An Open Grid Services Architecture

Ian Foster, Steven Tuecke, ANL
Karl Czajkowski, Carl Kesselman, USC/ISI
Jeff Frey, Steve Graham, Jeff Nick, IBM

OMG Web Services Workshop, March 7, 2002
The Grid World: Current Status

- Dozens of major Grid projects in scientific & technical computing/research & education
- Considerable consensus on key concepts and technologies
  - Open source Globus Toolkit™ a de facto standard for major protocols & services
  - Far from complete or perfect, but out there, evolving rapidly, and large tool/user base
- Industrial interest emerging rapidly
- Opportunity: convergence of eScience and eBusiness requirements & technologies
“Web Services”

- Increasingly popular standards-based framework for accessing network applications
  - W3C standardization; Microsoft, IBM, Sun, others
- **WSDL**: Web Services Description Language
  - Interface Definition Language for Web services
- **SOAP**: Simple Object Access Protocol
  - XML-based RPC protocol; common WSDL target
- **WS-Inspection**
  - Conventions for locating service descriptions
- **UDDI**: Universal Desc., Discovery, & Integration
  - Directory for Web services
The Need to Support Transient Service Instances

- “Web services” address discovery & invocation of persistent services
  - Interface to persistent state of entire enterprise
- In Grids, must also support transient service instances, 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 service instances
OGSA Design Principles

- Service orientation to virtualize resources
  - Everything is a service

- From Web services
  - Standard interface definition mechanisms:
    multiple protocol bindings, local/remote transparency

- From Grids
  - Service semantics, reliability and security models
  - Lifecycle management, discovery, other services

- Multiple “hosting environments”
  - J2EE, .NET, C, ...
OGSA Service Model

- System comprises (a typically few) persistent services & (potentially many) transient services
  - Everything is a service
- OGSA defines basic behaviors of services: fundamental semantics, life-cycle, etc.
  - More than defining WSDL wrappers
Open Grid Services Architecture: Fundamental Structure

1) **WSDL conventions and extensions** for describing and structuring services
   - Useful independent of “Grid” computing

2) **Standard WSDL interfaces & behaviors** for core service activities
   - portTypes and operations => protocols
WSDL Conventions & Extensions

- **portType** (standard WSDL)
  - Define an interface: a set of related operations

- **serviceType** (extensibility element)
  - List of port types: enables aggregation

- **serviceImplementation** (extensibility element)
  - Represents actual code

- **service** (standard WSDL)
  - instanceOf extension: map descr. -> instance

- **compatibilityAssertion** (extensibility element)
  - portType, serviceType, serviceImplementation
Structure of a Grid Service

Service Instantiation

Service Description

- service
  - instanceof instance
  - serviceImplementation
    - serviceType
      - PortType
- service
  - instanceof instance
  - serviceImplementation
    - serviceType
      - PortType
- service
  - instanceof instance
  - serviceImplementation
    - serviceType
      - PortType

= Standard WSDL

= compatibilityAssertion

April 22, 2002
Standard Interfaces & Behaviors: Four Interrelated Concepts

- **Naming and bindings**
  - Every service instance has a *unique name*, from which can discover *supported bindings*

- **Information model**
  - *Service data* associated with Grid service instances, operations for accessing this info

- **Lifecycle**
  - Service instances created by *factories*
  - Destroyed *explicitly* or via *soft state*

- **Notification**
  - Interfaces for *registering interest* and *delivering notifications*
OGSA Interfaces and Operations Defined to Date

- **GridService**  
  - FindServiceData  
  - Destroy  
  - SetTerminationTime

- **NotificationSource**  
  - SubscribeToNotificationTopic  
  - UnsubscribeToNotificationTopic

- **NotificationSink**  
  - DeliverNotification

- **Factory**  
  - CreateService

- **PrimaryKey**  
  - FindByPrimaryKey
  - DestroyByPrimaryKey

- **Registry**  
  - RegisterService
  - UnregisterService

- **HandleMap**  
  - FindByHandle

*Authentication, reliability are binding properties.*  
*Manageability, concurrency, etc., to be defined.*
Naming and Bindings

- Every service instance has a unique and immutable name: **Grid Service Handle** (GSH)
  - Basically just a URL
- Handle must be converted to a **Grid Service Reference** (GSR) to use service
  - Includes binding information; may expire
  - Separation of name from implementation facilitates service evolution
- The **HandleMap** interface allows a client to map from a GSH to a GSR
  - Each service instance has home HandleMap
Service Data

- A Grid service instance maintains a set of service data elements
  - XML fragments encapsulated in standard `<name, type, TTL-info>` containers
  - Includes basic introspection information, interface-specific data, and application data

- **FindServiceData** operation (GridService interface) queries this information
  - Extensible query language support

- See also notification interfaces
  - Allows notification of service existence and changes in service data
The **Registry** interface may be used to register Grid service instances with a registry
- A set of Grid services can periodically register their GSHs into a registry service, to allow for discovery of services in that set

Registrations maintained in a service data element associated with Registry interface
- Standard discovery mechanisms can then be used to discover registered services
- Returns a WS-Inspection document containing the GSHs of a set of Grid services
Lifetime Management

- GS instances created by factory or manually; destroyed explicitly or via soft state
  - Negotiation of initial lifetime with a factory (=service supporting Factory interface)
- **GridService** interface supports
  - **Destroy** operation for explicit destruction
  - **SetTerminationTime** operation for keepalive
- Soft state lifetime management avoids
  - Explicit client teardown of complex state
  - Resource “leaks” in hosting environments
Factory

- **Factory** interface’s **CreateService** operation creates a new Grid service instance
  - Reliable creation (once-and-only-once)
- **CreateService** operation can be extended to accept service-specific creation parameters
- Returns a **Grid Service Handle (GSH)**
  - A globally unique URL
  - Uniquely identifies the instance for all time
  - Based on name of a home handleMap service
Notification Interfaces

- **NotificationSource** for client subscription
  - One or more *notification generators*
    > Generates notification message of a specific type
    > Typed *interest statements*: E.g., Filters, topics, ...
    > Supports messaging services, 3rd party filter services, ...
  - Soft state subscription to a generator

- **NotificationSink** for asynchronous delivery of notification messages

- A wide variety of uses are possible
  - E.g. Dynamic discovery/registry services, monitoring, application error notification, ...
For More Information

- Two draft docs at www.globus.org/ogsa (hopefully at GGF site soon)
  - “The Physiology of the Grid: An Open Grid Services Architecture for Distributed Systems Integration”
  - “Grid Service Specification”
- Global Grid Forum
  - Proposed: Open Grid Services Infrastructure (OGSI) Working Group