Universidad de las Ciencias Informáticas

“CEIGE”

Título: “Propuesta de principios tecnológicos para el Marco de Trabajo en el desarrollo de Cedrux”

Trabajo de Diploma para optar por el título de Ingeniero en Ciencias Informáticas

Autor: Pedro Rodríguez Valdés

Tutor: MSc. Osmar Leyet Fernández

“Junio del 2012”

"Hay una fuerza motriz más poderosa que el vapor, la electricidad y la energía atómica: la voluntad"

Albert Finstein

Declaración de autoría

Declaro ser autor de la presente tesis y reconozco a la Universidad de las Ciencias Informáticas los derechos patrimoniales de la misma, con carácter exclusivo.

Para que así conste firmo la presente a los 28 días del mes de junio del año 2012.

_____

Pedro Rodríguez Valdés

_____

MSc. Osmar Leyet Fernández

Datos de contacto

Síntesis del Tutor:

MSc. Osmar Leyet Fernández: Profesor Instructor.

Miembro del equipo de implementación del sistema Saren, del proyecto Registros y Notarías.

Analista Principal del sistema de Comercio electrónico B2B para la empresa Cubalse de la república de Cuba.

Arquitecto de Tecnología del sistema Cedrux, del proyecto ERP Cubano.

Jefe de la línea de Contabilidad y Finanzas del sistema Cedrux, del proyecto ERP Cubano.

Jefe de departamento “Desarrollo de Productos” del centro CEIGE.

Agradecimientos

“Me gustaría agradecer a la Universidad de las Ciencias Informáticas por haberme dado la oportunidad de realizar mis sueños de hacerme ingeniero. Quiero agradecer especialmente a nuestro comandante en jefe Fidel Castro Ruz por haber tenido esta maravillosa idea que le ha traído tantas cosas buenas a la vida de muchos jóvenes. A mi esposa que me ha apoyado y guiado en el trabajo. Sería importante agradecerle a mis compañeros de carrera por haberme soportado todos estos años y a los profesores que tantas cosas nuevas me han transmitido, tanto desde el punto de vista profesional como personal, en fin a todas las personas que me han ayudado en el transcurso de mi carrera. A mis padres por apoyarme incondicionalmente y familiares que me han exhortado siempre a dar más de mi.”

Dedicatoria

“Quiero dedicarle esta tesis especialmente a mis padres, gracias a los dos, sin Uds. nunca lo hubiera logrado, les prometí que cuando fuera grande iba a ser ingeniero, aquí está mi regalo. A mi esposa y bebe, los amo. Además a toda mi familia que me ha ayudado de una forma u otra para completar mis sueños, gracias.”

Resumen

La gestión de la información en las entidades cubanas debe comenzar a ser controlada por sistemas informáticos como los sistemas de planificación de recursos empresariales (Enterprise Resource Planning) ERP, con el objetivo de mantener un mejor control y uso de los recursos que el estado pone en manos de esas empresas. La dirección del país le dio a la Universidad de las Ciencias Informáticas y la Unidad de Compatibilización, Integración y Desarrollo de Software para la Defensa, la tarea de desarrollar un ERP que cuente con todos los módulos necesarios para la gestión empresarial del país. Para el desarrollo de este ERP conocido como Cedrux se utiliza una herramienta informática llamada Sauxe, un marco de trabajo extendido de utilidades como el ZendFramwork, Doctrine y ExtJS. El marco de trabajo Sauxe presenta problemas en la gestión de transacciones y carece de un módulo para la gestión de avisos y alertas internas. Proponer un conjunto de soluciones en el marco de trabajo Sauxe para la adecuada gestión de las transacciones y alertas en el sistema Cedrux es el objetivo trazado para esta investigación. La propuesta está validada a través del método Delphi, donde se concluye que el nivel de concordancia entre los expertos es elevado, coincidiendo en caracterizar la propuesta como novedosa y arrojando una alta posibilidad de futura implantación de la misma.

Palabras Clave

Componente, Cedrux, Delphi, gestión de avisos y alertas, gestión de transacciones, marco de trabajo, Sauxe, subsistema.

Índice

Índice de Figuras

Índice de Tablas

Introducción

Cuba necesita que la informática ocupe el primer lugar en la gestión de los procesos y la eficiencia en sus empresas, con el objetivo de mejorar el control de los recursos que el estado pone a su disposición; una de las herramientas utilizada para ello son los sistemas de planificación de recursos empresariales ERP[1]. Los ERP permiten gestionar procesos de una empresa con el objetivo de integrar información a lo largo de las entidades y eliminar las barreras entre diferentes áreas, contribuyendo a la mejora de los procesos fundamentales.

Debido al alto costo a nivel internacional de los sistemas ERP, sus licencias y mantenimientos (INFORMATICA HOY, 2012), se le dio a la Universidad de las Ciencias Informáticas (UCI) en conjunto con la Unidad de Compatibilización, Integración y Desarrollo de Software para la Defensa (UCID) la tarea de desarrollar un ERP que cuente con los módulos necesarios para la gestión empresarial del país. La magnitud, complejidad y seguridad de un sistema de este tipo requiere hacer un diseño arquitectónico potente que soporte la gestión y transferencia de información. Para ello se creó un grupo central de Arquitectura que fue definiendo por cada capa, los componentes que formarían parte de la línea base. De estos componentes, algunos fueron reutilizados y otros extendidos de Zend Framework para dar lugar al marco de trabajo de Cedrux que adoptó el nombre de Sauxe. (Baryolo Gómez, y otros, 2009)

Un marco de trabajo es un conjunto de componentes que forman un diseño reutilizable que facilita y agiliza el desarrollo de sistemas informáticos. Los objetivos principales que persigue son: acelerar el proceso de desarrollo, reutilizar código ya existente y promover buenas prácticas de desarrollo como el uso de patrones. (Gutiérrez, 2009)

Sauxe contiene un conjunto de componentes reutilizables que provee la estructura genérica y el comportamiento para una familia de abstracciones, logrando una mayor estandarización, flexibilidad, integración y agilidad en el proceso de desarrollo. A medida que el sistema Cedrux avanza, mayores son los requerimientos que debe cumplir la arquitectura para soportar y brindar las prestaciones requeridas. Sauxe continúa desarrollando tecnologías robustas y cada vez son más los interesados en aprenderlas y trabajar sobre esta arquitectura.

Por el nivel de integración, asimilación y robustez de este marco de trabajo se encuentra generalizado en 15 proyectos de la UCI (Aduana, Sistema de Gestión Académica de Pregrado, Sistema para la gestión de las transportaciones militares en las FAR(DITRANS), Fuerza de Trabajo Calificada (FTC), Generador de reportes, Sistema de supervisión y control de los PSI (Hoyo), Informatización UCID, Plataforma Soberana para el Desarrollo de Sistemas de Información Geográfica (GENESIG), Minería de datos, (SEUNE), Sistema Informático para la Gestión de Auditoría y Control SIGAC, Sistema de Mando y Estado Mayor (SIMEM), Sistema de gestión estadística, Sistema de Registro, Control y Análisis de los Medios y Equipos de la Reserva Militar(SRCAMERM), el MINFAR y otras entidades del país como el Centro de Desarrollo de Holguín alcanzando resultados satisfactorios. (Baryolo Gómez, y otros, 2009)

[1] Enterprise Resource Planning (Planificación de recursos empresariales)

Aún así, Sauxe sigue presentando dificultades en elementos como el tratamiento de transacciones, que presenta inconvenientes a la hora de rectificar errores ocurridos en las transacciones y puede ocasionar inconsistencia en los datos modificados o requeridos por el usuario; y la ausencia de un módulo para el manejo de avisos y alertas de los productos que soporta el marco, que acarrea problemas de integración entre componentes, falta de información oportuna y actualizada a través de notificaciones en tiempo real; elementos que ayudarían a que el equipo obtenga mejores resultados en la implementación y uso del sistema en general. Para la comprensión del problema y la búsqueda de posibles soluciones o mejoras, se deben plantear las causas que lo originan, se puede afirmar que la excesiva modularidad de los distintos subsistemas que conforman el sistema Cedrux es una de las principales causas de los errores en las transacciones, modularidad insuficiente porque si se retira uno de los subsistemas, alguno de los otros no funcionaría correctamente, al igual que con sus componentes.

Dada la situación presentada anteriormente se plantea el siguiente **problema**: Los recursos tecnológicos que propone el marco de trabajo Sauxe para el desarrollo de un sistema ERP carece de soluciones para la adecuada gestión de las transacciones y alertas del sistema.

El **objeto de estudio** sería los marcos de trabajo con tecnologías PHP.

Se persigue como **objetivo general:** Proponer un conjunto de soluciones en el marco de trabajo Sauxe para la adecuada gestión de las transacciones y alertas en el sistema Cedrux.

El **Campo de acción** quedaría enmarcado en la gestión de transacciones y alertas en marcos de trabajo con tecnología PHP.

Para dar cumplimiento al objetivo general se trazaron los siguientes **objetivos específicos**:

- Elaborar un marco teórico relativo a los marcos de trabajo tecnológicos para el desarrollo de sistemas ERP.
- Proponer un conjunto de componentes tecnológicos acorde a las características del marco de trabajo Sauxe.
- Evaluar la propuesta realizada mediante el criterio de especialistas.

Los **aportes prácticos** esperados son:

Enriquecer el marco de desarrollo Sauxe, empleado para la construcción del sistema de planificación de recursos empresariales Cedrux con nuevas soluciones tecnológicas.

Para el desarrollo de este trabajo se usaron los siguientes métodos teóricos:

Histórico - lógico: Posibilitó estudiar de forma analítica la trayectoria histórica real de los fenómenos, su evolución y desarrollo. El método permitió realizar la primera parte de la investigación, al hacer un análisis bibliográfico de otros marcos de trabajo para revisar el uso de transacciones, estructura de sus módulos o componentes, el razonamiento y técnicas de programación más utilizadas en el desarrollo de este tipo de sistemas. Se utilizó para determinar a través de la evaluación de la bibliografía conceptos de estas

temáticas, que permiten conocer el estado de la evolución del fenómeno e identificar posibles mejoras y alternativas de solución.

Métodos empíricos utilizados:

La observación: Se usa durante todo el desarrollo de la investigación para mantener una visión del desarrollo de la misma, ver el grado de avance de la propuesta de solución y arribar a soluciones. En el presente trabajo se utilizó fundamentalmente para la fundamentación teórica de la investigación.

La encuesta: Se realiza cuando la información que se necesita puede ser obtenida a partir de la respuesta que una o varias personas puedan dar a un cuestionario pre elaborado y las mismas están dispuestas a colaborar con la investigación. Las encuestas se utilizaron en el capítulo tres para obtener evaluaciones y criterios de los especialistas en desarrollo de sistemas informáticos sobre las soluciones dadas.

Estructura de los capítulos de la tesis

La tesis está compuesta por tres capítulos.

Capítulo 1. Fundamentación Teórica.

Estado del arte de los principales marcos de trabajo PHP, Sauxe y sus principales características, principales herramientas a estudiar para la realización de las propuestas. Se identifican además los principales problemas a tratar en la investigación.

Capítulo 2. Propuestas de soluciones.

Se elaboran las propuestas de solución a los problemas tecnológicos identificados.

Capítulo 3. Validación de las propuestas.

Se validan las soluciones propuestas a través del método Delphi.

Capítulo 1

1. Fundamentación Teórica.

1.1 Introducción

En este capítulo se presentará un estudio del estado del arte, de los marcos de trabajo de interés para el tema, sus características, los componentes fundamentales que conforman el marco de trabajo Sauxe, así como algunas herramientas necesarias para la comprensión de las soluciones que se presentarán en la investigación.

1.2 Marcos de Trabajos

Los marcos de trabajo son desarrollados con el objetivo de brindarles a los programadores y diseñadores una mejor organización y estructura a sus proyectos. En sus inicios para llevar a cabo el desarrollo de Sauxe, se hizo necesario realizar un estudio de los Marcos de trabajos más reconocidos mundialmente por las facilidades que brindan a la hora de optimizar código y estandarizar la programación de una manera eficiente. Entre los principales y más usados se destacan: Symfony, CakePHP, CodeIgniter KumbiaPHP y Zend Framework. De ellos, el centro de desarrollo CEIGE seleccionó el Zend Framework debido a que brinda facilidades para desarrollar aplicaciones y servicios web, además por su facilidad a la hora de extender sus componentes.

1.3 Zend Framework

Se trata de un Marco de Trabajo para el desarrollo de aplicaciones y servicios Web con PHP, brinda soluciones para construir sitios web modernos, robustos y seguros. Este Marco de Trabajo está formado por una serie de métodos estáticos y componentes que usarán estos métodos. Como características fundamentales de Zend Framework están (Leopoldo, 2007):

- Trabaja con Modelo Vista Controlador (MVC).
- El Marco de Zend también incluye objetos de las diferentes bases de datos, por lo que es extremadamente simple para consultar su base de datos, sin tener que escribir ninguna consulta SQL.
- Es una solución para el acceso a base de datos que balancea el Mapeador de Objeto Relacional con eficiencia y simplicidad.
- Completa documentación y test de alta calidad.
- Soporte avanzado.
- Clases robustas para autenticación y filtrado de entrada.
- Muchas otras clases útiles para hacerlo tan productivo como sea posible.

Los 51 componentes de Zend Framework conforman un potente y extensible marco de trabajo de aplicaciones web al combinarse entre sí. De todos estos Sauxe utiliza solamente los siguientes (Baryolo, 2008):

1. Zend_Cache
2. Zend_Config
3. Zend_Controller
4. Zend_Loader
5. Zend_Log
6. Zend_Registry
7. Zend_Session
8. Zend_View

A partir de la extensión de algunos componentes de Zend Framework surge Zend ExtJS, desarrollado por el departamento de tecnología del CEIGE y la UCID, con el objetivo de crear un marco de trabajo extensible y configurable centrando el desarrollo de las aplicaciones, en la lógica del negocio, en las interfaces de usuario, con soporte para entornos multientidad y para una arquitectura de sistema orientada a componentes**.** Una aplicación que se hacía necesaria para el desarrollo de sistemas como los ERP.

1.4 Marco de trabajo Sauxe

Sauxe es un marco de trabajo que contiene un conjunto de componentes que provee la estructura genérica y el comportamiento para una familia de abstracciones, logrando una mayor estandarización, flexibilidad, integración y agilidad en el proceso de desarrollo (Baryolo, 2008). La unión de Zend ExtJS y Doctrine dio lugar al marco de trabajo Sauxe.

Sauxe está formado por los componentes: ZendExt_App, ZendExt_Aspect, ZendExt_ADT, ZendExt_Cache, ZendExt_ExpImp, ZendExt_MVC, ZendExt_Exception, ZendExt_FastResponse, ZendExt_GlobalConcept, ZendExt_IoC, ZendExt_Nomencladores, ZendExt_Trace, ZendExt_Validation, ZendExt_Portal, ZendExt_TransactionManager. Este último es el responsable de gestionar las transacciones dentro del marco de trabajo Sauxe.

1.4.1 Aspectos arquitectónicamente significativos

Los métodos del diseño arquitectónico coinciden en la importancia de tomar en cuenta los aspectos de calidad para dirigir la selección de la solución arquitectónica, sin embargo aún no hay acuerdos sobre métodos precisos que puedan ser utilizados en la práctica común. Distintas propuestas de métodos de diseño arquitectónico se han presentado hasta el momento, los mismos son fundamentados en conceptos que bien son equivalentes, complementarios o alternativos (Jacobson, 2000). Por tanto debido a la carencia de herramientas, métodos, o acuerdos que cumplan con estas características se hace necesario

crear herramientas capaces de gestionar los aspectos que van a dar solución a la arquitectura base de Cedrux. (Equipo Arquitectura del ERP Cedrux, 2008)

A continuación se hace un análisis de los principales aspectos arquitectónicamente significativos de este marco de trabajo.

Caché:

La caché no es más que un conjunto de información que generalmente se almacena en ficheros, tanto en clientes como en servidores web en un formato determinado, de forma tal que pueda ser recuperada rápida y eficientemente, permitiendo almacenar información de cualquier tipo, textos, clases, funciones, objetos o instancias de clases, estructuras complejas como listas, pilas, árboles, arreglos o tablas hash, páginas, ficheros, por lo general esta información no debe cambiar en un período de tiempo determinado. En caso de que alguna información almacenada en la caché cambie o expire debe ser actualizada instantáneamente, para evitar problemas de inconsistencia. La mayoría de los gestores de caché almacenan esta información en forma serializada para ganar en rendimiento y espacio de almacenamiento, un ejemplo de ello es el trabajo con las sesiones en PHP, que es una caché con estas características pero que se almacena en un servidor web para un cliente web específico. (Equipo Arquitectura del ERP Cedrux, 2008)

La ventaja principal de usar un mecanismo para el almacenamiento de la información que cambia poco como la configuración es que puede mejorar considerablemente el rendimiento de la aplicación, evitando tener que construir por ejemplo estructuras complejas, configuraciones, e incluso realizar peticiones innecesarias a bases de datos y aumentando el uso concurrente de recursos que varían poco en el tiempo, además puede ser usada con otros fines como la comunicación entre sesiones. Actualmente el marco de trabajo del proyecto Cedrux y de la UCID no cuenta con un componente para la gestión de la caché y debido a ello se decidió desarrollar, adaptar o integrar un componente al marco de trabajo con este objetivo. (Equipo Arquitectura del ERP Cedrux, 2008)

Trazas:

Las trazas permiten en el desarrollo de software crear un mecanismo de registro oficial de eventos durante un período de tiempo en particular, además de registrar datos o información sobre quién, qué, cuándo, dónde y por qué un evento ocurre para un dispositivo en particular o aplicación. Todo esto permite monitorear las actividades de la aplicación o dispositivo, donde se puede obtener una buena oportunidad para determinar eventos y tomar la acción necesaria para corregir el problema o iniciar una investigación en caso de un incidente de seguridad. (Equipo Arquitectura del ERP Cedrux, 2008)

Excepciones:

Los programadores de cualquier lenguaje se esfuerzan por escribir programas libres de errores, sin embargo, es muy difícil que los programas reales se vean libres de ellos. Aún cuando una sentencia o expresión sea sintácticamente correcta, puede causar un error al intentar ejecutarla. Una excepción es un evento que ocurre durante la ejecución del programa que interrumpe el flujo normal de las sentencias (ISIDRO, 2010). Las excepciones están destinadas para la detección y corrección de errores. Si hay un error, la aplicación no debería morirse y generar un core (o un crash en caso del DOS). Se debería lanzar

una excepción (throw) que se debería capturar (catch) y resolver la situación del error. Utilizadas en forma adecuada, las excepciones aumentan en gran medida la robustez de las aplicaciones. (Equipo Arquitectura del ERP Cedrux, 2008)

El sistema de gestión Cedrux presenta estos mismos problemas pero no basta con el mero hecho del lanzamiento y captura de excepciones sino también se imponen algunos requerimientos descritos a continuación (Equipo Arquitectura del ERP Cedrux, 2008):

- La internacionalización de las excepciones.
- Manejo declarativo del comportamiento de las excepciones.
- Mecanismo extensible de creación de nuevos tipos de excepciones.
- Codificación de las excepciones.
- Tratamiento igualitario de las excepciones propias o de los marcos de desarrollo subyacentes (Doctrine, Zend y EzComponent) que ya poseen su propia forma de gestionarlas. (Equipo Arquitectura del ERP Cedrux, 2008)

Validaciones:

La seguridad de las aplicaciones web es un tema muy importante ya que en esta se manejan informaciones de muchas instituciones, y si personal no autorizado tuviese acceso a ellas podría causar daños irreparables. Dada la importancia del ERP y toda la información que maneja se decidió prestar el máximo de interés a las validaciones de cada una de las aplicaciones que conforman este software. En la mayoría de las aplicaciones sólo se valida en la capa de presentación, esto trae como consecuencia que si el desarrollador hizo una mala validación o el atacante logra violar esta capa entonces tiene acceso total a la información almacenada. Como la seguridad se basa en la cantidad de barreras que un desarrollador implementa, desde la vista que se le muestra al usuario hasta los datos, se decidió desarrollar un componente para validar en el servidor todas las acciones que requiera de ellas. (Equipo Arquitectura del ERP Cedrux, 2008)

Este componente debe validar todos los datos y acciones que se activan ya sean por un usuario o por un sistema externo, que serán ejecutadas en un controlador específico de una aplicación. La acción será validada antes de llegar al controlador para no hacer peticiones innecesarias a este. (Equipo Arquitectura del ERP Cedrux, 2008)

Conceptos Globales:

Los contenedores de variables globales son aspectos muy importantes en los marco de trabajo ya que agrupan información común conceptualizada que se maneja en todo el sistema, o sea en todos los módulos y sesiones de una aplicación. Por lo general todo marco de trabajo que se implementa contiene un componente de este tipo para que cada parte del sistema pueda acceder de forma fácil y sencilla a datos que oferte otro sin incurrir en códigos engorrosos que implican gastos de tiempo y esfuerzos para los programadores.

Por lo tanto la implementación de estos contenedores de información común permite que la misma sea almacenada una sola vez y actualizada en el momento conveniente, así cuando un componente necesite

de esta información la toma del contenedor sin tener que estar realizando peticiones constantes a los servicios que oferten los demás componentes. Los marcos de trabajo permiten que en el desarrollo de una aplicación quede implementado que en un determinado momento se guarde la información común en el contenedor. Posteriormente un módulo accede a la variable que almacena la información en el mismo tantas veces le haga falta, pero internamente se invoca una sola vez al servicio de la parte que oferta la misma.

Un ejemplo de esto es el manejador de sesiones que implementa CodeIgniter, que al inicio de la aplicación permite implementar de manera fácil que sean guardados entre otros datos los del usuario que ejecuta el sistema y los coloca en las cookies del navegador, de esta forma es fácil para los módulos acceder a estos datos cuantas veces lo necesiten sin derrochar tiempo durante la ejecución de la aplicación.

En el marco de trabajo que se desarrolla en el seno del equipo de arquitectura del ERP (Cedrux) para que un módulo pueda acceder a información que oferta otro, debe pedírselo a través de un servicio cada vez que lo necesite, esto trae consigo muchas peticiones entre los módulos y por lo tanto genera una lentitud mayor cuando se ejecuta cada componente del sistema, ya que se generan un sin número de peticiones concurrentes desde cada módulo a determinadas partes de la aplicación que ofertan información común a cada fragmento del sistema. (Equipo Arquitectura del ERP Cedrux, 2008)

Es por ello que se decide implementar un componente capaz de obtener toda la información común conceptualizada en variables para que cada módulo pueda acceder a la misma de forma rápida y sin generar lentitudes al sistema. Este es un componente de conceptos globales cuyo nombre es GlobalConcept. (Equipo Arquitectura del ERP Cedrux, 2008)

1.4.2 Herramientas que conforman el marco de trabajo Sauxe

Unos de los problemas existentes para la realización del ERP, estaba dado por la falta de operaciones o interoperabilidad entre los componentes. A raíz de ello se comenzó a estudiar y buscar patrones desde la base conceptual del desarrollo basado en componentes. Entre las posibles soluciones encontradas para satisfacer los problemas existentes, se llevó a cabo un estudio de los Patrones de Integración, realizando un mayor énfasis dentro de estos en el Patrón de IoC y ServiceLocator. Permitiendo definir el trabajo para dar solución a las polémicas, mediante la utilización de Inyección de Dependencia, el cual es un Patrón de Diseño elaborado bajo el concepto de Inversión de Control. (Equipo Arquitectura del ERP Cedrux, 2008)

Se pensaba en un inicio que el problema de la gestión de transacciones estaba generado por el sistema de inyección de control (IOC), es por esto que se hace necesario su estudio para una mejor comprensión de la herramienta.

1.4.2.1 Inversión de control

Uno de los componentes fundamentales de Sauxe es el ZendExt_IoC, módulo de inversión de control del marco. La inversión de control es un principio abstracto que describe el diseño de una arquitectura de

software en la cual el flujo de control está invertido, en comparación con la arquitectura tradicional de "Librerías de software" (Fowler, 2006).

Tradicionalmente el programador especifica la secuencia de decisiones y procedimientos que pueden ocurrir durante el ciclo de vida de un programa mediante llamadas a funciones. En su lugar, en la inversión de control (IOC) se especifican respuestas deseadas a sucesos o solicitudes de datos concretas, dejando que algún tipo de entidad o arquitectura externa lleve a cabo las acciones de control que se requieran en el orden necesario y para el conjunto de sucesos que tengan que ocurrir. En general la IOC es un estilo de construcción de software donde, componentes reusables y genéricos controlan la ejecución de funciones específicas desde un ámbito exterior a estas. Con la importante connotación de que estas se encuentran diseñadas e implementadas por separado teniendo como resultado mayores posibilidades de mantenimiento y revisión del código (Fowler, 2006).

1.4.2.2 Funcionalidad del IoC

Se formaliza en el IoC en el fichero ioc.xml, el contrato (servicios) que brinda un determinado componente (o varios), permitiendo esto la comprensión de los servicios que pueda brindar el componente a realizarle la petición o encargado de responderla. La descripción del XML tiene un inyector: Que no es más que la función que permite que la clase interface que responde ha dicho servicio y describe el método que se va a ejecutar con sus parámetros y resultados. Dentro de cada componente a realizar la petición existe una carpeta donde se guardan las interfaces (que vienen definidas y tienen dentro operaciones y funciones que se van a ofrecer en cada servicio) ¿Cómo interactúan los componentes? Cuando un componente desea obtener información referenciada en otro componente, este invoca un contrato descrito en el XML (la clase IOC que es el contenedor del objeto), el mismo invoca la operación de la clase interfaz, recibe un resultado (objetos, atributos), y lo devuelve (inyecta) hacia el componente que solicitó la independencia. En caso de que el componente desee obtener una simulación de los valores referenciados de otro componente, entonces el IoC simula el resultado cumpliendo según la descripción guardada en el fichero XML sin conectarse al otro componente. (Equipo Arquitectura del ERP Cedrux, 2008)

Los especialistas del centro CEIGE en un principio pensaban que el problema de la gestión de transacciones estaba dado por la ineficiencia de la inyección de control, pero más adelante se dieron cuenta que el problema real estaba dado por la existencia de las llaves foráneas entre subsistemas que limitaba y hacia defectuoso el sistema de inyección de control que proporciona el Zend para el sistema Sauxe.

1.4.2.4 Marco de Persistencia de Objetos Relacionales

Persistencia

"Es la capacidad del programador para conseguir que sus datos sobrevivan a la ejecución del proceso que los creó, de forma que puedan ser reutilizados en otro proceso. Cada objeto, independiente de su tipo, debería poder llegar a ser persistente sin traducción explícita. También debería ser implícito que el usuario no tuviera que mover o copiar los datos expresamente para ser persistentes" (Castillo, 2009). El concepto

de persistencia está relacionado con el almacenamiento secundario de instancias de objetos. Un objeto se dice persistente cuando es almacenado en un archivo u otro medio permanente. Un programa puede grabar objetos persistentes y luego recuperarlos en un tiempo posterior. La persistencia de datos es sin duda alguna, una de las partes más importantes en el desarrollo de un software. En caso de que la aplicación esté basada en programación orientada a objetos, la persistencia se logra mediante la serialización de los objetos o su almacenando en una base de datos. La segunda opción es muy utilizada, pero también mucho más compleja, puesto que la mayoría de las bases de datos siguen un modelo relacional y este difiere en gran medida del modelo objetos (Castillo, 2009).

1.4.2.5 Marco de persistencia

Es una estructura de software orientada a mejorar la productividad a través de la reutilización de código. Es un conjunto de clases creadas para apoyar la escritura de código en un contexto determinado. *"Un marco de trabajo de persistencia es, por lo tanto, una librería de clases que facilita la tarea del programador al permitirle guardar objetos en bases de datos relacionales de manera lógica y eficiente, de otra manera tocaría hacerlo manualmente, y este es, potencialmente, un proceso tedioso, repetitivo y propenso a errores"* (Cadavid, 2008).

1.4.2.6 Marco de persistencia de objetos relacionales Doctrine

En la actualidad la mayoría de las bases de datos existentes siguen una estructura relacional, básicamente esto se traduce en un conjunto de tablas relacionadas compuestas por valores escalares. Sin embargo, la programación orientada a objetos (POO), muy difundida en los días de hoy, devino en su desarrollo posterior al surgimiento de la estructura relacional. Esto conlleva a que no sea factible el manejo de la estructura de esas bases de datos siguiendo una lógica de negocio orientada a objetos y sea necesario convertir este conjunto de tablas y relaciones en clases y objetos válidos para la programación. Como solución a dicha problemática surgen los ORM.

Doctrine

*"*Doctrine no es más que un ORM para la persistencia de datos que trabaja con lenguaje de programación PHP 5.2.3 o superior, situándose sobre su poderosa capa de abstracción de la base de datos (PHP DBAL). Es uno de los más conocidos que interactúa con este lenguaje y su utilización brinda grandes facilidades a los desarrolladores a la hora de su trabajo con los datos, automatizando la generación de ficheros que responden a las tablas relacionales de la base de datos" (Doctrine.org, 2012).

Principales características de Doctrine

Doctrine se divide en dos capas fundamentales que interactúan en conjunto, la DBAL del inglés Database Abstraction Layer o Capa de Abstracción de la Base de Datos y la compuesta por el ORM reflejadas en la siguiente imagen (SENCIOLABS, 2009).

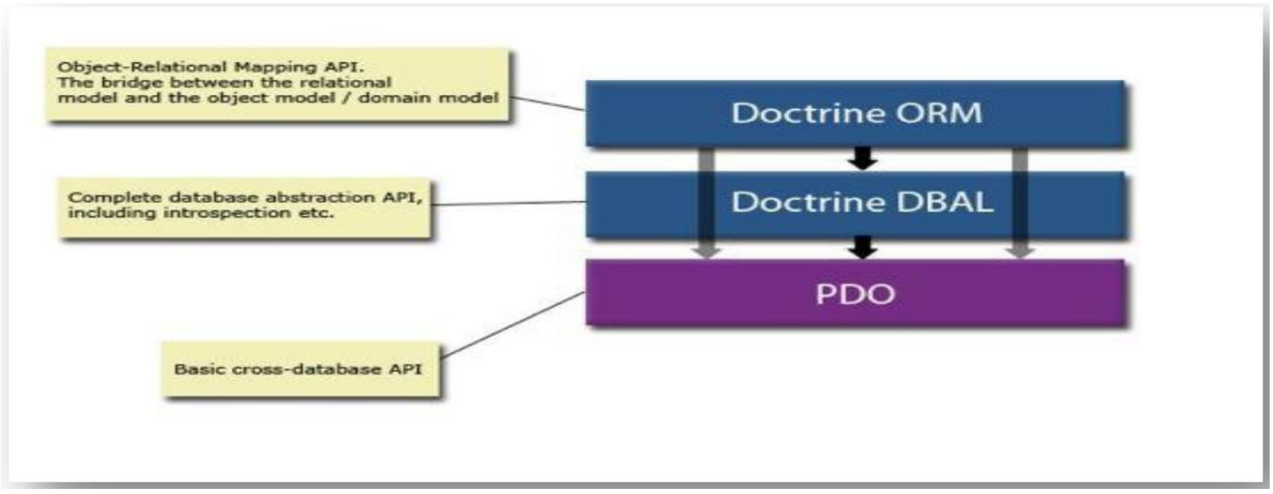

Figura 1 Estructura lógica del Marco de trabajo de persistencia Doctrine.

Doctrine es un marco de trabajo desarrollado básicamente sobre los principios de los patrones de Data Mapping (Mapeo de Datos), Meta Data Mapping (Mapeo de Metadatos) y Active Record (Registro Activo). Mediante una clase base nombrada Doctrine_Record, todas sus clases hijas asumen la interfaz común de Active Record (permite salvar, eliminar, actualizar.) facilitándole a Doctrine un sencillo uso y monitoreo del ciclo de vida de sus registros. Sin embargo, el trabajo de mapeo y creación de ficheros es mayormente reenviado a otros componentes como por ejemplo la clase Doctrine_Table (ServerGrove, 2010).

Este marco de trabajo tiene la facultad de construir consultas a la base de datos relacional utilizando un lenguaje OO (orientado a objetos) denominado DQL (Doctrine Query Language o Lenguaje de Consultas Doctrine) inspirado en el potente HQL (Hibernate Query Language) de Hibernate. Además, implementa un patrón CRUD (Crear, Recuperar, Actualizar y Eliminar) para operaciones comunes, ya sea desde crear un nuevo registro o actualizar los registros existentes. Automatiza la generación de clases PHP del modelo basado en YAML, y de sus respectivas relaciones desde una base de datos existente. Soporta varios motores de bases de datos e incluye dos tipos de herencia: Simple, para cuando todas las clases tienen las mismas columnas, y de agregación, que almacena un valor adicional en la tabla y permite la instanciación automática del tipo de modelo correcto cuando se realiza una consulta (ServerGrove, 2010).

Doctrine es capaz de aplicar varios comportamientos a los modelos, por ejemplo para un modelo Timestampable automáticamente se crearán 2 columnas: created_at y updated_at, las cuales guardarán la fecha de un registro creado y la de un registro actualizado respectivamente. Además, soporta opciones como Data fixtures (Instalación de Datos) que son utilizados para cargar datos de prueba a través de los modelos para poblar la base de datos con estos y realizarle pruebas, migraciones que permiten tener actualizadas las nuevas versiones de las bases de datos, cache mediante de un servidor memcache (Memoria cache) que facilita la rapidez y eficiencia de su trabajo, eventos con una flexible arquitectura de listeners (escuchadores) que no solo posibilita estar a la escucha de los posibles eventos que ocurran sino también alterar la ejecución de un determinado método, paginación e interfaz de línea de comando (ServerGrove, 2010).

Entre otros elementos se tiene la posibilidad de exportar una base de datos existente a sus clases correspondientes y también a la inversa, es decir convertir clases (convenientemente creadas siguiendo

las pautas del ORM) a tablas de una base de datos. Por otro lado, como la librería es bastante grande ésta tiene un método para ser “compilada” al pasar a producción (Baryolo, 2008).

Dentro de los ORM que dispone PHP uno de los más populares es Doctrine. En la actualidad, se dispone de una variada documentación acerca de su librería, elemento indispensable a la hora evaluar la usabilidad de cualquier técnica o herramienta.

Ventajas que facilitan enormemente tareas comunes y de mantenimiento

Reutilización

La principal ventaja que aporta un ORM es la reutilización permitiendo llamar a los métodos de un objeto de datos desde distintas partes de la aplicación e incluso desde diferentes aplicaciones (Mata, 2009).

Encapsulación

La capa ORM encapsula la lógica de los datos pudiendo hacer cambios que afectan a toda la aplicación únicamente modificando una función (Mata, 2009).

Portabilidad

Utilizar una capa de abstracción que permite cambiar en mitad de un proyecto de una base de datos MySQL (Sistema de gestión de base de datos relacional) a una Oracle sin ningún tipo de complicación. Esto es debido a que no se utiliza una sintaxis (MySQL, Oracle o SQLite) para acceder a nuestro modelo, sino una sintaxis propia del ORM utilizado que es capaz de traducir a diferentes tipos de bases de datos (Mata, 2009).

Seguridad

Los ORM suelen implementar mecanismos de seguridad que protegen nuestra aplicación de los ataques más comunes como una Inyección SQL (Mata, 2009).

Mantenimiento del código

Gracias al correcto ordenamiento de la capa de datos, modificar y mantener el código es una tarea sencilla (Mata, 2009).

Doctrine forma parte del marco de trabajo Sauxe, la versión actual de Doctrine que será liberada con Sauxe es la 1.2.4.

1.4.2.7 Herramienta ExtJS

Como se expuso anteriormente Sauxe también contiene ExtJS. ExtJS es una librería Java Script ligera y de alto rendimiento, compatible con la mayoría de los navegadores que permite crear páginas e interfaces web dinámicas. (Comunidad de desarrollo, 2006-2010) Sauxe utiliza ExtJS en la capa de presentación, por la gran gama de componentes que se pueden reutilizar para agilizar el proceso de desarrollo y mostrarle al usuario una interfaz más amigable y funcional (Baryolo, 2008).

Esta librería incluye (Corzo, 2008):

- Componentes Interfaz de Usuario (UI) del alto performance y personalizables.
- Modelo de componentes extensibles.
- Un API fácil de usar.
- Licencias de códigos abiertos y comerciales.

Una de las grandes ventajas de utilizar ExtJS es que permite crear aplicaciones complejas utilizando componentes predefinidos así como un manejador de layouts similar al que provee Java Swing, gracias a esto provee una experiencia consistente sobre cualquier navegador, evitando el tedioso problema de validar que el código escrito funcione bien en cada uno (Firefox, Internet Explorer, Safari). Además, la ventana flotante que provee ExtJS es excelente por la forma en la que funciona. Al moverla o redimensionarla solo se dibujan los bordes haciendo que el movimiento sea fluido lo cual le da una ventaja tremenda frente a otros (Corzo, 2008).

Como se ha visto hasta el momento Sauxe es el resultado de una importante mezcla entre: la extensión de algunos componentes de un potente Marco de Trabajo, un sistema ORM y una muy popular librería java script, dando lugar a un novedoso Marco de Trabajo, con funcionalidades y prestaciones específicas pero también fácilmente extensibles a cualquier aplicación web de gestión.

La versión de *ExtJS* que será liberada con Sauxe es la 1.1.1

1.4.3 Problemas tecnológicos identificados en Sauxe

Se han identificado dos situaciones con el marco de trabajo Sauxe que representan unos los principales problemas tecnológicos que afronta, estas se muestran a continuación:

1. No es posible realizar transacciones anidadas que implican ejecuciones de servicios de componentes externos al originario de las mismas. Este problema se refleja al iniciar una transacción desde un componente determinado y se debe hacer uso de servicios externos a este; en este caso se necesita hacer un cambio de conexión, por consiguiente ya la transacción no tiene forma de deshacer los cambios (Rollback) en esa nueva conexión que fue abierta. Por ejemplo: En una transacción generada desde el subsistema de capital humano con el fin de modificar el salario de un trabajador que sale de vacaciones, se necesita modificar en el subsistema de contabilidad y finanzas el régimen de pago del trabajador para que afecte el salario durante este período. Para satisfacer esta necesidad se ejecuta la petición *Asignar vacaciones*. Primeramente se hace una llamada a un servicio externo con una nueva conexión que será la encargada de cambiar el régimen de pago en el esquema *Contabilidad y Finanzas* cerrándose la comunicación con esta conexión al finalizar su petición, luego utilizando los resultado de la acción anterior se intentan modificar un grupo de campos en el esquema *Recursos Humanos*. En caso de que esta última parte de la transacción sea insatisfactoria se deshacen los cambios (Rollback), pero el sistema no tiene forma de volver a abrir la conexión con el servicio externo y deshacer los cambios en el esquema de *Contabilidad y Finanzas*.

2. No existe mecanismo de Sauxe para el manejo de alertas y avisos de los productos que soporta (integraciones entre subsistemas/componentes). Este problema está dado por la imposibilidad de Sauxe de enviar avisos a los usuarios de distintos subsistemas, sobre algún suceso relacionado específicamente con ellos, por ejemplo el departamento de *Contabilidad Material* utiliza una cuenta que se genera como todas las demás en *Economía*, para comprar distintos productos, el departamento de economía no tiene forma de enviar un aviso para notificar que la cuenta está cerrada o está agotada de forma automática.

1.5 Principales marcos de trabajo PHP

Luego de identificados los problemas principales a abordar en la investigación, se hace necesario realizar un estudio de los marcos de trabajo PHP más importantes y más usados en la universidad, haciendo hincapié fundamentalmente en los problemas detectados en el marco de trabajo Sauxe, y revisar como estos marcos de trabajo dan tratamiento a estas situaciones.

1.5.1 Symfony

Symfony es un marco de trabajo que simplifica el desarrollo de una aplicación web mediante la automatización de algunos de los patrones utilizados para resolver las tareas comunes. Está basado en el clásico patrón de diseño web conocido como arquitectura MVC (Modelo-Vista-Controlador). Proporciona varias herramientas y clases encaminadas a reducir el tiempo de desarrollo de una aplicación web compleja. También, automatiza las tareas más comunes, permitiendo al desarrollador dedicarse por completo a los aspectos específicos de cada aplicación (Potencier, 2008).

Symfony está desarrollado completamente con PHP 5. Ha sido probado en numerosos proyectos reales y se utiliza en sitios web de comercio electrónico de primer nivel. Symfony es compatible con la mayoría de gestores de bases de datos como MySQL, PostgreSQL, Oracle y SQL Server de Microsoft. Se puede ejecutar tanto en plataformas **nix* (Unix, Linux) como en plataformas Windows (Potencier, 2008).

Desde su primera versión, publicada en octubre de 2005 bajo la licencia de software libre por su creador Fabien Potencier, muchos desarrolladores de todo el mundo descargaron e instalaron el marco de trabajo, comenzaron a leer la documentación y construyeron sus primeras aplicaciones con Symfony, aumentando poco a poco la popularidad del mismo (Potencier, 2008).

En ese momento, los marcos de trabajo para el desarrollo de aplicaciones web estaban en pleno florecimiento, y era muy necesario disponer de uno realizado con PHP. Symfony proporcionaba una solución irresistible a esa carencia, debido a la calidad de su código fuente y a la gran cantidad de documentación disponible, dos ventajas muy importantes sobre otros marcos de trabajo disponibles. Los colaboradores aparecieron en seguida proponiendo parches y mejoras, detectando los errores de la documentación y realizando otras tareas muy importantes (Potencier, 2008).

El repositorio público de código fuente y el sistema de notificación de errores y mejoras mediante tickets permite varias formas de contribuir al proyecto y todos los voluntarios son bienvenidos (Potencier, 2008).

Estos beneficios que brindan los creadores de los marcos de trabajo atraen a los programadores, que ven en Symfony una poderosa herramienta para la creación de aplicaciones web. A través del foro de Symfony, las listas de correo y el IRC (canal de mensajería instantánea) se ofrecen otras alternativas válidas para el soporte del marco. De manera que los desarrolladores de Symfony alrededor del mundo están conectados en una comunidad que muestra muchas ganas de colaboración entre sus integrantes. Dentro de la misma se encuentran también los creadores del marco lo que nutre en gran medida las discusiones que se realizan en dicha comunidad (Potencier, 2008).

Estructura del proyecto: Aplicaciones, Módulos y Acciones

Symfony considera un proyecto como *"un conjunto de servicios y operaciones disponibles bajo un determinado nombre de dominio y que comparten el mismo modelo de objetos"* (Potencier, 2008).

Dentro de un proyecto, las operaciones se agrupan de forma lógica en aplicaciones. Normalmente, una aplicación se ejecuta de forma independiente respecto de otras aplicaciones del mismo proyecto. Lo habitual es que un proyecto contenga dos aplicaciones: una para la parte pública y otra para la parte de gestión, compartiendo ambas la misma base de datos. También es posible definir proyectos que estén formados por varios sitios web pequeños, cada uno de ellos considerado como una aplicación. En este caso, es importante tener en cuenta que los enlaces entre aplicaciones se deben indicar de forma absoluta (Potencier, 2008).

Cada aplicación está formada por uno o más módulos. Un módulo normalmente representa a una página web o a un grupo de páginas con un propósito relacionado. Por ejemplo, una aplicación podría tener módulos como home, artículos, ayuda, carritoCompra, cuenta (Potencier, 2008).

Los módulos almacenan las acciones, que representan cada una de las operaciones que se puede realizar en un módulo. Por ejemplo el módulo carritoCompra puede definir acciones como añadir, mostrar y actualizar. Normalmente las acciones se describen mediante verbos (Potencier, 2008).

Trabajar con acciones es muy similar a trabajar con las páginas de una aplicación web tradicional, aunque en este caso dos acciones diferentes pueden acabar mostrando la misma página (como por ejemplo la acción de añadir un comentario a una entrada de un blog, que acaba volviendo a mostrar la página de la entrada con el nuevo comentario) (Potencier, 2008).

La figura 2 muestra un ejemplo de organización del código para un proyecto de un blog, siguiendo la estructura de proyecto / aplicación / módulo / acción. No obstante, la estructura de directorios real del proyecto es diferente al esquema mostrado por esa figura (Potencier, 2008).

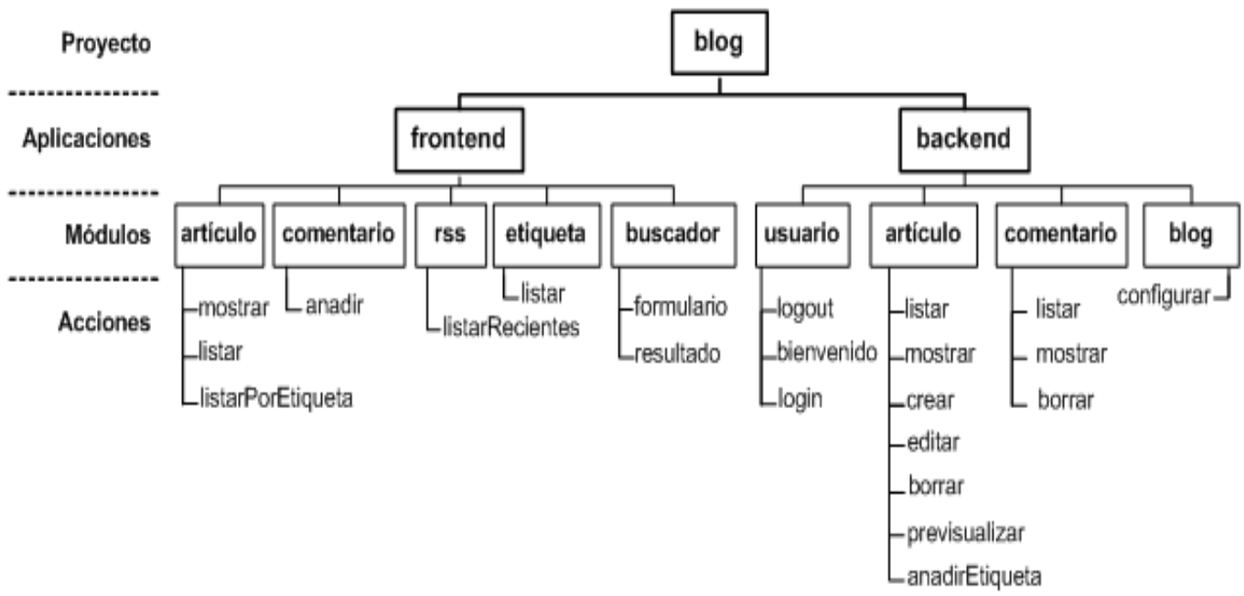

Figura 2 Ejemplo de organización del código.

Estructura del árbol de archivos

Normalmente, todos los proyectos web comparten el mismo tipo de contenidos, como por ejemplo (Potencier, 2008):

- Una base de datos, como MySQL o PostgreSQL.
- Archivo estáticos (HTML, imágenes, archivos de JavaScript, hojas de estilos.).
- Archivos subidos al sitio web por parte de los usuarios o los administradores.
- Clases y librerías PHP.
- Librerías externas (scripts desarrollados por terceros).
- Archivos que se ejecutan por lotes (*batch files*) que normalmente son scripts que se ejecutan vía línea de comandos o mediante cron.
- Archivos de log (las trazas que generan las aplicaciones y/o el servidor).
- Archivos de configuración.

Symfony proporciona una estructura en forma de árbol de archivos para organizar de forma lógica todos esos contenidos, además de ser consistente con la arquitectura MVC utilizada y con la agrupación proyecto / aplicación / módulo. Cada vez que se crea un nuevo proyecto, aplicación o módulo, se genera de forma automática la parte correspondiente de esa estructura. Además, la estructura se puede personalizar completamente, para reorganizar los archivos y directorios o para cumplir con las exigencias de organización de un cliente (Potencier, 2008).

Estructura de la raíz del proyecto

La raíz de cualquier proyecto Symfony contiene los siguientes directorios (Potencier, 2008):

apps/
    frontend/
    backend/
cache/
config/
data/
    sql/
doc/
lib/
    model/
log/
plugins/
test/
    bootstrap/
    unit/
web/
    css/
    images/
    js/
    uploads/

La tabla 1 describe los contenidos de los directorios mostrados anteriormente:

Directorio	Descripción
apps/	Contiene un directorio por cada aplicación del proyecto (normalmente, frontend y backend para la parte pública y la parte de gestión respectivamente).
cache/	Contiene la versión cacheada de la configuración y (si está activada) la versión cacheada de las acciones y plantillas del proyecto. El mecanismo de cache (que se explica en el Capítulo 12) utiliza los archivos de este directorio para acelerar la respuesta a las peticiones web. Cada aplicación contiene un subdirectorio que guarda todos los archivos PHP y HTML preprocesados.
config/	Almacena la configuración general del proyecto.
data/	En este directorio se almacenan los archivos relacionados con los datos, como por ejemplo el esquema dete una base de datos, el archivo que contiene las instrucciones SQL para

	crear las tablas e incluso un archivo de bases de datos de SQLite.
doc/	Contiene la documentación del proyecto, formada por tus propios documentos y por la documentación generada por PHP doc.
lib/	Almacena las clases y librerías externas. Se suele guardar todo el código común a todas las aplicaciones del proyecto. El subdirectorio model/ guarda el modelo de objetos del proyecto.
log/	Guarda todos los archivos de log generados por Symfony. También se puede utilizar para guardar los logs del servidor web, de la base de datos o de cualquier otro componente del proyecto. Symfony crea un archivo de log por cada aplicación y por cada entorno.
plugins/	Almacena los plugins instalados en la aplicación.
test/	Contiene las pruebas unitarias y funcionales escritas en PHP y compatibles con el marco de trabajo de pruebas de Symfony. Cuando se crea un proyecto, Symfony crea algunas pruebas básicas.
web/	La raíz del servidor web. Los únicos archivos accesibles desde Internet son los que se encuentran en este directorio.

Tabla 1 Directorios en la raíz de los proyectos Symfony.

Hoy día internet cuenta con cientos sitios creados con Symfony. Desde grandes redes sociales como: *Delicious*, *Daily motion* o *Yahoo! Answers* hasta medianos y pequeños sitio. De forma que millones de usuarios utilizan aplicaciones construidos con Symfony (Potencier, 2008).

Luego de analizadas las características del marco de trabajo Symfony, se puede apreciar que no cuenta con una clase para la gestión de transacciones como el Zend Framework (TransactionManager), para la gestión de las transacciones usa fundamentalmente la clase *doctrine manager* incluida en el marco de persistencia Doctrine. Sus componentes están bien organizados en la estructura de carpetas haciendo posible instanciar en un módulo (forma de llamarle en Symfony a los componentes) cualquier clase de otro de los módulos presentes en el sistema. Se puede concluir además que tampoco cuenta con una clase o pluggin embebida para la gestión de alertas y avisos.

1.5.2 CodeIgniter

CodeIgniter es relativamente un nuevo marco de trabajo, por los fabricantes de ExpressionEngine, y parece muy prometedor. Está inspirado en Ruby on Rails, y que ofrece una gran parte de la misma funcionalidad, como los Scaffolding. Tiene una excelente documentación, y se han incluido vídeo tutoriales. CodeIgniter es un poderoso marco de trabajo PHP con una muy pequeña huella, construido para los codificadores de PHP que necesita un simple y elegante conjunto de herramientas para crear todo lo que ofrece aplicaciones web (CodeIgniter, 2012).

CodeIgniter viene con una clase de base de datos llena de "features" y una muy rápido abstracta clase de base de datos que soporta tanto estructuras tradicionales y el patrón Active Record. Las funciones de base de datos ofrecen clara, simple sintaxis (EllisLab inc., 2012).

La Clase Active Record es globalmente activada o desactivada estableciendo la variable $active_record en el archivo de configuración para la base de datos, en TRUE/FALSE. Si no vamos a usar la Clase Active Record, establecerla en FALSE hará que se utilicen menos recursos cuando se inicialicen las clases de base de datos (EllisLab inc., 2012).

Toda la gestión de transacciones del marco de trabajo se realiza a través de esta clase ActiveRecord, es decir el marco de trabajo CodeIgniter no cuenta con una clase, la TransaccionManager, del Zend ExtJS creado en CEIGE para la gestión de las transacciones. No presenta ninguna clase para la gestión de avisos y alertas.

1.6 Técnicas de notificación

Las técnicas de notificación utilizadas en aplicaciones web permiten a los usuarios recibir notificaciones tan pronto como la información es publicada, sin necesidad de chequear la fuente original manualmente para obtener actualizaciones (Werdmuller, 2010).

Se hace necesario hacer un estudio de estas técnicas para identificar cuál de ellas utilizar en la propuesta de solución al problema de los avisos y alertas.

1.6.1 Long-Polling

Esta técnica está basada en el uso de Asynchronous JavaScript And XML (AJAX), a través de peticiones consecutivas partiendo del cliente y hacia el servidor. Una vez se recibe respuesta, comienza de nuevo otro proceso petición cliente-servidor. Funciona en todos los navegadores y como la mayoría de aplicaciones AJAX: Se lanza una petición al servidor, es mantenida una conexión y no se cierra hasta que se recibe la respuesta. Una vez recibida, automáticamente se lanza otra nueva petición (Oviedo Chaparro, 2010).

Algunas desventajas de dicha técnica son (Oviedo Chaparro, 2010):

- Debido al paradigma de la web, el cual se refiere al término ‘’síncrono’’ lo cual significa que el servidor envía datos solo en el caso que el cliente haga una petición.
- Si el escenario se encuentra en condiciones donde el volumen de datos sea relativamente grande puede suceder que los mismos sean divididos en dos o varias peticiones lo cual, dependiendo del ancho de banda ocupado y las características físicas de las vías de conexión puede hacer tardar el proceso y de esta manera rompería con el resultado deseado de transmitir datos en tiempo real.
- Esta técnica ocupa un ancho de banda alto, para cada pedido realizado por parte desde el navegador.

1.6.2 Websockets

Según W3C (World Wide Web Consortium) Es una nueva tecnología que, "permite a las aplicaciones web mantener una comunicación bidireccional con procesos en el lado del servidor". La misma sucede a través del capa 3 del modelo de referencia de Interconexión de Sistemas Abiertos (OSI, Open System Interconnection), y se realiza entre cliente – servidor utilizando un mecanismo para ponerlos en contacto: API (Application Programming Interface).Tanto el servidor y el cliente o clientes utilizan una aplicación, la cual es responsable de interactuar con los sockets, dándole responsabilidad el sistema operativo de utilizar un proceso para: crear, utilizar y luego cerrar el socket cuando ya se ha transmitido el mensaje. Este flujo de eventos constituye un uso de recursos innecesario de ambas partes, además de representar una relativa demora, la cual se puede incrementar considerablemente si se usa WebSocket Secure connections (wss), porque el mismo utiliza Transport Layer Security(TLS) y representa un costo adicional en cuanto al tiempo que puede tardar para encriptar (Barros, 2011).

Aunque el protocolo Websockets es indiferente a la conexión sobre servidores proxy o cortafuegos, implementa una negociación compatible con HTTP para que los servidores HTTP puedan compartir sus puertos HTTP y HTTPS por defecto (80 y 443) con una pasarela o servidor WebSocket y los mensajes enviados corren el riesgo de ser bloqueados, provocando que la conexión falle (Barros, 2011).

WebSocket no es solamente una nueva forma de comunicación entre el cliente y el servidor, sino un cambio de paradigma para el desarrollo de aplicaciones web estacionarias y móviles. Lograr comunicación bidireccional y tiempo real revoluciona muchos de los preceptos del desarrollo de software en la Web y muestra un grupo importante de nuevas potencialidades.

1.6.3 COMET

El término COMET describe a las aplicaciones donde el servidor se mantiene enviando los datos, o manteniendo un cúmulo continuo de datos a la aplicación cliente, en vez de tener al navegador realizando peticiones al servidor para actualizar el contenido, es decir COMET cambia fundamentalmente la naturaleza de la comunicación realizada entre la arquitectura conocida Cliente-Servidor (Oviedo Chaparro, 2010).

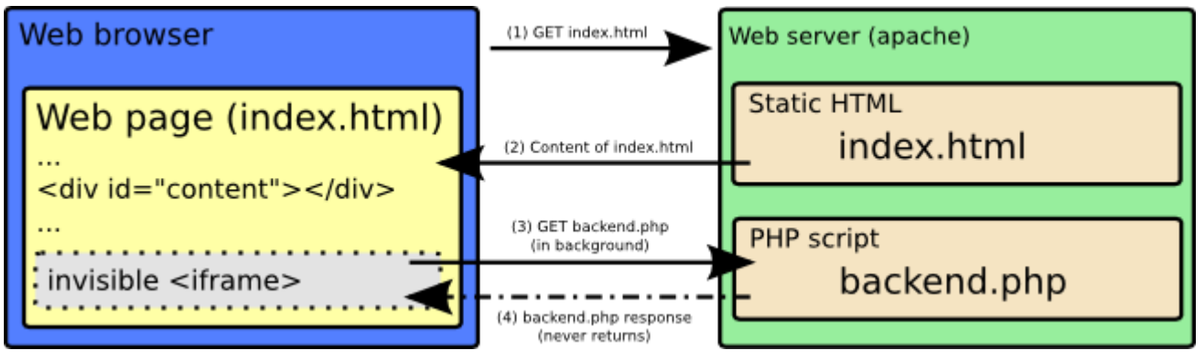

Figura 3 Arquitectura cliente servidor.

COMET se basa en que el servidor te envía datos aunque el cliente no los pida (HTTP Push), si haces una llamada, el canal se queda abierto y el servidor va mandando información, o lo que es lo mismo, que el servidor va a estar devolviendo el resultado en partes. En otras palabras todas las aplicaciones de COMET utilizan conexiones HTTP de larga duración para reducir a latencia con la cual los mensajes son pasados al servidor. En esencia no realizan pedidos ocasionalmente al servidor. En vez de eso el servidor mantiene una línea abierta de comunicación mediante la cual puede "empujar" datos hacia el cliente. COMET utiliza XMLHttpRequest para la entrega de datos entre el cliente y el servidor a través del protocolo HTTP. También es conocido como Server Push o HTTP Push (Oviedo Chaparro, 2010).

Debe quedar en claro que COMET no es un marco de trabajo o un conjunto de herramientas. Sino más bien es un concepto como lo es AJAX.

Este método podría significar un desperdicio en términos de sockets y threads, y también representaría una sobrecarga para los firewalls, los balanceadores de carga. En algunos foros de discusión se crítica esta técnica ya que al mantener abierta la conexión por un largo periodo, surgen problemas de escalabilidad como consecuencia de lo expuesto anteriormente (Oviedo Chaparro, 2010).

1.6.4 Jabber

Jabber es un sistema basado en XML Lenguaje de Marcado Extensible (Extensible Markup Language) para el intercambio en cuasi tiempo real de mensajes y presencia (Jabber.org, 2012).

Cualquier programa cliente-servidor establece una conexión con el servidor remoto y éste coordina los datos que se transfieren entre los distintos clientes. Dos contactos con cuentas del mismo servidor, pueden hablar entre ellos de forma directa, como no existe un solo servidor de Jabber, sino muchos independientes interconectados entre sí, no hay problema en que dos usuarios de servidores diferentes hablen entre sí, los servidores de estos usuarios se intercambian los mensajes, como los servidores son formados por asociaciones particulares o usuarios por gustos, cada uno contienen distintas características.

Ventajas (Jabber.org, 2012):

- Protocolo abierto: se puede implementar un servidor o un cliente o ver el código, entre otras cosas.

- Extensible: usando el potencial del lenguaje XML, cualquiera puede extender el protocolo de Jabber para una funcionalidad personalizada.

- Estándar: ha sido aprobado bajo el nombre de XMPP (Protocolo extensible de mensajería y comunicación de presencia. Sus especificaciones han sido publicadas como RFC 3920 y RFC 3921.

- Seguro: cualquier servidor Jabber está aislado del exterior. El servidor de referencia permite SSL [5] para comunicaciones cliente-servidor y algunos clientes aceptan GPG[6] como cifrado de las comunicaciones usando cifrado asimétrico. En desarrollo uso de claves de sesión y SASL[7].

- Descentralizado: cualquiera puede montar su propio servidor de Jabber, además está libre de patentes y no depende de ninguna empresa de modo que se puede usar ahora y siempre con total libertad.

- Multiredes: permite comunicarse con otros protocolos usados por clientes como MSN Messenger, ICQ, AOL o Yahoo!.

Desventajas (Jabber.org, 2012):

- El principal problema, es la estabilidad de los servidores públicos, se abusa de la capacidad de los multiprotocolos mediante pasarelas o transportes, ya que esto provoca la saturación de los servidores.
- Los servidores públicos para brindar un mejor servicio proporciona poca posibilidad de conexión con otras redes como IRC, MSN Messenger.
- La privacidad de cara a alguien que tenga acceso al servidor, ya que en él se almacenarán todos los mensajes que se reciban para las cuentas Jabber ID de los usuarios que no estén conectados.
- Si en una red empresarial, por citar un ejemplo, sólo existe un servidor de Jabber y éste permanece inactivo durante un tiempo, los clientes no podrán comunicarse entre ellos. Esto tiene su solución que es instalar un grupo de servidores de Jabber.

1.6.5 Protocolo Jabber/XMPP

Es el protocolo usado por Jabber, también el usado por otras aplicaciones como Google Talk y LiveJournal Talk por lo que son interoperables. El protocolo está estandarizado por el IETF Internet Engineering Task Force si bien las extensiones (XEP, XMPP Estandard Propásale, antes conocidas como JEP) extensiones a este estándar son definidas por la XMPP Software Foundation. Está basado en un conjunto tecnologías con XML's para aplicaciones en tiempo real. Fue desarrollado originalmente como un marco de trabajo de aplicaciones de mensajería instantánea y presencia para escenarios de empresas. El mismo utiliza ficheros en formato XML para hacer las transferencias de mensajes entre sus entidades: cliente-servidor o servidor-servidor (Werdmuller, 2010).

XMPP es un conjunto de tecnologías basado en XML para aplicaciones en tiempo real[2]. Originalmente fue desarrollado como un marco de apoyo a la mensajería instantánea y aplicaciones de presencia en entornos empresariales. En ese momento, las actuales redes de mensajería instantánea eran propietarias y en gran medida inadecuadas para el uso empresarial. *AOL Instant Messenger*, por ejemplo, no podría ser adaptado para comunicaciones seguras dentro de una empresa. Aunque las soluciones comerciales han existido, sus conjuntos de características fijas por lo general no podían ser adaptadas para satisfacer las necesidades de una organización (Werdmuller, 2010).

XMPP, entonces llamado Jabber, permite a las organizaciones construir sus propias herramientas personalizadas para facilitar la comunicación en tiempo real, así como instalar soluciones pre elaboradas de terceros (Werdmuller, 2010).

XMPP es una red de comunicación descentralizada, lo que significa que cualquier usuario XMPP puede enviar un mensaje a cualquier otro usuario XMPP, si la infraestructura de red lo permite. Servidores XMPP también pueden hablar uno con otro con un protocolo especializado de servidor a servidor, lo que abre posibilidades interesantes para las redes descentralizadas sociales y los marcos de colaboración, a pesar de que está fuera del alcance de este tutorial. Como su nombre indica, XMPP puede ser utilizado para satisfacer una amplia variedad de requisitos de características sensibles al tiempo. De hecho, Google Wave, un entorno multiusuario de gran colaboración, usa XMPP como base para su protocolo de comunicación. A pesar que XMPP surgió de un requisito para la mensajería instantánea de persona a persona, de ninguna manera se limita solo a esa tarea (Werdmuller, 2010).

Para facilitar la entrega de mensajes, cada usuario del cliente XMPP debe tener un identificador universal único. Por razones de herencia, éstos se conocen como Jabber ID o JIDs. Debido a la naturaleza distribuida del protocolo, es importante que la JID contenga toda la información necesaria para contactar a un usuario. La estructura de la JID es similar a una dirección de correo electrónico (aunque no hay ningún requisito para una JID a doblar como un destinatario de correo electrónico válida). Ambos nodos cliente y servidor, lo que me refiero colectivamente como entidades XMPP, tienen JIDs. Juan Pérez, un empleado de SomeCorp, podría tener la JID John.Doe @ somecorp.com. Aquí, somecorp.com es la dirección del servidor XMPP para SomeCorp y John.Doe es el nombre de usuario para John Doe (Werdmuller, 2010).

1.6.5.1 Categorías de comunicación

Un sistema de mensajería en tiempo real mediante XMPP se compone de tres grandes categorías de la comunicación (Werdmuller, 2010):

• Mensajería, en la que los datos se transfieren entre las partes.

- [2] Un sistema de tiempo real es aquel en el que la corrección de los resultados no depende sólo de la corrección de los cálculos realizados para producirlos, sino también del instante en el que éstos están disponibles.

• Presencia, que permite a los usuarios transmitir su estado en línea y disponibilidad.

• Info / consulta peticiones, que permiten a una entidad XMPP para hacer una solicitud y recibir una respuesta de otra entidad.

Estos son complementarios. Por ejemplo, a menudo no hay punto en enviar datos a un usuario, o hacer una solicitud de información / consulta de una entidad, si el usuario o entidad no está en línea (aunque en muchos casos de uso es deseable que el servidor mantenga los mensajes para los usuarios hasta que regresan). Cada uno de ellos se entrega a través de una estrofa XML, un elemento discreto de la información expresada en XML.

Las estrofas XMPP pueden ser de tres tipos y tienen los siguientes atributos en común (Werdmuller, 2010):

• *De*: La JID de la entidad de origen XMPP

• *A*: La JID del destinatario

• *Identificación:* Un identificador opcional para la conversación

• *Tipo*: Opcional subtipo de la estrofa

• *XML: lang:* Si el contenido es legible, la descripción del lenguaje del mensaje.

La transferencia de datos sobre XMPP tiene lugar en flujos XML, que opera en el puerto 5222 por defecto. Estos son efectivamente dos documentos XML completo, cada uno correspondiente a una dirección de comunicación. Una vez que se establece una sesión, el elemento de corriente es abierto. Se envuelve el documento de comunicación completa. Las estrofas son entonces inyecta en el segundo nivel del documento. Por último, una vez que termina la comunicación, el elemento de secuencia se cierra, formando un documento completo (Werdmuller, 2010).

1.6.6 Módulo de gestión universitaria

El módulo de Notificaciones y Alertas implementa funciones que se comportan como manejadores de notificaciones y manejadores de alertas los cuales se encargan de organizar las acciones a realizarse después de la ocurrencia de un evento o el análisis de una variable; en ambos casos se arma el mensaje o notificación a enviarse, se buscan las relaciones con los usuario y ejecuta el envió de acuerdo al dispositivo registrado. Los dispositivos que hoy se manejan como destino de la información son: dirección de correo electrónico y buzón de notificaciones y alertas del SGU (Santiesteban, y otros, 2010).

En base de dato se almacena información de: usuarios, configuraciones de notificaciones y alertas, además de cada uno de los avisos generados por el sistema. La propuesta permite: ver la cantidad de notificaciones y alertas del usuario autenticado en el SGU, mostrar el listado de notificaciones leídas y no leías del usuario autenticado, así como el listado de sus alertas y mostrar el mensaje completo de una notificación o alerta (Santiesteban, y otros, 2010).

Se prevé que la solución permita en nuevas versiones la configuración de notificaciones y alertas en modo visual. Es importante señalar en esta descripción, el impacto del uso de platillas en los mensajes. Para

lograr esto, los mensajes son parametrizados, donde los parámetros serían las partes del mensaje que varían, ejemplo: “El estudiante {1} pasó al estado {2} a través del trámite {3}.¨ ¨El trámite {3} fue realizado al estudiante {1} pasando al estado {2}.”

1- Nombre(s) y apellidos del estudiante.

2- Estado del estudiante.

3- Nombre del trámite.

El uso de plantillas permite la personalización e internacionalización de mensajes. Se hace necesario puntualizar que a la hora de reconstruir el mensaje los parámetros deben poseer el mismo significado. El diseño de la solución describe cómo los requisitos funcionales y no funcionales, junto con otras restricciones relacionadas con el entorno de implementación, tienen impacto en el sistema. La solución es una aplicación web por tanto se sigue un patrón de arquitectura Cliente-Servidor, el uso de CodeIgniter como marco de trabajo implica una restricción de diseño; CodeIgniter implementa el patrón de arquitectura Modelo-Vista-Controlador (Santiesteban, y otros, 2010).

El sistema en cuanto a diseño queda expuesto en 4 paquetes: El paquete Vistas_Notificaciones_ Aletas contiene las interfaces de usuario necesarias para satisfacer cada uno de los escenarios de uso del sistema. El paquete Controladoras_Notificaciones_Alertas contiene clases responsables de controlar el flujo de datos ente vistas, librerías y modelos. El paquete Librerías_Notificaciones_Alertas contiene clases que encapsulan métodos que permiten implementar lógica de negocio. El paquete Modelos_Notificaciones_Alertas son clases que permiten realizar llamadas y consultas a la base de datos (Santiesteban, y otros, 2010).

Por su aplicación en el Sistema de Gestión Universitaria es posible considerar su uso para la presente investigación.

1.7 Conclusiones parciales

En el capítulo se presenta el marco de trabajo Sauxe con el cual está desarrollado el sistema Cedrux, sus características y componentes más significativos, se identificaron problemas trascendentales presentes en el marco de trabajo que influyen de forma directa en el desarrollo de Cedrux y la necesidad de resolver dificultades con algunas de las herramientas tecnológicas del Marco de Trabajo. Se vieron los elementos primordiales que forman parte de Sauxe para mejor comprensión del mismo durante la investigación así como los conceptos fundamentales con los que este trabaja.

Se presenta un estudio del arte de algunos de los más importantes marcos de trabajo PHP usados en la universidad, donde se apreciaron algunas de sus características importantes para la investigación. Se llega a la conclusión de que el marco de trabajo Symfony presenta una estructura de carpetas muy interesante y que no presenta inconveniente para la gestión de transacciones, que podría ser implementado en el marco de trabajo Sauxe. Sin embargo ninguno de los marcos vistos, Symfony o CodeIgniter, presentan alguna clase para la gestión de avisos y alertas que pueda ser implementada en Sauxe.

Capítulo 1: Fundamentación Teórica

Se realiza un análisis de las distintas técnicas de comunicación y sistemas de notificación, donde se pueden apreciar las ventajas y desventajas de estas herramientas. Poder apreciar estas características permitirá diferenciar a la hora de plantear las soluciones, cual podría ser la más aceptable y con mayores posibilidades de ser implementada para dar solución al problema de las notificaciones y alertas.

Capítulo 2

2. Propuestas de soluciones

2.1 Introducción

Teniendo en cuenta los problemas identificados en el capítulo anterior y el estudio de las diversas herramientas y tecnologías que podrían utilizarse para las soluciones a los problemas de la gestión de transacciones y la gestión de avisos y alertas del sistema Sauxe, se podrá en este capítulo realizar propuestas de soluciones en cumplimiento del objetivo de la investigación.

Es preciso señalar que cualquier modificación, haciendo uso de las propuestas que se presentaran, en el marco de trabajo Sauxe, afectaría directamente la aplicación Cedrux, sistema de interés para el centro CEIGE y el país; por lo que se hará mención en ocasiones del sistema Cedrux durante la presentación de las soluciones.

2.2 Propuestas de solución para el problema de las transacciones

A continuación se presentan las propuestas para el problema de la gestión de transacciones.

2.2.1 Vista de sistema de la arquitectura

Para esta solución se propone la reestructuración de los subsistemas y sus componentes de modo similar a como se encuentra distribuido en los marcos de trabajo Symfony. De modo que el sistema Cedrux aun continúe dividido en subsistemas pero no en disímiles componentes internos.

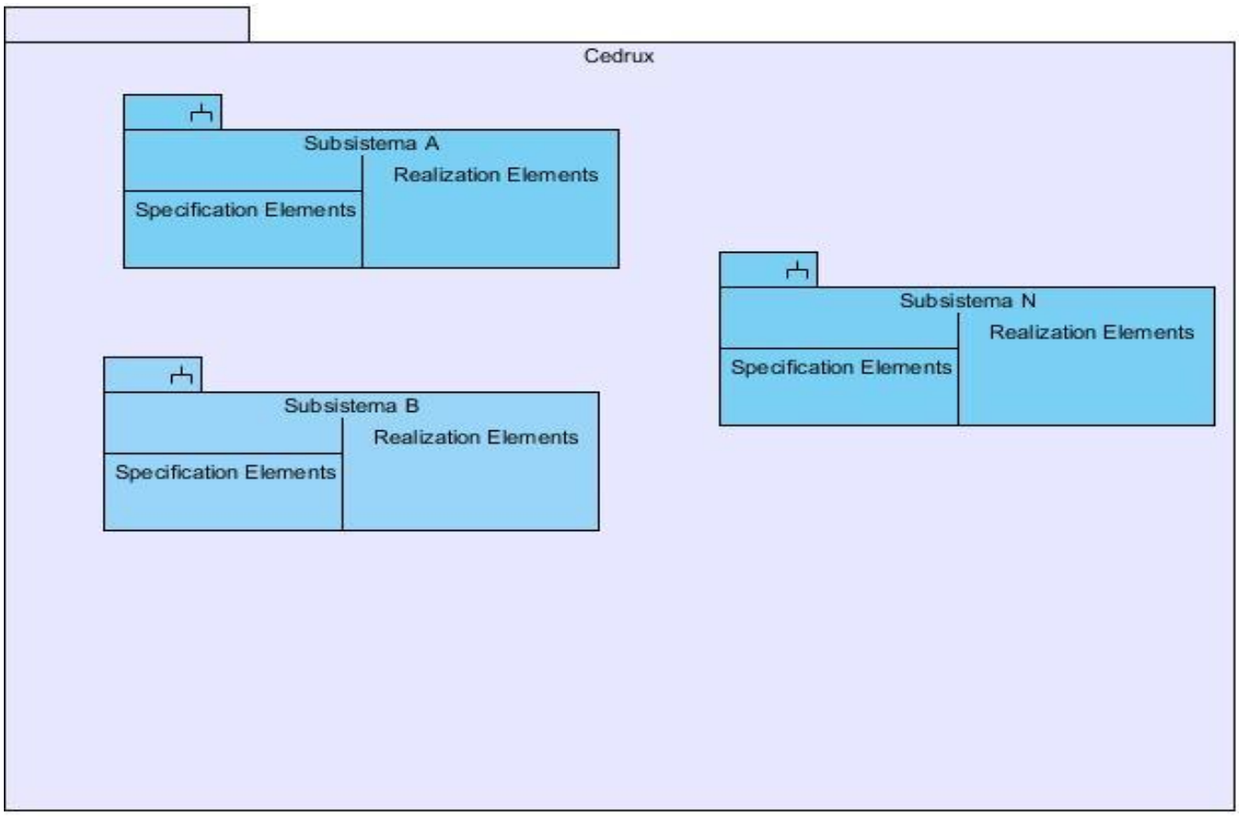

Figura 4 Diagrama de componentes actual del sistema.

En el epígrafe 2.2.3 se explica cómo los subsistemas están divididos a su vez en componentes independientes que pueden ser componentes distintos pero no necesariamente deben estar separados en componentes individuales, cuando realmente todos forman parte de un mismo subsistema.

Los componentes dentro de los subsistemas se harán efectivos solo a nivel de las vistas en la aplicación, pues la estructura de clases del marco de trabajo se guardaría en la forma de la estructura de la propuesta de solución (figura 13).

Sería necesario también, cambiar las tablas en los distintos esquemas en la base de datos para eliminar las llaves foráneas innecesarias y así eliminar la razón fundamental del problema de las transacciones en el sistema Cedrux.

Estas llaves foráneas solo serían eliminadas con otros subsistemas, dentro de un mismo subsistema no se hace necesario su eliminación (figura 6). Solamente sería eliminada la llave, no el campo; por ejemplo si la tabla trabajador del subsistema Capital Humano tiene una llave foránea de cuenta del subsistema nómina, llamada id_cuenta, este campo en la tabla trabajador solamente dejaría de ser llave foránea pero seguiría siendo un campo de la tabla.

De esta forma quedaría eliminada la dependencia de las tablas entre subsistemas, pero a su vez para garantizar la consistencia de datos se tendrían que hacer validaciones a nivel de negocio así como la verificación de los datos dependientes, o sea si en el subsistema nómina se decide eliminar el id_cuenta 13, que es una tupla de su tabla junto a otros records, el negocio tendría que encargarse de avisar a todas las tablas que usen id_cuenta de que ese id ha sido eliminado, o en caso de que esté en uso imposibilitar

la acción de eliminar la tupla en cuestión. La estructura actual de la base de datos (fragmento de diagrama de entidad relación) se muestra en la figura 5, donde se podrá apreciar la existencia de las llaves foráneas entre subsistemas y entre componentes internos.

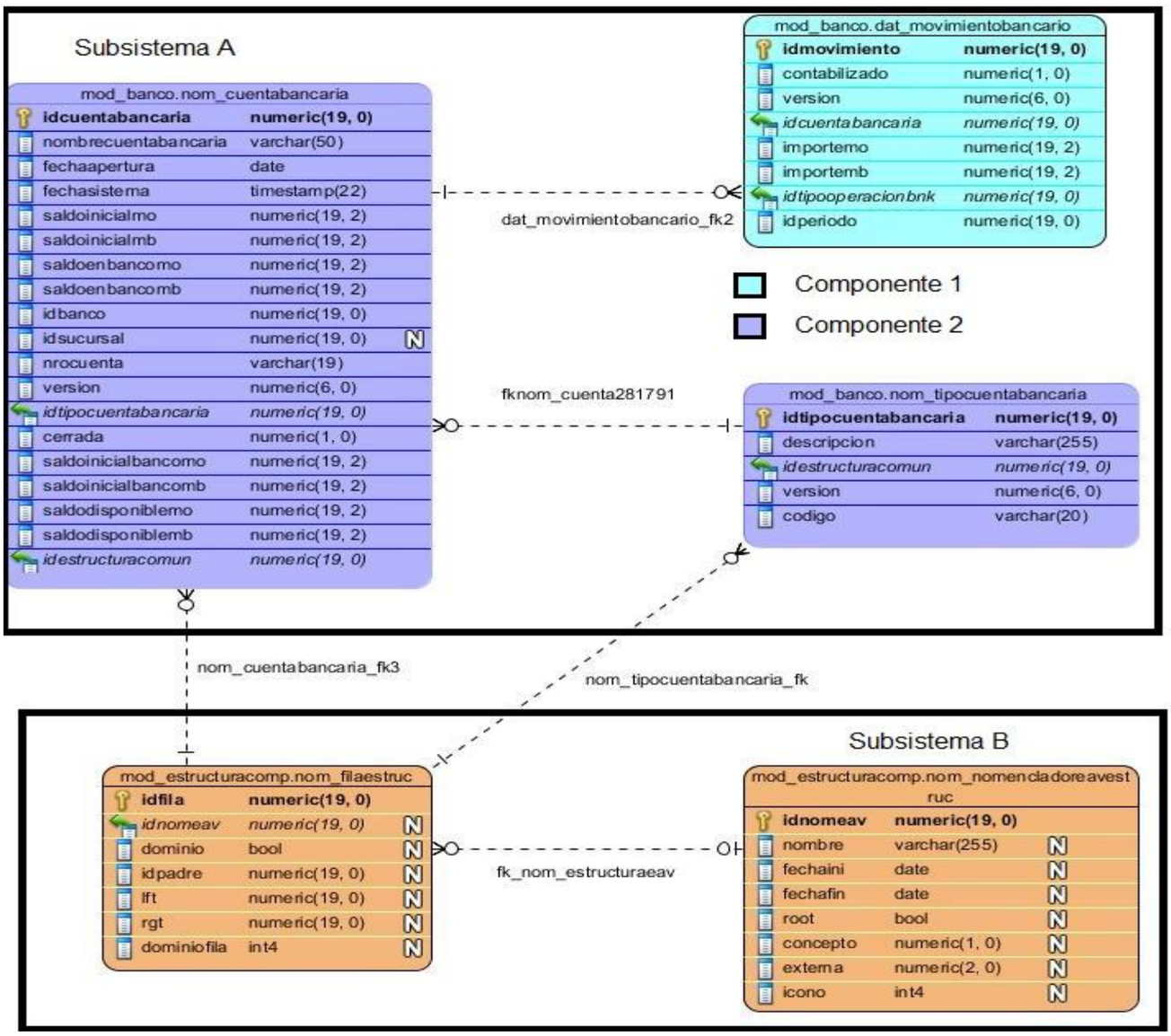

Figura 5 Diagrama relacional de la base de datos actual de Cedrux.

En la figura 6 se aprecia la modificación propuesta para la base de datos del sistema Cedrux, en la cual como se explicó anteriormente se eliminarían las llaves foráneas de otros subsistemas, dichas llaves quedarían como un campo entero (int) dentro de la tabla, tipo de datos igual al que tenía como llave foránea.

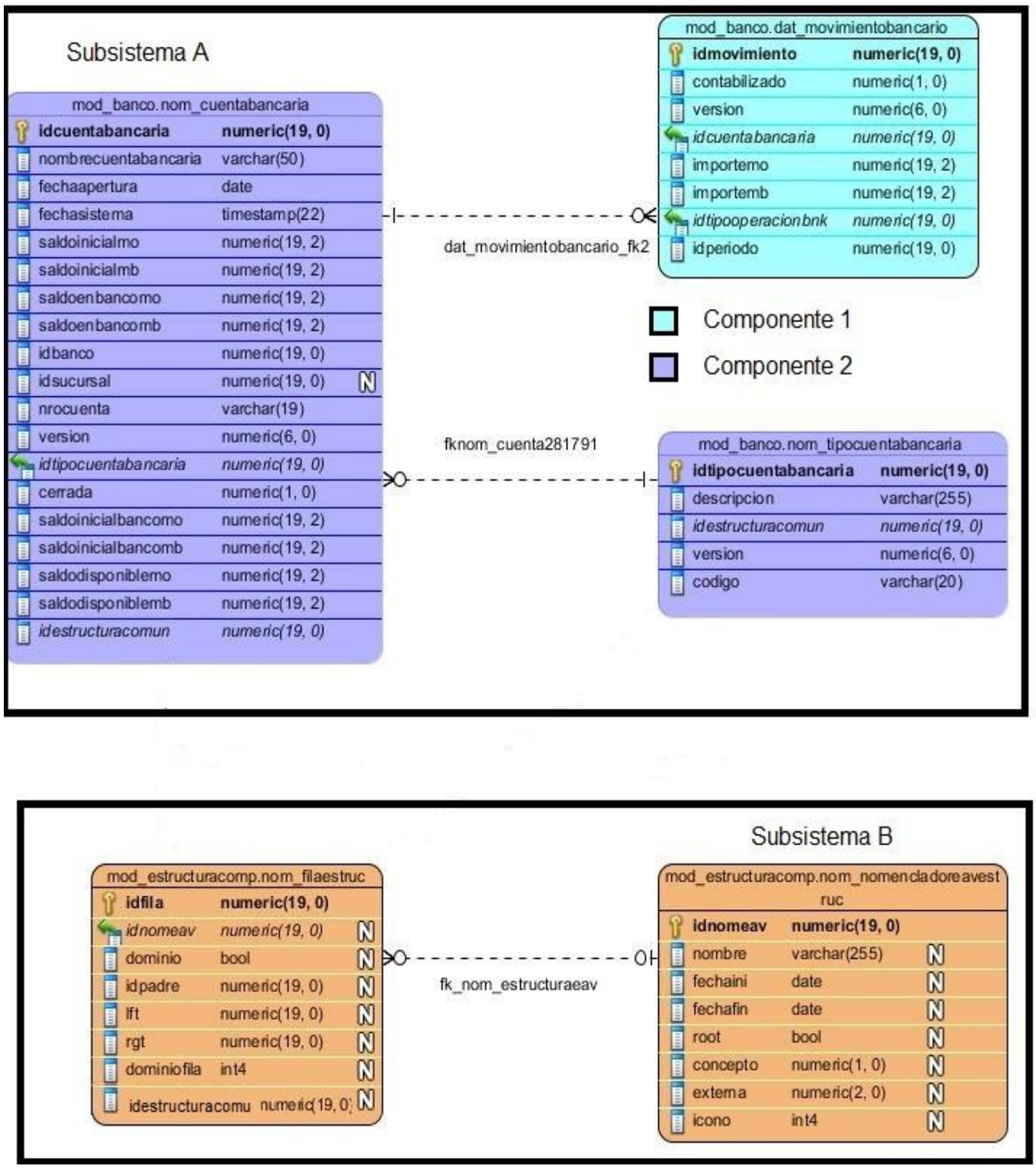

Figura 6 Diagrama relacional de la propuesta de base de datos modificada para Cedrux.

Para la reestructuración de un subsistema sería necesaria la refactorización del código existente, cambiando las dependencias que existen entre los distintos subsistemas y componentes, sería necesario entonces perfeccionar y modificar la relación a través de las clases del negocio de estas tablas para validar cualquier acción que se ejecute sobre ellas y así evitar la inconsistencia de los datos.

Sería excelente revisar el código del sistema para garantizar el respeto a la arquitectura en todos los componentes de Cedrux, porque por ejemplo puede darse el caso que en algún subsistema la gestión de

las transacciones se realice a través del marco de persistencia y su clase Doctrine_Manager y no por la clase establecida en el marco de trabajo TransactionManager.

Una de las principales dificultades que se plantean, es en cuanto a la comunicación entre los equipos de desarrollo de los distintos subsistemas, sin la necesaria comunicación y trabajo en equipo de parte de los responsables de la programación y la arquitectura de los distintos subsistemas y componentes, se haría realmente difícil garantizar la implantación de las soluciones.

2.2.2 Configuración de los usuarios para acceso a la base de datos

Esta solución está basada en el acceso a la base de datos por parte de los usuarios del sistema.

2.2.2.1 Única conexión

Se trata de crear una única conexión para que todos los usuarios del sistema realicen su trabajo en la base de datos a través de ella:

- Establecer una sola conexión para todo el sistema para que las transacciones sean con una sola conexión, de manera tal que cuando exista un error se dé rollback a la transacción y no sufran cambios los esquemas de la base de datos afectados o sea se retorna a un punto en que los datos están en un estado correcto.

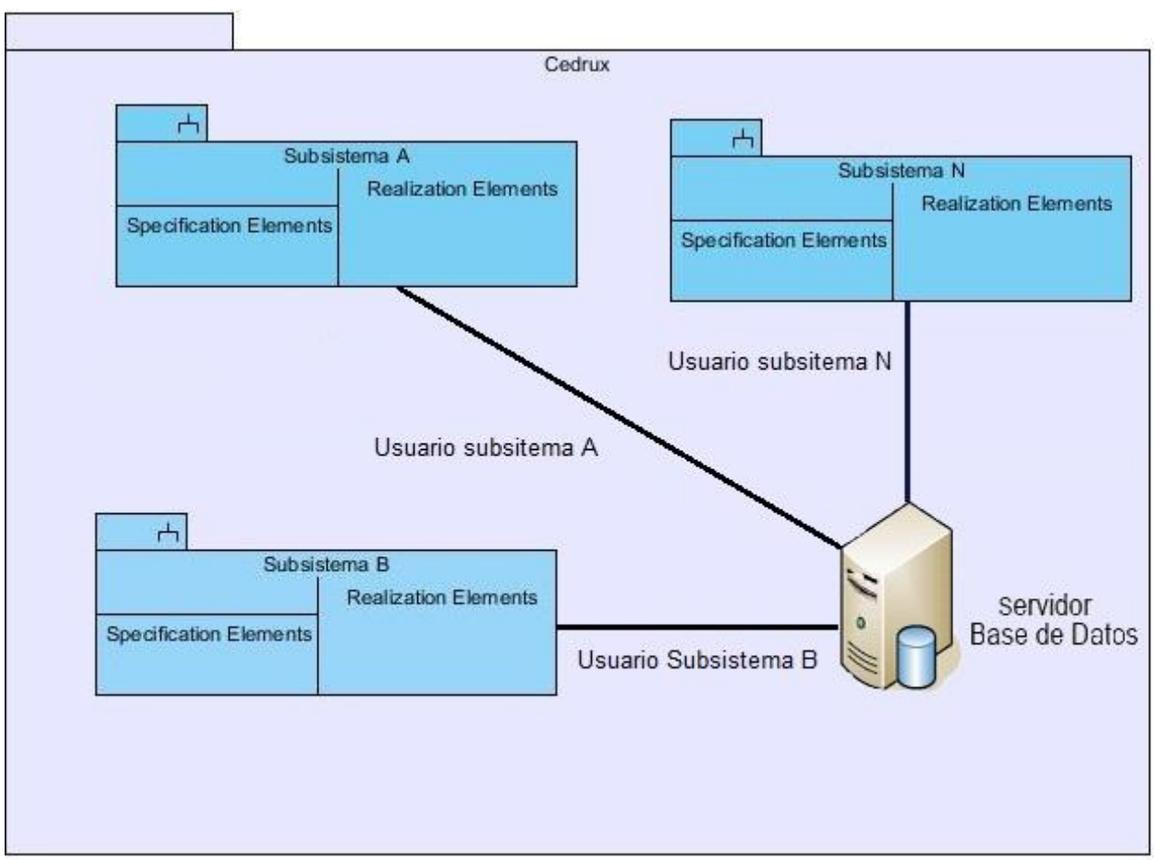

Figura 7 Conexión de los subsistemas a la base de datos por distintos usuarios.

En la figura 7 se aprecia cómo se realiza actualmente la conexión desde los subsistemas a la base de datos, cada subsistema tiene un usuario para realizar esta tarea.

El marco de trabajo Sauxe, arrastra consigo el problema de que existen para el Cedrux tantas conexiones diferentes para la base de datos como el número de módulos existentes, los marcos de trabajo estudiados, Symfony y CodeIgniter, solo utilizan una conexión.

En este caso existiría una sola conexión para todo el sistema a nivel de datos. Esta propuesta podría traer problemas en las trazas del sistema, ya que al conectarse todos los usuarios a través de una misma conexión, no se podría saber cuándo o quién está modificando los datos existentes, pero todas las transacciones podrían hacerse a través de una única conexión y se verían reducidos los errores en las transacciones al contarse con el mismo usuario para poder realizar la operación de rollback. Esta solución resuelve el problema de las transacciones para los componentes de un mismo subsistema, pero al estar presentes llaves foráneas de otros subsistemas, persiste el problema de las transacciones.

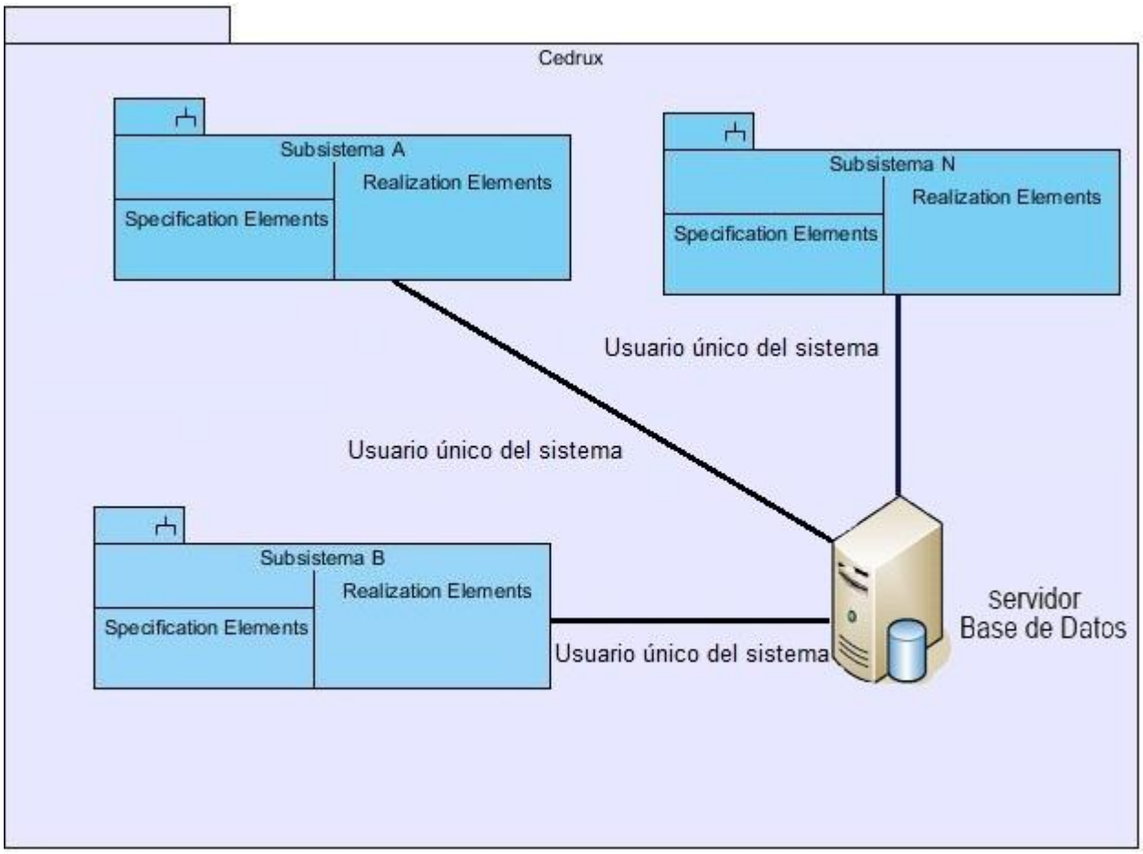

Figura 8 Conexión a la base de datos a través de un solo usuario.

Esta es la más sencilla de implementar debido a que requiere menos configuración a nivel de datos y de clases del sistema.

Su implementación en el marco de trabajo y en el sistema Cedrux requeriría un mínimo de esfuerzo para cambiar la configuración de la base de datos debido a que la complejidad de la propuesta no es muy elevada.

2.2.2.2 Conexión por usuario

Se trata de crear una conexión para cada usuario de los distintos subsistemas para realicen su trabajo en la base de datos a través de ella:

- Crear para cada usuario del subsistema un rol de base de datos con el que acceder, de manera tal que cuando un usuario acceda a un subsistema determinado, con el rol que se conecta a la base de datos es con un rol específico para él, de manera tal que toda la transacción se realiza con una sola conexión pero con la diferencia que se tiene a nivel de datos una mayor seguridad porque se sabe que realizó cada usuario, que no se tiene cuando existe una sola conexión para todo el sistema porque a nivel de base de datos todas las peticiones llegan a través de un mismo rol, por lo tanto no se puede diferenciar que usuarios accedieron o modificaron información del sistema a nivel de datos.

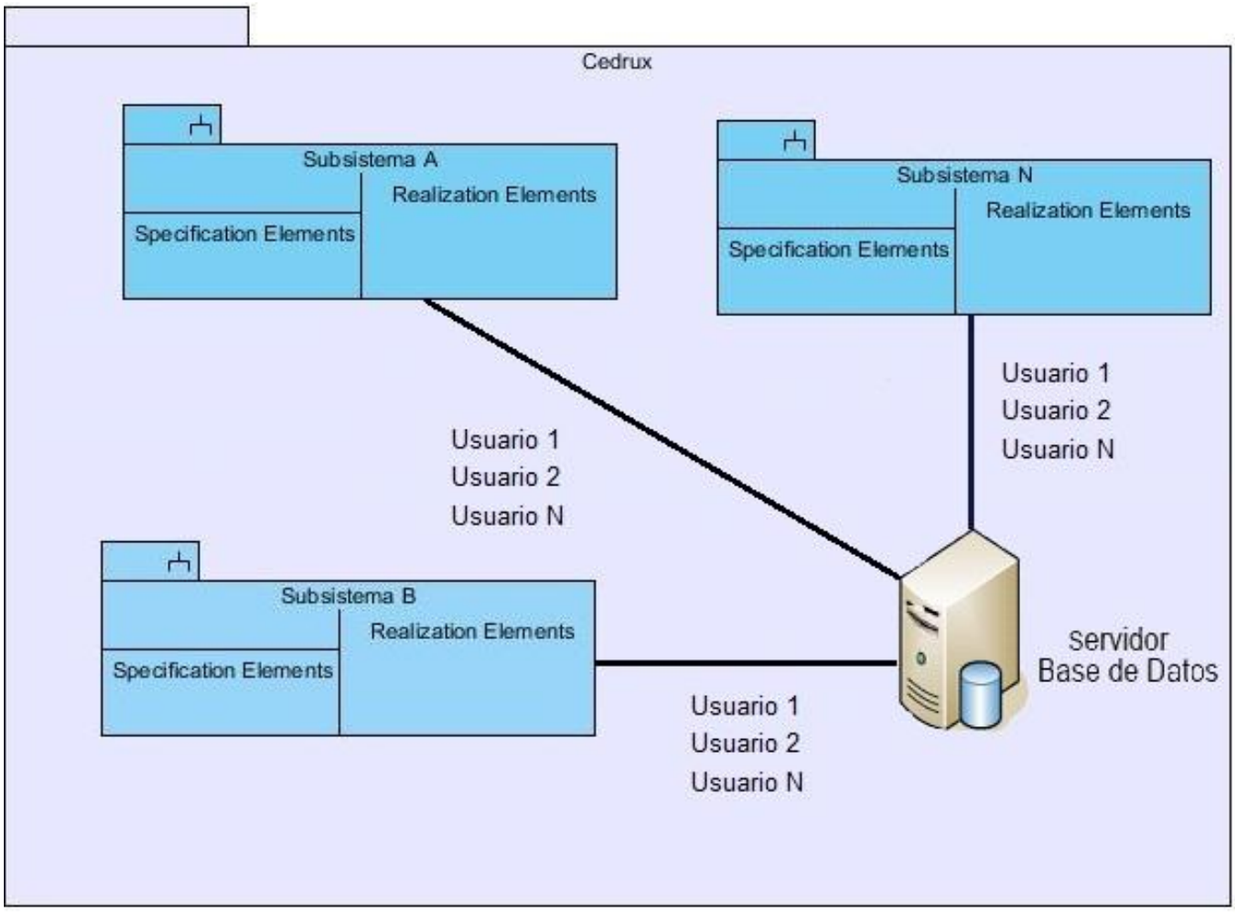

Figura 9 Conexión a la base de datos usando múltiples usuarios en cada subsistema.

Esta solución es un poco más difícil de implementar debido al gran volumen de configuración que requiere a nivel de base de datos para establecer el acceso, permisos y configuración de los usuarios en cada subsistema.

Para el acceso a datos existen hoy conexiones diferentes, una para cada subsistema, después de creados los usuarios nuevos para esta solución, cada uno de esos nuevos usuario tendría los permisos necesarios para modificar en cualquier esquema que su subsistema utilice.

Esta mantiene además las relaciones de llaves foráneas entre subsistemas del Cedrux, y como no modifica las clases dentro de los componentes, sigue existiendo la imposibilidad de instanciar clases dentro de un componente que pertenezcan a otro subsistema, pero reduciría al igual que la primera variante, en gran medida la ocurrencia de errores en las transacciones.

2.2.3 Reestructuración de los componentes

Cedrux tiene una estructura interna como se muestra en la siguiente imagen:

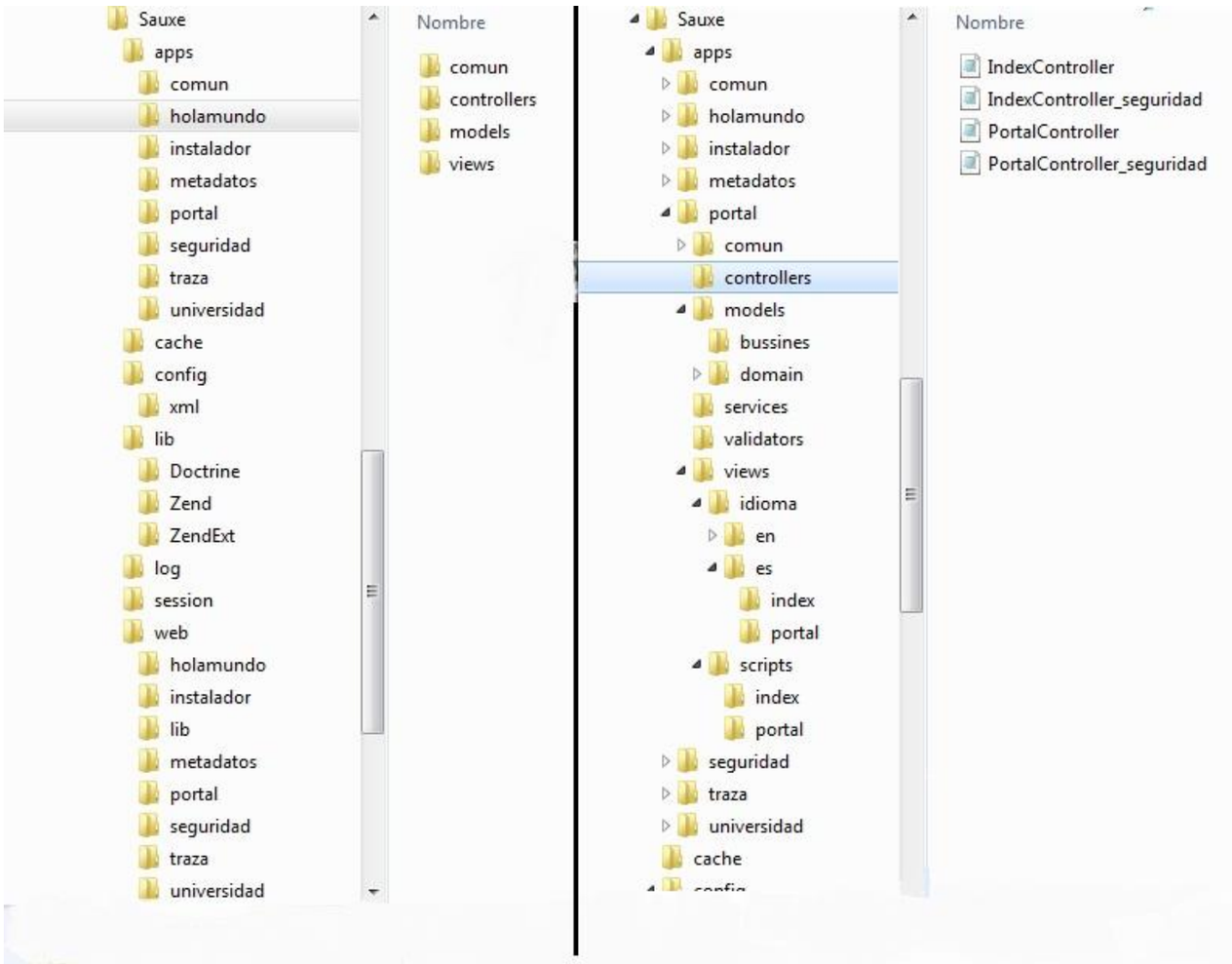

Figura 10 Estructura de Sauxe

Cada subsistema contiene las carpetas a*pp, cache, config, lib, log, sesión* y *web. App* es la carpeta que contiene toda la lógica fundamental del negocio y los componentes de los subsistemas que tienen dentro de ellos las carpetas *comun*, *controller, model* y *view.* La carpeta web es la que contiene todas las interfaces de los componentes. Dentro de app en cada componente las carpetas se encuentran de esta forma:

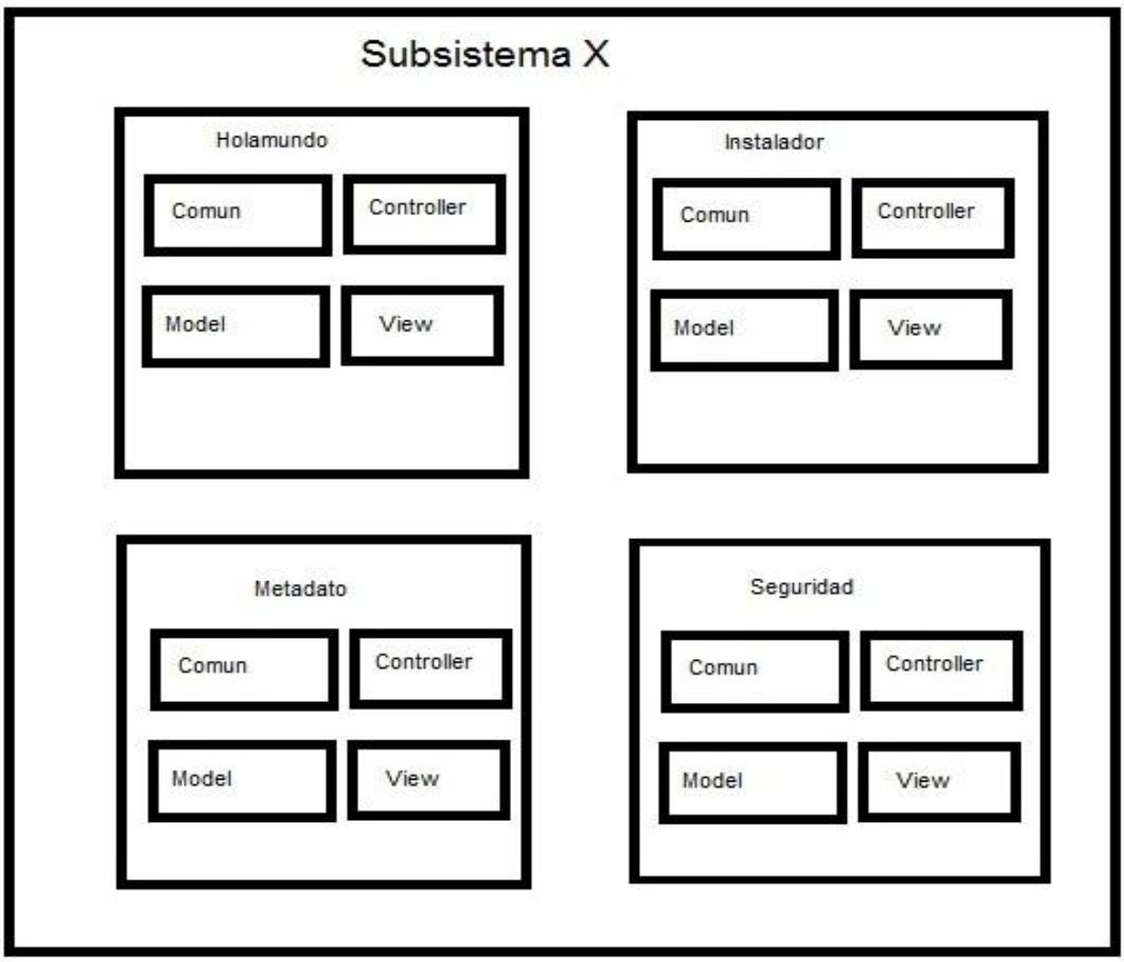

Figura 11 Vista interna de las carpetas del sistema.

En la figura 11 se muestra como el rectángulo externo es el subsistema x y los rectángulos interiores representan a sus componentes, luego dentro de cada componente se repiten las mismas carpetas *comun controller, model y veiw.*

Symfony implementa su arquitectura de forma que las clases del controlador, las clases del negocio y del domino de un subsistema están en una sola carpeta como se explica en el epígrafe 1.5.1, quedando de la siguiente forma:

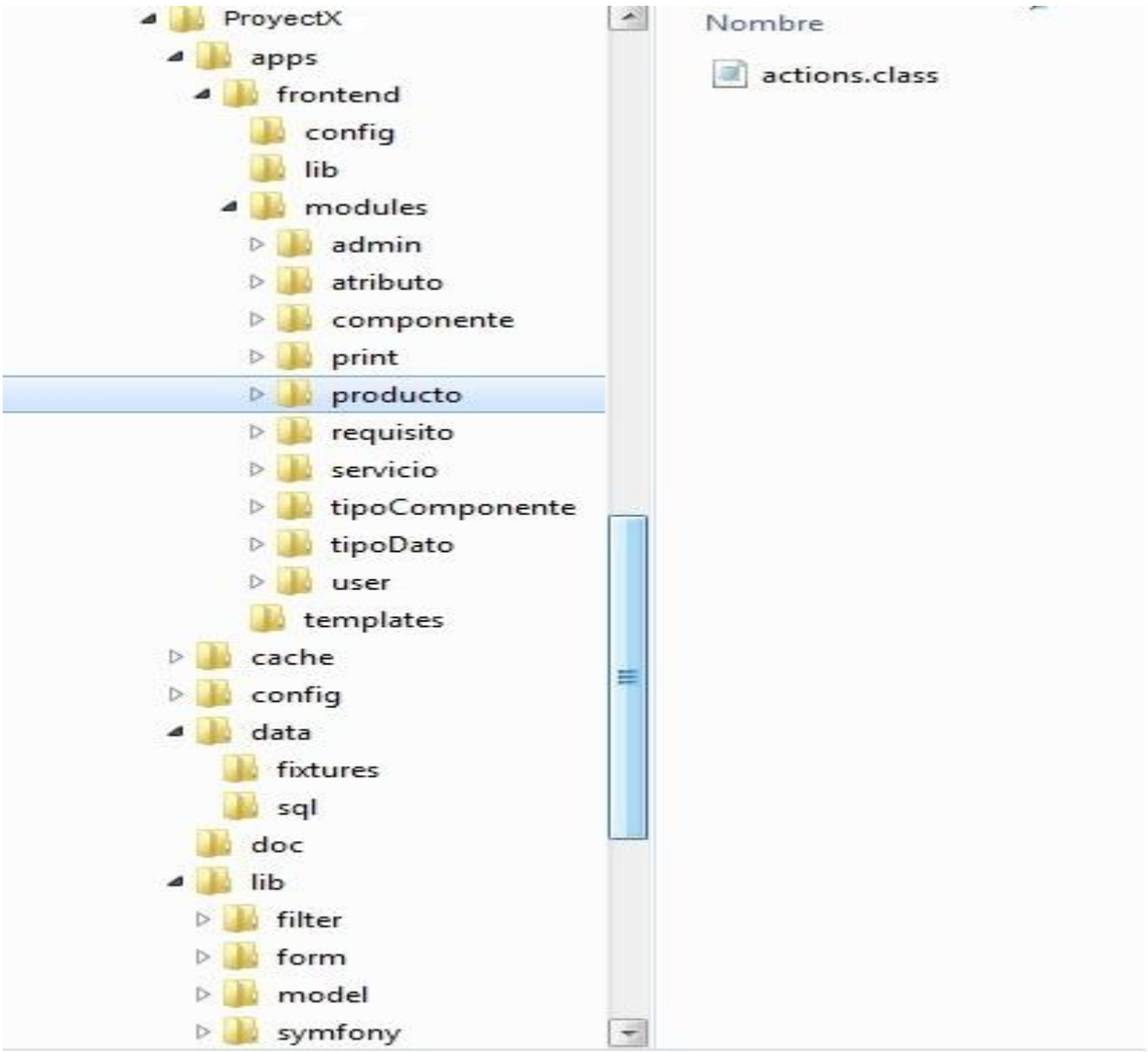

Figura 12 Ejemplo de estructura de carpetas y clases del marco de trabajo Symfony.

La solución que se plantea dejaría el subsistema X de manera tal que todos los módulos del subsistema dejen de existir estructuralmente. Todas las clases que antes se encontraban en carpetas distintas en módulos distintos, ahora van a estar en sus respectivas carpetas *comun, model controller y veiw*, de la siguiente forma:

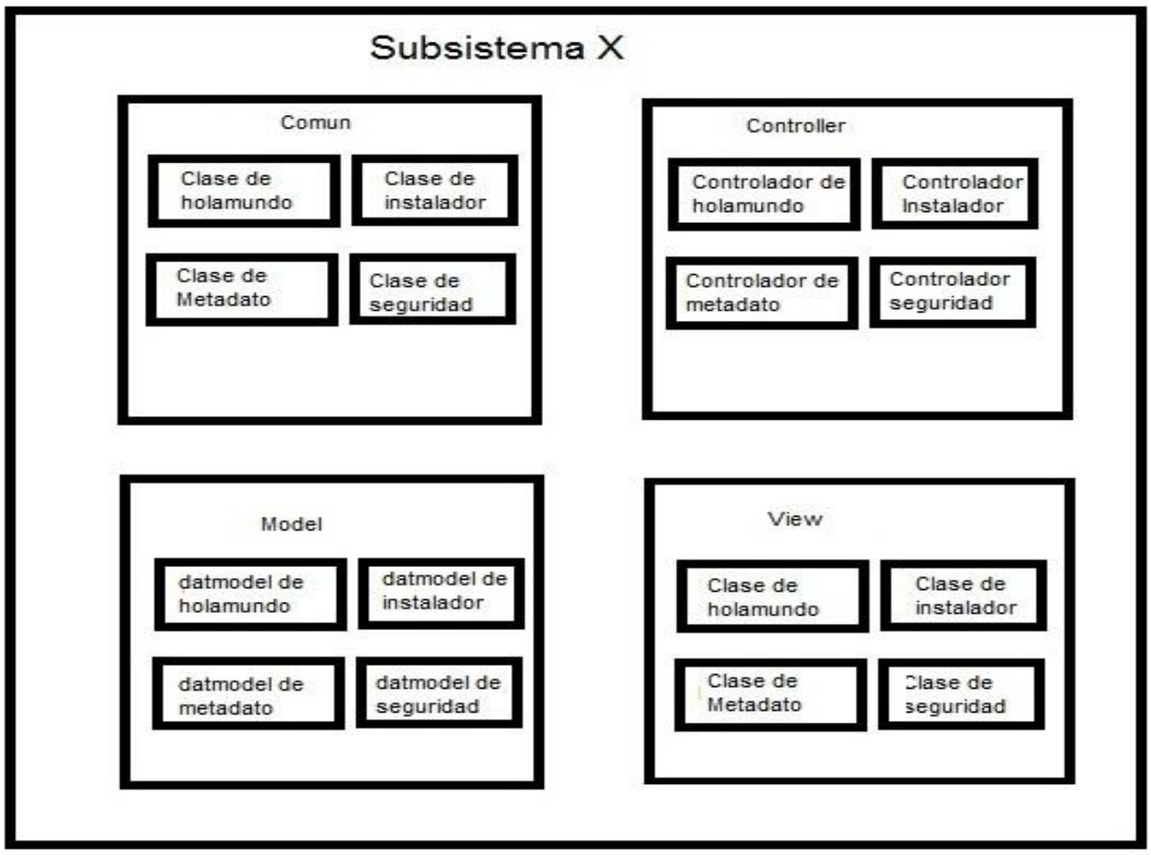

Figura 13 Sistema de archivos después de aplicada la solución.

Esta solución posibilita instanciar cualquier clase en un componente que pertenezca a otro componente, aunque se mantiene todavía la imposibilidad de invocar clases de otros subsistemas. Esta situación facilita la tarea de los programadores del sistema, a los cuales le será muy difícil trabajar con las clases necesarias de otros componentes para satisfacer determinadas necesidades en las clases del sistema. Esta solución pudiera ser implantada sin grandes esfuerzos en cada subsistema de Cedrux pues su complejidad es baja.

Esta propuesta está pensada para el marco de trabajo, que podría utilizarla para aplicaciones que sean desarrolladas en un futuro, al estar Cedrux ya implementado, se modificaría en todos sus subsistemas de forma independiente a las modificaciones del Sauxe.

2.3 Manejo de avisos y alertas.

2.3.1 Módulo de avisos y alertas del sistema de gestión universitaria.

Para la solución que se presentará se hizo un estudio en el epígrafe 1.6.8, capítulo 1, del trabajo "*Módulo de notificaciones y alertas del sistema de gestión universitaria*".

En la figura 14 se muestra una vista del módulo de avisos y alertas del Sistema de Gestión Universitaria.

Figura 14 Vista módulo de avisos del sistema de gestión universitaria.

La solución parece viable para el sistema de gestión universitaria, pero la factibilidad de su uso en el marco de trabajo Sauxe no es muy considerable, pues los avisos y alertas se gestionan usando CodeIgniter como marco de trabajo base, por lo que habría además que modificar el componente Zend del marco de trabajo Sauxe, el tipo de notificación (a través de correo electrónico) no era la más conveniente para el sistema Cedrux por lo que se recomienda revisar otras soluciones basadas en técnicas de comunicación asíncronas como el AJAX, el COMET o el XMPP que brindarían mejores prestaciones para el marco de trabajo Sauxe y el sistema Cedrux.

2.3.2 Sistema COMET

Para plantear esta segunda solución se analizó la tecnología COMET, vista en el epígrafe 1.6.3 del capítulo 1; el COMET permite a los servidores web sin ninguna necesidad de petición resolver el problema de la mensajería instantánea través de dos técnicas de comunicación: El lon-polling y el websockets.

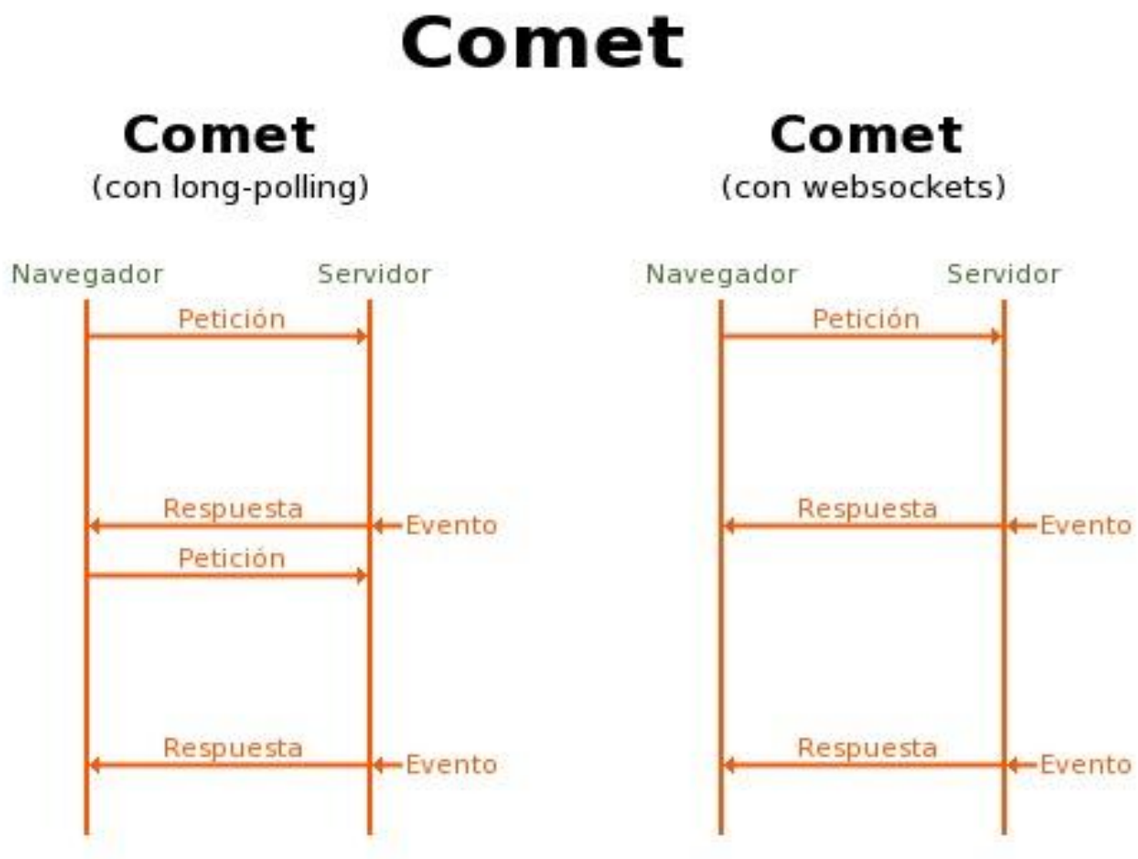

Figura 15 COMET.

Después de analizar este sistema de mensajería se llega a la conclusión de que requeriría grandes recursos físicos en el servidor para manejar un gran número de usuarios de un sistema, y sobrecarga demasiado el servidor web que usaría. Además sería necesario introducir elementos de los marcos de trabajo como DojoToolkit, Teleport y Pushlets, que implementan APIs para COMET. Además están las desventajas vistas en la investigación de esta herramienta en el capítulo 1. Incluso así puede ser implementado en el marco de trabajo Sauxe, no sin requerir un alto grado de investigación para su implementación por parte de los desarrolladores.

2.3.3 Creación del módulo de avisos y alertas para Sauxe

Para el manejo de alertas se ha realizado un estudio de las posibles soluciones que implementan otros sistemas, muchas de las cuales no eran viables por el consumo excesivo de recursos de los servidores o la dificultad de implementación de la solución. La posible solución para este problema fue encontrada en el artículo *Build a web-based notification tool with XMPP* publicado por IBM.

La tecnología XMPP es muy conocida en la comunidad UCI debido a que el *Jabber*[3] que es el chat interno de la Universidad, está basado en este sistema. Por lo tanto pruebas de rendimiento están de sobra

[3] Forma común en que se le llama en la universidad de las Ciencias Informáticas al chat

comprobadas, este sistema se sobre carga cerca ya de los 2500 usuarios online, una cifra realmente impresionante, más que suficiente para ser usado en Cedrux.

Modelo conceptual

Se tiene al sistema de gestión Cedrux como un tipo de software que va a permitir a las empresas controlar la información que se genera en cada departamento y en cada nivel de la misma. Este sistema contiene otros sistemas, los cuales tienen asociados usuarios, que al mismo tiempo son subscriptores del Msg_Bus, el cual es responsable de enviar notificaciones a los usuarios y además a través de un catálogo que mantiene relacionados a los componentes a través de eventos siendo suscriptores de igual forma que los son los usuarios, Msg_Bus es responsable de notificar a otros componentes.

Diagrama de clases del diseño

El diagrama de clases del diseño describe la estructura del sistema, mostrando sus clases, asociaciones, atributos, métodos y sus relaciones entre ellos. A continuación en la figura 16 se muestra el diagrama de clases del diseño con estereotipos web para la propuesta del modulo de avisos y alertas para Sauxe:

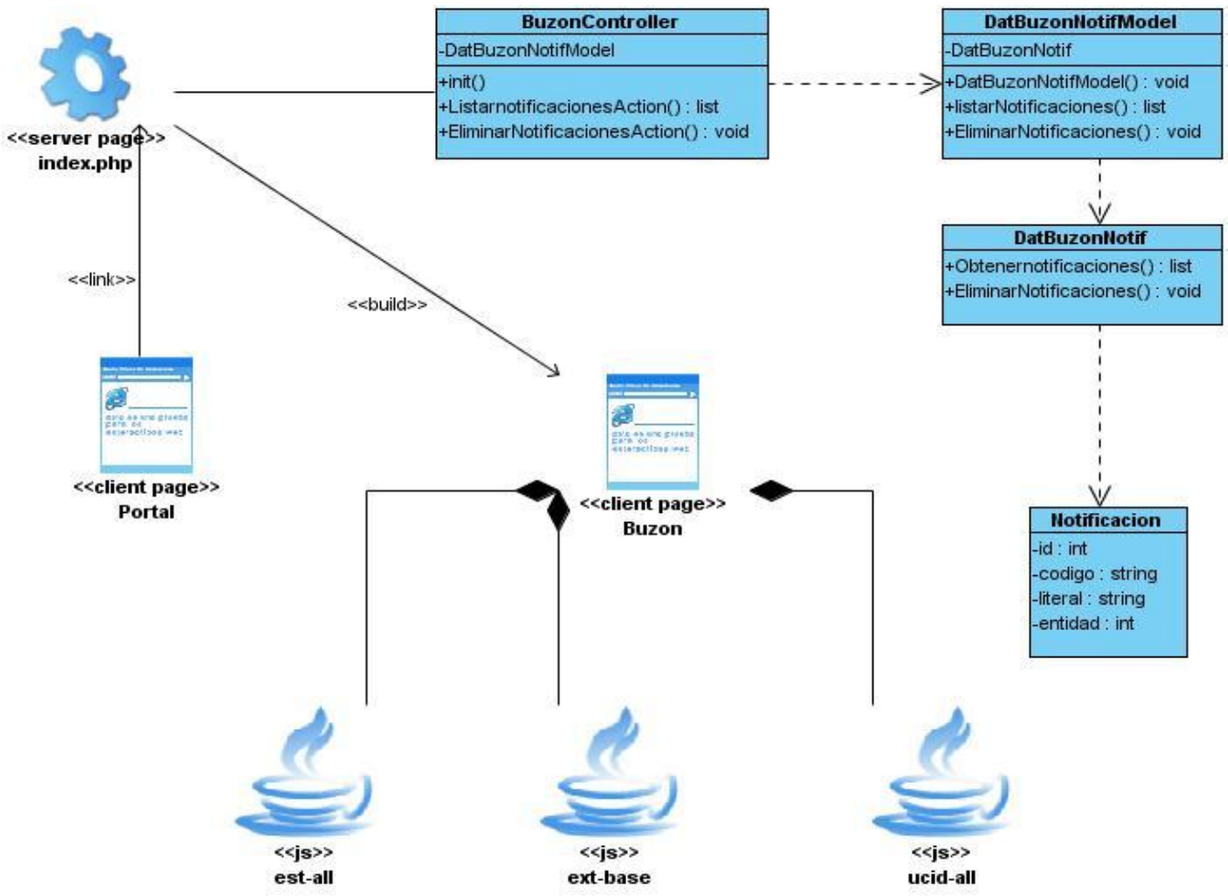

Figura 16 Diagrama de clases del diseño con estereotipos web del módulo de avisos y alertas para Sauxe.

Para implementar esta solución se agregan al marco de trabajo Sauxe un grupo de librerías que serán usadas por el sistema de mensajería para su trabajo así como la creación de una interfaz para la gestión de los mensajes y los buzones de mensajería para cada usuario del sistema y es una solución que podría ser objeto de una investigación para ser implementada en el centro CEIGE.

Su implantación a nivel de sistema no requeriría de un gran esfuerzo debido a la facilidad de incluir las librerías que serán usadas por el marco de trabajo y posteriormente en Cedrux. Se recomienda para cuando se termine de implementar la solución, la creación de un tutorial para su implantación en el sistema Cedrux que facilite su uso.

2.4 Conclusiones parciales

Se han detallado las posibles soluciones encontradas en la investigación para solucionar los problemas de la gestión de transacciones y el manejo de avisos y alertas, a través de modificaciones e incorporación de herramientas en el marco de trabajo Sauxe.

Las posibles soluciones presentadas para el problema de la gestión de transacciones, de ser implementadas solucionarían el problema en cuestión, son tres soluciones orientadas a solucionar distintos inconvenientes presentes en el sistema y el autor recomienda que sean implementadas todas. Estas soluciones por lo general no tienen una alta complejidad, pero sería necesario refactorizar gran cantidad de código en el sistema Cedrux y la reestructuración del marco de trabajo Sauxe. Además las soluciones que tienen que ver con la base de datos necesitarían gran cantidad de configuración para los cambios. En general estas propuestas de solución deben eliminar el problema de la gestión de transacciones presente en el marco de trabajo Sauxe.

En cuanto a las soluciones para la gestión de avisos y alertas, se recomienda la tercera solución, usando la tecnología XMPP, que puede solucionar de forma definitiva la carencia por parte del marco de trabajo Sauxe de un modulo de gestión de avisos y alertas.

Validar las soluciones propuestas será el próximo objetivo a cumplir en la investigación.

Capítulo 3

3. El método Delphi

Valorando que el objetivo de esta investigación se basa en una propuesta de soluciones, se define el Método Delphi como el más apropiado para desarrollar la validación científica de este trabajo. Este método se considera conveniente debido a que su fuente de información proviene de un grupo de personas que poseen un conocimiento elevado de la materia a evaluar, en este caso los especialistas de los sistemas Sauxe y Cedrux pueden brindar mediante su respuesta a cuestionarios sus criterios sobre la calidad y efectividad de los resultados de la investigación.

Este método presenta dos características fundamentales:

• Anonimato:

Durante el proceso de evaluación ningún experto conoce la identidad del resto que integran el grupo. Por lo que se evita otra influencia sobre las respuestas que no sea la lógica de los argumentos presentados (STIGARRAGA, 2003).

• Respuesta del grupo en forma estadística:

La información obtenida no es solo un punto de vista común entre los expertos, sino que una vez procesada mediante términos estadísticos se convierte en una importante herramienta para la toma de decisiones del investigador (STIGARRAGA, 2003).

Características que serán usadas durante el desarrollo este capítulo.

3.1 Desarrollo del Método Delphi.

Para la aplicación de este método se definen 3 fases (Estévez y Bravo, 2005):

1. **Fase preliminar:** Se delimita el contexto, los objetivos, el diseño, los elementos básicos del trabajo y la selección de los expertos.

2. **Fase exploratoria:** Elaboración y aplicación de los cuestionarios según sucesivas vueltas, de tal forma que con las respuestas más comunes de la primera se confecciona la siguiente.

3. **Fase final:** Análisis estadísticos y presentación de la información.

En la presente investigación se determinaron los siguientes pasos:

1.) Se definieron los criterios que serán utilizados en la evaluación atendiendo a las categorías de mayor interés. Se evaluó de manera independiente cada elemento en un rango de 1-10 estableciendo una escala, pues cada uno expresa un elemento determinista (STIGARRAGA, 2003):

- Del 0 al 3 la propuesta tiene un nivel Bajo.
- Del 5 al 7 la propuesta tiene un nivel Medio.
- Del 8 al 10 la propuesta tiene un nivel Alto.

2.) Para verificar el nivel que tienen los expertos sobre el dominio del tema se determinó el Coeficiente de competencia (K) a partir de su conocimiento o información sobre el tema (Kc) y el Coeficiente de argumentación o valoración (Ka) mediante la siguiente fórmula: K= ½ (kc + ka) (STIGARRAGA, 2003).

El coeficiente de conocimientos se obtuvo de la primera pregunta del cuestionario la cual recoge una autoevaluación del posible experto.

1	2	3	4	5
			X	

Tabla 2 Autovaloración del coeficiente de conocimiento.

En esta tabla las casillas están enumeradas del 1 al 5 y representan el conocimiento que se tiene de un tema determinado en esa escala. El posible experto debe marcar con una X la casilla que refleje el conocimiento que posee respecto al tema investigado. De esta forma, si marca la casilla 1 es que no tiene conocimiento alguno del tema y viceversa, si marca la casilla 5 es que tiene un gran conocimiento. Una vez realizada la selección, el número que representa la casilla se multiplica por 0,1 para poder ajustarla a la teoría de las probabilidades (STIGARRAGA, 2003).

Para determinar el coeficiente de argumentación o valoración (Ka) se ofrece una tabla con cierta información (encuesta de autovaloración). El experto debe marcar, según su criterio, los elementos que le permiten argumentar su evaluación del nivel de conocimiento seleccionado anteriormente (Ver Anexo 1). Las marcas de los expertos se traducen a puntos teniendo en cuenta la escala que se muestra en la siguiente tabla de escalas:

	Fuentes	Alto(A)	Medio(M)	Bajo(B)
P1	Análisis Teóricos realizados por Ud.	0,3	0,2	0,1
P2	Experiencia de trabajo en el campo	0,4	0,3	0,2
P3	Trabajos de autores nacionales	0,4	0,3	0,2
P4	Trabajos de autores Internacionales	0,4	0,3	0,2
P5	Su propio conocimiento del estado del problema	0,4	0,3	0,2
P6	Su intuición	0,3	0,2	0,1

Tabla 3 Escala de puntos para la determinación del coeficiente de argumentación.

El experto debe marcar en este caso los elementos que le permitan argumentar su evaluación del nivel de conocimiento que seleccionó en la pregunta 1. Los valores que se representan no son conocidos por los encuestados, son valores asignados luego por el encuestador para la realización de los cálculos.

Una vez calculados los coeficientes de conocimiento Kc y de argumentación Ka, se calculó el coeficiente de competencias K.

Capítulo 3: Validación de las propuestas

Los resultados obtenidos del coeficiente de competencia en el cuestionario de autoevaluación aplicado a los expertos seleccionados pueden verse en la siguiente tabla:

No.	Expertos	Conocimiento	P1	P2	P3	P4	P5	P6	Ka	Kc	K	Competencia
1	E1	4	0,3	0,4	0,3	0,2	0,4	0,2	1,8	0,4	1,1	Alta
2	E2	4	0,2	0,4	0,2	0,2	0,3	0,3	1,6	0,4	1	Alta
3	E3	4	0,1	0,3	0,2	0,3	0,4	0,2	1,5	0,4	0,95	Alta
4	E4	5	0,3	0,4	0,2	0,2	0,4	0,3	1,8	0,5	1,15	Alta
5	E5	4	0,1	0,2	0,4	0,3	0,3	0,3	1,6	0.4	1	Alta

Tabla 4 Coeficiente de competencia de expertos.

La interpretación de los coeficientes de competencias es la siguiente: (STIGARRAGA, 2003)

• Si 0,8 < k < 1,0 Coeficiente de competencia alto.

• Si 0,5 < k < 0,8 Coeficiente de competencia medio.

• Si k < 0,5 Coeficiente de competencia bajo.

Los 5 expertos encuestados poseen un coeficiente de competencia alto. Por lo cual se tomó la decisión de que los 5 sean incluidos en el grupo de expertos para la evaluación de la propuesta.

3.) Luego que se analizan los resultados obtenidos en la Tabla 4 se les entrega a los expertos la segunda encuesta (Ver Anexo 2) y las propuestas de solución.

3.1.1 Definición de las variables

Para los cálculos estadísticos es necesario dejar claro el significado de las variables que serán usadas:

C: número de criterios que van a evaluarse.

E: número de expertos que realizan la evaluación.

ΣEi: pesos dados a cada criterio, por los expertos.

ΣE: sumatoria total de los pesos

ΣEi correspondiente a cada criterio.

3.1.2 Formulación de las hipótesis

Se verificó la consistencia en el trabajo de los expertos, mediante la utilización del coeficiente de concordancia de Kendall y la prueba de Friedman con el estadígrafo Chi cuadrado (X2). (Legendre, 2005)

Este estadígrafo debe resultar entre 0 y 1, donde un resultado cercano a 1 refuta la hipótesis nula de Kendall acerca de la independencia de la prueba de la hipótesis de Friedman. Los valores del Coeficiente de Concordancia (W) deben oscilar entre 0 y 1 (0 < W < 1), mientras mayor sea el valor de W, es decir, cuanto más se acerque a uno, mayor será la dependencia entre las respuestas (Estévez y Bravo, 2005).

Para la prueba de la hipótesis de Friedman se plantean la nula y la alternativa de la siguiente manera:

- **H0**: Existe una opinión común de preferencia entre los expertos.
- **H1**: No existe concordancia entre los criterios de los expertos.

Se tienen en cuenta las siguientes ecuaciones estadísticas para obtener el grado de acuerdo entre los expertos en cuanto a los criterios evaluados en las encuestas (Estévez y Bravo, 2005):

MΣE: peso medio total de los criterios.

$$M\sum E = \frac{\sum E}{C}$$

ΔC: desviación de la media de cada criterio.

ΔC = ΣE - MΣE

S: dispersión

$$S = \sum \left(E_i - \frac{\sum E}{C} \right)^2 = \sum \Delta C^2$$

W: coeficiente de concordancia de Kendall.

$$W = \frac{12 * S}{E^2 (C^3 - C)}$$

X^2: estadígrafo Chi cuadrado

X^2= E (C-1) W

Una vez analizadas las respuestas de la primera encuesta se obtienen los resultados que se muestran a continuación en la **Tabla 5**.

	E1	E2	E3	E4	E5			Peso relativo
Criterios						ΣE	(ΔC)2	5,2
C 1	8	8	8	5	5	34	0,081632653	6,8
C 2	5	10	5	5	5	30	13,79591837	6
C 3	8	5	3	8	8	32	2,93877551	6,4
C 4	10	10	8	3	5	36	5,224489796	7,2

C 5	10	10	8	3	10	41	53,08163265	8,2
C 6	8	8	3	0	8	27	45,08163265	5,4
C 7	8	10	3	5	10	36	5,224489796	7,2
						S =	**125,4285714**	
		W =	**0,031027476**			**Estadígrafo**		
	Decisión					χ2 =	2,707235622	
	No puedo rechazar Ho(Concordancia significativa)					**Valor del estadígrafo en la tabla**		
						χ2 ≥	14,68365662	
	Índice de Aceptación		**6,742857143**					

Tabla 5 Evaluación del peso de los criterios emitidos por los expertos.

El resultado muestra un W=0,0310, menor de la media entre 0 y 1. Para probar la hipótesis se compara el resultado de X2 de Friedman con el tabulado en la tabla de estándares para α= 0,1 y c– 1 = 6, X2 calculada < X2 (α, c-1) entonces se rechaza H1 y se infiere que existe concordancia de criterios preferenciales entre los expertos, al considerar válida la hipótesis alternativa H0.

La comparación en correspondencia con las variables anteriores resulta 6,742< 14,683. Luego el resultado calculado es menor que el tabulado, se acepta entonces H0 y se puede decir que existe concordancia en el trabajo de los expertos (Estévez y Bravo, 2005).

3.2 Procesamiento de la información

Se le aplicaron 7 preguntas a cada uno de los expertos en la encuesta. De ellas, 3 se relacionaban con la categoría Mérito Científico, 3 con la categoría Implantación, 1 con la categoría Aceptación.

Categoría: Mérito Científico

1. Calidad de la investigación.

2. Valor científico de la propuesta.

3. Novedad científica.

Categoría: Implantación

4. Posibilidad de aplicación de la propuesta.

5. Mejoras de la propuesta con respecto a la solución existente.

6. Adaptabilidad de la propuesta a las características del proyecto

Categoría: Aceptación

7. Aceptación de la propuesta por el encuestado.

A continuación serán graficados los niveles de la propuesta en cuanto a las categorías anteriormente mencionadas:

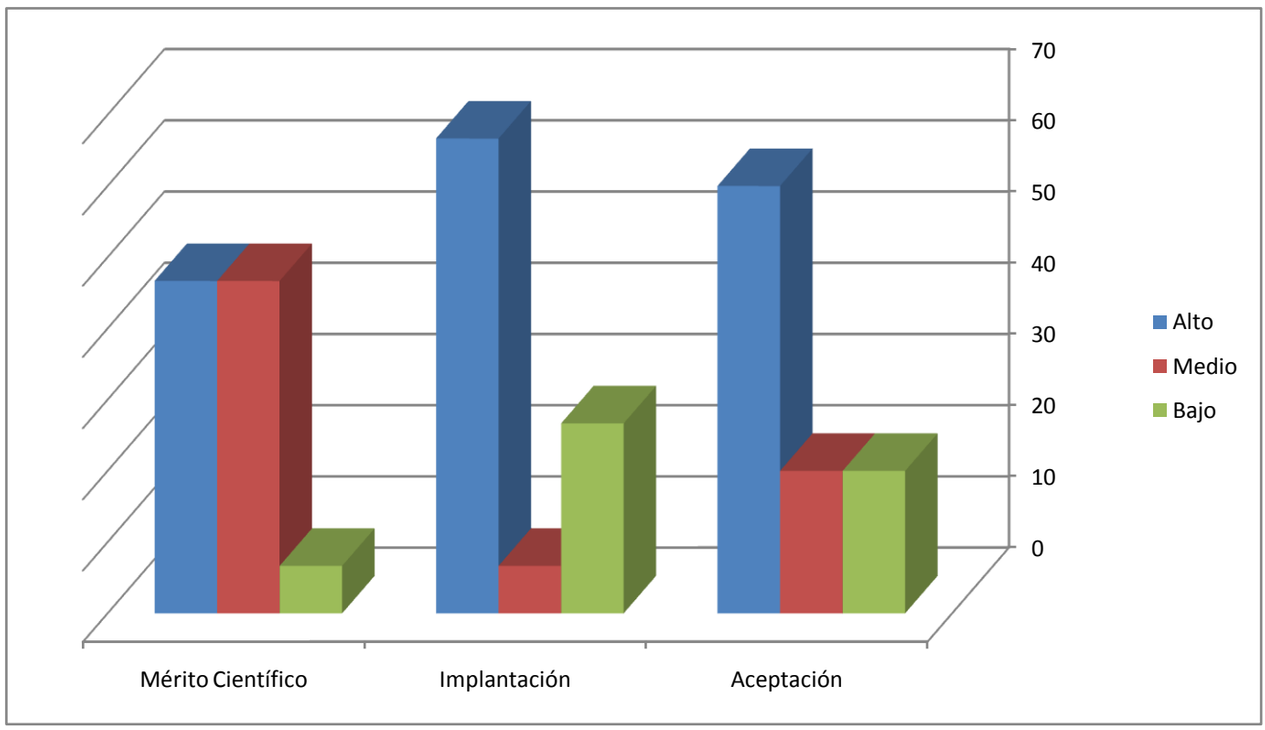

Figura 17 Gráfica sobre el criterio de los expertos en cuanto a los niveles de la propuesta por categoría.

- En la categoría Mérito Científico se obtiene que la propuesta tiene un nivel Alto de 46,66%, un nivel Medio de 46,66% y un nivel bajo de 6,66%; de acuerdo con los criterios emitidos.
- En la categoría Implantación se obtiene que la propuesta tiene un nivel Alto de 66,66%, un nivel Medio de 6,66% y un nivel bajo de 26,66%; de acuerdo con los criterios emitidos.
- En la categoría Aceptación se obtiene que la propuesta tiene un nivel Alto de 60%, un nivel Medio de 20% y un nivel bajo de 20%; de acuerdo con los criterios emitidos.

3.3 Conclusiones parciales

Como resultados de la validación mediante el método Delphi se determinó que existe una concordancia de criterios en cuanto a la aceptación de la propuesta de solución, siendo la misma parcialmente novedosa, con altas posibilidades de implantación en el marco de trabajo Sauxe y con un nivel de aceptación elevado por parte de los especialistas del centro CEIGE.

Con la aplicación del método Delphi queda validada la propuesta de solución.

Conclusiones

En la investigación se realizó un estudio del estado del arte para obtener un conocimiento previo del estado actual de las tecnologías relacionadas con el objeto de estudio del trabajo. Un estudio que aporto el conocimiento sobre el marco de trabajo Sauxe, en aras de poder comprender como funciona, qué herramientas lo componen y cuál es la función de cada una de ellas en el marco, objetivo que fue cumplido de forma satisfactoria.

Se formularon las propuestas de soluciones para cada uno de los problemas identificados, donde se hace un análisis crítico de las modificaciones que serían necesarias en el marco de trabajo y las ventajas y desventajas de su implementación, propuestas que fueron consideradas por los especialistas del centro CEIGE y se validaron los resultados a través del método estadístico Delphi para ver la concordancia de los expertos que fueron encuestados, llegándose a la conclusión de que las propuestas son correctas y la posibilidad de implementarlas es muy elevada.

Al concluir el presente trabajo se puede decir que el objetivo general de la investigación fue cumplido a partir del cumplimiento de los objetivos específicos.

Recomendaciones

- Mantener documentada toda la futura investigación sobre el tema para poder seguir avanzando en las futuras soluciones para estos u otros problemas que se puedan presentar en el marco de trabajo y directamente en el sistema de gestión integral Cedrux.
- Documentar la implementación de las posibles soluciones propuestas para mantener claros los cambios realizados y hacer más fácil cualquier estudio sobre el tema.

Bibliografía

Barros, Laura. 2011. . *Barros, Laura. Programacion Concurrente y Distribuida.* 2011.

Baryolo Gómez, Oiner y García Tejo, Darien. 2009. *XVI Fórum de Ciencia y Técnica.* 2009.

Baryolo, Oiner Gómez, y otros. 2008. *Plantilla Registro de la Propiedad intelectual(Sauxe).* 2008.

Cadavid, Andrés Navarro. 2008. Sistema de investigación de la Universidad Icesi. Sistema de investigación de la Universidad Icesi. [En línea] 2008. http://www.icesi.edu.co/investigaciones/proyecto.do?id=38.

Castillo, Juan Marmol. 2009. 2009. Universidad de Informática de Murcia. [En línea] 2009. http://dis.um.es/~jmolina/Persistencia%20de%20Objeto%20JDO.pdf.

CodeIgniter. 2012. Sitio oficial de CodeIgniter. [En línea] 2012. http://codeigniter.com/.

Corzo, Giancarlo. 2008. ExtJS lo bueno, lo malo y lo feo. [En línea] 2008. [Citado el: 8 de 4 de 2012.] http://blogs.antartec.com/desarrolloweb/2008/10/extjs-lo-bueno-lo-malo-y-lo-feo/.

Doctrine.org. 2012. Doctrine comunity. [En línea] 2012. http://www.doctrine-project.org/.

EllisLab inc. 2012. Manual de CodeIgniter en español. [En línea] 2012. http://neonetsi.com.ar/seppo/traduccion/.

Equipo Arquitectura del ERP Cedrux. 2008. *Especificación Técnica para el marco de la arquitectura.* 2008.

Estévez y Bravo, M y Arrieta J. 2005. *El método Delphi. Su implementación en una estrategia didáctica para la enseñanza de las demostraciones geométricas.* s.l. : Revista Iberoamericana de Educación, 2005. 1681-5653.

Facultad de ingenieria Uruguay. 2012. Facultad Ing Urg. [En línea] 2012. [Citado el: 15 de 11 de 2011.] http://www.fing.edu.uy/~asabigue/prgrado/2004eofgl/contenido/archivos/Anexo-III.pdf.

Fowler, M. 2006. Inversion of Control. [En línea] 2006. http://www.martinfowler.com.

Gutiérrez, Javier J. 2009. ¿Qué es un framework web? [En línea] 2009. http://www.lsi.us.es/~javierj/investigacion_ficheros/Framework.pdf.

INFORMATICA HOY. 2012. INFORMATICA HOY. [En línea] 2012. [Citado el: 14 de 6 de 2012.] http://www.informatica-hoy.com.ar/software-erp/Los-costos-de-implementacion-del-ERP.php.

Jabber.org. 2012. Mensajeria Instantanea Libre. [En línea] 2012. [Citado el: 16 de 11 de 2011.] http://www.jabberes.org/glosario.

Jacobson, I., Booch, G., & Rumbaugh, J. 2000. *In The Unified Software Development Process.* s.l. : Addison-Wesley, 2000.

Jive Software. 2012. Ignite realtime. [En línea] 2012. http://www.igniterealtime.org/projects/openfire/.

León, Rolando Alfredo Hernández. 2002. *EL PARADIGMA CUANTITATIVO DE LA INVESTIGACIÓN CIENTIFICA.* Ciudad de la Habana : EDUNIV, 2002. 959-16-0343-6.

Bibliografía

Leopoldo, Carlos. 2007. end Framework, una introducción. [En línea] 2007. [Citado el: 6 de 4 de 2012.] http://techtastico.com/post/zend-framework-una-introduccion/.

Mata, Manel Pérez. 2009. ¿Qué es Doctrine ORM? [En línea] 2009. [Citado el: 5 de 4 de 2012.] http://www.tecnoretales.com/programacion/que-es-doctrine-orm/comment-page-1/.

Oviedo Chaparro, Luis Enrique. 2010. *COMET: UN SIGUIENTE PASO AL AJAX MOVIENDO DE LAS APLICACIONES WEB TRADICIONALES A UN NUEVO ESTILO.* Asunción : Facultad de Ciencias y Tecnología, Universidad Católica de Asunción, 2010.

Potencier, Fabien y Zaninotto, François. 2008. *Symfony 1.2 La guía Definitiva.* France : Sensio SA, 2008.

Santiesteban, Diana Rosa Pérez, y otros. 2010. *MÓDULO DE NOTIFICACIONES Y ALERTAS DEL SISTEMA DE GESTIÓN UNIVERSITARIA.* 2010.

SENCIOLABS. 2009. *Doctrine ORM for PHP.* 2009.

ServerGrove. 2010. Doctrine. [En línea] 2010. [Citado el: 29 de 3 de 2012.] http://www.doctrine-project.org/.

STIGARRAGA, Eneko. 2003. *El método delphi.* San Sebastian : Universidad de Deusto, 2003. s.n.

Werdmuller, Ben. 2010. *Build a web-based notification tool with XMPP.* 2010.

Glosario de términos

1. Extender: Enseñarle a los usuarios como a partir de lo que actualmente se tiene, pueden reutilizar los componentes.

2. MySQL: Sistema de gestión de base de datos relacional, multihilo y multiusuario con más de seis millones de instalaciones.

3. Oracle: Sistema de gestión de base de datos relacional.

4. SQLite: Sistema de gestión de base de datos relacional.

5. SQL: Vulnerabilidad informática en el nivel de la validación de las entradas a la base de datos de una aplicación.

6. Rollback: Operación que devuelve a la base de datos a algún estado previo.

7. Layouts: Son capas. Se utilizan para crear la maquetación de un sitio.

8. Swing: Es una librería gráfica para Java.

9. Safari: Navegador web de código cerrado.

10. UML: Lenguaje Unificado de Modelado. Es un lenguaje para el desarrollo de software orientado a objetos, su propósito es visualizar, especificar, construir y documentar proyectos de software.

12. Script: Conjunto de instrucciones.

13. API: La interfaz de programación de aplicaciones es un conjunto de funciones residentes en bibliotecas.

Anexos

Anexo 1

Encuesta 1

Estimado compañero(a):

En el Centro de Informatización y Gestión de Entidades (CEIGE) de la Universidad de las Ciencias Informáticas se desarrolla una investigación sobre la búsqueda de soluciones que posibiliten mejoras en el marco de trabajo Sauxe usado por el centro para solucionar problemas presentes en dicho marco de trabajo.

En la investigación que se lleva a cabo para obtener el grado académico de Ingeniero en Ciencias Informáticas y que tiene como objetivo: "**Propuesta de principios tecnológicos para el Marco de Trabajo en el desarrollo de Cedrux** ", se requiere someter la propuesta al criterio de expertos.

Teniendo en cuenta que el autor considera que Ud. reúne los requisitos que lo avalan como experto, se le solicita que responda las preguntas que a continuación se presentan. Su valoración y los criterios que emita serán de gran valor para el perfeccionamiento de la propuesta. Una vez manifestada su disposición de colaborar con esta investigación, le pedimos que responda las siguientes preguntas antes de realizar el cuestionario.

Nombres y Apellidos:

_____

Institución: _____

Puesto de trabajo actual: _____

Licenciado: ___ Ingeniero: ___ Máster: ___ Doctor: ____ Otros: _____

Años de experiencia como docente: _____

Años de experiencia en el centro CEIGE: _____

Cargo que ocupa: _____

Años en el cargo: _____

Categoría docente: P. Inst.__ P. Asist. __ P. Aux.___ P. Tit. __

Además se necesita después de manifestada su disposición de colaborar en este importante empeño, una autovaloración de los niveles de información y argumentación que posee sobre el Marco de trabajo Sauxe.

1. Según su criterio, marque con una x, en orden creciente, el grado de conocimiento que usted tiene sobre el tema.

1	2	3	4	5

2. Entre las fuentes que le han posibilitado enriquecer su conocimiento sobre el tema, se someten a consideración algunas de ellas, para que las evalúe en las categorías de: Alto (A), Medio (M) y Bajo (B), colocando una x.

	Fuentes	Alto(A)	Medio(M)	Bajo(B)
P1	Análisis Teóricos realizados por Ud.			
P2	Experiencia de trabajo en el campo			
P3	Trabajos de autores nacionales			
P4	Trabajos de autores Internacionales			
P5	Su propio conocimiento del estado del problema			
P6	Su intuición			

Anexo 2

Encuesta a especialistas para someter a sus criterios la Propuesta de principios tecnológicos para el Marco de Trabajo en el desarrollo de Cedrux.

Compañero(a):

La presente encuesta forma parte de la aplicación del Método de valoración de los especialistas. Con este fin se solicita su valiosa colaboración, de antemano le aseguramos que sus criterios serán tenidos en cuenta para la aplicación de la propuesta. Valore el grado de factibilidad de cada pregunta de acuerdo a la siguiente escala: Muy adecuado (C1), Bastante adecuado (C2), Adecuado (C3), Poco adecuado (C4) No adecuado (C5).

Marque con una X el criterio que considere corresponde a cada pregunta.

Preguntas	Criterio del Experto				
	C1	C2	C3	C4	C5
1. Calidad de la investigación					
2. Valor científico de la propuesta.					
3. Novedad científica					
4. Posibilidad de aplicación de la propuesta					
5. Mejoras de la propuesta con respecto a la solución existente					
6. Adaptabilidad de la propuesta a las características del proyecto					
7. Aceptación de la propuesta por el encuestado.					