

ISO/IEC JTC 1/SC 34 N 1323

DATE: 2009-11-19

ISO/IEC JTC 1/SC 34
Document Description and Processing Languages
Secretariat: [Japan \(JISC\)](#)

DOC. TYPE	Meeting Report						
TITLE	TMCL Issues [WG 3]						
SOURCE	WG 3 Convenor [Mr. Patrick DURUSAU]						
PROJECT	JTC 1.34.19756						
STATUS	Notes from the WG 3 2009-09 Seattle Meeting on FCD 19756, Information technology -- Topic Maps -- Constraint language. This document is circulated to the SC 34 members for information.						
ACTION ID	FYI						
DUE DATE							
DISTRIBUTION	P, O and L Members of ISO/IEC JTC 1/SC 34 ; ISO/IEC JTC 1 Secretariat; ISO/IEC ITTF						
ACCESS LEVEL	Open						
ISSUE NO.	105						
FILE	<table><tr><td>NAME</td><td>1323.pdf</td></tr><tr><td>SIZE (KB)</td><td></td></tr><tr><td>PAGES</td><td>32</td></tr></table>	NAME	1323.pdf	SIZE (KB)		PAGES	32
NAME	1323.pdf						
SIZE (KB)							
PAGES	32						

Secretariat ISO/IEC JTC 1/SC 34 - IPSJ/ITSCJ (Information Processing Society of Japan/Information Technology Standards Commission of Japan)* Room 308-3, Kikai-Shinko-Kaikan Bldg., 3-5-8, Shiba-Koen, Minato-ku, Tokyo 105-0011 Japan *Standard Organization Accredited by JISC
Telephone: +81-3-3431-2808 ; Facsimile: +81-3-3431-2808; E-mail: kimura@itscj.ipsj.or.jp

TMCL issues

Seattle, September, 2009

Issue #845: non-disjoint-subtypes

- The use case is when the subtypes of a given type allow overlap
 - for example, person, and subtypes musician, author, ...
- Can be supported today with an overlaps declaration
 - however, this has to be extended for each new subtype that is added
- The proposal is to define a new declaration which does this
 - `tmcl:non-disjoint-subtypes(on:Person)`
- Have not been able to suggest an alternative way to meet the use case
 - (without requiring new subtypes to be added in two places, that is)
 - Is this only really an issue for hand authored ontology?
 - Tooling can remove the need for the new constraint type.
 - No enthusiasm for the idea. Leave.

<http://projects.topicmapslab.de/issues/845>

Issue #847: variant constraints

- TMCL is currently lacking these, but they are requested by some users

- One proposal:

- `reading isa transliteration-type.`

- `hiragana isa script.`

- `romaji isa script.`

- `realName isa tmcl:name-type;`

- `- "Real name";`

- `has-variant(script, 0, *).`

- `script has-variant-scope(transliteration-type, 1, 1).`

- Unfortunately, this has problems. See URL below

- Don't do these. Tell people to create TMQL to constrain variants.

<http://projects.topicmapslab.de/issues/847>

Issue #848: Association role reification

- It is currently not possible to constrain the reifiers of association roles
- The Japanese NB writes:
 - “I think the Role Type should be included in the Reifier Constraint.”
- Technically, this can be solved
 - in meta-schema change constrained-statement to some new association type
 - probably adjust wording in validation rule slightly, too
- However, the question is
 - should reification of association roles be allowed and encouraged?
 - TMDM changes are not on the cards right now
- **Not a key use case. Use TMQL to do this.**

<http://projects.topicmapslab.de/issues/848>

Issue #850: Define datatype matching

- Given a declared datatype for an occurrence type (t1), and an actual occurrence (with value v and datatype t2), under what circumstances are the occurrence valid?
 - the current draft doesn't say...
- This issue is complicated
 - datatypes in XML Schema are complicated
 - TMDM uses them in a slightly different way from what XML Schema does
 - it affects how much of the XML Schema type system TMCL implementations must be aware of
- The URL below has more information and background
- Can we just have a position on set of built ins. Anything else goes but we don't say how to validate values of these types.
- Does a new datatype need to define its lexical and value, spaces?
- Does a subtype of xsd:int validate against the xsd:int lex/val space? We think yes it should but only if no lex/ val space is defined for the subtype.
- How is this new lex /val space defined?

Issue #850: A proposal

- The occurrence is valid if
 - v is XML Schema datatype valid according to t1,
 - v is XML Schema datatype valid according to t2,
 - the value spaces of t1 and t2 overlap
- This means that if tmcl:card-min is declared as ctm:integer
 - (1, xsd:int) will be accepted
 - (1, ctm:integer) will be accepted
 - (1, xsd:byte) will be accepted
 - (1.0, xsd:decimal) will not be accepted
- For this to work, TMCL implementations must
 - know the lexical space of each supported datatype
 - know the value space relationships between supported datatypes
 - Why do we need to know if they overlap. Isn't subtyping enough?
 - We can use subtyping and that's OK.
 - In general this looks good.

Issue 850: Feedback

- Michael Sperberg-McQueen (co-editor of XSD 1.1) writes:

“It would seem natural to me that in the situations you describe, the type associated with the value should be substitutable for the declared type. That would rule out accepting ("43", xsd:decimal) when the declared type is xsd:int, which seems to me likely to be the right call. Refusing to accept values labeled with the name of an ancestor type is more important when user-defined types are supported, and probably less important if only specific datatypes are built-in.

At the risk of trying your patience, let me explain why I don't think *ancestor* types should be accepted.

If I specify that hat size is a subtype of integer, it may well be precisely because I want to draw a conceptual distinction between the two. Calculating an integer by dividing a street address by the height of the family's eldest child, and rounding, may yield an integer, but it does not yield a hat size. If the declared type is msm:hatsize, an arbitrary integer should NOT be accepted, even if it's in range. If a hat size is calculated in some way that the user can see is plausible, then the user should coerce the value to msm:hatsize and take responsibility for the claim that it's plausible as a hat size.”
- So perhaps only subtypes of the declared type should be accepted
 - in this case, subtyping relationships between supported types must also be known
 - (I'm assuming subtype = derived type)
 - Where would this be stored / managed / stated in TMCL?
 - Implementations must know this for built in datatypes. Implementations can support more types if they so choose. TMCL is silent on how they do this.

Issue #1334: Which datatypes?

- Should we require support for a minimal set of datatypes?
 - If so, which datatypes? (The ones below look good)
 - Should we also allow implementations to support more datatypes? Yes.
- The datatypes supported by CTM might be a suitable set:
 - xsd:anyURI,
 - xsd:decimal,
 - xsd:integer,
 - xsd:date,
 - xsd:dateTime,
 - xsd:string, and
 - ctm:integer.

<http://projects.topicmapslab.de/issues/1334>

Issue #851: Which regexp syntax?

- Some possible syntaxes
 - the XML Schema one (in current draft; Perl-like)
 - IEEE 1003.1 2004 (not Perl-like)
 - XPath 2.0/XQuery 1.0/SPARQL
 - these all use the same extended version of the XSD syntax
 - the extensions allow matching of substrings (and not just whole strings), which is of no interest to us
- Implementations
 - Xerces 2 has an implementation of the XSD and XPath regexp langs
 - there is a set of C implementations of the IEEE regexps, but they don't match the spec 100%
- **Suggestion to go for XPath 2.0.**

<http://projects.topicmapslab.de/issues/851>

Issue #852: How to find schema contents

- Given a topic map which
 - contains TMCL declarations,
 - does not use the `tmcl:belongs-to-schema` associations
- how does one find all the constraints/declarations in it?
- Should clause 4 define a general procedure for “validate TM1 against TM2”
 - where it’s assumed that TM2 contains a schema, and
 - where TM1 may == TM2?

<http://projects.topicmapslab.de/issues/852>

#852 ideas

- Should we say something about how to determine the contents of a schema given a topic map? Also, not sure definition below actually works unless schema topic map and topic map to be validated are merged
- We know all constraints are of type `tmcl:constraint` so from there you get the 'contents of a schema' OR if there is a schema topic all constraint topics connected to it.
- Virtually merged would address the issue of 'the below not working'. So say something about how the validation algorithm runs on a map that is the result of virtually merging the two maps (schema and data).

Issue #858: User-defined constraints

- There are two types:
 - global constraints, which consist only of a query, and
 - local constraints, which are attached to some topic/association/... type
 - local constraints use \$this to refer to instances of the type they apply to
- Note that local constraints can always be reformulated to global constraints
 - however, in a system with continuous validation, all global constraint queries have to be run after each change, which can be costly
- Unfortunately, local constraints are ugly
 - use 3 different association types to connect to topics of different type. There is constrained-topic-type, constrained-statement, and constrained-role-type
- Possible solutions
 1. Leave as is. Meta-schema will have to use a user-defined constraint to handle this.
 2. Allow only constrained-topic-type. (No loss of expressivity.)
 3. Create new “applies-to” (or similar) association type for this.
 4. Do not support local constraints.

Do 4.

<http://projects.topicmapslab.de/issues/858>

Issue #859: Is the schema topic required?

- Must each TMCL schema have a topic of type `tmcl:schema` representing it? **NO**.
- If not, how can one reliably detect whether or not a given topic map contains a TMCL schema? **By querying for instances of `tmcl:constraint`.**

<http://projects.topicmapslab.de/issues/859>

Issue #868: Language scoping

- Suggest solving it by introducing requires-scope
place isa tmcl:topic-type;
requires-scope(description, english, 1, 1);
requires-scope(description, german, 1, 1);
requires-scope(description, french, 1, 1);
requires-scope(description, dutch, 1, 1).
description has-scope(language, 1, 1).
- Together with the existing scope constraints this should be sufficient
 - however, one may question whether the topic type should be a parameter
 - **This looks good but does need the topic type parameter.**

<http://projects.topicmapslab.de/issues/868>

Issue #951: Reifier type

- “TMCL defines types defining the primary use of their instances, for instance a topic of type role-type is used for typing roles. Why not add reifier-type to make it clear, that instances of this topic are used as reifiers?”
- Arguments for
 - makes it easy to see which topic types are reifier types
- Arguments against
 - not needed (reifier-constraint already provides this information)
 - topics must have two types: reifier-type and (say) occurrence-type

No.

<http://projects.topicmapslab.de/issues/951>

Issue #1146: tmdm:subject

- Where should this PSI be defined?
- If not in TMCL, should we mention something about where it is defined?
- If we need it then we define it in TMCL if other people need it they can use it, if others define it we can define equivalence.
- Does tmcl make ontological statements that are define in tmdm?

<http://projects.topicmapslab.de/issues/1146>

Need reifier type and scope type

- Would like scope type so that it is possible to make constraints only about allowed scoping types (for schema modularisation).
 - Gdm -> The only UI benefit is to shorten the list of available topic types when creating scope-constraints. This isnt strong enough to include this additional constraint type.
- Reifier type – no.

Examples are inconsistent

- E.g 6.6. and 6.7 use employee for different illustrations. Editorial will fix.

\$AT and \$RT wrong way round in 7.7

- `tmcl:constrained-statement(tmcl:constrains : ?c, tmcl:constrained : $rt)`
`tmcl:constrained-role(tmcl:constrains : ?c, tmcl:constrained : $at)`
- **Editorial.**

TT is undefined

- **Constraint Validation Rule** for all constraints c of type `tmcl:topic-role-constraint`: c applies to a topic type t ,
- **Should be `tt`.**

Is it possible to define TT once

- Editors to see if the \$tt explanation, definition and usage can be simplified, clarified.

Define 'statement' more clearly

- By referencing tmdm. i.e. It says that it is a scope-able thing.

9.1 Editorial

- [TMCL:user-defined-constraint](#), rr error.

Reflexive association with oneself

- How to define and constrain that a topic can be in a reflexive association with itself.
- Is this a tmcl issue?
- Should we add a note to explain this in tmcl?

Add a is-transitive annotation

- No.

Unique value constraint naming

- `def is-unique($st)` consider renaming to make 'value' part of the name.

Global validation rules section numbering

- We need it for talking about it.

Loose strict processing

- Need to make a statement about this.

relaxing constraints defined on super types

- How can this be done?
- **Currently not.**

How to indicate the schema(s) that a tm uses

- We used to have this
- It might be useful
- Could add a PSI for occurrence type to be used on topicmap reifier.

What effect does the tmdm:subject have on Q's

- So does get types on a topic include in the result tmdm:subject and does tmdm:subject >> instances return all topics?
- This is a TMQL question. Not wanted here.