# Una solución basada en HTCondor para aprovechar la disponibilidad de recursos efímeros

Sergio Hernández, Javier Fabra, Joaquín Ezpeleta, Pedro Álvarez

Instituto de Investigación en Ingeniería de Aragón (I3A)
Departamento de Informática e Ingeniería de Sistemas
Universidad de Zaragoza, España
{shernandez,jfabra,ezpeleta,alvaper}@unizar.es

**Resumen** Clusters, grids y, más actualmente, clouds, representan las infraestructuras de computación dedicada más utilizadas por científicos e investigadores. Sin embargo, existen otras alternativas cuya principal función no es la computación y que pueden ser útiles para la resolución de problemas computacionalmente costosos. En el ámbito académico existen una gran cantidad de recursos que, durante gran parte del día, permanecen encendidos y desaprovechados, de forma que se podrían utilizar para tareas computacionales durante el tiempo que permanecen infrautilizados. Con este objetivo, en este artículo se propone la utilización del *middleware* HTCondor para aprovechar estos *recursos efímeros* existentes en nuestro departamento, así como su integración en un *framework* de computación distribuida desarrollado anteriormente y que en la práctica estamos utilizando y extendiendo para resolver problemas complejos. Esta integración se ha realizado mediante la adición de un componente en el *framework* que, basándose en el horario de reserva de los recursos integrados, es capaz de recomendar los más adecuados para la ejecución de cada trabajo. Finalmente, se ha utilizado el entorno de Amazon EC2, simulando el entorno real de ejecución, para configurar la nueva infraestructura y probar el nuevo componente.

**Palabras clave:** Recursos efímeros, computación en la nube, planificación de recursos, heterogeneidad, HTCondor.

## 1 Introducción

La investigación en ciertas áreas científicas tiene unos requisitos computacionales muy elevados [1]. Como respuesta a estas necesidades, se han utilizado infraestructuras de computación de altas prestaciones que proporcionan al usuario un elevado número de recursos, como ocurre en el caso de los *clusters* y los *grids* [2]. Recientemente, se ha propuesto el uso de la virtualización para aumentar las capacidades de este tipo de infraestructuras, dando lugar a la aparición del *cloud computing* [3, 4]. Como punto en común, estas infraestructuras proporcionan un elevado conjunto de recursos dedicados de forma exclusiva a tareas de computación.

Sin embargo, tanto en los entornos académicos como laborales existen una gran cantidad de recursos que no se utilizan durante gran parte del tiempo que permanecen encendidos (ordenadores de trabajo del personal, laboratorios de docencia, etc.). De esta forma, durante el tiempo en el que los recursos están infrautilizados, pueden ser usados para ejecutar tareas computacionales sin que afecte a los usuarios de dichos recursos. Estos recursos, denominados *recursos efímeros*, pueden ser utilizados como una infraestructura de computación alternativa que podría sustituir o complementar a los clusters, grids y clouds [5].

Para aprovechar este tipo de recursos, existen diferentes *middleware* de gestión. Uno de los más conocidos es BOINC [6], una plataforma que permite utilizar ordenadores distribuidos por todo el mundo a través de Internet para ejecutar aplicaciones pertenecientes a proyectos científicos. Sin embargo, BOINC está orientado a la ejecución de aplicaciones muy específicas en entornos no confiables. Una alternativa que extiende BOINC añadiendo la posibilidad de organizar los recursos jerárquicamente es el SZTAKI *Desktop Grid* [7]. Este sistema incluye una versión capaz de trabajar en entornos donde los recursos son confiables pero sigue permitiendo tan sólo la ejecución de unas pocas aplicaciones. Otro de los *middlewares* más utilizados es HTCondor [8]. Se trata de un sistema de computación de altas prestaciones que permite aprovechar los ciclos libres de ordenadores desaprovechados de una organización. Su principal ventaja con respecto a los sistemas anteriores es la posibilidad de ejecutar cualquier tipo de trabajo. Finalmente, existen plataformas como OpenStack [9], OpenNebula [10] o Eucalyptus [11] que permiten crear un *cloud* privado proporcionando máquinas virtuales para ejecutar trabajos. Sin embargo, este tipo de soluciones están orientadas a entornos donde los recursos están dedicados.

El objetivo de este artículo consiste en explotar la disponibilidad de los recursos efímeros en nuestro entorno de trabajo para su utilización futura en la resolución de problemas computacionalmente costosas. Para ello, se va a utilizar el *middleware* HTCondor y se va a integrar la nueva infraestructura en un *framework* de computación distribuida, que integra diferentes infraestructuras cluster, grid y cloud, desarrollado previamente por los autores [12]. Para la integración, se propone el desarrollo de un nuevo componente que sea capaz de utilizar información sobre la reserva de los recursos integrados para guiar el proceso de asignación de trabajos a recursos, mejorando la utilización y el aprovechamiento de los mismos. Finalmente, cabe destacar que un elemento clave para esta integración es la utilización de un entorno de computación en la nube, como es el caso de Amazon EC2 [13], para simular el entorno real y facilitar la implementación y configuración de la infraestructura y del nuevo componente.

El resto del artículo se organiza de la siguiente forma. En la sección 2 se describe el *framework* de computación y el entorno de recursos efímeros disponible. La sección 3 muestra el diseño del nuevo componente encargado de gestionar la infraestructura de recursos efímeros. En la sección 4 se detalla cómo se ha llevado a cabo la implementación de la infraestructura, así como su validación utilizando la infraestructura cloud de Amazon. Finalmente, la sección 5 concluye el artículo y reflexiona sobre las futuras líneas de trabajo.

## 2 Antecedentes

En esta sección se presenta el *framework* desarrollado por los autores y se muestra cómo encaja la nueva infraestructura de recursos efímeros en el mismo. Adicionalmente se presentan las características más relevantes de la nueva infraestructura y de su implementación utilizando HTCondor.

### 2.1 Descripción del *framework* de computación distribuida

En [12] presentamos un *framework* de computación orientado a servicios para el despliegue y ejecución de *workflows* científicos en entornos de computación heterogéneos. Este *framework* integra y gestiona de forma transparente un amplio conjunto de recursos de computación y los ofrece al usuario como un único y potente entorno de ejecución. La figura 1 muestra el diseño arquitectural del *framework*, formado por tres capas. La *capa de interfaz de usuario* permite programar las aplicaciones a ejecutar mediante diferentes lenguajes y herramientas. La *capa de ejecución* es la responsable de gestionar el despliegue de las aplicaciones en las infraestructuras de computación integradas. Para ello, incluye dos tipos de componentes que se comunican a través de un *bus de mensajes*: los *componentes de gestión*, que gestionan el ciclo de vida de las aplicaciones; y los *mediadores*, que interactúan con cada infraestructura concreta a través de su *middleware*. Finalmente, la *capa de infraestructuras de computación* engloba los recursos de ejecución utilizados. Actualmente se han integrado un clúster (HERMES) y dos grids (AraGrid y PireGrid), pertenecientes a nuestra Universidad, y la infraestructura cloud de Amazon [13].

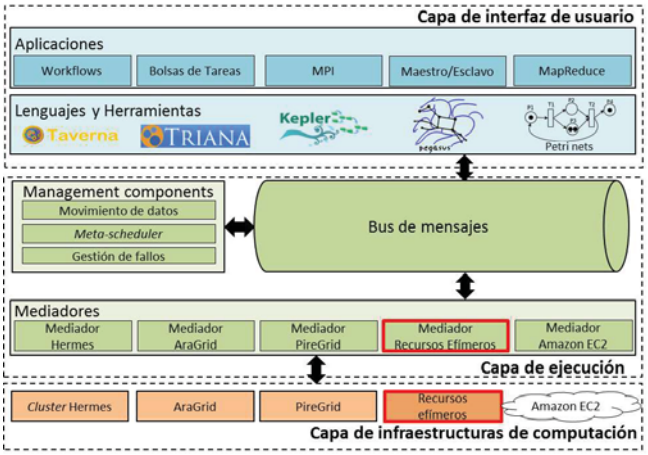


Figura 1: Diseño arquitectural del framework de comptutación.

Como puede observarse en la figura, la integración de esta nueva infraestructura afecta tanto a la capa de ejecución como a la de infraestructuras de computación. Por un lado, debe desarrollarse un nuevo mediador capaz de interactuar con la infraestructura de recursos efímeros. Por otro lado, debe implementarse

la nueva infraestructura de recursos efímeros mediante la instalación y configuración del *middleware* HTCondor en nuestro entorno de trabajo.

En lo referente a este último punto, la figura 2 muestra la topología del escenario de recursos efímeros. Como puede observarse, está compuesta por un servidor, que integrará los servicios de gestión necesarios en HTCondor, y varios laboratorios de docencia de nuestro departamento, en los que se ejecutarán las tareas computacionales. En total, se utilizan 154 recursos de ejecución con un total de 350 procesadores y 494 GB de RAM de los cuales calculamos que cada hora hay aproximadamente un 35% disponibles.

Para la implementación de HTCondor en el entorno disponible se ha utilizado el servidor de gestión para alojar las funciones de gestión (*Central Manager* y Condor Connection Broker, CCB) y envío de trabajos (*Submit*), mientras que los recursos de ejecución albergan la función de ejecución (*Execute*) y también permiten el envío de trabajos de forma remota (*Submit remote*). La figura 3 muestra dicho diseño indicando las funciones configuradas en cada recurso.

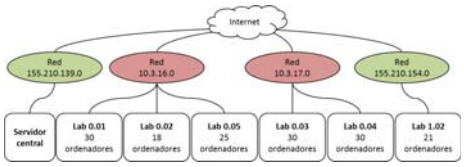


Figura 2: Topología de red de la infraestructura de recursos efímeros.

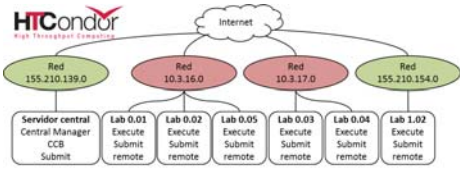


Figura 3: Funciones de HTCondor en la infraestructura de recursos efímeros.

## 3 Diseño de un mediador de gestión de recursos efímeros

Para integrar la infraestructura de recursos efímeros en el *framework* anterior, se ha programado un nuevo mediador capaz de gestionar este tipo de recursos e interactuar con HTCondor. Este mediador es capaz de obtener los trabajos a ejecutar procedentes del *framework*, someter los trabajos para su ejecución, monitorizar el estado de los recursos y los trabajos en ejecución, recuperar los resultados de los trabajos finalizados y proporcionárselos a los usuarios a través del *framework* y gestionar el horario de ocupación de los recursos.

La idea general y principal contribución de este nuevo mediador es la de utilizar información acerca de la reserva de los recursos para guiar el proceso de planificación (asignación de trabajos a recursos computacionales). Esto es posible ya que, en este caso, los recursos integrados se utilizan para la realización de prácticas por los alumnos por lo que se dispone de información sobre los horarios de reserva de dichos recursos. El nuevo mediador utiliza esta información para seleccionar los recursos más adecuados para la ejecución de cada trabajo de acuerdo a su disponibilidad. En cualquier caso, no se descarta la ejecución de trabajos en recursos que estén reservados si el resto de recursos están ocupados o no disponibles. Esta decisión se debe a que el hecho de que un laboratorio esté reservado no implica que todos sus recursos vayan a ser ocupados y, por tanto, es posible que haya recursos libres que puedan ser utilizados para ejecutar tareas.

Adicionalmente, se plantea la utilización de información histórica que complemente a la información sobre la ocupación de los recursos. En cuanto a este tipo de información estamos interesados en guardar información tanto de los trabajos ejecutados (duración, recurso de ejecución, número de expulsiones, etc.) como de los propios recursos (tiempo medio libre y ocupado, instantes en los que el recurso pasa de estar libre a ocupado y viceversa, etc.). Con este registro histórico, se pretende poder mejorar en un futuro el proceso de planificación.

La figura 4 muestra el diseño arquitectural del mediador de recursos efímeros. En la misma, los componentes que afectan de forma exclusiva a la gestión de recursos efímeros aparecen resaltados. Antes de detallar los aspectos más relevantes referentes a estos componentes, vamos a describir el ciclo de vida base de los trabajos (correspondiente únicamente a los componente sin resaltar):

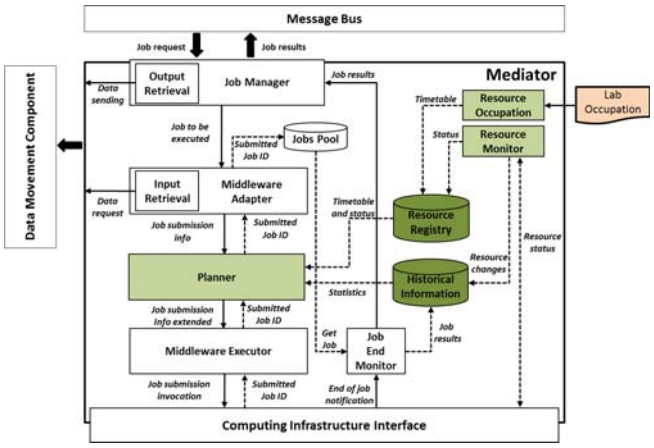


Figura 4: Diseño arquitectural del mediador de gestión de recursos efímeros.

En primer lugar, el *Job Manager* obtiene los trabajos a ejecutar del *framework* a través del *Message Bus* y los envía al *Middleware Adapter*. Este componente los traduce de un lenguaje de descripción estándar utilizado por el *framework* a un lenguaje que sea interpretable por el *middleware* de gestión de la infraestructura, en este caso HTCondor; y se encarga de mover a la infraestructura los datos de entrada necesarios para la ejecución del trabajo. Una vez realizada dicha traducción y que se han movido los datos de entrada necesarios, el trabajo es enviado al *Middleware Executor* (el uso del *Planner* es, por tanto, opcional) y el identificador asignado por la infraestructura es almacenado en el *Jobs Pool* junto con la descripción del trabajo. Cuando el trabajo finaliza su ejecución, el *Job End Monitor* lo detecta, consulta el *Jobs Pool* y se lo indica al *Job Manager* el cual recupera los resultados del trabajo y los envía al *framework* vía el *Message Bus* para que se pueda notificar al usuario.

Una vez mostrado el ciclo de vida del mediador, describimos de forma individual los componentes dedicados a la gestión específica de recursos efímeros:

– *Resource Monitor*: este componente monitoriza de forma periódica el estado de los recursos (libre, ocupado, eliminado, etc.).

- *Resource Occupation*: este componente es el encargado de gestionar la información sobre la ocupación de los recursos. Recibe un fichero de texto con información de la ocupación de los recursos, obtenido de la página web de nuestro departamento, lo traduce y lo almacena en el *Resource Registry*.
- *Resource Registry*: este almacén de información se encarga de gestionar la información referente a los recursos. Contiene el estado actual de los recursos (procedente del *Resource Monitor*) e información acerca de su ocupación (proporcionada por el *Resource Ocuppation*).
- *Historical Information*: este almacén de información se encarga de almacenar los logs con información histórica. Contiene información sobre los trabajos ejecutados (procedente del *Job End Monitor*) y eventos referentes a la adición/eliminación de los recursos integrados en el entorno de computación (procedente del *Resource Monitor*) . También permite obtener estadísticas que puedan utilizarse durante la planificación.
- *Planner*: este componente permite seleccionar el recurso o grupo de recursos en los que se recomienda ejecutar un trabajo. El componente recibe la información del trabajo a someter y le añade el recurso o grupo de recursos recomendados. Para ello, puede utilizar diferentes fuentes de información como son: el estado actual de los recursos y la ocupación prevista para los mismos (obtenidos del *Resource Registry*) y estadísticas de uso (proporcionadas por la componente *Historical Information*). Actualmente sólo utiliza información acerca de la ocupación de los recursos.

## 4 Validación de la solución utilizando recursos cloud

En esta sección se describe cómo se ha utilizado el entorno de computación en la nube de Amazon para implementar y configurar la infraestructura de recursos efímeros. Además, se presenta una prueba de concepto que simula una ejecución real con el objetivo de validar el mediador desarrollado y mostrar sus ventajas.

La razón por la que se ha utilizado un entorno virtual, como Amazon EC2 para la implementación, configuración y validación de la solución es doble: por una parte, evita una serie de problemas que nos hubiésemos encontrado si hubieses utilizado directamente el entorno real, como problemas de acceso a los recursos, falta de permisos de administración, disponibilidad limitada de los recursos, pérdida de control en la configuración del sistema, etc.; por otra parte, proporciona ventajas adicionales, ya que facilita la administración del sistema y de los recursos, la configuración de los mismos, la definición y validación de diferentes escenarios de error, el acceso a los recursos en todo momento, etc.

### 4.1 Despliegue y configuración de HTCondor

Para el despliegue y configuración del *middleware* HTCondor, que gestiona la infraestructura de recursos efímeros, se ha definido un entorno simulado lo más parecido al entorno real mostrado en la figura 3. Esto simplifica la labor de implantar posteriormente el sistema en el entorno real ya que, una vez

configurado y validado el entorno simulado, sólo es necesario replicar dicha configuración en el entorno real realizando cambios mínimos en los ficheros de configuración. De esta forma, la figura 5 muestra la topología del entorno simulado creada en Amazon EC2. Como puede observarse, al comparar este escenario con el escenario mostrado en la figura 3, ambos escenarios son análogos.

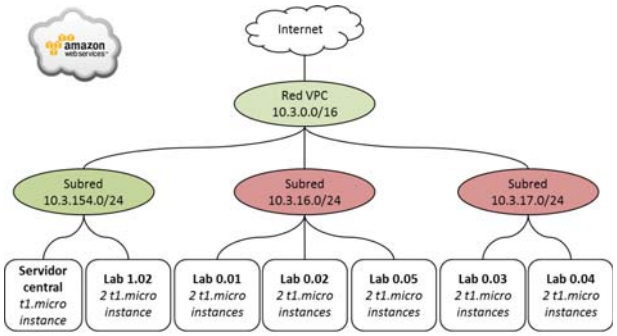


Figura 5: Topología del escenario de recursos efímeros simulado en Amazon EC2.

El aspecto más importante en la definición del escenario simulado es la topología de red, ya que se deben distinguir y separar las redes públicas de las privadas. Para simular la configuración de red se ha utilizado la función VPC (*Virtual Private Cloud*) de Amazon que permite crear redes virtuales privadas, con varias subredes y una puerta de enlace con salida a Internet. Utilizando esta función y configurando adecuadamente los grupos de seguridad (similar a un *firewall*) de las máquinas creadas, se ha replicado una topología de red como la existente en nuestro departamento universitario, creando una subred en el entorno simulado por cada subred existente en el entorno real. La única diferencia reseñable es que en el entorno de Amazon se utiliza una sola red pública, en lugar de las dos existentes en el entorno real, pero este hecho no tiene ninguna implicación a la hora de configurar HTCondor.

Para simular los recursos de los laboratorios se han utilizado instancias *t1.micro*, al ser las más baratas ofrecidas por Amazon, creadas dentro de la subred adecuada. Además, para facilitar el despliegue posterior en el entorno real, se los recursos se han instanciado con el mismo sistema operativo utilizado por los recursos de nuestro departamento (CentOS 6.4 de 64 bits).

Para el despliegue y configuración de HTCondor en el entorno creado se han utilizado diferentes AMIs (*Amazon Machine Images*) que permiten gestionar y replicar de forma sencilla los componentes del sistema. Concretamente, se han creado tantas AMIs como tipos de recursos diferentes hay en el sistema: una para el nodo central de gestión, otra para los recursos de ejecución en redes públicas y otra para los recursos de ejecución en redes privadas.

Finalmente, una vez probado el sistema en Amazon EC2, se ha procedido a la implantación del mismo en el entorno real. Para ello, tan sólo ha sido necesario instalar HTCondor en los recursos del departamento y replicar la configuración usada en Amazon con pequeñas modificaciones en los ficheros de configuración.

### 4.2 Validación del mediador de recursos efímeros

Para la validación del mediador se ha utilizado el entorno creado en Amazon (figura 5), lanzando una máquina virtual por laboratorio. La validación se ha realizado sobre el entorno simulado y no sobre el real para garantizar que las pruebas se realizaban en las mismas condiciones y eran comparables.

En cuanto a la ocupación de los recursos, se han tomado cuatro horas que consideramos suficientemente representativas de una situación habitual de ocupación. La tabla 1 muestra dicha ocupación indicando para cada hora y laboratorio si los recursos están libres u ocupados. Para simular el horario, los experimentos se han configurado para que el cambio de los recursos de libre a ocupado se produzca 5 minutos después del comienzo de la hora y el paso de ocupado a libre se produzca 5 minutos antes del fin de la hora.

Tabla 1: Horario de ocupación de los laboratorios durante los experimentos

| | **Lab 0.01** | **Lab 0.02** | **Lab 0.03** | **Lab 0.04** | **Lab0.05** | **Lab 1.02** |
|---|---|---|---|---|---|---|
| **1ª hora** | Ocupado | Libre | Ocupado | Libre | Libre | Libre |
| **2ª hora** | Ocupado | Libre | Ocupado | Ocupado | Libre | Ocupado |
| **3ª hora** | Ocupado | Ocupado | Libre | Ocupado | Libre | Ocupado |
| **4ª hora** | Libre | Ocupado | Libre | Libre | Libre | Ocupado |

En cuanto a los experimentos, se ha realizado una prueba de concepto probando dos configuraciones, una con *tareas largas* (50 minutos) y otra con *tareas cortas* (10 minutos), con el fin de valorar la influencia de la duración de las tareas ejecutadas y evaluar la efectividad del mediador de recursos efímeros ante diferentes tipos de tareas. En ambos casos, cada hora se lanza un número de tareas que equivale a un tiempo total de ejecución de 100 minutos.

La tabla 2 muestra los resultados obtenidos en los experimentos. En ella podemos observar el número total de tareas ejecutadas, la duración de cada una, la cantidad de tareas sometidas simultáneamente al comienzo de cada hora y los resultados en cuanto al número de tareas que han sido expulsadas. El número de expulsiones es el número de veces que una tarea ha tenido que ser reubicada al ejecutarse inicialmente en un recurso que posteriormente fue ocupado.

Tabla 2: Resumen de los resultados de la experimentación

| **Tipo de experimento** | **Tareas largas** | | **Tareas cortas** | |
|---|---|---|---|---|
| **Nº de tareas** | 8 | | 40 | |
| **Duración** | 50 | | 10 | |
| **Nº de tareas simultáneas** | 2 | | 10 | |
| **Uso de planificador** | Sí | No | Sí | No |
| **Nº de expulsiones** | 0 | 3 | 5 | 5 |

En el experimento con tareas largas, la utilización del planificador consigue evitar la aparición de expulsiones ya que el éste garantiza que los trabajos se ejecutan en recursos que van a estar libres. En cambio, si no se usa el planificador existe la posibilidad de que se ejecuten trabajos en recursos que posteriormente estarán ocupados ya que la asignación la realiza HTCondor de forma aleatoria.

Sin embargo, en el experimento con tareas cortas se obtienen los mismos resultados con y sin planificador. Esto era lo esperado ya que que el número de tareas a ejecutar cada hora es mayor que el número de recursos libres. De esta forma, aunque las primeras tareas se envían a los recursos libres indicados por el planificador, el resto se envían a recursos que posteriormente pasarán a estar ocupados, y, por tanto, serán expulsadas. Este comportamiento es el esperado ya que el sistema intenta aprovechar el máximo número de recursos ya que el hecho de que un recurso esté reservado no garantiza que vaya a ser ocupado y podría permanecer libre.

Los resultados de esta prueba de concepto nos han permitido validar el funcionamiento del mediador de recursos efímeros y también han permitido extraer algunas conclusiones que nos han permitido establecer líneas futuras de trabajo. La principal conclusión es que el mediador es capaz de recomendar los recursos más adecuados para ejecutar tareas en situaciones en las que se conoce con seguridad la ocupación futura de los recursos. Además, se ha observado la importancia de elegir un tamaño adecuado de tarea.

En general, es deseable ejecutar tareas cortas para aprovechar todos los recursos disponibles y porque en caso de que un recurso pase a ser ocupado, la pérdida de tiempo es potencialmente menor. Sin embargo, la ejecución de tareas más largas en recursos que sabemos que van a estar disponibles puede beneficiarnos al evitar la aparición de fallos. Por tanto, parece que el comportamiento óptimo sería utilizar un número de tareas igual al número de recursos libres en cada momento, si bien queremos estudiar esto en el futuro.

## 5 Conclusiones y trabajo futuro

En este artículo se han integrado los recursos académicos disponibles en nuestro departamento dentro de un framework de computación, con el fin de utilizarlo como infraestructura de cómputo aplicada a la resolución de problemas complejos. Para ello, se ha desplegado el *middleware* de gestión de recursos HTCondor sobre un entorno simulado en el *cloud* de Amazon. Posteriormente, tras realizar una configuración adecuada del entorno, se ha implantado el sistema en el entorno físico real. El uso de un *cloud* público para desplegar y configurar HTCondor simulando el entorno físico real nos ha aportado varias ventajas en cuanto a la gestión y administración de los recursos y ha permitido disminuir el tiempo necesario para desplegar el sistema en comparación con el tiempo que hubiese sido necesario para llevarlo a cabo en el sistema real.

Para la integración de esta nueva infraestructura en el *framework* de computación previamente desarrollado por los autores se ha extendido uno de sus componentes, incluyendo un planificador que tiene en cuenta el horario de ocupación de los recursos. De esta forma, se puede guiar la asignación de trabajos en diferentes recursos minimizando la expulsión de trabajos en ejecución, mejorando el aprovechamiento de los recursos disponibles y reduciendo el tiempo total de ejecución de problemas computacionalmente costosos.

Como principal línea de trabajo futuro se plantea el estudio de diferentes algoritmos de planificación. Actualmente, el planificador sólo hace uso del horario de ocupación de los recursos, pero también se pretende utilizar el histórico de disponibilidad de los recursos ya que el hecho de que un recurso esté reservado no implica que vaya a ser utilizado y el hecho de que un recurso no esté reservado no implica que esté libre, por lo que la información histórica puede ayudar a localizar y tratar estas situaciones. Finalmente, también hemos observado que una política que puede mejorar el tiempo de ejecución total es ejecutar un número de tareas igual al número de recursos disponibles, de esta forma, en el futuro pensamos investigar la posibilidad de que el mediador sea capaz de agrupar varias tareas de forma dinámica en función del estado de los recursos y la ocupación prevista.

## Agradecimientos

Este trabajo ha sido financiado parcialmente por el proyecto JIUZ-2013-TEC-03 y la Red Temática Científico-Tecnológica en Ciencias de los Servicios (TIN2011-15497-E) financiada por el Ministerio de Economía y Competitividad de España. Los autores también quieren agradecer a Marcos Molina su colaboración en la implementación del sistema presentado en este artículo.

## Bibliografía

1. Taylor, I.J., Deelman, E., Gannon, D., Shields, M.: Workflows for e-Science. Springer-Verlag London Limited (2007)
2. Foster, I., Kesselman, C.: The Grid 2: Blueprint for a New Computing Infrastructure. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA (2003)
3. Vaquero, L.M., Rodero-Merino, L., Caceres, J., Lindner, M.: A break in the clouds: towards a cloud definition. SIGCOMM Comput. Commun. Rev. **39**(1) (2008) 50–55
4. Foster, I., Zhao, Y., Raicu, I., Lu, S.: Cloud computing and grid computing 360-degree compared. In: Proceedings of the Grid Computing Environments Workshop. GCE '08 (2008) 1–10
5. Kondo, D.: Scheduling task parallel applications for rapid turnaround on desktop grids. PhD thesis, University of California, San Diego (2005)
6. BOINC. `http://boinc.berkeley.edu/` Accedido 25 Abril 2014.
7. Kacsuk, P., Kovacs, J., Farkas, Z., Marosi, A.C., Gombas, G., Balaton, Z.: Sztaki desktop grid (szdg): a flexible and scalable desktop grid system. Journal of Grid Computing **7**(4) (2009) 439–461
8. HTCondor. `http://research.cs.wisc.edu/htcondor/` Accedido 25 Abril 2014.
9. OpenStack Software. `http://www.openstack.org/` Accedido 25 Abril 2014.
10. OpenNebula Software. `http://www.opennebula.org/` Accedido 25 Abril 2014.
11. Eucalyptus Software. `https://www.eucalyptus.com/` Accedido 25 Abril 2014.
12. Fabra, J., Hernández, S., Ezpeleta, J., Álvarez, P.: Solving the interoperability problem by means of a bus. an experience on the integration of grid, cluster and cloud infrastructures. J. Grid Comput. (2013) 1–25
13. Amazon Elastic Compute Cloud (Amazon EC2). `http://aws.amazon.com/ec2/` Accedido 25 Abril 2014.