Actas de las Jenui, vol. 4. 2019.
Páginas: 167-174

Gestión elástica en la nube de recursos computacionales para actividades docentes: caso de uso en el Diseño de Sistemas Digitales

Sergio López Huguet
Instituto de Instrumentación para Imagen Molecular (I3M)
Universitat Politècnica de València
serlohu@upv.es

David de Andrés Martínez
ITACA
Universitat Politècnica de València
ddandres@disca.upv.es

J. Damián Segrelles Quilis
Instituto de Instrumentación para Imagen Molecular (I3M)
Universitat Politècnica de València
dquilis@dsic.upv.es

Resumen

Hoy en día, los entornos computaciones son clave en las titulaciones STEM, dado que mejoran significativamente sus procesos de enseñanza-aprendizaje. Por ello, las universidades buscan la racionalización de las infraestructuras de cómputo que dan soporte a dichos entornos, con el objetivo de abaratar su coste (adquisición y mantenimiento). En este sentido, la nube está empujando a las universidades a moverse hacia un modelo de "pago por uso" de las infraestructuras, por lo que se requieren nuevos e innovadores servicios que gestionen dicho modelo de forma eficiente. En este artículo se presenta el recurso docente *Cluster Elasticity Manager (CEM)* cuyo fin es gestionar eficientemente en la nube las infraestructuras que dan soporte a entornos computacionales en actividades educativas que necesitan de elasticidad. Para el diseño de dicho recurso, se ha tomado como punto de partida un escenario de referencia donde se desarrollan actividades educativas diseñadas para formar alumnos en el Diseño de Sistemas Digitales. Finalmente, se presenta un análisis de tiempos en el despliegue de los entornos computacionales requeridos en el escenario de referencia planteado a través de CEM.

Abstract

Nowadays, computing environments are key in STEM degrees, since they significantly improve their teaching-learning processes. For that, universities seek to rationalize the computing infrastructures that support these environments, with the aim of lowering their cost (acquisition and maintenance). In this sense, the cloud is pushing universities towards a "pay-per-us" model of infrastructures, which is why new and innovative services are required to manage this model efficiently. This article presents the teaching resource *Cluster Elasticity Manager (CEM)*, which objective is to efficiently manage in the cloud the elastic infrastructures that support computer environments in educational activities. For the design of this resource, a reference scenario has been analysed as a starting point, where educational activities designed to train students in the Design of Digital Systems are performed. Finally, an analysis of time is presented in the deployment of the computational environments required in the reference scenario proposed through CEM.

Palabras clave

Cloud, Virtualización, Sistemas Digitales.

1. Introducción

En un aprendizaje basado en competencias como el que impulsa el Espacio Europeo de Educación Superior (EEES), los Entornos Computaciones (EC) como herramientas de soporte a las Actividades Educativas (AE) que desarrollan y evalúan competencias son clave, y más si cabe en el marco de las enseñanzas STEM (Science, Technology, Engineering and Mathematics). Por ello, encontramos multitud de EC en forma de simuladores, laboratorios virtuales o paquetes software específicos que sirven como herramienta para llevar a cabo AE en áreas de conocimiento relacionadas con la ciencia [5, 8], tecnología [2, 4], Ingeniería [10] y Matemáticas [12, 14] y que han ayudado a mejorar los procesos de enseñanza aprendizaje en multitud de titulaciones.

Actualmente, la nube está empujando a las universidades hacia un modelo de "pago por uso" de las infraestructuras de cómputo que dan soporte a los EC, por lo que se requieren de nuevos e innovadores servicios que las gestionen de forma eficiente. La nube, como

proveedor de infraestructura en la educación superior es ya un hecho, y es por ello que existen numerosos trabajos que muestran las ventajas que ofrece esta tecnología aplicada a la educación [6].

Los autores de este trabajo, ya han demostrado a través de varias experiencias los beneficios obtenidos desde el punto de vista de los centros educativos [15], profesores [16] y estudiantes [15]. Entre dichos beneficios se encuentra la capacidad de la nube de proporcionar elasticidad y escalabilidad de las infraestructuras de cómputo que dan soporte a los CE. Sin embargo, en ninguna de esta experiencias se presentó un servicio capaz de gestionar dicha elasticidad de forma automática, siendo gestionada directamente por el profesorado de una forma manual. Por ello, en este artículo, se presenta un recurso docente que gestione de manera automática, en función de la demanda de recursos en tiempo real por parte de los usuarios, la elasticidad de los recursos computacionales en la nube para la puesta en marcha de AE.

El artículo se organiza de la siguiente manera. Primero, se describe la asignatura *Diseño de Sistemas Digitales (DSD)* impartida en el Grado de Ingeniería Informática (GII) de la Escuela Técnica Superior de Ingeniería Informática (ETSINF) de la Universitat Politècnica de València (UPV), dado que será el escenario de referencia a partir del cual se extraerán los requisitos del recurso docente. A continuación, se enumeran dichos requisitos y la arquitectura propuesta del recurso. Finalmente, se muestra su interfaz de usuario y un análisis de tiempos del despliegues de los EC requeridos para la asignatura presentada como escenario de referencia.

2. Escenario de Referencia

La asignatura *Diseño de Sistemas Digitales (DSD)* se imparte en el segundo semestre del tercer curso del Grado en Ingeniería Informática (GII), dentro de la intensificación de Ingeniería de Computadores, de la Universitat Politècnica de València (UPV). La matriculación media durante los últimos seis cursos ha sido de 39 alumnos.

Se trata de una asignatura eminentemente práctica, donde los alumnos deben desarrollar las competencias necesarias para modelar sistemas digitales por medio de lenguajes de descripción de hardware (HDL), verificar su correcto funcionamiento e implementarlos utilizando dispositivos lógicos programables de tipo Field-Programmable Gate Array (FPGA). Para facilitar la adquisición de estas competencias, los alumnos realizan diversas actividades: i) en el aula, como un Puzzle the Aronson [1] acerca de la arquitectura interna de las FPGA o técnicas de desarrollo de la creatividad, innovación y emprendimiento, como S.C.A.M.P.E.R. [11], para conocer el uso actual de estos dispositivos; ii) fuera del aula, como la lectura de artículos relativos a los algoritmos utilizados para sintetizar e implementar sistemas digitales en FPGA, como Simulated Annealing [13] o PathFinder [9], y la respuesta a diversas preguntas para facilitar la asimilación de los contenidos; y iii) en el laboratorio informático, donde tienen lugar los seminarios y prácticas de la asignatura, de tal forma que los alumnos aprenden VHDL [7] (el HDL más utilizado en Europa) y deben desarrollar y entregar una serie de sistemas digitales de complejidad incremental.

Este trabajo se centra en las actividades de laboratorio, concretamente en el diseño e implementación de sistemas digitales utilizando FPGA, el cual va ligado al flujo de diseño *semicustom* (modelado, síntesis, implementación y verificación en cada una de las etapas), por lo que el laboratorio informático debe incorporar todo el instrumental software y hardware específico necesario para facilitar la adquisición de las competencias necesarias para el manejo con soltura de este flujo. Así pues, el laboratorio dispone de 22 puestos de trabajo (más otro para el profesor) que constan de un PC con un procesador Intel de 4 núcleos y 8 GB de memoria RAM, con sistema operativo Linux. En estas máquinas se encuentra instalado el entorno de desarrollo que soporta el flujo de diseño *semicustom* para FPGAs de Xilinx, Vivado Design Suite - HLx Editions [1], que puede descargarse de forma gratuita de la web del fabricante. Asimismo, se dispone de 10 placas de prototipado Nexys A7 de Digilent. Estas placas incorporan una FPGA Artix-7 XC7A100T-1CSG324C de Xilinx para implementar los diferentes circuitos diseñados en las prácticas de la asignatura, y diferentes puertos y periféricos, como conectores USB, VGA y Ethernet, e interruptores, botones, *displays* de 7 segmentos y leds, para poder interactuar con el circuito. El acceso al laboratorio y, por tanto, el uso del software y placas de protipado está limitado a las 15 sesiones de 90 minutos que se destinan a aprender a utilizar el entorno de desarrollo, aprender el lenguaje VHDL, y realizar las 4 prácticas entregables.

2.1. Identificación de las necesidades

Las necesidades detectadas en el escenario de referencia son las siguientes:

N01 La actividad del alumnado relativa al instrumental (software específico y hardware que lo soporte) utilizado, que es almacenada en forma de logs, no queda registrada de una forma centralizada, dado que cada estudiante puede utilizar ordenadores diferentes a la hora de realizar las activi-

[1] Página web de Vivado Design Suite - HLxEditions: `https://www.xilinx.com/products/design-tools/vivado.html`

dades presenciales y/o no presenciales. Por tanto, los logs están desubicados y es imposible recuperar esta información por parte del profesorado. El poder recuperar y procesar esta información de cada estudiante es de vital importancia para poder realizar estudios con el objetivo de encontrar correlaciones de su actividad en los ordenadores con su resultados académico de forma que, en un futuro, se pueda ofrecer a los estudiantes información precisa para su autorregulación.

N02 El instrumental utilizado es incompatible con los ordenadores de arquitectura Mac OS X, por lo que una parte significativa de alumnos no pueden realizar o acabar los ejercicios prácticos fuera del laboratorio informático a través de sus propios dispositivos.

N03 El instrumental requiere de unos requerimientos hardware que en algunos casos, no son compatibles o aconsejables con los requisitos de los dispositivos del alumnado, por lo que algunos de ellos no pueden emplear sus dispositivos para realizar las prácticas.

2.2. Requisitos

Dado el escenario de referencia descrito y las necesidades identificadas, desde el punto de vista del uso de proveedores cloud para el despliegue del instrumental hardware y software especifico, el recurso docente que se plantea debe:

R01 Proporcionar recursos computacionales con todo el instrumental necesario para el desarrollo de los boletines de prácticas de una forma ubicua en en cloud. Es decir, Máquinas Virtuales (MV) con los requisitos hardware suficientes (CPU, RAM etc..) para que el software específico funcione de forma apropiada y eficiente.

R02 Posibilitar al alumnado utilizar sus propios dispositivos (portátiles, PCs personales etc.) para interactuar con el instrumental, sin necesidad de que estos tengan en sus dispositivos los recursos apropiados.

R03 Proporcionar al alumnado una disponibilidad 24x7 al instrumental desplegado en la nube.

R04 Ofrecer la interacción con el instrumental a través de los interfaces gráficos que el própio instrumental ofrece.

R05 Escalar de forma elástica y dinámica el instrumental disponible en la nube en función del número de alumnos que estén activos. El alumnado solicitarán bajo demanda el instrumental que requieran para la realización de los ejercicios prácticos, de forma que cuando estos dejen de utilizarse se liberen automáticamente los recursos (MV) asociadas. El objetivo es tener activos el mínimo número de recursos con el objeto de racionalizar de forma eficiente sus costes en los proveedores cloud públicos.

R06 Configurar el instrumental de forma personalizada en el menor tiempo posible, de forma que el tiempo que transcurre desde la petición del instrumental por parte del alumnado, hasta que éste esté disponible en la nube, sea razonable.

R07 Recuperar todos los trabajos y logs generados por el instrumental durante la realización de todos los ejercicios prácticos, de forma que se tenga un registro de toda la actividad generada por el alumnado.

R08 Desplegar los recursos docentes en los proveedores cloud más populares, tanto públicos (Microsoft Azure [2], Amazon Web Services [3] o Google Cloud Platform [4]) como privados (OpenNebula [5] u OpenStack [6]).

R09 Personalizar el sistema a la hora de definir el número de alumnos y alumnas por recurso (MV) desplegado, con el fin de optimizar el coste en los despliegues realizados en proveedores públicos.

R10 Personalizar la elasticidad del sistema, definiendo el tiempo que puede estar un usuario inactivo con recursos asignados o el tiempo puede estar un recurso en estado IDLE (configurado correctamente pero sin estudiantes asignados) antes de ser suspendido o eliminado de la infraestructura, con el fin de optimizar el coste en los despliegues realizados en proveedores públicos.

R11 Definir el número mínimo de recursos computaciones en estado IDLE o STOPPED. Este número debe ser configurable por el profesorado. En función de ciertos parámetros como el tiempo de despliegue o los picos de demanda de recursos, puede ser interesante tener recursos preconfigurados para que el tiempo de espera del alumnado hasta tener disponible el instrumental sea razonable.

R12 Tener todos sus componentes OpenSource.

3. Cluster Elasticity Manager

Cluster Elasticity Manager (CEM) [7] ha sido diseñado para cumplir todos requisitos descritos en 2.2. Para

[2] Página web de Microsoft Azure: `https://azure.microsoft.com/`.

[3] Página web de Amazon Web Services: `https://aws.amazon.com/`

[4] Página web de Google Cloud: `https://cloud.google.com/`.

[5] Página web de OpenNebula: `https://opennebula.org/`

[6] Página web de OpenStack: `https://www.openstack.org/`

[7] Repositorio GitHub de Cluster Elasticity Manager: `https://github.com/grycap/cem`

ello, se propone la arquitectura mostrada en la Figura 1, la cual está formada por dos tipos de recursos (MVs): *front-end* y *worker*. Por seguridad, el único nodo accesible desde el exterior es el *front-end*. Los *workers* están preparados para ser accesibles a través de un cliente de escritorio remoto y así poder utilizar la interfaz gráfica del software específico (para el caso abordado en este trabajo, Xilinx Vivado), cumpliendo con el requisito R04. Debido a que el único recurso accesible desde el exterior es el nodo *front-end*, es necesario realizar una redirección de los puertos de los escritorios remotos de los nodos *worker* a algún puerto del nodo *front-end*. Para ello, se utiliza *iptrest* [8], un desarrollo OpenSource accesible mediante llamadas API REST que crea o elimina las reglas correspondientes del cortafuegos para que los alumnos puedan conectarse por escritorio remoto al recurso correspondiente sin ninguna necesidad de hardware específico (R02). La creación de los recursos en los proveedores cloud y su posterior configuración se delega en Infrastructure Manager (IM) [3], el cual posibilita, mediante interfaz web, cliente CLI y API REST, la configuración automatizada de los nodos por medio de recetas y el manejo de infraestructuras en los clouds (privados y públicos) más populares, cumpliendo con los requisitos R01, R06 y R08.

Los dos tipos de recursos CEM los forman tres componentes principales: *CEM-Server*, *CEM-Agent* y *CEM-Web*. *CEM-Server* se encarga de gestionar la infraestructura y (des)asignar recursos. *CEM-Web* ofrece una interfaz web a los usuarios para la interacción con *CEM-Server*. Tanto *CEM-Server* como *CEM-Web* se encuentran desplegados en el recurso *front-end*. En cada recurso *worker*, se despliega el componente *CEM-Agent*, el cual se encarga de monitorizar el recurso y de controlar que no hay usuarios no autorizados por CEM. Los usuarios deben disponer de sus archivos y configuraciones cada vez que acceden al sistema y, dado que se les puede asignar un recurso distinto cada vez, es necesario utilizar un servicio que posibilite su acceso en cualquier nodo. Para ello, se utiliza Network File System (NFS) para almacenar los directorios `home` de cada usuario y, además, los *logs* de su trabajo. Gracias a esto, es posible cumplir con el requisito R07. Además, para almacenar la información, CEM utiliza la base de datos SQLite [9]. Cabe destacar que *iptrest*, IM, NFS, SQLite y CEM son desarrollos OpenSource, lo que permite cumplir con el requisito R12.

En CEM, el rol de los usuarios puede ser alumnado, profesorado o administrador/a. Los usuarios con rol de alumnado solamente pueden demandar recursos o cancelar sus recursos asignados. El profesorado puede solicitar recursos de la misma manera que los alumnos y también tienen acceso total a los estados de los usuarios, las asignaciones de recursos actuales y de los nodos. Además, pueden solicitar la eliminación, suspensión, reinicio o creación de recursos y la liberación de recursos asignados a los usuarios. Los administradores tienen acceso a la información de los *logs* de los componentes y pueden reiniciar los servicios. El estado de los usuarios (primer diagrama de la Figura 2) se ha modelizado con las siguientes categorías: *UNKNOWN*, *NOTHING_RESERVED* cuando el usuario todavía no ha demandado recursos, *WAITING_RESOURCES* cuando ha solicitado recursos pero no están disponibles, *RESOURCES_ASSIGNED* cuando tiene recursos asignados pero no los está utilizando o *ACTIVE* si está utilizando sus recursos asignados.

Los recursos computacionales se modelizan con dos tipos de estado: utilización y configuración. El estado de configuración describe en qué punto se encuentra el recurso desde el punto de vista de la administración de sistemas. Para ello, se modelizan los recursos como *UNKNOWN*, *PENDING* cuando el recurso está iniciándose en el proveedor cloud, *CONFIGURING* cuando está siendo (re)configurado, *CONFIGURED* cuando está correctamente configurado, *UNCONFIGURED* si el proceso de configuración falló, *OFF* si el recurso ha sido eliminado, *STOPPED* si el recursos está suspendido o *FAILED* si ha habido cualquier problema en la interacción con el proveedor cloud. Por otra parte, el estado de utilización es *UNKNOWN* mientras el recurso no está en el estado *CONFIGURED* y, en función de los usuarios que lo tienen reservado, el estado puede ser: *IDLE* cuando no hay ningún usuario asignado, *USED* cuando al menos tiene un usuario asignado o *FULL*, cuando ya tiene todos sus slots de usuario ocupados. El diagrama de flujo de la parte inferior de la Figura 2 muestra los posibles estados configuración y de utilización, coloreados, respectivamente, en negro y azul.

Con los posibles estados de los usuarios y de los recursos en mente, se va a describir más detalladamente la finalidad de los componentes de CEM. *CEM-Server* es un servicio accesible mediante API REST encargado de procesar las peticiones de recursos, gestionar los recursos *worker* y las asignaciones de recursos a usuarios. La gestión de recursos tiene en cuenta la configuración de los usuarios *administrador* (utilizando el fichero de configuración de CEM) respecto al número máximo de usuarios autorizados por recurso, al tiempo de inactividad de los usuarios una vez se les ha asignado un recurso, al tiempo de los recursos en estado *IDLE* y al número mínimo de recursos en estado *IDLE* o *STOPPED* junto con las demandas en tiempo real de los usuarios mediante *CEM-Web*. Gracias a

[8] Repositorio GitHub de Iptrest: `https://github.com/grycap/iptrest`

[9] Página web de SQLite: `https://www.sqlite.org/index.html`

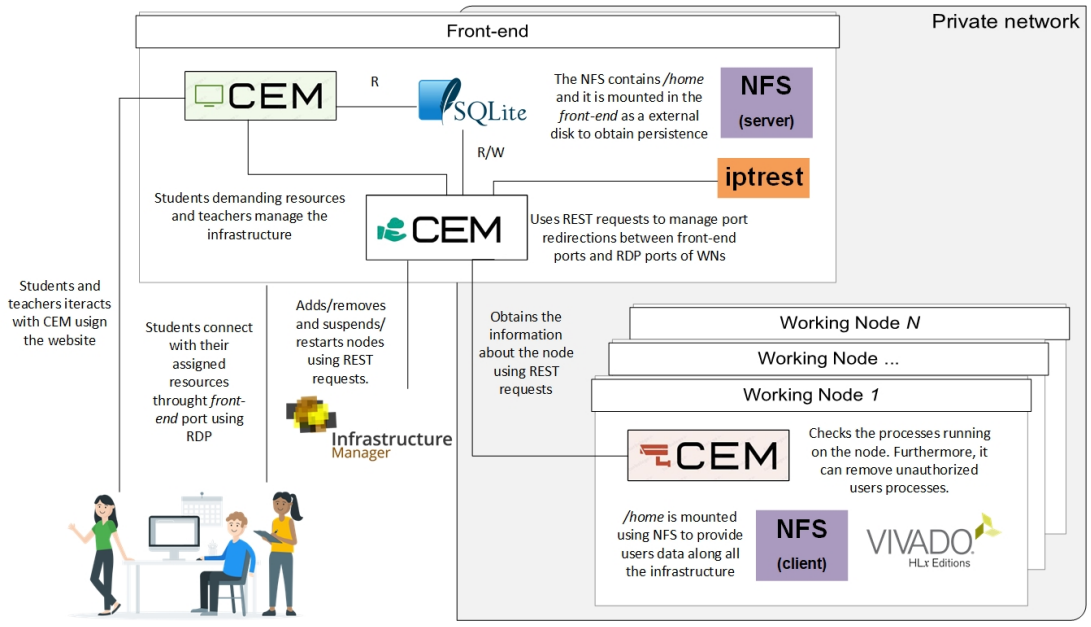

Figura 1: Arquitectura propuesta

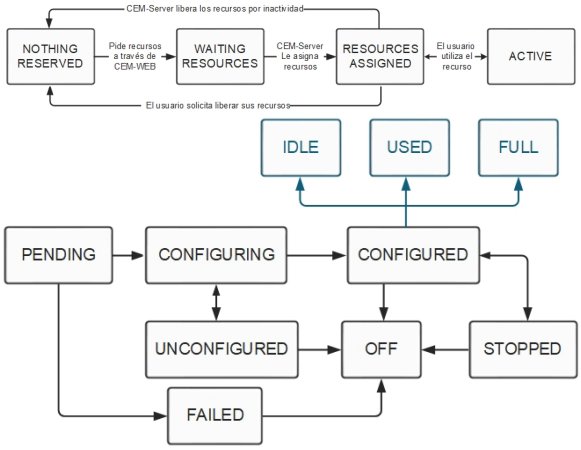

Figura 2: Ambos diagramas representan el flujo de los posibles estados de los usuarios (arriba), de recursos (parte inferior de color negra) y los estados de utilización (parte inferior de color azul).

esta personalización de *CEM-Server* es posible cumplir con los requisitos R09, R10 y R11. *CEM-Web* permite a los usuarios *profesor* y *administrador* interactuar con *CEM-Server* para crear nuevos recursos, eliminar o suspender los que ya han sido desplegados o reiniciar aquellos que están suspendidos. Los usuarios *alumno* solicitan que se les asignen recursos mediante *CEM-Web* y, una vez *CEM-Server* se los asigna, se les proporciona el usuario, la contraseña y la configuración necesaria para conectar al escritorio remoto. En las Figuras 3-5 se puede observar la información de los usuarios *alumno* en cada posible estado. La información accesible desde los roles *profesor* y *administrador* en *CEM-Web* puede observarse en las Figuras 6-9, las cuales describen en tiempo real, respectivamente,

Figura 3: Vista de los usuarios *alumno* durante el estado *NOTHING_RESERVED*.

el estado de los recursos, las asignaciones, el estado de los usuarios y el histórico de asignaciones por usuario. Para que *CEM-Server* tenga la información suficiente para determinar si los usuarios *alumno* están en estado *ACTIVE*, es necesario otro componente, *CEM-Agent*, el cual se encarga de monitorizar los todos procesos y, adicionalmente, de controlar que ningún usuario no autorizado esté utilizando ese recurso. La comunicación entre los componentes CEM se realiza mediante API REST.

Gracias a todos los componentes descritos durante la sección, es posible cumplir con el requisito R05, el cual sea probablemente el más complejo de los descritos en la Sección 2.2 y, también, altamente estudiado en el el campo de investigación cloud. El cumplimiento del requisito R03 es proporcionado por la plataforma cloud elegida para el despliegue de la infraestructura.

4. Análisis de resultados

En esta sección, en primer lugar se realiza un análisis para determinar los requerimientos necesarios de

Número de núcleos	Operación	Windows 10 Enterprise Edition			Linux Ubuntu		
		Tiempo de CPU (s)	Tiempo transcurrido (s)	Memoria máxima (MB)	Tiempo de CPU (s)	Tiempo transcurrido (s)	Memoria máxima (MB)
1	Síntesis	24.00	31.00	801.45	18.00	31.00	1683.65
	Implementación	35.00	31.00	1343.82	27.00	19.00	2399.03
	Bitstream	15.00	15.00	1571.22	9.00	11.00	2370.74
2	Síntesis	23.00	30.00	803.08	18.00	31.00	1685.5
	Implementación	35.00	30.00	1342.91	26.00	18.00	2393.07
	Bitstream	16.00	15.00	1574.19	9.00	11.00	2370.74
4	Síntesis	24.00	30.00	803.70	18.00	31.00	1683.65
	Implementación	35.00	30.00	1338.36	25.00	18.00	2400.03
	Bitstream	16.00	15.00	1576.00	9.00	11.00	2370.74

Cuadro 1: Tiempo necesario y memoria máxima consumida para ejecutar las operaciones principales del entorno Xilinx Vivado, utilizando diferente número de núcleos de procesamiento, en un PC con Windows 10 y Linux.

Figura 4: Vista de los usuarios *alumno* durante el estado *WAITING_RESOURCES*.

Figura 5: Vista de los usuarios *alumno* durante los estados *RESOURCES_ASSIGNED* y *ACTIVE*.

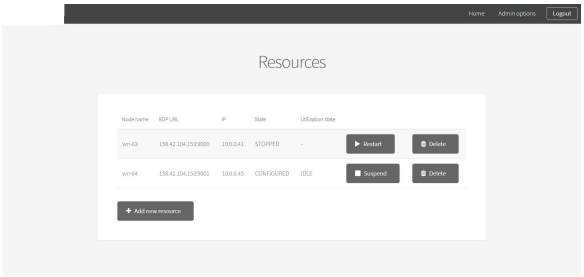

Figura 6: Recursos computaciones y sus estados.

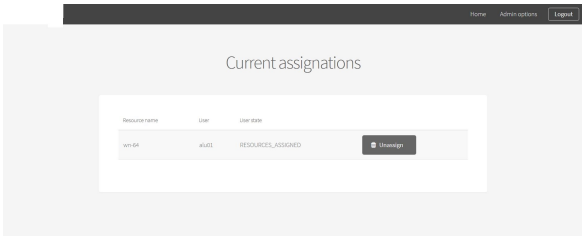

Figura 7: Asignaciones de recursos a los usuarios en tiempo real.

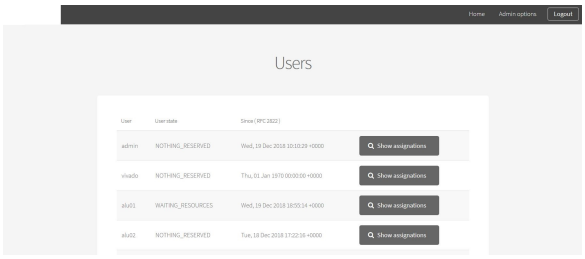

Figura 8: Estado de todos los usuarios en tiempo real.

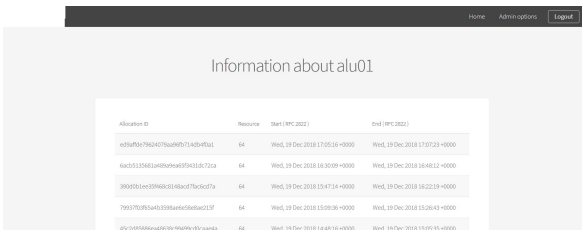

Figura 9: Histórico de asignaciones del *alumno* “alu01”.

los recursos que den soporte al instrumental específico que debe proporcionar CEM a cada uno de los estudiantes de forma individual en la asignatura presentada en el escenario de referencia. A continuación, se presenta un análisis de los tiempos de despliegue.

4.1. Recursos necesarios

Para poder determinar la configuración mínima de los recursos que el sistema cloud debe proporcionar, se ha procedido a realizar, en un ordenador de sobremesa, la práctica más demandante desde el punto de vista computacional. Esta máquina dispone de un procesador Intel Core i7-4790 a 3.60 GHz, 16 GB de memoria RAM, y sistemas operativos Windows 10 Enterprise N y Linux Ubuntu 18.04.1 LTS con el entorno Vivado instalado.

El Cuadro 1 muestra los tiempos de ejecución de las principales operaciones del flujo de diseño *semicustom* y la cantidad máxima de memoria consumida, cuando se configura el entorno para utilizar diferente número de núcleos de ejecución. El tiempo de CPU indica el tiempo total dedicado por el procesador a ejecutar las diferentes operaciones indicadas, mientras que el tiempo transcurrido se refiere al tiempo total de ejecución de la operación, lo que incluye todas las operaciones de entrada/salida.

Tal y como se aprecia en el Cuadro 1, el número de núcleos disponibles en el procesador no influye en el tiempo requerido para la ejecución de las operaciones de síntesis, implementación y generación del *bitstream*. La operación de síntesis es la que más tiempo requiere para su procesamiento, y es prácticamente idéntico tanto en Windows como Linux. Sin embargo, la operación de implementación, que en Windows consume un tiempo similar al de síntesis, en Linux no solo requiere de menor tiempo de CPU, sino que el tiempo total transcurrido se reduce enormemente. Asimismo, el tiempo de generación del *bitstream* es sensiblemente menor en Linux. Esto puede deberse a que en Windows el entorno limita las operaciones a solo 2 hilos de ejecución simultáneos, mientras que en Linux permite hasta 8 hilos de ejecución.

Por otro lado, puede apreciarse que el consumo máximo de memoria durante la ejecución de las operaciones es aproximadamente un 40 % menor en Windows que en Linux. A pesar de esto, el consumo máximo de memoria no supera los 2.4 GB.

Por ello, siempre y cuando la memoria requerida por los recursos que deban desplegarse no supongan una limitación, parece recomendable proporcionar acceso a Vivado a través de un sistema operativo Linux para reducir al máximo el tiempo de ejecución de las operaciones requeridas.

Tipo de nodo	Operación	Tiempo (s)
Front-end	Creación	00:08:33
Worker	Creación	00:08:40
Worker	Suspensión	00:00:04
Worker	Reinicio	00:00:10

Cuadro 2: Tiempo de despliegue y configuración para cada tipos de nodo (*front-end* y *worker*) y las operaciones disponibles de cada tipo de nodo (creación, suspensión y reinicio).

4.2. Tiempos de despliegue

Se ha realizado un estudio del tiempo necesario de despliegue y configuración del entorno propuesto en este artículo en una infraestructura privada. Esta infraestructura está formada por dos tipos de nodo. El primer tipo de nodo se compone de dos procesadores Intel(R) Xeon(R) CPU E5-2683 v3 2.00GHz (14 núcleos), 64 GB de memoria RAM, 240 GB de disco dura de tipo Solid State Disk (SSD) y tres adaptadores de red (dos 1 GB Ethernet y un 10 GB Ethernet). El segundo tipo de nodo se compone de dos procesadores Intel(R) Xeon(R) CPU E5-2660 v4 2.00GHz (14 núcleos), 128 GB de memoria RAM, 250 GB de disco dura de tipo Solid State Disk (SSD) y tres adaptadores de red (dos 1 GB Ethernet y un 10 GB Ethernet). La red de área de almacenamiento (en inglés, Storage Area Network) es Dell Equallogic PS4210 con 16 TB disponibles. Toda la infraestructura está controlada por el proveedor cloud privado OpenNebula Cloud Management Framework y el hipervisor Kernel-based Virtual Machine [10].

La infraestructura virtual propuesta para el caso de uso de la asignatura DSD se compone, como se ha descrito a lo largo del documento, de dos tipos de nodo: *front-end* y *worker*. El nodo *front-end* está copuesto de 4 vCPUs, 8 Gb de memoria RAM y dos interfaces de red (una para la red privada y otra para la red pública). Los nodos *worker* están compuestos de 3 vCPU, 12 Gb de memoria RAM y una interfaz de red privada. El Cuadro 2 resume los tiempos necesarios para que los nodos estén totalmente configurados y listos para utilizar.

5. Conclusiones

En este trabajo se ha realizado un estudio de las necesidades de la asignatura *Uso en el Diseño de Sistemas Digitales* y del intrumental que se utiliza en la asignatura. Además, se ha desarrollado una herramienta, *Cluster Elasticity Manager*, que tiene como objeti-

[10] Página web de KVM: `https://www.linux-kvm.org/page/Main_Page`

vo la gestión elástica de infraestructuras cloud (privadas o públicas) centradas en ofrecer un entorno gráfico al alumnado que es totalmente personalizado y monitorizado por el profesorado. Adicionalmente, se ha propuesto una arquitectura fácilmente adaptable a otros escenarios de referencia, compuesta por herramientas OpenSource y que puede desplegarse en los proveedores cloud más populares. Finalmente, se ha medido el despliegue de los dos tipos de nodo de manera completamente automatizada y, en el caso de los recursos *worker*, también se ha medido el tiempo que se consume para suspender y reiniciar el recurso, siendo estos tiempos razonables y operativos.

Respecto a los requisitos identificados a partir de las necesidades detectadas en el escenario de referencia, con la herramienta implementada, la arquitectura propuesta y el despliegue efectivo de un escenario del referencia estudiado, puede concluirse que se ha logrado realizar un trabajo que cumple con todos los requisitos especificados.

Agradecimientos

Los autores agradecen este trabajo por la financiación recibida por el Vicerrectorado de Estudios, Calidad y Acreditación de la Universitat Politècnica de València (UPV) para desarrollar el Proyecto de Innovación y Mejora Educativa (PIME) “Comunidades de Aprendizaje como servicios en la nube para el desarrollo y evaluación automática de Competencias Transversales y Objetivos Formativos específicos”, con referencia B29.

Referencias

[1] Elliot Aronson y Shelley Patnoe. *Cooperation in the Classroom: The Jigsaw Method*. Pinter & Martin Ltd, 3rd edition, 2011.

[2] Ángela Belcastro, Roger David Alanes, Macarena Quiroga, Juan Giménez, Santiago Santana, Pablo Dibez y Rodolfo Alfredo Bertone. Juegos interactivos en ARDUINO y Java, para motivar y despertar el interés en Informática. En *XIX Workshop de Investigadores en Ciencias de la Computación*, 2017.

[3] Miguel Caballer, Ignacio Blanquer, Germán Moltó y Carlos de Alfonso. Dynamic Management of Virtual Infrastructures. *Journal of Grid Computing*, 13(1):53–70, 2015.

[4] López Hung, Eduardo and Muguercia Bles, Alcides. Potencialidades didácticas del simulador Packet Tracer para la enseñanza – aprendizaje del diseño de redes de computadoras. *V Jornada Virtual de Educación Médica 2017*, 2017.

[5] Susana B. Fiad y Ofelia D. Galarza. El Laboratorio Virtual como Estrategia para el Proceso de Enseñanza-Aprendizaje del Concepto de Mol. *Formación universitaria*, 8(4):03–14, 2015.

[6] José A González-Martínez, Miguel L Bote-Lorenzo, Eduardo Gómez-Sánchez y Rafael Cano-Parra. Cloud computing and education: A state-of-the-art survey. *Computers & Education*, 80:132–151, 2015.

[7] IEEE Standard VHDL Language Reference Manual. Standard, IEEE Computer Society, 2008.

[8] María del Mar López Guerrero, Gema López Guerrero y Santiago Rojano Ramos. Uso de un simulador para facilitar el aprendizaje de las Reacciones de Óxido-Reducción. Estudio de caso en la Universidad de Málaga. *Educación Química*, 29(3):79, aug 2018.

[9] Larry McMurchie y Carl Ebeling. *PathFinder: A Negotiation-based, Performance-driven Router for FPGAs*, chapter 16, pages 365–381. Morgan Kaufmann Publishers Inc., 2007.

[10] A.P. Medina, G.H. Saba, J. Hernández y E. Ladrón de Guevara. Los laboratorios virtuales y laboratorios remotos en la enseñanza de la ingeniería. *Academia Journals*, 4, 2011.

[11] Michael Michalko. *Thinkertoys: A Handbook of Creative-Thinking Techniques*. Random House USA Inc, 2nd edition, 2006.

[12] Julio Cesar Ponce, Antonio Silva Sprock, Jaime Muñoz-Arteaga y Francisco Ornelas. Sistemas Multiagentes y sus Aplicaciones a la Gestión de Objetos Digitales Educativos View project Data mining and big data applications View project. Technical report, 2014.

[13] Rob. A. Rutenbar. Simulated Annealing Algorithms: An Overview. *IEEE Circuits and Devices Magazine*, 5(1):19–26, 1989.

[14] Ricardo Adán Salas-Rueda. Uso del modelo TPACK como herramienta de innovación para el proceso de enseñanza-aprendizaje en matemáticas. *Perspectiva Educacional*, 57(2):3–26, 2018.

[15] J. D. Segrelles and G. Moltó. Assessment of cloud-based computational environments for higher education. En *2016 IEEE Frontiers in Education Conference (FIE)*, pages 1–9, 2016.

[16] J. D. Segrelles, G. Moltó y M. Caballer. Remote Computational Labs for Educational Activities via a Cloud Computing Platform. En *Proceedings of the Information Systems Education Conference (ISECON)*, pages 309–310, 2015.