TMRAP

A Web Service Protocol for Topic Maps

TMRA '05

Lars Marius Garshol
Development Manager, Ontopia
<larsga@ontopia.net>

2005-10-05
Overview

• Introduction
  – what is TMRAP?
  – uses for TMRAP
  – relation to previous TMRAP

• The protocol
  – general principles
  – the methods

• Conclusion
  – status
  – further work
Introduction

What is it?

Why did we make it?

Relationship to TMRAP 0.2
What is TMRAP?

- **Topic Maps Remote Access Protocol**
  - a web service interface to a Topic Maps server
  - a set of defined methods which can be invoked by remote clients

- **Enables real knowledge integration**
  - Topic Maps no longer restricted to monolithic applications
Connecting portals
Connecting portals (2)

Central TM server

Prerequisites
- førerkort kl. A
- kjøre villmann
Fragment-based applications

Application

topic
map server
Integrating with other applications

CMS

Topic map server
Building knowledge hubs

Application 1

Application 2

Application 3

Application 4

topic map
But there is a TMRAP already?

- Yes, there is: TMRAP 0.2
- TMRAP 0.2 is implemented in the OKS
- It contains two requests
  - get-topic
  - get-topic-page
- The current proposal expands TMRAP with new requests
  - the existing two requests are slightly updated
The protocol

General principles

The methods
TMRAP basics

• A set of methods with defined parameters and results
• Abstract protocol
  – independent of any specific technology
  – plain HTTP binding created
  – SOAP binding to come
• Follows traditional HTTP style, rather than REST
• Operations are coarse-grained
  – aim is to reduce number of operations needed for any given task
get-topic

• Parameters:
  - indicator: a set of URIs (subject identifiers of wanted topic)
  - subject: a set of URIs (subject locators of wanted topic)
  - source: a set of URIs (item identifiers of wanted topic)
  - topicmap: identifier for topic map being queried
  - syntax: string identifying desired Topic Maps syntax in response
  - view: string identifying TM-Views view used to define fragment

• Response
  - topic map fragment representing topic in requested syntax
  - default is XTM fragment with all URI identifiers, names, occurrences, and associations
  - in default view types and scopes on these constructs are only identified by one `<*Ref xlink:href="..."/>` XTM element
  - the same goes for associated topics
get-topic-page

• **Parameters:**
  - **indicator**: a set of URIs (subject identifiers of wanted topic)
  - **subject**: a set of URIs (subject locators of wanted topic)
  - **source**: a set of URIs (item identifiers of wanted topic)
  - **topicmap**: identifier for topic map being queried

• **Response is an XML structure**

```xml
<topic-pages>
  <server-name>A displayable name for the server</server-name>
  <topic-page>
    <topicmap-handle>The handle of the topic map.</topicmap-handle>
    <topicmap-name>A displayable name for the topic map.</topicmap-name>
    <topic-name>A displayable name for the topic.</topic-name>
    <view-uri>URI of topic page</view-uri>
    <edit-uri>URI of edit page (optional)</edit-uri>
  </topic-page>
</topic-pages>
```
get-tolog

• **Parameters:**
  - **tolog**: tolog query
  - **topicmap**: identifier for topic map being queried
  - **syntax**: string identifying desired syntax of response
  - **view**: string identifying TM-Views view used to define fragment

• **Response**
  - if syntax is “tolog”: an XML syntax for representing tolog query results
    • basically represents table structure of result
  - otherwise, a topic map fragment containing multiple topics is returned
    • syntax then treated as for get-topic
add-fragment

• **Parameters:**
  - `fragment`: topic map fragment
  - `topicmap`: identifier for topic map being added to
  - `syntax`: string identifying syntax of request fragment

• **Result**
  - fragment imported into named topic map
delete-topic

• Parameters:
  – **indicator**: a set of URIs (subject identifiers of wanted topic)
  – **subject**: a set of URIs (subject locators of wanted topic)
  – **source**: a set of URIs (item identifiers of wanted topic)
  – **topicmap**: identifier for topic map being queried

• Result
  – deletes the given topic
add-type-listener

- **Parameters:**
  - **indicator:** a set of URIs (subject identifiers of wanted topic)
  - **subject:** a set of URIs (subject locators of wanted topic)
  - **source:** a set of URIs (item identifiers of wanted topic)
  - **topicmap:** identifier for topic map being queried'
  - **client:** handle of client to be notified

- **Result**
  - every time a topic of the identified type is modified the client is notified

- A remove-type-listener method can be used to unregister the listener
Client operations

- These operations are invoked on the client by the server when topics are changed
  - **topic-created**
    - contains fragment representing new topic
  - **topic-updated**
    - contains fragment representing updated topic
  - **topic-deleted**
    - contains URI identifiers for deleted topic
Conclusion

Status

Further work
Status and further work

• **TMRAP 0.2 implemented in OKS**
  – Vizigator applet (Vizlet) uses TMRAP to download topic map fragments
  – already realizes one of the use cases in real life

• **TMRAP 1.0 about to be implemented**
  – release expected this year
  – will be used to realize several more of the use cases, again in real life

• **Further work needed to simplify updates**
  – an update-topic operation combined with a view would make it much easier to update merged topic maps coming from different sources