Repositories Management

Adrian Marte, STI Innsbruck
Michael Schneider, FZI Karlsruhe
Overall Architecture

SEALS Service Manager

Evaluation Organizers

Runtime Evaluation Service

Technology Providers

Technology Adopters

Evaluation Descriptions Repository Service

Test Data Repository Service

Tools Repository Service

Results Repository Service
SEALS Repositories

- Test Data Repository Service
  - Persistently stores test data sets
  - Stores references to external test data sets
  - Stores synthetic test data generators
  - Generates synthetic test data

- Tools Repository Service
  - Stores the tools involved in an evaluation

- Results Repository Service
  - Stores the raw result of an evaluation
  - Stores interpretations over one or more raw results
ARCHITECTURE
Artifacts and Artifact Versions

• Repositories store artifacts and artifact versions
  – Artifact
    • Usually a collection of versions
    • Usually does not have data associated with it
    • Test data collection, tool, result, interpretation, etc.
    • E.g., the tool “Protégé” is an artifact
  – Artifact version
    • Concrete version of an artifact
    • Comes with concrete data
    • E.g., the tool version “Protégé 4.1” is an artifact version
Metadata

• Artifacts and artifact versions can be described using the SEALS metadata ontology
• Repositories store data and metadata separately
• Used to identify, interlink and search for artifacts and artifact versions
Repositories Architecture

**External Interface**
- REST Resources

**Internal Interface**
- Test Data Repository
  - Implementation
- Tools Repository
  - Implementation
- Results Repository
  - Implementation

**Storage**
- RDF Repository
- File Repository

**Diagram Components**
- HTTP Request
- File
- Java objects
- RDF Triples
- REST Resources
Storage Layer

- **File Repository**
  - Stores submitted tools, generators, test data, etc.
  - Uses underlying file system
  - Allows for storage of many large files

- **RDF Repository**
  - Stores RDF encoded metadata
  - Allows to pose SPARQL queries over metadata
  - Based on Sesame RDF repository
  - Can be accessed remotely over HTTP
Internal Interface Layer

- Java interfaces for repositories
- Use underlying storage layer
- Allows repository functionality to be exposed using different protocols
- Allows other implementations that use different storage mechanisms
External Layer

- RESTful interface based on Restlet
  - Framework for RESTful applications
  - Open-source
- HTTP resources represent repository functionality
- Dispatch HTTP requests to corresponding repositories
- Use underlying repository interface implementations
MANAGING ENTITIES
Describing Entities

• Metadata is RDF/XML encoded
• Submitted to repository when registering or updating entities in the repository
• Used to identify, interlink and search for entities stored in the repositories
Storing Entities

• Artifact along with its RDF/XML encoded metadata is registered in repository using a HTTP POST request
• Artifact version along with metadata and data (in the form of a ZIP file) is submitted to the resource of the artifact using a HTTP POST request
• GET, PUT, DELETE or POST can be used to retrieve, update, delete or add additional artifacts and artifact versions
• POST is also used to publish an artifact, in which case it is allowed to be used in an evaluation
Accessing Entities

• Repositories create unique HTTP resources for submitted artifacts and artifact versions using the (auto-generated) dcterms:identifier value in the metadata

• Identifier is auto-generated by repository if not defined in the metadata

• Each repository defines URL schemes to access and manage the stored entities
TEST DATA REPOSITORY SERVICE
Test Data Repository Service

- Manages *test data collections* and *test data versions* used for evaluations
- Stores persistent test data (suites)
- Allows to reference and persist external test data
- Provides means for generating and storing synthetic test data using a user-submitted test data generator
Describing Test Data: Example

• Ontology matching test data: conference testsuite

```xml
<rdf:RDF xmlns:rdfs="..." xmlns:rdf="..." xmlns:dcterms="..."
  xmlns:seals="http://www.seals-project.eu/ontologies/SEALSMetadata.owl#">
  <seals:PersistentTestData>
    <seals:hasName rdf:datatype="&xsd:string">Conference Testsuite</seals:hasName>
    <dcterms:description rdf:datatype="&xsd:string">The conference testsuite.</dcterms:description>
  </seals:PersistentTestData>
</rdf:RDF>

<rdf:RDF xmlns:rdfs="..." xmlns:rdf="..." xmlns:dcterms="..."
  xmlns:seals="http://www.seals-project.eu/ontologies/SEALSMetadata.owl#">
  <seals:TestDataVersion>
    <seals:hasVersionNumber rdf:datatype="&xsd:string">2010</seals:hasVersionNumber>
    <dcterms:description rdf:datatype="&xsd:string">Version of the conference dataset used in 2010.</dcterms:description>
  </seals:TestDataVersion>
</rdf:RDF>
```
# Accessing Test Data

<table>
<thead>
<tr>
<th>URI</th>
<th>Op.</th>
<th>Accept</th>
<th>Effect</th>
</tr>
</thead>
<tbody>
<tr>
<td>[domain]/testdata</td>
<td>GET</td>
<td>RDF</td>
<td>Search/browse</td>
</tr>
<tr>
<td>[domain]/testdata/persistent</td>
<td>POST</td>
<td>RDF</td>
<td>Register new persistent test data collection</td>
</tr>
<tr>
<td>[domain]/testdata/persistent/[collection]</td>
<td>GET</td>
<td>RDF</td>
<td>Retrieve metadata of test data collection</td>
</tr>
<tr>
<td></td>
<td>GET</td>
<td>ZIP</td>
<td>Retrieve current test data version</td>
</tr>
<tr>
<td></td>
<td>PUT</td>
<td>RDF</td>
<td>Update metadata of test data collection</td>
</tr>
<tr>
<td></td>
<td>POST</td>
<td>RDF</td>
<td>Add test data version</td>
</tr>
</tbody>
</table>
## Accessing Test Data

<table>
<thead>
<tr>
<th>URI</th>
<th>Op.</th>
<th>Accept</th>
<th>Effect</th>
</tr>
</thead>
<tbody>
<tr>
<td>[domain]/testdata/persistent/[collection]/[version]</td>
<td>GET</td>
<td>RDF</td>
<td>Retrieve metadata of test data version</td>
</tr>
<tr>
<td>[domain]/testdata/persistent/[collection]/[version]</td>
<td>PUT</td>
<td>RDF</td>
<td>Update metadata of test data version</td>
</tr>
<tr>
<td></td>
<td>DELETE</td>
<td></td>
<td>Remove test data version</td>
</tr>
<tr>
<td>[domain]/testdata/persistent/[collection]/[version]/pub</td>
<td>POST</td>
<td>RDF</td>
<td>Publish test data version</td>
</tr>
<tr>
<td>[domain]/testdata/persistent/[collection]/[version]/pub</td>
<td>GET</td>
<td>RDF</td>
<td>Retrieve metadata of test data version</td>
</tr>
<tr>
<td></td>
<td>GET</td>
<td>ZIP</td>
<td>Retrieve data of publish test data version</td>
</tr>
<tr>
<td></td>
<td>DELETE</td>
<td>ZIP</td>
<td>Unpublish test data version</td>
</tr>
</tbody>
</table>
## Accessing Test Data

<table>
<thead>
<tr>
<th>URI</th>
<th>Op.</th>
<th>Accept</th>
<th>Effect</th>
</tr>
</thead>
<tbody>
<tr>
<td>[domain]/testdata/persistent/[collection]/[version]/suite/</td>
<td>GET</td>
<td>RDF</td>
<td>Retrieve suite metadata</td>
</tr>
<tr>
<td>[domain]/testdata/persistent/[collection]/[version]/suite/[suite item]/item/[data item]</td>
<td>GET</td>
<td></td>
<td>Retrieve data item</td>
</tr>
<tr>
<td>[domain]/testdata/persistent/[collection]/[version]/suite/[suite item]/component/[component type]</td>
<td>GET</td>
<td></td>
<td>Retrieve component</td>
</tr>
</tbody>
</table>
Synthetic Test Data Generators

• Test Data Repository Service supports storage and on-the-fly generation of synthetic test data using a user-defined generator

• Synthetic Test Data Generator API enables implementation of a generator
TOOLS REPOSITORY SERVICE
Tools Repository Service

- Manages *tools* and *tool version* that are evaluated on the platform
- Tools are expected to be packaged such that they can be integrated in an evaluation workflow
Describing Tools

Artifact

hasVersion

Artifact Version

rdfs:subClassOf

rdfs:subPropertyOf

rdfs:subClassOf

Tool

hasToolVersion

Tool Version

Tool

hasToolCategory

Tool Category

 rdf:type

Ontology EngineeringTool

ReasoningOr StorageTool

Ontology MappingTool

SemanticSearch Tool

SemanticWeb ServiceTool

Operating System

runsOnSystem

Tool Version

runsOnPlatform

Hardware Platform
RESULTS REPOSITORY SERVICE
Results Repository Service

• Manages raw results and interpretations thereof that are created during an evaluation of a tool
• Can be described using suite ontology
Describing Results

![Diagram showing the relationships between different result categories: Artifact, Result, Raw Result, Interpretation, Execution Request, and Result Category. Relationships include rdfs:subClassOf and hasResultCategory.]
THANK YOU!