Informes de la Construcción
Vol. 66, EXTRA 1, m001
diciembre 2014
ISSN-L: 0020-0883
doi: http://dx.doi.org/10.3989/ic.13.075

Aplicación web para el análisis y diseño de estructuras

Web application for the analysis and design of structures

J. Calvo (*); J. Gracia (*); E. Bayo (*)

RESUMEN

El *«Cloud computing»* describe un modelo innovador de prestación de servicios de forma centralizada usando Internet como medio de distribución. A día de hoy, el análisis de estructuras, que requiere grandes prestaciones de cálculo, no ha sido «migrado» al entorno de Internet, y suele ser desarrollado como software de escritorio en contra de la tendencia actual del mercado informático.

En este artículo se expone la arquitectura de una aplicación web para el análisis y diseño de estructuras, que consistirá en un paquete de servicios web como SaaS (*Software as a Service*) y estará compuesta por 4 módulos: un modelador 3D, un software de análisis, un software de diseño y un módulo CAD. También se exponen las herramientas actuales para su desarrollo como una aplicación integrada en el *cloud*.

Por último, se desarrolla una versión preliminar del módulo CAD que genera la documentación gráfica final en formato DXF.

Palabras clave: Cloud computing; servicio web; SaaS; estructuras; diseño integral.

ABSTRACT

"Cloud computing" describes an innovative use of computing resources that are delivered as a service over a network, typically Internet. Today, structure analysis that requires high performance calculation has not been fully "migrated" to the Internet environment, structural software is usually developed as a desktop program against the current trend of the computer market.

In this paper, the architecture of a structural analysis web application will be presented. This application, which would be a package of web services as SaaS (Software as a Service), will consist of 4 modules: a 3D modeler, a structural analysis solver, a structural design solver and a graphic documentation render. Also, current tools to develop this service as cloud application will be presented.

Finally, a preliminary version of the CAD module, which generates graphical output as DXF format, is developed.

Keywords: *Cloud computing; web service; SaaS; structures; integral design.*

(*) Universidad de Navarra, España.
Persona de contacto/*Corresponding author:* jcalvov@alumni.unav.es (J. Calvo)

Cómo citar este artículo/*Citation:* Calvo, J., Gracia, J., Bayo, E. (2014). Aplicación web para el análisis y diseño de estructuras. *Informes de la Construcción*, 66(EXTRA-1): m001, doi: http://dx.doi.org/10.3989/ic.13.075.

Recibido/*Received*: 07/06/2013
Aceptado/*Accepted*: 15/07/2013

1. INTRODUCCIÓN

El *«Cloud computing»* describe un modelo innovador de prestación de servicios de forma centralizada usando Internet como medio de distribución. La propia arquitectura del sistema permite que el usuario acceda a un catálogo de servicios de forma flexible pagando únicamente por el consumo efectuado. Por tanto, este sistema beneficia tanto a los proveedores, que pueden incrementar de forma ágil y trasparente la capacidad del servicio en caso de picos de demanda no previstos, como a los consumidores, que acceden a un sistema «transparente» en el que solamente se abona el consumo realizado.

El concepto de la computación en la «nube» comenzó con proveedores de servicios de Internet a gran escala como Google, Amazon WS o Microsoft, entre otros, que a partir de sus propios desarrollos e infraestructuras empiezan a ofrecerlas comercialmente al resto del mundo. A partir de ese momento surge una arquitectura de sistemas basada en recursos distribuidos horizontalmente como servicios virtuales de TI (Tecnología de la Información) escalados masivamente.

Dentro del *Cloud Computing*, tal y como se muestra en la tesis de Erb (1), se diferencian tres esquemas (capas) de servicio. La capa más alta, denominada «Software como Servicio (SaaS)», ofrece una aplicación completa (software) como servicio. El software corre en la infraestructura del proveedor sirviendo a múltiples clientes. Google Apps o MS Office 365 con su *suite* ofimática *online* son ejemplos de ello. La segunda capa se refiere a «Plataforma como Servicio (PaaS)» en la que el proveedor proporciona una plataforma con un entorno de desarrollo y APIs completa. Los principales ejemplos de este modelo son Windows Azure o Google App Engine. Por último, «Hardware o Infraestructura como Servicio (HaaS)», en la que se proporciona un almacenamiento básico y capacidad de cómputo como servicios de red. El caso más conocido es Amazon Web Services.

El aumento del rendimiento de los equipos informáticos actuales ha incrementado las características y la capacidad de cálculo del software para el análisis y diseño de estructuras, sólo limitados por los recursos de hardware de los ordenadores. Dicho aumento, con marcado carácter exponencial, obliga a una constante revisión y adaptación de los desarrollos actuales. Hoy en día, el análisis de estructuras, que requiere grandes prestaciones de cálculo, no ha sido «migrado» al entorno de Internet; suele ser desarrollado como software de escritorio en contra de la tendencia actual del mercado informático, donde las tareas comunes más cotidianas como el correo electrónico, hojas de cálculo o servicios de almacenamiento, ya han sido migradas a la nube.

En el presente artículo, se expone el desarrollo de una aplicación para el diseño integral de estructuras. Esta aplicación, que consiste en un paquete de servicios web como SaaS *(Software as a Service)*, está formada por 4 módulos: un modelador 3D, un software de análisis estructural, un software de diseño estructural y un generador de la documentación gráfica. Toda la aplicación estará distribuida en un entorno *cloud*, con posibilidad para escalarse de forma horizontal en función de la demanda solicitada. También se presentan las herramientas actuales para desarrollar esta aplicación como un servicio integrado en el *cloud*, dando lugar a un servicio basado en REST *(Representational State Transfer)* y explicando el funcionamiento de una API *(Application Programming Interface)* creada para dar soporte a otras plataformas. Se estudia también la oferta actual que ofrece Amazon Web Services para este tipo de aplicaciones *cloud* y como se puede integrar el servicio web presentado en esta plataforma.

Por último, se desarrolla una versión preliminar del último módulo, el de CAD, el cual generará la documentación gráfica final como un fichero DXF. El módulo permite obtener la documentación gráfica de una tipología de nave industrial cuya definición se aporta en un archivo «provisional» con los parámetros necesarios. El módulo, una vez interpretada toda la definición de la nave aportada en el archivo, permite al usuario modificar los parámetros más relevantes. Posteriormente, previa confirmación por parte del usuario, el servidor genera toda la documentación sin necesidad de ningún tipo de cálculo en el equipo cliente. La documentación generada (Figura 1), estará disponible al usuario por medio de un enlace de descarga. Para la programación de esta versión beta, se desarrolla adicionalmente un entorno principal que permite el acceso a la anterior a través del registro del usuario y sus privilegios.

2. APLICACIÓN PARA EL DISEÑO INTEGRAL DE ESTRUCTURAS

Este artículo, en su primera parte, pretende dar un enfoque global desde un punto de vista teórico, de cómo desarrollar un servicio web que abarque todas las ramas necesarias para el diseño completo y análisis de un sistema estructural, desde el modelado de la misma a través de una aplicación gráfica,

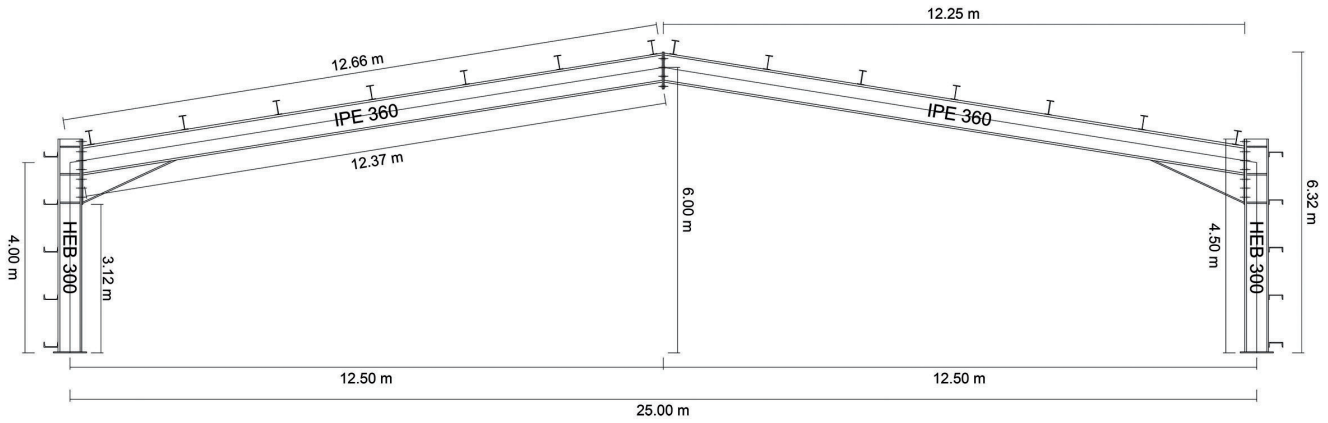

Figura 1. Ejemplo de pórtico generado con el sistema.

hasta la solución final que detalle gráficamente el proyecto para su puesta en obra.

El proyecto consiste en una aplicación con varios módulos integrada totalmente en la nube, a la que el usuario cliente accede, bien mediante una aplicación web cliente, o bien por medio de aplicaciones de escritorio o móviles haciendo uso de las APIs correspondientes. El acceso mediante la utilización de las APIs queda en uso tanto para aplicaciones de escritorio como para aplicaciones de dispositivos móviles, en las que la interfaz de usuario es adaptada para el uso cómodo y sencillo desde estos terminales. De la misma forma, la API permite el acceso a los servicios desde aplicaciones programadas por terceros, facilitando así la integración de los servicios desde otras aplicaciones o comunicaciones con empresas del sector que utilizan otras herramientas, para la realización de un proyecto en conjunto.

El resultado final es un SaaS, fácilmente portable a cualquiera de las HaaS existentes, con capacidad para escalar en función de la demanda. Este servicio permite que el usuario pueda acceder a la aplicación solamente con una conexión a Internet, en cualquier punto y sin necesidad de disponer de un equipo acorde a la complejidad del cálculo.

2.1. Módulos de la aplicación

El usuario es guiado a través de la interfaz gráfica de la aplicación por los diferentes módulos que entran en juego para desarrollar el proyecto estructural. Estos son enumerados a continuación.

1. Modelador 3D.
2. Aplicación para análisis estructural.
3. Aplicación de diseño estructural.
4. Módulo de CAD. Generador de la documentación gráfica.

El cliente, a través del menú de la aplicación, tiene la posibilidad bien de crear un nuevo proyecto, en donde será guiado por los diferentes módulos, o bien modificar uno en curso. La interacción del usuario con las diversas partes de la aplicación hasta del desarrollo DXF final se ofrece de una manera lineal, pero pudiendo también intervenir diferentes agentes en el mismo proyecto; mediante la asignación de privilegios por parte del administrador o compartiendo el proyecto a otros socios; gracias al alojamiento de todo el proyecto en la nube.

El primer módulo, el modelador 3D, está centrado en definir gráficamente la estructura conforme a la definición del proyecto. Aquí el usuario, a través de la correspondiente aplicación web cliente, dispone de un entorno gráfico 3D donde modelar gráficamente, mediante las herramientas proporcionadas, el esqueleto del sistema estructural. La poca exigencia gráfica de este proceso y el soporte de WebGL de los navegadores actuales, permitiendo el acceso al procesador gráfico por parte de la aplicación web; nos permite la ejecución de esta aplicación sobre el navegador del usuario, soportando este una pequeña carga de trabajo gráfico exclusivamente, tal y como muestran Gracia y Bayo (2). De esta forma, eliminamos costes adicionales del servicio así como la pequeña latencia visual que podría generarse.

Una vez realizado el trabajo en este primer módulo, los datos de la definición de la estructura quedan almacenados en el fichero del proyecto, definiendo ya un esqueleto de la tipología del mismo.

El módulo para el análisis estructural es el encargado de realizar un análisis conforme a la elección del cliente, obteniendo un dimensionado y análisis de la estructura, y realizándose todo el trabajo computacional en el entorno *cloud computing*. Es en el módulo de diseño estructural donde, con los datos del cálculo de la estructura, se definen las uniones y otros elementos restantes conforme a la normativa vigente.

El usuario tiene la posibilidad de modificar cualquier parámetro del proyecto en cualquier momento, aunque esto conllevará el recalculado de todas las demás partes, teniendo que revisar los usuarios oportunos esta actualización.

Por último, después de la intervención de las fases anteriores, el proyecto es procesado por el módulo CAD; este genera la documentación gráfica detallada del sistema estructural completo. Esta información es generada como fichero DXF que el usuario podrá descargar y utilizar con programas CAD de escritorio, y utilizarlo para la puesta en obra del proyecto.

2.2. Arquitectura REST y la API

El desarrollo del servicio web que nos ocupa, se implementa mediante una arquitectura tipo REST, creando una API que es consumida por JavaScript y por aplicaciones web programadas bajo PHP. Se usa por lo tanto el protocolo HTTP *(Hypertext Transfer Protocol)* como método de comunicación y JSON *(JavaScript Object Notation)* o XML *(Extensible Markup Language)* para el intercambio de datos.

REST es una técnica o familia de arquitecturas software para sistemas hipermedia distribuidos como la World Wide Web. El término se originó en el año 2000, en una tesis doctoral sobre la web escrita por Roy Fielding, uno de los principales autores de la especificación del protocolo HTTP, como explican Richardson y Ruby (3).

REST implica que una URL es la representación de un objeto o recurso, cuyos contenidos son accesibles mediante HTTP (Figura 2) (3) (4). Los clientes trabajan con estos recursos a través de las operaciones estándar («verbos») de HTTP, como GET, POST, PUT y DELETE para descargar o modificar una copia de la representación del recurso. La alternativa más conocida, SOAP *(Simple Object Access Protocol)*, es un protocolo estándar que utiliza una capa adicional por debajo de HTTP para la transmisión de información. Las arquitecturas que siguen los principios de REST son denominadas 'RESTful'.

El uso de REST frente a SOAP nos proporciona una mayor escalabilidad del servicio (5). Al tratarse de un protocolo cliente - servidor sin estado cada mensaje HTTP contiene toda la información necesaria para comprender la petición. Como resultado, ni el cliente ni el servidor necesitan recordar ningún estado de las comunicaciones entre mensajes.

Además, REST proporciona una arquitectura mucho más fácil, así como peticiones más simplificadas, permitiendo todo ello un mayor rendimiento y velocidad; y permite interpretar las URIs *(Uniform Resource Identifier)* de manera visual.

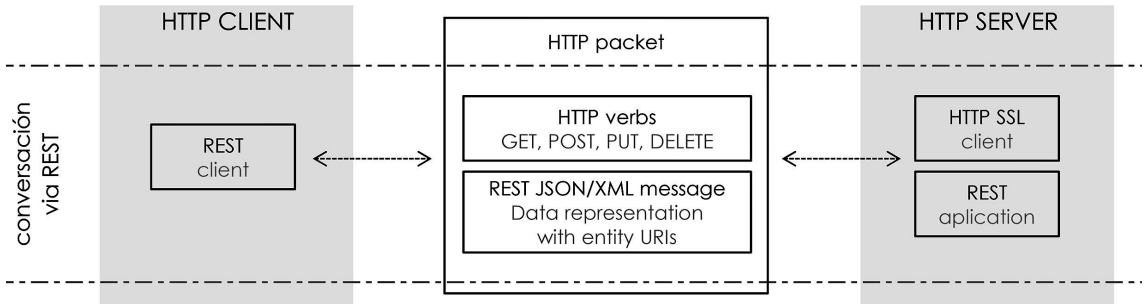

Figura 2. Conversación REST.

Por contra, la información es transferida de forma no optimizada según la aplicación y hacen falta múltiples llamadas para conseguir información compleja.

Un tipo de arquitectura SOAP es aconsejable cuando se describen en la comunicación todas las funciones de la interfaz o cuando es necesario abordar requerimientos complejos no funcionales. Por el contrario, la arquitectura basada en REST es útil cuando el servicio web no necesita tener estado, cuando se busca rendimiento o cuando tanto el cliente como el servidor conocen el contenido a intercambiar.

Las llamadas a recursos se hacen por medio de URIs, debiendo introducir en cada llamada la petición completa sin poder tener un estado previo como en SOAP, pero haciendo el sistema más flexible y eficaz, así como poder orientar el servicio mediante el uso de APIs a todo tipo de dispositivos fácilmente.

Como se ha indicado antes, el servicio consta de una API desarrollada mediante REST. Esta API permite el acceso, mediante las funciones implementadas en ella, a aplicaciones de escritorio y móviles; o la utilización de características del servicio a aplicaciones de terceros.

Una API es un grupo de funciones que permiten que desarrolladores puedan acceder al servicio mediante sus propias aplicaciones, de forma que pueden comunicarse con la base de datos sin necesidad de disponer del código fuente de nuestro sitio. El ejemplo más claro es Twitter que con su API se han creado multitud de sitios en Internet que, disponiendo de los datos de los usuarios de Twitter, pueden crear multitud de aplicaciones para ayudar o añadir valor al servicio.

Una API puede disponer o no de dos grandes funcionalidades:

1. Recoger y disponer de los datos del sitio para mostrarlos de forma conveniente.
2. Alterar esos datos, cómo modificarlos, añadir nuevos datos o eliminarlos.

Para la comunicación interna de la aplicación, como para el formato de almacenaje de los datos del proyecto del usuario (archivo de proyecto) se usa JSON. Esto nos permite una buena eficiencia en entornos donde se utilice AJAX *(Asynchronous JavaScript And XML)*, como demuestra Wang (6). JSON es un formato ligero para el intercambio de datos, generalizado como alternativa a XML en AJAX gracias a su simplicidad. JSON es una parte de la definición del estándar ECMA-262 en que se basa JavaScript. Se utiliza generalmente en entornos donde el tamaño del flujo de datos entre cliente - servidor es de vital importancia y cuando la fuente de los datos es de fiar.

La fácil escritura de un *parser* JSON y la posibilidad de análisis mediante la función *eval()* en JavaScript ha contribuido a su expansión actual. Conviene mencionar la falta de seguridad que plantea el uso de la función *eval()*, aunque no requerida para el intercambio de datos o funciones al usando la definición correcta de Literal JavaScript (LSJ).

Así, destacamos algunas ventajas de JSON sobre XML, como su facilidad de interpretación y creación en entornos como PHP, donde existen funciones nativas para crear objetos JSON; el menor consumo de ancho de banda en la comunicación de datos al evitar partes de estructura propias del lenguaje como tabulaciones y etiquetas del XML; y su soporte nativo por parte de JavaScript pudiendo acceder a los datos fácilmente mediante el operador "." (*objeto.propiedad)*, facilitando la extracción de contenido frente a XML.

3. INTEGRACIÓN DE LA APLICACIÓN EN UN HAAS

En este apartado se exponen las características principales de Amazon Web Services (AWS) en cuanto a las posibilidades que ofrece para alojar el servicio web explicado, dando lugar a una aplicación web escalable, así como los recursos necesarios para alojar la aplicación en sus servidores.

Se ha elegido AWS por ofrecer todas las herramientas necesarias para alojar nuestro servicio web, proporcionando de una manera muy flexible los recursos necesarios en cada momento en función de la demanda, así como por su estabilidad y precios. Sin embargo, hoy en día, existen numerosos servicios de similares características que permiten la integración de esta aplicación.

Si contratamos los servicios de Amazon WS dispondremos de un VPS *(Virtual Private Server)* a diferencia del *hosting* tradicional. En un VPS disponemos de un servidor propio virtualizado dentro de los servidores de la empresa, que aunque compartiendo recursos, tenemos garantizada una capacidad de cálculo del CPU y memoria en función de lo contratado. Al disponer en un VPS de acceso *root*, podemos instalar el software deseado. En AWS también es posible contratar un servidor dedicado, sin compartir recursos con otros usuarios.

Amazon ofrece diversos servicios y herramientas, siendo Amazon EC2 *(Amazon Elastic Compute Cloud)* el servicio principal. Amazon EC2 ofrece un servicio web que proporciona capacidad informática con tamaño modificable en la nube, donde podemos contratar distintos servidores en función de necesidades específicas, pudiendo elegir una mayor capaci-

dad de cómputo o mayor almacenamiento, facilitando unos recursos informáticos escalables. A cada uno de estos servidores virtuales se denomina «instancia».

Otros servicios muy importantes de AWS son: *Amazon Simple Storage Service* (Amazon S3), donde disponemos de un servicio de almacenamiento para Internet altamente escalable, fiable, y seguro; Amazon SQS *(Amazon Simple Queue Service)* como sistema de gestión de colas de trabajo para almacenar mensajes a medida que se transfieren entre sistemas; y Amazon EBS *(Amazon Elastic Block Store)* que nos proporciona volúmenes de almacenamiento a nivel de bloque para utilizarlos con las instancias EC2.

3.1. Distribución del SaaS en AWS

Para la integración del servicio web comentado a lo largo de este artículo en Amazon Web Services, se necesita el uso de varias instancias reservadas EC2, así como varios de los servicios mencionados. Estas instancias, para una mayor facilidad de gestión, llevarán Ubuntu como sistema operativo. Las instancias serán generadas teniendo en cuenta el servicio que van a correr, con una mejor capacidad gráfica para el módulo CAD, o mejor capacidad de computación en el caso del módulo de análisis. Para el servicio de bases de datos no usaremos Amazon SimpleDB, en vez de este utilizaremos una instancia EC2 ejecutando MySQL como servidor de bases de datos.

Se detalla a continuación un esquema de distribución del servicio en caso de alta demanda por gran cantidad de clientes simultáneos (Figura 3).

Para la distribución del módulo de CAD del servicio web, dedicaremos varias instancias EC2 ejecutando el código desarrollado. Estas instancias estarán controladas por otra de rendimiento reducido (instancia de control), encargadas de modificar dinámicamente las anteriores en función de la demanda, tanto aumentando sus características, como creando nuevas instancias.

Al disponer de distintas instancias ejecutando el módulo de CAD, debemos situar MySQL en otra independiente para el acceso a la base de datos a través del dominio interno ofrecido por Amazon para esta. En caso de tener una sola instancia EC2 podríamos instalar MySQL, si nos conviniese, en esta para acceder localmente a través de la IP 127.0.0.1, dejando la base de datos cerrada a conexiones exteriores para mayor seguridad.

La aplicación principal correrá en otra EC2 independiente, usando Amazon S3 como almacenamiento de los ficheros de proyecto de cada usuario. Cuando sea necesario generar la solución final de un proyecto, se creará un mensaje en una cola de Amazon SQS con una referencia al fichero de proyecto en Amazon S3. El software de la instancia EC2 disponible leerá la solicitud de la cola y recuperará el proyecto de Amazon S3 para su procesado. Después de esto lo colocará de nuevo en S3 creando otro mensaje en la cola de Amazon SQS indicando la finalización.

La instancia de control supervisará constantemente la cola de trabajo, y en función del número de mensajes de la misma, ajustará de forma dinámica el número de instancias de Amazon EC2 dedicadas al módulo CAD necesarias para cumplir con los requisitos de tiempo de respuesta de los clientes.

4. DESARROLLO DEL PROTOTIPO PARA EL MÓDULO CAD

Se desarrolla una versión preliminar del módulo CAD que genera la documentación gráfica final como un fichero DXF

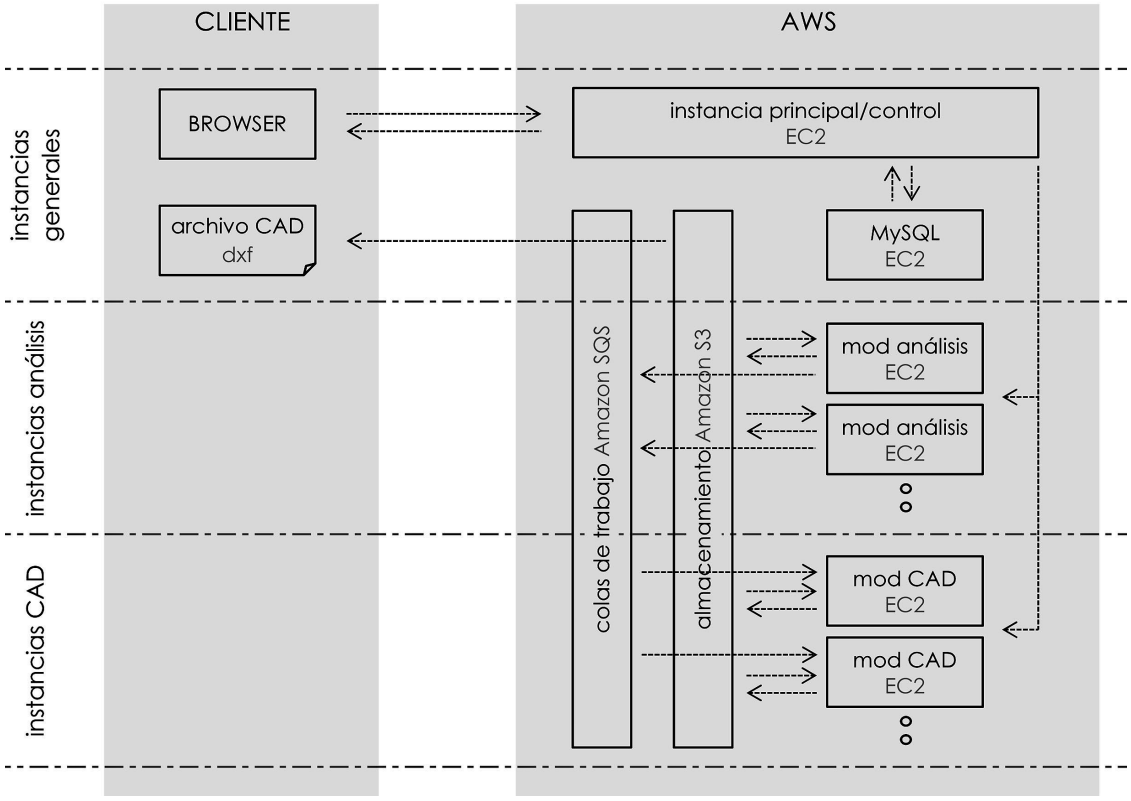

Figura 3. Esquema de distribución en AWS.

para una determinada tipología de nave industrial. Esta documentación será aportada al programa mediante un fichero externo ya definido. Este fichero «provisional» e intermedio es necesario para definir el proyecto al no existir los módulos previos. Una vez leídos los datos por el servidor, el módulo da la posibilidad al usuario de modificar los parámetros más relevantes. Una vez aceptados los datos se generará la documentación gráfica correspondiente en el servidor, sin ningún procesamiento en el equipo cliente. Esto finalizará con la información del fichero generado y su posible descarga como documento DXF (Figura 4).

Para el desarrollo de este servicio web se ha utilizado principalmente PHP como lenguaje de programación. PHP permite la introducción de contenido HTML que, junto con las instrucciones propias del lenguaje, será procesado completamente en el equipo servidor, sin que el usuario deba procesar nada ni pueda acceder a partes del servicio fuera de su acceso. Esta aplicación genera un nuevo documento HTML que utiliza CSS para definir la estructura de presentación o formato del mismo, y contiene instrucciones JavaScript. Estas rutinas son ejecutadas por el cliente para agilizar los tiempos de respuesta y, en su mayoría, son funciones para la validación de formularios y asignación de eventos a elementos HTML.

Esta aplicación PHP, durante la parte fundamental de creación de los ficheros DXF resultantes, se comunica con un ejecutable escrito en C++ pasándole los parámetros del dibujo como comandos. Así, la aplicación base se ejecuta por completo en el servidor sin requerir ningún tipo de proceso por parte del cliente y sin necesidad de incluir ninguna parte del código en la aplicación web, disminuyendo el tráfico de datos en la red y evitando accesos indebidos al mismo.

Para la realización de este prototipo, se ha montado la plataforma WAMP Server 2.2 sobre Windows 7. El proyecto WAMP es una infraestructura de Internet, bajo licencia libre, que engloba las siguientes herramientas: Apache como servidor web, MySQL como gestor de bases de datos y PHP (generalmente), Perl o Python como lenguajes de programación.

4.1. Implementación del módulo de CAD

Para un mejor seguimiento con la aplicación desarrollada, se procede a explicar la misma distinguiendo los 4 pasos que muestra visualmente durante el proceso, hasta la generación de la documentación final en DXF.

Recordemos que la programación de esta aplicación se basa fundamentalmente en PHP, por lo que el documento generado por estas rutinas contiene elementos HTML así como links en la cabecera del documento a dos ficheros externos: una hoja de estilo CSS y un documento con JavaScript.

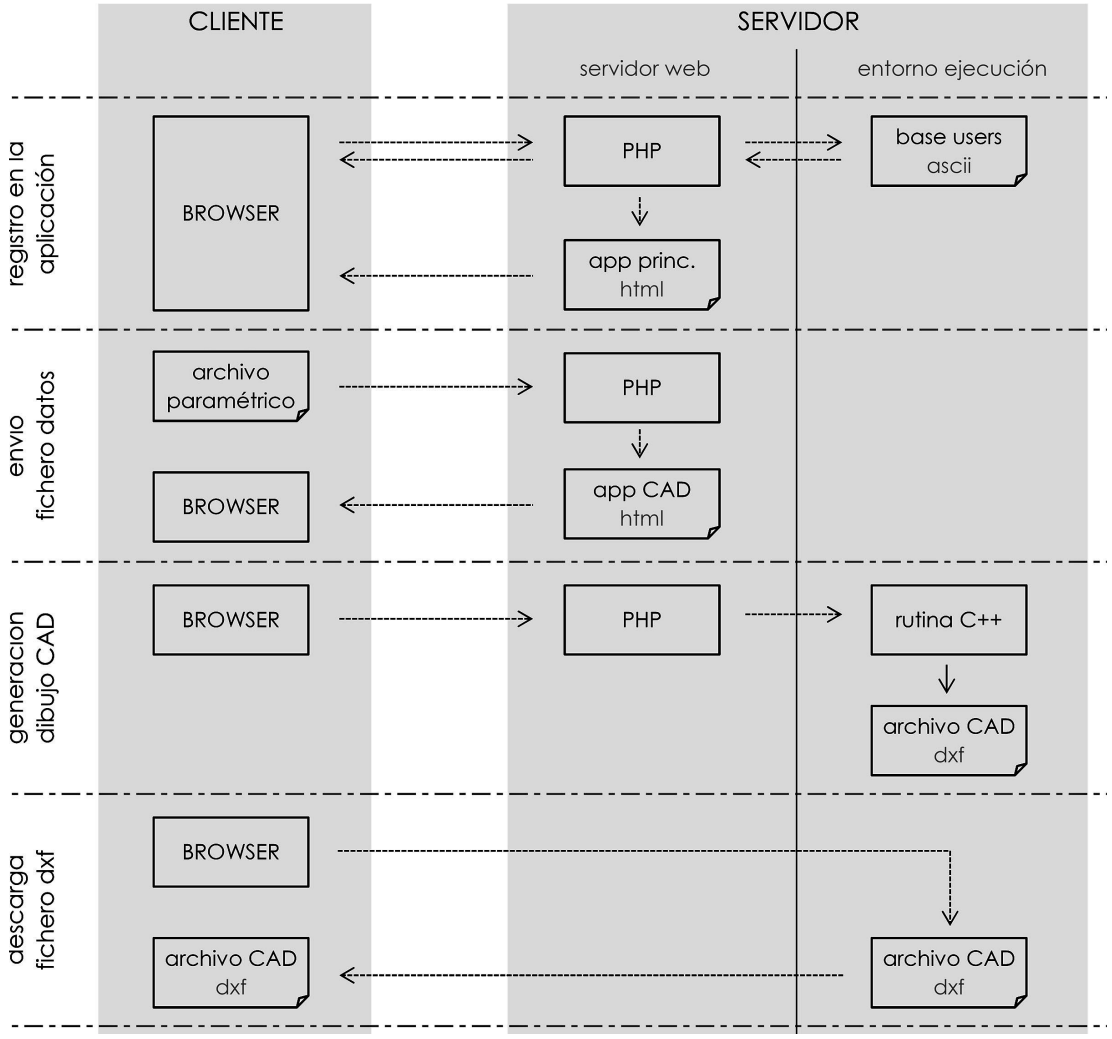

Figura 4. Esquema de distribución de la aplicación.

Para las peticiones cliente/servidor, se usa el método POST haciendo uso de un *submit* y enlazando el *action* del *form* con un documento PHP. Para no crear diferentes páginas PHP, una por cada paso descrito, se enlaza con el mismo documento recursivamente. Para que este muestre la sección correcta, PHP lee un campo oculto en el formulario que indica el paso actual.

```
1. $Paso=$_REQUEST[“Form0_1”];
```

En cualquier momento de la ejecución podemos volver al inicio de la aplicación presionando el botón superior ‘Reiniciar’.

Envío del fichero paramétrico:

En la primera sección de la aplicación, se pide al usuario mediante un *input* de tipo *file*, que seleccione el fichero con los parámetros del dibujo a crear.

Lectura del fichero:

En esta fase, la rutina de PHP lee e interpreta el archivo enviado. Para ello creamos un manejador que referenciará al archivo para trabajar con él:

```
1. $hadle=fopen($_FILES[‘Form1_1’][‘tmp_name’], “r”);
```

A medida que el fichero es interpretado por el servidor, los datos se van guardando en diversas variables que podrán ser modificadas posteriormente por el usuario mediante un formulario (Figura 5).

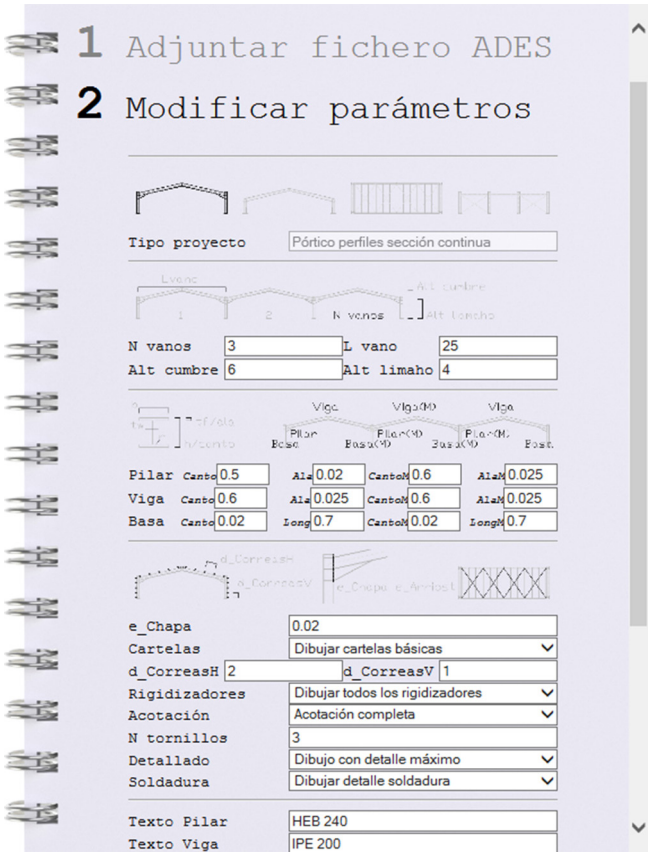

Figura 5. Prototipo de la aplicación de CAD.

Cuando se haga clic en el botón ‘Continuar’, se enviarán también todos los valores de los formularios, tanto si han sido modificados como si no.

Generación archivo DXF:

Una vez recuperados todos los valores finales, generamos una cadena de texto *$command* que contiene todos los parámetros que pasaremos al ejecutable externo de la forma siguiente:

```
1. exec($command, $output, $return_var);
```

Mediante las variables *$output* y *$return_var* obtenemos la información que ofrece el ejecutable C++, que mostraremos al cliente en un campo ‘textarea’.

Descarga documento:

Por último se permite al usuario la opción de descargar el fichero DXF generado. Para ello, se ha optado por relacionar el *submit* de un formulario directamente con el fichero. De esta forma no es necesario crear un enlace y la aplicación conserva la misma estética de los apartados anteriores:

```
1. <form method=”post”
   action=”portico.dxf”
   enctype=”multipart/form-data”>
```

Otra forma de realizar el proceso, más correcta en cuanto a la gestión de tráfico de datos en la red, hubiese sido crear un enlace como el siguiente:

```
1. <a href=”CaVaAC_4.php”>Descargar</a>
```

Y posteriormente controlar la descarga mediante PHP, generando una cabecera al siguiente documento HTML.

```
1. $filename=’portico.dxf’;
2. $fp=fopen($filename, ‘r’);
3. $output=fread($fp, filesize($filename));
4. fclose($fp);
5. header(“Content-type: application/autocad-dxf”);
6. header(“Content-Disposition; attachment;
   filename=Filename.dxf”);
7. echo $output;
8. exit();
```

4.2. Implementación del prototipo de la aplicación principal

Por último, se expone brevemente la implementación de la aplicación principal que permite el acceso a todos los módulos de la aplicación. Esta aplicación se presenta al usuario por medio de un entorno de trabajo con diversos iconos. El cliente accede por medio de un registro con su cuenta personal y el servidor le proporciona el acceso a los módulos para los que esté registrado. Este entorno, permite el registro simultáneo de diferentes usuarios, cada uno con sus permisos, sin interferir en los demás.

Este programa también está desarrollado en PHP introduciendo el uso de sesiones. Las sesiones nos permiten guardar el registro de un mismo usuario durante la duración de su visita a la aplicación, independientemente de que abra otra

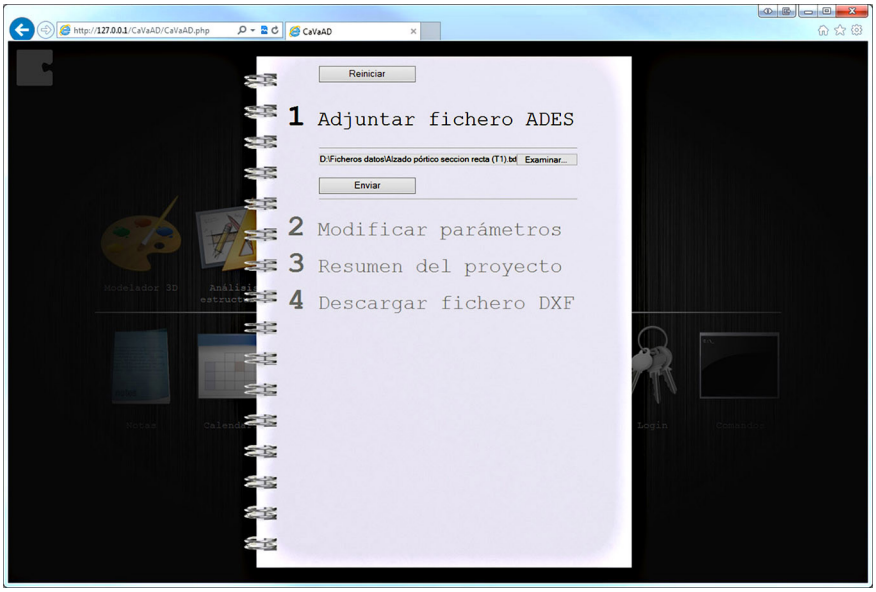

Figura 6. Prototipo de la aplicación de CAD sobre el menú del servicio.

página en su navegador y posteriormente regrese. Esto proporciona una mejor experiencia de usuario al cliente y nos permite mantener su sesión abierta mientras no cierre su navegador o finalice la sesión en el programa.

Si observamos la aplicación relacionada con el módulo de CAD (Figura 6), veremos que sobre un *div* se muestra totalmente funcional dicha aplicación, sin necesidad de recargar la página 'padre' para interactuar con ella. Esto se ha conseguido mediante la incrustación de un 'marco' posicionado en otro *div*. Mediante la etiqueta <*iframe*> insertamos el contenido de otro documento HTML sin necesidad de usar *frameset*. Esta porción de la página web toma el control y se ejecuta como si fuese suya la página.

5. CONCLUSIONES

Este artículo presenta una arquitectura diferente en cuanto a las aplicaciones de diseño y análisis de estructuras tradicionales. Esta arquitectura, basada en las tecnologías web actuales, ha demostrado, aun a estar a comienzos de su desarrollo, numerosos beneficios, tanto para el usuario como para los desarrolladores, enumerados a continuación:

1. Posibilidad de acceso a la aplicación desde cualquier ubicación.
2. Compatibilidad con todos los equipos independientemente de su sistema o de su hardware, solamente es necesario un navegador web como aplicación cliente. Por lo tanto no es necesario un equipo con altas prestaciones de hardware ya que el cálculo se desarrolla en la infraestructura y el cliente sólo recibe/envía la información o se le asigna una parte de computo proporcional a su equipo.
3. Al ser un entorno centralizado en el 'cloud', el usuario accede a versiones actualizadas del servicio sin ningún tipo de interacción por parte este. Cada vez que se actualiza la aplicación web, esta pasa a ser la que se distribuye para todos y cada uno de los clientes, independientemente de que su acceso se produzca por la web o mediante las APIs.
4. Se trata de una plataforma de uso basada en consumo y no en adquisición de productos evitando así altos costes iniciales.
5. Existe una mayor capacidad de adaptación, recuperación de desastres y reducción al mínimo de los tiempos de inactividad.
6. Se produce un uso más eficiente de los recursos con menores consumos energéticos, debido a que la aplicación se encuentra centralizada y permite la escalabilidad en función del uso requerido.

REFERENCIAS

(1) Erb, B. (2012). *Concurrent Programming for Scalable Web Architectures*. Diploma Thesis.
(2) Gracia, J., Bayo, E. (2013). Integrated 3D Web Application for Structural Analysis Software as a Service. *Journal of Computing in Civil Engineering*, 27(2): 159-166, doi: http://dx.doi.org/10.1061/(ASCE)CP.1943-5487.0000217.
(3) Richardson, L., Ruby, S. (2007). *RESTful Web Services*. USA: O'Reilly Media.
(4) Chuan-Jun, S., Chang-Yu, Ch. (2012). Enabling successful Collaboration 2.0: A REST-based Web Service and Web 2.0 technology oriented information platform for collaborative product development. *Computers in Industry*, 63(9): 948-959, doi: http://dx.doi.org/10.1016/j.compind.2012.08.018.
(5) Fielding, R. T., Taylor, R. N. (2002). Principled design of the modern Web architecture. *ACM Transactions on Internet Technology*, 2 (2): 115-150, doi: http://dx.doi.org/10.1145/514183.514185.
(6) Wang, G. (2011). Improving Data Transmission in Web Applications via the Translation between XML and JSON. *Third International Conference on Communications and Mobile Computing*, doi: http://dx.doi.org/10.1109/CMC.2011.25.

* * *