Computación Grid e Ingeniería del Software Basada en Componentes en CSCL

Miguel L. Bote Lorenzo, Juan I. Asensio Pérez, Yannis A. Dimitriadis, Eduardo Gómez Sánchez, Luis M. Vaquero González, Guillermo Vega Gorgojo

ETSI de Telecomunicación, Universidad de Valladolid
Camino Viejo del Cementerio s/n, 47011 Valladolid, Spain
{migbot@, juaase@, yandim@, edugom@, lvaqgon@ribera, guiveg@}tel.uva.es

Resumen

Este artículo presenta el trabajo desarrollado por los autores para habilitar el uso de las infraestructuras de computación *grid* como soporte de aplicaciones de Aprendizaje Colaborativo Apoyado por Ordenador (*Computer Supported Collaborative Learning*, CSCL) desarrolladas de acuerdo con los principios de la Ingeniería del Software Basada en Componentes (ISBC). Para ilustrar y justificar el interés de este objetivo se presenta un ejemplo significativo de aplicación CSCL basada en componentes soportada por un *grid*. Este ejemplo sirve de punto de partida para plantear el problema de la planificación (*scheduling*) de aplicaciones en el ámbito del software CSCL, además de para proponer la necesidad de un servicio de despliegue dinámico en el ámbito de la arquitectura OGSA (*Open Grid Services Architecture*, Arquitectura Abierta de Servicios Grid).

Palabras clave: CSCL, grid, componentes, planificación, despliegue.

1. Introducción

Los beneficios potenciales de la ISBC se han hecho evidentes dentro del campo del software educativo, donde las particularidades sociales y pedagógicas de cada contexto educativo se han traducido, tradicionalmente, en la aparición de grandes conjuntos de aplicaciones monolíticas e incompatibles entre sí. Este hecho ha impedido, en gran medida, que las tecnologías de la información tuvieran una amplia aceptación en las aulas [Roschelle98]. Este problema es aún más destacado en el caso de un tipo particular de software educativo: el de las aplicaciones CSCL.

El CSCL, surgido en parte como evolución del CSCW (*Computer-Supported Colaborative Work* o Trabajo Colaborativo Apoyado por Ordenador), refleja un nuevo paradigma de investigación y práctica educativa fuertemente interdisciplinar [Koschmann96]. Éste se caracteriza por realzar la importancia de las interacciones sociales (colaboración) como elemento esencial del aprendizaje, por la preferencia por un enfoque interpretativo frente al positivista tradicional a la hora de evaluar el aprendizaje, así como por el papel del análisis y diseño participativo de toda la comunidad en la creación de nuevos entornos tecnológicos que reflejen estos principios [Martínez03].

Este diseño participativo implica que las aplicaciones CSCL deben soportar la funcionalidad requerida por una importante diversidad de actores (profesores, estudiantes, pedagogos, psicólogos, etc...). Así pues, el esfuerzo invertido en su desarrollo sólo se justifica si dichas aplicaciones se pueden emplear en un número suficientemente grande de situaciones de aprendizaje y si, además, son capaces de sobrevivir a la evolución de los

requisitos funcionales y a más que posibles cambios tecnológicos [Martínez03]. Por lo tanto, la ISBC se muestra como una tecnología prometedora para ser empleada en el desarrollo de herramientas software CSCL reutilizables, adaptables e integrables.

Además de todo lo anterior, existe una sinergia más que destacable entre la ISBC y la computación basada en *grids*: varios proyectos de investigación en marcha, tales como ICENI [Furmento02], sugieren que la computación *grid* constituye una tecnología muy adecuada para soportar el procesamiento distribuido de aplicaciones basada en componentes. En esta misma dirección, OGSA, que ha surgido como el estándar de facto para la construcción de sistemas *grid* [Foster02a], reconoce como una alternativa interesante la utilización de contenedores de componentes software para albergar la implementación de la funcionalidad de los denominados *Servicios Grid* [Foster02b].

Además de esta doble sinergia (ISBC-CSCL y Grid-ISBC) también es posible establecer una tercera relación entre CSCL y computación *grid*: esta última tecnología ha demostrado que es capaz de mejorar las interacciones síncronas y asíncronas entre personas [Foster98], lo cual hace suponer que las infraestructuras *grid* pueden proporcionar un soporte muy valioso a las aplicaciones CSCL. Sin embargo, y a pesar de que se considera a la educación "un campo importante (y obvio) de aplicación de tecnologías *grid*", el uso de éstas en contextos CSCL no se ha tratado, a juicio de los autores, con suficiente profundidad. Consecuentemente, los autores primeramente realizaron un trabajo [Bote03a] para profundizar en las principales características de la computación *grid* descritas en la literatura. El análisis de dichas características confirma que el uso de infraestructuras *grid* puede proporcionar importantes beneficios a las aplicaciones CSCL [Bote03b]: gran escala de estas infraestructuras, amplia distribución de los recursos, soporte de relaciones entre organizaciones diferentes y heterogeneidad de los recursos compartidos en un *grid*.

En este contexto, el artículo presenta el trabajo en el que los autores están actualmente involucrados y que se orienta hacia la fusión de las tecnologías *grid*, la ISBC y el CSCL. Para ello, el artículo describe un posible escenario de aplicación de esta combinación tecnológica, gracias al cual es posible identificar dos importantes problemas cuya resolución debe ser investigada para permitir que las aplicaciones CSCL se beneficien de la ISBC y la computación *grid*. Estos dos problemas se centran, por una parte, en la planificación de aplicaciones CSCL basadas en componentes en un *grid* y, por otra, en el despliegue dinámico de los componentes de las mismas.

La organización del artículo es la siguiente: la sección 2 describe y discute el mencionado escenario en el que se pueden identificar los problemas ya comentados. La sección 3 aborda el primero de ellos y que se relaciona con la planificación de aplicaciones CSCL. Por su parte, la sección 4 hace lo propio con el segundo de los problemas: el relacionado con la necesidad de disponer de capacidades de despliegue dinámico de aplicaciones basada en componentes en infraestructuras *grid*. Por último, en la sección 5 se pueden encontrar las conclusiones más importantes y las líneas futuras de trabajo que se derivan de lo aquí presentado.

2. Ejemplo de Escenario de Aplicación CSCL basada en Componentes soportada por *Grid*

El uso conjunto de las tecnologías *grid* y los principios de la ISBC puede proporcionar importantes ventajas para aplicaciones CSCL como la siguiente: alumnos de diferentes colegios interactúan asíncronamente para publicar, colaborativamente, y leer una revista electrónica. En este escenario, a los alumnos se les proporciona un editor colaborativo (para la confección de los artículos) y una herramienta auxiliar con la que organizar colaborativamente las ideas que pretenden incluir en sus artículos. Esta última, que incluso podría venir acompañada de software "inteligente" de elevados requisitos computacionales, permitiría la generación de los denominados "mapas cognitivos" [Honkela00]. La herramienta debería también dar soporte a la resolución de conflictos: los alumnos proponen nuevos conceptos y relaciones, los discuten y, por último, acuerdan un único mapa cognitivo final. Es más, esta herramienta debería facilitar el acceso de los alumnos a fuentes de información (por ejemplo, páginas *web*, artículos anteriores, etc...) para poder enlazarlas con los conceptos y relaciones que previamente han propuesto. Los objetivos de aprendizaje de este contexto abarcan tanto la adquisición de habilidades de escritura como la comprensión de los principales conceptos relacionados con los artículos que los alumnos escriben y/o leen.

Si se aplican los principios de desarrollo propios de la ISBC, la aplicación CSCL de soporte del escenario descrito podría ser el resultado del ensamblaje de componentes software tales como

una interfaz gráfica de usuario, un coordinador de alumnos, un gestor de grupos, un gestor de los repositorios de datos, los ya mencionados editor colaborativo y editor de mapas cognitivos, etc... La funcionalidad de dichos componentes podría ser replicada y ejecutada en diferentes nodos de un *grid* (potencialmente de diferentes colegios) aprovechando las ventajas ya comentadas de gran escala y amplia distribución geográfica. Todas estas cuestiones facilitarían el que un gran número de alumnos pudieran estar involucrados en las tareas de lectura y/o escritura de artículos. Además, el repositorio de artículos podría estar replicado y distribuido por todo el *grid* con lo que los tiempos de respuesta a la hora de acceder a los artículos se reduciría, siempre y cuando se tuviera en cuenta la disposición geográfica de los colegios y sus alumnos. También la herramienta de generación de mapas cognitivos podría ser mejorada con la inclusión de un componente que ofreciera un servicio de minería de datos *grid* (como la descrita en [Foster02a]) con el que obtener, de forma eficiente, información de utilidad para los alumnos durante la confección de los mapas.

La consecución de los beneficios identificados en este escenario pasa por la disponibilidad de determinados mecanismos en las infraestructuras *grid* susceptibles de dar soporte a aplicaciones CSCL basadas en componentes: (1) un planificador de aplicaciones CSCL basadas en componentes que decida qué componentes han de ser migrados/replicados sobre qué nodos del *grid* (en función de la disponibilidad de los recursos y de, en el ejemplo, la distribución geográfica de colegios y alumnos); (2) una capa *middleware* con mecanismos de coordinación y consistencia que den soporte a la replicación e, incluso, a la migración de los componentes software.

Las secciones siguientes se dedican a describir el estado actual del trabajo de los autores en lo referente a estos dos aspectos.

3. Planificación de Aplicaciones CSCL basadas en Componentes

La planificación de aplicaciones puede mejorar el rendimiento y, en consecuencia, la colaboración [Bote03b]. Más concretamente, en el caso de las aplicaciones CSCL basadas en componentes, un nivel de rendimiento adecuado se puede alcanzar (si ello es posible) distribuyendo adecuadamente (es decir, desplegando) los componentes de la aplicación de acuerdo con los recursos disponibles. Puesto que tanto el estado de los recursos como la demanda impuesta por los usuarios varían con el tiempo, el despliegue de los componentes de las aplicaciones debe ser dinámico.

Así pues, un planificador de aplicaciones CSCL debería ser capaz de *seleccionar dinámicamente los recursos* donde se van a desplegar los componentes, *asignar cada réplica de cada componente* (si el planificador decide que un componente se ha de replicar) a uno de los recursos seleccionados y *configurar las comunicaciones* entre los ejemplares de los componentes desplegados. El problema de la planificación de aplicaciones CSCL se puede, por tanto, reformular como la exploración de un espacio de soluciones definido por todas las combinaciones posibles de selección, asignación y configuración para una aplicación CSCL dada. En [Bote03b] los autores presentan un modelo genérico de planificador de aplicaciones CSCL. Para validar dicho modelo se llevaron a cabo simulaciones analíticas de una aplicación de *chat* colaborativa basada en componentes. Los resultados de dichas simulaciones mostraron que el rendimiento de la aplicación de *chat*, medida en términos de tiempo de notificación de nuevos mensajes, mejora hasta en un 60%, en los casos en que los componentes son distribuidos por el planificador propuesto, con respecto a las situaciones de distribución tradicional sin planificación previa (distribución aleatoria, distribución nula, etc...). Aunque estos son únicamente resultados preliminares, de ellos se puede inferir que el uso de planificadores adecuados es un aspecto clave si se desea sacar provecho del uso de infraestructuras *grid* como soporte de aplicaciones CSCL basadas en componentes. No obstante, el uso de estos planificadores impone ciertos requisitos a esas infraestructuras *grid* subyacentes. Estos requisitos se analizan en la siguiente sección.

4. El problema del Despliegue Dinámico en Grids basados en OGSA

OGSA [Foster02b] se ha convertido en el estándar de facto para las infraestructuras *grid* actuales. De hecho, es la arquitectura adoptada por la nueva versión (3.0) de *Globus Toolkit* (GT) [Globus03]: el entorno de desarrollo de *grids* más ampliamente utilizado en la comunidad de investigación de tecnologías *grid*. Los servicios de OGSA, denominados *Servicios Grid*, son un tipo especial de *Servicios Web* que permiten a las organizaciones la provisión de acceso externo implícito a sus recursos de computación (y no solamente a sus datos, como sucede actualmente con los servidores Web). Los Servicios Grid ocultan la forma en que las organizaciones implementan la funcionalidad de los servicios que ofrecen y los recursos que emplean

para ello. Con este modelo, OGSA aboga por un escenario en que las aplicaciones *grid* ofrecen su funcionalidad mediante la composición o concatenación de accesos a Servicios Grid proporcionados por organizaciones diferentes y, posiblemente, dispersas. Por lo tanto, el problema de la *selección de recursos* inherente a la computación *grid* se desplaza, en cierto modo, hacia el problema de la *selección de Servicios Grid*. Esta nueva situación, más orientada a los servicios, implica que haya que replantearse el problema de la planificación de aplicaciones CSCL descrito en la sección anterior.

Aunque OGSA considera la posibilidad de usar componentes software para implementar la funcionalidad ofrecida por los Servicios Grid, hay importantes diferencias entre un Servicio Grid y un componente software [Szyperski03]. En lo referente al problema de la planificación, la diferencia más importante es que un componente software es parte integral de la aplicación a la que pertenece, mientras que un Servicio Grid es accedido y utilizado por una aplicación (pero no pertenece a ella). Este hecho implica que un planificador potencial CSCL no sería capaz de decidir dónde un servicio utilizado por una aplicación debe ser ejecutado o, incluso, cuál es el número de réplicas de dicho servicio que deberían ser distribuidas para mejorar el rendimiento de la aplicación. De hecho, en el caso extremo en que toda la funcionalidad de una aplicación pudiera ser ofrecida mediante la concatenación de accesos a Servicios Grid existentes, la responsabilidad del planificador quedaría reducida a la selección de la mejor de las organizaciones (si hay más de una) de entre todas las que ofrecen un mismo servicio. No obstante, y si no se llega a ese caso extremo, el planificador de aplicaciones aún tendría control sobre aquellos componentes de la aplicación cuya funcionalidad no se ofrece por Servicios Grid de terceros.

Así pues, surge el problema de que, aunque un planificador de aplicaciones CSCL pudiera decidir cómo distribuir los componentes software correspondientes, OGSA (de acuerdo con su visión de un escenario orientados al acceso a servicio) no ha definido ningún mecanismo estándar para desplegar dinámicamente los componentes software sobre los nodos *grid* ofrecidos por las organizaciones. Por lo tanto, parte del esfuerzo investigador de los autores se dirige hacia la definición de un "Servicio Dinámico de Despliegue CSCL": un Servicio Grid ofrecido por organizaciones capaces de albergar la ejecución de componentes software de aplicaciones CSCL y de usar los planificadores asociados. El uso de este servicio redundaría en el aumento potencial del rendimiento de las aplicaciones CSCL gracias a una asignación de los recursos *grid* más apropiada. Esa asignación sería, por naturaleza, dinámica para que las aplicaciones se adaptaran a los cambios en la disponibilidad de los recursos y en la demanda de los usuarios de las aplicaciones. Los autores están actualmente trabajando en la definición de la interfaz del servicio y su implementación con el objetivo de desplegar componentes CSCL de tecnología *Enterprise Java Beans* en nodos *Globus Toolkit 3.0*. La definición de la interfaz se basa en la versión actual de la API del servicio de despliegue estándar para la plataforma J2EE [Searls02]. La implementación del servicio está reutilizando ideas de las propuestas actuales de OGSA para el soporte de Servicios Grid mediante componentes *Enterprise Java Beans*.

5. Conclusiones y Trabajo Futuro

Este artículo ha presentado una serie de argumentos a favor de la fusión de las tecnologías *grid*, CSCL e ISBC. Además, se ha presentado y discutido un posible escenario de aplicación de la mencionada fusión. También, y de acuerdo con los resultados de simulaciones preliminares, se ha mostrado cómo el uso de planificadores específicos de aplicaciones CSCL puede contribuir a mejorar el rendimiento de las mismas. Por último, se han descrito los pasos, dados hasta ahora, encaminados hacia la definición y validación de un servicio, compatible con OGSA, de despliegue dinámico de aplicaciones CSCL basadas en componentes. Los planes de trabajo futuro en esta línea incluyen el desarrollo de un planificador para la aplicación CSCL descrita en la sección 2 así como la implementación del servicio de despliegue dinámico propuesto.

Agradecimientos

Este trabajo de investigación ha sido financiado por los proyectos TIC2002-04258-C03-02, TIC2000-1054 y VA 117/01.

Referencias

[Bote03a] Bote-Lorenzo, M.L., Dimitriadis, Y.A., and Gómez-Sánchez, E., 'Grid characteristics and uses: a grid definition'. Proceedings of the 1st European Across Grids Conference, en CD. Santiago de Compostela, España, 2003.

[Bote03b] Bote-Lorenzo, M.L., Dimitriadis, Y., Gómez-Sánchez, E., Asensio-Pérez, J.I.,

Vaquero-González, L.M., and Vega-Gorgojo, G., 'Asignador de recursos grid para aplicaciones CSCL basadas en componentes'. Actas de las IV Jornadas de Ingeniería Telemática, JITEL 2003, pp. 575-576. Gran Canaria, España, 2003.

[Foster98] Foster, I., 'Computational grids'. En Foster, I. and Kesselman, C. (eds), The Grid: blueprint for a future computing infrastructure, pp. pp. 15-52. Ed. Morgan Kaufmann Publishers. (1998).

[Foster02a] Foster, I., Kesselman, C., Nick, J.M., and Tuecke, Steven. 'Grid services for distributed system integration'. Computer, 35(6), pp. 37-46. (2002).

[Foster02b] Foster, Ian, Kesselman, Carl, Nick, J.M., and Tuecke, S., 'The physiology of the Grid: an Open Grid Services Architecture for distributed systems integration', Technical Report, Open Grid Service Infrastructure WG, Global Grid Forum. (2002).

[Furmento02] Furmento, N., Mayer, A., McGough, S., Newhouse, S., Field, T, and Dalington, J., 'ICENI: optimisation of component applications within a grid environment'. Parallel Computing, 28, pp. 1753-1772. (2002).

[Globus03] The Globus Project. Disponible en http://www.globus.org. Última visita en Septiembre 2003.

[Honkela00] Honkela, T., Leinonen, T., Lonka, K., and Raike, A., 'Self-organizing maps and constructive learning'. Proceedings of the International Conference on Educational Uses of Communication and Information Technologies, ICEUT 2000, pp. 339-343. Beijing, China. (2000).

[Koschmann96] Koschmann, T., 'Paradigm shift and instructional technology'. En Koshmann, T. CSCL (ed.), Theory and Practice of an emerging paradigm, pp. 1-23. Ed. Lawrence Erlbaum. (1996).

[Martínez03] Martínez, A., 'Método y modelo para el apoyo computacional a la evaluación en CSCL', Tesis Doctoral, Universidad de Valladolid. (2003).

[Roschelle98] Roschelle, J., Kaput, J., Stroup, W., and Kahn, T.M., 'Scalable integration of educational software: exploring the promise of component architectures', Journal of Interactive Media in Education, 98(6). (1998).

[Searls02] Searls, R., 'Java 2 Enterprise Edition Deployment API Specification, version 1.0', Sun Microsystems. (2002).

[Szyperski03] Szyperski, C., 'Component Technology - What, Where, and How', Actas de la 25ª International Conference on Software Engineering, ICSE 2003. Portland, Oregón, EEUU, 2003.