CrossBroker: gestión de aplicaciones paralelas e interactivas en entornos Grid

CrossBroker: Management of Parallel and Interactive Applications on Grid Environments

E. Fernández, A. Morajko, A. Fernández et al.

Resumen

Tradicionalmente, el uso de los sistemas Grid se ha limitado a trabajos de tipo *batch* para los que el usuario final realiza las tareas de planificación y control. En el proyecto europeo CrossGrid, se ha extendido el uso de estos entornos a aplicaciones interactivas y paralelas, proporcionando herramientas para la ejecución automática y transparente de este tipo de aplicaciones. En este artículo se presenta el gestor de trabajos CrossBroker, encargado de la planificación de trabajos de forma automática en CrossGrid.

Palabras clave: Grid, CrossGrid, interactividad, aplicaciones distribuidas, flujos de tareas.

Summary

Traditionally, Grid systems has been limited to batch-like jobs, where the final user is responsible for scheduling and control tasks. The CrossGrid European Project has extended the use of Grid environments to support parallel and interactive jobs, providing tools for automatic and transparent execution of these applications. This paper describes the CrossBroker resource manager, which schedules users applications automatically in CrossGrid.

Keywords: Grid, CrossGrid, interactivity, distributed applications, workflows.

El gestor de trabajos CrossBroker se encarga de la planificación de trabajos de forma automática en CrossGrid

1. Introducción

La computación Grid a gran escala necesita servicios de gestión de recursos que afronten el proceso de localizar los distintos tipos, prepararlos para su uso, utilizarlos y monitorizar su estado. En los sistemas tradicionales, la gestión de recursos es un problema bien estudiado para el que existe un número significativo de soluciones. Todos estos gestores de recursos están diseñados y operan con control total del recurso gestionado y, por lo tanto, pueden implementar mecanismos y políticas para el uso eficiente de dicho recurso de forma aislada.

La computación Grid necesita servicios de gestión de recursos que afronten el proceso de localizar los distintos tipos, prepararlos para su uso, utilizarlos y monitorizar su estado

En los entornos Grid, la gestión de recursos tiene que tratar con recursos dispersos geográficamente que pertenecen a diferentes dominios administrativos, por lo que las suposiciones de los sistemas tradicionales no son aplicables.

La mayoría de los sistemas descritos en la literatura siguen un patrón similar para la planificación de un trabajo en un Grid. Existen tres fases típicas:

- **Descubrimiento de recursos**, que genera una lista de recursos potenciales que pueden usarse para una aplicación dada. Esta fase requiere que el usuario tenga acceso a un conjunto de ellos y que se especifique un conjunto mínimo de requisitos de la aplicación. Estos requisitos se usarán para filtrar los recursos que no los cumplan.
- **Obtención de información** sobre los recursos disponibles y elección del mejor a nivel individual o conjunto de recursos. Esta elección, realizada a partir de la lista obtenida en la fase anterior, puede realizarse aplicando mecanismos heurísticos que usen información adicional sobre el estado dinámico de los recursos.
- **Ejecución del trabajo,** que incluye la transferencia de archivos y la limpieza final. Una vez se han elegido los recursos, la aplicación puede enviarse para su ejecución.

El gestor de recursos CrossBroker [2], desarrollado en el proyecto europeo CrossGrid [3], sigue este mismo esquema pero centrándose en nuevos tipos de aplicaciones. En CrossGrid se proponen 4 aplicaciones interactivas y con una alta demanda computacional: simulación y visualización para cirugía, un sistema de ayuda a la toma de decisiones en casos de inundación, análisis distribuido de datos en física de altas energías y polución aérea combinada con predicción del tiempo. Estas aplicaciones introducen nuevos requisitos no existentes en las aplicaciones *batch* a la hora de realizar la planificación:

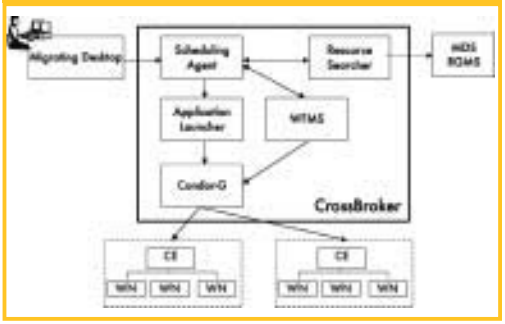
- Co-asignación. Los trabajos pueden necesitar más de un recurso para su ejecución, por lo que hay que crear mecanismos para la búsqueda de conjuntos de recursos y la ejecución simultánea del trabajo en ellos.
- Control de dependencias. Para flujos de trabajos; el planificador debe controlar las dependencias existentes para enviar en el momento adecuado cada uno de los subtrabajos.
- Control de prioridades. Los trabajos interactivos deben iniciarse casi inmediatamente, por lo que mecanismos de control de prioridades y multiprogramación pueden ser útiles para su planificación.
- Envío de los flujos de entrada y salida. El usuario debe poder interactuar con las aplicaciones interactivas, para lo que debe recibir la salida de los trabajos y poder enviarle la entrada durante su ejecución.

En CrossGrid se proponen 4 aplicaciones interactivas y con una alta demanda computacional

2. Arquitectura del gestor de recursos CrossBroker

El gestor de trabajos y recursos Crossbroker es el encargado de realizar todas las tareas asociadas a la planificación de trabajos en CrossGrid de la forma más eficiente posible, para llegar a un correcto balance entre la ejecución de cada aplicación y el uso de los recursos.

FIGURA 1: ARQUITECTURA DEL CROSSBROKER



En la figura se presentan los componentes principales de CrossBroker[4] y su relación con otros de CrossGrid. Típicamente, el usuario, a través de herramientas de interfaz desarrolladas en CrossGrid como el Migrating Desktop, envía su trabajo al CrossBroker para su ejecución.

El usuario, a través de las herramientas de interfaz desarrolladas en CrossGrid, envía su trabajo al CrossBroker para su ejecución

El trabajo se describe mediante el lenguaje JDL (*Job Description Language*) [5], en el que se incluyen atributos que definen el tipo de trabajo, sus ficheros de entrada y salida, los requisitos y las preferencias del usuario. Por ejemplo, en la figura 2 se muestra un JDL para un trabajo interactivo y paralelo (*mpich*) que necesita 8 CPU para su ejecución, requiere máquinas con sistema operativo Linux y prefiere las máquinas con mejor puntuación en el *benchmark* SpecInt2000. El ejecutable "simula_mpi" también se especifica como fichero a transferir al sitio remoto.

FIGURA 2: EJEMPLO DE JDL

```
[
Executable      = "simula_mpi";
JobType         = { mpicj, interactive };
NodeNumber      = 8;
InputSandbox    = { /users/enol/simula_mpi.config }
Requirements    =  other. GlueHostOperatingSystemName ==  linux ;
Rank            = other.GlueHostBenchmarkSI00;
]
```

Para cada trabajo enviado, el CrossBroker invoca al Scheduling Agent, que realiza siempre la misma secuencia de pasos. En primer lugar, obtiene del Resource Searcher (RS) una lista de recursos disponibles para ejecutar la aplicación. El RS, a partir de la descripción del trabajo y sus requisitos, realiza un proceso de búsqueda de recursos disponibles para la ejecución del trabajo usando servicios de directorio basado en MDS [6] o RGMA [7].

A continuación, el SA elige el recurso o conjunto de recursos en el que ejecutar la aplicación, y pasa esta información junto con el trabajo al Application Launcher (AL) que será el responsable del envío del trabajo para su ejecución en el sitio o sitios remotos. El Application Launcher se encarga además de monitorizar el estado de las aplicaciones y proporcionar una ejecución fiable de las mismas, gracias a los servicios ofrecidos por Condor-G [8]. Cada uno de los sitios está formado por un nodo de entrada denominado Computing Element (CE) en el que existe un gestor de colas local (como por ejemplo PBS, LSF, Condor) y una serie de Worker Nodes (WN) en los que se ejecutan las aplicaciones.

En el caso de los flujos de tareas, el Workflow Managent System (WFMS) se encargará de analizar las dependencias e invocar al Scheduling Agent para cada una de las subtareas que realizará el proceso de búsqueda y selección de recursos.

El Application Launcher se encarga además de monitorizar el estado de las aplicaciones y proporcionar una ejecución fiable de las mismas

2.1. Gestión de trabajos paralelos

CrossBroker da soporte a aplicaciones paralelas compiladas con la librería MPICH [9], tanto *intracluster* (mpich-p4) como *intercluster* (mpich-g2). Las aplicaciones mpich-p4 permiten el uso de múltiples máquinas de un único *cluster,* mientras que las aplicaciones compiladas con mpich-g2 pueden ejecutarse en múltiples *clusters* de forma simultánea.

El Resource Searcher se encarga de realizar la búsqueda de los *clusters* capaces de ejecutar las aplicaciones mpich-p4 usando la capacidad de *matching* de la librería de ClassAd [10] de Condor. Para el caso de aplicaciones mpich-g2 se ha creado un nuevo algoritmo capaz de crear conjuntos de recursos y asociar los mismos a un solo trabajo.

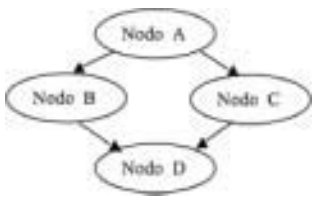
El Application Launcher para aplicaciones mpich-g2, junto a los mecanismos de Condor-G, garantiza la co-asignación de recursos, el restablecimiento en caso de error y una semántica de ejecución única para estos trabajos. Constituye por lo tanto un servicio confiable para el envío de aplicaciones mpich-g2 y sustituye a los servicios de envío de trabajos provistos por el *toolkit* de Globus [11].

El Resource Searcher se encarga de realizar la búsqueda de los *clusters* capaces de ejecutar las aplicaciones mpich-p4

2.2. Planificación de flujos de tareas

En el caso de los flujos de tareas o *workflows* [12], se ha ampliado el lenguaje JDL para describir DAG (Directed Acyclic Graph) en los que cada nodo es una tarea y las aristas indican las dependencias entre las mismas. En la figura 3 se muestra un DAG de ejemplo compuesto por cuatro tareas, en las que el nodo B y C dependen del nodo A y el nodo D depende de los nodos B y C.

FIGURA 3: EJEMPLO DE DAG



A la hora de planificar estos flujos, el módulo WFMS se encarga de navegar a través del flujo, determinando qué tareas están libres de dependencias y monitorizar la ejecución de los trabajos correspondientes. Para cada tarea del DAG se realizan los siguientes pasos:

- Fase inicial: preparación de toda la información para la ejecución del trabajo y búsqueda de recursos para ejecutarlo posteriormente usando Condor-G.
- Ejecución del trabajo en el sitio remoto.
- Fase final: comprobación de la correcta ejecución del trabajo. Si el trabajo se ejecutó de forma exitosa, se marca como finalizado. En caso contrario se marca como fallido y se reintentará.

En el caso de no poder continuar la ejecución de un workflow tras agotar todos los reintentos indicados por el usuario se proporciona la posibilidad de recuperar la ejecución a través de un fichero de recuperación generado automáticamente por el CrossBroker. A partir de este fichero se puede reenviar el workflow en un instante posterior de forma que CrossBroker reintente ejecutar de nuevo sólo los nodos que quedaron pendientes la vez anterior.

2.3. Servicios interactivos

A la hora de planificar las aplicaciones interactivas, el gestor de trabajos debe procurar que su inicio se lleve a cabo en un futuro predecible

A la hora de planificar las aplicaciones interactivas, el gestor de trabajos debe procurar que su inicio se lleve a cabo en un futuro predecible y los mecanismos de interacción con la aplicación durante su ejecución. Estos requisitos son difíciles de cumplir en la mayoría de las infraestructuras Grid existentes, puesto que se orientan a trabajos de tipo *batch*. Sin embargo, en nuestro sistema se incluyen características específicas para estos trabajos.

De forma ideal, las aplicaciones interactivas deberían empezar su ejecución justo tras su envío al gestor de recursos; sin embargo, pueden darse casos en los que los recursos remotos no estén disponibles porque están ejecutando otros trabajos *batch*. CrossBroker incorpora un mecanismo especial para tratar estas situaciones permitiendo que una aplicación interactiva y una *batch* compartan una misma máquina, de forma que las aplicaciones que lo requieran puedan iniciarse lo más pronto posible (a no ser que todos los recursos ya estén ocupados por otras aplicaciones interactivas).

Para conseguir este esquema de multiprogramación, se hace uso de un mecanismo de agentes que permiten obtener el control de las máquinas remotas independientemente del planificador del sitio. El agente crea en cada máquina dos virtuales para crear grupos separados de recursos dedicados para los dos tipos de aplicaciones contemplados: los trabajos *batch* se ejecutarán en una de las máquinas y los interactivos en otra. De este modo los trabajos *batch* no tendrán que ser suspendidos ni retrasarán el inicio de los interactivos, ya que los dos tipos de trabajos no compiten por la misma máquina virtual.

Las sesiones interactivas se gestionan mediante el uso de dos agentes: Console Agent y Console Shadow

Cuando se lanza un trabajo interactivo, el usuario debe poder controlar tanto la entrada como la salida en tiempo casi real desde la máquina en la que se envió el trabajo. Por lo tanto cada vez que se inicia un trabajo de este tipo, se crea una sesión interactiva que comunica la máquina local con la remota.

Las sesiones interactivas se gestionan mediante el uso de dos agentes (Console Agent y Console Shadow) que permiten, gracias a un mecanismo de interposición de librerías, obtener de forma casi continua la entrada y salida de las aplicaciones ejecutadas remotamente en una red poco fiable.

El CrossBroker coloca el Console Agent entre la aplicación y el sistema operativo de forma que, cuando el programa realiza ciertas llamadas al sistema (las relacionadas con la entrada y salida), el agente toma el control y las reenvía al proceso Console Shadow en la máquina del usuario. El uso de este sistema se realiza de forma transparente sin necesidad de cambios en las aplicaciones.

3. Conclusiones

El gestor de recursos de CrossGrid, CrossBroker se ha presentado en este trabajo como un planificador capaz de gestionar de manera automática tanto trabajos secuenciales o *batch,* como los trabajos a los que se enfoca CrossGrid: aplicaciones paralelas, aplicaciones interactivas y *workflows*. Este gestor se ha desarrollado en la UAB con la colaboración del IFIC como parte del proyecto CrossGrid partiendo de un desarrollo existente en el proyecto DataGrid, proyecto europeo antecesor de CrossGrid.

El CrossBroker se compone de tres módulos principales:

- *Scheduling Agent*, encargado de la selección del recurso más adecuado para cada trabajo.
- *Resource Searcher* realiza el proceso de descubrimiento y búsqueda de recursos adecuados para cada trabajo.
- *Application Launcher*. Módulo responsable de la ejecución fiable de las aplicaciones.

El CrossBroker se compone de tres módulos:
• Scheduling Agent
• Resource Searcher
• Application Launcher

Para la gestión y ejecución de trabajos interactivos se ha desarrollado un sistema de agentes que permite la transmisión en tiempo casi-real de la entrada y salida de las aplicaciones; y un sistema de priorización que permite ejecutar, mediante el uso de multiprogramación, las aplicaciones interactivas incluso cuando los recursos existentes están ocupados.

Nuestro sistema de gestión de recursos forma parte de las herramientas desarrolladas en el proyecto CrossGrid y se distribuye como parte de la distribución estable 3.0.4 del *middleware* de CrossGrid. A través del CrossBroker se han enviado trabajos a las máquinas del testbed distribuido que reúne a 21 instituciones de 11 países diferentes.

Durante el desarrollo del CrossBroker, nos hemos centrado en los distintos mecanismos base que permitan el envío fiable de todos los tipos de trabajos de forma automática sin intervención por parte del usuario y siguiendo políticas básicas. A partir de estos mecanismos, existen varias líneas de trabajo a considerar en el futuro, incluyendo la evaluación de políticas de planificación más complejas que lleven a un mejor aprovechamiento de recursos junto con la posibilidad de realizar planificación distribuida mediante la colaboración de varios Scheduling Agents.

Para la gestión de trabajos interactivos se ha desarrollado un sistema de agentes que permite la transmisión en tiempo casi-real de la entrada y salida de las aplicaciones

Enol Fernández
(enol@aomail.uab.es)
Anna Morajko
(ania@aomail.uab.es)
Elisa Heymann
(elisa.heymann@uab.es)
Miquel A. Senar
(miquelangel.senar@uab.es)
Dept. d'Arquitectura de Computadors i Sistemes Operatius
Universitat Autònoma de Barcelona
Álvaro Fernández
(alferca@ific.uv.es)
Instituto de Física Corpuscular
Valencia

Referencias

[1] J. Schopf. "Ten Actions When Grid Scheduling" capítulo de Grid Resource Management: State of the Art and Future Trends. Editores: Jarek Nabrzyski, Jennifer M. Schopf and Jan Weglarz. 2003

[2] E. Fernández, A. Fernández, E. Heymann, M. A. Senar, J. Salt. "Managing MPI Applications in Grid Environments". 2nd European Across Grids Conference, Nicosia, Cyprus. 2004

[3] European CrossGrid Project. *www.eu-crossgrid.org*

[4] E. Fernández, A. Fernández, A. Mora Jko, E. Heymann, and M. Senar. "Job Management of Parallel Applications in the CrossGrid Project".

[5] F. Pacini, "Job Description Language Howto", *http://server11.infn.it/workload-grid/docs/DataGrid-01-TEN-0102-0_2-Document.pdf*

[6] K. Czajkowski, S. Fitzgerald, I. Foster, and C. Kesselman. "Grid Information Services for Distributed Resource Aharing". In 10th IEEE Symp. On High Performance Distributed Computing, 2001.

[7] Andrew W. Cooke, Alasdair J. G. Gray, Lisha Ma et alt. "R-gma: An information Integration System for Grid Monitoring". In Cooperative Information Systems (CoopIS) International Conference, pages 462–481, 2003.

[8] James Frey, Todd Tannenbaum, Miron Livny, Ian T. Foster, and Steven Tuecke. "Condor-g: A Computation Management Agent for Multi-institutional Grids". Cluster Computing, 5(3): 237–246, 2002.

[9] William Gropp, Ewing L. Lusk, Nathan Doss, and Anthony Skjellum. "A High-Performance, Portable Implementation of the Mpi Message Passing Interface Atandard". Parallel Computing, 22(6):789–828, 1996.

[10] Rajesh Raman, Miron Livny, and Marvin H. Solomon. "Matchmaking: Distributed Resource Management for High Throughput Computing". In IEEE International Symposium on High Performance Distributed Computing, pages 140–, 1998.

[11] Ian Foster and Carl Kesselman. "Globus: A Metacomputing Infrastructure Toolkit". The International Journal of Supercomputer Applications and High Performance Computing, 11(2):115–128, Summer 1997.

[12] A. Morajko, E. Fernández, A. Fernández, E. Heymann, and M. Senar. "Workflow Management in the Crossgrid Project". In European Grid Conference, Amsterdam, 2005.