TopicMap Merge Platform

Latest: 20170520 **2**

Background **2**

Present Flows 2

Planned Flows 2

Merge Use Cases **3**

Same Label 3

Same URL 3

Same URI/PSI 3

Merge Architecture **3**

APIs **3**

IMergeAgent 3

IMergeImplementation 3

IMergeResultLisstener 3

ISubjectProxy 3

ISubjectProxyModel 3

Classes **4**

BaseVirtualizer 4

DefaultVirtualizer 4

MergeBean 4

MergeInterceptor 4

SameLabelDetector 4

SameLabelMergeHandler 4

SubjectProxy 4

SubjectProxyModel 4

VirtualizerHandler 4

Glossary **4**

Merge 4

MergeAssertion 4

SubjectProxy 4

UnMergeAssertion 4

Virtual Merge 4

VirtualProxy 4

Notes (Blog) 5

References 5

Latest: 20170520

Background

A merge platform for the TQElasticKnowledgeSystem

Present Flows

Here is a trace of flows for a new topic entering the database:

- `ITQDataProvider.putNode` persists the node to the topic map.
- If no persistence errors
    - The new topic is sent to `MergeInterceptor.acceptNodeForMerge`
    - MergeInterceptor then queues that topic to be processed in worker thread.
    - Worker Thread performs two processes
        - The topic is sent to SameLabelDetector.acceptProxy
            - There, the proxy is subjected to Same Label analysis
                - <TODO>Document this action</TODO>
        - If a “MergeListenerPropagate” flag is set in the topicmap-properties.xml file
            - MergeInterceptor will send the topic as a JSON string out a specific TCP stocket to any listeners
            - NOTE: the TCP socket is blocking if there are no listeners
            - TODO: change this to a KafkaBackside source

Planned Flows

<INTENT>To map out the entire Kafka-based agent ecosystem as relates specifically to merge operations and topic map maintenance</INTENT>

Merge Use Cases

Same Label

Same URL

Same URI/PSI

Merge Architecture

APIs

`IMergeAgent`

<new>

`IMergeImplementation`

`IMergeResultLisstener`

`ISubjectProxy`

`ISubjectProxyModel`

Classes

`BaseVirtualizer`

`DefaultVirtualizer`

`MergeBean`

`MergeInterceptor`

`SameLabelDetector`

`SameLabelMergeHandler`

`SubjectProxy`

`SubjectProxyModel`

`VirtualizerHandler`

Glossary

Merge

MergeAssertion

SubjectProxy

UnMergeAssertion

Virtual Merge

VirtualProxy

Notes (Blog)

20170520 JP

- Starting to dive deeply into the merge platform in TQ-Elastic-KS
    - Noted that agent propagation of a proxy for external operations is turned off since the TCP socket blocks if nobody is listening
    - TIME to bring in KafkaBackside
        - Right now, it is not a Maven project
            - TODO
                - Mavenize KafkaBackside
                - Move it from the knowledgegarden repo to the topicquests repo
    - Borrowed the IMergeAgent class from the SolrSherlock codebase
        - It's a root entry point which accepts a proxy in the forms
            - Locator string (to go fetch)
            - JSONObject
            - ISubjectProxy
        - It's task is to then fire up a series of tests which examine the new proxy along many dimensions, all of which seek to compare subject identities with other topics in the topic map
        - If subject identities match, then perform a merge operation

References