Introduction to Globus Toolkit 4

Gergely Sipos
MTA SZTAKI
sipos@sztaki.hu
Credits

- Globus Toolkit v4 is the work of many talented Globus Alliance members, at
  - Argonne Natl. Lab & U.Chicago
  - USC Information Sciences Corporation
  - National Center for Supercomputing Applns
  - U. Edinburgh
  - Swedish PDC
  - Univa Corporation
  - Other contributors at other institutions
- Supported by DOE, NSF, UK EPSRC, and other sources
Overview

- Web services
- Grids meets Web services: WSRF
- WSRF based services in Globus Toolkit 4
The definition of WSRF means that Grid and Web communities can move forward on a common base.
Web services technology

• Web designed for application to human interactions
• Served very well its purpose:
  ◆ Information sharing: a distributed content library.
  ◆ Enabled B2C e-commerce.
  ◆ Non-automated B2B interactions.
• How did it happen?
  ◆ Built on very few standards: http + html
  ◆ Shallow interaction model: very few assumptions made about computing platforms.
• The Web is everywhere. There is a lot more we can do!
• Current approach is ad-hoc on top of existing standards.
  ◆ e.g., application-to-application interactions with HTML forms.

• Goal: enabling systematic application-to-application interaction on the Web.
What is the WS technology?

- **Web services define a technique**
  - For describing software components to be accessed
  - Methods for accessing these components
  - Discovery methods that enable the identification of relevant service providers

- A web service is a piece of software that is made available on the Internet and utilizes a standardized XML messaging system. In other words a web service is a remote procedure call over the Internet using XML messages.

- **Web services standards are being defined within the W3C (World Wide Web Consortium) and other standard bodies and form the basis for major new industry initiatives such as**
  - Microsoft .Net
  - IBM Dynamic e-Business
  - Sun One, ...
WS standards 1: SOAP and WSDL

- **SOAP** provides a means of messaging between a service provider and a service requester.
- **SOAP** is a simple enveloping mechanism for XML payloads that defines an RPC convention.
- **SOAP** is independent of the underlying transport protocol
- **SOAP** client reads a **WSDL file** to get
  - the address and message information of a web service.
- Once the **WSDL file** is read, the client can start sending SOAP messages to the web service.
- Benefit: loosely coupling components by document oriented communication
A WSDL example

```xml
<wsdl:definitions targetNamespace="...">
  <wsdl:types>
    <schema>
      <xsd:element name="fooInput" .../>
      <xsd:element name="fooOutput" .../>
    </schema>
  </wsdl:types>
  <wsdl:message name="fooInputMessage">
    <part name="parameters" element="fooInput"/>
  </wsdl:message>
  <wsdl:message name="fooOutputMessage">
    <part name="parameters" element="fooOutput"/>
  </wsdl:message>
  <wsdl:portType name="fooInterface">
    <wsdl:operation name="foo">
      <input message="fooInput"/>
      <output message="fooOutput"/>
    </wsdl:operation>
  </wsdl:portType>
</wsdl:definitions>
```
How can I discover business partners with compatible web service solutions?

How do let other business know about my exposed web services?

*Web services are great, after you find out about them, but the discovery process is difficult*

*Information system for Web services:*
  *UDDI - Universal Description, Discovery and Integration*
The WS vision

1. SW companies, standards bodies, and programmers populate the registry with descriptions of different types of services

2. Businesses populate the registry with descriptions of the services they support

3. UBR assigns a programmatically unique identifier to each service and business registration

4. Marketplaces, search engines, and business apps query the registry to discover services at other companies

5. Business uses this data to facilitate easier integration with each other over the Web

Business processes realized by on-demand workflows of Web services
The server side in a Web Services application

- A piece of software that knows how to handle HTTP messages (Apache HTTP server)
- A piece of software that provides a 'living space' for applications that must be accessed by different clients (Jakarta Tomcat)
- A piece of software that knows how to handle SOAP requests and responses (Apache Axis)
Grid community meets Web services: Open Grid Services Architecture (OGSA)


- Service orientation to virtualize resources
  - **Everything is a service!**
- From Web services
  - Standard interface definition mechanisms
  - Evolving set of other standards: security, etc.
- From Grids (Globus Toolkit)
  - Service semantics, reliability & security models
  - Lifecycle management, discovery, other services
- OGSA implementation: WSRF – A framework for the definition & management of composable, interoperable services
WSRF: The Web Services Resource Framework

- Web services technology does not give support for state management
- WSRF: It's all about state
WSFR as a stateful Web Service invocation
The resource approach to statefulness
WS-Resource
The WSRF specification

The Web Services Resources Framework is a collection of 4 different specifications:

- WS-ResourceProperties
- WS-ResourceLifetime
- WS-ServiceGroup
- WS-BaseFaults

- Related specifications
  - WS-Notification
  - WS-Addressing
The Emergence of Open Grid Standards

- Computer science research → Managed shared virtual systems
- Web services, etc. → Open Grid Services Arch
- Internet standards → Globus Toolkit
  - Internet standards
  - Custom solutions
  - Defacto standard
  - Single implementation

WSRF structure

- A standard substrate: **the Grid service**
  - A Grid service is a special type of Web service
  - Standard interfaces and behaviors that address key distributed system issues: naming, service state, lifetime, notification
- ... supports standard service specifications
  - Agreement, data access & integration, workflow, security, policy, diagnostics, etc.
  - Target of current & planned OGF efforts
- ... and arbitrary application-specific services based on these & other definitions
Why Open Standards Matter

- Ubiquitous adoption demands open, standard protocols
  - Standard protocols enable *interoperability*
  - Avoid product/vendor lock-in
  - Enables innovation/competition on end points

- Further aided by open, standard interfaces and APIs
  - Standard APIs enable *portability*
  - Allow implementations to port to different vendor platforms
Web services vs. Grid services

- “Web services” address discovery & invocation of persistent services
  - Interface to persistent state of entire enterprise
- In Grids we also need transient services, created/destroyed dynamically
  - Interfaces to the states of distributed activities
  - E.g. workflow, video conf., dist. data analysis
- Significant implications for how services are managed, named, discovered, and used
  - In fact, much of our work is concerned with the management of services
Open Grid Services Infrastructure (OGSI) Specification

- Defines fundamental interfaces (using extended WSDL) and behaviors that define a Grid Service
  - A unifying framework for interoperability & establishment of total system properties
- Defines WSDL conventions and extensions
  - For describing and naming services
- Defines basic patterns of interaction, which can be combined with each other and with custom patterns in a myriad of ways
OGSI: Standard Web Services Interfaces & Behaviors

- **Naming and bindings (basis for virtualization)**
  - Every service instance has a unique name, from which can discover supported bindings

- **Lifecycle (basis for fault resilient state management)**
  - Service instances created by factories
  - Destroyed explicitly or via soft state

- **Information model (basis for monitoring & discovery)**
  - Service data (attributes) associated with GS instances
  - Operations for querying and setting this info
  - Asynchronous notification of changes to service date

- **Service Groups (basis for registries & collective svcs)**
  - Group membership rules & membership management

- **Base Fault type**
Relationship between OGSA, GT4, WSRF, and Web Services
Globus Toolkit version 2
(based on custom protocols)
Globus Toolkit version 3
OGSI based (~pre WSRF)

<table>
<thead>
<tr>
<th>Web Services Components</th>
<th>Non-WS Components</th>
</tr>
</thead>
<tbody>
<tr>
<td>Community Authorization</td>
<td>MDS3</td>
</tr>
<tr>
<td>Data Access &amp; Integration</td>
<td>Java WS Core</td>
</tr>
<tr>
<td>Reliable File Transfer</td>
<td>Grid Resource Alloc. Mgmt (WS GRAM)</td>
</tr>
<tr>
<td>GridFTP</td>
<td>Grid Resource Alloc. Mgmt (GRAM)</td>
</tr>
<tr>
<td>Replica Location</td>
<td>Monitoring &amp; Discovery (MDS)</td>
</tr>
<tr>
<td>Pre-WS Authentication Authorization</td>
<td>C Common Libraries</td>
</tr>
<tr>
<td>Security</td>
<td>eXtensible IO (XIO)</td>
</tr>
<tr>
<td>Data Mgmt</td>
<td>Execution Mgmt</td>
</tr>
<tr>
<td>Execution Mgmt</td>
<td>Info Services</td>
</tr>
<tr>
<td>Info Services</td>
<td>Common Runtime</td>
</tr>
</tbody>
</table>
**Globus Toolkit version 4**

WSRF based

---

**Earliest**
- Data Replication
- Community Scheduling Framework
- Grid Telecontrol Protocol

**Today**
- Reliable File Transfer
- Grid Resource Allocation & Management
- WebMDS
- Python WS Core

**In this presentation**
- Trigger
- Index
- Replica Location

**Next presentation**
- Monitoring & Discovery
- Extensible IO (XIO)
- Indexing

---

**Security**
- Data Mgmt
- Execution Mgmt
- Info Services
- Common Runtime

---

**Contrib/Preview**
- Web Services Components
- Non-WS Components

---

**Core**
- Delegation
- Community Authorization
- Pre-WS Authentication Authorization
- Credential Mgmt

---

www.globus.org

---

Earlier today

In this presentation

Next presentation
Globus Toolkit: Open Source Grid Infrastructure

Globus Toolkit v4
www.globus.org
GT4 Data Management

- **Stage/move** large data to/from nodes
  - GridFTP, Reliable File Transfer (RFT)
  - Alone, and integrated with GRAM
- **Locate** data of interest
  - Replica Location Service (RLS)
- **Replicate** data for performance/reliability
  - Distributed Replication Service (DRS)
- **Provide access** to diverse data sources
  - File systems, parallel file systems, hierarchical storage: GridFTP
  - Databases: OGSA DAI
GridFTP

- A high-performance, secure, reliable data transfer protocol optimized for high-bandwidth wide-area networks
- GridFTP server ~ high performance FTP server with GSI
- Multiple nodes work together and act as a single GridFTP server
- Each node moves (reads or writes) only the pieces of the file that it is responsible for.
- Pluggable
  - Front-end: e.g., future WS control channel
  - Back-end: e.g., HPSS, cluster file systems
  - Transfer: e.g., UDP, NetBLT transport
Striped GridFTP Service

- A distributed GridFTP service that runs on a storage cluster
  - Every node of the cluster is used to transfer data into/out of the cluster
  - Head node coordinates transfers
- Multiple NICs/internal busses lead to very high performance
  - Maximizes use of Gbit+ WANs
Reliable File Transfer: Third Party Transfer

- Fire-and-forget transfer
- Web services interface
- Many files & directories
- Integrated failure recovery
- Has transferred 900K files
Data services on a Grid: role of OGSA-DAI

Simple data files
- Middleware supporting
  - Replica files
  - Logical filenames
  - Catalogue: maps logical name to physical storage device/file
  - Virtual filesystems, POSIX-like I/O

Structured data
- RDBMS, XML databases
- Requires extendable middleware tools to support move computation near to data, easy access, controlled by AA, integration and federation
The OGSA-DAI Framework
Globus Toolkit:
Open Source Grid Infrastructure

Globus Toolkit v4
www.globus.org
Execution Management (GRAM)

- **Common WS interface to schedulers**
  - Unix, Condor, LSF, PBS, SGE, ...
- **More generally: interface for process execution management**
  - Lay down execution environment
  - Stage data
  - Monitor & manage lifecycle
  - Kill it, clean up
- **A basis for application-driven provisioning**
GT4 WS GRAM

- 2nd-generation WS implementation optimized for performance, flexibility, stability, scalability
- Streamlined critical path
  - Use only what you need
- Flexible credential management
  - Credential cache & delegation service
- GridFTP & RFT used for data operations
  - Data staging & streaming output
  - Eliminates redundant GASS code
GT4 WS GRAM Architecture

Service host(s) and compute element(s)

- GT4 Java Container
- GRAM services
- Delegation
- RFT File Transfer
- Job events
- Local job control
- Delegate
- FTP control
- FTP data
- Remote storage element(s)
- GridFTP
- Local scheduler
- User job
- Compute element
- SEG
- GridFTP adapter
- User job

Client

Job functions

Delegate

Delegation

Transfer request

FTPl control

FTP data

Remote storage element(s)
Delegated credential can be: Made available to the application
GT4 WS GRAM Architecture

Service host(s) and compute element(s)

Delegated credential can be:
Used to authenticate with RFT
GT4 WS GRAM Architecture

Delegated credential can be:
Used to authenticate with GridFTP
Submitting a Sample Job

- Specify a remote host with \texttt{-F}
- \texttt{-s} is short for \texttt{-streaming}
- The output will be sent back to the terminal, control will not return until the job is done

\texttt{globusrun-ws -submit -s}
\texttt{\hspace{1em} -F remote.cluster.hu -c /bin/hostname}
Describing complex jobs: RSL

globusrun-ws -submit -F remote.cluster.hu -f jobRSL.xml

<job>
<executable>/bin/echo</executable>
<argument>this is an example_string</argument>
<argument>Globus was here</argument>
<stdout>${GLOBUS_USER_HOME}/stdout</stdout>
<stderr>${GLOBUS_USER_HOME}/stderr</stderr>
</job>
<job>
  <executable>/bin/echo</executable>
  <directory>/tmp</directory>
  <argument>12</argument>
  <environment><name>PI</name>
    <value>3.141</value>
  </environment>
  <stdin>/dev/null</stdin>
  <stdout>stdout</stdout>
  <stderr>stderr</stderr>
</job>
Staging Data – Stage In

- GRAM’s RSL allows many fileStageIn/fileStageOut directives

```xml
<fileStageIn>
  <transfer>
    <sourceUrl>
      gsiftp://job.input.host:2811/bin/echo
    </sourceUrl>
    <destinationUrl>
      file:///${GLOBUS_USER_HOME}/my_echo
    </destinationUrl>
  </transfer>
</fileStageIn>
```
Staging Data – Stage Out

```xml
<fileStageOut>
  <transfer>
    <sourceUrl>
      file://${GLOBUS_USER_HOME}/stdout
    </sourceUrl>
    <destinationUrl>
      gsiftp://job.output.host:2811/tmp/stdout
    </destinationUrl>
  </transfer>
</fileStageOut>
```
Batch Submission

- Your client does not have to stay attached to the execution of the job
- `-batch` will disconnect from the job and output an End Point Reference (EPR)
  - You may redirect the EPR to a file with `-o`
  - Note: EPR → submitted job is a WS-resource
- Use the EPR file with `-monitor` or `-status`
- You may also kill the job using `-kill`
Specifying Scheduler Options

- RSL lets you specify various scheduler options
  - what queue to submit to
  - which project to select for accounting
  - max CPU and wallclock time to spend
  - min/max memory required
- All defined online under the schema document for GRAM
Choosing User Accounts

- You may be authorized to use more than one account at the remote site
- By default, the first listed in the grid-mapfile will be used
- You may request a specific user account using the `<localUserId>` element
Long term GRAM architecture
Workspace Service: The Hosted Activity

Client

Allocate/provision
Configure
Initiate activity
Monitor activity
Control activity

Policy
Activity
Environment

Interface
Resource provider
For Example ...

- **Deploy service**: JVM
- **Deploy container**: JVM
- **Deploy virtual machine**: VM, VM
- **Deploy hypervisor/OS**: Hypervisor/OS
- **Procure hardware**: Physical machine

Provisioning, management, and monitoring at all levels
Summary
The Globus Toolkit is a Collection of Components

- A set of loosely-coupled components, with:
  - Services and clients
  - Libraries
  - Development tools
- GT components are used to build Grid-based applications and services
  - GT can be viewed as a Grid SDK
- GT4 use WS protocols for service interactions
- GT4 services work according to WSRF behavior paradigms
Further readings

- **Service Oriented Architecture**

- **Web services**
  - Web Services Specifications - [http://www.w3.org/2002/ws/](http://www.w3.org/2002/ws/)

- **OGSA, WSRF**
Questions?