Componentes de una solución integradora (JSP)

Software

Hoy en día el desarrollo de software está enfocado al manejo de requerimientos cambiantes. El proceso de desarrollo debe permitir, no solo la adaptación al cambio, sino también la reducción del esfuerzo, el tiempo y los costos, a la vez que debe promover la escalabilidad, fiabilidad y reutilización del software. Es por esto que el desarrollo de software basado en componentes surge como una línea de la ingeniería del software que construye y utiliza técnicas para la implementación de sistemas abiertos y distribuidos mediante el ensamblaje de partes reutilizables.

Antecedentes

Los avances en la programación paralela y concurrente, junto con los requerimientos de integración de información localizada en distintos sitios geográficos dieron lugar al desarrollo de la tecnología para la construcción de aplicaciones distribuidas. Esto sumado al auge de Internet potenció el desarrollo de sistemas descentralizados, abiertos y distribuidos.

La necesidad de dividir el sistema de software en partes relacionadas y residentes en distintas máquinas, junto con el principio de reutilización de software existente para construir aplicaciones empresariales, dio como resultado un nuevo paradigma de programación: el desarrollo de software basado en componentes, que propone metodologías y tecnologías para la construcción de aplicaciones mediante el ensamblaje de piezas de software reutilizables.

¿Qué es un componente de software?

"Un componente es una unidad binaria de composición de aplicaciones software, que posee un conjunto de interfaces y un conjunto de requisitos, y que ha de poder ser desarrollado, adquirido, incorporado al sistema y compuesto con otros componentes de forma independiente, en tiempo y espacio".

Un componente de software tiene un ciclo de vida. Según Cheesman y Daniels se identifican 5 diferentes “visiones” en las que el término componente aparece en las etapas de desarrollo de software:
- Especificación del componente: Define la unidad de software que describe el comportamiento (interfaces) del componente y la unidad de implementación. La implementación realiza la especificación.

- Interfaz del componente: Define el conjunto de operaciones ofrecidas por un objeto componente.

- Implementación del componente: Realización de la especificación del componente que puede ser implantada, instalada y reemplazada de forma independiente en uno o más archivos.

- Componente instalado: Copia instalada de una implementación del componente.

- Objeto componente: Instancia de un “componente instalado”. Es un objeto con su propio estado e identidad que ejecuta el comportamiento implementado. Un componente instalado puede tener múltiples objetos componente o uno solo.

ARQUITECTURA MULTICAPAS

Una arquitectura multicapas promueve la separación de responsabilidades por capas. En el caso particular de 3-capas se separa la presentación de la lógica de negocio y ésta de la de datos. La capa de presentación no accede directamente a la base de datos, sino que lo hace únicamente a través de la capa de negocio. La arquitectura multicapas introduce muchas mejoras importantes dentro del diseño de la aplicación, incluyendo la flexibilidad a través de una adecuada separación entre la capa de presentación y la lógica de negocio.

La arquitectura n-capas es una generalización de la arquitectura 3-capas, en la que se incluye una capa adicional, responsable de la interacción con el usuario, y capas adicionales para manejar la conexión a la fuente de datos. En la figura 2 se ilustran las capas típicas de una arquitectura multicapas:

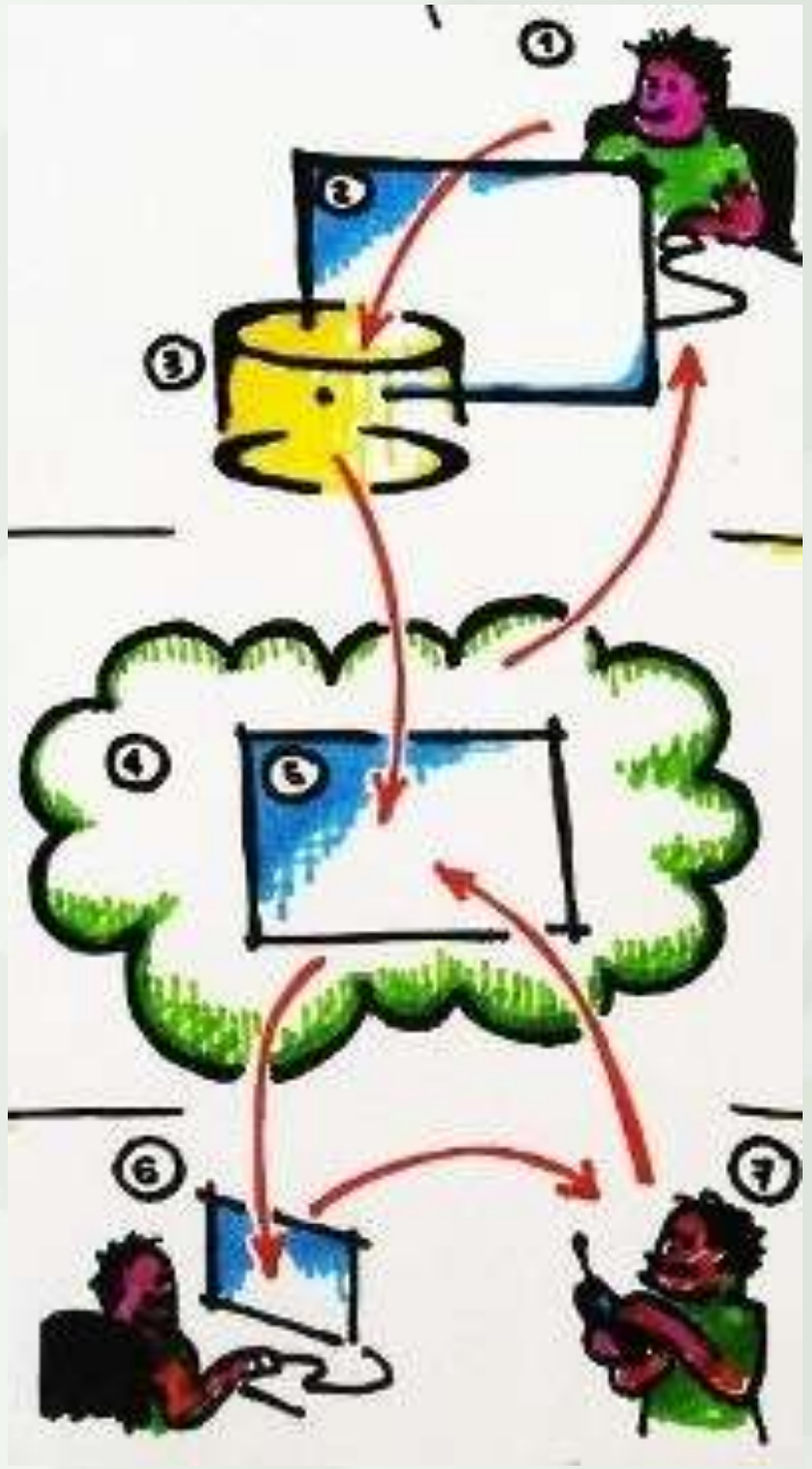
Capa Cliente: Es la responsable de la interacción entre el sistema y el usuario a través de una presentación específica. La separación con la capa de negocio permite manejar diferentes tipos de interfaz de usuario, como un browser web o un dispositivo móvil.

Capa de presentación: Permite al sistema soportar múltiples interacciones con un único usuario. Coordina y mantiene una sesión de usuario, manipulando los datos asociados con la sesión y con la interacción con la capa de negocio; coordina y mantiene la integridad con múltiples actividades para el mismo usuario; ejecuta procesos que no requieren acceso a los recursos empresariales. Es de aclarar que no hay una instancia de los recursos de negocio para cada usuario, sino que hay sólo una instancia de la capa de negocio y de datos que es compartida por todos los usuarios del sistema.

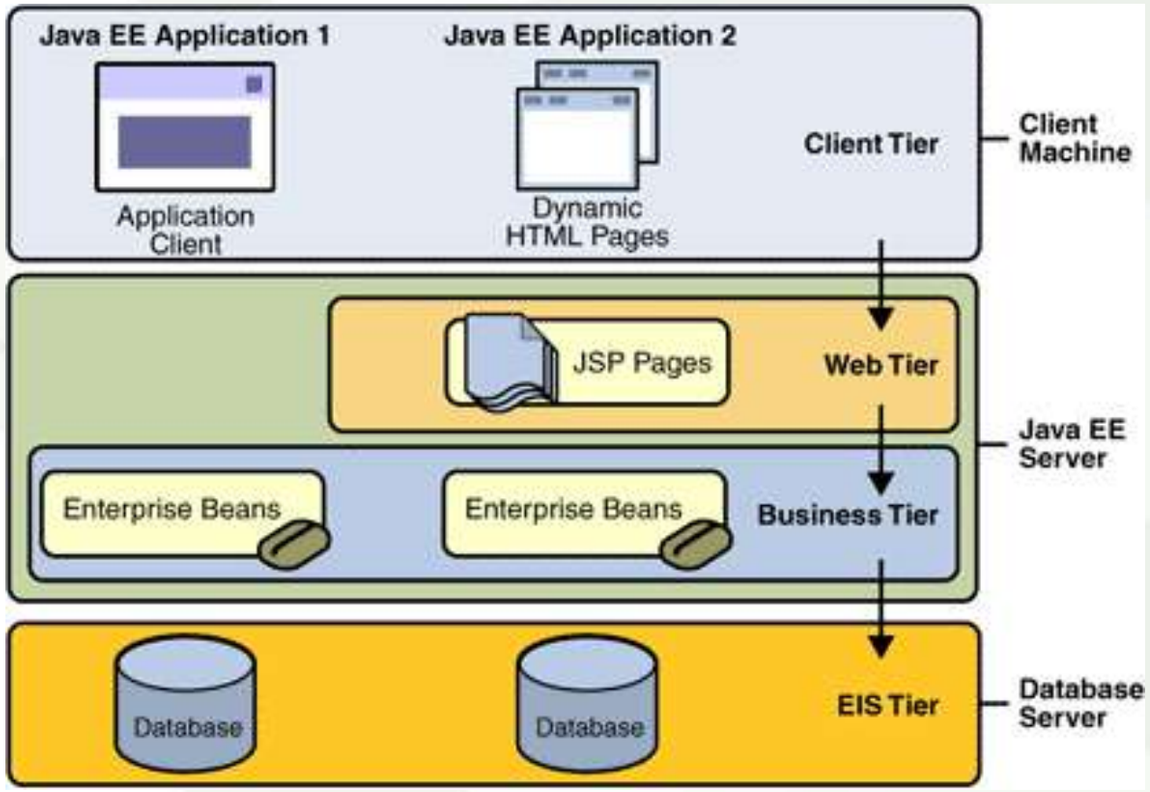
Capa de negocio: Es responsable de implementar los procesos y las entidades de negocio, y de hacer sus funciones disponibles a través de interfaces. Mantiene la integridad y fuerza el cumplimiento de reglas de negocio; maneja el control de transacciones. Provee un único punto de acceso a los recursos empresariales de manera que estos pueden ser compartidos y reutilizados por múltiples aplicaciones y usuarios.

Capa de integración: Es la responsable del manejo y acceso a los recursos empresariales compartidos. Provee acceso a las fuentes de datos y sistema legados2.

Capa de datos: Fuentes de datos con cualquier modelo (sistemas de bases de datos o archivos de datos) y sistemas legados.



ARQUITECTURA JEE



En el mundo de la tecnología de la información, las aplicaciones empresariales deben ser diseñadas, construidas y producidas con menos dinero, con mayor rapidez y con menos recursos . Además de estas restricciones, las aplicaciones deben integrar sistemas heterogéneos y distribuidos, y considerar el escalamiento y la reutilización como parte de la solución.
Desarrollar este tipo de aplicaciones requiere bastante esfuerzo porque se deben escribir muchas líneas de código para el manejo transaccional, la concurrencia, la asignación de recursos, y otra serie de actividades complejas de bajo nivel de detalle. Estas condiciones han llevado a distribuir responsabilidades desde el punto de vista técnico y de desarrollo, delegando tareas al servidor de la aplicación, de tal modo que el desarrollador se pueda centrar en la implementación de la lógica del negocio..

CAPAS DE LA ARQUITECTURA JEE

- Capa cliente: Puede estar compuesta de un cliente web o una aplicación cliente.
- Cliente web: Es un “cliente delgado” que interactúa con el usuario; usualmente no accede directamente a la fuente de datos ni ejecuta lógica compleja, sino que accede mediante un browser a los componentes de negocio que se encuentran del lado del servidor de la aplicación. Con un cliente delgado, la responsabilidad de la seguridad, desempeño y confiabilidad se le delega a tecnologías del lado del servidor JEE.
-Aplicación cliente: Provee una interface GUI escrita en AWT o Swing. Puede acceder a la capa web del servidor comunicándose con un servlet a través de HTTP.
- Capa web: Es la capa de presentación. Puede incluir
- Servlets: Son clases java que procesan dinámicamente solicitudes (requests) y construyen respuestas (responses).

CAPAS DE LA ARQUITECTURA JEE

- Páginas JSP: Son documentos basados en texto que se ejecutan como los servlets pero procesan de una manera más natural el contenido estático.

- Java Server Faces: Es una tecnología que se construye sobre servlets o JSP y que posee un conjunto de componentes GUI para aplicaciones web.

- Capa de negocio: Maneja la lógica de negocio a través de los componentes Enterprise Java Bean EJB. Un componente EJB puede ser de uno de los siguientes tipos:

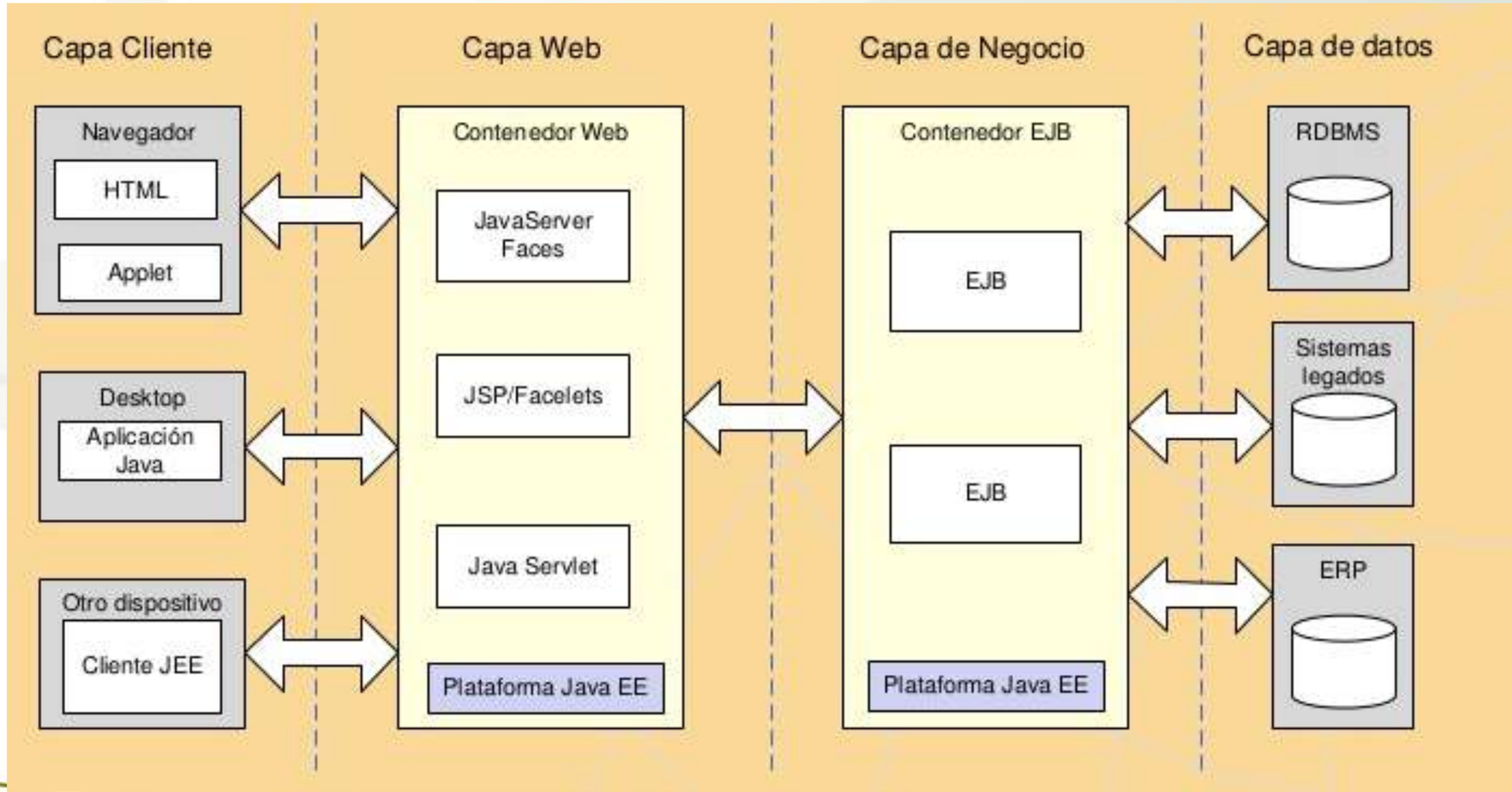
- Sesión bean: Representa una conversación persistente con el cliente. Encapsula la lógica del negocio que puede ser invocada programáticamente por un cliente local, remoto o un servicio web. Cuando el cliente termina la ejecución, el Session bean y sus datos son descartados.

CAPAS DE LA ARQUITECTURA JEE

- Entity bean: Representa datos persistentes en una base de datos. Si la sesión termina o el servidor falla, los servicios subyacentes aseguran que el entity bean sea guardado.

- Message bean: Combina funciones de un Session bean y Java Masaje Services (JMS) permitiendo que los componentes del negocio reciban mensajes asincrónicamente.

- Capa de conectividad y de recursos: Compuesta por el software e infraestructura para manejar persistencia. Incluye sistemas manejadores de bases de datos relacionales (RDBMS) como Oracle, Mysql, SysBase o cualquier otro, sistemas legados (Cobol) o sistemas de archivos (Excel).



https://es.slideshare.net/fernando_jmt/aplicaciones-empresariales-java-ee-en-la-nube

SERVIDOR JEE

Provee servicios subyacentes y transversales a una aplicación como seguridad, transaccionalidad, servicio de nombres y conectividad remota (acceso a recursos remotos como si estuvieran en la misma máquina virtual). También maneja servicios no configurables como el ciclo de vida del componente, el pool de conexiones con la base de datos y diferentes opciones de persistencia.
El servidor JEE provee un contenedor web para el manejo de la ejecución de las páginas web y servlets, y un contenedor para la ejecución de los EJB's. El proceso de despliegue de la aplicación se encarga de instalar cada componente dentro de su contenedor.
Algunos ejemplos de servidores JEE son *Java System Application Server, Oracle Application Server, Jboss, GlassFish y WebLogic Server.*

FRAMEWORKS DE DESARROLLO QUE SE INTEGRAN CON LA ARQUITECTURA JEE

Con el fin de facilitar la construcción de aplicaciones por componentes, se han desarrollado algunos frameworks compatibles con la especificación JEE. En esta sección se incluyen los utilizados en la implementación del caso de estudio.

Java Server Faces

Es un framework de la capa web basado en el patrón MVC (Modelo Vista Controlador), que utiliza servlets o páginas JSP y que incluye un conjunto rico de componentes GUI de alto nivel que encapsulan elementos HTML. Utiliza clases java denominadas Backing Beans para procesar eventos e invocar los métodos de los componentes EJB de negocio. JSF es un framework de presentación robusto que permite manejar características adicionales como reglas de navegación entre las páginas, validadores en los componentes GUI y soporte a la internacionalización.

Java Server Faces

JSF tiene diferentes implementaciones dentro del software libre. Una de ellas es Richfaces, que además de las características anteriores, soporta funcionalidades Ajax (Asynchronous JavaScript And XML) para mantener una comunicación asíncrona con el servidor de aplicaciones de manera que se pueden hacer cambios en las páginas sin necesidad de recargarlas completamente, mejorando así el desempeño de la aplicación.

Bibliografía

NIETO, ALBA CONSUELO. (2014). ARQUITECTURA POR COMPONENTES JEE, UN CASO PRÁCTICO. JEE COMPONENT ARCHITECTURE, A CASE STUDY
http://revistas.uis.edu.co/index.php/revistagti/article/view/4867/5669