OGSA-DAI Architecture

OGSA-DAI Technology Update
GridWorld Community Activity
GGF15, Boston, MA (USA)

Amy Krause
EPCC
a.krause@epcc.ed.ac.uk
Outline

- Data Services
- Internals
- Extensibility Points
- Perform Documents
- Sessions
- The Activity Engine
- Activities
- Data Resources
• OGSA-DAI uses data services to represent and provide access to a number of data resources
Data Service Internals

- **Query Activity**
- **Transform Activity**
- **Delivery Activity**
- **Activity Engine**
- **Role Mapper**

Connections:
- Query Activity to Transform Activity
- Transform Activity to Delivery Activity
- Delivery Activity to Query Activity
- Transform Activity to Activity Engine
- Activity Engine to Delivery Activity
- Role Mapper to Activity Engine

Data Flow:
- Query to Data Resource
- Data Resource to Transform Activity
- Transform Activity to Delivery Activity
- Delivery Activity to Response Document
- Activity Engine to Delivery Activity
- Role Mapper to Data Resource

Roles:
- **Session ID**
- **Credentials**
- **Role**
Extensibility Points

- Extensibility points in OGSA-DAI can be used to implement very powerful additions
  - Activity Framework
  - Data Resources
  - Role Mapping
  - Engine
- Allow applications to be tailored for specific tasks
The Activity Engine

- The Engine is the central OGSA-DAI component
- Dictates behaviour when perform documents are submitted
  - Manages concurrent requests and sessions
  - Parses and validates perform document
  - Identifies required activities implementations
  - Processes activities
  - Composes response document
  - Returns response document to the service layer
Concurrent Requests

• New in Release 7:
• Concurrent requests
• Request queueing
• Sessions
Perform Documents

• Perform documents
  – Encapsulate a serialisation of multiple interactions with a service into a single interaction
  – Abstract each interaction into an “activity”
  – Data can flow from one activity to another
  – No control constructs present
    – no conditionals, loops or variables

• Not intended for human consumption
  – Generated and processed by client toolkit
<?xml version="1.0" encoding="UTF-8"?>

<perform xmlns="http://ogsadai.org.uk/namespaces/2005/03/types">

  <documentation>
  This example performs a simple select statement to retrieve 100 rows from the test database. The results are delivered in the response document.
  </documentation>

  <sqlQueryStatement name="statement">
  <expression>select * from littleblackbook where id &lt;= 100</expression>
  <webRowSetStream name="statementOutput"/>
  </sqlQueryStatement>

</perform>
Sessions

• Sessions provide the ability to store state across multiple requests
  – e.g. a JDBC connection can be shared in order to process multiple requests in a transaction

• State is stored and modified in property objects which can be accessed by all activities within a session
Sessions

- A new session for multiple request must be requested explicitly, by sending a `openSession` request
- The service provides a unique session ID
- Follow-on requests may join this session by referencing the session ID
- A session may be terminated explicitly or left to expire

- An implicit session is started automatically by any request that doesn’t reference another session
  - This session is terminated after the request
OGSA-DAI

Sessions

Service

creates

Session

uses

open session

Session ID

Client

6 October 2005

http://www.ogsadai.org.uk/
Activities

• An Activity dictates an action to be performed
  – Query a data resource
  – Transform data
  – Deliver results

• Engine processes a sequence of activities

• Subset of activities available to a Data Resource
  – Specified in configuration files

• Data can flow between activities
OGSA-DAI Deck of Activities

- Relational Activities
  - SQLUpdateStatement
  - UpdateStatement

- XML Activities
  - XPathStatement
  - xsltTransform

- Delivery Activities
  - deliverToGDT
  - deliverFromGFTP
• Predefined activities fall into three categories:

• Statement
  –Interact with the data resource, e.g. direct an SQL query to a DBMS

• Delivery
  –Deliver data to a third party

• Transform
  –Perform transformations on data, e.g. XSL Transform, compression
Building Blocks – Predefined Activities

- sqlQueryStatement
- sqlStoredProcedure
- sqlUpdateStatement
- sqlBulkLoadRowset
- sqlStoredProcedure
- sqlQueryStatement
- xPathStatement
- xQueryStatement
- xmlResourceManagement
- xmlCollectionManagement
- relationalResourceManager

- DeliverFromGDT
- DeliverToGDT
- DeliverToStream
- DeliverFromGFTP
- DeliverToGFTP
- DeliverToURL
- DeliverFromURL
- DeliverToStream
- DeliverFromGDT

- gzipCompression
- xslTransform
- zipArchive
- inputStream
- outputStream
- Extensibility point
- All Activity implementations extend the abstract Activity class
Activity Inputs and Outputs

- Activities read and write blocks of data
  - Allows efficient streaming between activities
  - Reduces memory overhead

- A block is a Java Object
  - Untyped but usually a String or byte array

- Interfaces for reading and writing
  - BlockReader and BlockWriter
Custom Activities

• Users can develop additional activities
  – To support different query languages
    – e.g. XQuery
  – To perform different kinds of transformation
    – e.g. STX
  – To deliver results using a different mechanism
    – e.g. WebDAV

• An activity has the following components
  – Java implementation
  – XML schema
  – Configuration (optional)
  – Client toolkit implementation (optional)
Advantages of the Activity Model

- Avoid multiple message exchanges
- Extensible
  - Developers can add functionality
  - Could import third party trusted activities
- Allows for optimisation
  - Engine can optimise workflow
• Provides clients with a concept of a logical data resource against which activities can be run.
• Holds all the configuration information needed by activities to access the external physical data resources that make up the logical data resource.
• Manages the activities' interaction with the external physical data resources.
Data Resources

- Provided for JDBC, XML:DB and file systems
- Can be extended for any other data source
  - e.g. web server, WebDAV server, streams

Diagram:
- Relational database
  - open connection
  - close connection
- JDBC Connection Manager
  - open connection
  - release connection
- SQL Query Statement
Summary

- The Engine is the central component of a data service
- Activities perform actions
  - Querying, Updating
  - Transforming
  - Delivering
- Data Resources manage access to underlying data resources
- Architecture designed for extensibility
  - New Activities
  - New Role Mappers
  - New Data Resources