

TripCom Implementation Meeting

18/19 Mar 2008 Day 1



- 10:00 - 13:00
 - current status of implementation and integration (TUW)
 - discussion of open issues and problems (which functionality has been realized), occasional adaptations of components
- 14:00 – 18:00
 - testing the 2nd prototype: current status, preparation of additional test data
 - performance and scalability evaluation: current status, technical and organizational issues, contributions by partners within scalability PM's
 - demonstration cases for review: specification, details of presentation

- What is complete ?
 - Full processing of Core API
- What remains to be done?
 - Security information extraction
 - Internal error handling
 - Groundings - REST/SOAP to be further developed
 - Processing of all other API operations (simple extension)
 - Testing

- What remains to be done?
 - Implementation of query processing using simple template in SPARQL format.
 - Interaction of internal components of distribution manager (Interaction for the query processing is just partially done)
 - Processing of conjunctive and disjunctive queries.
 - Implementation of Controller and Monitor
 - Interaction test with other components (Security manager, TS API, ...)
 - Unit Testing

Metadata Manager

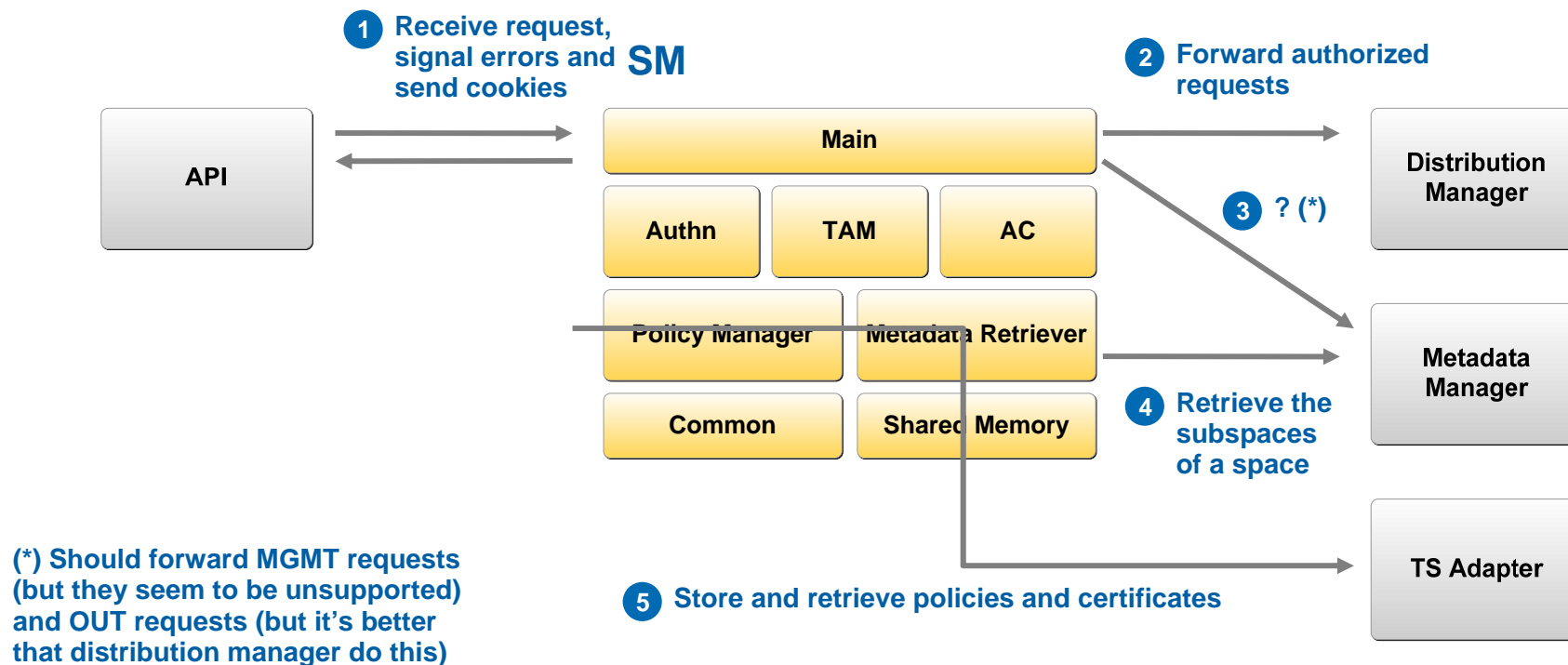


- Summary of progress of the 9.th week
 - Corrections and enhancements of tests
- Red flags
 - at the moment none
- Open issues
 - Integration

- What is complete?
 - The Query Preprocessor interfaces have been defined
- What remains to be done?
 - Update of entry types and component interaction matrix allowing for agreement by the other component developers, expect to implement the component interaction with the Query Preprocessor.
 - Dummy implementation will implement the external and internal interfaces without any functionality

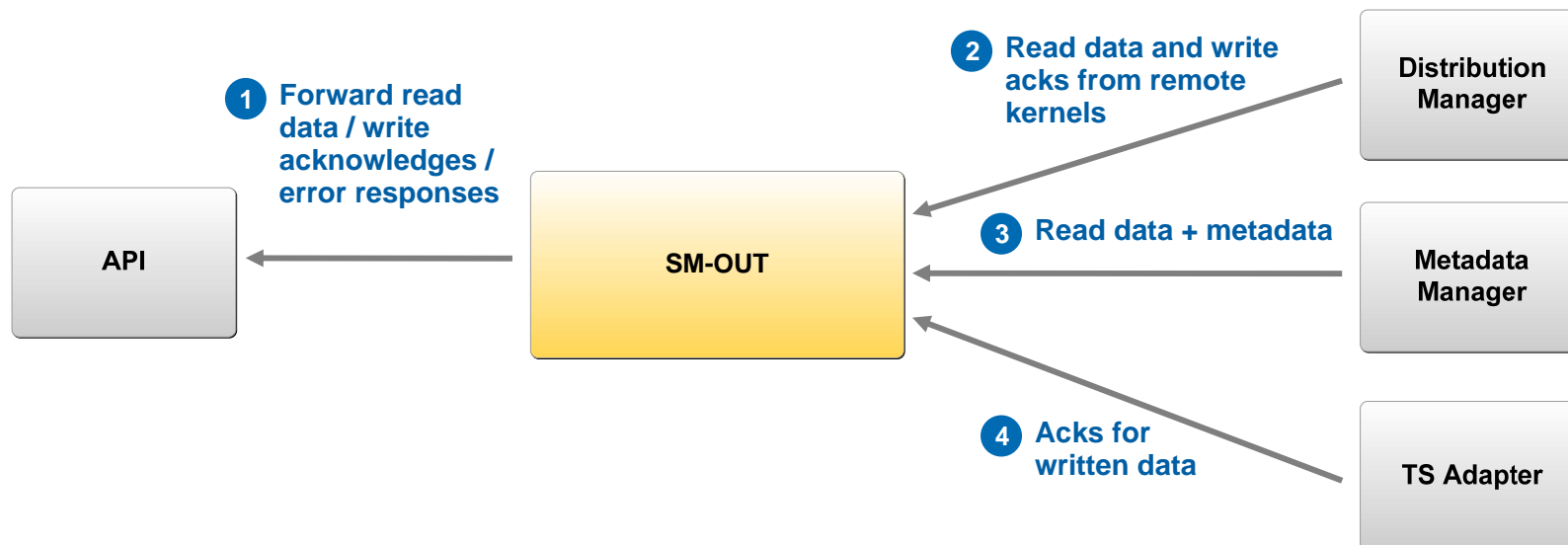
Security Manager status (1)

- Two sub-components: **SM** and **SM-OUT**
- SM handles incoming requests
 - Implementation work is completed, ongoing unit and component level testing
 - Need to provide a way for the end user / for us to configure security policies
 - Need to test “on the field” the following interactions



Security Manager status (2)

- SM-OUT handles outgoing responses
 - Only a placeholder to be possibly refined in the next prototype without altering the current kernel architecture
 - Implementation and testing are ongoing
 - Open issue: **error notification and propagation** to the API
 - Need to validate the following interactions



- What is complete?
 - Completed the first part of sample data generation
- What remains to be done?
 - Linking the tsadapter interface to these of Security and Metadata Manager

Web Service Discovery aims to locate Web services that can potentially fulfil a request from an user, a.k.a. a specific user goal

- Web service discovery. Matching and selecting all Web services that can potentially be used for fulfilling a certain goal
- An user request can be presented in format of WSDL or WSML. WSDL and WSML need to be converted into RDF triples

■ Current Status

- WS discovery in WSML format
- WS discovery in UDDI requests

■ Depending on TS-API (extendAPI) to

- discover desired Web services, and
- save the result into TipleSpace

■ Open Issues

- Assume that only valid users can send a discovery request to TripCOM

Web service invocation aims to achieve a specific function provided by a Web service through a Web service transport. TripCOM supports RPC-style service invocation with a pair of request/response

- Web service invocation. Retrieving the data published by a given Web service, which is stored in a subspace, to response the invocation
- An invocation request can be presented in format of a WSML goal or a WSDL request

■ Current Status

■ Invoke

- WSDL Web services by the request in WSDL request
- WSML Web services by the request in WSML goal

■ Dependency

- Depending on TS-API (extendAPI) to save the result into TipleSpace
- For invoking WSML Web services, these Web services must be stored in WSMX, and ontologies used by the goal must be stored in WSMX

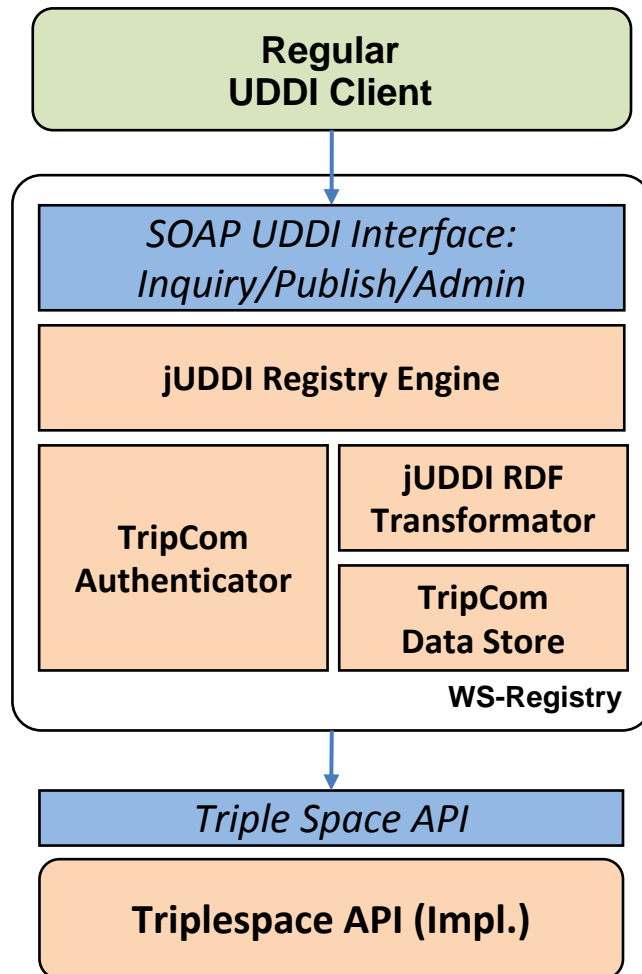
■ Open Issues

- For invoking WSML Web services,
 - Needs to save ontologies used by the WSML goal into WSMX,
 - A specific adapter for processing ontologies if a WSML request to invoke a WSDL Web services

We ask use cases guys to give us the WSML Web services, WSML goal and all ontologies beforehand. Thanks!!!

- Assume that only valid users can send an invocation request to TripCOM

- Provides an implementation of a Triplespace-based Web service registry
- Is realized as an extension of Apache jUDDI (<http://www.apache.org/juddi>)
- Will be bundled and deployed as a WAR
- Requires a servlet container (e.g. Apache Tomcat)



- **External Interface**
 - Regular UDDI Interface via SOAP/HTTP
 - Provided by jUDDI
- **Internal Interface**
 - Triple Space API
- **Necessary Extensions**
 - **jUDDI RDF Transformator**
 - Transformation of jUDDI data model objects to collections of RDF statements and back
 - **TripCom Data store**
 - Storing registry data to and retrieving registry data from Triplespace
 - **TripCom Authenticator**
 - Authentication of registry clients

■ **jUDDI RDF Transformator**

- Implementation of both transformations (jUDDI → RDF; RDF → jUDDI) done
- Unit tests available

■ **TripCom Data Store**

- Storage operations implemented
- Retrieval operations implemented as SPARQL pseudo code
 - Implementation will be completed as soon as format of queries has been defined
- To provide a complete implementation of the WS registry, some features of the Extended and Further Extended API are required (e.g. transactions, destructive consumption)

■ **TripCom Authenticator**

- For this prototype there will only be simple user/password authentication for information publishers
- Future prototypes will be based on TripCom's security mechanisms

Testing status

Current test status



	mvn test	other tests	JavaDoc
Distribution Manager	0	yes	yes
Integration	11	no	no
Metadata Manager	67	yes	yes
Security Manager	65	no	yes
Transaction Manager	4	no	yes
TripleSpace API	0	yes	yes
tsadapter	0	no	no
ws/discovery	0	no	no
ws/invocation	0	yes	no
ws/registry	17	no	yes

Error Handling

■ Error

- in case of an error (e.g., invalid Transaction)
- **ErrorResultExternal**
- central place with the error numbers / enumeration of errors (e.g., integration package)

■ Exception

- **ExceptionEntry**
- can be monitored with blitz dashboard
- „monitor“ component could be implemented
 - e.g. write exceptions to log file

Configuration

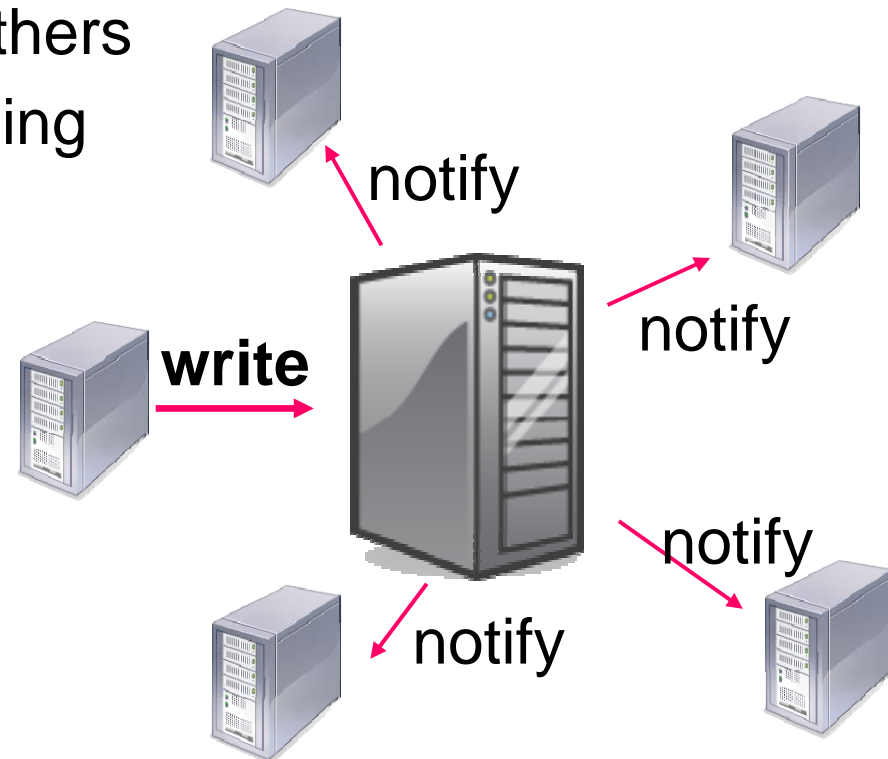
- Unify kernel configuration
- Via java properties
- Configuration position
 - `org.tripcom.<component>.file=<component>.props`
- Component specific configuration:
 - `org.tripcom.<component>.XYZ=<value>`
- Kernel configuration
 - `org.tripcom.ABC=<value>`

Scalability Evaluation

Scalability Test Setup for Testbed Assessment



- Central server node(s)
 - provide “shared memory”
 - propagate modification of contents by one client to others
- Increase of resources via adding server nodes



- Simplistic RMI based implementation for assessing Campus Grid testbed
- Clients send random integer numbers with stochastic frequency
 - on average 500 ms between write operations
 - test runs 30 seconds
 - single server node

Number of clients	Avg. update time (ms)
5	5
25	61
50	935
127	5391