arm:Interface_arm.Interface_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_ref linkend="interface:arm:Interface_arm.Interface_specification" />
is referred to as an Interface Control Document or Drawing.
```

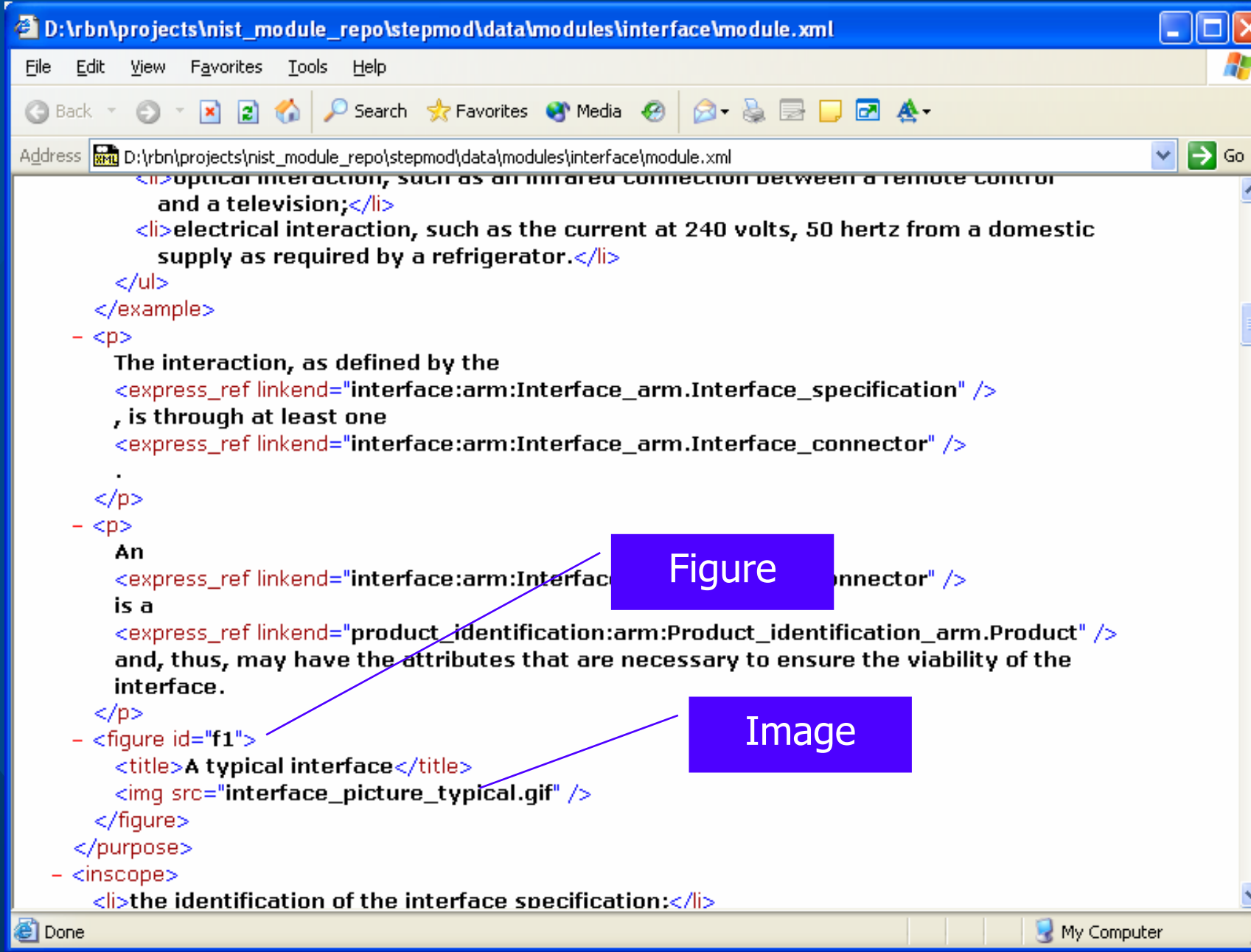
Link to Figure

Link to EXPRESS

Examples need numbers

Done My Computer

eurostep®



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_contraint
- ~~*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

eurostep®

Browsing a mapping table

```
- <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="pdmmodules.editor">
    <firstname>Darla</firstname>
   <lastname>Nettles</lastname>
```

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

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

eurostep®

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>
```

eurostep®

Bibliography

- bibliography
  - content: (bibitem.inc | bibitem)+
- bibitem – an item in the bibliography
  - same content as stdref
  - attribute: id (an identifer 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 bibligraphic item
  - no content
  - attribute: ref

Example normref and bibliography

```
checklist.conveller="1064" 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

```
- <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>
```

eurostep®

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>
```

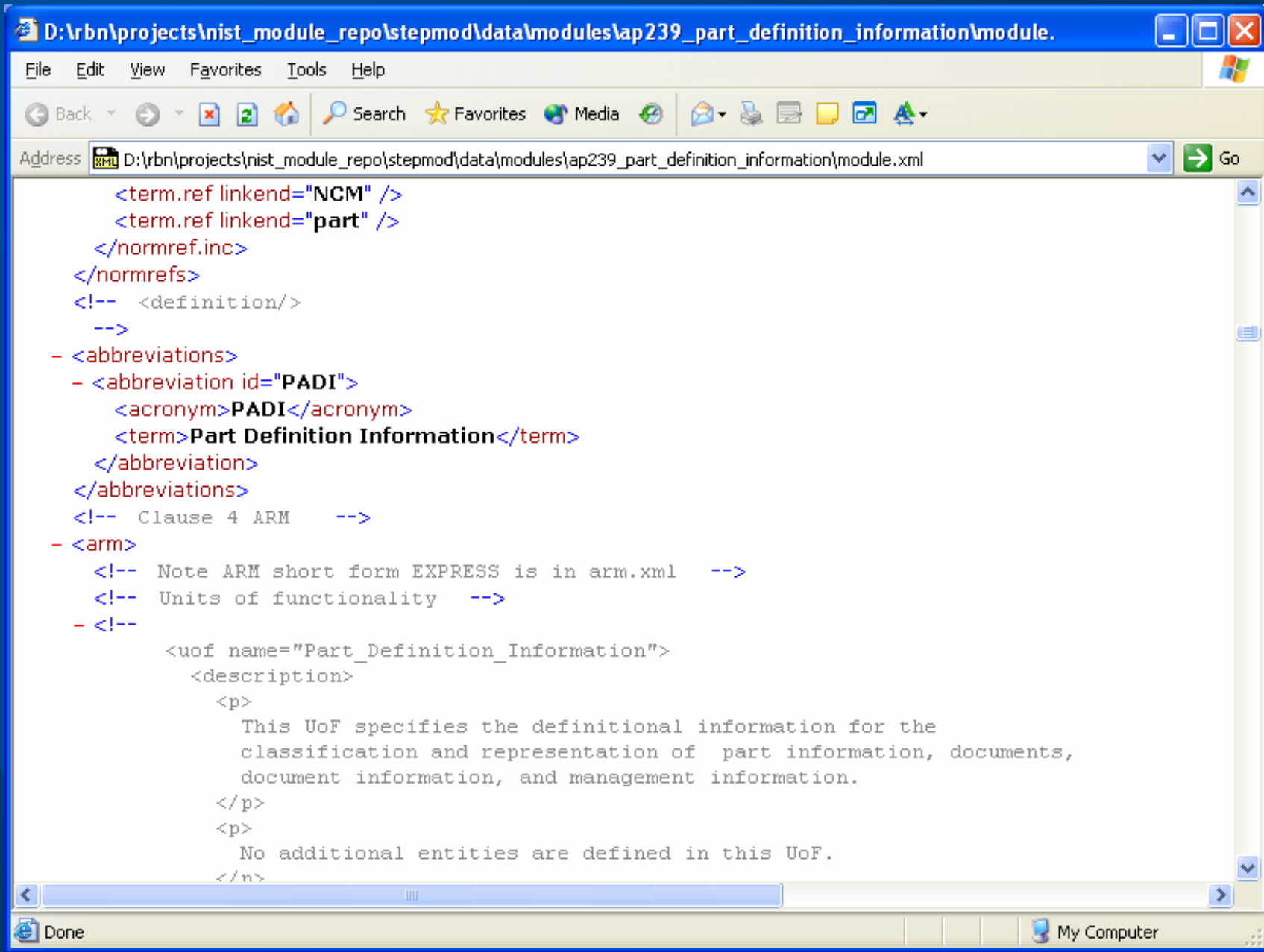
stepmod\data\modules\[module]\module.xml

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

eurostep®

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



The Module Files

- ARM and MIM EXPRESS
  - arm.exp and mim.exp – The EXPRESS text itsef
  - 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
  - armexpgn.xml and mimexpgn.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.contraint – new in EXPRESS 2
  - function – just what is says (algorithm not exploded!)
  - procedure – just what is 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

```
- <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

```
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;
```

eurostep®

external descriptions

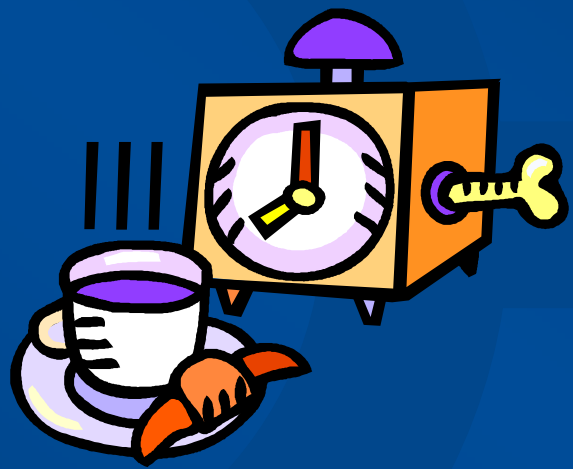
- description in one file references EXPRESS in another
  - arm_description.xml and mim_description.xml
- ext_descriptions
  - attributes: module_directory, schema_file, rcs.date, rcs.revision, describe.selects, describe.subtype_constraints
  - content: ext_description*
- ext_description
  - attributes: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

An example external description

```
- <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 src="armexpg1.gif">
      <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>
```



eurostep®

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

eurostep®

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

eurostep®

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

eurostep®

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

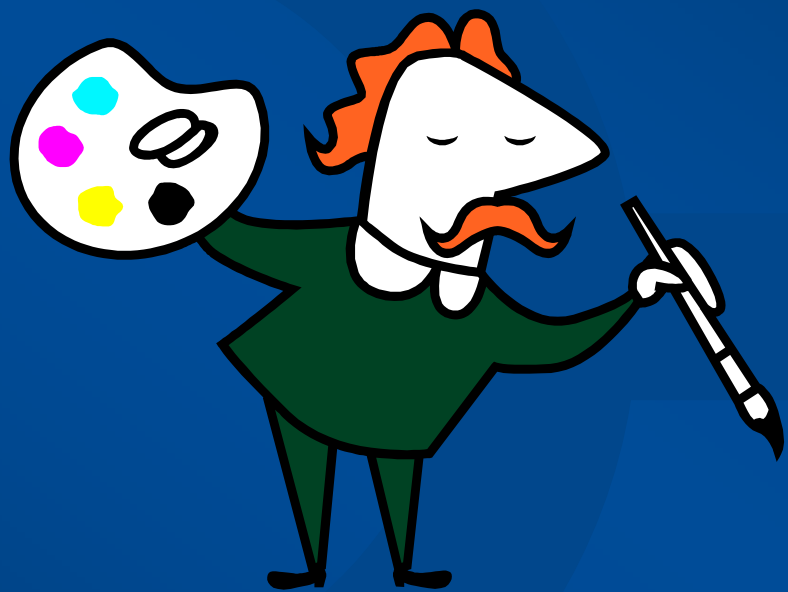
- ARM entities start with Upper case.
  - E.g Activity
- MIM entities are lower case
  - E.g. action
- All types, in ARMs and MIMs, are lower case.
  - E.g. activity_item

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



eurostep®

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

eurostep®

Prequisites

- The utility functions are JScripts 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.asp?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

- Issues
- System folders
- ARM files
- MIM files
- Module template

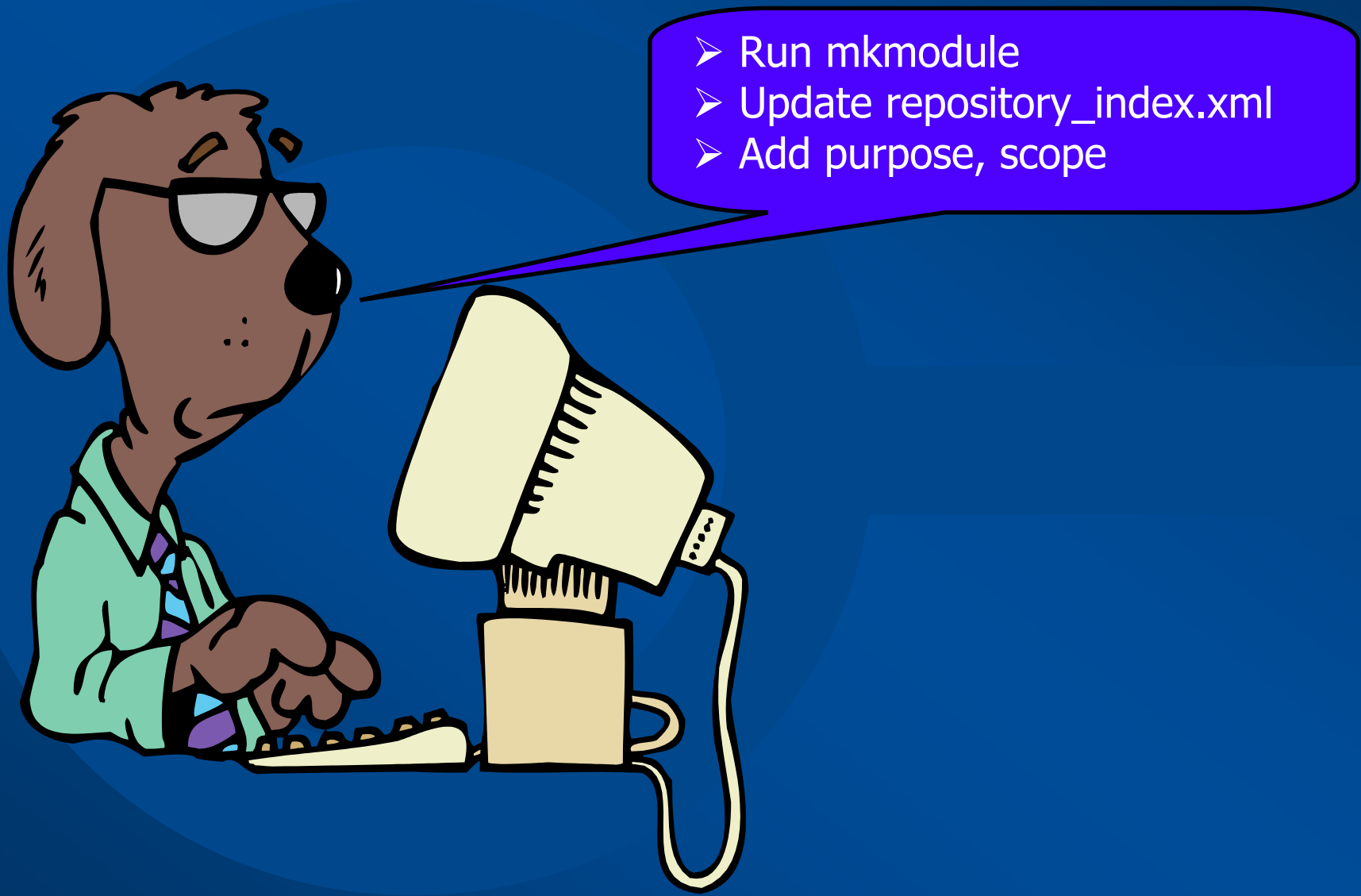
eurostep®

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



eurostep®

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

eurostep®

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

eurostep®

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>
```

eurostep®

Demo GraphicalEXPRESS

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

eurostep®

Add EXPRESS descriptions

- Descriptions can be embedded in arm.xml

<entity name="Classification">

*<description> A classification is a relationship between a class and a thing, that indicates the thing is a member of the class. *

<note>The meaning of this entity is identical to the entity

classification 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*

eurostep®

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 <b>Activity</b> is the identification of the occurrence
  of an activity.
</ext_description>
```

eurostep®

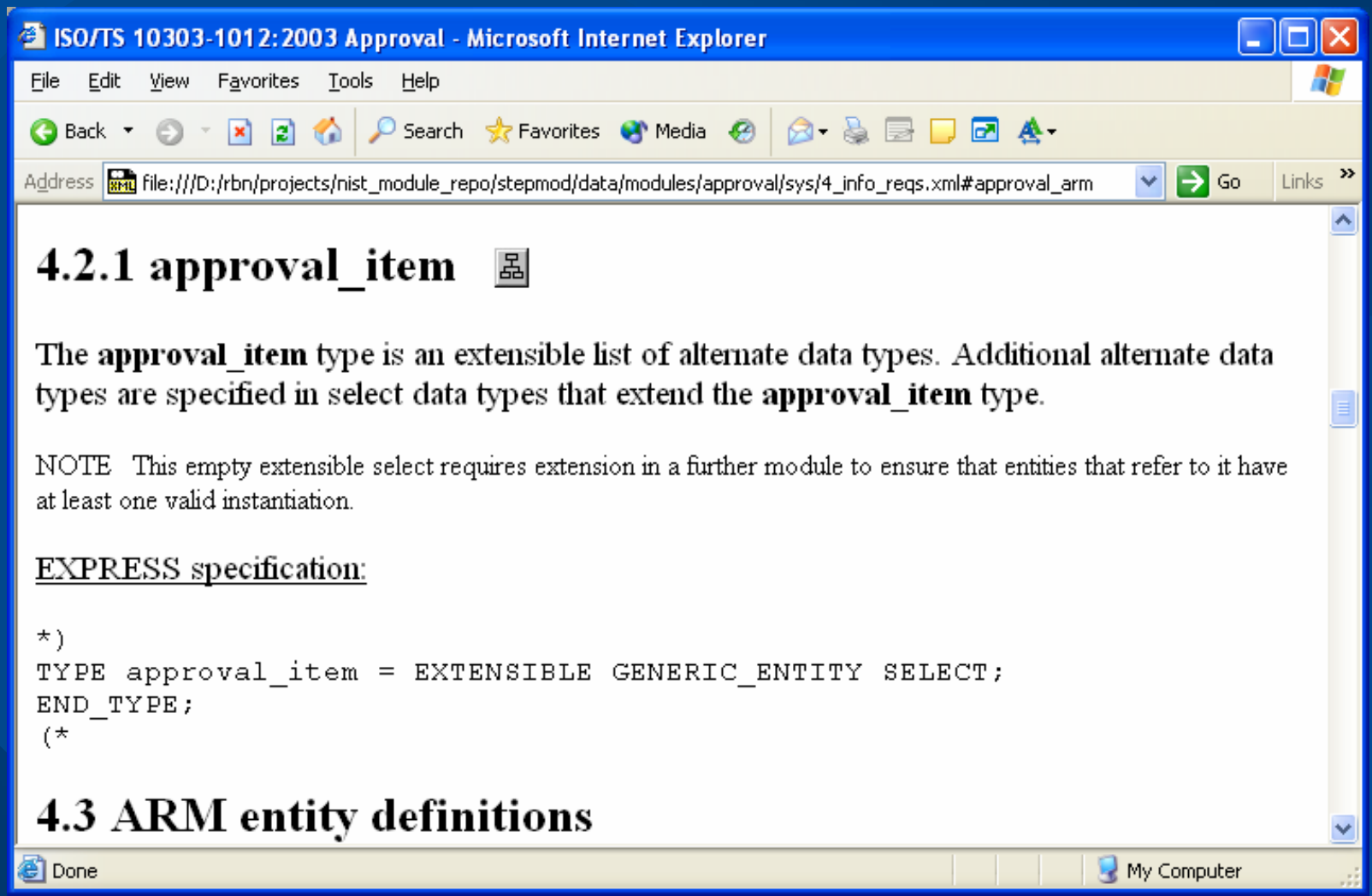
Automated select descriptions

```
<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">
```

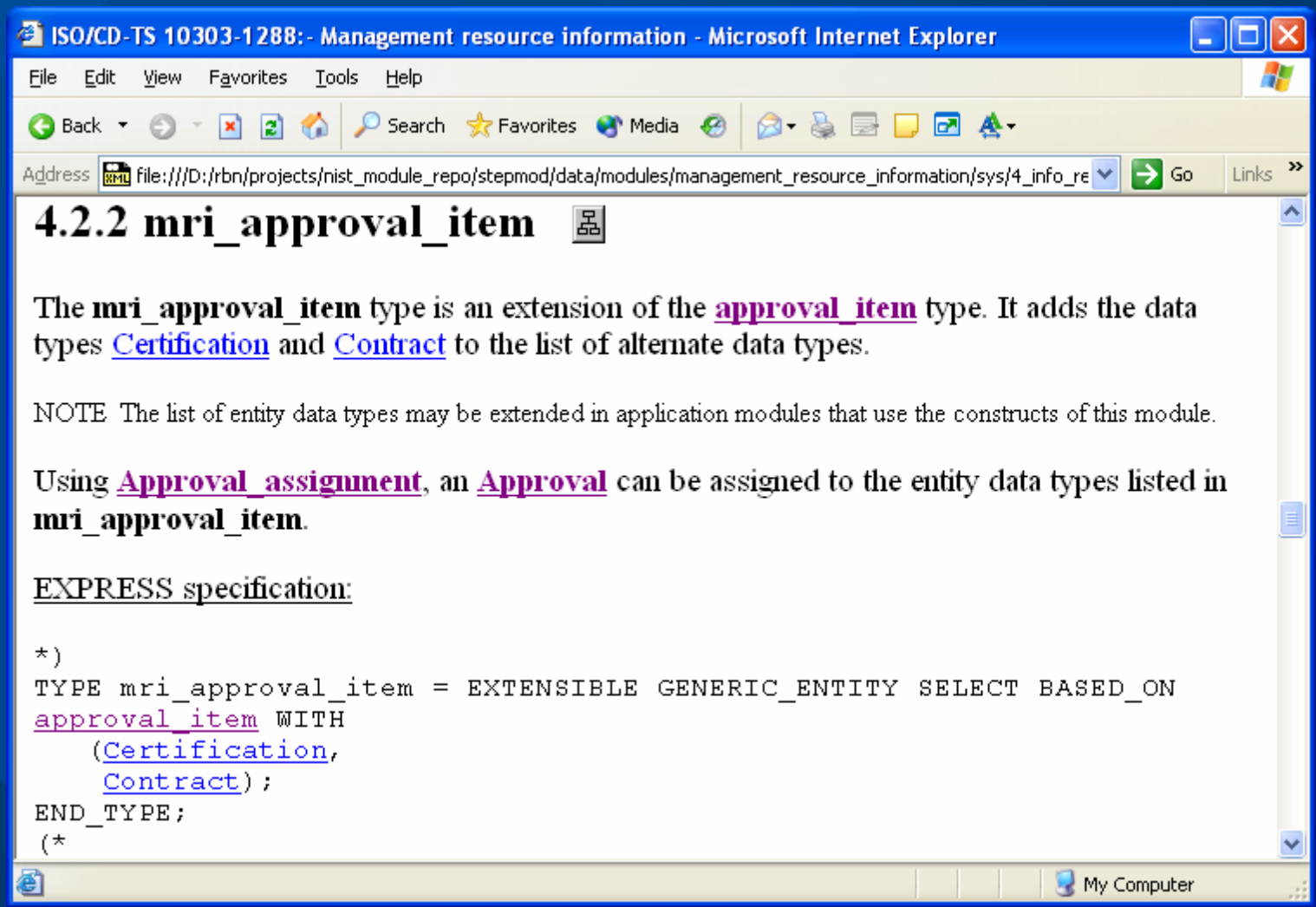
- No description necessary
- XSL does the work

eurostep®

E.g. Empty extensible select



E.G populated extensible select



eurostep®

Demo extractDescriptions

eurostep®

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

eurostep®

Mapping

Mapping defined in module.xml

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

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

ARM entity being mapped

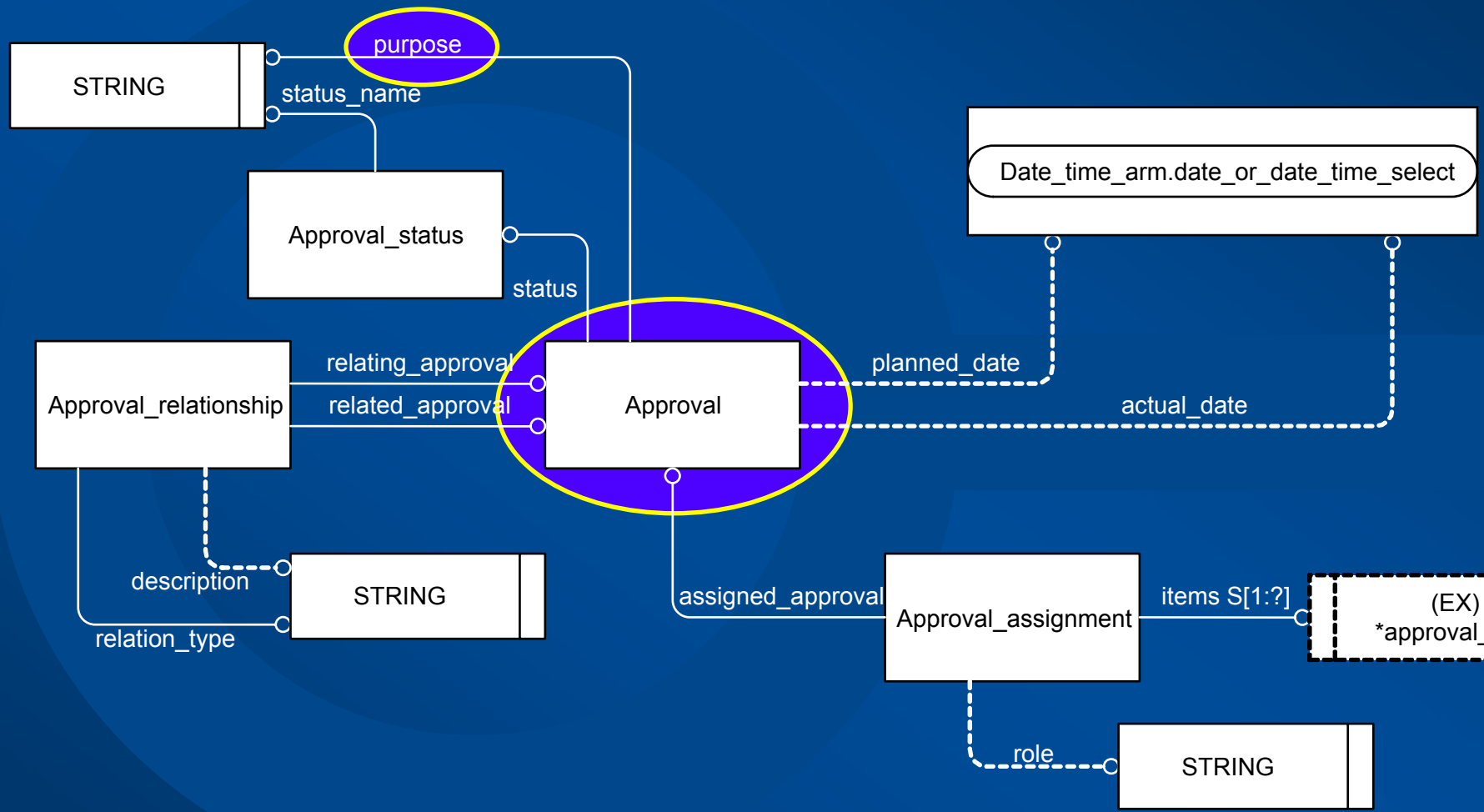
MIM entity mapped to

Document in which MIM entity defined

Mapping reference path

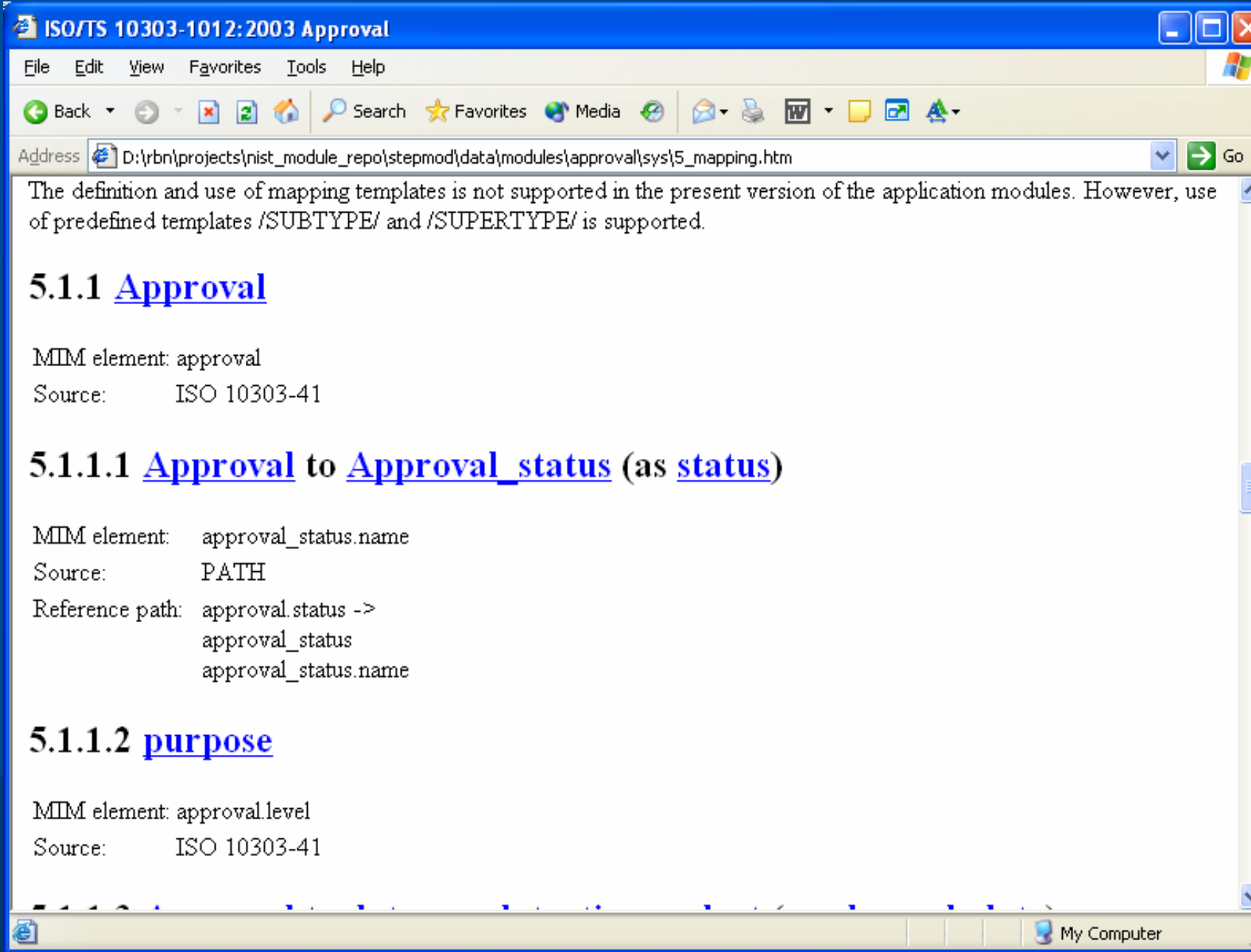
ARM attribute (simple type being mapped

Mapping

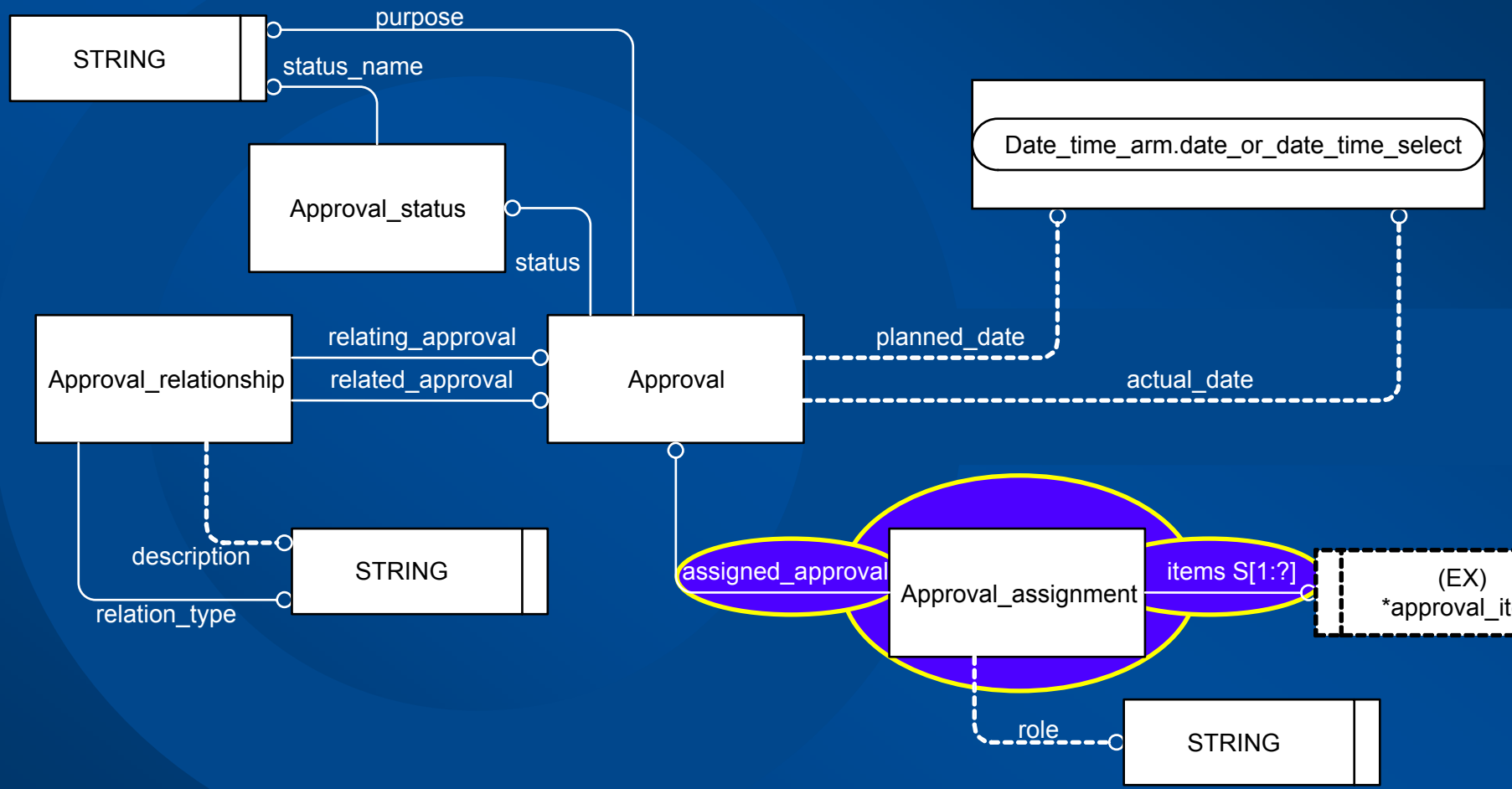


Mapping of Entities + attributes of simple type

```
<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>
```



Mapping



Mapping of assertions

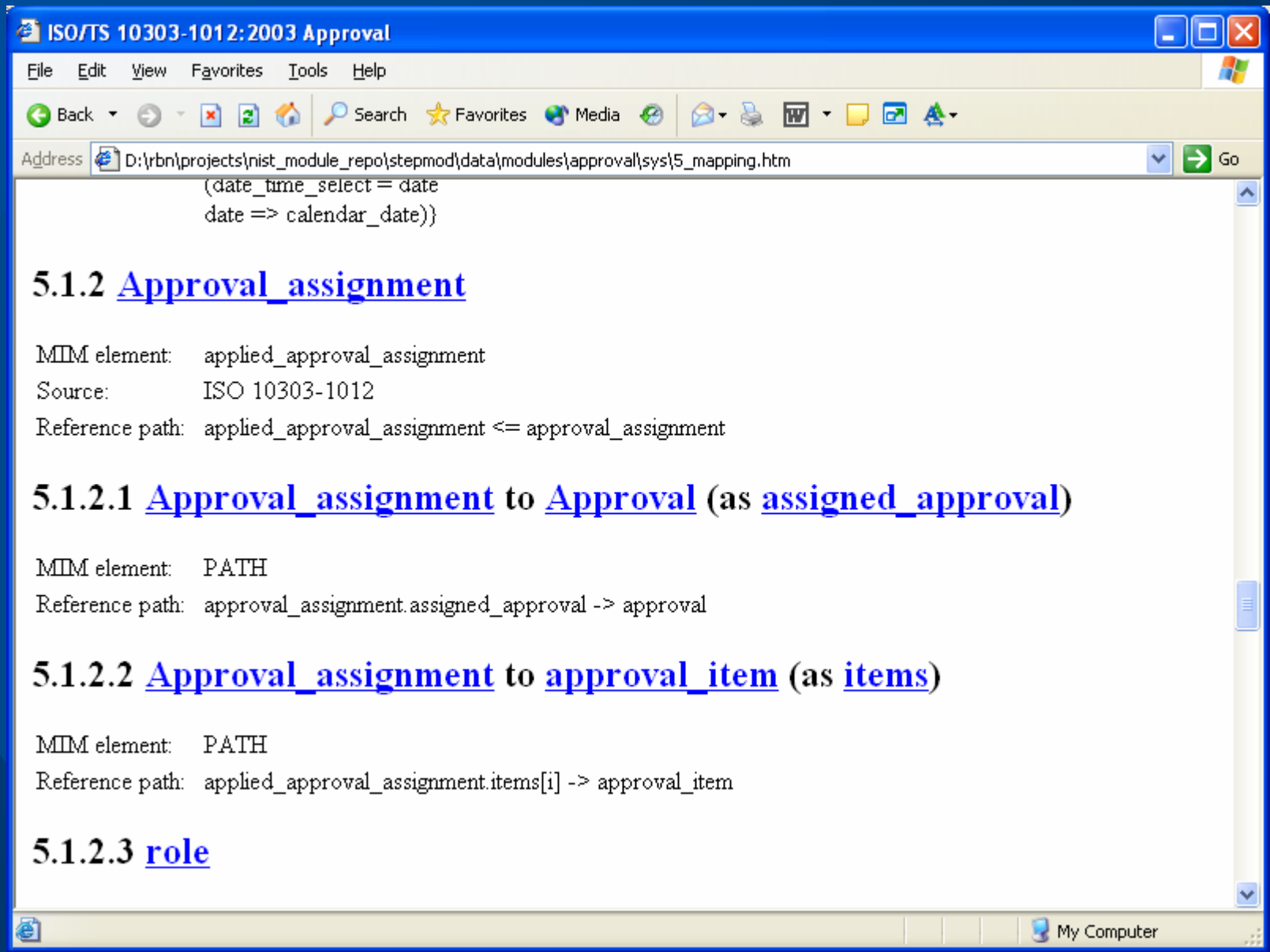
```
<ae entity="Approval_assignment">
 <aimelt>applied_approval_assignment</aimelt>
  <source>ISO 10303-1012</source>
  <refpath>
   applied_approval_assignment &lt;= approval_assignment
  </refpath>

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

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

Entity mapping

< is <
> is >

Assertion to an Entity

Assertion to a Select

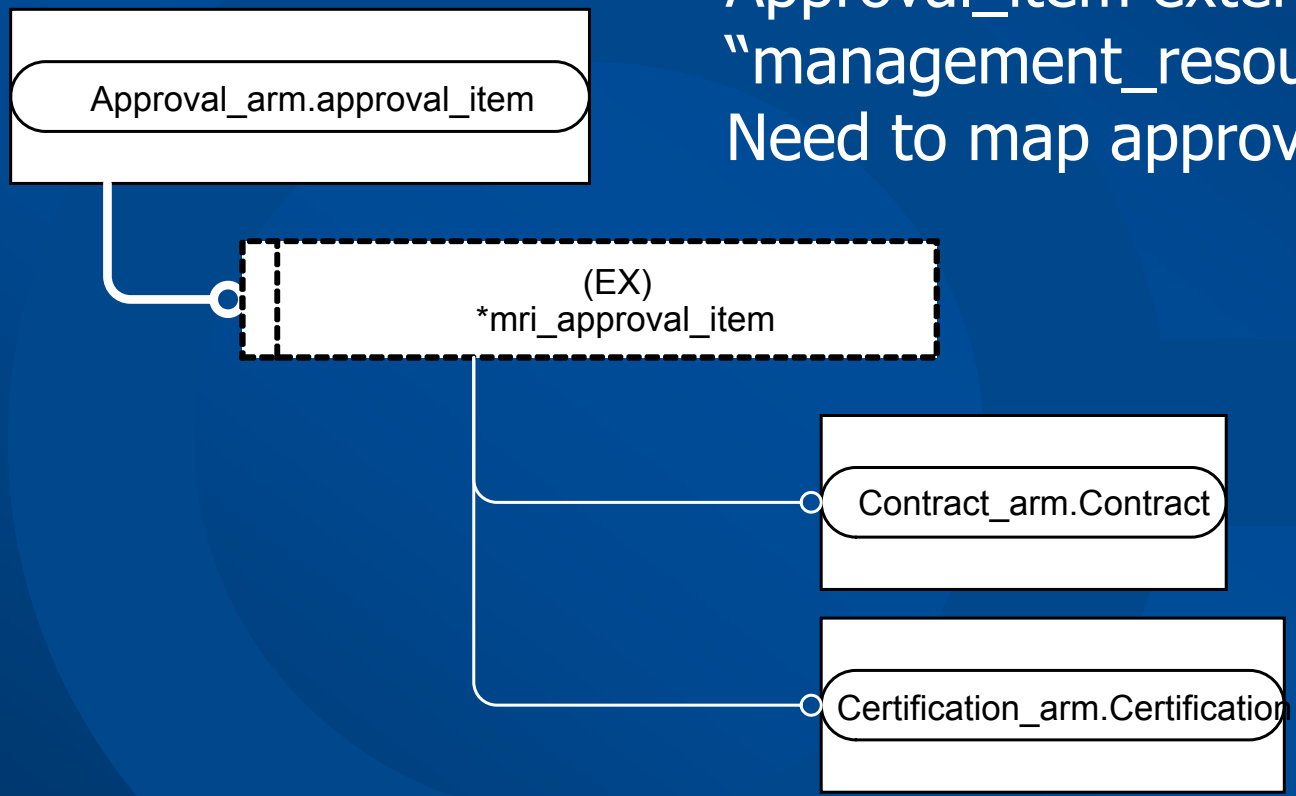


Mapping of extensible selects

- 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



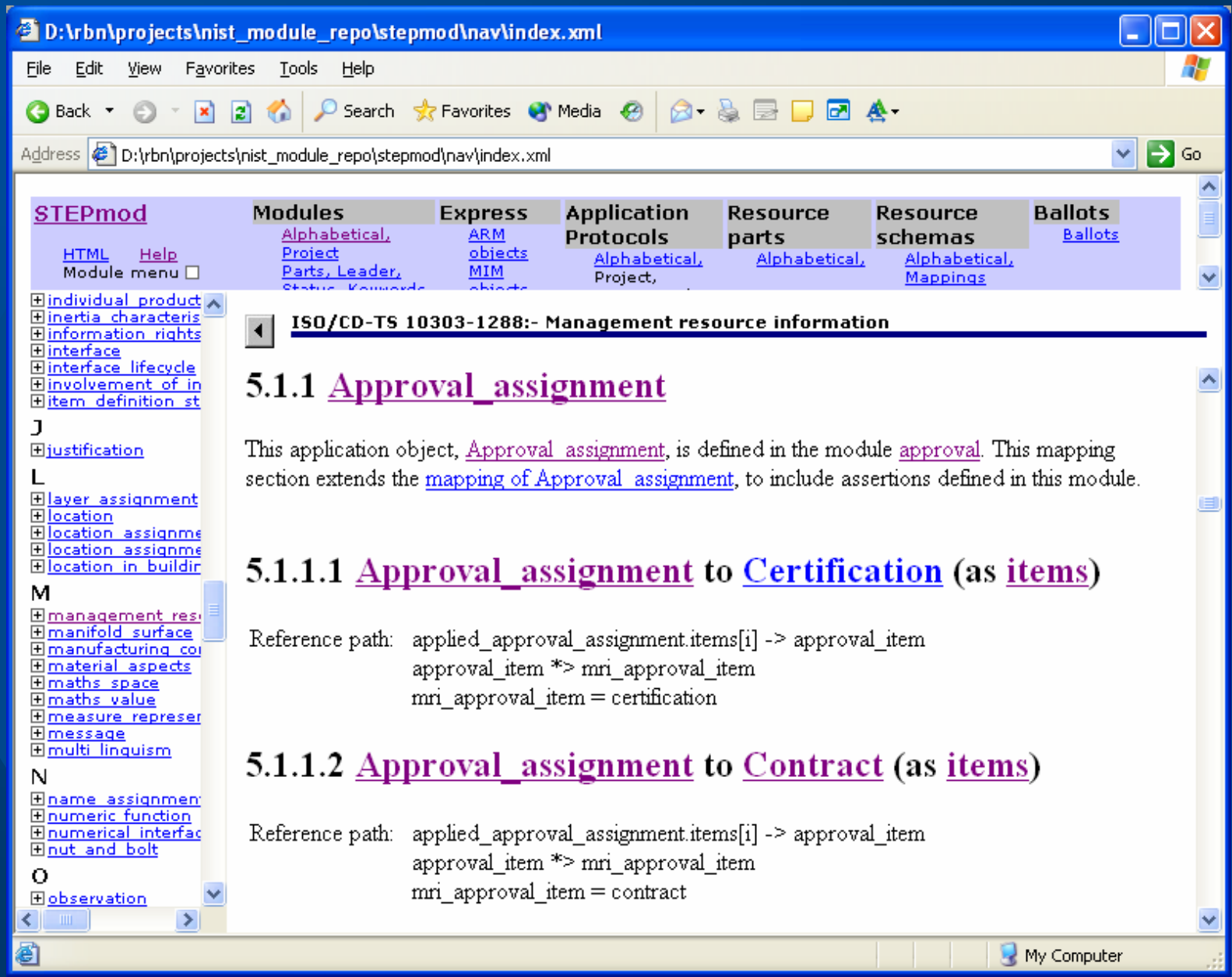
Mapping of extensible selects

```
<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>

   <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>
```

Module in which initial mapping to select defined

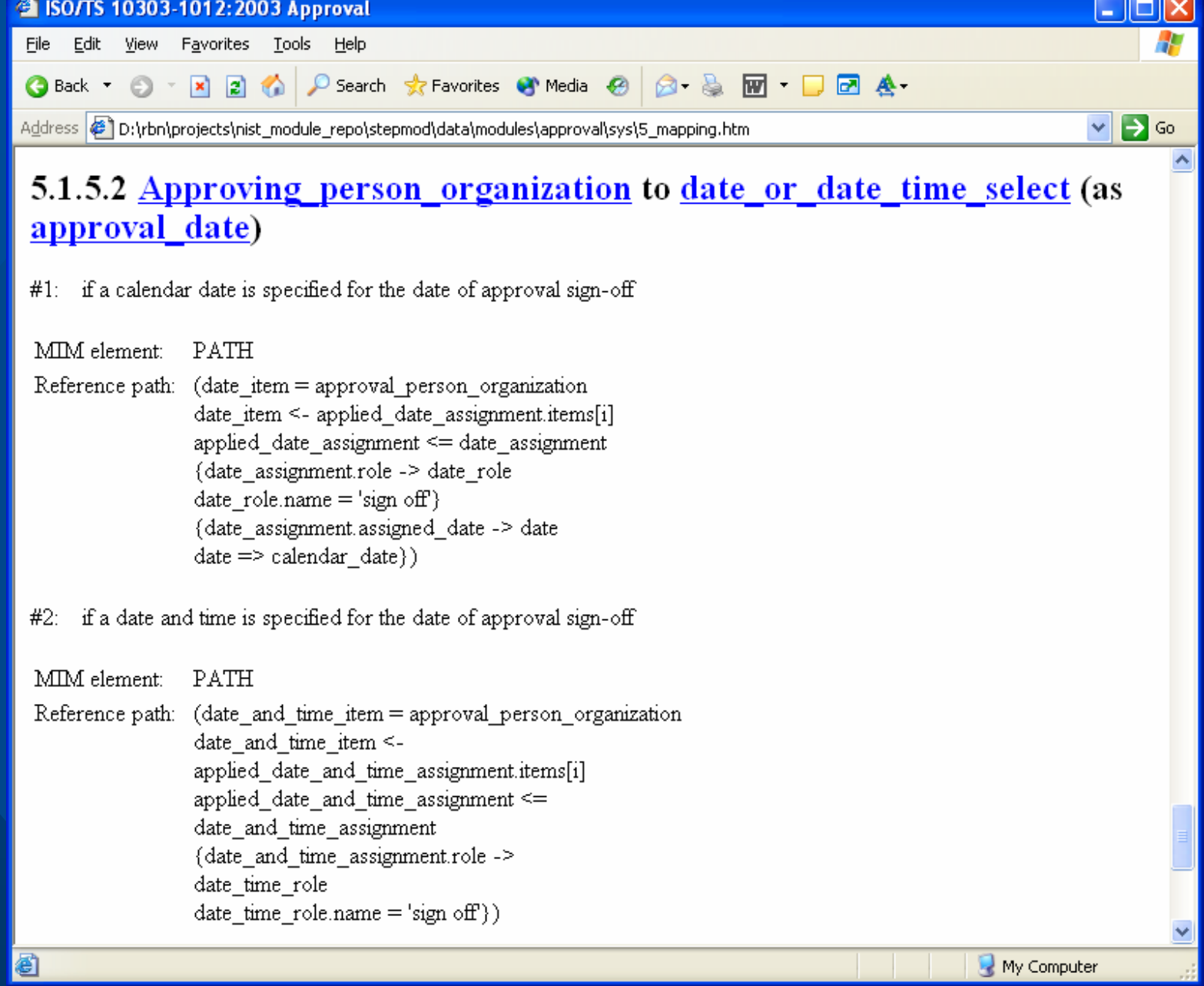
ARM select being extended

MIM select being extended



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>
```



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">
  .....
```

eurostep®

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

eurostep®

Demo Mapping

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

eurostep®

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

eurostep®

Contacts

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

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

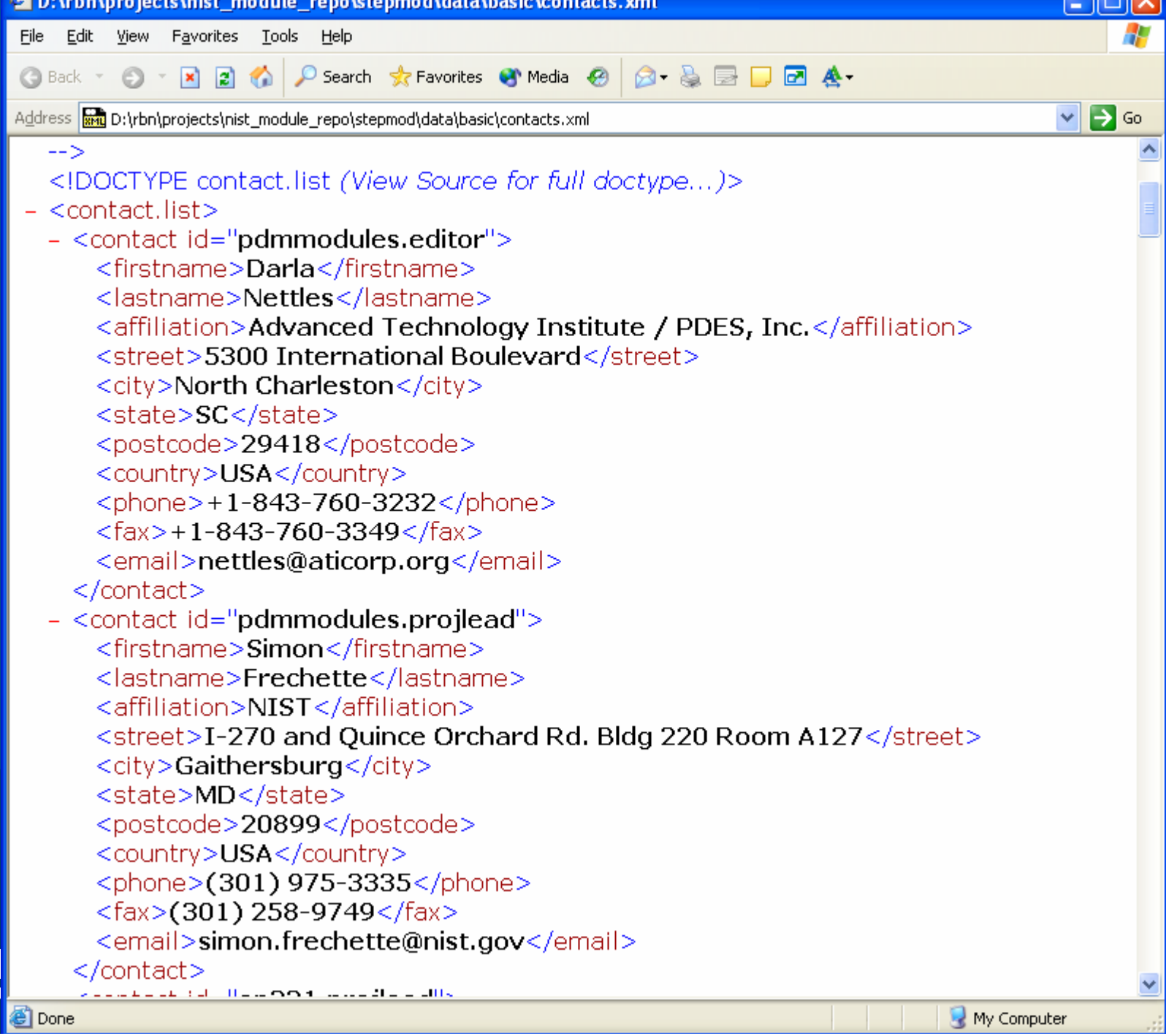
stepmod\data\basic\contacts.xml

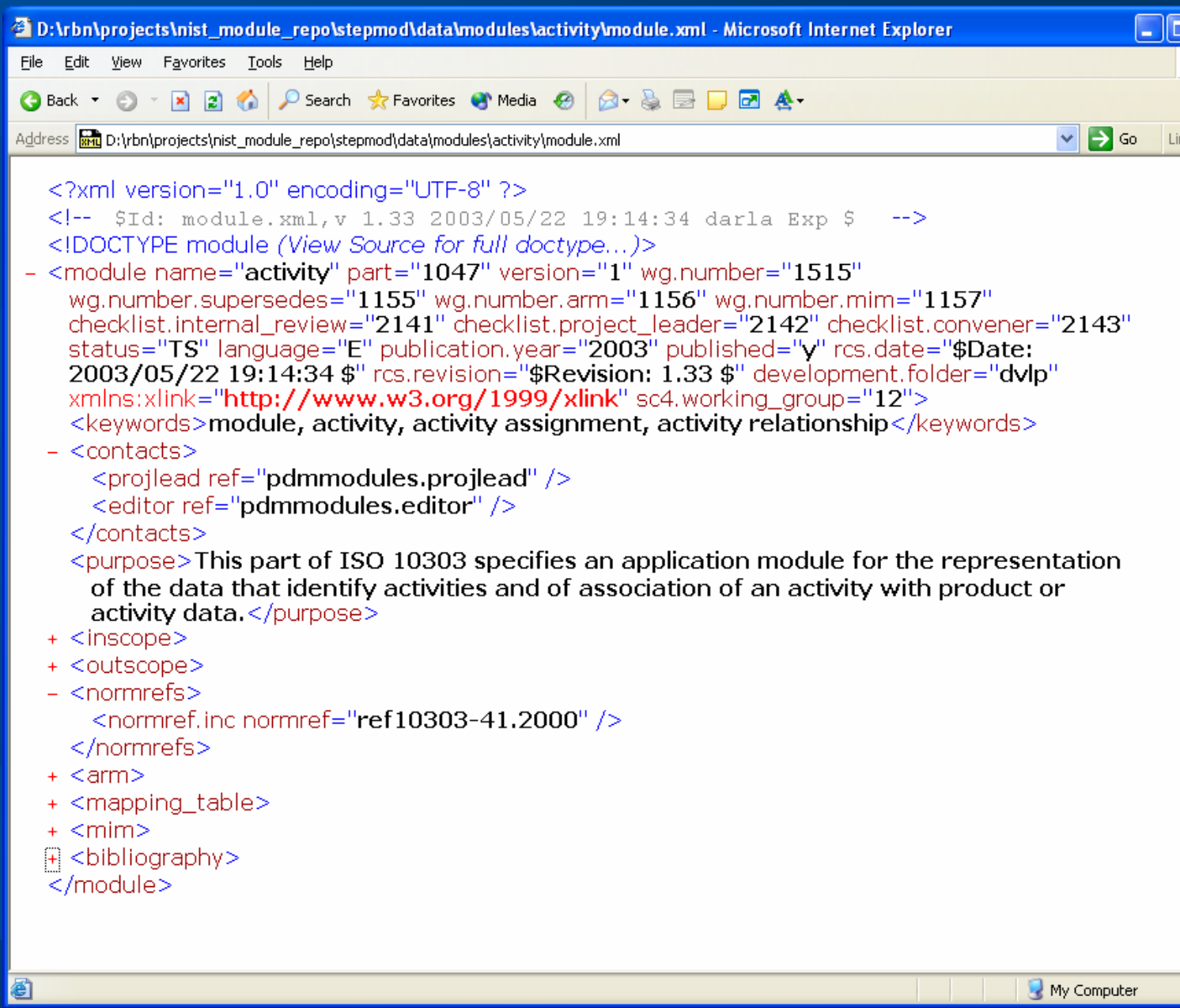
```
<contact.list>
  <contact id="pdmmodules.editor">
    <firstname>Darla</firstname>
   <lastname>Nettles</lastname>
```

module.xml

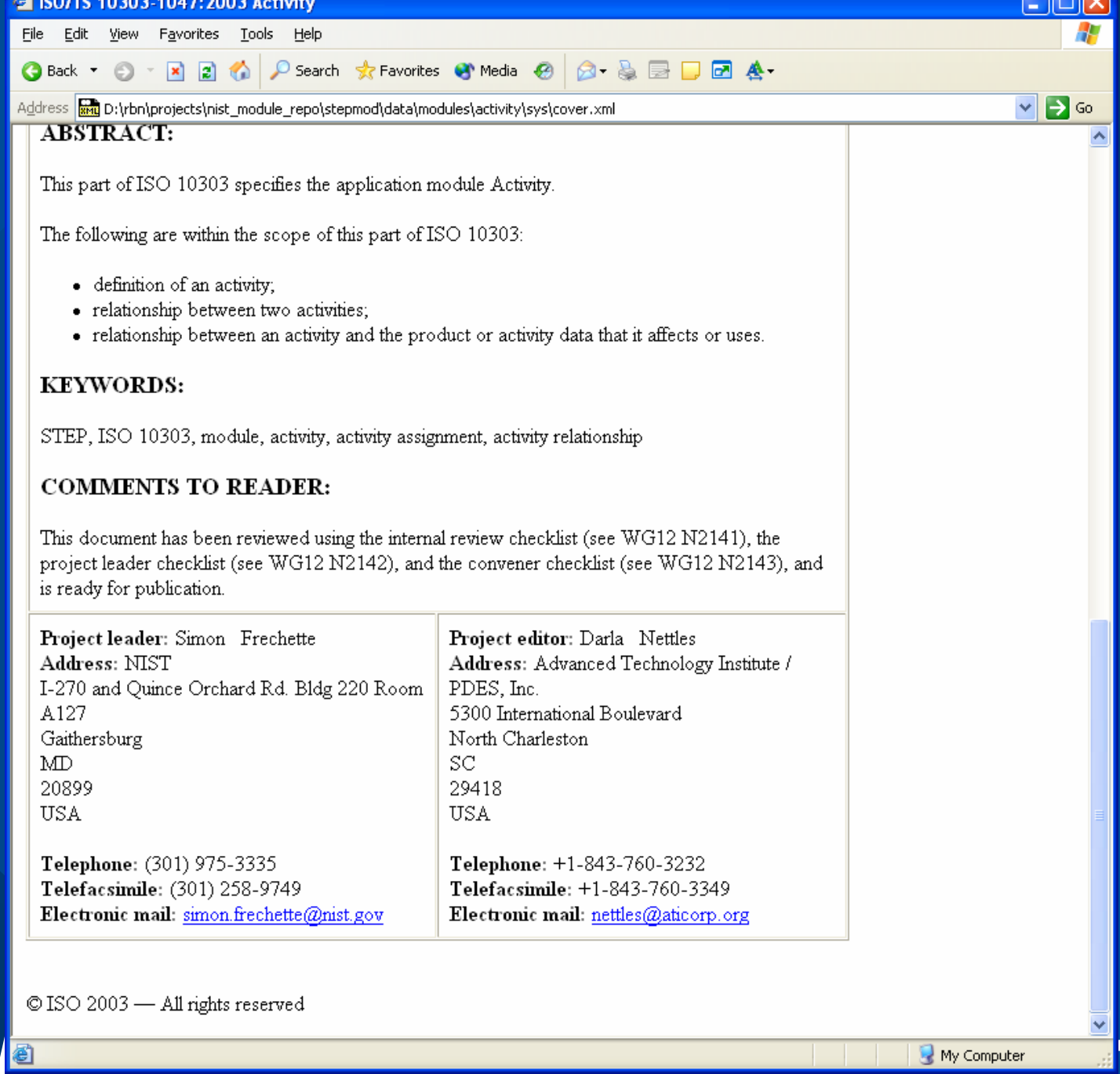
```
<contacts>
  <projlead ref="pdmmodules.projlead"/>
  <editor ref="pdmmodules.editor"/>
</contacts>
```

eurostep®


D:\rbn\projects\nist_module_repo\stepmod\data\basic\contacts.xml
File Edit View Favorites Tools Help
Back Search Favorites Media
Address D:\rbn\projects\nist_module_repo\stepmod\data\basic\contacts.xml
Go
-->
<!DOCTYPE contact.list (View Source for full doctype...)>
<contact.list>
<contact id="pdmmodules.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="pdmmodules.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>
Done
My Computer



eurostep®



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

eurostep®

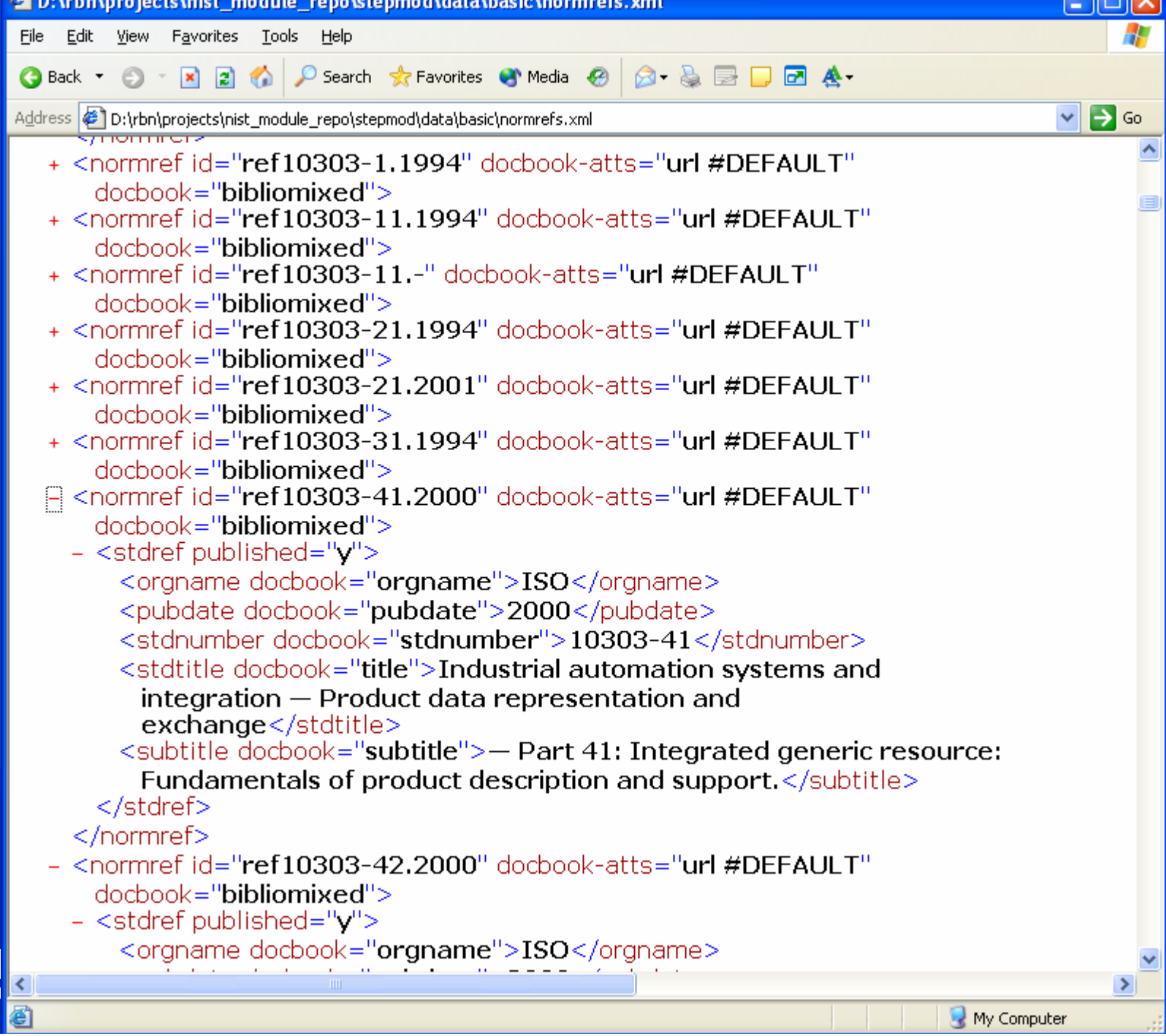
Normrefs

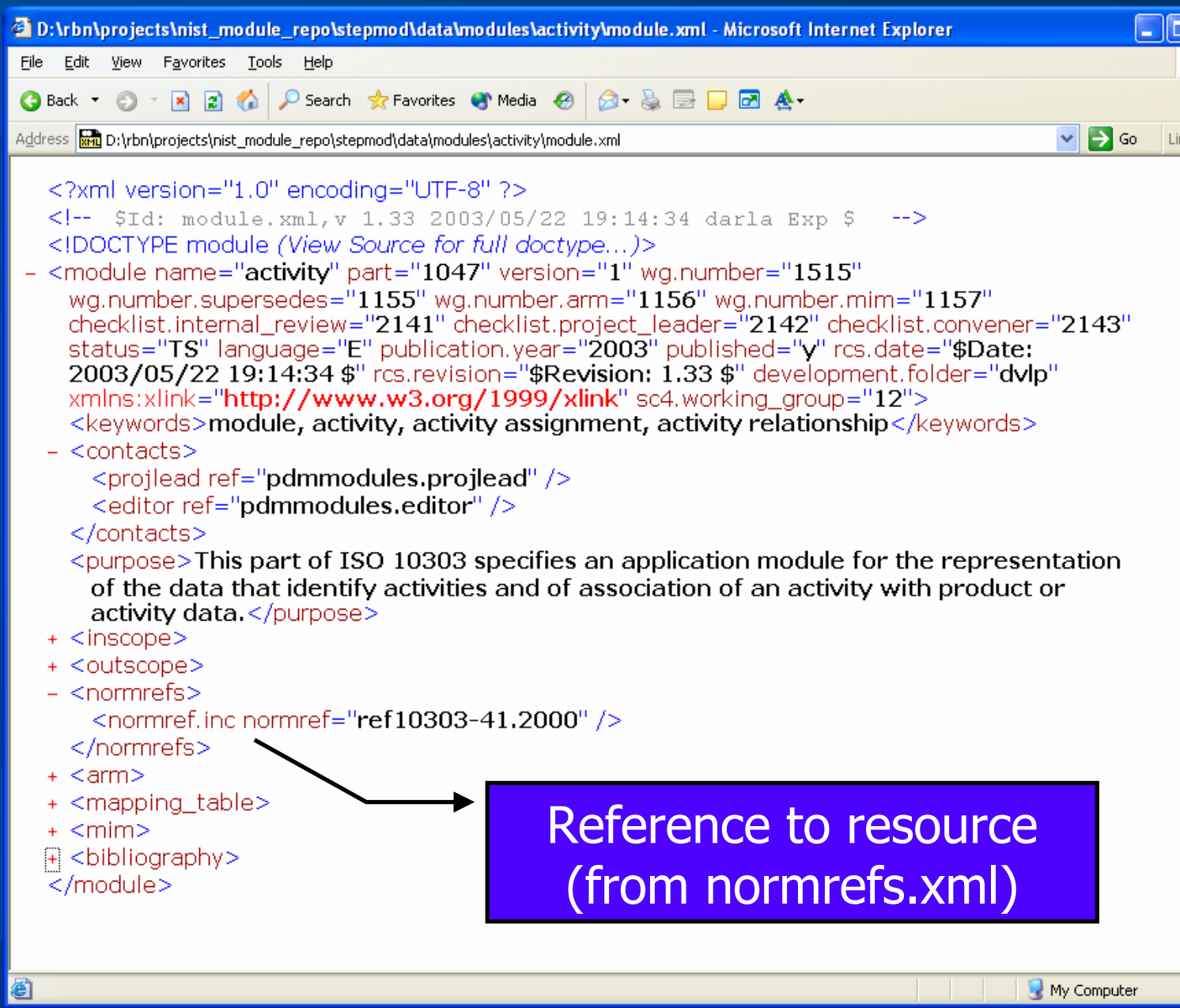
- Normative refs for all modules used in EXPRESS added automatically
- Manually add any Resource parts used to module.xml
- Manually add any module referenced in scope
- Normative refs stored in: stepmod\data\basic\normrefs.xml
- Referenced from module.x

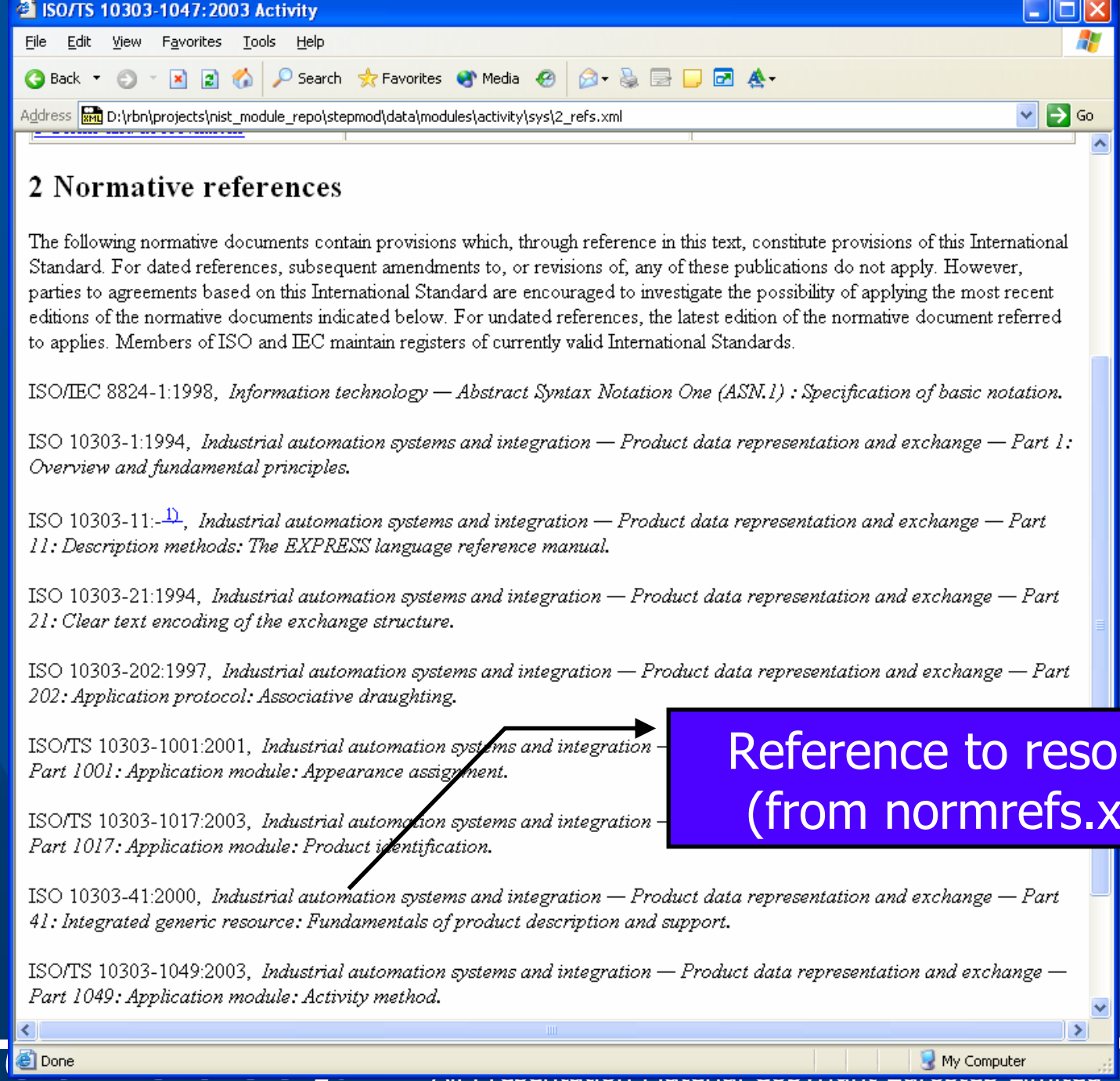
```
<normrefs>
 <normref.inc
  module.name="product_identification"/>
  <normref.inc normref="ref10303-41.2000"/>
</normrefs>
```

Reference to module

Reference to resource (from normrefs.xml)

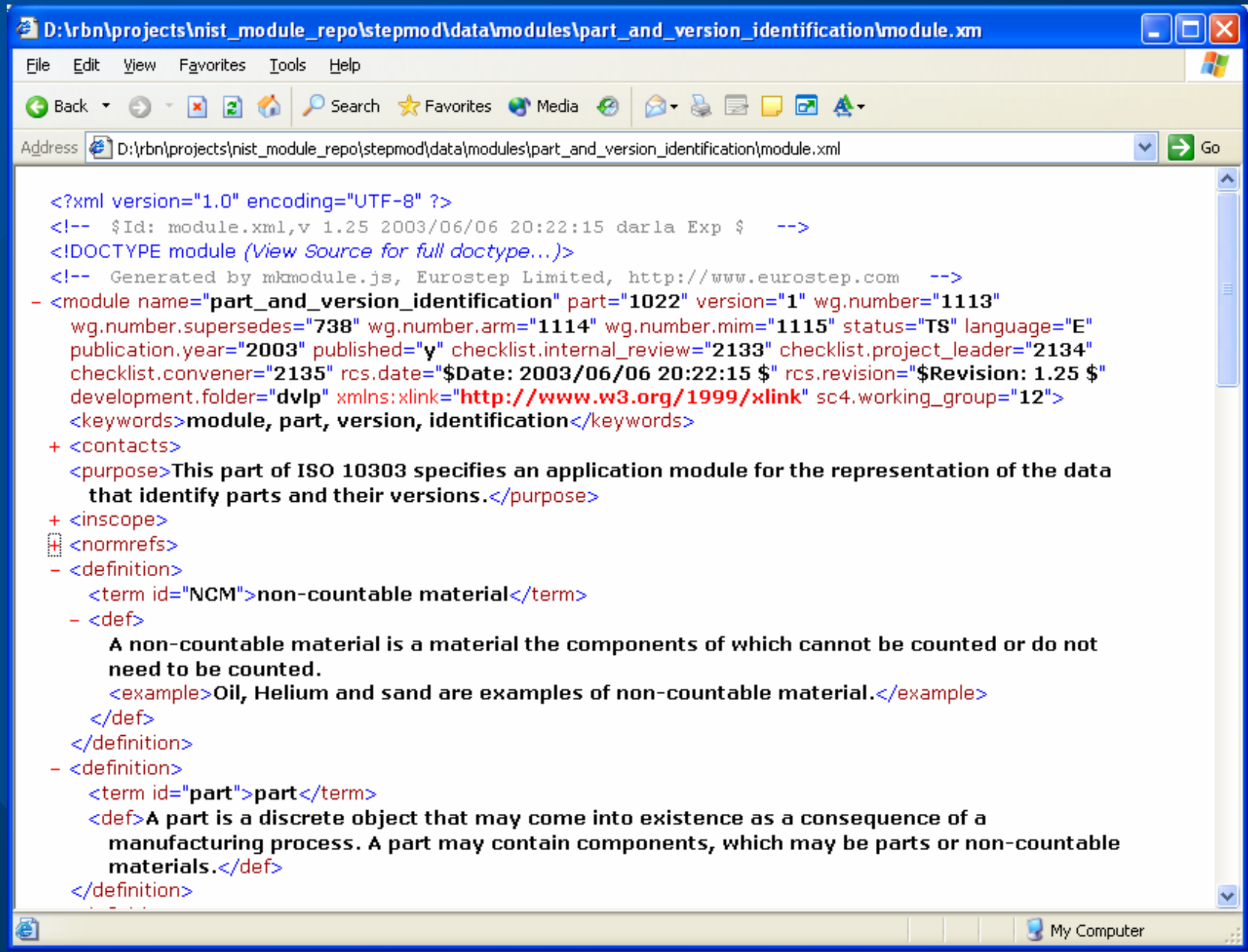

D:\rbn\projects\nist_module_repo\stepmod\data\basic\normrefs.xml
File Edit View Favorites Tools Help
Back Search Favorites Media
Address D:\rbn\projects\nist_module_repo\stepmod\data\basic\normrefs.xml
Go
+ <normref id="ref10303-1.1994" docbook-atts="url #DEFAULT"
docbook="bibliomixed">
+ <normref id="ref10303-11.1994" 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>
My Computer




ISO/TS 10303-1047:2003 Activity
File Edit View Favorites Tools Help
Back Search Favorites Media
Address D:\rbn\projects\nist_module_repo\stepmod\data\modules\activity\sys\2_refs.xml
Go
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:-1), 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.
Done
My Computer
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



ISO/TS 10303-1022:2003 Part and version identification

File Edit View Favorites Tools Help

Back Search Favorites Media

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

- common resources.

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

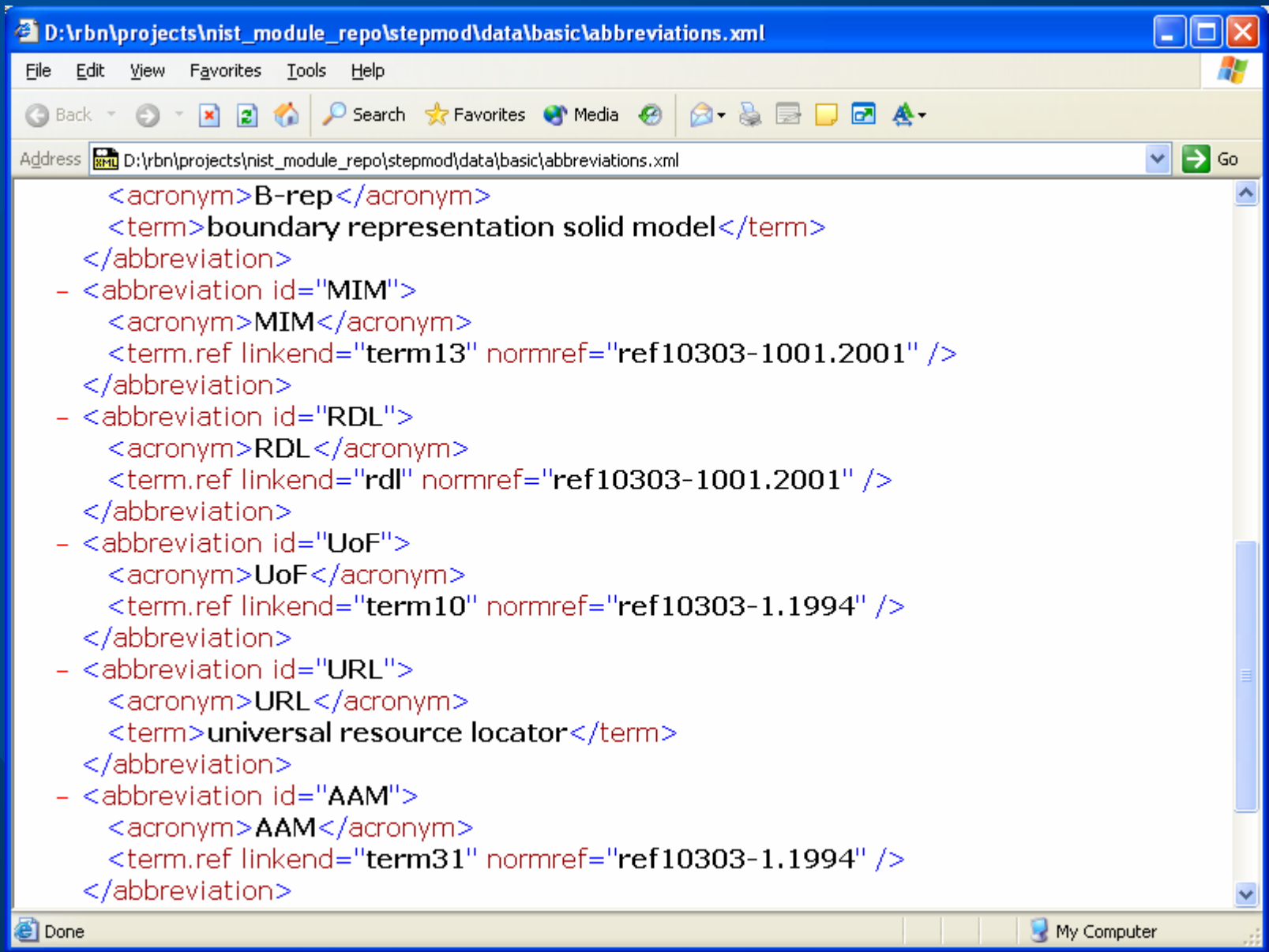
A part is a discrete object that may come into existence as a consequence of a manufacturing process. A part may contain components, which may be parts or non-countable materials.

3.5.3
raw material

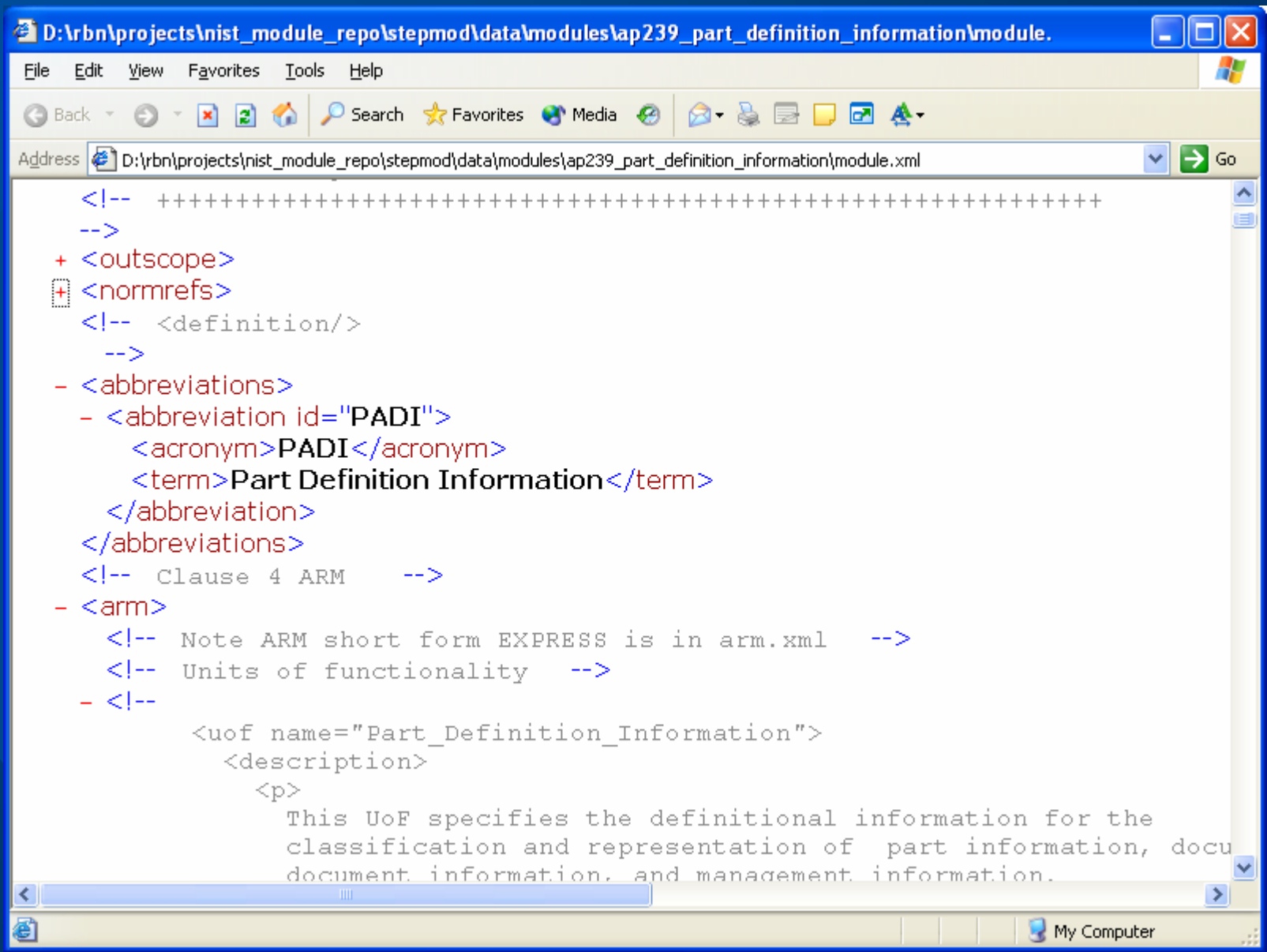
My Computer

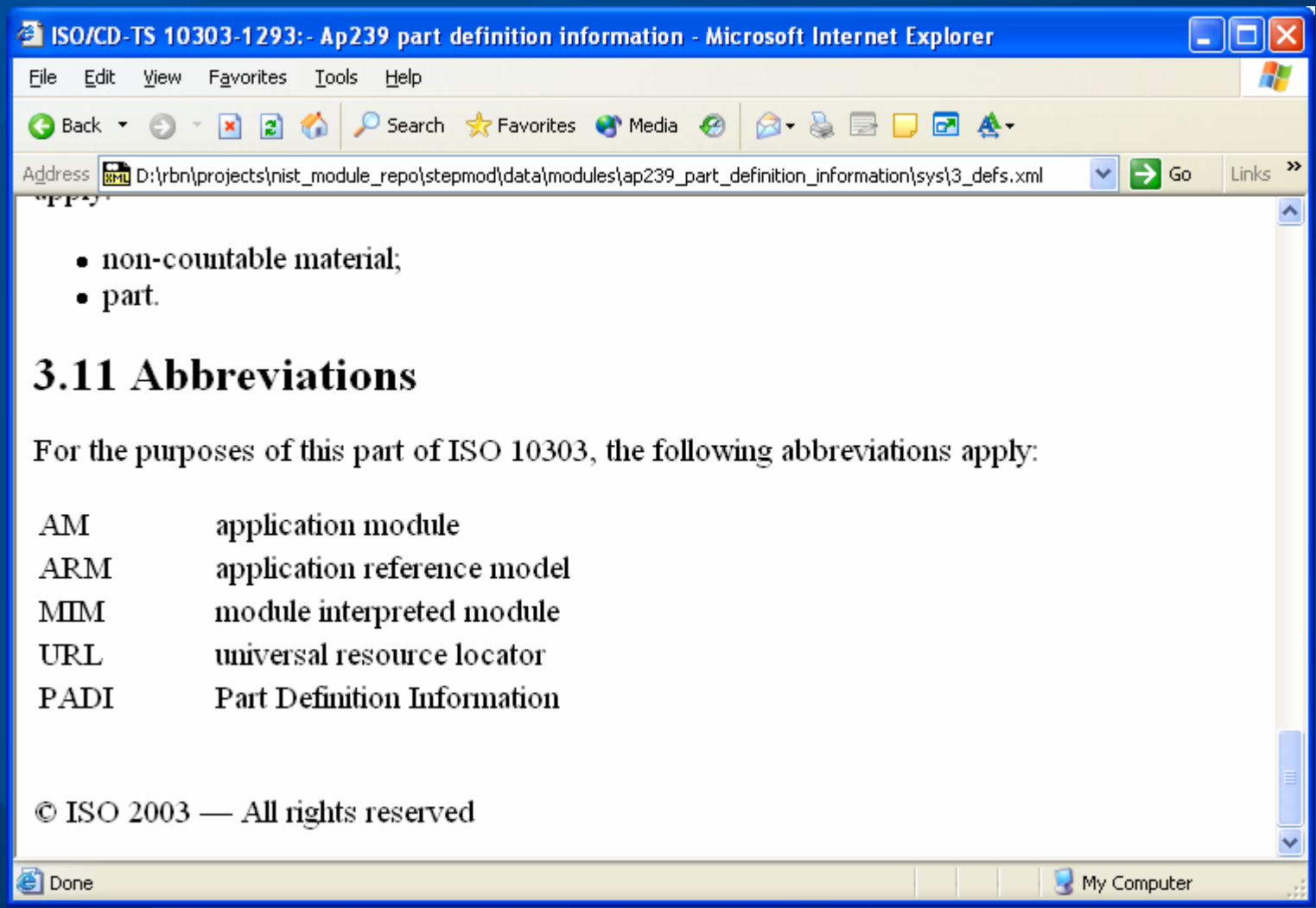
Abbreviations

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



eurostep®





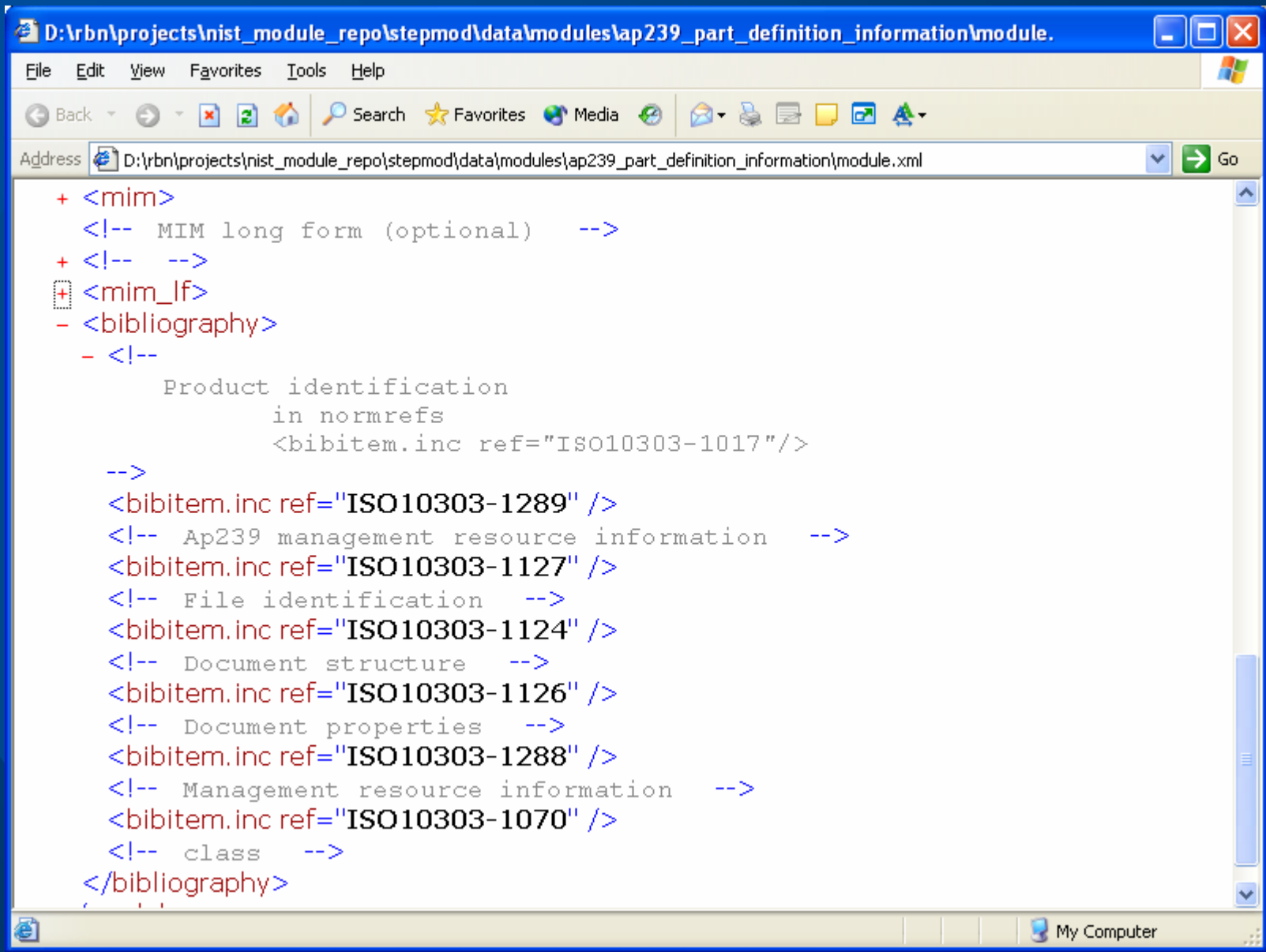
Simple few steps

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

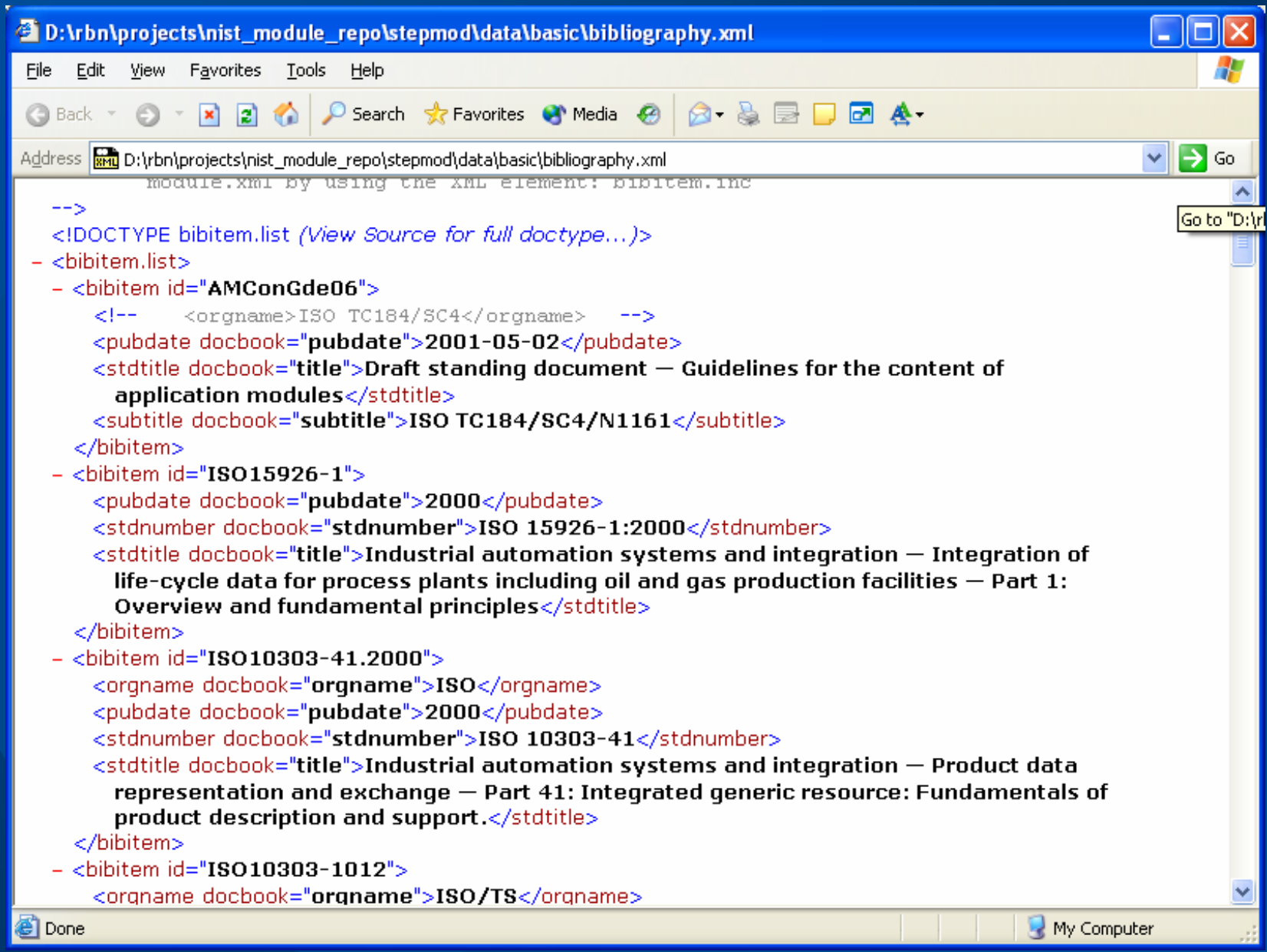
eurostep®

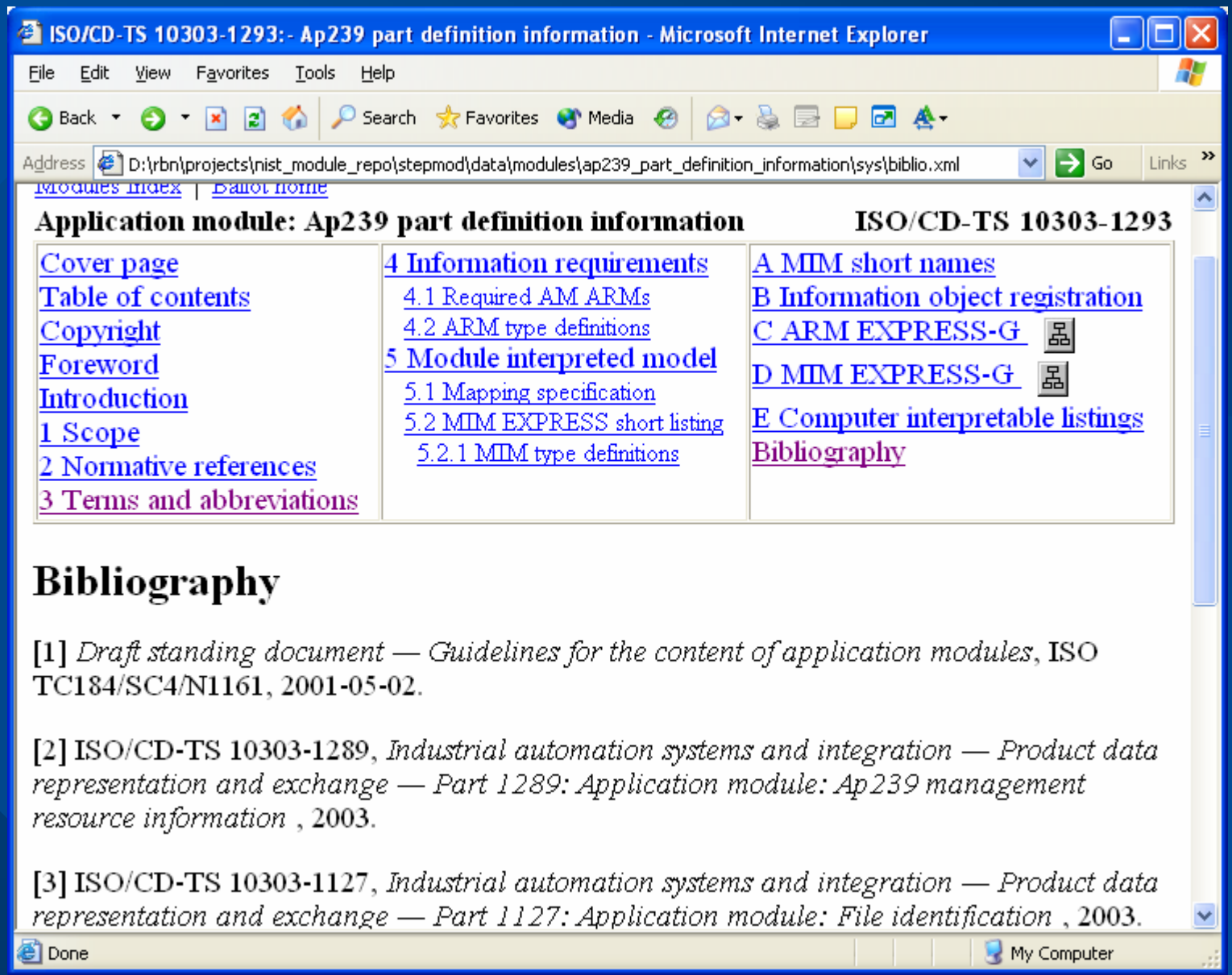
Bibliography

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



eurostep®





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

eurostep®

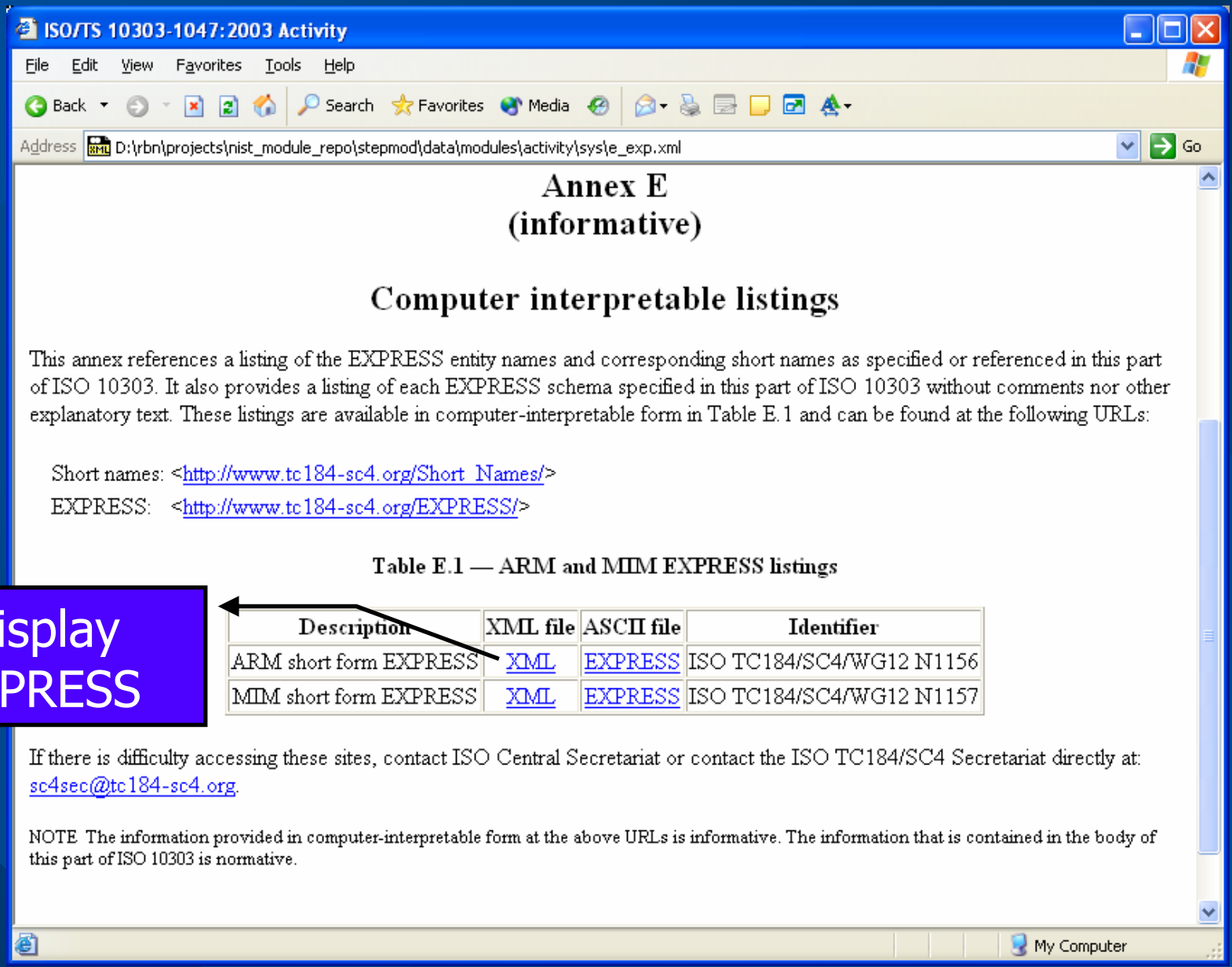
WG numbers

<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



Display EXPRESS

ISO/TS 10303-1047:2003 Activity

File Edit View Favorites Tools Help

Back Search Favorites Media

Address D:\rbn\projects\nist_module_repo\stepmod\data\modules\activity\sys\e_exp_arm.xml

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


```
(*
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;
```

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
7. Add bibliography
8. Add WG numbers
9. Add short names

eurostep®

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>
```

eurostep®

ISO/TS 10303-1047:2003 Activity - Microsoft Internet Explorer

File Edit View Favorites Tools Help

Back Search Favorites Media

Address D:\rbn\projects\nist_module_repo\stepmod\data\modules\activity\sys\a_short_names.xml Go Links

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

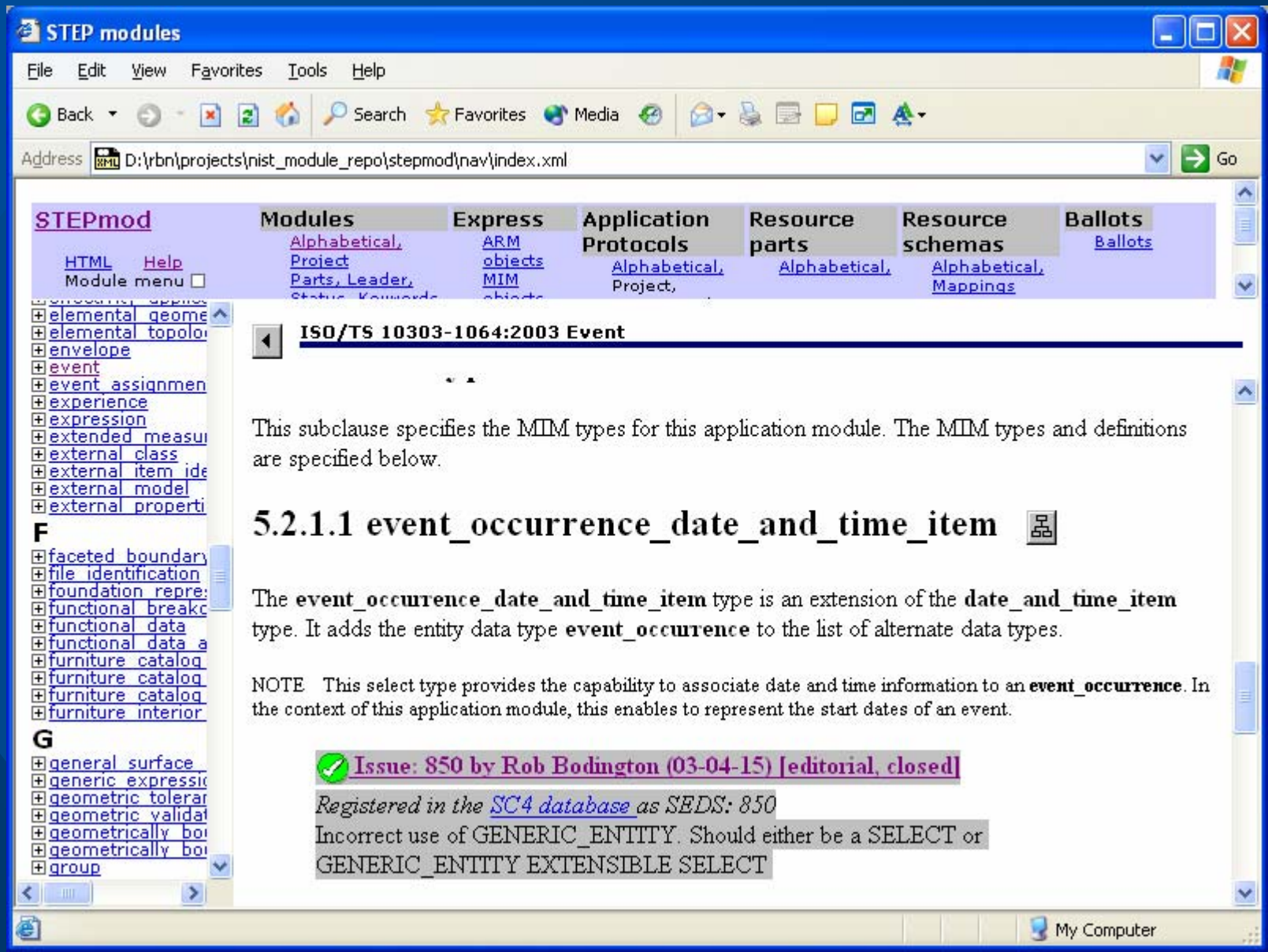
My Computer

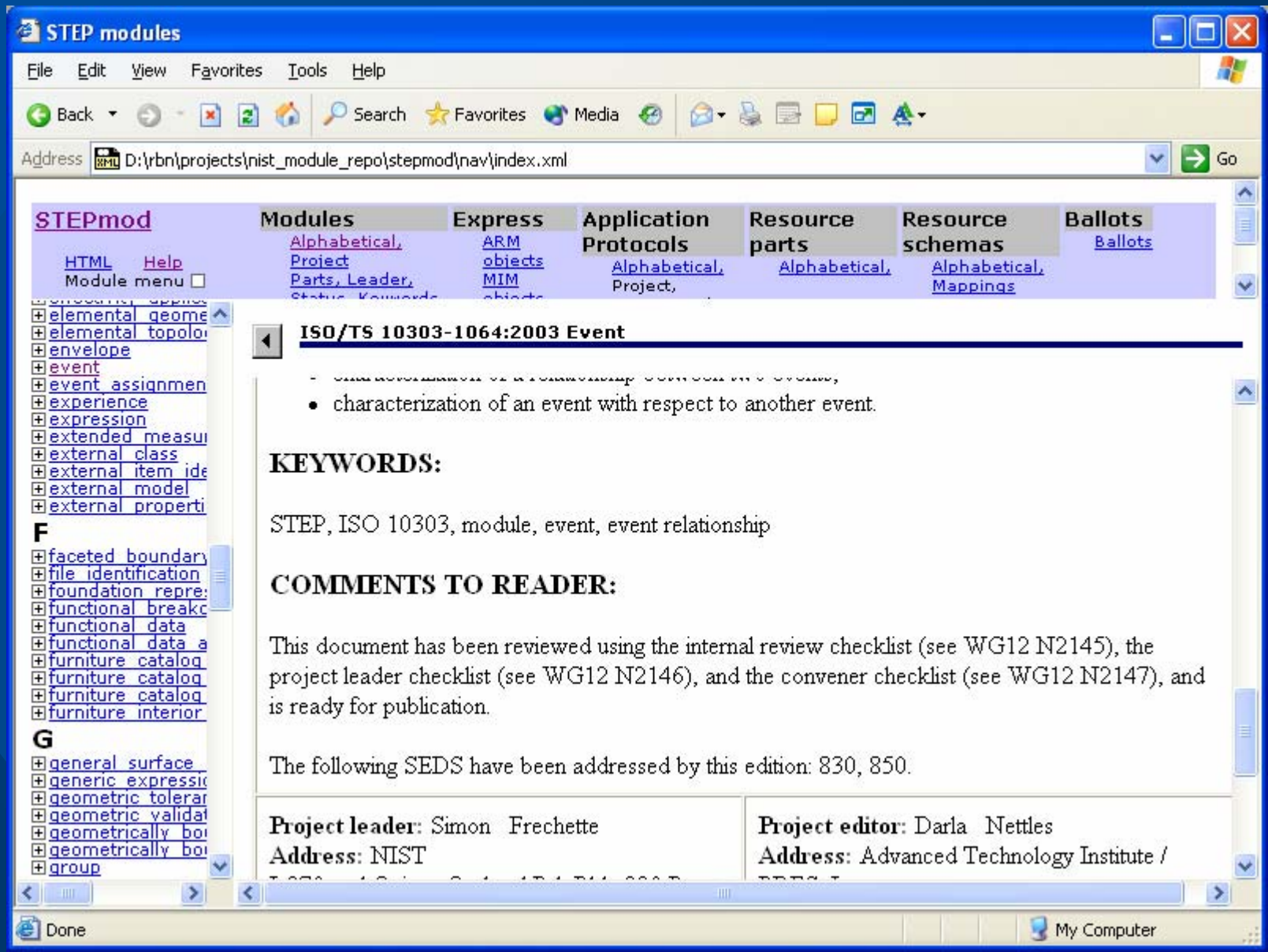
The module is complete!

eurostep®

Issues and SEDS

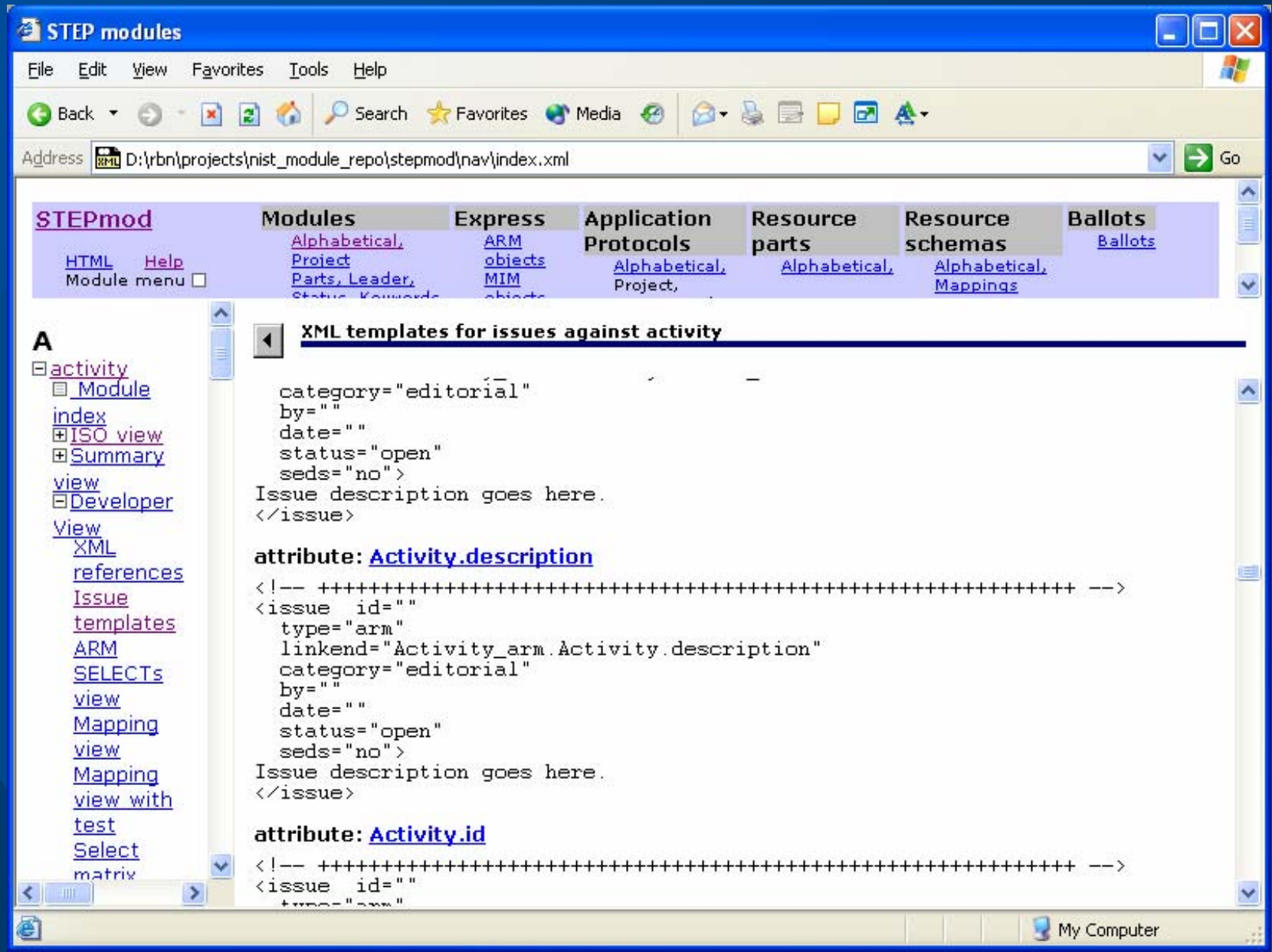
- Add issues to: stepmod\data\modules\[module]\dvlp\issues.xml
- Will be displayed in line
- If marked as SEDS will be displayed on cover page
  - See event, event_occurrence_date_and_time_item





Issues and SEDS

- XML can be copied from the developer view of repository



In addition to modules

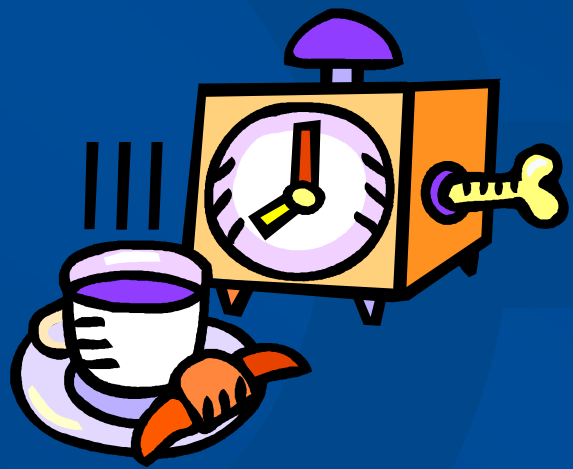
The repository will support:

- 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

eurostep®



eurostep®

Overall Agenda

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

eurostep®

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

Repository
Local copy

CVS commit

CVS update

CVS @ SourceForge
Repository

Repository
Local copy

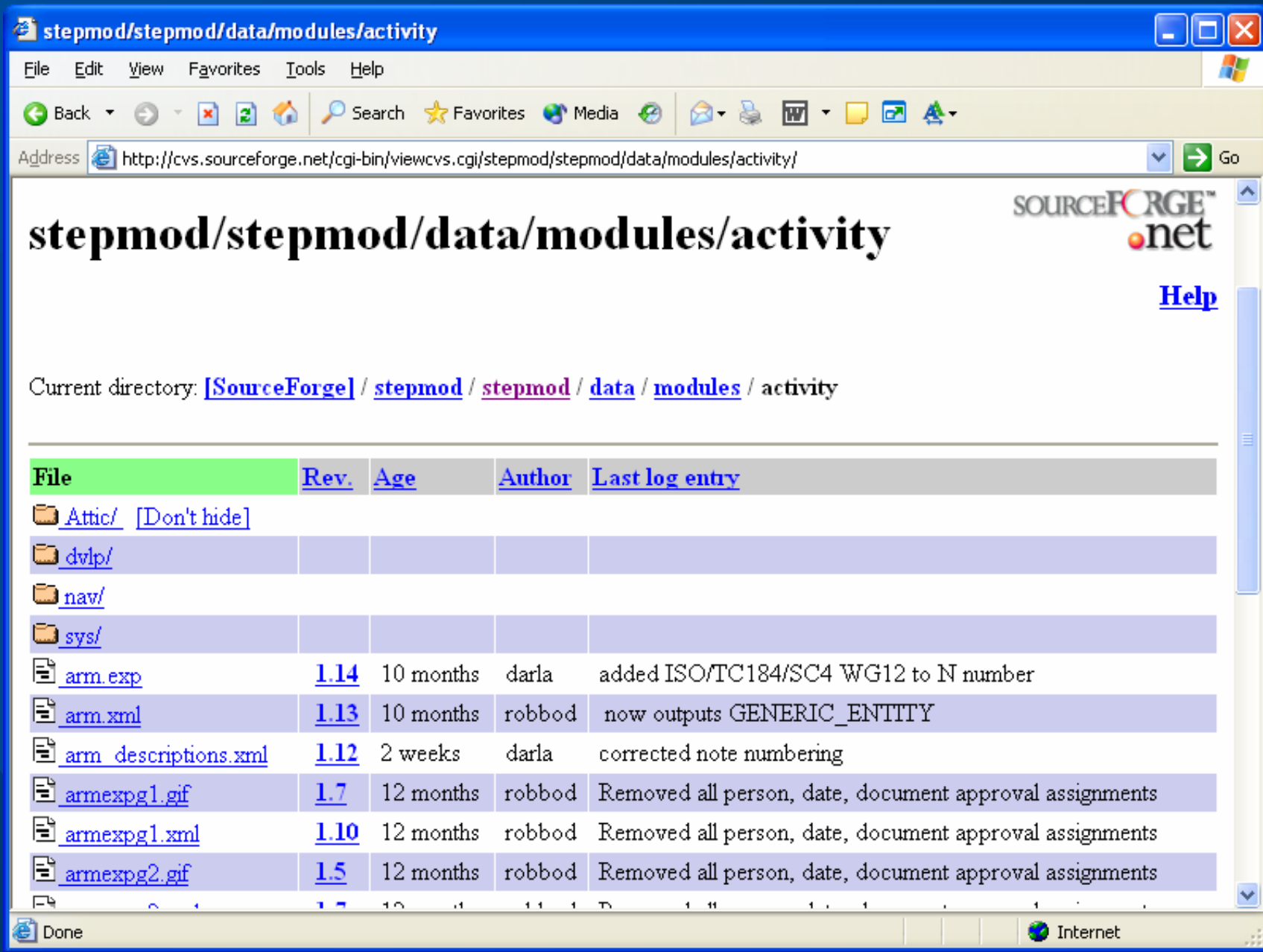
Repository
Local copy

- All files stored @ SourceForge
- Take a local copy (CVS update)
- Make changes
- Check them back in (CVS commit)

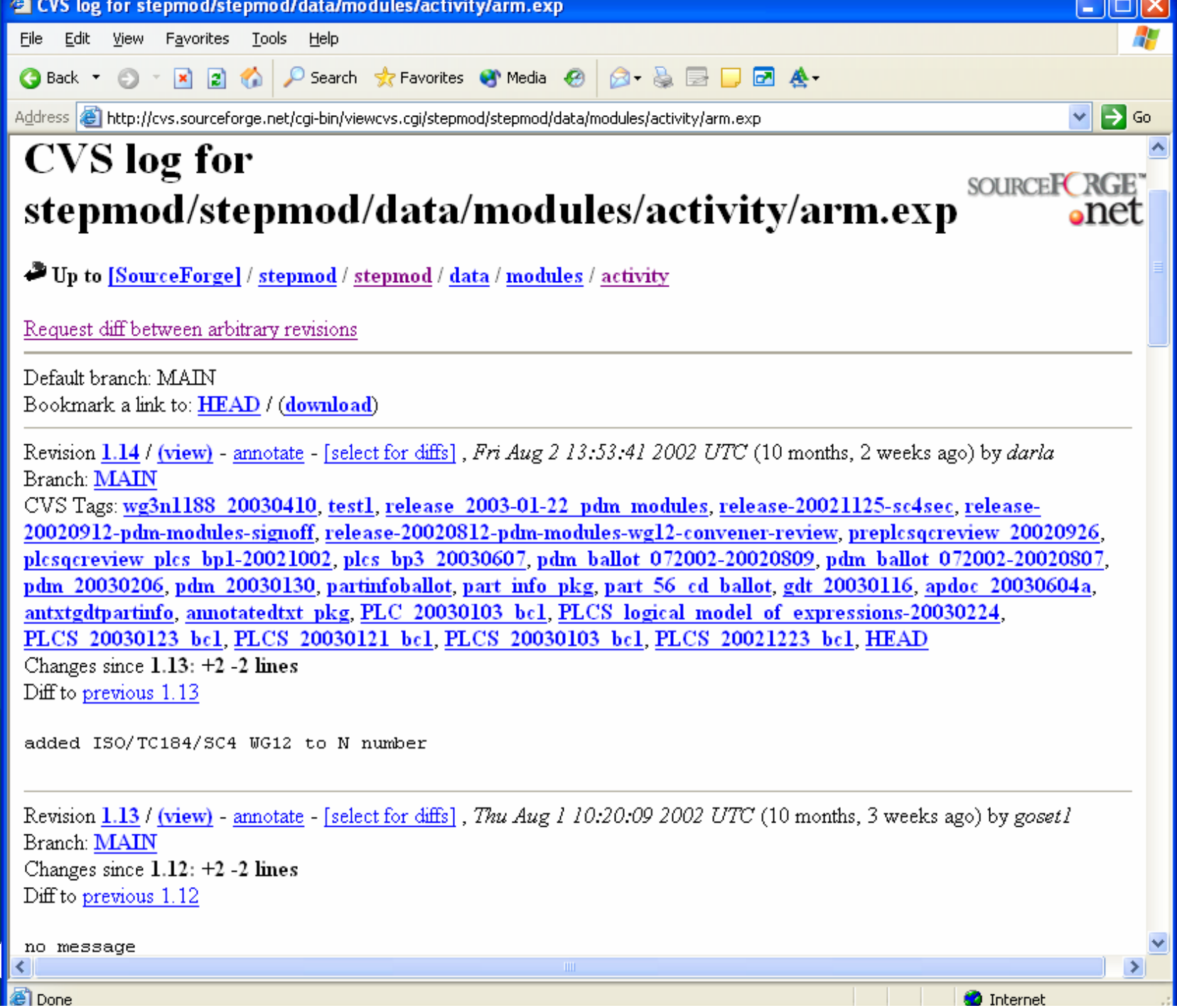
eurostep®

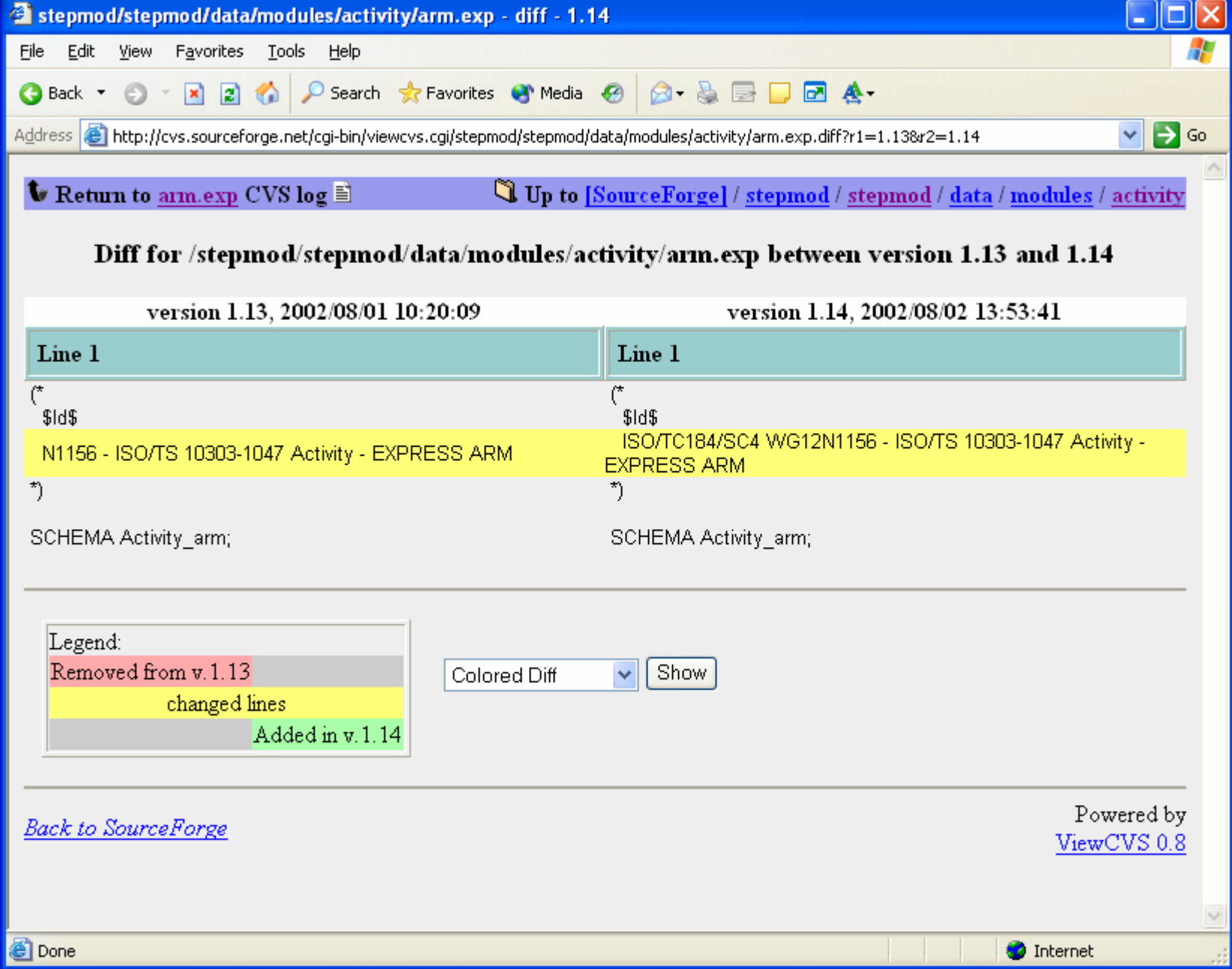
View history

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



eurostep®


CVS log for stepmod/stepmod/data/modules/activity/arm.exp
File Edit View Favorites Tools Help
Back Search Favorites Media
Address http://cvs.sourceforge.net/cgi-bin/viewcvs.cgi/stepmod/stepmod/data/modules/activity/arm.exp
Go
CVS log for stepmod/stepmod/data/modules/activity/arm.exp
SOURCEFORGE.net
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_bc1, PLCS_logical_model_of_expressions-20030224, PLCS_20030123_bc1, PLCS_20030121_bc1, PLCS_20030103_bc1, PLCS_20021223_bc1, 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 goset1
Branch: MAIN
Changes since 1.12: +2 -2 lines
Diff to previous 1.12
no message
Done
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

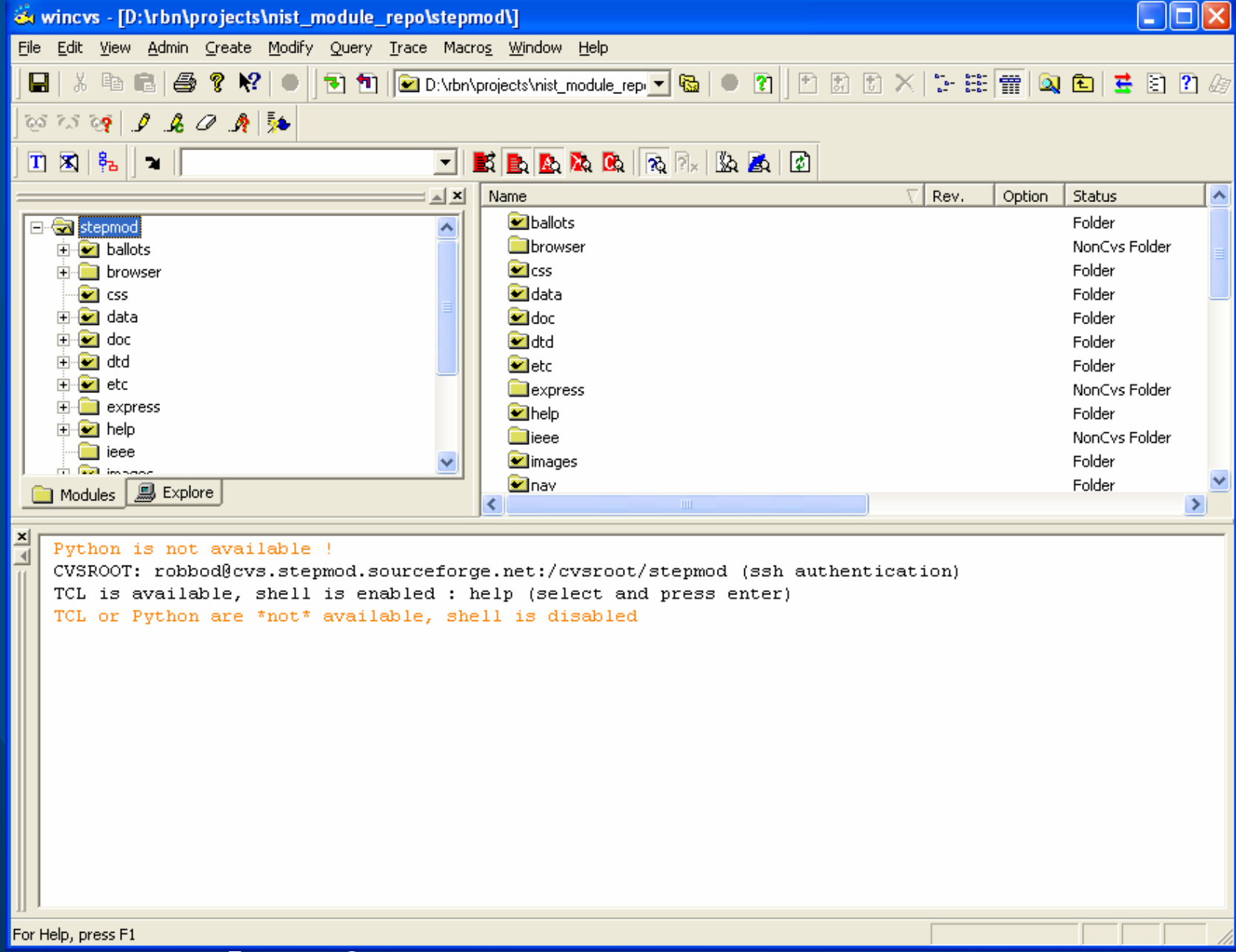
CVS tools

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 wrire: set 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

eurostep®



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

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

eurostep®

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

eurostep®

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

eurostep®

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®

eurostep®

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