Introduce

Grid Service Authoring Toolkit

Shannon Hastings

hastings@bmi.osu.edu

Multiscale Computing Laboratory
Department of Biomedical Informatics
The Ohio State University

Globus World 2006
Outline

• Introduce Generated Grid Services
  – Driving requirements
  – Architecture
• Introduce
  – What can it do?
  – Architecture
  – Toolkit
• Future work
• Accessing data in a grid will not be enough. The data will need to be processed, transformed, analyzed, integrated, etc
• It should be very easy for service providers to deliver these services
• The services should take in and return strongly typed data
• These services should be easily discovered and invoked
• These services should be easily configured for security
All services must be presented as **Grid Services**.

They must have *strongly typed interfaces* using types which can be verified by an XML schema registered in the **Global Model Exchange GME**.

Must provide *standard service metadata* as well as service specific metadata.

The service must register itself and its metadata to an **Index Service**.

The service must provide *object oriented client API* for locating and invoking the service.
Introduce Goals

- A framework which enables fast and easy creation of strongly typed and highly interoperable grid services.
- Provide easy to use graphical service authoring tool.
- Hide all “grid-ness” from the developer.
- Utilize best practice layered grid service architecture.
- Handle all service architecture requirements of the grid.

- Toolkit for creating and manipulating grid services
  - Command line and GUI tools for service skeleton generation and automatic service/client code generation
- Utilizes other grid services and architecture components
  - GME for data types
  - Security service architecture (GSI)
  - Advertisement and Registration configuration and caGrid Index Service for discovery
Introduce Graphical Development Environment (GDE)

- GUI for creating and manipulating a grid service
  - Provides means of simple creation of service skeleton that a developer can then implement, build, and deploy
  - Automatic code generation of complete caBIG compliant grid service which is configured to provide:
    - Security
    - Advertisement
    - Discovery
    - Complete Client API
  - Provides a set of tools which enable the developer to add/remove/modify/import methods of the service as well create sub-services.
    - Automatic code generation of all the required code, Globus grid service code/configuration, service configuration, implementation of the client, and stubbed implementation of the service
Introduce Generated Grid Service Architecture

- Base service is a GT4 based WSRF capable grid service.
- Utilize compositional inheritance (in lieu of non-standard port type extensions) to enable the service to inherit required features such as providing service security metadata and access to resource properties.
- Utilize JNDI for server side configuration properties, and resources and resource properties.
- Provide client and service side wrappers which implement the service designers interface as opposed to the document literal interface generated by Axis.
- Provide metadata registration to the index service by configuring the Resource to register it’s service groups to a predefined caGrid MDS based Index Service.
Introduce Grid Service

Resource Properties

Uses the services JNDI file to configure resources and resource properties for the service so that the service can leverage the resource properties.

Metadata Registration

Utilize the globus class “gov.nih.nci.cagrid.service.globus.resource.ResourceConfiguration” and configure it in the services’ JNDI file so that the resource registration configuration file resource. This registration file will tell MDS what to resource properties to register to which index services.

Core Service

Provides the methods that this particular service is presenting as an analytical tool.

Service Security Metadata

Uses the “securityMetadata” parameter in the server-deploy.wsdd to point to the configuration file (etc/\<service-name>-security-metadata.xml) describing the services security requirements for each service and each method in the service. This data is exposed through an anonymous function called getServiceSecurityMetadata() which is available for each caGrid service.
Introduce Service Development Overall Flow
The Introduce Service Properties are a set of base properties required for creating this service such as desired location to create the service, namespace to use, package name to use, and service name.
Command Line Creation

- Variables can be preset in a properties file to use non-interactively
  - call the ant target and set the introduce.properties variable to point to this properties file:
    - ant -Dintroduce.properties=myAnalytical.properties createService

- An example properties file:

```properties
introduce.skeleton.destination.dir=C:\HelloWorld
introduce.skeleton.service.name=HelloWorld
introduce.skeleton.namespace.domain=http://cagrid.nci.nih.gov/HelloWorld
introduce.skeleton.package=gov.nih.nci.cagrid.helloworld
```
Created Skeleton Layout

- = Introduce generated
- = Globus/Axix generated
- = developers contribution
Created Skeleton Layout (cont)

- = implements the developer defined interface and calls into the generated client port type stub.

- = the developer defined grid service interface

- = manages the resources of this grid service

- = implements the port type and calls into the actual clean unboxed interface the developer defined.

- = developers implementation of the defined interface.
Created Skeleton Layout (cont)

- service metadata registration configuration
- describes the services security configuration
- services WSDL file for axis
- configuration files for eclipse development
- ant build files
- client configuration file for axis
- deployment time service properties
- introduce representation of
- service resources configuration
- namespace mappings for axis
- server configuration file for axis
The Introduce Service Description document is the input to the introduce engine. The user can hand craft this document and call the engine via command line or API or can use the Introduce GDE to modify this document and execute the engine.
• run the command
  – “ant resync” –Dintroduce.skeleton.destination.dir=<path of service>
  – will read in the “introduce.xml” document and determine the changes between the developers new description of the service and the actual code
  – will then modify the skeleton appropriately:
  • Modify all source files
  • Modify all security configuration files
  • Modify the WSDL, JNDI, and other configuration files for the service
Inside the Introduce created service

• Services have many moving and configurable parts which are supported/enabled/generated via Introduce:
  – Invocation
  – Security
  – WSRF Resources
  – Service Properties
  – Advertisement
  – Discovery

• The Introduce Toolkit can keep all these features in sync as the developer creates and modifies his/her grid service
Introduce is a framework where it’s core functionality can be extended to provide customized solutions by leveraging the Introduce toolkit.

There are currently two types of extensions in Introduce:
- Service
- Types Discovery

Extensions provide an “extension.xml” extension description and configuration. The “extensions” directory of Introduce holds the extensions in separate folders enabling them to easily be added and removed from the Introduce toolkit.
Service Development Extensions

Timeline for execution of service extension components.
Introduce Demonstration
Conclusion

• **Requirements**
  – Strongly typed grid service development

• **Architecture**
  – Grid service framework which is encapsulated and layered on Globus 4

• **Introduce Service Layout**
  – Simple grid service layout making it easy to locate and manage pieces of the implementation

• **Introduce Toolkit**
  – Enables easy development of grid services of any type size of shape.
Introduce Team

Shannon Hastings
Scott Oster
Stephen Langella
David Ervin
Tahsin Kurc
Joel Saltz
Get the software

• Beta release comes with caGrid beta software stack.
  – https://cabig.nci.nih.gov/News_Folder/caGrid_1.0_Beta

• 1.0 release will be available in December and will be soon available on GlobDev (globus development projects location)
Questions
Populate required variables for service creation

- **Name** = published service name
- **Creation Direction** = directory to create the skeleton
- **Package** = the java package you wish to use for your service
- **Namespace Domain** = the namespace to be used to define the service interface and types
- **Extensions** = extensions are plugins that you want to be executed during the development of this service
Introduce GDE Service Modification

- Service modification is the main GDE component. It enables the modification of all caGrid grid service features.

- The first main task of any service modification is to discover and add the Types to be used for operation input/output and/or metadata.
Once types are discovered and added to the grid service operations can be added/removed/modified or imported.
The operation modification screen will enable the complete description of the method signature, including inputs output and faults, its security configuration, and options for importing the definition and/or the implementation from another grid service.
• Adding and removing service metadata (resource properties) to the service is done under the Service Metadata tab.
• This feature enables the addition or removal of metadata types which will be populated at deployment time and can be accessed at runtime.
• Each metadata property can be loaded file statically at grid service startup or populated by the service dynamically at runtime.
• Each metadata property can be selected to be registered with an index service so that they can be used during discovery.
Introduce GDE Service Properties Modification

- Service properties are properties which only the service itself is exposed to.
- These properties are intended to be configured at deployment time to enable the service leverage them for its own configuration.
Service security, as with operation security, enables the developer to create specific security constraints which the caller must meet in order to execute any method of the service.

Operation security configuration will take precedence over this overall service security configuration.
• The Service Contexts tab in the Modification View is intended to address enable developers to take advantage of the WSRF support provided by the Globus Toolkit.

• Each service context added is essentially another resource definition in the service. With each new service definition the developer can add methods which are specific to that resource as well as properties to that resource.
Introduce is provides an extension framework where it’s core functionality can be extended to provide customized solutions by leveraging the Introduce toolkit.

- There are currently two types of extensions in Introduce:
  - Service
  - Types Discovery

- Extensions provide an “extension.xml” extension description and configuration. The “extensions” directory of Introduce hold the extensions in separate folders enabling them to easily be added and removed from the Introduce toolkit.
Introduce Extension Framework (cont)
Service Development Extensions

Timeline for execution of service extension components.
Service Development Extensions

xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:gme://gov.nih.nci.cagrid.introduce/1/Extension" extensionType="SERVICE"
extensionPreferencesPanel="gov.nih.nci.cagrid.introduce.portal.extension.example.ExampleExtensionPreferencesConfigurationPanel"
 xsi:ServiceExtensionDescription displayName="Example Extension" name="service_example"
 xsi:CreationPostProcessor>gov.nih.nci.cagrid.introduce.extension.example.ExampleCreationPostProcessor</xsi:CreationPostProcessor>
 xsi:CreationUIPanel>gov.nih.nci.cagrid.introduce.extension.example.ExampleCreationUIPanel</xsi:CreationUIPanel>
 xsi:CodegenPreProcessor>gov.nih.nci.cagrid.introduce.extension.example.ExampleCodegenPreProcessor</xsi:CodegenPreProcessor>
 xsi:CodegenPostProcessor>gov.nih.nci.cagrid.introduce.extension.example.ExampleCodegenPostProcessor</xsi:CodegenPostProcessor>
 xsi:ServiceModificationUIPanel>gov.nih.nci.cagrid.introduce.extension.example.ExampleServiceModificationPanel</xsi:ServiceModificationUIPanel>
</xsi:ServiceExtensionDescription>
Data Type Discovery Extensions

- Data type discovery extensions are plugins to introduce to enable the discovery and usage of data types in the grid service.
- Currently there are discovery tools for GME and caDSR as well as utility ones for globus and the file system.
Data Type Discovery Extensions
Conclusion

• Requirements
  – Basic strongly typed grid requirements plus semantically interoperable caBIG requirements

• Architecture
  – Grid service framework which is encapsulated and layered on Globus 4.0.2

• Introduce Service Layout
  – Simple grid service layout making it easy to locate and manage pieces of the implementation

• Introduce Toolkit
  – Enables easy development of caBIG compliant grid service of any type size of shape.
Future Work

- Stabilization, clean up.
- Creation of Metadata population utilities.
- Registration of service descriptions to caDSR.
- Streamline common caBIG usage scenarios such as wizards for SDK data services.
- Enhanced user documentation.
Introduce Team

Shannon Hastings
Scott Oster
Stephen Langella
David Ervin
Tahsin Kurc
Joel Saltz
Introduce
Grid Service Authoring Toolkit

Shannon Hastings
hastings@bmi.osu.edu

Multiscale Computing Laboratory
Department of Biomedical Informatics
The Ohio State University