TMCL Issues addressed in Leipzig 2007

TMCL OWL

- Conversion back and forth between OWL and TMCL needs to be considered carefully. There should be no unnecessary incompatibilities.
  - **Not yet addressed.**

- Marketing job on TMCL
- OWL to TMCL mapping

- Cursory glance across the two to avoid catastrophic faux pas

TMCL Validate

Given:

TopicMap: t

Constraint : c

Then:

Validate(t, c) => true | false

This is out of date, no? There is a good point here, though. Prof Lee wants to be able to have the schema in a separate TM. Would be good to be clear on how that's possible.

>> You misunderstand. It doesn't say where the Constraint C comes from. i.e it can be anywhere, any map.

TMCL Import Directive

%import http://www.isotopicmaps.org/tmcl/templates as tmcl

Actually not a TMCL directive, but a predefined URI for a "magic" CTM file.

>> Prose needs fixing as this is a CTM directive.

URI should perhaps end in .ctm?

>> yes.

TMCL Schema

%import http://www.isotopicmaps.org/tmcl/templates as tmcl

*schema http://www.example.com/myschema

Why *schema if there's only one? Also, could we define %include in CTM in such a way that that the "*schema" topic can be defined in the included CTM file? Or, actually, is it that way already? I think this problem is avoidable.

>> This is issue is due to the fact that the templates cannot return a new constraint and need to attach it to something. We could force people to pass in the schema topic.

>> It could be defined in the import. Yes do it.

TMCL Schema

schema
  http://psi.isotopicmaps.org/tmcl/schema
  isa http://psi.isotopicmaps.org/tmcl/topictype

Should this topic be called tmcl-schema instead?

>> Maybe, but related to previous issue.

>> Maybe yellow box wrong. Yes.

TMCL Namespace in CTM

The CTM here should use the new default namespace feature of CTM.

```
>> sure.
```

TMCL Constraint Eval Function Name

constraint_eval_function
  http://psi.isotopicmaps.org/tmcl/constraint_eval_function
  isa http://psi.isotopicmaps.org/tmcl/occurrencetype

Could we think of a better name for this one?

>> I quite like it as it states pretty clearly what it is.

constraint_eval_semantic ?

>> General review of the naming in TMCL.

TMCL topictype isa topictype

topictype isa topictype, surely?

>> probably. topic-type

TMCL constraint in map

```
def TopicTypeConstraint()
*ttc1 isa topicTypeConstraint
*ttc1 constraint_eval_function """
TMQL for if any topics used as a topic type and not instance of topictype then fail.
"""
AddConstraintToSchema(*schema, *ttc1)
End
```

Do we actually need this in the topic map? Better just to have in spec.

What if I put it in with different TMQL? What does that mean?

>> the whole point is that all constraints are defined this way. So I think they should be in the map.

TMCL CTM template prefix

We killed the ability in CTM to import templates with a prefix. Might want to reinstate that, or alternatively allow templates to be defined with a QName. That is, you write
def tmcl:topictypeconstraint() ...

>> Check again with the latest CTM draft.

TMCL topictypeconstraint

// create topic type constraint topic
tmcl:topictypeconstraint()

Why should the user have to invoke this template? Can't the CTM file we include do it? Given that it must always be there, I mean. And why do we need it in the TM at all.

Same for assoctype, roletype etc.

>> you might not want to have this validation but have others and you should be able to express that. If its always on in validation that's a problem.

TMCL nametype

Missing nametype

>> yep.

TMCL name for catt

Steve suggested applies-to

>> maybe, lets discuss.

TMCL template defs

If you (1) use the default namespace feature, (2) define a nice template for catt, and (3) use topic blocks in the actual constraint templates, I think this can become quite nice and clean.

>> great

TMCL standard constraints

- Consider defining TMQL queries for standard constraints in the specification only without putting them into CTM templates. E.g. topicoftype constraint

- >> leave as is and NB to review.

TMCL sledgehammer query

>> we need a constraint for finding if there are constructs defined that are not covered by any constraints.

LMG did part of this, looks good and will replicate and complete in the next draft

TMCL template defs

- SubjectIdentifierConstraint(person, 1, 1, "*"), Say I wanted any number of SIs to be allowed; how would I write that? As far as I can tell, I must give an integer...

- use strings and casts
- extra templates
- named parameters, with IF in template
- default values _ e.g with use strings and casts
- overloading, same name, different signatures. +1
- leave as is

- Write up examples of all of these for discussion. Definition and invocation.

TMCL binary assoc templates

- Binary associations should use a simplified template
- Yes. Will be added to next draft.

TMCL Scoping Constraints

Possibly underpowered in current draft.

Needs more discussion.

TMCL *schema

- Current situation with '*schema' does not look very attractive, consider including '*schema' inside of the TMCL standard declaration and using "%include" directive in CTM

- yep.