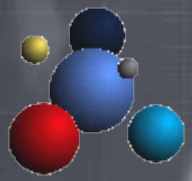
Análisis de la Arquitectura de Globus Toolkit 4

José Luis Vázquez Poletti

Grupo de Arquitectura de Sistemas Distribuidos y Seguridad
Dpto. Arquitectura de Computadores y Automática (DACYA)
Universidad Complutense de Madrid

http://asds.dacya.ucm.es/jlvazquez/
jlvazquez@fdi.ucm.es



¿Qué vamos a ver?

- Introducción a los Web Services
- WSRF de Globus
- Migración conceptual de GT 2.4 a GT 4
- Instalación y Configuración (Diario de Bitácora)

Morpheus: *"I imagine that right now you're feeling a bit like Alice, tumbling down the rabbit hole."*

Introducción a los Web Services

XML: Un problema cotidiano

- EJEMPLO: 750 recetas de cocina
  - ¿Cómo almacenar información para usarla con diferentes propósitos?
  - ¿Estructuración?

Agent Smith: *"Never sned a human to do a machine's job"*

XML: Usos

- Formato universal
  - SVG 1.0 (Scalable Vector Graphics)
  - DSML 1.0 (Directory Services Markup Language)
  - XHTML 1.0 (eXtensible HyperText Markup Language)

XML: Sintáxis

- Posible solución para problema de recetas:

```
<receta>
...
    <necesitamos>
        <ingrediente>2 cucharadas de azúcar</ingrediente>
        <ingrediente>3 manzanas</ingrediente>
    </necesitamos>
...
</receta>
```

XML: Utilidades

- Programas interpretan bien el dato
- Programas hablan entre ellos sin intervención humana
  - Computación Distribuida
  - Interoperatividad
  - Monitorización

Servicios Web XML

- Servicio Web = Rutina de Internet
- ¿Por qué Web?
  - Uso de HTTP: Métodos PUT y GET
- Resultado: Internet Paralelo sólo para máquinas

Agent Smith: *"Like the dinosaur... Look out that window. You had your time. The future is our world, Morpheus. The future is our time"*

Protocolos de Web Services

- 2 Tendencias: XML-RPC y SOAP
- Globus escoge SOAP:
  - Soporte completo y minucioso
  - Más potente
  - Más difícil

WSDL: Introducción y elementos fundamentales

- Lenguaje de Definición de Web Services
- Describe operaciones ofrecidas por Web Service
- Elementos:
  - **<portType>** Operaciones realizadas (comparable a librería de funciones)
  - **<message>** Mensajes usados
    - Divisibles en partes (parámetros de función)
  - **<types>** Tipos de datos usados
  - **<binding>** Protocolos de comunicación usados

WSDL: Ejemplo con elementos

```
<message name="getTermRequest">
  <part name="term" type="xs:string"/>
</message>
<message name="getTermResponse">
  <part name="value" type="xs:string"/>
</message>
 <portType name="glossaryTerms">
  <operation name="getTerm">
     <input message="getTermRequest"/>
     <output message="getTermResponse"/>
  </operation>
</portType>
```

WSDL: Ports

- Definen punto de conexión a Web Service
- Operaciones permitdas:
  - **One-Way:** Recibir mensaje sin devolver respuesta
  - **Request-Response:** Recibir mensaje y devolver respuesta
    - Más extendida
  - **Solicit-Response:** Enviar petición y esperar respuesta
  - **Notification:** Enviar mensaje sin esperar respuesta

WSDL: Ejemplo de Port 'One-Way'

```
<message name="newTermValues">
  <part name="term" type="xs:string"/>
  <part name="value" type="xs:string"/>
</message>
<portType name="glossaryTerms">
  <operation name="setTerm">
    <input name="newTerm" message="newTermValues"/>
  </operation>
</portType >
```

WSDL: Ejemplo Port 'Request-Response'

```
<message name="getTermRequest">
  <part name="term" type="xs:string"/>
</message>
<message name="getTermResponse">
  <part name="value" type="xs:string"/>
</message>
<portType name="glossaryTerms">
 <operation name="getTerm">
    <input message="getTermRequest"/>
    <output message="getTermResponse"/>
 </operation>
</portType>
```

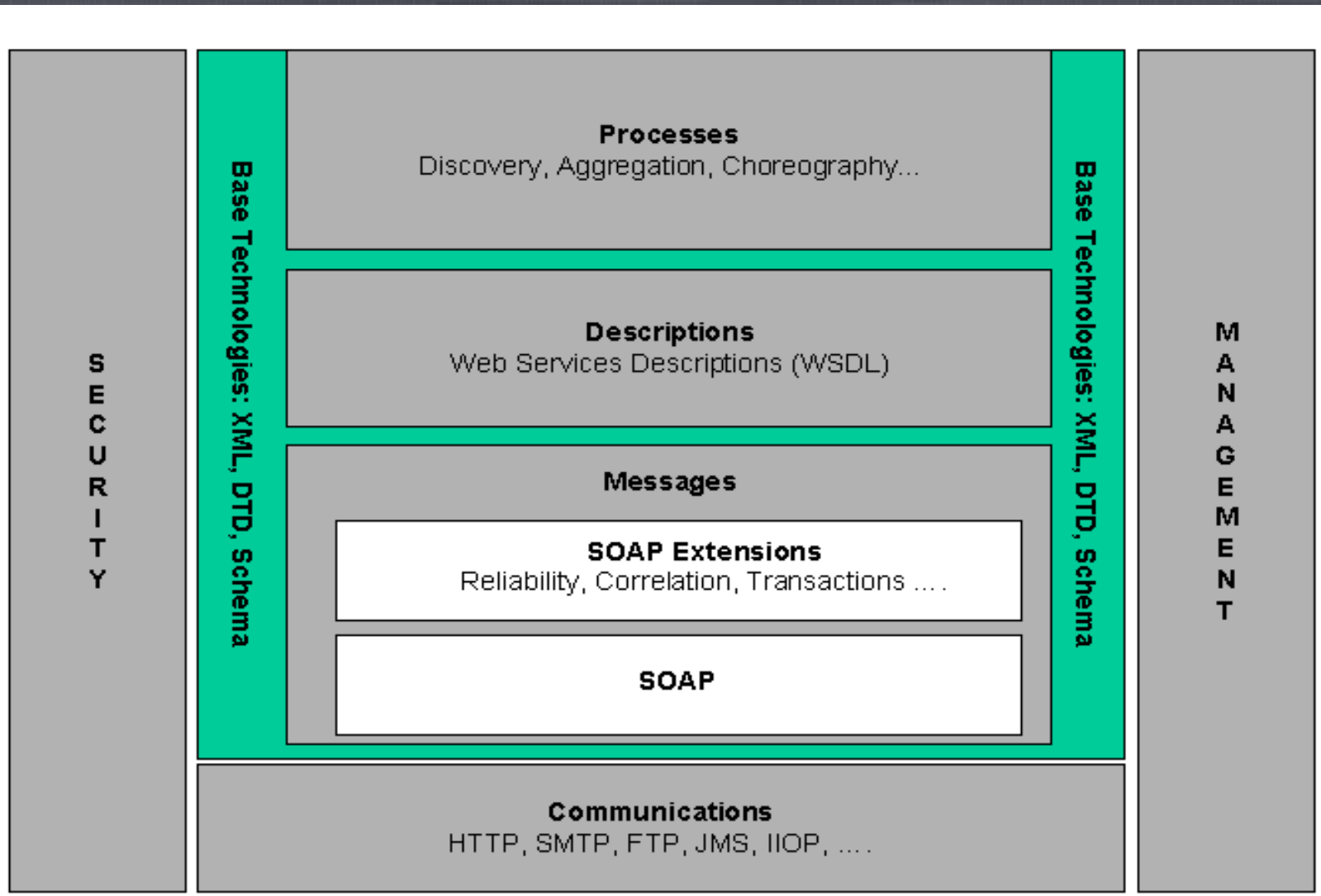
WSDL: Binding

- Atributos **binding**
  - Nombre
  - Tipos de atributos para **port** del **binding**
- Atributos **soap:binding**
  - Estilo: [rpc | document]
  - Transporte (HTTP)
- **Operation**
  - Cada operación ofrecida por **Port**
  - Se define acción SOAP

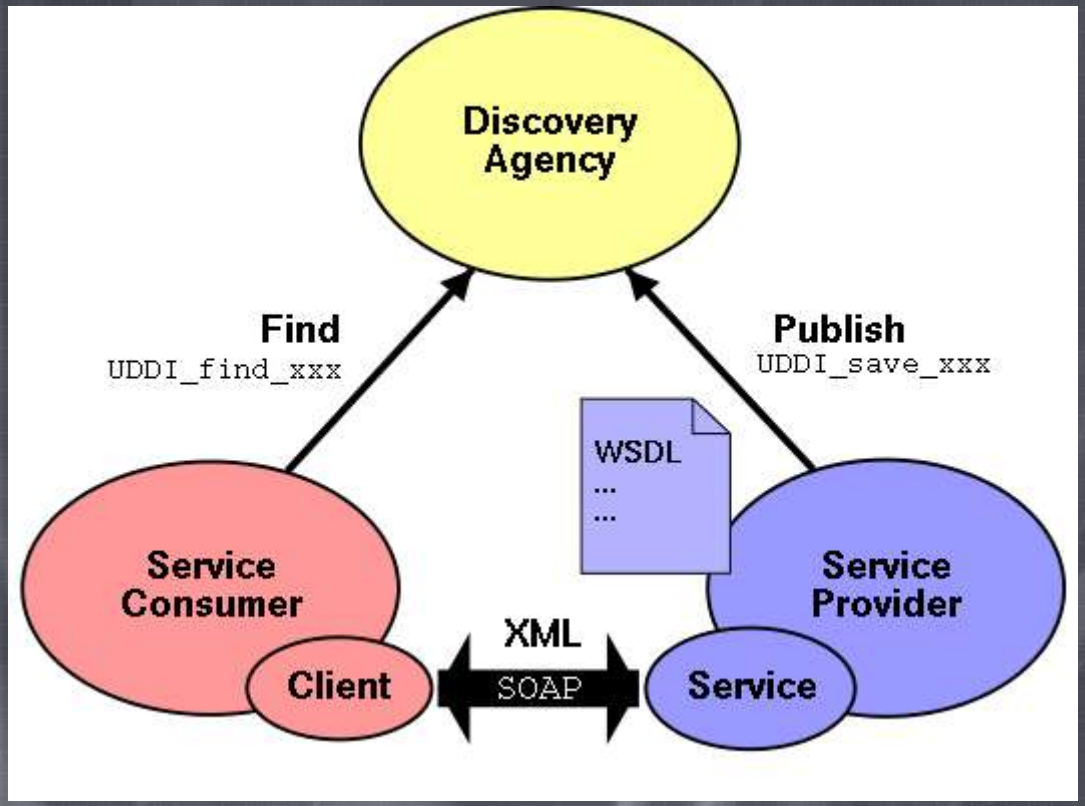
WSDL y UDDI

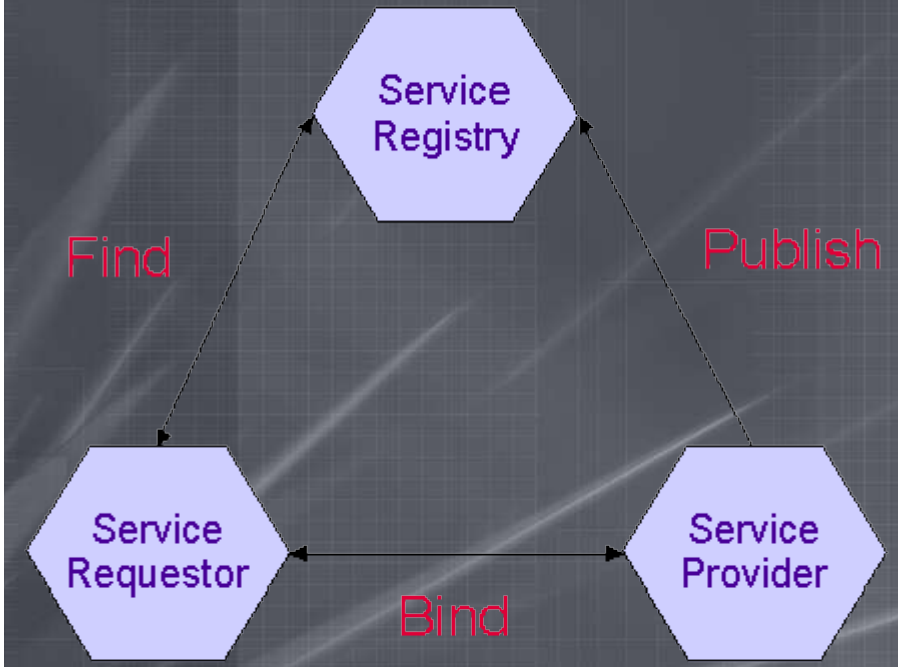
- Universal Description, Discovery and Integration
- Servicio de directorio donde:
  - Registrar Web Services
    - Sólo definidos con wsdl
  - Buscar Web Services

Juntando todo en la foto (I)



Juntando todo en la foto (II)





"Ahora tú": Casos prácticos

- **API de Google:** Buscador de escritorio
- **API de Babelfish:** Diccionario de escritorio
- **Farenheit – Celsius:** Cliente y Servidor

Morpheus: *"Neo, sooner or later you're going to realize just as I did that there's a difference between knowing the path and walking the path"*

“Ahora tú”: SOAP::Lite

- CPAN: Comprehensive Perl Archive Network

perl -MCPAN -e shell

install SOAP::Lite

quit

“Ahora tú”: API de Google (I)

- google.pl

```
#!/usr/bin/perl -w

use SOAP::Lite;

print "\nBuscando en Google \"$ARGV[0]\"...\n\n";

my $query = "$ARGV[0]";

my $googleSearch = SOAP::Lite -> service
      ("http://api.google.com/GoogleSearch.wsdl");

my $result = $googleSearch -> doGoogleSearch(

   "iwnUXUtHj3bteg5FWfBJDwui3SPeB+iy",     # Clave de
      acceso al API de Google

   $query,                    # Palabras clave de búsqueda

   0,                         # Indice del primer resultado
      mostrado

   10,                        # Numero de resultados obtenidos

   "false",                   # Filtro

   '',                        # Restricción

   "false",                   # Búsqueda Segura

   '',                        # lr

   'latin1',                  # ie

   'latin1'                   # oe

);

print $result->{resultElements}->[0]->{title};

print "\n";

print $result->{resultElements}->[0]->{URL};

print "\n";

print "\n¡Busqueda finalizada!\n";
```

“Ahora tú”: API de Google (II)

- Ejecución

[jlvazquez@auriga Seminario081004]$ ***./google.pl gridway+dacya***

Buscando en Google "gridway+dacya"...

Distributed Systems Architecture & Security

http://asds.dacya.ucm.es/GridWay/index.php

¡Busqueda finalizada!

“Ahora tú”: API de Babelfish (I)

- babelfish.pl

```
#!/usr/bin/perl -w
use SOAP::Lite;
print ">> $ARGV[0] en castellano es: ";
print SOAP::Lite
    ->proxy('http://services.xmethods.net/perl/soaplite.cgi')
    ->uri('urn:xmethodsBabelFish')
    ->BabelFish("en_es","$ARGV[0]")
    ->result;
print "\n";
```

“Ahora tú”: API de Babelfish (II)

- Ejecución

[jlvazquez@auriga Seminario081004]$ ***./babelfish.pl Grid***

>> Grid en castellano es: rejilla

“Ahora tú”: F – C (I)

- temper.cgi (Servidor)

```
#!/usr/bin/perl -w
 use SOAP::Transport::HTTP;
 SOAP::Transport::HTTP::CGI
   -> dispatch_to('Temperatures')
   -> handle;
 package Temperatures;
 sub f2c {
     my ($class, $f) = @_;
     return 5 / 9 * ($f - 32);
 }
 sub c2f {
     my ($class, $c) = @_;
     return 32 + $c * 9 / 5;
 }
```

“Ahora tú”: C – F (II)

- Ejecución (f -> c)

[jlvazquez@auriga Seminario081004]$ ***./temperatura.pl f2c 911***

488.333333333333

- Ejecución (c -> f)

[jlvazquez@auriga Seminario081004]$ ***./temperatura.pl c2f 20***

68

- Ejecución (???)

[jlvazquez@auriga Seminario081004]$ ***./temperatura.pl ImAWannabeHacker***

No humans allowed... only machines here!

WSRF de Globus

Retomando el concepto

- Web Services: Habilidad para acceder y manipular:
  - Estados
  - Valores persistentes
- WSRF: Estándar de Globus
  - Define convenciones sobre manejo de estados
  - Mejora (recursos):
    - Descubrimiento
    - Inspección
    - Interacción

WS-Resource: Definición

- WS-Resource
  - Web Service
  - Recurso asociado a estados
    - Asociación entre documento XML y Web Service
    - Accedido de acuerdo con patrón de recurso implícito
- WSRF permite (WS-Resource)
  - C.R.U.D.
  - 5 especificaciones

WS-ResourceLifetime (I)

- Mecanismos de destrucción de WS-Resources
  - Incluye destrucción “programada”
- Patrón WS-Resource Factory
  - Patrón de Diseño Factory:
    - Separa responsabilidad de creación compleja en objetos de apoyo
    - Ocultación de lógica de creación potencialmente compleja
    - Posibilidad de introducir estrategias para mejorar rendimiento de gestión de memoria
  - ¿Quién es el responsable de creación de objetos?

WS-ResourceLifetime (II)

- Identidad del WS-Resource
  - Identificación por componente Web Service
    - Incluye identificador del recurso en propiedades de referencia del 'endpoint' (WS-Addressing)
    - 'Endpoint' -> 'WS-Resource Calificado'
      - Disponible a otras entidades
- Destrucción de un WS-Resource
  - Uso de 'endpoint' de 'WSR Calificado'
  - Tipos:
    - Inmediata
    - Programada

WS-Resource Properties (I)

- C.R.U.D. de propiedades de un WS-Resource
- “Permisos R/W” según tipo, valor y estado
- 'Documento de propiedades'
  - WSDL (Esquema XML)
- Clientes piden/modifican documento XML con estado del WS-Resource
- 'Propiedad de Recurso'
  - Componente individual del estado de un WS-Resource
  - Tiene propio documento XML

WS-Resource Properties (II)

- Documento de Propiedades de un WS-Resource
  - 'XML Global Element Declaration' (GED)
    - Tipo de recurso asociado a un estado
    - Clientes pueden descargar y examinarla
      - Declaración del **portType** en Web Service
- Composición de Propiedades del WS-Resource
  - Una interfaz de Web Service no es productiva
  - 'Copiar y pegar' operaciones definidas en **portTypes**
    - ***<xs:ref>***

WS-Resource Properties (III)

- Accediendo a valores de Propiedades
  - Estado de un WSR puede leerse/modificarse
    - Documento de Propiedades
    - Empleando mensajes Web Service
      - Operaciones WSDL en cualquier **portType** que use *wsrp:ResourceProperties*
  - Operaciones 'get' y 'set'
    - Usando tipos vistos en WSDL (según sentido de comunicación)

WS-RenewableReferences

- Ampliación de WS-Addresing
  - Políticas
- Renovar referencia de 'endpoint' inválido
  - Útil en caso de apuntar a WS-Resource
- Información de WS-Resources es susceptible de cambio

WS-ServiceGroup

- Interfaz para colecciones heterogéneas de Web Services
- Grupos
  - Colección de miembros con mismo valor de ciertas variables (Propiedades del Recurso)

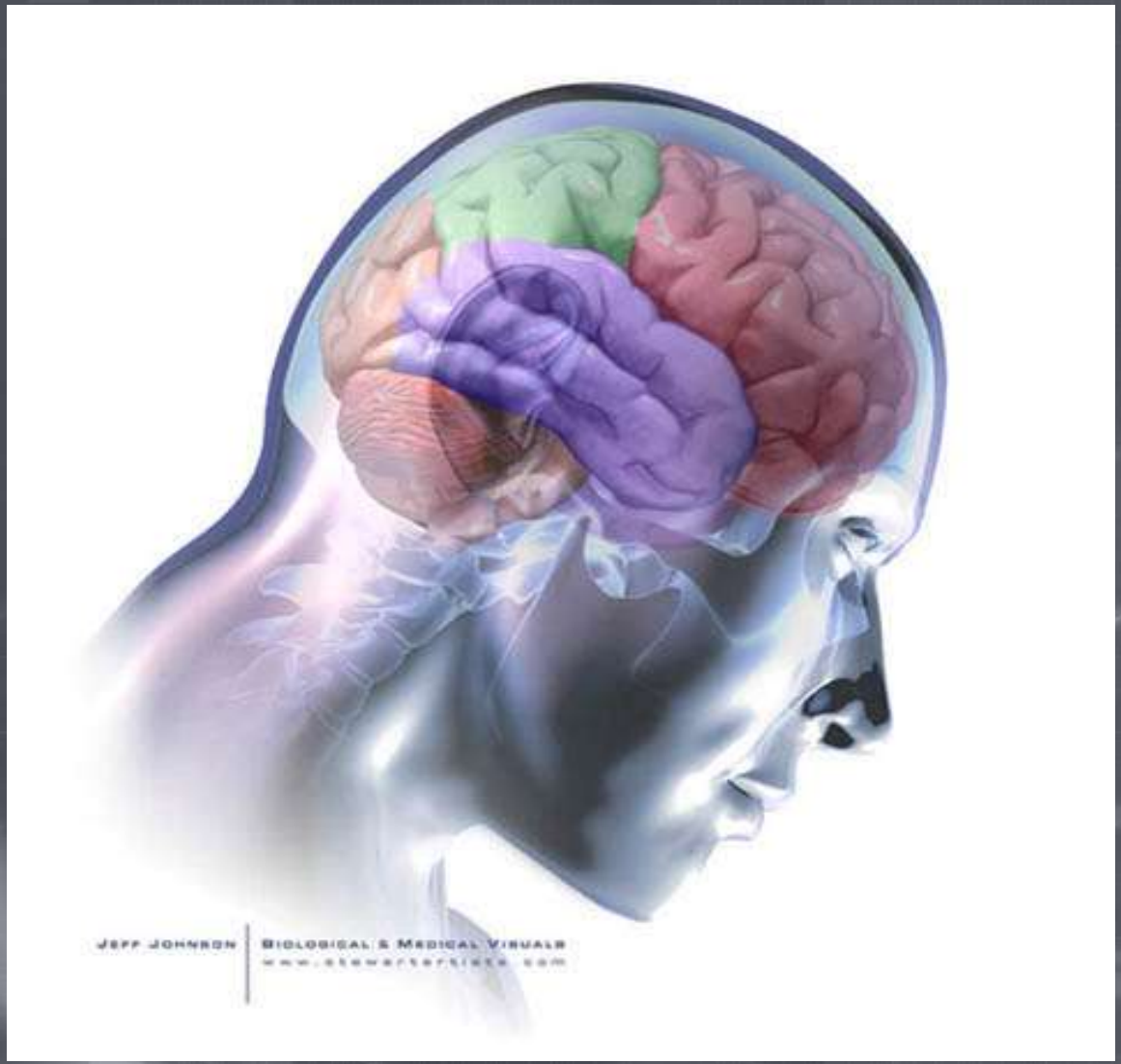
WS-BaseFaults

- Tipificación XML
  - Fallos en intercambio de mensajes Web Services

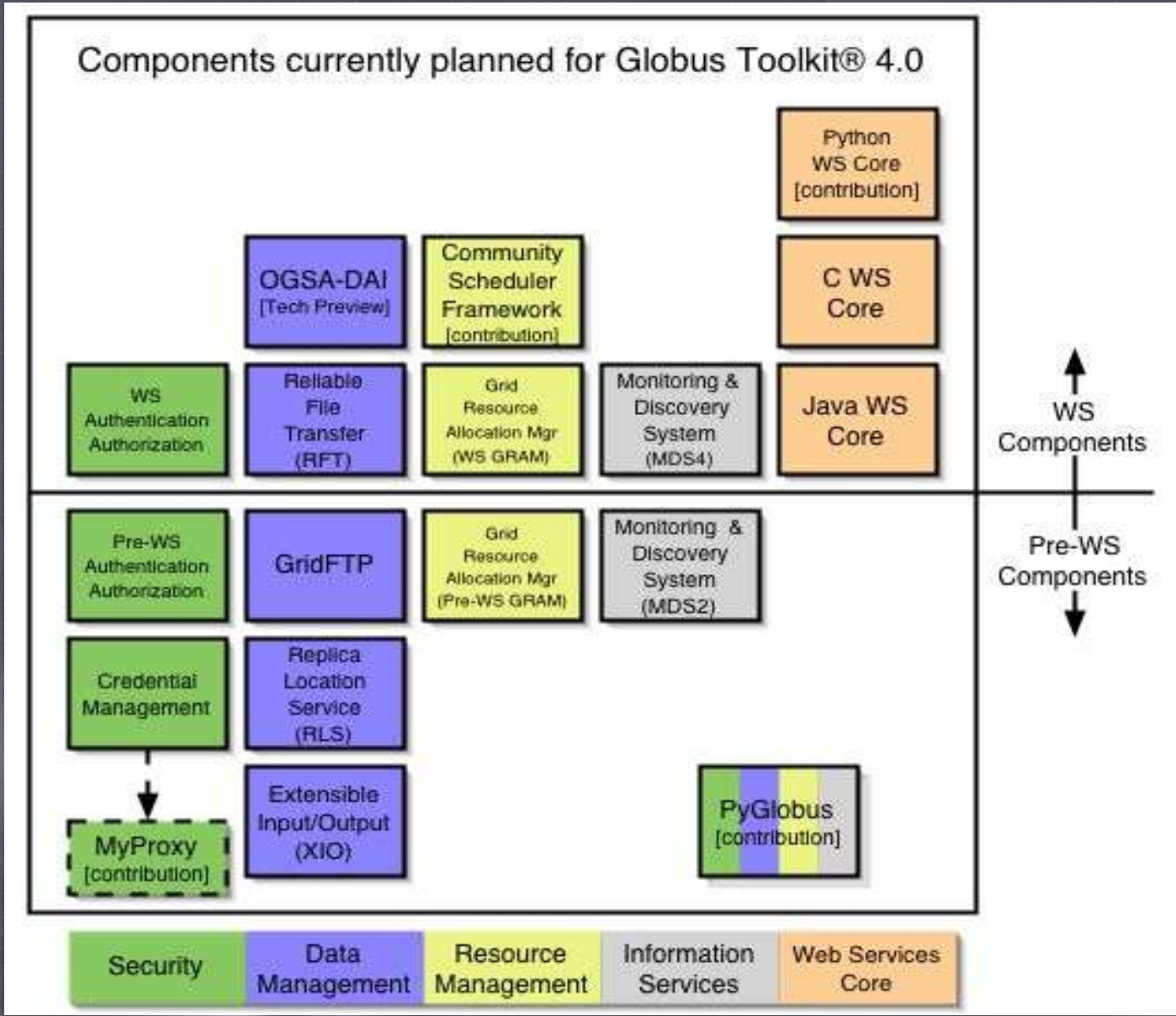
WS-Notification

- Familia de especificaciones separada
- Notificaciones de interacciones con Framework
  - Categorizadas por temas
- Subespecificaciones
  - **WS-BaseNotification:** Roles básicos, conceptos y patrones requeridos para registro
  - **WS-Topics:** Descripción XML de temas y metadatos
  - **WS-BrokeredNotification:** Interfaz de 'NotificationBroker' que administra suscripciones para otras entidades

Migración conceptual GT 2.4 a GT 4



Una visión panóptica



WS Authentication & Authorization

- Librerías de Autorización y Autenticación WS
- Funcionalidades:
  - Autenticación basada en X.509
  - Autenticación basada en GSI SecureConversation
  - Protección de mensajes
  - Framework de autorización con varios mecanismos

OGSA-DAI

- Objetivo: Proveer métodos uniformes
  - Descubrir, acceder e integrar recursos de datos heterogéneos (Bases de Datos)
- SGBs oficialmente admitidos:
  - MySQL 3.2.x / 4.0.x
  - DB2 8
  - Oracle 9i / 10g
  - Xindice 1.0
  - SQL Server 2000
  - PostgreSQL 7.x

Reliable File Transfer (RFT)

- Rol similar a planificador batch
- Envío de trabajos
  - n ficheros
  - Información varia
    - Reinicio (de ser necesario)
    - Notificaciones
- Información almacenada en Base de Datos
- Transferencia GridFTP a terceros

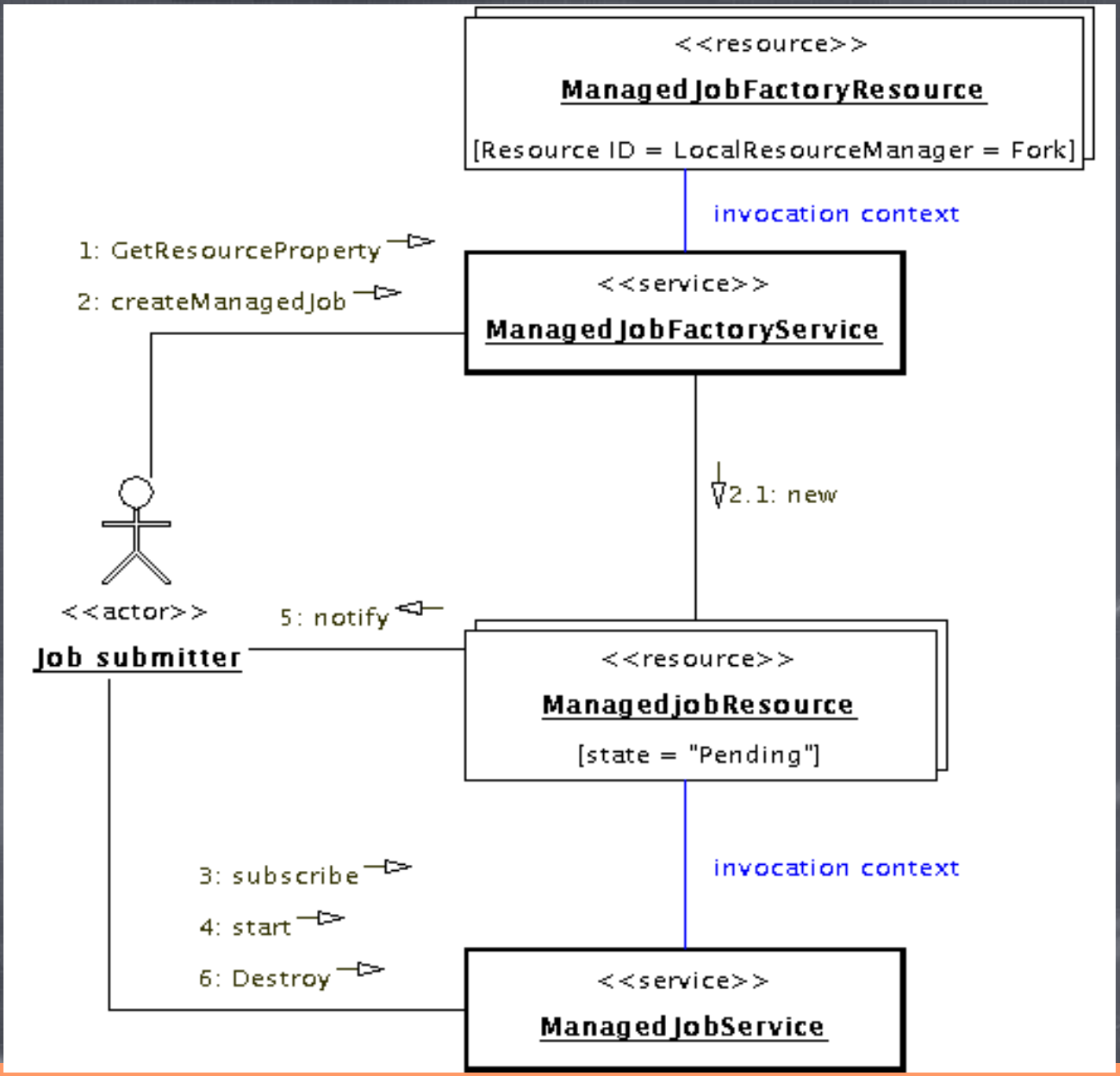
WS GRAM

- Único interfaz
  - Petición de recursos remotos
  - Uso de recursos remotos
- 2 implantaciones:
  - Pre-WS
  - WS

WS MDS4

- Monitor de nivel de Grid basado en WSRF
- Descubrimiento y manejo de recursos
  - Información de recursos básicos y colas
  - Interfaz para monitorización de clusters (Ej: Ganglia)
  - Aprovecha la gran información ofrecida por WSRF
- Servicio de indexado
  - Almacenamiento en Xindice (XML)
  - Interfaz web

Más a fondo: WS GRAM



WS GRAM: Desarrollo

1. El **Cliente** obtiene información sobre el **host** y el **sistema de planificación** (petición a **ManagedJobFactoryResource**)
2. El **Cliente** pide al **Factory** la creación de un nuevo **Job Resource** en función de la descripción dada
3. El **Cliente** se suscribe a las **Notificaciones** sobre el cambio en el estado del trabajo
4. El **Cliente** inicia el trabajo
5. El **Job Resource** notifica al/los listener(s) remoto(s) sobre los cambios de estado
6. El **Cliente** destruye el **Job Resource** cuando ya no lo necesita

WS GRAM: Ficheros relacionados (Cliente)

- *managed-job-globusrun*
  - Script SH que llama al verdadero programa en JAVA pasándole el CLASSPATH oportuno
  - *org.globus.exec.client.GlobusRun*
- *GlobusRun.java*
  - Fuente del verdadero programa (en directorio de la pre-instalación)
- Ejemplo:
  - *managed-job-globusrun -factory http://host:8080 -file prueba.xml*

WS GRAM: Ficheros relacionados (Servidor)

- *globus-start-container*
  - Script SH que llama al verdadero programa en JAVA pasándole el CLASSPATH oportuno
  - *org.globus.wsrf.container.ServiceContainer*
- *ServiceContainer.java*
  - Fuente del verdadero programa (en directorio de la pre-instalación)
- Cualquier contenedor podría valer
  - Este contenedor es de prueba
  - Globus invita a usar Apache Tomcat (uso de AXIS)

WS GRAM: SandBox (I)

- Separación del Cliente WS GRAM
- Permite estudiar el funcionamiento
- Ficheros:
  - *Globusrun.java*
    - Fuente Java del Cliente
    - Aviso en cada evento importante
  - *gramclientcompiler*
    - Script SH para compilar correctamente el Cliente
  - *managed-job-globusrun*
    - Script SH modificado para lanzar correctamente el Cliente

WS GRAM: API WS GRAM (cliente)

```
import org.globus.exec.client.GramJob; // Añadido para la versión de estudio
import org.globus.exec.client.GramJobListener; // Añadido para la versión de estudio
import org.oasis.wsrf.faults.BaseFaultType;
...
public class Globusrun implements GramJobListener {

public void main(){
URL factoryURL = ManagedJobFactoryHelper.getServiceURL(contactString).getURL();
FactoryEndpoint = ManagedJobFactoryHelper.getFactoryEndpointReference(
                    factoryURL,
                    factoryType);
job = new GramJob(rslFile); // fichero xml con el job ver test.xml del sandbox
//job = new GramJob(RSLHelper.makeSimpleJob(simpleJobCommandLine)); a la Pre-WS
```

WS GRAM: API WS GRAM (cliente)

```
job.setTimeOut(timeout);//¿?

job.setAuthorization(authorization); // Dos tipos HostAuthorization.getInstance()
SelfAuthorization.getInstance();

job.setMessageProtectionType(xmlSecurity); // Dos Tipos  Constants.SIGNATURE
Constants.ENCRYPTION

job.setDuration(duration); //¿?

job.setTerminationTime(terminationDate); //¿?

job.submit(factoryEndpoint, batch, limitedDelegation);

job.addListener(this);

job.start();

synchronized (this) {
        while (!this.done) {
            try { wait(); } catch (InterruptedException ie) {logger.error("interrupted  finish",ie);}
}
```

WS GRAM: API WS GRAM (cliente)

```
public void stateChanged(GramJob job) {
        StateEnumeration jobState = job.getState();
        System.out.println("========== State Notification ==========");
        System.out.println(JOB_STATE_PREFIX + jobState.getValue());
        System.out.println("=======================================");
        synchronized (this) {
            if (   jobState.equals(StateEnumeration.Done)
                || jobState.equals(StateEnumeration.Failed)) {
                this.done = true;
                notifyAll();
            }
}
```

WS GRAM: API WS GRAM (cliente)

Otras cosas que puede hacer el servicio WS GRAM (todo está en el sandbox)

- **Matar un trabajo:** GramJob job = getExistingJob(jobHandle); **destroyJob**(job);
- **Obtener la lista de trabajos de usuario:** jobHandles = GramJob.**getStartedJobs (factoryEndpoint);**
- **Obtener el estado de un trabajo:** GramJob **job = getExistingJob(jobHandle);** job.**getState**();

Además la clase Job tiene más atributos y métodos de los comentados hoy:

http://www-unix.globus.org/api/javadoc-3.9.2-gram/client/org/globus/exec/client/GramJob.html

Sin embargo, la mayoría de los métodos no están comentados

WS GRAM: SandBox (II)

[jlvazquez@columba Sandbox]$ ./gramclientcompiler

*** COMPILADOR DEL CLIENTE DE WS GRAM ***

*** Calculando Classpath ***

*** Compilando Globusrun.java ***

[jlvazquez@columba Sandbox]$ ./managed-job-globusrun -factory http://columba.dacya.ucm.es:8080 -file test.xml

>>> FactoryEndPoint (Dirección y Reference Properties):

Address: http://columba.dacya.ucm.es:8080/wsrf/services/ManagedJobFactoryService

Reference property[0]:

<ns1:ResourceID xmlns:ns1="http://www.globus.org/namespaces/2004/06/gram"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:type="xsd:string">Fork</ns1:ResourceID>

WS GRAM: SandBox (III)

Added shutdown hook

Starting job

========== State Notification ==========

Job State: Unsubmitted

==========

Job has started.

WAITING FOR JOB TO FINISH

>>> Sincronizando... ¿Ha terminado ya? ¿Ha terminado ya?

========== State Notification ==========

Job State: Active

==========

========== State Notification ==========

Job State: Done

==========

WS GRAM: SandBox (IV)

========== State Notification ==========
Job State: Active
==========

>>> El trabajo ha terminado... Notficando
destroying job service from shutdown hook
destroyJob() called in run()
DESTROYING SERVICE
SERVICE DESTROYED
[jlvazquez@columba Sandbox]$

MUCHAS GRACIAS POR VUESTRA ATENCIÓN

http://asds.dacya.ucm.es/jlvazquez/