QC-N157
An XML DTD for Mapping Tables

Markus Maier, PD Tec GmbH
Motivation

- There is no commonly agreed computer-interpretable format for Mapping Tables
  - each AP project uses its own tools for editing mapping tables and checking consistency
    - AP210: SGML
    - AP214: "mfile" format (SGML-like) + XML based
    - ...

- Benefits of computer interpretable format:
  - consistency checks
  - generation of different output formats (HTML, RTF, ...)
  - generation of EXRPESS-X mappings
  - use of commonly available tools
What we have...

- AP214 mfile
- map.dtd
- consistency checker
- other tools
- MS Excel (CSV)
- MS Word (RTF)

Conforms to:
- MS Excel (CSV)
- MS Word (RTF)
- HTML (table structure)
- EX-PRESS -X
- HTML (chapter structure)

Perl Script, XSLT Stylesheet, Prototype
How it is used

EX-PRESS

EX-PRESS

AP214 development environment

mfile

LaTex

AP dependent

AP independent

XML

RTF/CSV

other tools

Edit

Check ...
* DTD conformance
* Reference Path Syntax
* OR-case references
* UoF references
* ...

EX-PRESS -G

mfile

other tools

Edit

Check ...
* DTD conformance
* Reference Path Syntax
* OR-case references
* UoF references
* ...

RTF/CSV
What the DTD does include

- Application Objects grouped by UoF
- Application Elements and Application Assertions grouped by Application Object
- OR-case documentation
  - OR-case identifier
  - explaining text
  - UoF-specific OR-cases
  - Conformance Class specific OR-cases
- AIM Elements
  - + references to OR-cases
  - + reference to source part
- Reference Paths
  - + references to OR-cases
  - + comments (→ footnotes)
What the DTD does not include

- References to ARM and AIM EXPRESS entities
- Markup of Reference Path Elements
- It does not define the grammar of the reference paths
- Support for Mapping Templates
Todo

- Extend / Improve DTD
  - DTD is currently flavoured by the mfile SGML markup
  - Support AIM/ARM EXPRESS entity references
  - Add Clause 4 specific parts
  - Support for Mapping Templates

- Change conversion tools to support new syntax

- Tools for conversions to other data formats (LaTex, Postscript, ...)

EXPRESS-X MAPs for Mapping Specs.

- The new EXPRESS-X spec. contains "Path Operators" concept for MAPs, which is very close to the mapping table path syntax
  - this makes the transformation of EXPRESS-X mapping tables into EXPRESS-X more straightforward
  - automatic generation of EXPRESS-X out of Mapping Tables is possible up to some extend

- The new syntax would also be useful for constraint specifications in EXPRESS
  - automatic generation of constraints specs. possible
  - would help AP-Developers and AP-implementers
  - EXPRESS-X tool support already available
MAP surface_condition_map AS surface_condition;
FROM r : representation;
WHERE
  r<-used_representation{property_definition_representation}
    ::definition{property_definition}
    <-derived_definition{general_property_association}
    ::base_definition{general_property | name = 'surface condition'}
    <> [];
SELECT
  ...
value_determination ::= \
  r.items\{qualified_representation_item\} \
  :::qualifiers\{type_qualifier | name IN \n    ['required', 'designed', 'calculated', \n    'measured', 'estimated']\}[1];
| surface_condition to process_operation_occurrence (as is_caused_by) | PATH | representation <- property_definition_representation.used_representation property_definition_representation property_definition_representation.definition -> represented_definition represented_definition = property_definition property_definition property_or_shape_select = property_definition property_or_shape_select <- process_property_association.property_or_shape process_property_association process_property_association.process -> process_property_association.process property_process <= action |

```plaintext
is_caused_by := action_map(
  r<-used_representation{property_definition_representation} ::definition{property_definition}
  <-property_or_shape{process_property_association} ::process{action}[1]);
END_MAP;
```
ENTITY document_file
  SUBTYPE OF (characterized_object, document);
  WHERE
  wr1: (SIZEOF(QUERY(adr<* QUERY(dr <* USEDIN(SELF,
    'FEATURE_BASED_PROCESS_PLANNING.DOCUMENT_REFERENCE.ASSIGNED_DOCUMENT' )
   | 'FEATURE_BASED_PROCESS_PLANNING.APPLIED_DOCUMENT_REFERENCE'
   IN TYPEOF(dr))))
  | 'FEATURE_BASED_PROCESS_PLANNING.EXTERNALLY_DEFINED_FEATURE_DEFINITION'
   IN TYPEOF(adr.items)) = 1) OR
  (SIZEOF(QUERY (duc <* USEDIN(SELF,
    'FEATURE_BASED_PROCESS_PLANNING.DOCUMENT_USAGE_CONSTRAINT.SOURCE')
  | NOT
  (SIZEOF(QUERY(aduc <* QUERY(duca <* USEDIN(duc,
    'FEATURE_BASED_PROCESS_PLANNING.DOCUMENT_USAGE_CONSTRAINT_ASSIGNMENT.'
   'ASSIGNED_DOCUMENT_USAGE')
   | 'FEATURE_BASED_PROCESS_PLANNING.'
   'APPLIED_DOCUMENT_USAGE_CONSTRAINT_ASSIGNMENT'
   IN TYPEOF(duca)))
  | 'FEATURE_BASED_PROCESS_PLANNING.EXTERNALLY_DEFINED_FEATURE_DEFINITION'
   IN TYPEOF(aduc.items)) = 1))) = 0);
  wr2: (SIZEOF(QUERY(drt <* USEDIN(SELF,
    'FEATURE_BASED_PROCESS_PLANNING.'
   'DOCUMENT_REPRESENTATION_TYPE.REPRESENTED_DOCUMENT')
    (drt.name='physical'))) = 1);
END_ENTITY;
ENTITY document_file
  SUBTYPE OF (characterized_object, document);
WHERE
  wr1: (SIZEOF(
    SELF<-assigned_document{applied_document_reference}
      ::items{externally_defined_feature_definition}) = 1) OR
  (SIZEOF(
    SELF<-source{document_usage_constraint |
      NOT(SIZEOF(assigned_document_usage{applied_document_usage_contraint}
        ::items{externally_defined_feature_definition}) = 1)
    }) = 0;
  wr2: SIZEOF(SELF<-represented_document{document_representation_type |
      name = 'physical'}) = 1;
END_ENTITY;

- much more readable
- ~ 60% less code
- type safe
- 100% upward compatible
- Mapping to EXPRESS ed.1 defined in EXPRESS-X