



Sistemas Grid Basados en GT3



Módulo 1 Introducción a OGSA, OGSI, y GT3

Borja Sotomayor
3 de marzo de 2004



Introducción

- ▶ En este módulo veremos una introducción teórica a OGSA (*Open Grid Services Architecture*), OGSI (*Open Grid Services Infrastructure*) y el Globus Toolkit 3.
- ▶ También comentaremos la reciente publicación de WSRF (*Web Services Resource Framework*), la nueva generación de estándares en las que se apoyará GT4.





Índice

- ▶ Orígenes de OGSA
- ▶ Introducción a los Web Services y SOA
- ▶ Grid Services (OGSI)
- ▶ Globus Toolkit 3
- ▶ Relación entre OGSA, OGSI, y GT3
- ▶ WSRF/GT4

Sistemas Grid Basados en GT3



Índice

- ▶ Orígenes de OGSA
- ▶ Introducción a los Web Services y SOA
- ▶ Grid Services (OGSI)
- ▶ Globus Toolkit 3
- ▶ Relación entre OGSA, OGSI, y GT3
- ▶ WSRF/GT4

Sistemas Grid Basados en GT3



OGSA

- ▶ El *Open Grid Services Architecture* (OGSA) fue presentado por I.Foster, C.Kesselman, J.Nick, y S.Tuecke en "Physiology of the Grid".
 - ▶ Arquitectura estándar para afrontar los desafíos de las aplicaciones Grid.
- ▶ A partir de OGSA han surgido muchos otros términos como OGSi y 'Grid Services'. Además, es habitual equiparar GT3 con OGSA. ¿Qué significan exactamente todos estos términos y cómo están relacionados entre sí?

Sistemas Grid Basados en GT3



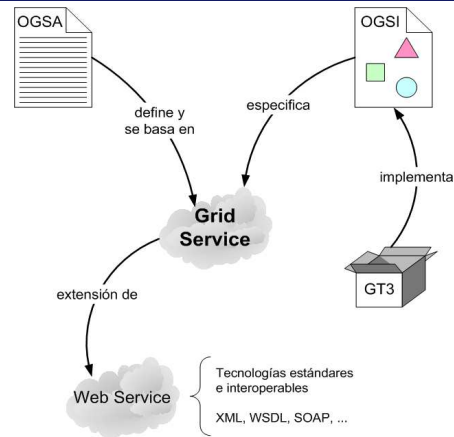
OGSA, OGSi, GT3 (I)

- ▶ ¿OGSA == GT3?
- ▶ ¿OGSA == Grid Services?
- ▶ ¿OGSi?
- ▶ ¿Qué tienen que ver los Grid Services con los Web Services?
- ▶ ¿Por qué ese empeño en utilizar Web Services? ¿Por qué no CORBA, RMI, ...?

Sistemas Grid Basados en GT3



OGSA, OGSi, GT3 (II)



Sistemas Grid Basados en GT3



OGSA, OGSi, GT3 (III)

- ▶ OGSA *define* el concepto de *Grid Service* como *extensión* de los Web Services.
 - ▶ Los Web Services tienen ciertas características que los hacen idóneos como *base* para la arquitectura (OGSA).
 - ▶ Sin embargo, tienen ciertas carencias, por lo que es necesario extenderlos.
 - ▶ Veremos esto en más profundidad más adelante.
- ▶ Los Grid Services son la base de OGSA.
 - ▶ No son lo *único* que tiene OGSA.
 - ▶ Más adelante veremos que OGSA es una arquitectura que consiste de muchos servicios estandarizados.

Sistemas Grid Basados en GT3



OGSA, OGSÍ, GT3 (IV)

- ▶ Los detalles exactos y técnicos de los Grid Services están *especificados* en la *Open Grid Services Infrastructure* (OGSI).
- ▶ El *Globus Toolkit 3* (GT3) es una *implementación* de OGSI.
 - ▶ Sin embargo, OGSI no es lo *único* que tiene GT3.
 - ▶ Más adelante veremos que GT3 incluye muchas aportaciones propias.

Sistemas Grid Basados en GT3



Índice

- ▶ Orígenes de OGSA
- ▶ Introducción a los Web Services y SOA
- ▶ Grid Services (OGSI)
- ▶ Globus Toolkit 3
- ▶ Relación entre OGSA, OGSI, y GT3
- ▶ WSRF/GT4

Sistemas Grid Basados en GT3



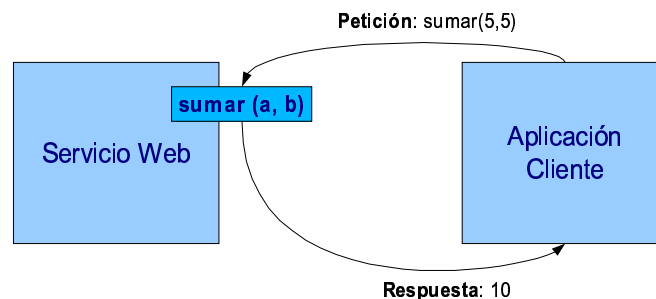
Web Services (I)

- ▶ ¿Qué son los Web Services?
 - ▶ Otra tecnología distribuida *más* (como CORBA, RMI, EJBs, etc.)
 - ▶ Otro middleware *más* con el que implementar aplicaciones cliente/servidor
 - ▶ En realidad, tienen poco o nada que ver con “La Web” (entendida como 'colección distribuida de documentos de hipertexto'). Incluso se debatió cambiar el nombre, eliminando la palabra 'Web'.



Web Services (II)

- ▶ Entonces... ¿no es más que una invocación remota de procedimientos? ¿Qué añade?





Web Services (III)

- ▶ Una de las más importantes características de los Web Services es que tienen una *separación entre el interfaz (“lo que hace”) y la implementación (“cómo lo hace”)*, y que dicho interfaz está escrito en un lenguaje neutro (independiente de lenguaje de programación y plataforma).



Web Services (IV)

Interfaz

Un servicio web 'expone' las operaciones que es capaz de realizar mediante un interfaz. El interfaz está escrito en un lenguaje neutro (independiente de lenguaje de programación y de plataforma).

Implementación

La implementación y el interfaz se mantienen separados. La implementación se realiza en un lenguaje de programación concreto.

Entorno de Ejecución

La implementación y el interfaz están alojados en un entorno de ejecución, que se encarga de procesar las peticiones entrantes, generar las respuestas, etc.



Web Services (V)

- ▶ Esto implica lo siguiente:
 - ▶ El lenguaje de programación del cliente y el servicio web no tienen que coincidir. Se puede generar el código de invocación para cualquier lenguaje.
 - ▶ Ídem para la plataforma.
 - ▶ P.ej. Un servicio web programado en C++ bajo Linux podría comunicarse con un servicio web programado en Java bajo Windows.



Web Services (VI)

- ▶ Todo esto nos permite realizar SOAs (*Service-Oriented Architectures*)
 - ▶ Arquitectura en la que una aplicación está compuesta de varios servicios independientes (o *débilmente acoplados*) que cooperan para resolver una tarea común.
 - ▶ “débilmente acoplado”: Un cambio en la implementación de un servicio no afecta a los demás servicios.
 - ▶ Al utilizar un lenguaje *común* de descripción (del interfaz) es sencillo indexar y descubrir servicios.



Web Services (VII)

- ▶ Las SOAs y, en particular, los Web Services son ideales para aplicaciones Grid porque:
 - ▶ Funcionan bien en entornos heterogéneos donde existe una gran diversidad de plataformas y lenguajes de programación.
 - ▶ Se puede acceder dinámicamente a servicios, aunque no fuesen conocidos en tiempo de implementación (el código de invocación se puede generar dinámicamente en tiempo de ejecución).

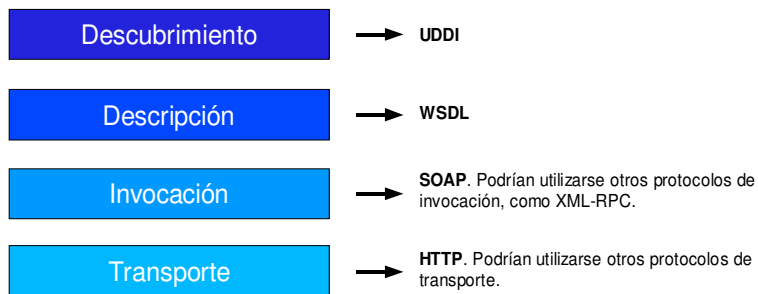


Web Services (VIII)

- ▶ Desventajas
 - ▶ Rendimiento. Utilizar lenguajes XML como base no es tan eficiente como utilizar un lenguaje binario puro.
 - ▶ Falta de versatilidad. Los web services proporcionan un modelo de invocación bastante sencillo, al menos en comparación con tecnologías como CORBA. Esta falta de versatilidad es la que pretenden suplir OGSF y OGSA.



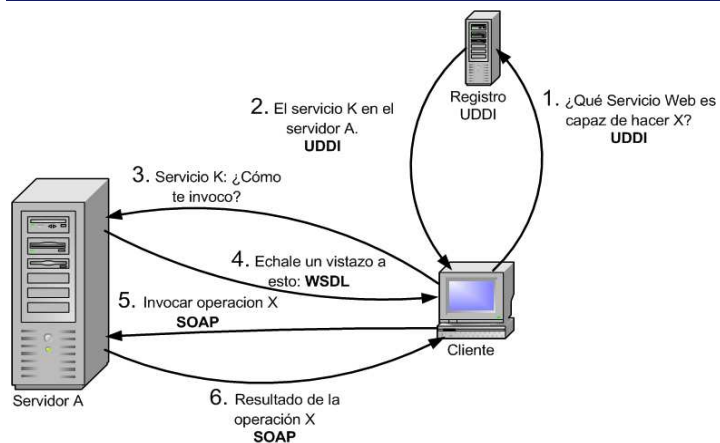
Arquitectura de los Web Services



Sistemas Grid Basados en GT3



Una invocación básica (I)



Sistemas Grid Basados en GT3



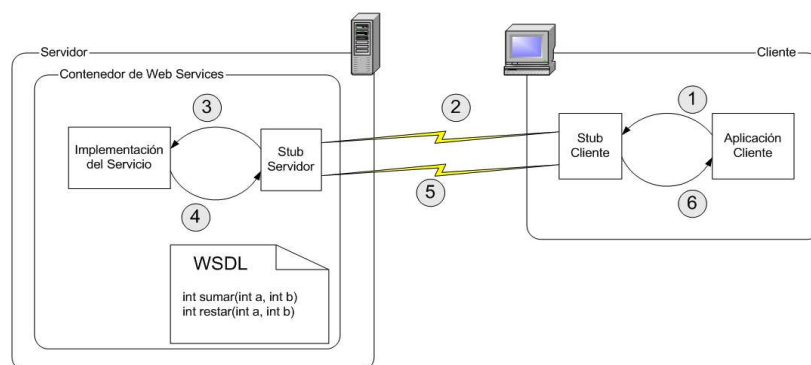
Una invocación básica (II)

- ▶ La figura muestra una invocación totalmente dinámica (todo el código de invocación se genera sobre la marcha).
- ▶ No es habitual consultar un registro *cada vez* que se va a invocar un servicio web.
- ▶ Lo normal es consultar el registro una vez para localizar un servicio concreto y generar unos *stubs* que se reutilizarán cada vez que se invoque el servicio:
 - ▶ stub: Código que encapsula todo el código de invocación para un interfaz concreto.

Sistemas Grid Basados en GT3



Una invocación básica (III)



Sistemas Grid Basados en GT3



Una invocación básica (IV)

- ▶ Todavía nos puede hacer falta utilizar un registro en este escenario. Por ejemplo, podríamos tener 100 servicios con el mismo interfaz (lo que significa que podemos utilizar el mismo *stub*), y lo que queremos que nos diga el registro es qué servicio está en una máquina con ciertas características (menos carga de CPU, más memoria, etc.)



Índice

- ▶ Orígenes de OGSA
- ▶ Introducción a los Web Services y SOA
- ▶ **Grid Services (OGSI)**
- ▶ Globus Toolkit 3
- ▶ Relación entre OGSA, OGSI, y GT3
- ▶ WSRF/GT4



Grid Services (I)

- ▶ A pesar de que los Web Services sean la mejor opción para OGSA, tienen importantes limitaciones.
 - ▶ Stateless
 - ▶ No-transientes ('Persistentes')
 - ▶ No tienen 'servicios de apoyo' (notificaciones, gestión del ciclo de vida, etc.)
- ▶ La especificación OGSF *amplía* los Web Services para superar estas limitaciones.
 - ▶ Grid Service = Web Service *ampliado*
 - ▶ Los Grid Services son compatibles con los WS

Sistemas Grid Basados en GT3



Mejoras de los Grid Services

- ▶ Mejoras de los Grid Services
 - ▶ Soporte para herencia de interfaces
 - ▶ Servicios con estado y potencialmente transientes
 - ▶ Gestión del ciclo de vida
 - ▶ Service Data ("Datos del servicio")
 - ▶ Notificaciones
 - ▶ Agrupación de servicios

Sistemas Grid Basados en GT3



Mejoras de los Grid Services

- ▶ Mejoras de los Grid Services
 - ▶ Soporte para extensión de interfaces
 - ▶ Servicios con estado y potencialmente transientes
 - ▶ Gestión del ciclo de vida
 - ▶ Service Data (“Datos del servicio”)
 - ▶ Notificaciones
 - ▶ Agrupación de servicios

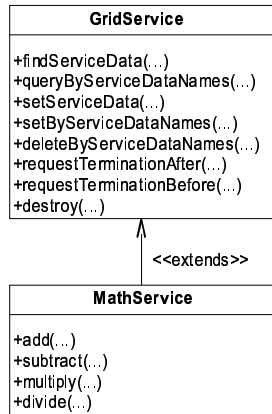


Extensión de Interfaces (I)

- ▶ Los Grid Services permiten definir un interfaz en función de otro.
 - ▶ Los interfaces en Grid Services (y Web Services) reciben el nombre de *portType*
- ▶ OGSi define un *portType* estándar llamado *GridService* que todos los servicios deben implementar.
- ▶ Siempre hay que extender de este interfaz.



Extensión de Interfaces (II)

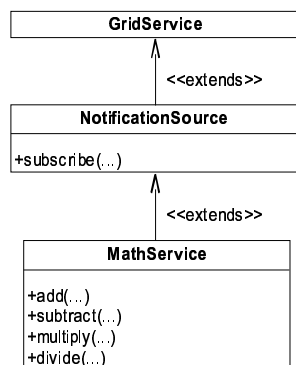


- ▶ Extendiendo del interfaz GridService evitamos tener que redeclarar los métodos estándar en cada uno de nuestros portTypes.

Sistemas Grid Basados en GT3



Extensión de Interfaces (III)



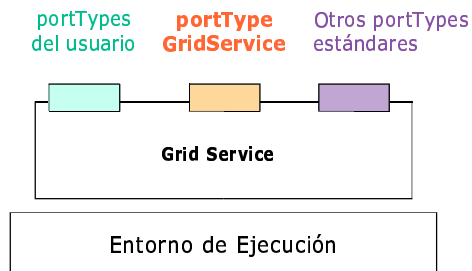
- ▶ OGSi define una gran cantidad de portTypes estándar de los que podemos extender.

Sistemas Grid Basados en GT3



Extensión de Interfaces (IV)

- ▶ En general, nuestro servicio puede tener 3 tipos de portTypes:



Extensión de Interfaces (V)

- ▶ La extensión de portTypes no está soportado en WSDL 1.1 (la versión actual), por lo que OGSF utiliza una versión ampliada de WSDL llamada GWSDL.
 - ▶ El GWSDL se convierte a WSDL 1.1 para mantener la interoperabilidad.
 - ▶ WSDL 1.2/2.0 (Working Draft) si soporta extensión de portTypes. En cuanto pase a ser una recomendación de la W3C, se abandonará GWSDL en favor de WSDL.



Mejoras de los Grid Services

- ▶ Mejoras de los Grid Services
 - ▶ Soporte para herencia de interfaces
 - ▶ Servicios con estado y potencialmente transientes
 - ▶ Gestión del ciclo de vida
 - ▶ Service Data (“Datos del servicio”)
 - ▶ Notificaciones
 - ▶ Agrupación de servicios



Servicios con Estado y Transientes (I)

- ▶ Los servicios web no mantienen el estado de una llamada a otra.
 - ▶ Son *stateless*
- ▶ En OGSI, los grid services tienen que conservar su estado de una llamada a otras.
 - ▶ Son *stateful*



Servicios con Estado y Transientes (II)

- ▶ Los servicios web son siempre *persistentes*
 - ▶ Se activan en el momento en el que se inicia el contenedor de web services y no son destruidos hasta que se para el contenedor.
 - ▶ ¡Ojo! No es 'persistencia' en el sentido de almacenar los datos del servicio en un fichero o BD.
- ▶ Los servicios grid son *potencialmente transientes*.
 - ▶ Un grid service puede ser activado/destruido cuando queramos. Su ciclo de vida no está ligado al contenedor de grid services.

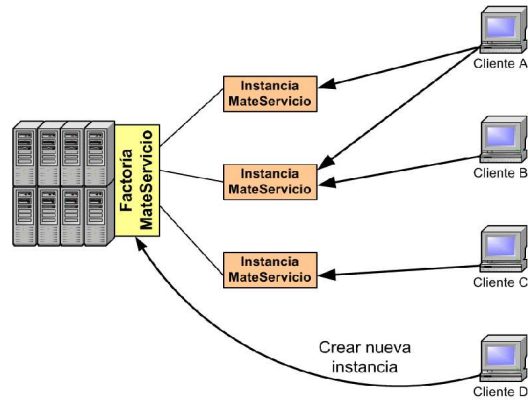


Servicios con Estado y Transientes (III)

- ▶ Esto se consigue con un patrón Factoría/Instancia.
 - ▶ Podemos configurar una *factoría de servicios* que crea *instancias* de un servicio.
 - ▶ Las instancias se crean a través de la factoría.
 - ▶ Cuando queremos invocar una operación del servicio, accedemos a la instancia no a la factoría.
 - ▶ Podemos tener una instancia por cliente, varias instancias por cliente, varios clientes por instancia, ...
 - ▶ La destrucción de la instancia puede correr a cargo del cliente o de la factoría.



Servicios con Estado y Transientes (IV)



Sistemas Grid Basados en GT3



Mejoras de los Grid Services

- ▶ Mejoras de los Grid Services
 - ▶ Soporte para herencia de interfaces
 - ▶ Servicios con estado y potencialmente transientes
 - ▶ Gestión del ciclo de vida
 - ▶ Service Data (“Datos del servicio”)
 - ▶ Notificaciones
 - ▶ Agrupación de servicios

Sistemas Grid Basados en GT3



Gestión del Ciclo de Vida

- ▶ En OGSÍ podemos tener servicios con ciclos de vida no triviales (no ligados al ciclo de vida del contenedor de servicios).
- ▶ OGSÍ permite gestionar el ciclo de vida (principalmente orientado a los servicios transientes)
 - ▶ Tiempo antes del cual debe ser destruida la instancia
 - ▶ Tiempo después del cual se espera que sea destruida la instancia.
- ▶ GT3 añade más funcionalidad:
 - ▶ Código a ejecutar antes de la creación, después de la creación, antes de la destrucción, ...

Sistemas Grid Basados en GT3



Mejoras de los Grid Services

- ▶ Mejoras de los Grid Services
 - ▶ Soporte para herencia de interfaces
 - ▶ Servicios con estado y potencialmente transientes
 - ▶ Gestión del ciclo de vida
 - ▶ Service Data (“Datos del servicio”)
 - ▶ Notificaciones
 - ▶ Agrupación de servicios

Sistemas Grid Basados en GT3



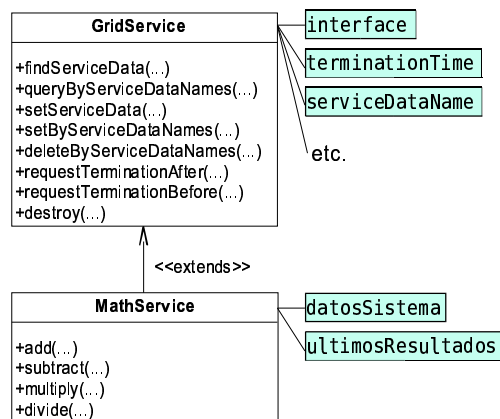
Service Data (I)

- ▶ Una de las aportaciones más interesantes de OGISI a los web services es el *service data*.
- ▶ En OGISI es posible asociar datos estructurados a un servicio, generalmente de dos tipos:
 - ▶ Información de estado: resultados de operaciones, información de ejecución, resultados intermedios, etc.
 - ▶ Metadatos: Datos sobre el propio servicio (configuración, coste, carga actual, etc.)
- ▶ El *service data* facilita el descubrimiento, inspección, monitorización y gestión de los grid services.

Sistemas Grid Basados en GT3



Service Data (II)

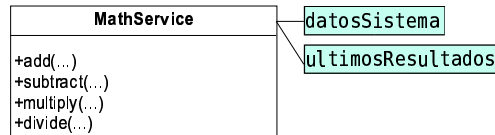


- ▶ El service data del servicio se define en el portType, en el fichero GWSDL.
- ▶ Al extender de un portType existente, también se hereda su service data.
- ▶ WSDL a secas tampoco soporta service data

Sistemas Grid Basados en GT3



Service Data (III)



- ▶ Un servicio puede definir 0..N *Service Data Elements* (SDEs).
 - ▶ MathService tiene dos SDEs: datosSistema y ultimosResultados
- ▶ Un SDE puede tener 0..N *valores*
 - ▶ Cada valor debe tener el *mismo* tipo de datos
 - ▶ La cardinalidad la especificamos nosotros (en el GWSDL). Por ejemplo, podríamos especificar que ultimosResultados puede tener 0..10 valores.

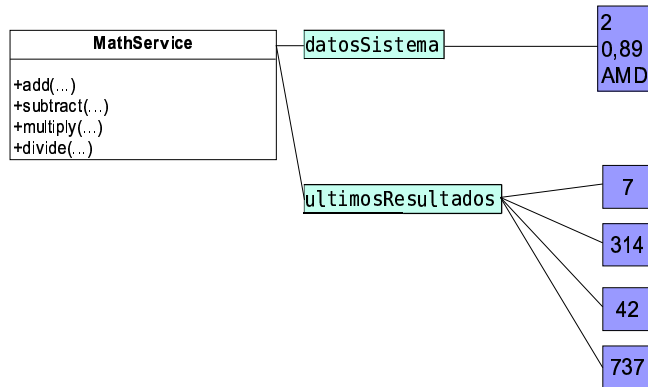


Service Data (IV)

- ▶ El tipo de datos del valor se especifica en XML Schema.
 - ▶ ultimosResultados: Podría ser simplemente xsd:int (un número entero)
 - ▶ datosSistema: Podría ser un tipo de datos complejo que incluye:
 - ▶ numeroCPUs: xsd:int
 - ▶ cargaActual: xsd:float
 - ▶ marcaCPU: xsd:string



Service Data (V)



Mejoras de los Grid Services

- ▶ Mejoras de los Grid Services
 - ▶ Soporte para herencia de interfaces
 - ▶ Servicios con estado y potencialmente transientes
 - ▶ Gestión del ciclo de vida
 - ▶ Service Data (“Datos del servicio”)
 - ▶ **Notificaciones**
 - ▶ Agrupación de servicios



Notificaciones (I)

- ▶ OGSi integra un patrón de *notificaciones*.
 - ▶ También conocido como el patrón “Observer-Observable” o “Model-View-Controller”.
 - ▶ Dos roles: El 'observador' y el 'observable'.
 - ▶ El observador quiere saber cuando se produce un cambio en el observable. Una manera es preguntarle cada cierto tiempo (*polling*). Una solución más óptima es el patrón de notificaciones. El observador se *suscribe* en el observable. El observable le envía una *notificación* al observador cuando se produce el cambio.

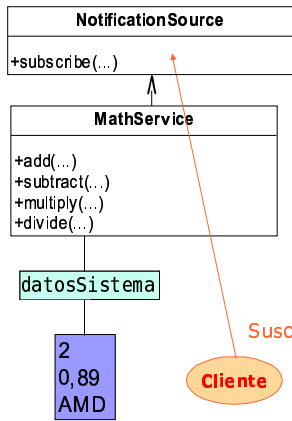


Notificaciones (II)

- ▶ Terminología OGSi:
 - ▶ Observable: *Notification source* (fuente de notificaciones)
 - ▶ Observador: *Notification sink* (sumidero de notificaciones)
- ▶ En OGSi un cliente puede suscribirse a un SDE concreto de un servicio.
- ▶ La notificación no se envía *siempre* que cambia el valor del SDE. Es responsabilidad del programador del servicio decidir cuándo se enviará la notificación.
 - ▶ Por ejemplo, en MathService no sería eficiente enviar una notificación *siempre* que cambia la carga del sistema (se enviarían notificaciones constantemente).



Notificaciones (III)

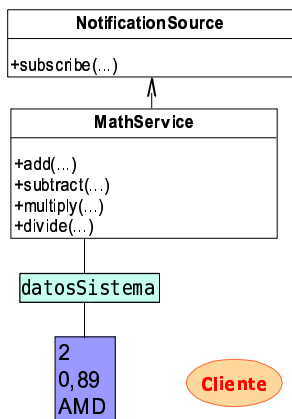


- ▶ Por ejemplo, un cliente quiere saber cuando la carga del sistema es menor que 0'5.
- ▶ Para ello, se suscribe al SDE datosSistema

Sistemas Grid Basados en GT3



Notificaciones (IV)

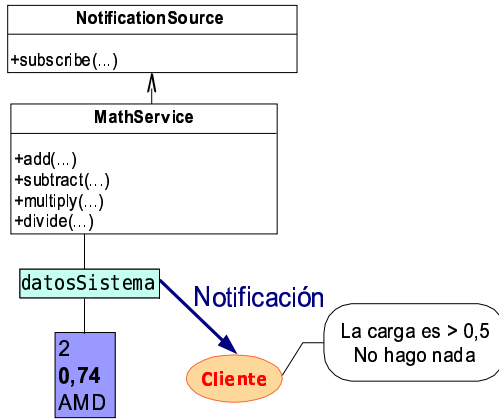


- ▶ La implementación interna de MathService actualiza datosSistema si se produce un cambio de 0,1 en la carga del sistema.
- ▶ En cada actualización se le enviará una notificación al cliente.

Sistemas Grid Basados en GT3



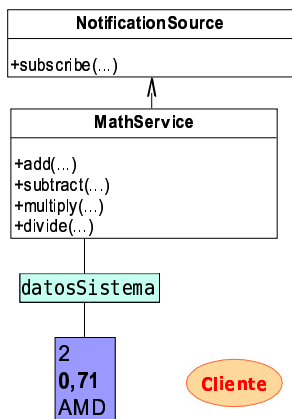
Notificaciones (V)



Sistemas Grid Basados en GT3



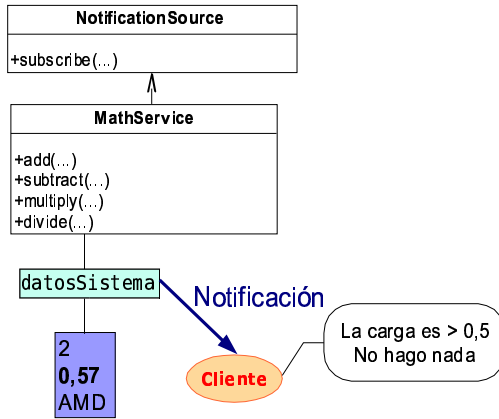
Notificaciones (VI)



Sistemas Grid Basados en GT3



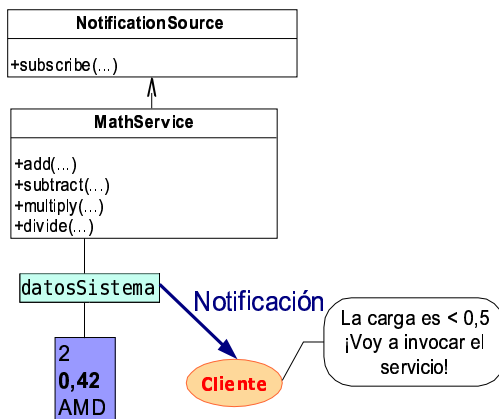
Notificaciones (VII)



Sistemas Grid Basados en GT3



Notificaciones (VIII)



Sistemas Grid Basados en GT3



Notificaciones (IX)

- ▶ Las notificaciones en OGSi son del tipo *push*
 - ▶ Notificaciones *pull*: Los datos que son objeto de notificación *no* son enviados con la notificación.
 - ▶ Notificaciones *push*: Los datos que son objeto de notificación (en el caso de OGSi, el SDE) son enviados junto con la notificación.
 - ▶ Ventaja: No hay que hacer un 'viaje' adicional al servicio para obtener el dato.
 - ▶ Desventaja: Puedo recibir notificaciones con datos que no voy a utilizar.

Sistemas Grid Basados en GT3



Mejoras de los Grid Services

- ▶ Mejoras de los Grid Services
 - ▶ Soporte para herencia de interfaces
 - ▶ Servicios con estado y potencialmente transientes
 - ▶ Gestión del ciclo de vida
 - ▶ Service Data ("Datos del servicio")
 - ▶ Notificaciones
 - ▶ Agrupación de servicios

Sistemas Grid Basados en GT3



Agrupación de Servicios (I)

- ▶ OGSi proporciona una serie de interfaces para crear *grupos de servicios*.
 - ▶ Este grupo de servicios (ServiceGroup) será a su vez un grid service.
 - ▶ Podemos añadir y eliminar servicios del grupo.
 - ▶ Esto nos permite agrupar servicios y gestionarlos utilizando un punto de entrada único.



Agrupación de Servicios (II)

- ▶ La funcionalidad de los ServiceGroups de OGSi es muy sencilla, pero sirve como base para implementar registros/directorios de servicios.
 - ▶ El IndexService de GT3 se basa en los ServiceGroups y añade mucha funcionalidad extra.

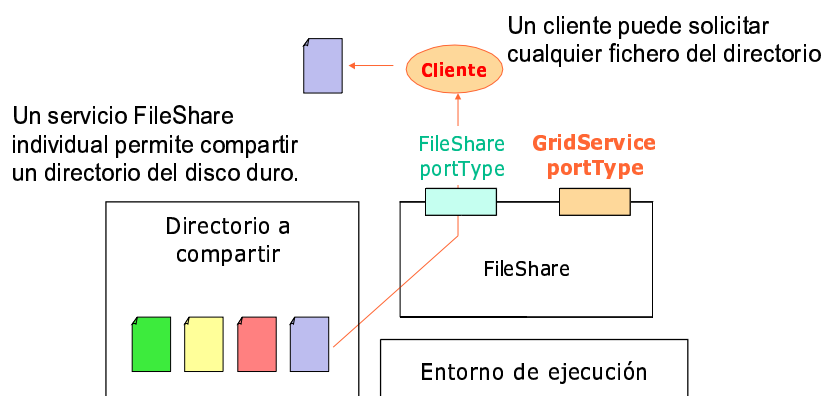


Un ejemplo (I)

- ▶ Aplicación para compartir ficheros de distintas máquinas.
- ▶ En cada máquina se pueden compartir varios directorios.
- ▶ Existe un registro central a través del cual se puede saber qué ficheros están disponibles (y en qué máquina concreta)
- ▶ *FileShare*

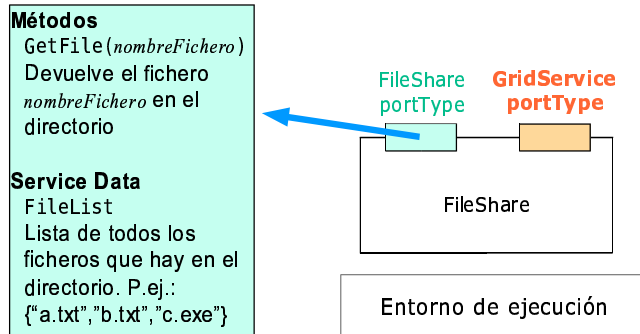


Un ejemplo (II)





Un ejemplo (III)

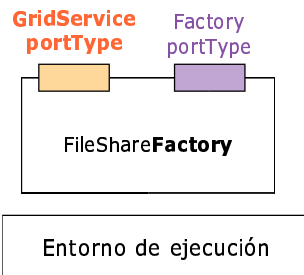


Un ejemplo (IV)

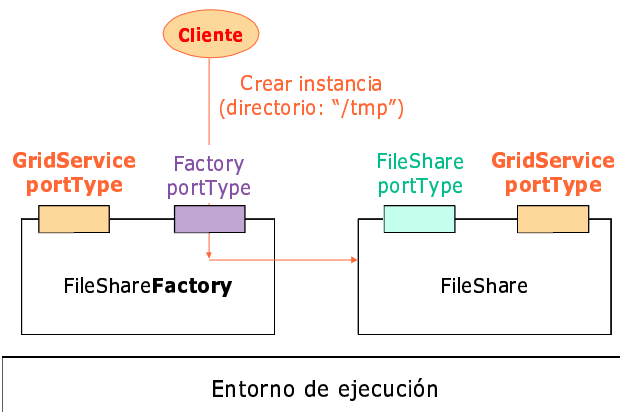
- ▶ Esto está bien para compartir un único directorio de una máquina pero... ¿y si queremos compartir varios directorios?
- ▶ Con servicios web 'a secas' tendríamos que tener configurados (estáticamente) tantos servicios como directorios.
- ▶ Con grid services:
 - ▶ 1 Factoría de servicios FileShare
 - ▶ Creamos 1 instancia por cada directorio que queremos compartir. La creación y destrucción de la instancia pueden ser en cualquier momento.



Un ejemplo (V)

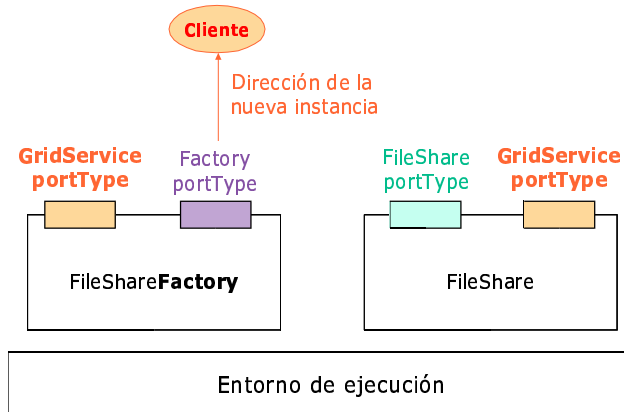


Un ejemplo (VI)

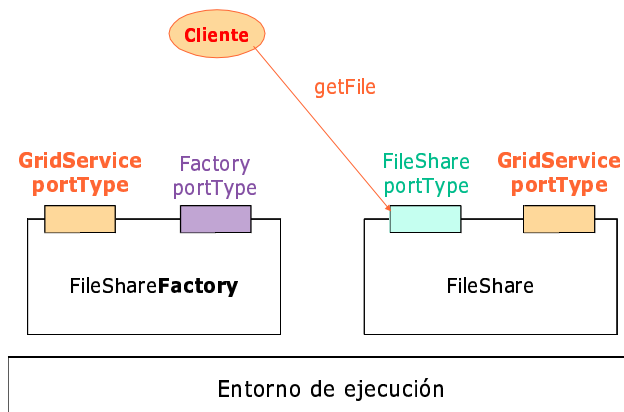




Un ejemplo (VII)

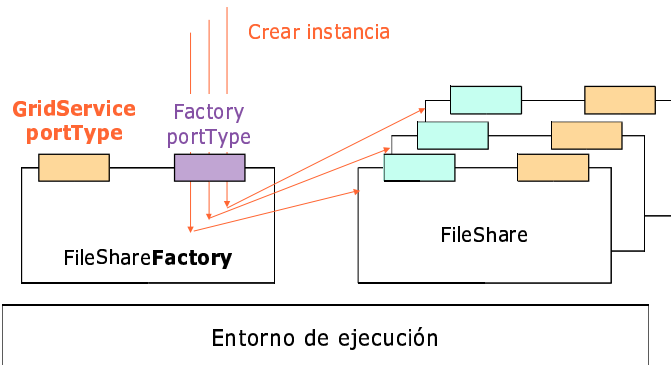


Un ejemplo (VIII)

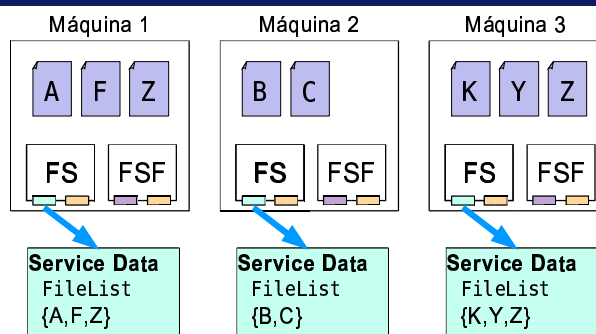




Un ejemplo (IX)



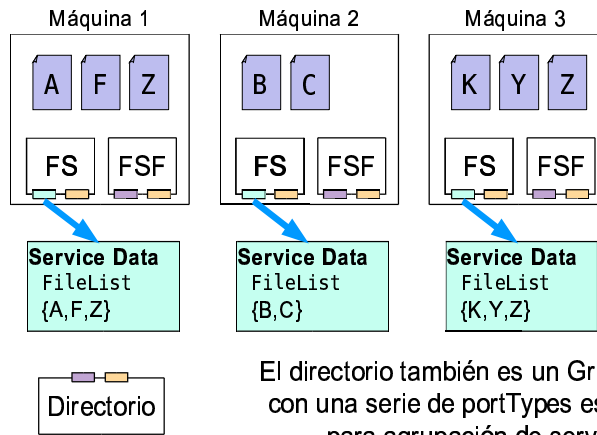
Un ejemplo (X)



¿Cómo localizamos un fichero concreto?



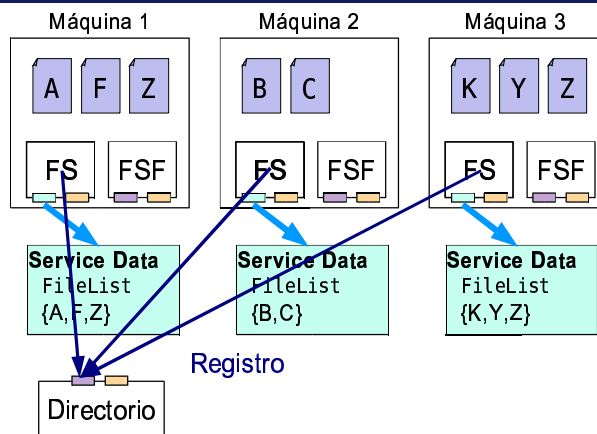
Un ejemplo (XI)



El directorio también es un Grid Service, con una serie de portTypes estándares para agrupación de servicios

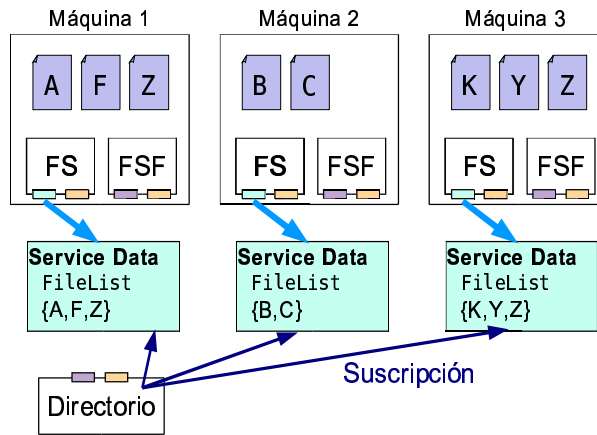


Un ejemplo (XII)





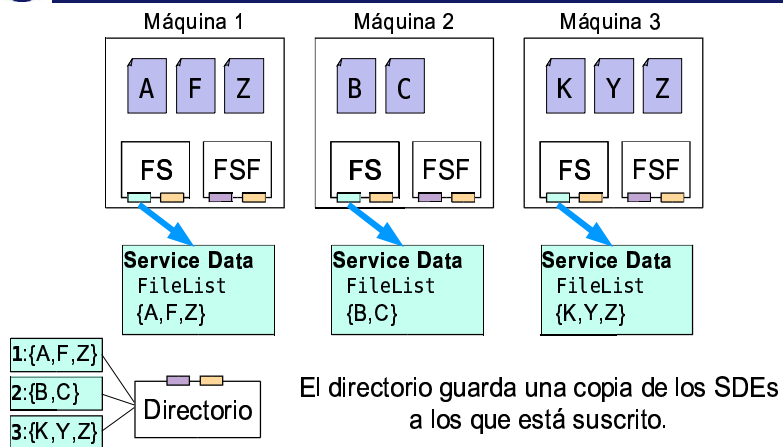
Un ejemplo (XIII)



Sistemas Grid Basados en GT3



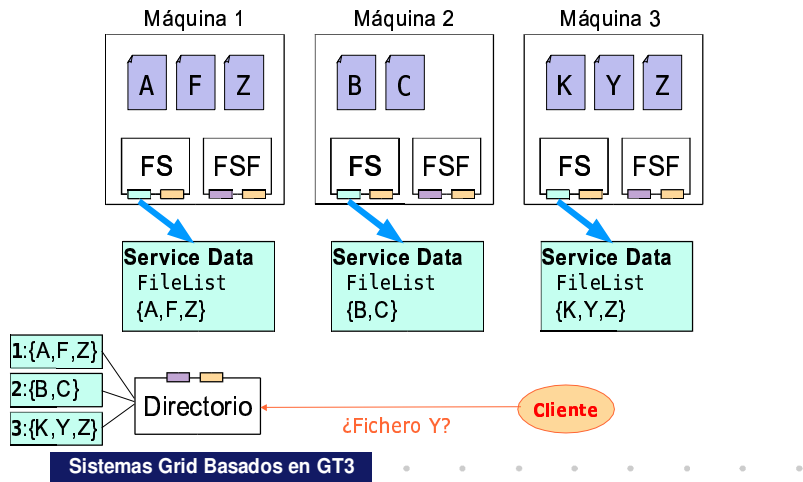
Un ejemplo (XIV)



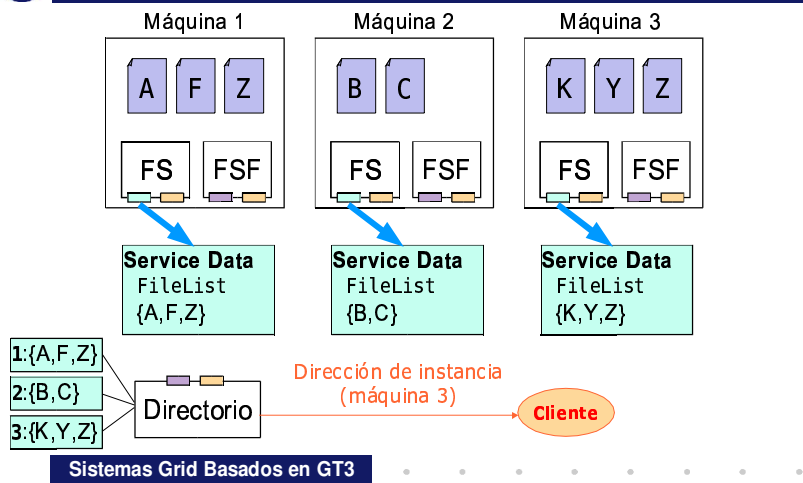
Sistemas Grid Basados en GT3



Un ejemplo (XV)

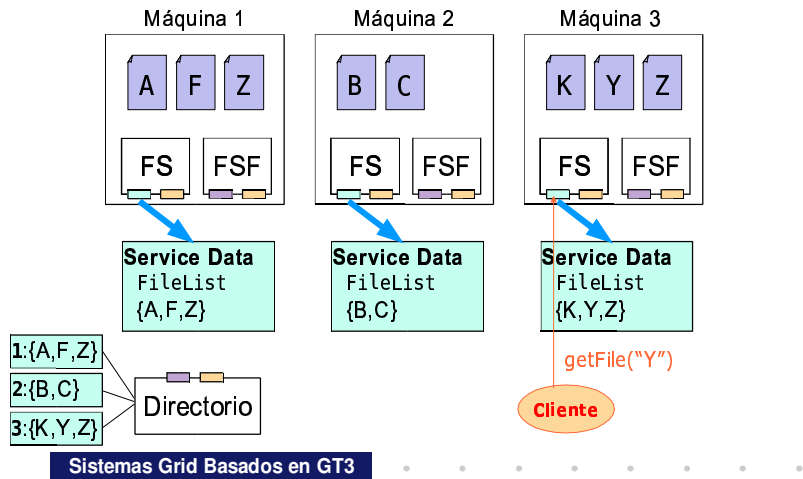


Un ejemplo (XVI)

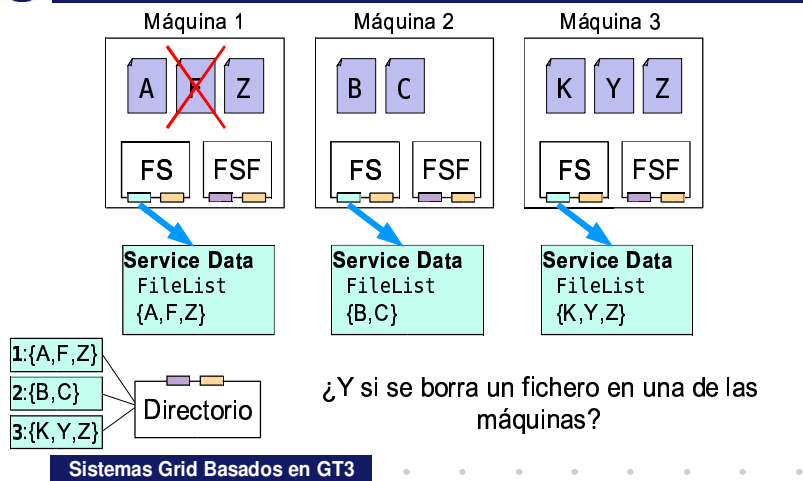




Un ejemplo (XVII)

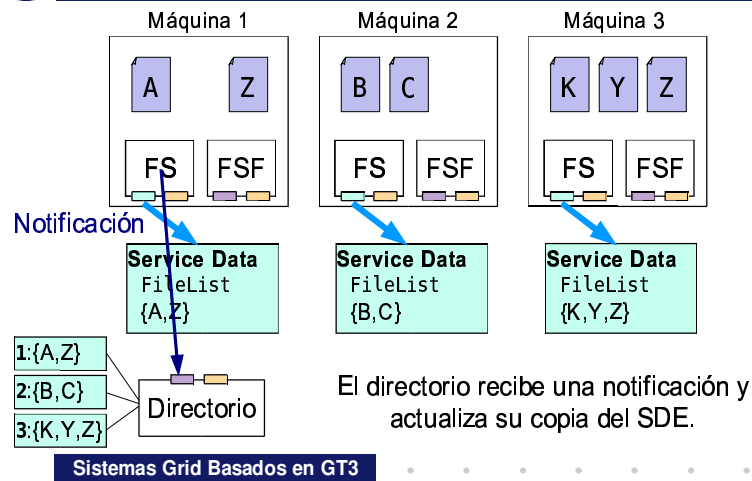


Un ejemplo (XVIII)





Un ejemplo (XIX)



Índice

- ▶ Orígenes de OGSA
- ▶ Introducción a los Web Services y SOA
- ▶ Grid Services (OGSI)
- ▶ **Globus Toolkit 3**
- ▶ Relación entre OGSA, OGSI, y GT3
- ▶ WSRF/GT4



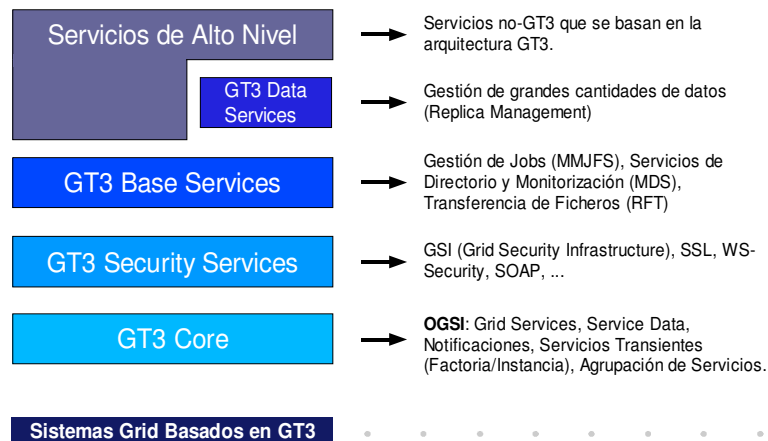
Globus Toolkit 3

- ▶ El Globus Toolkit 3 (GT3) es una implementación completa, en Java, de OGSi.
 - ▶ No es la única implementación disponible, aunque sí la más completa.
- ▶ ¡Ojo! GT3 no es *únicamente* una implementación de OGSi. Incluye muchos otros servicios basados en OGSi.
 - ▶ También incluye una distribución GT2.4 completa.

Sistemas Grid Basados en GT3



Arquitectura de GT3 (I)





Arquitectura de GT3 (II)



Requisitos (I)

- ▶ **Sistemas en los que podemos instalar GT3**
 - ▶ El toolkit *completo* sólo está disponible para sistemas UNIX.
 - ▶ Sin embargo, como el núcleo y buena parte de los Base Services son Java puro (sin código nativo), también existe una versión reducida para Windows.
- ▶ **Requisitos de hardware**
 - ▶ El toolkit en si no tiene requisitos específicos de hardware.
 - ▶ Estos requisitos los marcarán las aplicaciones que queramos ejecutar utilizando GT3.

Sistemas Grid Basados en GT3



Requisitos (II)

- ▶ Requisitos de software
 - ▶ JDK 1.4 o superior
 - ▶ <http://java.sun.com/j2se>
 - ▶ También es posible utilizar JDK 1.3.1 o superior, pero requiere una instalación especial.
 - ▶ Jakarta Ant 1.5 o superior
 - ▶ <http://ant.apache.org/>
 - ▶ Junit 3.8.1 (opcional)
 - ▶ <http://www.junit.org/>



Desarrollo con GT3

- ▶ Metodología de desarrollo
 - ▶ Definir el interfaz del servicio – **GWSDL**
 - ▶ Implementar el servicio – **Java**
 - ▶ Definir detalles del despliegue – **WSDD**
 - ▶ ¿Qué dirección tendrá el servicio?
 - ▶ ¿En qué fichero se encuentra la implementación?
 - ▶ etc.
 - ▶ Compilar todo, generando un fichero GAR – **Ant**
 - ▶ Compilar Java, generar stubs
 - ▶ GAR: Grid ARchive. Incluye ficheros WSDL y Java compilados
 - ▶ Desplegar servicio (fichero GAR) en entorno de ejecución – **Ant**



Entornos de Ejecución (I)

- ▶ 4 entornos de ejecución:
 - ▶ Standalone Container
 - ▶ Contenedor de grid services incluido con GT3.
 - ▶ Sencillo. Rendimiento bajo.
 - ▶ Java Servlet Engine
 - ▶ Tomcat - <http://jakarta.apache.org/tomcat/>
 - ▶ WebSphere - <http://www.ibm.com/websphere/>
 - ▶ Más complejo. Rendimiento medio-alto.



Entornos de Ejecución (II)

- ▶ J2EE Enterprise JavaBeans Container
 - ▶ El grid service se aloja en un contenedor de EJBs.
 - ▶ WebSphere Application Server
- ▶ Embedded Container
 - ▶ Contenedor mínimo para que clientes y servidores ligeros puedan exponer grid services.



Problemas de GT3 (I)

- ▶ Problemas de GT3
 - ▶ Rendimiento
 - ▶ En ciertos casos se dan tiempos de respuesta prohibitivos, no aptos para entornos de producción.
 - ▶ Usabilidad
 - ▶ API poco intuitiva. Nomenclatura dispar.
 - ▶ Documentación
 - ▶ Poca documentación didáctica, API poco documentada, falta de un repositorio central de documentación.



Índice

- ▶ Orígenes de OGSA
- ▶ Introducción a los Web Services y SOA
- ▶ Grid Services (OGSI)
- ▶ Globus Toolkit 3
- ▶ Relación entre OGSA, OGSI, y GT3
- ▶ WSRF/GT4



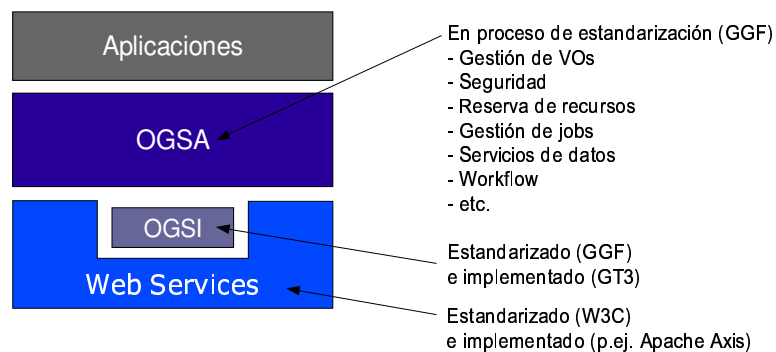
Relación entre OGSA, OGSi, y GT3 (I)

- ▶ Hemos visto:
 - ▶ OGSA: Una arquitectura de servicios para aplicaciones Grid. Se basa en...
 - ▶ OGSi: Propone un modelo de Web Services *mejorados* cuya principal implementación es...
 - ▶ GT3: Un toolkit de desarrollo que también incluye servicios de seguridad, gestión de jobs, etc.
- ▶ Repasemos la relación entre ellos...

Sistemas Grid Basados en GT3



Relación entre OGSA, OGSi, y GT3 (II)



Sistemas Grid Basados en GT3



Relación entre OGSA, OGSi, y GT3 (III)

- ▶ Otras implementaciones de OGSi
 - ▶ OGSi::Lite (Perl)
<http://www.sve.man.ac.uk/Research/AtoZ/ILCT>
 - ▶ OGSi.net (.NET)
<http://www.cs.virginia.edu/~humphrey/GCG/ogsi.net.html>
 - ▶ MS.NETGrid (.NET)
<http://www.epcc.ed.ac.uk/~ogsanet/>
 - ▶ pyOGSi (Python)
<http://www-itg.lbl.gov/gtg/projects/pyOGSi/>



Índice

- ▶ Orígenes de OGSA
- ▶ Introducción a los Web Services y SOA
- ▶ Grid Services (OGSi)
- ▶ Globus Toolkit 3
- ▶ Relación entre OGSA, OGSi, y GT3
- ▶ WSRF/GT4



WSRF/GT4 (I)

- ▶ A pesar del impulso que se le ha dado a OGSi y OGSA en la comunidad Grid, OGSi en particular ha sido recibido con recelo por la comunidad de Web Services.
 - ▶ El GGF esperaba que OGSi y los estándares de Web Services convergieran en el futuro. Sin embargo, esa convergencia no se está produciendo.



WSRF/GT4 (II)

- ▶ Defectos de OGSi
 - ▶ Especificación demasiado extensa y espesa
 - ▶ No se integra bien con herramientas actuales de Web Services
 - ▶ Demasiado orientado a objetos
 - ▶ Los Web Services no son orientados a objetos, a pesar de que puedan tener por detrás un sistema de objetos.

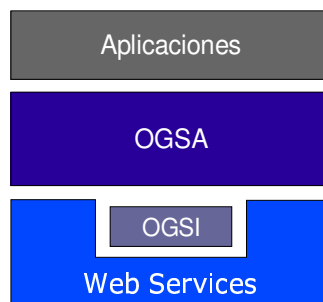


WSRF/GT4 (III)

- ▶ Para conseguir que los grid services y los web services puedan converger, y para corregir los defectos de OGSi, se anunció (por sorpresa) un nuevo estándar durante GlobusWorld 2004 (enero 2004).
- ▶ Este nuevo estándar sustituirá a OGSi.
- ▶ **WSRF** – Web Services Resource Framework



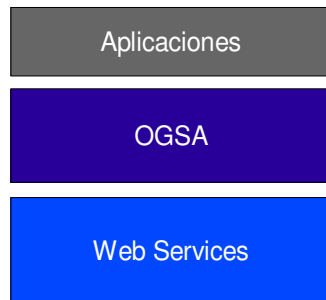
WSRF/GT4 (IV)





WSRF/GT4 (V)

OGSA se apoyaría en servicios web directamente, en vez de tener que utilizar una 'versión especial' ('servicios OGS!')



WSRF sería una parte de la colección de estándares WS, en lugar de un 'parche' (como OGS!)

Sistemas Grid Basados en GT3



WSRF/GT4 (VI)

- ▶ **Cómo corrige WSRF los defectos de OGS!**
 - ▶ *“Especificación demasiado extensa y espesa”*
 - ▶ WSRF está dividido en cinco documentos y una especificación complementaria (WS-Notification)
 - ▶ *“No se integra bien con herramientas actuales de Web Services”*
 - ▶ WSRF está más en sintonía con la comunidad de WS: uso menos agresivo de XML Schema, WSDL 1.1 puro en lugar de 'WSDL parcheado'
 - ▶ *“Demasiado orientado a objetos”*
 - ▶ Diferenciación clara entre el servicio y el 'recurso' (el estado), puesto que un web service no puede tener estado. Se elimina el patrón Factoría/Instancia puesto que un web service tampoco puede tener instancias.

Sistemas Grid Basados en GT3



WSRF/GT4 (VII)

- ▶ A nivel conceptual, WSRF y OGSI incluyen la misma funcionalidad.
 - ▶ Cambios sintácticos
 - ▶ Factorización en varios sub-estándares
- ▶ En teoría, la transición de OGSI a WSRF será sencilla.
 - ▶ Correspondencia directa de OGSI a WSRF
- ▶ ¿Qué pasa con OGSI/GT3? ¿Merece la pena seguir desarrollando?
 - ▶ Respuesta oficiosa: Sí, GT3 seguirá estando soportado, y la gente que esté familiarizada con OGSI no tendrá ningún problema en dar el salto a WSRF.

Sistemas Grid Basados en GT3



WSRF/GT4 (VIII)

- ▶ ¿Por qué tiene WSRF más posibilidades de triunfar que OGSI?
 - ▶ Está más en sintonía con los web services.
 - ▶ Cuenta el apoyo de IBM y HP
- ▶ Primera implementación de WSRF: GT4
 - ▶ IBM incluirá soporte para WSRF en algunas de sus herramientas.
- ▶ Web de WSRF: <http://www.globus.org/wsrf>

Sistemas Grid Basados en GT3



Evolución de GT3/GT4

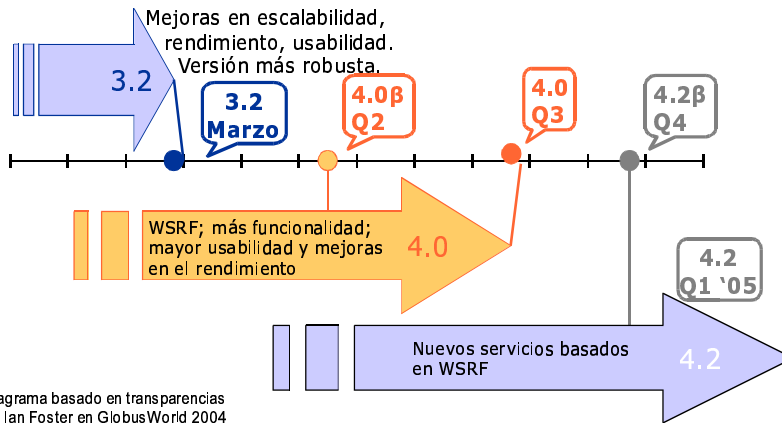


Diagrama basado en transparencias de Ian Foster en GlobusWorld 2004

Sistemas Grid Basados en GT3

¿Preguntas?



Borja Sotomayor
Facultad de Ingeniería - ESIDE
Universidad de Deusto
bsotomay@eside.deusto.es

Sistemas Grid Basados en GT3