Universidad de las Ciencias Informáticas
Facultad 3

Descripción de Escenarios Tecnológicos para el desarrollo de Sistemas de Gestión Empresarial

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

Autor: Yesmín Caveda Báez

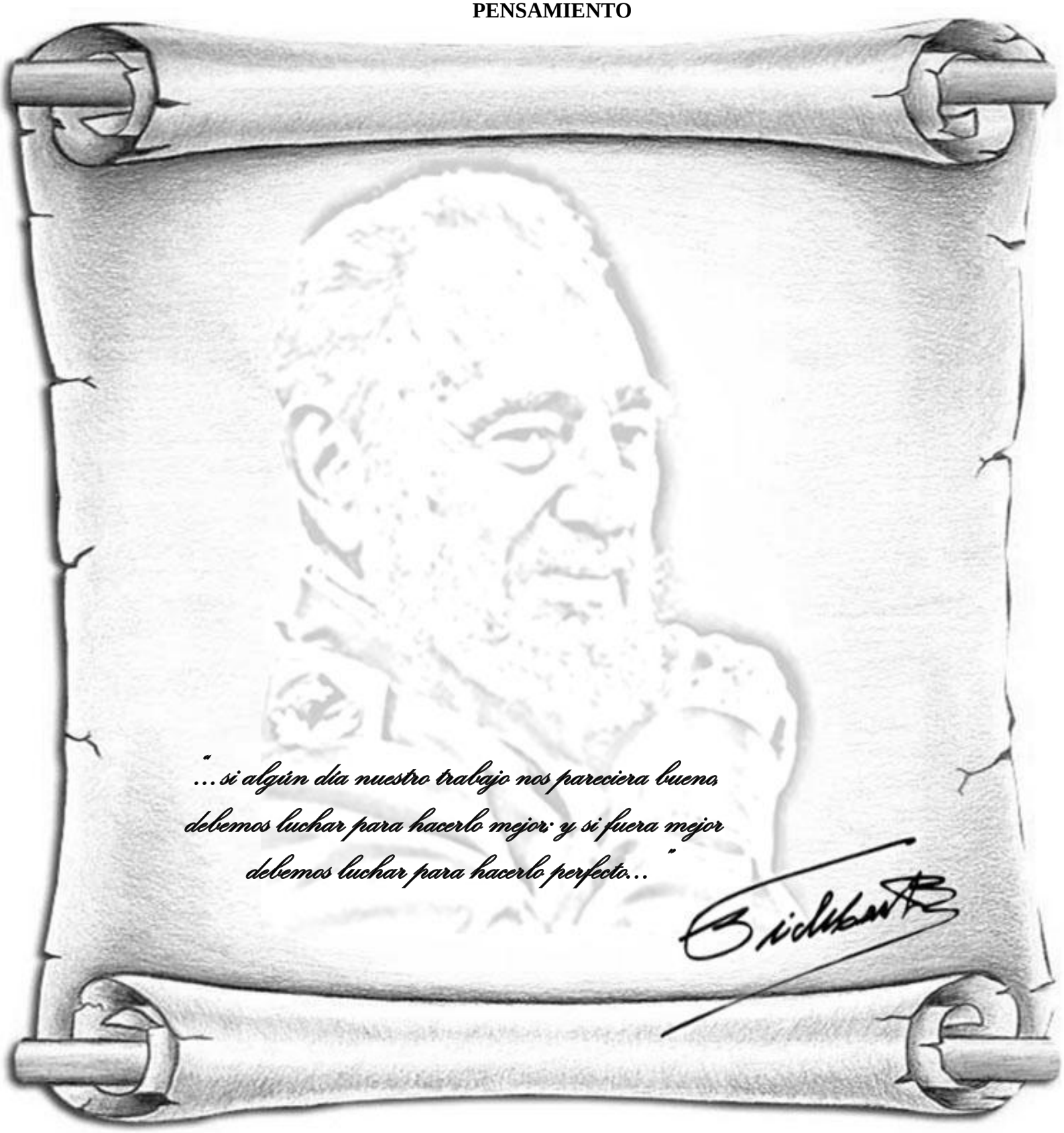
Tutor: MsC. Oiner Gómez Baryolo

Co-tutor: Ing. Mileidy Magalys Sarduy Pérez

Asesor: Ing. Omar Antonio Díaz Peña

Ciudad de La Habana, junio 2011

PENSAMIENTO



DECLARACIÓN DE AUTORÍA

Declaro que soy el único autor de este trabajo y autorizo al Centro de Investigación para la Gestión de Entidades (CEIGE) de la Universidad de las Ciencias Informáticas a hacer uso del mismo en su beneficio.

Para que así conste firmo la presente a los ____ días del mes de _____ del año _____.

_____
Firma del autor
Yesmín Caveda Báez

_____
Firma del tutor
MSc. Oiner Gómez Baryolo

_____
Firma del co-tutor
Ing. Mileidy Magalys Sarduy Pérez

DATOS DE CONTACTO

Nombre y Apellidos: MSc. Oiner Gómez Baryolo
Cargo: Jefe del Departamento de Tecnología del CEIGE
Institución: Universidad de las Ciencias Informáticas
Dirección: Carretera de Carretera a San Antonio Km 2 ½. Torrens. Boyeros. Ciudad de La Habana. Cuba
Teléfono: (53)-7-835-8296
Correo: oiner@uci.cu

Nombre y Apellidos: Ing. Mileidy Magalys Sarduy Pérez
Cargo: Analista Principal del Proyecto de Gestión Integral de Seguridad (ACAXIA)
Institución: Universidad de las Ciencias Informáticas
Dirección: Carretera de Carretera a San Antonio Km 2 ½. Torrens. Boyeros. Ciudad de La Habana. Cuba
Teléfono: (53)-7-835-8296
Correo: mmsarduy@uci.cu

AGRADECIMIENTOS

Los agradecimientos van dirigidos hacia todas aquellas personas que han colaborado en el desarrollo de este trabajo de diploma. Sin mencionar nombres para no herir susceptibilidades se puede decir que a mis tutores, a mis amigos y compañeros, incluyendo especialmente a aquellos que se sentaron a trabajar conmigo codo a codo y esos otros que desde la distancia, hicieron invaluables aportes. A mi familia por el apoyo incondicional, quienes sé, también se gradúan conmigo y a ADOLFO, para quien no tengo palabras de agradecimiento que describan lo que ha hecho por mí.

Agradecimientos Especiales:

Por último y no menos importante un agradecimiento, el más especial, para el Señor de los Cielos, a quien le debo todo.

DEDICATORIA

Sin duda alguna este Trabajo de Diploma está dedicado a mi familia, a mi MAMÁ, a mi COTORRITA, y en especial a mis TÍOS, quienes han sabido ser mis mejores tutores y la inspiración para siempre seguir adelante.

Dedicatoria Especial:

Este apartado especial es dedicado a mi ABUELO, que aunque ya no se encuentra entre nosotros es el alma de la familia y la persona a la que debemos todos encontrarnos en el camino del perfeccionamiento y el ascenso profesional.

RESUMEN

La arquitectura tecnológica es la vista arquitectónica de la infraestructura de software y hardware que apoya la organización y garantiza diferentes escenarios tecnológicos que permiten el correcto funcionamiento del sistema.

La presente investigación surge como una necesidad del Centro de Informatización para la Gestión de Entidades (CEIGE) perteneciente a la Universidad de las Ciencias Informáticas (UCI) de contar con una descripción de los escenarios tecnológicos que debe cubrir la arquitectura de un Sistema de Gestión Empresarial; que permita al Departamento de Tecnología construir componentes y soluciones tecnológicas para el sistema de Planificación de Recursos Empresariales (ERP) que allí se desarrolla dentro de los cronogramas previstos. Evitando que el desarrollo de las tecnologías requeridas para la informatización de los procesos de negocio, y estos procesos, se realice de forma paralela.

De esta manera se desarrolla una propuesta de los escenarios que debe cubrir la Arquitectura Tecnológica de un sistema de gestión empresarial, así como las tecnologías, herramientas y patrones que han de colaborar para darles solución.

PALABRAS CLAVES:

Arquitectura Tecnológica, Escenarios tecnológicos, Planificación de Recursos Empresariales, Sistemas de Gestión Empresarial.

TABLA DE CONTENIDOS

ÍNDICE DE TABLAS

INTRODUCCIÓN

Con la importancia que se le ha atribuido a las Tecnologías de la Información (TI) y su alineación con las estrategias del negocio para mejorar los procesos claves de la empresa, muchas son las compañías que en la actualidad se dedican al desarrollo de software para la gestión empresarial.

En la cúspide evolutiva de estos sistemas se encuentran los ERP, por sus siglas del inglés Enterprise Resource Planning. Lo más destacable de un ERP es que modela y automatiza muchos procesos de negocio integrando la información a través de la compañía y eliminando vínculos complejos y costosos entre los sistemas de cómputo en diferentes áreas.

Esto es posible entre otros factores, gracias a la Arquitectura Tecnológica (AT), vista arquitectónica proveedora de la base para la implementación y mantenimiento del sistema. Esta vista es la contenedora de todas aquellas características, restricciones y requisitos no funcionales que responden a los escenarios tecnológicos requeridos para el correcto funcionamiento y puesta en marcha del sistema. Los mismos constituyen la forma menos ambigua de expresar un uso deseado o anticipado de un software desde la perspectiva tecnológica involucrando un conjunto de atributos de calidad.

A nivel mundial los principales desarrolladores de ERP del mercado como SAP, Oracle o Sage, por citar algunos, han provisto sus productos con potentes plataformas tecnológicas que les han garantizado el éxito entre sus clientes. Estas constituyen los cimientos técnicos de la arquitectura tecnológica del sistema y garantizan un conjunto de escenarios que aseguran la ejecución e implantación de todos los procesos de negocio impulsando el crecimiento de las empresas y la adaptación al cambio. Están sometidas a diferentes estándares internacionales que garantizan la interoperabilidad con otros entornos y plataformas además de dotar al producto de una capacidad de adaptación, flexibilidad, seguridad, integridad y confiabilidad, con bajos costos de mantenimiento y un alto rendimiento en la ejecución de las operaciones.

En Cuba el desarrollo de software es aún muy joven y particularmente el desarrollo de sistemas de gestión empresarial como los ERP. En este sentido uno de los mayores polos productivos es la Universidad de las Ciencias Informáticas. La misma se encuentra enfrascada en el desarrollo de varios proyectos de software entre los que destacan por su importancia e impacto en la economía los llevados a cabo en el Centro de Informatización para la Gestión de Entidades (CEIGE), como es el caso del programa ERP-Cuba.

El producto de software denominado Cedrux se encuentra aún en desarrollo, pero al igual que sus homólogos internacionales ha sido provisto de tecnologías que cubren un conjunto de escenarios propios

de sistemas de gestión, tarea llevada a cabo por el Departamento de Tecnología del centro. Estas tecnologías se integran en un marco de trabajo denominado Sauxe que contiene un conjunto de componentes reutilizables que proveen la estructura genérica y el comportamiento para una familia de abstracciones, garantizando aspectos como la seguridad, la integración, las configuraciones, entre otros. Sin embargo en la medida que avanza el desarrollo del sistema, mayores son los requerimientos que debe cumplir la arquitectura tecnológica, identificándose así, nuevos escenarios no resueltos tecnológicamente que retrasan el trabajo de los desarrolladores; ya que al no contarse con una descripción de los escenarios tecnológicos que debe cubrir la arquitectura tecnológica de un sistema de gestión empresarial, el desarrollo de los procesos de negocio y las tecnologías se realiza de forma paralela, cuando deben ser estas últimas la base para la informatización de estos procesos. Lo que desencadena un retraso en los cronogramas previstos.

Dando lugar al ***problema a resolver***: no existe una identificación proactiva de los escenarios que soportan el desarrollo de sistemas de gestión empresarial.

Por tanto el ***objeto de estudio*** de la investigación es la Arquitectura Tecnológica para sistemas de gestión empresarial.

Objetivo general: Describir escenarios tecnológicos para el desarrollo de sistemas de gestión empresarial.

Como guía de la investigación se trazan los siguientes ***objetivos específicos***:

- Realizar un estudio de la arquitectura de sistemas de gestión empresarial, específicamente desde la arista tecnológica.
- Identificar y describir los escenarios que debe cubrir la arquitectura tecnológica de un sistema de gestión empresarial.
- Especificar patrones y proponer herramientas y tecnologías.
- Validar la propuesta.

Campo de acción: Escenarios tecnológicos para sistemas de gestión empresarial.

Si se identifican y describen los escenarios tecnológicos que soportan el desarrollo de sistemas de gestión empresarial, se lograrán implementar soluciones tecnológicas de forma proactiva que contribuyan al cumplimiento de los cronogramas previstos. Siendo esta la ***idea a defender*** de la presente investigación.

De la cual se espera como ***resultado***: una especificación de escenarios tecnológicos para el desarrollo de sistemas de gestión empresarial.

Para el desarrollo de la investigación se emplearán diferentes ***Métodos Científicos***, tanto teóricos como empíricos que faciliten el progreso y correcto desarrollo de la misma.

Métodos teóricos: Permiten estudiar las características del objeto de investigación que no son observables directamente, facilitan la construcción de modelos e hipótesis de investigación y crean las condiciones para ir más allá de las características fenomenológicas y superficiales de la realidad, contribuyendo al desarrollo de las teorías científicas.

- ✓ *Métodos histórico-lógicos:* En la investigación se analiza la trayectoria de la AS durante los diferentes períodos de la historia, poniendo de manifiesto su desenvolvimiento y desarrollo, así como los principales acontecimientos teóricos que revelan un conocimiento más profundo de su esencia. Estos métodos expresan en forma teórica la esencia del objeto, explican la historia de su desarrollo, reproducen el objeto en su forma superior y permiten unir el estudio de la estructura del objeto de investigación con su concepción histórica.
- ✓ *Método hipotético-deductivo:* La investigación sigue además un método hipotético deductivo porque a partir del problema concreto se plantea objetivos específicos y una idea a defender que en el transcurso de la investigación se resuelven deduciendo y explicando regularidades particulares de menor nivel de generalidad y abstracción a partir de propuestas de mayor generalidad, abstracción y lógica.
- ✓ *Método de la modelación:* Este método está presente de forma implícita en la investigación a través de la propuesta de solución que se modela, donde es el investigador quien elabora el modelo creando abstracciones con el objetivo de explicar la realidad, manteniendo una correspondencia objetiva entre el modelo y el objeto. La necesidad práctica para la cual se ejecuta la modelación y la posible solución del problema de investigación da la medida en que se logra dicha correspondencia, la que es determinada por el investigador escogiendo una alternativa de acuerdo con su criterio.

Métodos empíricos: Los métodos empíricos describen y explican las características fenomenológicas del objeto, representan un nivel de la investigación cuyo contenido procede de la experiencia y es sometido a cierta elaboración racional.

- ✓ *Método Entrevista:* En la investigación se utiliza la entrevista como una técnica particular de recolección de datos. La entrevista es una conversación planificada entre el investigador y el entrevistado para obtener información. Su uso constituye un medio para el conocimiento cualitativo de los fenómenos.
- ✓ *Encuesta:* La encuesta es una técnica muy parecida a la entrevista que en lugar de ser una conversación planificada para obtener información, descarta la forma oral y se basa en la forma escrita, donde queda constancia mediante un documento.

Estructura del documento:

El presente documento cuenta con un Resumen, una Tabla de Contenidos, Introducción, tres Capítulos, seguido de Conclusiones, Recomendaciones, Bibliografía, Glosario de Términos y Anexos.

Capítulo I: "Fundamentación teórica."

Este capítulo recoge la fundamentación teórica del trabajo, que incluye los principales conceptos relacionados con la arquitectura tecnológica del software o vista tecnológica, los escenarios contemplados desde esta perspectiva y los sistemas de gestión empresarial. También se realiza un estudio de las principales soluciones de gestión empresarial del mundo donde se presenta una panorámica de las características de la arquitectura tecnológica de las mismas teniendo en cuenta los escenarios que resuelven y las tecnologías que utilizan.

Capítulo II: "Solución propuesta"

Este capítulo describe detalladamente los escenarios que deben tenerse en cuenta para el desarrollo de la arquitectura tecnológica de un sistema de gestión empresarial, incluyendo aquellos no cubiertos por la del sistema Cedrux y se documentan además posibles soluciones tecnológicas, patrones y herramientas a usar por el equipo de desarrollo tecnológico del mismo.

Capítulo III: "Validación de la propuesta"

Este capítulo refleja el análisis y evaluación de la propuesta realizada en el capítulo anterior, en aras de comprobar si esos escenarios son válidos para el desarrollo de un sistema de gestión empresarial.

CAPÍTULO I: FUNDAMENTACIÓN TEÓRICA

1.1 Introducción

La base tecnológica de un sistema de software juega un papel fundamental en su desarrollo y correcto desempeño, de ella depende el resto de la pirámide organizacional y tecnológica así como la estrategia a seguir por la empresa productora encargada de la informatización de los procesos. Razón por la que deberá estar alineada a una arquitectura que cubra todos los escenarios tecnológicos que facilitarán el desarrollo del sistema y su funcionamiento como un todo integral. En orden de establecer una mejor claridad en el tema, el presente capítulo describe los principales conceptos asociados, así como las características presentes en la arquitectura tecnológica de los principales sistemas de gestión empresarial.

1.2 Arquitectura de Software

La bibliografía existente acerca de la Arquitectura de Software (AS) está compuesta por numerosos escritos que recogen disímiles definiciones del término sin converger en un concepto unánime. No existe todavía una definición universalmente aceptada, pero sí muchas candidatas en las que el término es interpretado y definido de muchas maneras. Aunque resulta interesante la existencia de ideas comunes entre las mismas sin encontrar planteamientos contradictorios, sino complementarios. En la esencia de todas estas opiniones hay un enfoque en razonar sobre los problemas estructurales de un sistema y aquellos atributos o propiedades que no aportan directamente una funcionalidad pero que tienen igual peso en el resultado final. Al respecto Paul C. Clements, Linda M. Northrop (1) y otros autores (2) refieren que estas definiciones no se contraponen unas a otras, ni representan un conflicto en lo que respecta a la definición de la Arquitectura de un sistema de Software, sino que permiten tener una clara visión de lo que el término significa y el énfasis que debe ponerse en la arquitectura- sus partes constituyentes, la entidad como un todo, el comportamiento una vez construida o durante la construcción- , y cómo tomadas juntas, todas estas definiciones forman una vista, un consenso, de lo que es arquitectura de software, permitiendo la visión de un cuadro más completo.

Por esta razón, teniendo en cuenta los planteamientos más relevantes de las mismas se puede decir que la AS constituye a grandes rasgos, una vista del sistema que describe un alto nivel de configuración de los componentes que lo conforman y las conexiones que coordinan las actividades y la interacción de esos componentes según se la percibe desde el resto del sistema. Y no solo es afectada por la estructura y el comportamiento, sino también por el uso, la funcionalidad, las restricciones y

compromisos económicos y tecnológicos que en conjunto permiten alcanzar la misión del sistema. ((3); (4); (5); (1); (6); (7); (8))

Independientemente de su conceptualización, la arquitectura de software ha pasado a ser un artefacto esencial dentro del proceso de desarrollo de software moderno debido a sus múltiples usos: documentación de decisiones tempranas de diseño, guía para la implementación y mantenimiento del sistema y base para la comunicación con los distintos involucrados, sin embargo, llegar a construir una arquitectura para el software que sea apropiada para dar cabida a lo anteriormente expuesto es una tarea que dista muchísimo de ser sencilla, por ello es necesario sea contemplada desde diferentes puntos de vista.

1.3 Vistas Arquitectónicas

Muchas personas y organizaciones están interesadas en la construcción de sistemas de software. Estos son llamados involucrados (*stakeholders*): el cliente, los usuarios finales, los diseñadores, el probador, el gerente del proyecto, los de mantenimiento, e incluso ésos que comercializan el sistema (9) y cada uno de ellos tiene diferentes preocupaciones respecto a las características o requisitos que debe cumplir o mejorar el sistema y que afectan y son afectadas por la arquitectura. Esta última debe ser capaz de establecer la comunicación entre dichos involucrados proporcionando un lenguaje común en el que las preocupaciones puedan ser expresadas, negociadas y resueltas a un nivel intelectualmente manejable aún en caso de sistemas complejos. (1) Sin tal lenguaje, es difícil entender los sistemas grandes y complejos lo suficiente como para tomar las decisiones tempranas que influyen en la calidad, la utilidad (9) y otros requisitos que deben ser garantizados.

¿Cómo lograrlo?

Como un sistema de software es difícil de abarcar visualmente porque no existe en un mundo de tres dimensiones, (6) es conveniente considerar más de una perspectiva arquitectural así como las relaciones entre ellas. Propiedades adicionales a la funcionalidad como distribución física, comunicación entre procesos y sincronización deben ser razonadas en el nivel arquitectónico. Estas propiedades son abordadas en múltiples estructuras a las que se les denomina vistas arquitecturales o modelos arquitectónicos.

Cada vista refleja un conjunto específico de preocupaciones que son de interés para un grupo dado de involucrados en el sistema. Estas son por consiguiente, las abstracciones, cada una con respecto a un

criterio diferente. Cada vista puede ser considerada un diagrama del sistema de software y cada una puede usar su propia anotación, puede reflejar su propia opción de estilo arquitectónico, y puede definir los componentes, así como sus relaciones mutuas, principios y pautas. Sin embargo, las vistas no son totalmente independientes. Los elementos en una se pueden relacionar a los elementos en otra, entonces mientras que es útil considerar cada una separadamente, también es necesario razonar rigurosamente sobre las interrelaciones de estas vistas. Ninguna de estas vistas de forma individual describe adecuadamente la AS del sistema, esto solo es logrado con el conjunto de todas las vistas y la información que las trasciende. (8)

¿Qué vistas son relevantes?

A esta interrogante Bachmann y otros (8) responden que está en dependencia de la metas del sistema. El documentar las diferentes vistas de la AS puede servir a muchos propósitos: una declaración de la misión para los implementadores, una base para el análisis, la especificación para la generación automática de código, el punto de partida para el entendimiento del sistema y la recuperación de recursos o el diagrama para la planeación del proyecto. Estas vistas también exponen diferentes atributos de calidad a diferentes grados. Por consiguiente, los atributos de calidad más importantes para los involucrados del desarrollo del sistema afectarán la decisión de qué vista documentar.

Muchos son los autores ((8), (4), (3), (10) por citar algunos) que hacen referencia a las vistas en sus escritos de AS y algunos han establecido como estándar aquellas que a su criterio son indispensables para un sistema. En su libro Documenting Software Arquitecture, Bachmann y sus compañeros muestran algunas de las propuestas:

En 1995 Kruchten de la Rational Software Co. escribió un influente documento describiendo cuatro vistas fundamentales de la AS: la Vista Lógica, la de Procesos, la de Desarrollo, la Física y una distinguida quinta vista, la Vista de Escenarios/Casos de Uso que mantiene atadas las otras cuatro conformando lo que se conoce como las “4+1” vistas de la arquitectura.

En este mismo año Soni, Nord y Hofmeister de la Corporación Siemens hacen una observación similar desde el punto de vista del uso en la práctica industrial describiendo la Vista Conceptual, la de Interconexión Modular, la de Ejecución y la Vista de Código.

Diferentes vistas soportan disímiles metas y usos en dependencia del sistema que describen. Con el avance tecnológico y la revolución de estos aspectos ha emergido un nuevo conjunto de vistas entre las

que se encuentra la Vista Tecnológica. El presente trabajo se centrará en la vista tecnológica de la arquitectura.

1.3.1 Arquitectura Tecnológica

Bachmann informa de otro conjunto de vistas emergentes entre las que se encuentran las prescritas por Herzum y Sims en 1999 en su libro Business Component Factory. Donde distinguen las siguientes cuatro como las más importantes: la Arquitectura Tecnológica, Arquitectura de la Aplicación, la Arquitectura de Gestión de Proyecto y la Arquitectura Funcional.

La Arquitectura Tecnológica es descrita como la concerniente al entorno de ejecución de los componentes, el conjunto de herramientas, el marco de trabajo (*framework*) de la interfaz de usuario, y cualquier otro servicio/facilidad técnica requerida para desarrollar o ejecutar un sistema.

El Marco de Referencia Arquitectónico Open Group (TOGAF, The Open Group Architectural Framework) también hace alusión a la arquitectura tecnológica. TOGAF reconoce cuatro arquitecturas entre las que se encuentran: Arquitectura de Negocios, Arquitectura de Datos/Información, Arquitectura de Aplicación y Arquitectura Tecnológica. (11)

En este marco la Arquitectura Tecnológica describe el software lógico y las capacidades de *hardware* que se requieren para soportar el despliegue del negocio, los datos y los servicios de la aplicación. Lo que incluye infraestructura de tecnologías de la información, *middleware*, redes, comunicaciones, procesamiento, normas, etc. (12)

La Estrategia de Arquitectura de Microsoft (MSF, Microsoft Solution Framework) define cuatro vistas coincidentes con las descritas por TOGAF, ocasionalmente también llamadas arquitecturas, en las que incluyen: Arquitectura de Negocios, Arquitectura de Aplicación, Arquitectura de Información y Arquitectura de Tecnología.

Donde una arquitectura tecnológica es definida como la arquitectura de la infraestructura de *hardware* y *software* que apoya la organización e implementa los requisitos operacionales (o no funcionales), particularmente las arquitecturas de aplicación e información de la organización. Describe la estructura y la interrelación de las tecnologías usadas, y cómo esas tecnologías soportan los requisitos operacionales de la organización. (13) En este caso abordan fundamentalmente dos arquitecturas, en las que centran su completa atención por el gran peso que tienen para el correcto funcionamiento del sistema: la arquitectura

de aplicación como aquella que soporta y garantiza los requisitos funcionales y la arquitectura tecnológica como aquella que soporta e implementa los requisitos no funcionales.

Puede decirse entonces que la arquitectura tecnológica es la encargada de sentar la base tecnológica que dará soporte al sistema. La misma describe la infraestructura del sistema y el entorno de desarrollo, precisando tecnologías y herramientas a utilizar y la forma en que estas se combinan para implementar los requisitos no funcionales. Incluye protocolos, estándares, normas, restricciones tanto de software como de hardware y cuestiones legales y de licencia. Por lo que puede afirmarse que una buena arquitectura tecnológica puede proporcionar seguridad, disponibilidad, fiabilidad, y apoyar una variedad de otros requisitos operacionales que responden a los escenarios tecnológicos de la aplicación. Pero si esta no se diseña para tomar ventajas de las características que brinda la arquitectura tecnológica, puede desempeñarse pobremente o puede ser difícil de desplegar y operar.

1.4 Propiedades Extra-funcionales

La arquitectura es el resultado de un conjunto de decisiones técnicas y de negocio. Estas consideraciones de negocio determinan requerimientos que deben acomodarse en la arquitectura de un sistema y ser garantizados por las tecnologías de la información. Dichos requisitos se encuentran encima y debajo de la funcionalidad que es la declaración básica de las capacidades del sistema, servicios y comportamiento. Aunque la funcionalidad y estas otras propiedades están estrechamente relacionadas, la funcionalidad toma a menudo no sólo el asiento delantero en el esquema de desarrollo sino el único asiento. Sin embrago, esto es algo inconsecuente. Muchos sistemas son frecuentemente rediseñados no porque son funcionalmente deficientes – ya que los reemplazos a veces son funcionalmente idénticos- sino porque son difíciles de mantener, portar, o escalar, o son demasiado lentos, o se han visto comprometidos por los *hackers*. (9) De ahí la importancia de que estos sean incluidos y garantizados en la arquitectura.

La arquitectura tecnológica como se ha mencionado antes, es la encargada de garantizar estas propiedades no funcionales en pos de que el sistema pueda ejecutar la funcionalidad deseada. Esta debe ocuparse de cómo conseguir los requisitos para el rendimiento, capacidad, fiabilidad, seguridad, capacidad de adaptación y otras características con las que el sistema debe contar. (7)

1.4.1 Requisitos

El requisito es la forma esencial que se utiliza para describir un software desde varios puntos de vista, expresan las preocupaciones de los diferentes involucrados del sistema y son recogidos en las vistas de la

arquitectura. En el Manual de Requerimientos de la Ingeniería (14) se define un requisito como un atributo *necesario* en el sistema, una declaración que identifica una capacidad, característica o factor de calidad de un sistema con el fin de que este tenga valor y utilidad para un cliente o usuario; mediante las cuales se pretende cumplir con determinadas necesidades o restricciones operativas, y que contribuyen a solucionar un problema en un entorno real. Otra definición aceptada es aquella emitida por Sommerville donde plantea que los requisitos para un sistema son la descripción de los servicios proporcionados por el mismo y sus restricciones operativas.

1.4.1.1 Categorías de requisitos

Categorías propuestas por Sommerville: (15)

- **Requisitos Funcionales (RF):** Son declaraciones de los servicios que debe proporcionar el sistema, de la manera que este debe reaccionar a entradas particulares y de cómo se debe comportar en situaciones particulares. En algunos casos, los requerimientos funcionales de los sistemas también pueden declarar explícitamente lo que el sistema no debe hacer.
- **Requisitos No Funcionales (RNF):** Son restricciones de los servicios o funciones ofrecidos por el sistema. Incluyen restricciones de tiempo, sobre proceso de desarrollo y estándares. Los requisitos no funcionales a menudo se aplican al sistema en su totalidad. Normalmente apenas se aplican a características o servicios individuales del sistema.
- **Requisitos del Dominio (RD):** Son requerimientos que provienen del dominio de aplicación del sistema y que reflejan las características y restricciones de ese dominio pueden ser funcionales o no funcionales.

1.4.1.2 Restricciones

Este concepto resulta complejo de entender y guarda en su simple definición el mayor problema de su aplicación pragmática. Como factor común los especialistas asumen la definición de restricción como la restricción en sí, sin lograr un pragmatismo de su definición y gestión. Tan importante resulta el término para la Ingeniería del Software y el proceso de conceptualización del producto, que una mala aplicación del mismo puede afectar un entendimiento contractual o generar un costoso error de implementación. Puede entenderse por restricción como la característica limitante de una solución de software y está relacionada con incapacidades tecnológicas (lenguaje, paradigma, tecnología base, etc.), de infraestructura (redes, velocidad, protocolos, servidores, servicios, plataformas) y de diseño las que se

relacionan con características de abstracción que debe cumplir o no una solución (Multilenguaje, multimoneda, multientidad, estándares etc.). (16)

1.4.1.3 Importancia de los Requisitos No Funcionales

A efectos de esta investigación el mayor peso recae sobre los requisitos no funcionales que especifican o restringen las propiedades emergentes del sistema. Por lo tanto pueden especificar el rendimiento, la protección, la disponibilidad y otras propiedades. Lo que significa que a menudo son más críticos que los requisitos funcionales particulares, pues los usuarios normalmente pueden encontrar formas de trabajar alrededor de una función del sistema que realmente no cumple sus necesidades. Sin embargo, el incumplimiento de un requisito no funcional puede significar que el sistema entero sea inutilizable. (15)

Estos requisitos no solo se refieren al sistema de software a desarrollar, algunos de estos pueden restringir el proceso que se debe utilizar para desarrollar el sistema. Surgen de las necesidades del usuario concernientes a restricciones de presupuesto, políticas de la organización, necesidad de interoperar con otros sistemas de *hardware* o *software* u otros factores externos como regulaciones de seguridad o legislaciones sobre privacidad. (15)

1.4.1.4 Categorías de Requisitos No Funcionales

Algunos autores como Boch y Molin (17) han subcategorizado estos requisitos en ***RNF de desarrollo*** y ***RNF operacionales*** que coinciden con la clasificación que dan Clements y Northrop de atributos de calidad: ***observables vía ejecución*** y ***no observables vía ejecución*** respectivamente.

Los primeros autores plantean que los RNF de desarrollo son cualidades del sistema que son relevantes desde la perspectiva de la Ingeniería de software, por ejemplo: la mantenibilidad (*maintainability*), la reusabilidad (*reusability*) y la flexibilidad (*flexibility*). Los RNF operacionales son las cualidades del sistema en el funcionamiento, por ejemplo el rendimiento (*performance*), la fiabilidad (*reliability*), la robustez (*robustness*) y tolerancia a fallos (*fault-tolerance*).

Los segundos autores describen las primeras clasificaciones como las referentes a aquellos atributos que pueden ser medidos ejecutando el software y observando sus efectos, rendimiento, seguridad, fiabilidad y funcionalidad, mientras que las segundas incluyen aquellos que no pueden ser medidos observando el sistema, pero si observando las actividades de desarrollo o mantenimiento. Esta categoría incluye el mantenimiento en todos sus sentidos: adaptabilidad, portabilidad, reutilización, y otros.

1.4.2 Atributos de Calidad

Los atributos de calidad son aspectos del sistema, que en general, no afectan directamente a la funcionalidad necesitada, sino que definen la calidad y las características que el sistema debe soportar.

A grandes rasgos, se establece una clasificación de los atributos de calidad en dos categorías: ((1); (2))

- **Observables en vía de ejecución:** Son aquellos atributos que se determinan del comportamiento del sistema en tiempo de ejecución.
- **No observables en vía de ejecución:** Son aquellos atributos que se establecen durante el desarrollo del sistema o las actividades de mantenimiento.

1.4.3 Escenarios

Kazman, Abowd, Bass y Clementns plantean que los escenarios son herramientas importantes en una arquitectura para ganar información sobre la aptitud de un sistema con respecto a un conjunto de atributos de calidad deseados. Han sido ampliamente usados y documentados como una técnica durante la captura de requisitos, especialmente los operacionales. En el caso particular de estos autores, los escenarios son usados para expresar las instancias particulares de cada atributo de calidad importante para el cliente del sistema, analizando la arquitectura bajo la consideración de cuan bien o cuan fácil esta satisface las restricciones impuestas por cada escenario. Y los definen como una breve descripción del uso deseado o anticipado de un sistema y suelen llamarlos viñetas o guiones.

¿Por qué escenarios?

Desde la perspectiva de la arquitectura hay tres problemas fundamentales que se presentan para un arquitecto respecto a los atributos de calidad: (9)

- Las definiciones provistas de atributos de calidad no son operacionales. No tiene sentido decir que un sistema será modificable. Cada sistema es el modificable con respecto a un conjunto de cambios y no lo es con respecto a otros. Con otros atributos sucede de forma similar.
- Un enfoque de discusión es a menudo, la calidad a que un aspecto particular pertenece. ¿Es una falla del sistema un aspecto de disponibilidad, un aspecto de seguridad, o un aspecto de utilidad? Las tres comunidades del atributo reclamarían la pertenencia de la falla del sistema.
- Cada comunidad de atributo ha desarrollado su propio vocabulario. La comunidad del rendimiento tiene "eventos" que llegan a un sistema, la comunidad de seguridad tiene "ataques" que llegan a un sistema, la comunidad de disponibilidad tiene "fallos" de un sistema, y la comunidad de utilidad

tiene la "entrada del usuario." Todos éstos realmente pueden referirse a la misma ocurrencia, pero se describen usando términos diferentes.

Como una solución a los dos primeros problemas (las definiciones no operacionales y el solapamiento de las incumbencias de los atributos) surgen los escenarios, brindando un medio de caracterizar los atributos de calidad del sistema. Y la solución al tercer problema es proporcionar una discusión breve de cada atributo que se concentra en sus incumbencias subyacentes para ilustrar los conceptos que son fundamentales a la comunidad del atributo. (9)

1.4.3.1 Tipos de escenarios

Los escenarios difieren ampliamente en tamaño y alcance. Kazman, Abowd, Bass y Clements en su reporte técnico “Scenario-Based Analysis of Architecture" hacen una distinción entre escenarios, definiendo dos tipos:

- **Escenarios directos:** Constituyen aquellos escenarios que son soportados directamente por el sistema.
- **Escenarios indirectos:** Constituyen aquellos escenarios que no son directamente soportados por el sistema, significando esto que debe haber un cambio en el sistema que se pueda representar arquitectónicamente. Este cambio podría ser un cambio a la forma en que uno o más componentes realizan una actividad asignada, la adición de un componente para realizar alguna actividad, la adición de una conexión entre los componentes existentes, o una combinación de lo anterior.

1.4.3.2 Partes de un escenario

Bass, Clements y Kazman en su libro “Software Architecture in Practice” describen seis partes integrantes de un escenario (ver Figura 1):

- *Fuente del estímulo:* Ésta es alguna entidad (un humano, un sistema de cómputo, o cualquier otro mediador) que genera el estímulo.
- *Estímulo:* El estímulo es una condición que necesita ser considerada cuando llega a un sistema.
- *Ambiente:* El estímulo ocurre dentro de condiciones certeras. El sistema puede estar en una condición de sobrecarga excesiva o puede estar ejecutándose cuando el estímulo ocurre, o alguna otra condición que puede ser verdad. El ambiente describe esas condiciones en las cuales se encuentra el sistema en el momento que se recibe el estímulo.

- *Artefacto:* Algún artefacto es estimulado. Este puede ser el sistema entero o algunas partes de él. Esta es la parte del escenario que describe el artefacto que es afectado con la ocurrencia de un estímulo.
- *Respuesta:* La respuesta es la actividad emprendida después de la llegada del estímulo. Describe, a través de la arquitectura, cómo debería responder el sistema ante el estímulo. Permite establecer cuál es el atributo de calidad asociado.
- *Medida de la respuesta:* Cuando la respuesta ocurre, debe ser medible en alguna manera para que el requisito pueda probarse.

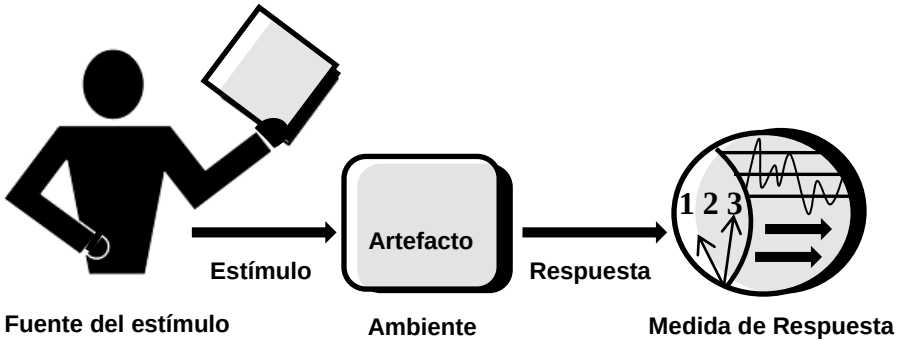


Figura 1. **Partes de un escenario. Imagen basada en** (9)

Escenario tecnológico

En el contexto del documento un escenario es tecnológico cuando se contempla dentro de la arquitectura tecnológica. Por tanto un escenario tecnológico es definido como la forma menos ambigua de expresar un uso deseado o anticipado de un software involucrando aquellas características y requisitos no funcionales concernientes a la arquitectura tecnológica.

1.4.4 Familias de Sistemas Relacionados

El diseño arquitectónico deberá dibujarse sobre patrones repetibles que se basen comúnmente en el diseño de familias de sistemas similares. En esencia, el diseño deberá tener la habilidad de volver a utilizar los bloques de construcción arquitectónicos.

A lo largo del proceso de diseño y desarrollo de la arquitectura, los requisitos no funcionales juegan un papel importante, pues en base a estos se generan importantes decisiones de diseño. Dado que la arquitectura de software inhibe o facilita estos requisitos, resulta de particular interés analizar la influencia

de ciertos elementos de diseño utilizados para su definición, determinando sus características. Estos elementos de diseño son los estilos arquitectónicos, los patrones arquitectónicos y los patrones de diseño. Los estilos sólo se manifiestan en arquitectura teórica descriptiva de alto nivel de abstracción; los patrones, por todas partes. Los estilos se encuentran en el centro de la arquitectura y constituyen buena parte de su sustancia. Los patrones de arquitectura están claramente dentro de la disciplina arquitectónica, solapándose con los estilos, mientras que los patrones de diseño se encuentran más bien en la periferia. (11)

1.4.4.1 Estilos Arquitectónicos

Shaw y Garlan definen estilo arquitectónico como una familia de sistemas de software en términos de un patrón de organización estructural que determina el vocabulario de los componentes y tipos de conectores que pueden ser usados en el caso particular de ese estilo, junto con una serie de restricciones de cómo pueden ser combinados. Para muchos estilos puede existir uno o más modelos semánticos que especifiquen cómo determinar las propiedades generales del sistema partiendo de las propiedades de sus partes. (11) Según Buschmann los estilos expresan componentes y las relaciones entre éstos, con las restricciones de su aplicación y la composición asociada, así como también las reglas para su construcción. Así mismo, se considera como un tipo particular de estructura fundamental para un sistema de software, conjuntamente con un método asociado que especifica cómo construirlo.

Si se analiza la bibliografía de AS puede observarse que varios autores ((4), (1), (9), (18), (6), entre otros) han propuesto sus definiciones de estilo e incluso los han identificado y agrupado de forma diferente. Como parte de la estrategia de la arquitectura de Microsoft, Reynoso y Kicillof hacen un estudio exhaustivo denominado “Estilos y Patrones en la Estrategia de Arquitectura de Microsoft” donde exponen varias propuestas de diferentes autores conformando una lista bastante completa que incluye 19 estilos, constituyendo este, el principal referente usado.

1.4.4.2 Patrones

Buchmann y sus compañeros (18) describen de forma general un patrón para la arquitectura de software como un problema de diseño particular y recurrente que aparece en contextos específicos de diseño, y presenta un esquema genérico bien probado para su solución. El esquema de la solución es especificado describiendo sus componentes constitutivos, sus responsabilidades y relaciones, y las maneras en que estos colaboran.

Según estos autores esto permite adoptar un esquema de tres partes que describe a cualquier patrón:

- Contexto: Describe situaciones de diseño en la que aparece un problema.
- Problema: Esta parte del esquema de descripción de un patrón describe el problema que se presenta repetidamente en el contexto dado. Comienza con una especificación general del problema, captando su esencia y se completa con un conjunto de fuerzas que denotan cualquier aspecto del problema que debe ser considerado al resolverlo, como:
  - Los requisitos que la solución debe completar.
  - Las restricciones que deben considerar.
  - Las propiedades deseables que la solución debe tener.
- Solución: La parte de la solución de un patrón muestra cómo resolver el problema recurrente, o bien, cómo equilibrar las fuerzas asociadas a él. En la arquitectura del software tal solución incluye dos aspectos:
  - La estructura o una configuración espacial de elementos que engloba los aspectos estáticos. Desde que tal estructura puede verse como una micro-arquitectura, consiste, como cualquier AS en componentes y su las relaciones.
  - Comportamiento en tiempo de ejecución, engloba los aspectos dinámicos de la solución, cómo los participantes del patrón colaboran, cómo se organiza el trabajo entre ellos y cómo se comunican entre sí.

Patrones arquitectónicos

Los patrones de arquitectura representan los patrones de alto nivel de abstracción y ayudan a fundamentar la estructura de la solución mediante la aplicación de las mejores prácticas. Los mismos se centran más en las responsabilidades, características macro funcionales e interfaces contractuales que en las especificaciones del diseño, la relación entre clases y los procedimientos internos. (16) El principal referente teórico utilizado para los patrones de arquitectura lo constituye el libro “Patterns Oriented Software Architecture” POSA, obra que santifica esta rama de la disciplina.

Patrones de Diseño

Las características del diseño arquitectónico de menor nivel de abstracción es objeto de estudio de los patrones de diseño, lo que se enfocan más en la composición de las estructuras de datos, las clases funcionales y colaboración entre estas, están más concentrados al diseño de bajo nivel de abstracción y

facilita la reutilización de contexto entre clases o grupos de clases, el principal representante de este género arquitectónico es el libro conocido como “Gain of four” que lleva por título “Design Patterns”.

1.5 Sistemas de Gestión Empresarial

La capacidad de las empresas de adaptarse a cambios en el clima de los negocios depende significativamente de su capacidad de reconfigurar sus procesos de producción para que éstos respondan a las nuevas condiciones en el mercado. No obstante, alinear estrategias, procesos de negocio y tecnologías puede ser una tarea difícil si no se cuenta con una herramienta que centralice toda la información. Los sistemas de Planificación de Recursos Empresariales, ERP, proveen esa herramienta necesaria para potenciar la integración y gestionar todos los procesos generados en la empresa. Según los Laudon (19), los sistemas ERP son sistemas de información que congregan los procesos vitales de la organización de forma que la información fluya a través de ésta, mejorando la planeación, la ejecución, la coordinación y el proceso de toma de decisiones.

1.5.1 Características de los Sistemas ERP

Los sistemas ERP se diferencian de otros sistemas de información porque pretenden integrar todas las necesidades de información dentro de la organización, por su naturaleza estándar de solución sobre un único y potente repositorio de datos, una arquitectura funcional de los módulos y porque pueden ser implementados de acuerdo con las particularidades de cada organización. (20)

Integrales: Integrales, porque permiten controlar los diferentes procesos de la compañía entendiendo que todos los departamentos de una empresa se relacionan entre sí, es decir, que el resultado de un proceso es punto de inicio del siguiente. Por ejemplo, en una compañía, el que un cliente haga un pedido representa que se cree una orden de venta que desencadena el proceso de producción, de control de inventarios, de planeación de distribución del producto, cobranza, y por supuesto sus respectivos movimientos contables. Si la empresa no usa un ERP, necesitará tener varios programas que controlen todos los procesos mencionados, con la desventaja de que al no estar integrados, la información se duplica, crece el margen de contaminación en la información (sobre todo por errores de captura) y se crea un escenario favorable para malversaciones. Con un ERP, el operador simplemente captura el pedido y el sistema se encarga de todo lo demás, por lo que la información no se manipula y se encuentra protegida. (21)

Modulares: Los ERP entienden que una empresa es un conjunto de departamentos que se encuentran interrelacionados por la información que comparten y que se genera a partir de sus procesos. Una ventaja de los ERP, tanto económica como técnicamente es que la funcionalidad se encuentra dividida en módulos, los cuales pueden instalarse de acuerdo con los requerimientos del cliente. Ejemplo: Ventas, Materiales, Finanzas, Control de Almacén, etc. (21)

Base de Datos central: Permite almacenar en una base de datos central toda la información que proviene de los distintos módulos o aplicaciones y a su vez entregar desde sus repositorios la información que éstos necesitan.

Adaptables: Los ERP están creados para adaptarse a la idiosincrasia de cada empresa. Esto se logra por medio de la configuración o parametrización de los procesos de acuerdo con las salidas que se necesiten de cada uno. Por ejemplo, para controlar inventarios, es posible que una empresa necesite manejar la partición de lotes pero otra empresa no. (21)

Desde el punto de vista tecnológico, la arquitectura de un ERP deberá ser abierta, flexible, fácilmente escalable e integrable con el resto de aplicaciones empresariales. El reto al que se enfrentan las empresas actualmente, es el de diseñar un sistema de información perfectamente integrado y consecuente con la estrategia de negocio. (22)

1.6 Características de la Arquitectura Tecnológica de diferentes sistemas ERP

La tecnología de la información representa un papel fundamental en los sistemas de gestión empresarial cuando se habla de competencia y crecimiento, control de costos, diferenciación y mayor porción de mercado, sin la pérdida de la flexibilidad que los clientes exigen, y sin la pérdida de un margen de ganancia. El mercado ofrece a las empresas diferentes opciones que varían no sólo en precio sino en integración, plataformas soportadas y funcionalidad. A continuación se mencionan las características de la arquitectura tecnológica de algunos de estos sistemas:

1.6.1 SAP R/3

SAP trabaja en el sector de software de planificación de recursos empresariales, siendo R/3 el principal producto de la compañía, en el que la R significa procesamiento en tiempo real y el número 3 se refiere a las tres capas de la arquitectura de proceso: bases de datos, servidor de aplicaciones y cliente. El predecesor de R/3 fue R/2. SAP cuenta con una plataforma de tecnología abierta, SAP NetWeaver, que proporciona la infraestructura que da soporte al sistema. SAP NetWeaver se ha convertido en la columna

vertebral del conjunto de soluciones de SAP. Además de contar con prestaciones de servidor de aplicaciones, portales, Business Intelligence (BI), movilidad o interacción, dispone de dos grandes novedades: una es BMP o gestión de procesos de negocio, que viene a ser lo que se conocía como *workflow* y con el que se definen los flujos de trabajo en función de tareas y personal. En segundo lugar, puede encontrarse la piedra angular de este planteamiento, esto es, el repositorio central de datos maestros (CMDM). Se trata de un único repositorio en el que quedan almacenados todos los datos maestros de los diferentes módulos (ERP, CRM, etc.), con lo que se simplifica la elaboración de los flujos de trabajo al evitar el análisis de estos datos módulo por módulo. (23) Esta plataforma tecnológica convierte a SAP en un programa *Web-enable*, lo que significa que estaría totalmente preparado para trabajar con él mediante la web. Las múltiples ventajas que brinda la arquitectura tecnológica del software SAP R/3 hacen que se haya convertido en uno de los estándares dentro de las grandes corporaciones:

Integrado: Tal cantidad de módulos no aportarían demasiado valor añadido a la empresa si no fuera por la integración. Las interrelaciones estrechas entre módulos de SAP permiten tener disponible en tiempo real y con exactitud los principales indicadores de gestión. Como ejemplo ilustrativo puede decirse que una entrada de mercancías en R/3 puede producir una actualización del inventario de almacén, un apunte contable en la contabilidad financiera, una actualización del sistema de información del control de costes y un aviso a producción de que hay nueva materia prima en almacén. (24)

Exhaustivo: El sistema R/3 engloba la prácticamente la totalidad de los procesos de gestión de la empresa. (24)

Abierto: Tecnológicamente hablando, SAP es un sistema abierto. Puede ser implantado en una variedad enorme de servidores diferentes y ejecutarse sobre sistemas operativos y sistemas de gestión de bases de datos de diversos fabricantes. Esto permite escalar el sistema adecuándolo al tamaño de empresa del cliente y elegir a los proveedores de hardware y software de sistemas sin estar atados a ninguno.

Flexible: Pueden utilizarse junto con SAP R/3 otros productos de software de otros fabricantes, existen interfaces con productos de Microsoft, Lotus u Oracle entre otros. SAP posee también un amplio menú de parametrización que permite adecuar el sistema a las necesidades de los clientes, así como un completo sistema de desarrollo para crear nuevos programas y que mantengan la integración con el estándar. (24)

Global: El sistema R/3 soporta su utilización en varios idiomas, la contabilización de documentos en cualquier moneda y tiene recogidas las particularidades fiscales y de gestión de recursos humanos de un gran número de países. Esta globalidad es el argumento de mayor peso en la decisión de una

multinacional a la hora de adquirir SAP. Actualizado dos de los grandes problemas de los departamentos de TI a finales de los 90 han sido el efecto 2000 y la entrada en vigor del euro. El software SAP R/3 tiene contemplados y solucionados estos problemas. Además, la constante investigación llevada a cabo por SAP hace que su software esté al día incluyendo las últimas tecnologías disponibles. (24)

Aunque la cantidad de aplicaciones desarrolladas por SAP es enorme, siempre existe la posibilidad de que el cliente que compre R/3 tenga alguna necesidad tan específica de su negocio que no esté contemplada en el estándar. También puede darse el caso de que la funcionalidad que ofrece el estándar no se ajuste completamente a las necesidades del cliente. Para resolver estas situaciones existe un entorno completo de desarrollo de nuevas aplicaciones integradas en R/3. Este entorno, que SAP denomina ABAP/4 Development Workbench, se compone de una serie de herramientas integradas que permiten crear desarrollos nuevos en poco tiempo. (24)

El sistema permite realizar trabajos de fondo con el fin de ejecutar tareas periódicas sin la intervención humana, como son los trabajos de impresión. Permite la generación de diferentes tipos de gráficos, Manejo de reportes o informes donde incluye miles de reportes programados, manejo de transacciones, gestión y control de versiones, gestión de perfiles de usuario, administración, monitoreo, mantenimiento y operación con bases de datos, acceso rápido a la información relevante.(25)

Para la valoración y análisis de este sistema se tuvo en cuenta estas y otras características descritas en detalle por José Antonio Hernández en su libro “SAP R/3 HandBook”. (25)

1.6.2 Oracle

La oferta de soluciones ERP de Oracle es muy variada. Este cuenta con diferentes productos mejorados a través de sus adquisiciones (PeopleSoft y JD Edwards). Oracle une las mejores aplicaciones empresariales del mundo para ofrecer la mejor tecnología de infraestructura de software. Oracle Technology Foundation provee a JD Edwards EnterpriseOne todos los componentes necesarios para ejecutar sus aplicaciones. Este es un paquete completo de productos de software de estándares abiertos que permite integrar y mantener fácilmente las aplicaciones de JD Edwards, está integrado por Oracle Database 11g y Oracle Fusion Middleware. Oracle Database 11g brinda desempeño y escalabilidad récord en servidores UNIX, Linux y Windows. Brinda un ROI rápido al permitir a los usuarios pasar de un solo servidor a *grid computing* sin tener que cambiar ni una sola línea de código. Su funcionalidad de Clusters en Tiempo Real permite la máxima disponibilidad de datos. Oracle Fusion Middleware da soporte

al ciclo de vida completo para aplicaciones orientadas a servicios y ofrece una alta interoperabilidad con la infraestructura IT existente en una empresa gracias a su arquitectura modular única. Contiene componentes como Oracle Application Server Integration B2B, Oracle Application Server Java Edition, Oracle Application Server Portal, Oracle Application Server Web Cache, Oracle BPEL Process Manager, Oracle Business Activity Monitoring, Oracle Business Intelligence Discoverer, Oracle Business Intelligence Publisher. Estas tecnologías hacen de JD Edwards un sistema flexible y escalable que soporta diferentes sistemas operativos y bases de datos. Brindan a sus socios y clientes acceder un solo punto de entrada para acceder a la documentación disponible del producto y los recursos de la solución a través de su Portal de Información para clientes y socios. Dotan al ERP de globalidad, o sea, la habilidad de operar en diferentes países con diferentes idiomas. Además permiten la integración con otros productos Oracle. Estas tecnologías soportan una amplia variedad de procesos de negocio con una base de datos común. ((26); (27); (28))

Las aplicaciones PeopleSoft Enterprise de Oracle están diseñadas para satisfacer los requisitos empresariales más complejos. La plataforma tecnológica de PeopleSoft, PeopleSoft Enterprise PeopleTools dota a este sistema de una fácil y rápida navegación de datos jerárquicos a través de menús contextuales. Habilita páginas con ayuda precisa. Permite la visualización de contenido a través de diferentes gráficos, fáciles de manipular. Esta plataforma permite a través de PeopleSoft Applications Portal unificar todos los procesos de negocio de las aplicaciones PeopleSoft y configurar y orquestar más de 10 000 transacciones; posee menús, opciones de registro, gestión de contenido, fórums de discusión y mensajería instantánea mejorando el potencial de la comunidad de usuarios. PeopleTools Integration y Service Oriented Architecture (SOA) benefician al sistema con servicios web estándar y una conectividad superior, reduciendo costos y complejidad. PeopleTools Administration permite administrar y mantener de forma sencilla las aplicaciones Peoplesoft. PeopleTools Security ayuda a administrar la autenticación, integración LDAP, control de contraseñas y listas de usuarios y roles. Las aplicaciones Peoplesoft soportan un amplio rango de tecnologías, incluyendo plataformas de hardware, servidores de base de datos, servidores web y de aplicaciones, así como buscadores. ((29); (30); (31); (32); (33))

1.6.3 Sage ERP X3

Sage ERP X3 está desarrollado sobre la plataforma tecnológica SAFE X3 (Sage Application Framework for the Enterprise), que es el fundamento tecnológico de toda la aplicación. Entre otras características,

esta plataforma de arquitectura Web nativa y orientada a servicios (SOA) proporciona a la empresa óptimas capacidades para la colaboración (Servicios Web, motor de flujos de trabajo de 2ª generación, etc.), potentes herramientas de Inteligencia de negocio (*Business Intelligence)* y una interfaz para usuarios finales integrada y altamente colaborativa. (34) La Arquitectura tecnológica de Sage ERP X3 permite la integración natural con Microsoft Office posibilitando a los usuarios exportar y tratar los datos en el formato estándar de Office, sin perder la precisión y la consistencia que ofrece trabajar con una fuente única y actualizada, como es la base de datos del sistema ERP. Además, los documentos Office (imágenes, presentaciones, documentos Word o Excel) pueden ser creados o modificados en el propio entorno del ERP y se pueden almacenar en Sage ERP X3 como parte de la base de datos corporativa. Los usuarios pueden acceder a todas las funciones de Sage ERP X3 a través de un navegador Web y posee un Configurador de producto personalizable. El sistema de workflow que incorpora permite poner en marcha circuitos de información (dentro y fuera de la organización), acciones de seguimiento, alertas y flujos de validación, a partir de cualquier suceso registrado en el ERP, siguiendo los criterios establecidos para la gestión de situaciones excepcionales y críticas. Para una administración eficaz de la seguridad, la arquitectura tecnológica de Sage ERP X3 facilita el establecimiento de controles de acceso a las funciones, acciones, campos y datos, por cada usuario, por grupos de usuarios o por perfiles. La conexión Web por https permite encriptar los datos que circulan por la red. Se pueden definir limitaciones de longitud y de renovación de contraseñas, bloquear a los usuarios que realicen un intento de intrusión o desconectar a los usuarios inactivos, transcurrido un tiempo establecido. Permite la trazabilidad hacia delante y hacia atrás y para el control de la misma, es posible activar distintos niveles de rastreo en función de los usuarios, definir alertas a través de flujos sobre cualquier suceso particular. Existen procesos de firma integrados en la aplicación base, concretamente para las compras, los presupuestos, los recibos a pagar, etc. La trazabilidad de modificaciones de datos sensibles puede ser configurada directamente en la base de datos. Es posible rastrear los detalles de las modificaciones. La arquitectura Web nativa del sistema permite trabajar vía Internet con clientes y proveedores. Para los informes contables y las simulaciones, Sage ERP X3 proporciona un generador de cuadros de mando financieros. Así mismo, la aplicación incorpora un conjunto de cuadros de mando predefinidos de forma estándar (balance, cuentas de resultados, saldos intermedios de gestión, etcétera). Para informes impresos, Sage ERP X3 integra el generador de informes Crystal Reports™ y presenta una biblioteca de cerca de 400 formatos de informes listos para imprimir: libros de contabilidad, balances y extractos, facturas, órdenes de

expedición y de preparación, informes de fabricación, etiquetas, listados de inventarios, análisis y consultas. El motor de importación / exportación de datos facilita la recuperación de los datos históricos, interfaceado automático y carga de datos a herramientas externas. Puede ser configurada e integra más de 100 modelos de captura predefinidos: bases de datos, inventarios, balances, presupuestos, tarifas, etc. Es posible utilizar formatos ASCII y Unicode. Por otra parte, la aplicación ofrece una completa documentación técnica y funcional, accesible en cualquier momento, en cada módulo y para cada campo, gracias a la ayuda en línea. Se puede implementar de forma global o en distintas fases -por áreas funcionales, por departamentos, por delegaciones, etcétera- a medida que la empresa crece o cambia su organización. Todas las funciones están integradas en la aplicación, sólo tienen que activarse cuando las necesites, sin necesidad de definir un proyecto desde cero y además dispone de capacidades multi-lenguaje, multi-divisa, multi-empresa, multi-planta y multi-legislación, de tal forma que se puede disponer de una solución sofisticada en un entorno multinacional tan fácilmente como si se tratara de una implantación a nivel local. (35)

1.7 Características de la Arquitectura Tecnológica de Cedrux

Cedrux es un sistema ERP implementado sobre la plataforma tecnológica Sauxe. Este último es un marco de trabajo que siguiendo el paradigma de independencia tecnológica por el cual apuesta el país, reutiliza diversas tecnologías y herramientas libres, entre las que se encuentran los lenguajes de programación PHP y JavaScript, PostgreSQL para extraer y almacenar información de la base de datos, PGAdmin III como aplicación gráfica para administrar el gestor de bases de datos PostgreSQL, Zend Framework para desarrollo de aplicaciones Web y servicios Web con PHP, ExtJs utilizado en el desarrollo de aplicaciones Web con AJAX, Doctrine como sistema ORM (Object Relational Mapper) (36), entre otras. Esta plataforma tecnológica permite gestionar, configurar y administrar las excepciones de forma dinámica y declarativa. Permite a los usuarios la gestión de objetos a nivel de datos (bases de datos, esquemas, secuencias, disparadores, funciones, roles, tablas, atributos, entre otros) sin necesidad de interactuar con algún gestor específico. Permite erradicar los conflictos o abrazos fatales que se producen cuando dos o más usuarios acceden y modifican información de un mismo fichero. Permite gestionar, configurar y administrar las validaciones para sucesos del sistema. Facilita la integración entre componentes y subsistemas permitiendo ejecutar las funcionalidades requeridas. Permite integrar todas las representaciones o respuestas del sistema que navegan hasta la capa cliente de la arquitectura en un área de trabajo única,

taxonómicamente ordenada, que se integra con los mecanismos de autenticación y autorización de la plataforma, con el marco proveedor de estructuras y con el marco de representación de mensajes. Permite la interoperación entre subsistemas para intercambiar información, así como importar y exportar datos. Facilita la accesibilidad a la información de forma rápida a través de la caché, donde se almacenan valores o conceptos muy reutilizados por los usuarios. Permite acceder a datos generales desde cualquier parte de la aplicación a través de variables globales. Permite seguir la secuencia de acciones de un usuario sobre el sistema a través de las trazas. Las operaciones de modificación sobre el modelo de dominio son manejadas como transacciones por defecto, definiendo para cada transacción cuándo termina y cuándo acaba. Permite activar o desactivar aspectos que pueden utilizarse o no en el desarrollo o ejecución del sistema. (37) Este sistema provee además un mecanismo de seguridad que permite la autenticación, autorización y auditoría, de forma que puede detectarse cualquier acción indebida de un usuario, proceso o sistema, manteniendo protegidos los datos contra intentos de acceso sin previa autenticación o permisos requeridos. (38)

Valoración de las soluciones estudiadas

Como se evidencia en la descripción anterior, el sistema Cedrux reúne diversos escenarios que lo hacen ser un producto seguro, flexible, capaz de interoperar con otros sistemas, sin embargo adolece de otros que poseen sistemas homólogos como lo son SAP, JD Edwards, PeopleSoft y Sage. Estos últimos cuatro han hecho buen uso de la tecnología construyendo potentes plataformas que permiten ofrecer al cliente un mejor producto, que se adapte mejor a sus necesidades.

La tecnología que sirve de base a sus productos de software garantiza escenarios que responden a diversos atributos de calidad. Estos presentan escenarios comunes relacionados con la seguridad, la integración e interoperabilidad, las configuraciones, las transacciones, trazas, entre otros, que coinciden con los cubiertos por el sistema Cedrux, pero también otros escenarios relacionados con el manejo de flujos de trabajo, alertas, integración con el office, manejo de las tareas de impresión, gráficos, entre otros, que los convierten en un potente referente que sirve de base para determinar los distintos escenarios tecnológicos que actualmente no cubre la arquitectura tecnológica del sistema Cedrux.

Todos tienen escenarios comunes, básicos, pero también específicos, que marcan la diferencia entre sus productos, sin embargo la recopilación de todos sus escenarios (los comunes más los específicos de cada sistema) proporcionan la base para determinar aquellos que debe cubrir la arquitectura tecnológica e un sistema de gestión empresarial.

1.8 Conclusiones parciales

- Se cumplimentó un estudio del estado del arte de la disciplina de la Arquitectura de Software y sus propiedades fundamentales. Dentro de esta, la Arquitectura Tecnológica y su función rectora como encargada de sentar la base tecnológica del sistema. Se enfatizó la importancia crucial de concederle un puesto principal a los aspectos no funcionales del software ya que estos determinan la portabilidad, mantenibilidad y otros atributos relevantes para la aplicación. Y se logra caracterizar el papel de los escenarios en la arquitectura, así como la gran connotación que tienen para el correcto desarrollo y funcionamiento de la aplicación y la satisfacción de los requisitos no funcionales.
- Las Arquitecturas Tecnológicas de los principales sistemas ERP estudiados demuestran que todos tienen puntos en común pero características propias y peculiaridades que marcan la diferencia entre sus productos finales.
- Aunque la arquitectura tecnológica del sistema Cedrux cubre escenarios básicos de un sistema de gestión empresarial, el resto de los sistemas estudiados satisfacen otro conjunto de escenarios que Cedrux no, por lo que constituyen una fuente invaluable en la identificación de escenarios que complementen la arquitectura tecnológica de Cedrux.
- La agrupación de los escenarios tecnológicos de todos los sistemas estudiados constituye la base para la identificación y descripción de los escenarios que debe cubrir la arquitectura tecnológica de un sistema de gestión empresarial.

CAPÍTULO II: DESCRIPCIÓN DE ESCENARIOS

2.1 Introducción

A lo largo de este documento se han referenciado distintos elementos que han de colaborar para, equiparados con tecnologías de punta y novedosas herramientas llevar a feliz término el proceso de construcción de una arquitectura tecnológica que garantice los principales escenarios presentes en un ERP. En este capítulo se describen los principales escenarios que debe cubrir la arquitectura tecnológica de un ERP, tanto los cubiertos por el sistema Cedrux como aquellos que no y se propone una serie de tecnologías, herramientas, patrones y estilos a utilizar para darles solución.

2.2 Escenarios que debe garantizar la Arquitectura Tecnológica de un ERP

La arquitectura tecnológica de Cedrux descrita en el capítulo anterior constituye la base para el desarrollo del sistema. La misma provee un conjunto de componentes genéricos reutilizables que garantizan los escenarios tecnológicos que apoyan el correcto funcionamiento del sistema. Sin embargo existe un cúmulo de escenarios no resueltos tecnológicamente que la misma debe cubrir; para satisfacer esta necesidad, a continuación se listarán los principales escenarios que debe garantizar la arquitectura tecnológica de un sistema de este tipo, incluyendo aquellos que no cubre la AT de Cedrux y las tecnologías, herramientas y patrones que han de colaborar para darles solución.

Una vez arribado a este punto, se hace necesario definir tres conceptos fundamentales para el entendimiento de los mismos:

Gestión: Se define en el contexto del documento como las actividades orientadas a la visualización, consulta y búsqueda de determinado concepto con respecto a la colección o dominio de representantes del mismo que han sido registrados en el sistema, incluye además las actividades asociadas a integrar conceptos o a inferir conocimiento de la relación del mismo con respecto a otros.

Configuración: Se define en el contexto del documento como las actividades orientadas a la inserción, modificación y eliminación de un concepto, donde se especifican los valores de sus atributos y combinaciones de estos, de modo que el resultado final del proceso quede persistido.

Administración: Se define en el contexto del documento como las actividades orientadas a la modificación del cambio de estado de un concepto, activarlo, desactivarlo, auditarlo, dar seguimiento a las operaciones y acciones sobre el mismos.

ESCENARIO DE SEGURIDAD

Descripción General

La arquitectura tecnológica debe proveer algún mecanismo de autenticación, autorización, administración de perfiles de usuario, administración de conexiones y auditoría, de forma que se detecte cualquier acción indebida de un usuario, proceso o sistema que y que se mantengan protegidos los datos contra intentos de acceso sin previa autenticación o permisos requeridos.

Autenticación: La autenticación (o autentificación) es el proceso de verificar formalmente la validez de la identidad de las entidades participantes en una comunicación o intercambio de información. Por entidad se entiende tanto personas, como procesos o computadoras. Para contar con una buena autentificación es necesario utilizar al menos dos de las técnicas existentes, siendo bastante frecuente el uso de sistemas basados en algo que el individuo es y que lo identifica unívocamente: la autentificación biométrica, que se basa en la identificación de personas por medio de algún atributo físico (huellas dactilares, patrón de voz, escritura) y el uso de sistemas basados en algo que solamente el individuo conoce: contraseñas (número de PIN, frases), que es el modelo de autenticación más básico, consistente en decidir si un usuario es quien dice ser simplemente basándose en una prueba de conocimiento que a priori sólo ese usuario puede superar. ((39); (40); (41))

Autorización: Es el proceso de determinar si una entidad identificada y verificada tiene permisos para el acceso a los recursos, permitiendo que sólo sean usados por aquellos consumidores a los que se les ha concedido autorización para ello. Los recursos incluyen archivos y otros objetos de datos, programas, dispositivos y funcionalidades provistas por aplicaciones. Este proceso se basa en políticas de control de acceso (reglas para especificar quién puede acceder, a que recursos). En la solución debe incluirse el concepto multientidad donde se puedan gestionar permisos de usuarios sobre sistemas en un dominio de entidades. ((39); (40); (41))

Administración de perfiles: Debe garantizarse la personalización de las aplicaciones de este dominio a nivel de cada usuario, definiéndose como perfil, los datos únicos de cada recurso dentro del sistema que define el comportamiento del mismo ante las entradas emitidas por este recurso y las salidas entregadas por el (los) subsistema (s), esto garantizaría un sin número de bondades tanto de usabilidad como de configurabilidad a la solución en cuestión. (41)

Administración de conexiones: Consiste en un grupo de procesos dedicados a la gestión de las conexiones a la base de datos de un sistema determinado ubicado en un servidor de bases de datos que

debe definirse también como un parámetro configurable, así como el gestor en uso. Esta solución debe incluir la gestión dinámica de usuarios de bases de datos y los permisos sobre ellas. (41)

Auditoría: Constituye un conjunto de procedimientos y técnicas para evaluar y controlar total o parcialmente un sistema informático con el fin de proteger sus activos y recursos, verificar si sus actividades se desarrollan eficientemente de acuerdo con las normas informáticas y generales existentes en cada empresa y para conseguir la eficacia exigida en el marco de la organización correspondiente.

Partes del escenario (ver Anexo 1)

ESCENARIO DE ASPECTOS

Descripción General

La arquitectura tecnológica debe proveer alguna solución para la configuración dinámica y centralizada de aspectos, permitiendo activar o desactivar aspectos como la seguridad, las trazas, los historiales, las validaciones u otros que pueden o no ser usados en el desarrollo o ejecución de un sistema.

Partes del escenario (ver Anexo 2)

ESCENARIO DE CACHÉ

Descripción General

La arquitectura tecnológica debe proveer algún mecanismo que permita configurar y gestionar de manera dinámica la caché para facilitar el manejo y accesibilidad a información importante de forma rápida.

La misma deberá estar orientada al cliente (paginas, imágenes, css, js), al código fuente (acelerador de código, caché del binario que se instancia), al dominio de la solución (caché de instancia de objetos, estructuras, consultas, ficheros XML, etc.) Esta solución debe además integrarse con el mecanismo de las excepciones para los casos de ocurrencia de algún fallo en el acceso o almacenamiento de la información. (37)

Partes del escenario (ver Anexo 3)

ESCENARIO DE PERSISTENCIA DE DATOS

Descripción General

La arquitectura tecnológica debe proveer algún mecanismo de abstracción de la capa relacional de persistencia que permita la persistencia compuesta, actualización y modificación sincronizada tanto en el modelo mapeado como en el modelo relacional, contando de esta forma con un mecanismo de

sincronización práctico y eficiente. Debe permitir además la administración de conexiones y de transacciones, así como realizar operaciones de consulta u operaciones CRUD sobre el esquema mapeado ejecutándose los mismos en la base de datos de forma transparente para el desarrollador o usuario, y exportar una base de datos existente a sus clases correspondientes y también a la inversa, es decir convertir clases a tablas de una base de datos.

Partes del escenario (ver Anexo 4)

ESCENARIO DE VALIDACIONES

Descripción General

La arquitectura tecnológica debe proveer algún mecanismo de gestión, configuración y administración de validaciones para sucesos del sistema, inyectando tanto al script presente o enviado al lado del cliente como al script o código perteneciente al lado del servidor, las instrucciones o reglas de validación. Las mismas deberán estar basadas en un formato y podrán ser extendidas, modificadas o adicionadas para una configuración arquitectónica específica. La validación debe estar concentrada en dos áreas fundamentales: una primera área de validación orientada a las clases de equivalencia de los atributos precisados según el contrato definido o bien para el formulario cliente o para la funcionalidad del lado del servidor que se invoque y la otra área orientada a las reglas pre-condicionales de las unidades de ejecución (acciones por ejemplo), de manera que el sistema pueda mediante al marco de validación chequear mediante el uso de inversión de control las reglas de validación declarativamente configuradas para la acción en concreto. Este marco de solución de la plataforma debe poder integrarse con el marco de trazas para dejar registro de las validaciones que fallan. ((41); (37))

Partes del escenario (ver Anexo 5)

ESCENARIO DE EXCEPCIONES

Descripción General

La solución arquitectónica debe proveer algún mecanismo de gestión, configuración y administración de excepciones de manera dinámica y declarativa. Este mecanismo debe integrarse con el elemento de las trazas cuando ocurra el suceso de una excepción y definir además familias de excepciones y dentro de las familias de excepciones, excepciones tipo concretas. De cada excepción deben poder definirse características tales como: línea de código e instrucción en que se generó, excepción que la inició, fecha y hora, entre otros metadatos que permitan la visualización y el análisis de las trazas emitidas por el sistema

para cada excepción. También debe poder definirse de manera declarativa el comportamiento de la excepción, si se persistirá, si será presentada con un mensaje de notificación que navegue hasta la capa cliente, si será una excepción ciega, o una excepción a reemplazar por otra nueva excepción que formatee la expresión. La solución de excepción debe controlar además, aquellas excepciones disparadas por el sistema y conocidas como excepciones no controladas o errores internos y debe permitir configurar varios publicadores para cada familia de excepción o para cada excepción en concreto, usando para ello algún mecanismo de inversión de control, o idóneamente colaborando con la solución de traza con la que debe contar la solución tecnológica. (37)

Partes del escenario (ver Anexo 6)

ESCENARIO DE PORTAL FRONTAL

Descripción General

La arquitectura tecnológica debe tener la capacidad de integrar todas las representaciones o respuestas del sistema que naveguen hasta la capa cliente de la arquitectura en un área de trabajo única, taxonómicamente ordenada que se integre con los mecanismos de autenticación y autorización, con el marco proveedor de estructuras y composiciones y con el marco de representación de mensajes. La misma debe basarse en una plataforma que soporte las tecnologías de presentación (como HTML, CSS, DHTML, JS) y mecanismos de comunicación asíncrona con el servidor (como Ajax). (37)

Partes del escenario (ver Anexo 7)

ESCENARIO DE METADATOS

Descripción General

La arquitectura tecnológica debe proveer algún mecanismo para la gestión de conceptos y atributos dinámicos (metadatos), que no son más que aquellos conceptos que generen la necesidad de incorporar atributos a una entidad de negocio, una vez que haya sido modelada bajo determinado escenario. Este mecanismo debe permitir que los conceptos identificados crezcan en atributos sin tener que modificar el código fuente o el modelo de datos de manera que el usuario defina los conceptos que se ajusten a sus características. La idea es que dado una determinada entidad de negocio del sistema (Ej.: Persona), se le puedan adicionar todos aquellos atributos (Ej.: color de ojos, sexo, edad) que se deseen, a los cuales se les podrá luego asignar un dominio que será validado al momento de ingresar el dato. (Ej.: El dominio válido para el color de ojos es: pardos, verdes, azules)

Partes del escenario (ver Anexo 8)

ESCENARIO DE INSTALACIÓN

Descripción General

La arquitectura tecnológica debe proveer un mecanismo que le permita al usuario final la instalación integral del sistema o por módulos teniendo en cuenta la dependencia entre los mismos, permitiendo insertar tareas en el flujo de trabajo de instalación para la configuración inicial. Además debe permitir que el sistema se pueda actualizar a partir de y hasta una versión una versión dada, así como dar la posibilidad de instalar en entornos donde la app y la base de datos puedan encontrarse en un mismo servidor o no.

Partes del escenario (ver Anexo 9)

ESCENARIO DE INTEROPERABILIDAD

Descripción General

La arquitectura tecnológica debe proveer algún mecanismo que permita comunicarse con otros sistemas externos para transmitir y recibir información, notificar o gestionar sucesos, así como para importar y exportar datos.

Partes del escenario (ver Anexo 10)

ESCENARIO DE CONCURRENCIA

Descripción General

La arquitectura tecnológica debe proveer algún mecanismo de configuración y gestión dinámica de las características de concurrencia sobre entidades o dominios de la solución lógica que se modela. Este mecanismo debe permitir configurar el esquema de concurrencia que se desee sobre las entidades (conceptos) o estructuras de entidades del esquema de persistencia de una solución específica. Las opciones de concurrencia deben permitir implementar el mecanismo pesimista o el mecanismo optimista según se desee. También, definir los tipos de instanciación (lectura, escritura, bloqueado) del recurso o estructura lógica a la que se le gestione el esquema de concurrencia y que estas características de instanciación en el esquema de concurrencia estén asociadas al valor de algún atributo del concepto o a un rol de acceso específico.

Partes del escenario (ver Anexo 11)

ESCENARIO DE TRANSACCIONES

Descripción General

La arquitectura tecnológica debe proveer un mecanismo de administración de transacciones, de manera que las operaciones de modificación sobre el modelo de dominio sean transaccionales por defecto. Este mecanismo debe ser transparente para el programador y debe permitir configurar el esquema transaccional de un dominio específico de la solución que se construye, definiéndose en este caso para una transacción de negocio, cuando inicia y cuando debe terminar. Además de permitir configurar notificaciones ante la posibilidad de fallas en la transacción, capturándose como un evento más el rollback de la transacción.

Partes del escenario (ver Anexo 12)

ESCENARIO DE PISCINA DE CONEXIONES

Descripción General

La arquitectura tecnológica debe proveer algún mecanismo o componente de administración y configuración de la Piscina de conexiones con los gestores más utilizados, de esta forma los demás componentes contarán con una interfaz única de configuración y se controlarán centralizadamente todas las conexiones, pudiéndose auditar y ejecutar acciones que mejoren el rendimiento y la trasmisión de datos.

Partes del escenario (ver Anexo 13)

ESCENARIO DE PROCESAMIENTO EN LOTE

Descripción General

La arquitectura tecnológica debe proveer un mecanismo de gestión, configuración y administración de procesamiento en lote, el mismo debe permitir configurar tanto el responsable del comportamiento del procesamiento, como los destinos a los que debe emitirse el resultado del procesamiento, así como el buffer de transmisión, el mecanismo de mensajería a utilizar y el buffer de la cola de encargos a procesar. Este marco de solución arquitectónica debe estar integrado además al sistema de traza para dejar registro del lote que se procesó, notificar ante cualquier fallo, o bien para realizar un monitoreo de cuanta información queda por procesar y cuanta se ha procesado.

Partes del escenario (ver Anexo 14)

ESCENARIO DE TRAZAS

Descripción General

La arquitectura tecnológica debe proveer algún mecanismo de administración y configuración dinámica de trazas de la solución, de forma que se pueda configurar la arquitectura de traza de un dominio de aplicación específico, permitiendo para ello, crear categorías de trazas que constituyan familias de trazas, donde a su vez cada categoría de traza presente instancias de trazas específicas o sub-categorías. Las categorías de trazas pueden ser de diferentes dominios, según interés del cliente de la tecnología, aunque la solución debe presentar categorías de trazas tecnológicas por defecto que permitan registrar los sucesos de excepciones, entradas al sistema de los usuarios, ejecución de una acción, ejecución de una integración interna o externa dentro de la arquitectura. Estas trazas por defecto presentarán los respectivos metadatos de información que los caracteriza. La solución arquitectónica de traza debe permitir además, que cada traza presente una cola de subscritores, o lo que es lo mismo, una colección de flujos o notificaciones que deben ser disparados ante la ocurrencia de la misma, además de presentar una cola de publicadores, colección de elementos que se encargan de implementar mecanismos de persistencia específicos de la traza. Ambas colas deben poder utilizar algún mecanismo de inversión de control para implementar sus funcionalidades y también algún componente de gestión centralizada de notificaciones y observadores. Esta solución debe permitir también una fácil integración con otros elementos tecnológicos o de negocio, en el caos de los componentes que instancian los desarrolladores. (37)

Partes del escenario (ver Anexo 15)

ESCENARIO DE OBJETOS A NIVEL DE DATOS

Descripción General

La arquitectura tecnológica de la plataforma provee un mecanismo para la gestión de objetos a nivel de datos (bases de datos, esquemas, secuencias, disparadores, funciones, roles, tablas, atributos, etc.) desde la capa de aplicación. De esta forma los usuarios pueden configurar a nivel de sistema operaciones como exportar e importar datos, gestionar estructuras dinámicas como nomencladores, atributos y relaciones entre las estructuras desde las aplicaciones sin necesidad de interactuar directamente con un cliente de algún gestor en específico.

Partes del escenario (ver Anexo 16)

ESCENARIO DE CONFIGURACIONES

Descripción General

La arquitectura tecnológica de la plataforma debe proveer un mecanismo que maneje de forma dinámica y centralizada las configuraciones de las aplicaciones que se instancien sobre la plataforma tecnológica. El mismo debe ser capaz de refactorizar los ficheros de configuración y crear objetos o entidades de configuración centralizadas en el marco de solución que se describe, constituyendo el proveedor centralizado de configuraciones de la plataforma tecnológica. Este mecanismo debe estar integrado al marco de caché para elevar la eficiencia con que estos recursos persistidos en ficheros de configuración son accedidos y modificados, además de impedir situaciones de concurrencia que lleven a abrazo fatal sobre los ficheros de configuración de aspectos y variables globales de la solución arquitectónica.

Partes del escenario (ver Anexo 17)

ESCENARIO DE VARIABLES GLOBALES

Descripción General

La arquitectura tecnológica debe proveer algún mecanismo para la configuración dinámica y centralizada de variables globales, que son aquellos aspectos que agrupan información común conceptualizada que se maneja en todo el sistema, o sea, en todos los módulos y sesiones de una aplicación; de forma que cada parte del sistema pueda acceder de manera fácil y sencilla a datos que oferte otra sin incurrir en códigos engorrosos que impliquen gastos de tiempo y esfuerzo para los programadores. La implementación de estos contenedores de información común debe permitir 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. (37)

Partes del escenario (ver Anexo 18)

ESCENARIO DE INTEGRACIÓN

Descripción General

La arquitectura tecnológica debe proveer algún mecanismo que permita la integración entre componentes, subsistemas y sistemas para llevar a cabo la realización de las funcionalidades requeridas. Esta integración debe permitirse en tres niveles diferentes:

1er nivel: Es el nivel más simple de integración, se presenta cuando se combinan componentes para dar soporte a las funcionalidades de un subsistema.

2do nivel: Se presenta cuando los procesos abarcan funcionalidades que son responsabilidad de distintos subsistemas. Es un tanto más complejo que el nivel anterior pues los subsistemas involucrados pueden estar distribuidos en diferentes entornos y se deben evitar las dependencias fuertes.

3er nivel: Se presenta cuando las aplicaciones no son capaces de cubrir por sí solas todos los procesos que necesitan las entidades y deben integrarse con otras aplicaciones que las soporten. Es el nivel más complejo pues las aplicaciones a integrar no siempre comparten directrices tecnológicas.

Partes del escenario (ver Anexo 19)

ESCENARIO DE BASE DE DATOS

Descripción General

La arquitectura tecnológica debe proveer algún manejador de base de datos que proporcione un entorno que sea a la vez conveniente y eficiente para ser utilizado al extraer, almacenar y manipular información de la base de datos, donde todas las peticiones de acceso a la misma se manejen centralizadamente. Este mecanismo debe garantizar que los usuarios autorizados tengan acceso a los datos cuando lo necesiten para atender las necesidades del negocio. Debe permitir además la realización de aquellas tareas destinadas al control y respaldo de las bases de datos como pueden ser: control de integridad, chequeo de consistencia, control del rendimiento del servidor, verificación de que el disco duro no esté sobrecargado, copias de seguridad o compactación de las bases y también realizar auditorías con el objeto de saber qué o quién realizó una determinada modificación y en qué momento.

Partes del escenario (ver Anexo 20)

ESCENARIO DE EXPORTACIÓN E IMPORTACIÓN DE DATOS

Descripción General

La arquitectura tecnológica debe permitir a los diferentes usuarios que operan sobre el sistema importar y exportar datos en diferentes formatos.

Partes del escenario (ver Anexo 21)

ESCENARIO DE ESTRUCTURA Y COMPOSICIÓN

Descripción General

La arquitectura tecnológica debe proveer algún mecanismo que permita crear la estructura y composición de la empresa o entidad que utilice el sistema. Este mecanismo debe permitir la gestión estructural, proveyendo así, la capacidad de soportar una estructura que contenga muchas entidades y que se adapte a las especificidades de cada una, nutriéndose así de un mecanismo comunicativo entre estas. Debe permitir crear todas las entidades y su nivel de subordinación para que luego el resto de los módulos se encarguen de desarrollar los procesos especializados para algunas de estas entidades, así como la modificación de las ya existentes, o sea, debe permitir al usuario definir la estructura organizativa en la cual se va a ubicar su entidad y la estructura dentro de dicha entidad, así como especificar los cargos por los cuales van a estar compuestas las diferentes áreas dentro de las unidades teniendo en cuenta el marco jurídico y legal. Esta solución además debe ser configurable para que los usuarios no estén obligados a usar los mismos conceptos estructurales. (42)

Partes del escenario (ver Anexo 22)

ESCENARIO DE AYUDA

Descripción General

La arquitectura tecnológica debe proveer algún mecanismo generador de ayuda para brindar la información referente al funcionamiento de cada una de las partes que conforman el sistema y facilitar y mejorar el trabajo de los usuarios con el mismo. Este mecanismo debe brindar una forma fácil de gestionar y configurar documentación técnica y sistemas de ayuda que incluyan elementos tales como temas de ayuda, tablas de materias, índices, glosarios y ayuda contextual. Asimismo debe permitir generar ayudas en diferentes formatos (como HTML, CHM) y documentación impresa, además de exportar e importar una ayuda previamente creada.

Partes del escenario (ver Anexo 23)

ESCENARIO DE RÉPLICA DE DATOS

Descripción General

La Arquitectura tecnológica debe proveer algún mecanismo que permita la copia y distribución de datos y objetos de base de datos desde una base de datos a otra, para luego sincronizarlas y mantener su coherencia. Este mecanismo de replicación debe permitir distribuir datos entre diferentes ubicaciones y entre usuarios remotos o móviles mediante redes locales y de área extensa, conexiones de acceso telefónico, conexiones inalámbricas e Internet. Asimismo debe poder realizarse la replicación tanto en

entornos con conexión o sin ella y que al realizar cambios en los objetos de las base de datos, ya sea en la empresa principal o en cualquier entidad, este cambio sea actualizado en todas las base de datos donde se encuentre una instancia de dicho objeto, de forma tal que el flujo de datos se comporte en ambos sentidos (bidireccional), permitiendo la replicación desde las entidades a la institución central y viceversa.

Partes del escenario (ver Anexo 24)

ESCENARIO DE REPORTES DINÁMICOS

Descripción General

La arquitectura tecnológica debe proveer algún mecanismo que permita de forma dinámica la generación, administración, configuración y visualización de reportes. Esta solución debe permitir que los usuarios diseñen los reportes con los estándares definidos, imprimir dichos reportes en formatos como PDF, HTML, WORD entre otros, además de configurar los atributos que se desean mostrar, los nombres que los identificarán y de que modelos de datos serán extraídos.

Partes del escenario (ver Anexo 25)

ESCENARIO DE FLUJOS DE TRABAJO

Descripción General

La arquitectura tecnológica debe proveer algún mecanismo de gestión, configuración y administración para flujos de trabajo de forma dinámica y declarativa que permita configurar arquitecturas de flujos de trabajo expresando para ello tres elementos arquitectónicos concretos: las actividades, que pueden ser clasificadas en actividades simples, compuestas, actividades provistas (las que viene como resultado de un evento, una traza o un servicio Web) y condicionales; las reglas de control o reglas pre-condicionales de cada actividad del flujo y por último el flujo en sí, que quedará compuesto por la secuencia de actividades y reglas de control de cada actividad, en una configuración arquitectónica de flujo de trabajo que podrá ser exportado, probado o simulado, de manera que tecnológicamente se cuente con la capacidad de trabajar con programas de flujos de trabajo, que podrán ser persistidos, exportados e instanciados en cualquier subrutina secuencial no interna de un flujo de trabajo, siendo tecnológicamente transparente para el programador la instanciación del marco que se describe. El marco para la gestión de flujos de trabajo, debe establecer además algún mecanismo para persistir el estado de un flujo de trabajo, para guardar el historial de las actividades y de esta forma permitir que el flujo de trabajo pueda ser

además recorrido hacia adelante y hacia tras. Los flujos de trabajos además deben poder configurar su carácter transaccional, integrando este marco al de transacciones, excepciones e inversión de control según sea el caso de aplicación que finalmente se decida.

Partes del escenario (ver Anexo 26)

ESCENARIO DE GRÁFICOS

Descripción General

La arquitectura tecnológica debe proveer algún mecanismo que permita la generación de diferentes tipos de gráficos para realizar análisis estadístico.

Partes del escenario (ver Anexo 27)

ESCENARIO DE MAPAS

Descripción General

La arquitectura tecnológica debe proveer algún mecanismo de generación de mapas que permita hacer uso de la cartografía temática para la representación de indicadores y el análisis de comportamiento de la distribución espacial de información socioeconómica. Posibilitando comprender de una forma más intuitiva y gráfica los resultados estadísticos que pueden resultar de vital importancia para la toma de decisiones.

Partes del escenario (ver Anexo 28)

ESCENARIO DE ALERTAS Y AVISOS

Descripción General

La arquitectura tecnológica debe proveer algún mecanismo de alertas y avisos que permita la subscripción de los componentes a eventos disparados por otros componentes de manera que se envíe una alerta del momento en que cierto servicio o recurso ha sido liberado y puede comenzar a usarse. Estas alertas pueden ser enviadas por diferentes vías: telefónica, mensajería instantánea, correo, bíper o por el mismo sistema.

Partes del escenario (ver Anexo 29)

ESCENARIO DE COMPLEMENTOS OFFICE

Descripción General

La arquitectura tecnológica debe proveer algún mecanismo que permita integrar funciones tipo del ERP con las funcionalidades brindadas por el Office, de manera que los usuarios puedan utilizar las facilidades brindadas por la hoja de cálculo para el trabajo con funciones tipo de los módulos contables y financieros.

Partes del escenario (ver Anexo 30)

ESCENARIO DE HISTORIALES

Descripción General

La arquitectura tecnológica de la plataforma debe proveer un mecanismo para la configuración dinámica y centralizada de la gestión de historiales. Este mecanismo debe permite configurar a que sistema o sistemas se les va a gestionar historiales, el tiempo que permanecerá activo, los parámetros o datos que se van a registrar, acceder a ellos filtrando por parámetros como fecha y entidades y en un momento determinado incluir en las bases de datos que manipula el sistema todos o parte de los datos registrados en el historial.

Partes del escenario (ver Anexo 31)

ESCENARIO DE GESTIÓN DOCUMENTAL

Descripción General

La arquitectura tecnológica debe proveer algún mecanismo permita la gestión documental, llevando a cabo el procesado, almacenamiento, búsqueda, recuperación y distribución de documentos al conjunto de usuarios que operen sobre el mismo.

Partes del escenario (ver Anexo 32)

ESCENARIO DE IMPRESIÓN

Descripción General

La arquitectura tecnológica debe proveer algún mecanismo que permita la configuración administración y monitorización de las actividades de impresión, de forma que un usuario pueda programar las actividades de impresión (tiempo de comienzo, impresora que realizará el trabajo) y continuar trabajando en otras labores, pudiendo desactivar o monitorizar el servicio de impresión cuando desee.

Partes del escenario (ver Anexo 33)

2.3 Herramientas y Tecnologías

El primer elemento que debe regir la selección adecuada de tecnologías radica en las restricciones de diseño que se deben asumir, basadas en las posibilidades de adquisición las mismas, la capacidad técnica del equipo de desarrollo, los intereses jurídicos mercantiles, así como la infraestructura de la organización productora y cliente. Es recomendable trabajar con un mismo marco tecnológico por problemas de incompatibilidad, integración, rendimiento, entre otros. En el caso particular del sistema Cedrux la mayoría de los escenarios identificados deben encontrar solución en las herramientas y tecnologías definidas por el Marco de Trabajo Sauxe.

Lenguajes de modelado y desarrollo:

Lenguaje de modelado

Los lenguajes de modelado de objetos constituyen el conjunto estandarizado de símbolos y las distintas combinaciones de la disposición para modelar un diseño de software.

- **UML:** Lenguaje Unificado de Modelado es un lenguaje para modelar y ensamblar una aplicación visualmente que define reglas sintácticas que especifican cómo combinar elementos del lenguaje e incluye también un cierto número de reglas semánticas. (6)

Lenguajes de programación

Lenguaje del lado del servidor

Se clasifica así al lenguaje de programación en la tecnología cliente servidor que se ejecuta del lado del servidor y del cual los usuarios solo obtienen el beneficio del procesamiento de la información.

- **PHP:** Hypertext Processor es un lenguaje sencillo, de sintaxis cómoda y dispone de muchas librerías que facilitan en gran medida el desarrollo de las aplicaciones; convirtiéndolo en el favorito de millones de programadores en todo el mundo. Entre las ventajas que presenta se encuentran: (43)
  - Es multiplataforma
  - Es Open Source, lo cual significa que el usuario no depende de una compañía específica para arreglar cosas que no funcionan.
  - Capacidad de conexión con la mayoría de los motores de base de datos que se utilizan en la actualidad, destaca su conectividad con MySQL y PostgreSQL.
  - Es completamente expandible. Está compuesto de un sistema principal (escrito por Zend), un conjunto de módulos y una variedad de extensiones de código.

- Rapidez. PHP generalmente es utilizado como módulo de Apache, lo que lo hace extremadamente veloz.

Lenguaje del lado del cliente

Un lenguaje del lado cliente es totalmente independiente del servidor, lo cual permite que la página pueda ser albergada en cualquier sitio.

- **JavaScript:** Es un lenguaje de programación que se utiliza principalmente para crear páginas web dinámicas. Técnicamente, JavaScript es un lenguaje de programación interpretado, por lo que no es necesario compilar los programas para ejecutarlos. En otras palabras, los programas escritos con JavaScript se pueden probar directamente en cualquier navegador sin necesidad de procesos intermedios. A pesar de su nombre, JavaScript no guarda ninguna relación directa con el lenguaje de programación Java. Legalmente, JavaScript es una marca registrada de la empresa Sun Microsystems. (44)
- **CSS:** Es un lenguaje de hojas de estilos (Cascading Style Sheets) creado para controlar el aspecto o presentación de los documentos electrónicos definidos con HTML y XHTML. Al crear una página web, se utiliza en primer lugar el lenguaje HTML/XHTML para marcar los contenidos, es decir, para designar la función de cada elemento dentro de la página: párrafo, titular, texto destacado, tabla, lista de elementos, etc. Una vez creados los contenidos, se utiliza el lenguaje CSS para definir el aspecto de cada elemento: color, tamaño y tipo de letra del texto, separación horizontal y vertical entre elementos, posición de cada elemento dentro de la página, etc. El propósito del desarrollo de CSS es separar la estructura y el contenido de la presentación estética en un documento, lo que permite una mejor accesibilidad y un control mayor del documento y sus atributos, convirtiendo al HTML en un documento muy versátil y liviano. (45)
- **XML:** Siglas en inglés de Extensible Markup Language (lenguaje de marcado extensible), es un metalenguaje extensible de etiquetas. Permite definir la gramática de lenguajes específicos, por lo tanto no es realmente un lenguaje en particular, sino una manera de definir lenguajes para diferentes necesidades. Este concepto no ha nacido sólo para su aplicación en Internet, sino que se propone como un estándar para el intercambio de información estructurada entre diferentes plataformas. Se puede usar en bases de datos, editores de texto, hojas de cálculo y casi cualquier aplicación imaginable. (46)

- **HTML:** Hypertext Markup Language es un lenguaje de marcado predominante para la construcción de páginas web. Es un formato no propietario basado en SGML, y puede ser creado y procesado por una gama amplia de herramientas, desde simples editores de texto planos hasta las herramientas sofisticadas de autorización WYSIWYG . HTML usa etiquetas como < h1> y </ h1> para estructurar texto en los títulos, párrafos, listas, vínculos de hipertexto, etc. (47)

Tecnologías AJAX

- **Ajax:** El término AJAX se presentó por primera vez en el artículo "Ajax: A New Approach to Web Applications" publicado por Jesse James Garrett el 18 de Febrero de 2005. Hasta ese momento, no existía un término normalizado que hiciera referencia a un nuevo tipo de aplicación web que estaba apareciendo. En realidad, el término AJAX es un acrónimo de Asynchronous JavaScript + XML, que se puede traducir como "JavaScript asíncrono + XML". El artículo define AJAX de la siguiente forma: Ajax no es una tecnología en sí mismo. En realidad, se trata de varias tecnologías independientes que se unen de formas nuevas y sorprendentes.Las tecnologías que forman AJAX son:
  - XHTML y CSS, para crear una presentación basada en estándares.
  - DOM, para la interacción y manipulación dinámica de la presentación.
  - XML, XSLT y JSON, para el intercambio y la manipulación de información.
  - XMLHttpRequest, para el intercambio asíncrono de información.
  - JavaScript, para unir todas las demás tecnologías. (Eguíluz Pérez)

Marcos de trabajo

- **Zend:** Se trata de un Marco de Trabajo de código abierto para el desarrollo de aplicaciones y servicios web con PHP, brinda soluciones para construir sitios web modernos, robustos y seguros. Es una implementación que usa código 100% orientado a objetos. Este Marco de Trabajo está formado por una serie de métodos estáticos y componentes que usarán estos métodos. Aunque se pueden utilizar de forma individual, los componentes de la biblioteca estándar de ZendFramework conforman un potente y extensible Marco de Trabajo de aplicaciones web al combinarse. Como características fundamentales ZendFramework tiene: (37)

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.
- 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.
- Robustas clases para autenticación y filtrado de entrada.
- Muchas otras clases útiles para hacerlo tan productivo como sea posible

Ventajas principales de Zend Framework: (37)

- Estandariza los procesos más frecuentes, dotándolos de gran robustez.
- Facilita el mantenimiento de las aplicaciones.
- Ofrece muchas facilidades para el acceso a recursos avanzados que de otro modo resultan bastante más costosos de desarrollar.
- A diferencia de otros Marcos de Trabajo, es posible utilizarlo en modo "desacoplado", es decir, aquellas clases o componentes que sean necesarios en cada proyecto, sin arrastrar todo el Marco de Trabajo detrás para cualquier pequeña necesidad.
- Tiene el respaldo de la propia ZEND, creadora de PHP, lo que asegura su continuidad futura tanto como la del propio lenguaje PHP.

- **ExtJs:** ExtJs es una librería Java Script de alto rendimiento, compatible con la mayoría de los navegadores y permite crear páginas e interfaces web dinámicas. 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. Esta librería incluye: Componentes UI (Interfaz de Usuario) del alto rendimiento 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 se evita 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 sólo se dibujan los bordes haciendo que el movimiento sea fluido lo cual representa una ventaja frente a otros.

- **Doctrine:** Doctrine es un potente y completo sistema ORM (Mapeador de Objeto Relacional) para PHP 5.2.3+ que incorpora una DBL (Capa de Abstracción a Base de Datos). Sauxe utiliza en la capa de acceso a datos el Lenguaje de Consulta de Datos (DQL) que implementa Doctrine. La documentación de éste tiene todas las características necesarias para ser funcional en casi cualquier proyecto. 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) a tablas de una base de datos. Entre las ventajas que facilitan enormemente tareas comunes y de mantenimiento se encuentran: (37)
  - 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.
  - Encapsulación: La capa ORM encapsula la lógica de los datos pudiendo hacer cambios que afectan a toda la aplicación únicamente modificando una función.
  - Portabilidad: Utilizar una capa de abstracción que permite cambiar en mitad de un proyecto de una base de datos MySQL2 a una Oracle sin ningún tipo de complicación. Esto es debido a que no se utiliza una sintaxis (MySQL, Oracle3 o SQLite4) para acceder al modelo, sino una sintaxis propia del ORM utilizado que es capaz de traducir a diferentes tipos de bases de datos.
  - Seguridad: Los ORM suelen implementar mecanismos de seguridad que protegen nuestra aplicación de los ataques más comunes como una Inyección SQL5.
  - Mantenimiento del código: Gracias al correcto ordenamiento de la capa de datos, modificar y mantener el código es una tarea sencilla

Entorno Integrado de Desarrollo

Un entorno de desarrollo integrado o IDE (en inglés: Integrated Development Environment), es un programa informático compuesto por un conjunto de herramientas de programación que permite de forma cómoda y ágil editar, compilar, ejecutar y depurar programas. (48)

- **Netbeans:** NetBeans IDE es una buena herramienta para que los programadores puedan escribir, compilar, depurar y ejecutar programas. Es multilenguaje, completo y modular, pues aunque está

escrito en Java puede servir para cualquier otro lenguaje de programación como C/C++, Ruby on Rails, PHP, Groovy, Phyton, JavaScript. Es gratis y Opensource. Posee gran cantidad de módulos de terceros (plugins) Cracterísticasas el Neatbeans: ((49); ((50))

- Mejoras en el editor de código
- Soporte para varios lenguajes
- Enlazar datos con el Swing GUI
- Cracterísticas visuales para el desarrollo web
- Mejoras para SOA y UML
- Soporte para PHP

Herramienta CASE

Las Herramientas CASE (Computer Aided Software Engineering, “siglas en inglés” Ingeniería de Software Asistida por Ordenador) pueden definirse como un conjunto de programas y ayudas que dan asistencia a los analistas, ingenieros de software y desarrolladores, durante todos los pasos del ciclo de vida de desarrollo de un software. (51)

- **Visual Paradigm:** Es una herramienta CASE que utiliza UML como lenguaje de modelado y está desarrollada por Visual Paradigm Internacional, una de las principales compañías de herramientas CASE. Es una herramienta profesional que soporta el ciclo de vida completo del desarrollo de software: análisis y diseño orientados a objetos, construcción, pruebas y despliegue. El software de modelado ayuda a una más rápida construcción de aplicaciones de calidad, mejores y a un menor coste. Permite dibujar todos los tipos de diagramas de clases, código inverso, generar código desde diagramas y generar documentación. (51)

Características del Visual Paradigm: (51)

- Visual Paradigm utiliza UML como lenguaje de modelado para la construcción de los sistemas ofreciendo soluciones de software que permiten a las organizaciones desarrollar aplicaciones de calidad de forma rápida y barata.
- Tiene la capacidad de ejecutarse sobre diferentes sistemas operativos.
- Fácil de instalar y actualizar.

- Integra diferentes funcionalidades para el desarrollo de aplicaciones como el modelado de UML, el modelado de base de datos, el modelado de requerimientos, el modelado del proceso de negocio, la interoperabilidad, la generación de documentación entre otros.
- Ofrece compatibilidad entre ediciones.

Herramienta de desarrollo colaborativo

Control de versiones

El control de versiones es el arte de manejar cambios en la información. Ha sido desde siempre una herramienta crítica para los programadores, quienes típicamente emplean su tiempo haciendo pequeños cambios al software y luego deshaciendo o comprobando esos cambios al día siguiente.

- **SVN:** También conocido como SVN, es un sistema de control de versiones que se ha popularizado bastante, en especial dentro de la comunidad de desarrolladores de software libre. Está preparado para funcionar en red y se distribuye bajo licencia libre.

Ventajas del SVN: (52)

- Manejo de datos consistente: Subversion expresa las diferencias entre ficheros usando un algoritmo de diferenciación binario, que funciona exactamente igual tanto en ficheros de texto (legibles por los humanos) como en ficheros binarios (que no son legibles por nosotros). Ambos tipos de ficheros se almacenan igualmente comprimidos en el repositorio, y las diferencias se transmiten en ambas direcciones por la red.
- Etiquetado y creación de ramas eficiente: El coste de crear una rama o una etiqueta no necesita ser proporcional al tamaño del proyecto. Subversion crea ramas y etiquetas símplemente copiando el proyecto, utilizando un mecanismo similar a los vínculos duros. Por tanto estas operaciones llevan un tiempo pequeño y constante, y muy poco espacio en el repositorio.
- Extensibilidad: Subversion no tiene lastre histórico; está implementado como una colección de librerías C compartidas con APIS bien definidas. Esto hace que Subversion sea extremadamente mantenible y se pueda utilizar por otras aplicaciones y lenguajes.

Servidor de aplicaciones Web

- **Apache:** Es un servidor web gratuito y potente que ofrece un servicio estable. Es sencillo de mantener y configurar e indiscutiblemente uno de los mayores logros del Software Libre. Es multiplataforma,

aunque idealmente está preparado para funcionar bajo Linux, es Open source, cuenta con amplias librerías de PHP y Perl a disposición de los programadores, posee diversos módulos que permiten incorporarle nuevas funcionalidades, y son muy simples de carga. Es capaz de utilizar lenguajes como PHP, TCL, Python, entre otros. Tiene amplia aceptación en la red: desde 1996, es el servidor HTTP más usado en el mundo y alcanzó su máxima cuota de mercado en 2005 siendo el servidor empleado en el 70% de los sitios web en el mundo. (53)

Sistema Gestor de Base de Datos

- **PostgreSQL:** Es un servidor de base de datos relacional, libre. Tiene soporte total para transacciones, disparadores, vistas, procedimientos almacenados, almacenamiento de objetos de gran tamaño. Se destaca en la ejecución de consultas complejas, consultas sobre vistas, subconsultas y *joins* de gran tamaño. Permite la definición de tipos de datos personalizados e incluye un modelo de seguridad completo. Como toda herramienta de software libre PostgresSQL tiene entre otras ventajas las de contar con una gran comunidad de desarrollo en Internet, su código fuente está disponible sin costo alguno y algo muy importante es que dicha herramienta es multiplataforma. Además de sus ofertas de soporte, cuenta con una importante comunidad de profesionales y entusiastas de PostgreSQL de los que los centros de desarrollos pueden obtener beneficios y contribuir. Está disponible en casi cualquier Unix (34 plataformas en la última versión estable), y una versión nativa de Windows está actualmente en estado beta de pruebas. (53)

Estilos y Patrones

La arquitectura del marco de trabajo Sauxe está conformada por las diferentes estilos y patrones arquitectónicos que serán especificados a continuación:

Arquitectura basada en capas: Las ventajas que reporta este estilo de arquitectura en capas son obvias. Primero que nada, el estilo soporta un diseño basado en niveles de abstracción crecientes, lo cual a su vez permite a los implementadores la partición de un problema complejo en una secuencia de pasos incrementales. En segundo lugar, el estilo admite muy naturalmente optimizaciones y refinamientos. En tercer lugar, proporciona amplia reutilización. Al igual que los tipos de datos abstractos, se pueden utilizar diferentes implementaciones o versiones de una misma capa en la medida que soporten las mismas interfaces de cara a las capas adyacentes. Esto conduce a la posibilidad de definir interfaces de capa estándar, a partir de las cuales se pueden construir extensiones o prestaciones específicas. (11)

Sauxe presenta 5 capas o niveles:

1. Capa de Presentación: En esta capa se emplea las facilidades que brinda el Marco de Trabajo ExtJS para la construcción de interfaces amigables a la vista de los usuarios. Ext centra su desarrollo en tres componentes fundamentales JS-File, CSS-File y Client-page y Server-Page.

2. Capa de Control o Negocio: En esta capa se emplea el patrón de arquitectura Modelo Vista Controlador (MVC). De forma vertical al modelo descrito hasta este momento, estarán los aspectos fundamentales del sistema así como el encargado del tratamiento de la seguridad a nivel de aplicación Web.

3. Capa de Acceso a Datos: En esta capa estará presente el ORM (Object Relational Mapping) Doc-trine, como Marco de Trabajo de persistencia para la comunicación con el servidor de datos mediante el protocolo PDO, también estará un Persistidor de Configuración que es el encargado de comunicarse vía XML con los Ficheros de Configuración del sistema denominado FastResponse.

4. Capa de Datos: En esta capa estará ubicado como servidor de base de datos PostgreSQL y un conjunto de Ficheros de Configuración de la arquitectura tecnológica.

5. Capa de Servicios: En esta última capa se encuentran todos los subsistemas que prestan y consumen servicios entre sí.

El patrón **Modelo-Vista-Controlador (MVC)** separa el modelado del dominio, la presentación y las acciones basadas en datos ingresados por el usuario en tres clases diferentes: (11)

Modelo: El modelo administra el comportamiento y los datos del dominio de aplicación, responde a requerimientos de información sobre su estado (usualmente formulados desde la vista) y responde a instrucciones de cambiar el estado (habitualmente desde el controlador).

Vista: Maneja la visualización de la información.

Controlador: Interpreta las acciones del ratón y el teclado, informando al modelo y/o a la vista para que cambien según resulte apropiado.

Tanto la vista como el controlador dependen del modelo, el cual no depende de las otras clases.

Esta separación permite construir y probar el modelo independientemente de la representación visual.

Entre las ventajas que reporta este patrón está la adaptación al cambio ya que los requerimientos de interfaz de usuario tienden a cambiar con mayor rapidez que las reglas de negocios. Los usuarios pueden preferir distintas opciones de representación, o requerir soporte para nuevos dispositivos como teléfonos

celulares. Dado que el modelo no depende de las vistas, agregar nuevas opciones de presentación generalmente no afecta al modelo. (11)

En Sauxe el uso de este patrón se evidencia en la utilización de los diferentes marcos de trabajo utilizados. Para la Vista: Extjs-Framework, el cual es muy utilizado en el desarrollo de aplicaciones Web con tecnología AJAX, para el Controlador: Zend-Framework quien emplea específicamente el estilo Modelo- Vista- Controlador como base de su funcionamiento y para agilizar el acceso a datos en el Modelo se utilizó Doctrine, un potente y completo sistema ORM (Mapeo Objeto Relacional).

Otras Herramientas y Tecnologías

Para aquellos casos de escenarios que no encuentran solución utilizando las herramientas y tecnologías antes mencionadas, se propone tener en cuenta las características y funcionalidades de las que se mencionan a continuación:

Para la replicación de datos existen diferentes herramientas como SymmetricDS, Magic@ Data Replication eXtensible Solution, PgCluster, Pgpool-II, Slony I o **Reko**. Se recomienda usar esta última ya que es un producto multiplataforma desarrollado en la Universidad de las Ciencias Informáticas que puede ser administrado y configurado desde la web facilitando la interacción con el mecanismo de forma remota. Realiza actividades de monitoreo permitiendo conocer el estado de los datos transmitidos, puede realizarse en tiempo real a través de la web, así como dar seguimiento al funcionamiento interno del mecanismo. Esta herramienta puede detectar errores de conexión y mantener los datos de réplica en un estado estable en caso de desconexión. Al restablecerse la misma, automáticamente sincroniza los datos entre bases de datos y para el envío de los datos utiliza protocolos de comunicación seguros como HTTP y SSL. (54)

Para la generación dinámica de reportes se recomienda el uso del generador de reportes dinámicos **PADTSI** (Paquete de Herramientas para la ayuda en la Toma de Decisiones) ya que es una aplicación Web que tiene como objetivo generar reportes de forma rápida, interactiva y con una amplia gama de alternativas para los usuarios. La extensión en su uso puede estandarizar la generación de reportes en diferentes aplicaciones independientemente del Sistema Gestor de Base de Datos que utilicen ya sea MySQL, Oracle o PostgreSQL. Permite a los usuarios, entre otras opciones, abstraerse a los conocimientos relacionados con los Gestores de Bases de Datos, agilizar la toma de decisiones y generar

reportes en varios formatos (HTML, CSV, EXCEL, PDF) y con gran variedad de opciones en su diseño. El sistema está compuesto por varias aplicaciones entre las que se encuentran el Visor de reportes, el Diseñador de modelos, el Diseñador de reportes, el Diseñador de consulta y el Administrador de reportes. Ofrece una organización jerárquica de reportes, ordenados por categorías. Los mismos se pueden visualizar a través del filtrado por campos o parámetros y puede seleccionarse un modelo existente o realizar un nuevo diseño. (55)

Para el manejo de flujo de trabajo se recomienda tener en cuenta una serie de estándares como la notación desarrollada por Business Process Management Initiative (BPMI), llamada **Business Process Modeling Notation(BPMN)**. Se trata de un estándar para el modelado de procesos de negocios y servicios web necesario para coordinar y graficar la secuencia de pasos y actividades de un proceso de negocio, este modela tanto la secuencia de actividades como los datos o mensajes intercambiados entre los distintos participantes, formando diagramas de procesos de negocios (BPD). BPMN no está pensado para modelar aplicaciones, sino procesos que correrán dentro de dichas aplicaciones. Por ello, la salida de BPMN necesita ser expresada en algo que no sea un lenguaje de programación. Es aquí donde entran en juego los lenguajes de descripción de negocios (BPMLs) que permiten llevar a cabo el modelado de procesos basándose en **XML** y realizar la integración con sistemas de información y de gestión utilizando servicios web, generalmente a través de un lenguaje de descripción de servicios, que puede ser **WSDL** (Web Service Description Language). A través de estos lenguajes se dispone de la traducción de un formato gráfico (para ser leído por personas) a un formato leíble por máquinas (y permitir el intercambio entre distintas herramientas). Entre los principales BPML destacan **BPML (Business Process Modeling Language)** desarrollado por la BPMI y **BPEL4WS o WS BPEL** que en estos momentos es considerado como estándar para la ejecución y gestión de procesos. BPEL es el eslabón perdido para congregar e integrar los servicios web en procesos de negocio real y BPEL4WS estandariza la automatización de procesos entre los servicios web. (56) Puede usarse **XPDL (XML Process Definition Language)** como la forma de almacenar y permitir el intercambio de diagramas de procesos. XPDL es una representación basada en XML de un proceso, es decir es la forma de almacenar el proceso en formato físico (fichero). Como Motor de ejecución de flujos de trabajo puede usarse el **ezComponents** ya que es una librería de componentes PHP de propósito general que se puede usar junto o independiente de aplicaciones PHP en desarrollo y que proporciona a los desarrolladores una especie de capa en la forma de una máquina

virtual abstracta para la programación orientada a gráficos (GOP) con PHP. (57) Contiene un componente de workflow que proporcionando la funcionalidad necesaria para administrar, ejecutar y monitorear flujos de trabajo. Como sistema de referencia puede tomarse la plataforma de herramientas para análisis de negocios, ARIS, de la IDS Scheer AG, quien es líder del mercado en software, soluciones y servicios de gestión de procesos empresariales (BPM) y ocupa el primer lugar en el "Cuadrante Mágico para BPA (Análisis de procesos de negocios)" de Gartner. (58)

Para la generación de gráficos existen gran cantidad de librerías que proporcionan aproximadamente las mismas funcionalidades, podría usarse la librería de ExtJs, **RaphaelJs** que permite la creación de diversos tipos de gráficos bidimensionales, u otras como **FusionCharts** que destaca por el dinamismo y la animación de sus gráficos, así como su diversidad en la implementación**.** Permite la creación de gráficos y diagramas en flash, dentro de sus mejores características están la interactividad y los excelentes efectos de animación en la muestra de los datos en forma de gráfico, además de que no sólo puede ser usado en webs, sino también en aplicaciones de escritorio y presentaciones; **XML/SWF Charts** tiene al igual que el anterior un excelente diseño de presentación y diversidad de tipos. Esta librería se basa en XML para obtener los datos y en películas SWF para mostrar sus resultados en forma de gráficas, aunque si los datos son tomados de una BD es obligatoria la creación de un fichero XML intermedio entre la BD y la película de flash que muestra el gráfico. También puede tenerse en cuenta **pChart** que es una clase orientada al trabajo para la creación de gráficos y diagramas en PHP, los datos pueden ser tomados desde extructuras SQL, ficheros CVS o introducidos manualmente, los mismos se muestran creando imágenes utilizando la librería GD de PHP, que aunque no tiene un diseño muy refinado y debe tener instalado la librería GD en su servidor de PHP, aún se encuentra en desarrollo y constantemente se le hacen mejoras. Y lo más importante, si se intuye que las gráficas se van a generar muchas veces y/o los datos necesarios para su generación (datos provenientes de consultas complejas o de orígenes de datos con mucha carga) son costosos de obtener, se puede hacer uso de su caché (pCache class) para ganar en rendimiento y evitar cargas innecesarias y repetitivas en el servidor. ((59); (60); (61))

Para las alertas y avisos se recomienda usar el **Protocolo Extensible de Mensajería y Comunicación de Presencia (XMPP),** ya que es una tecnología abierta para la comunicación en tiempo real, que tiene una amplia gama de aplicaciones incluyendo mensajería instantánea, presencia chat multi-parte, llamadas de voz y video, rutas generalizadas de datos XML y otras. Es basado en XML y deja establecida una

plataforma para el intercambio de este tipo de datos que puede ser usada para la mensajería instantánea y permite hacer funcionalidades a medida para mantener la interoperabilidad. (62) Como servidor puede usarse **Openfire** ya que es un servidor de colaboración en tiempo real, multiplataforma que usa el único protocolo abierto ampliamente adoptado para la mensajería instantánea, es increíblemente fácil de instalar y administrar y ofrece una sólida seguridad y rendimiento. Entre las principales características ofrecidas por este servidor pueden mencionarse su facilidad de autenticación vía Certificados, Kerbeos, LDAP, PAM y Radius, el almacenamiento en Active Directory, LDAP, MS SQL, MySQL, Oracle y PostgreSQL, mensajes offline, comprensión de datos, transferencia de archivos, etc. Aunque opcionalmente, podría usarse el ejabberd, el psyced o el Tigase que también son libres y multiplataforma. ((63); (64); (65)) Pueden usarse además librerías como **XMPPPHP** que es una librería jabber para PHP que permite realizar acciones sobre el protocolo XMPP, lo que resulta muy útil pues se pueden enviar mensajes y alertas a los usuario, ((66); (67) y librerías como **Strophe.js**, librería XMPP para JavaScript cuyo propósito primario es permitir a las aplicaciones de tiempo real basadas en la web ejecutarse en cualquier buscador, es muy usada en sistemas de notificación. ((68); (69)) Para este mecanismo se recomienda usar los principios del estilo de arquitectura basada en eventos donde la idea dominante es que, en lugar de invocar un procedimiento en forma directa (como se haría en un estilo orientado a objetos) un componente puede anunciar mediante difusión uno o más eventos. Un componente de un sistema puede anunciar su interés en un evento determinado asociando un procedimiento con la manifestación de dicho evento. Cuando el evento se anuncia, el sistema invoca todos los procedimientos que se han registrado para él. De este modo, el anuncio de un evento implícitamente ocasiona la invocación de determinados procedimientos en otros módulos. (11)

Para la generación de ayudas puede tenerse como referencia (ya que o son herramientas privativas o no está desarrolladas para trabajar con PHP o trabajan sobre un solo sistema operativo) múltiples soluciones informáticas como DocBuilder, HTML Help Workshop, Windows Help DesignerHTML Edition o Help&Manual. (70) Existen otras herramientas libres que podrían ayudar en este sentido como los generadores de documentos: **phpDocumentor** es un generador de documentación de código abierto escrito en PHP que puede usarse desde la línea de comando o una interfaz web para crear documentación profesional del código fuente de PHP. Tiene soporte para realizar vínculos entre la documentación, incorporando documentos de nivel de usuario como tutoriales y creación de código fuente

resaltado con referencias cruzadas para documentación general PHP. Usa un sistema de plantillas extenso para cambiar los comentarios de código de fuente a un formato humano leíble y útil. Este sistema permite la creación de documentación en 15 diferentes versiones pre-diseñadas de HTML, formato PDF, formato Windows Helpfile CHM y en Docbook XML. Se trata de un proyecto de código abierto y se distribuye bajo la licencia LGPL. (71)

Para le representación geoespacial a través de mapas puede usarse o tomarse como referencia el sistema de información geográfica **Genesic,** que posee un componente para la generación de mapas temáticos. Esta es una herramienta creada en la Universidad de las Ciencias Informáticas que posee una amplia gama de funcionalidades para la representación de mapas. Es basada en tecnologías libres y permite la creación de mapas temático de coropletas, corocromático, gráficas (de pastel y barras) y símbolos proporcionales. Permite el trabajo con varios formatos tales como Esri Shape (.shp), Mapinfo (.TAB, .MIF), GML y Postgres/Postgis; siendo este último el tomado como nativo para la aplicación. Además de permitir la tematización con datos en estos formatos, posibilita la integración de datos espaciales almacenados en Postgres/Postgis con otros datos alfanuméricos que se encuentren almacenados en otra base de datos relacionados con la información espacial representada. (72)

Para la gestión documental se recomienda usar el **Alfresco** ya que es una aplicación desarrollada en Java que puede emplear multitud de sistemas gestores de bases de datos y su funcionamiento no depende de los sistemas operativos en los que se ejecute. Dispone de diferentes interfaces de acceso, que en todos los casos son intuitivas para los usuarios y en la mayoría de los casos ya conocidas. Es accesible mediante un navegador web, una carpeta de red, y/o directamente desde una aplicación de escritorio. De este modo un documento escaneado puede ser almacenado directamente en sin intervención del usuario, lo mismo ocurre con los documentos digitales. Este archivo digital velará por la confidencialidad de los documentos, y el establecimiento de la seguridad queda en manos de los responsables del archivo y de aquellos usuarios autorizados para ello, de una forma sencilla e intuitiva. De este modo los documentos y su información serán visibles cuando se dispongan de los oportunos permisos, y no serán visibles en el resto de los casos. Tanto las tecnologías empleadas para su desarrollo, como las mencionadas interfaces de acceso, y su seguridad permiten emplear Alfresco en la red local o bien en la intranet/extranet, creando así una potente fuente de conocimiento en la organización, así como una ágil herramienta para la gestión de la misma, que conlleva una mayor productividad, al

reducir los tiempos para el almacenamiento, localización, proceso y manipulación de los documentos. Pasamos de gestionar papeles a gestionar información. Su diseño y modo de funcionamiento permite un amplio abanico de usos, desde una simple estructura de carpetas dotadas de potentes herramientas de clasificación y búsqueda de información (alfresco utiliza un potente motor de búsquedas tipo Google), hasta una total personalización a los modelos y procedimientos documentales de la organización. ((73); (74))

2.4 Conclusiones parciales

- Con la identificación de los principales escenarios que debe cubrir la arquitectura tecnológica de un sistema de gestión, así como las tecnologías, herramientas y patrones que han de colaborar para solucionar los mismos, se da cumplimiento a varios de los objetivos más importantes de la investigación.
- A este proceso de selección de los escenarios según las pautas seguidas por los sistemas ERP más acertados a nivel mundial, se le atribuye un alto nivel de importancia, ya que estos como paralelos de los conocidos casos de uso para la ingeniería, determinan el camino que debe cubrir la arquitectura de cualquier sistema de gestión empresarial, y por ende, constituyen una guía para el sistema Cedrux aún en desarrollo.
- Sin embargo solamente el proceso de selección e identificación de estos escenarios y sus soluciones no asegura que la propuesta definida sea la más completa e idónea, ni se puede asegurar que con su aplicación se pueda obtener una arquitectura tecnológica robusta capaz de proveer la base que sostendrá el sistema. Es por esto que se hace necesario evaluar estos escenarios para tener seguridad de que el trabajo presentado cuanta con los componentes necesarios para llevar a cabo el proceso de desarrollo de una arquitectura para sistemas de gestión.

CAPÍTULO III: VALIDACIÓN DE LA PROPUESTA

3.1 Introducción

Para tener la seguridad que los escenarios propuestos cubren realmente la base tecnológica de un sistema de gestión empresarial es necesario realizar una evaluación de los mismos. En este capítulo se exponen los mecanismos utilizados en la investigación para validar la solución propuesta. Se presentan los elementos teóricos fundamentales para la comprensión del procedimiento, para continuar con la descripción de las evaluaciones y concluir mostrando los resultados.

3.2 ¿Por qué evaluar la arquitectura de software?

La arquitectura es el primer artefacto del ciclo de vida del desarrollo de un software, que incorpora importantes decisiones de diseño: las decisiones son fáciles de tomar, pero difíciles de cambiar una vez que el sistema se aplica. Todos los diseños arquitectónicos implican desventajas en las cualidades del sistema, ya que estas dependen en gran medida de las decisiones arquitectónicas. Por lo que garantizar la calidad del sistema es a menudo a expensas de garantizar calidad en la arquitectura y esto es posible si se realiza una evaluación de la misma. (46)

Una evaluación es un estudio de factibilidad que pretende detectar posibles riesgos y buscar recomendaciones para contenerlos. El objetivo de evaluar una arquitectura es saber si esta puede habilitar los requerimientos, atributos de calidad y restricciones para asegurar que el sistema a ser construido cumple con las necesidades de los clientes. (46)

Una mala arquitectura puede llevar a un proyecto al fracaso. Todos los requerimientos de calidad pueden quedar insatisfechos. La arquitectura también determina la estructura del proyecto: configuración, agenda y presupuesto, alcance, entre otros aspectos. Es mejor cambiar la arquitectura antes que otros artefactos, que están basados en ella, se estabilicen. Realizar una evaluación de la arquitectura es la manera más económica de evitar resultados catastróficos. (46)

3.3 ¿Cómo evaluar la arquitectura de software?

Una de las verdades más importantes sobre la arquitectura de un sistema es que conociéndola esta dirá las propiedades importantes del sistema, aún si el sistema no existe todavía. Los arquitectos toman las decisiones de diseño debido a los efectos que tendrán sobre el sistema que se está construyendo, y estos

efectos son conocidos y predecibles. Si no lo fueran, el proceso de creación de una arquitectura no sería mejor que tirar dados al aire: Se escogería una arquitectura al azar, se construiría un sistema de ella, se vería si este tiene las propiedades deseadas, y se regresa a la tabla de dibujo si no. Sin embargo se puede hacer algo más que estimación aleatoria. (9)

Una técnica eficaz de estimar si la arquitectura es correcta o no, es a través de la evaluación de los diferentes escenarios que la conforman mediante la aplicación de algún método de evaluación de arquitectura. La misma puede efectuarse en cualquier etapa del ciclo de vida de una arquitectura, aunque existen dos etapas fundamentales: tempana y tardía.

En la primera no tiene por qué estar especificada completamente la arquitectura, en la mayoría de los casos es utilizada para examinar las decisiones arquitectónicas ya tomadas y decidir entre las opciones que están pendientes. Por otro lado la evaluación tardía es realizada tanto cuando la arquitectura está terminada, como cuando la implementación está completa. Este es el caso general que se presenta en el momento de la adquisición de un sistema ya desarrollado. Se considera muy útil la evaluación del sistema en este punto, porque puede observarse el cumplimiento de los atributos de calidad asociados al sistema, y cómo será su comportamiento general. (75)

En general, una evaluación debe realizarse cuando hay suficiente de la arquitectura como para justificarla. Una buena regla sería: realizar una evaluación cuando el equipo de desarrollo empieza a tomar decisiones que dependen de la arquitectura y el costo de deshacerlas sobrepasa al costo de realizar una evaluación. (46)

3.4 Método de Análisis de Arquitectura de Software (SAAM)

La evaluación de una arquitectura mediante técnicas basadas en escenarios no produce resultados cuantitativos, sino que ayuda a encontrar debilidades y garantiza la identificación de riesgos asociados a la misma, asegurando que las decisiones de diseño arquitectónico que han sido tomadas son las correctas.

SAAM es el primer método de análisis de arquitectura basado en escenarios, más ampliamente promulgado. Este es un método capaz expresar las diferentes calidades que demandan las arquitecturas de software por medio de los escenarios y en la práctica ha demostrado ser útil para evaluar muchos atributos de calidad rápidamente e indicar los puntos débiles y fuertes de la arquitectura. (76)

3.4.1 Pasos Descritos por SAAM

SAAM define un conjunto de pasos bajo los cuales se enmarca el desarrollo de la metodología, a continuación se describen los resultados de la aplicación del proceso de análisis y evaluación basado en este método. Es necesario aclarar que dadas las condiciones y características del método no puede ser puramente empleado en la investigación, ya que el mismo establece que se deben recopilar los escenarios por el criterio emitido por involucrados externos como pueden ser el usuario final, el cliente, entre otros. Mientras que en la investigación no ha sido exactamente así, pues los mismos fueron recopilados por la información obtenida de diferentes sistemas de gestión. Otro punto de disyuntiva es el hecho de que son los arquitectos quienes se reúnen y describen los escenarios y este criterio es posteriormente evaluado, mientras que en la investigación se describieron los escenarios en base a la información obtenida de varios sistemas. Sin embargo, la esencia de evaluar escenarios se mantiene. Por lo que teniendo en cuenta que este método requiere del criterio emitido por diferentes equipos constituidos a partir del rol que desempeñan sus integrantes, ya que son estos quienes valoran y certifican la validez de los escenarios presentados, se seleccionaron tres equipos diferentes y llevaron a cabo los pasos a través del criterio recopilado en diferentes encuestas (ver Anexo 34). El primer equipo quedó constituido por aquellos involucrados internos, que van a ser todas aquellas personas vinculadas directamente a la arquitectura del sistema, o sea, los arquitectos. Un segundo equipo constituido por aquellas personas que no trabajan directamente con la arquitectura pero que de una forma u otra le competen las decisiones tomadas; en este caso particular se encuentra el equipo de tecnología, ya que la investigación se centra en los escenarios de la arquitectura tecnológica. Y el tercer y último equipo para hacer el proceso más formal está constituido por aquel personal externo con cierto nivel de experticia en el tema. A continuación (Tabla 1) se presenta como quedaron conformados los equipos:

1 Equipo de Arquitectura		2 Equipo de Tecnología		3 Equipo Externo	
1	César Lage Codorniú	1	René R. Bauta Camejo	1	Karel Gómez Velázquez
				2	Henrik Pestano Pino
2	Joisel Pérez Pérez	2	Javier Ruiz Durán	3	Jósev Pérez Rivero
				4	Alain E. Rodríguez Arias

Tabla 1 Equipos seleccionados para el proceso de análisis y evaluación de escenarios.

Paso I Desarrollo de Escenarios

El objetivo fundamental de este paso es identificar los tipos de actividades que debe soportar el sistema mediante su agrupación en diferentes escenarios. Durante el desarrollo de estos escenarios (Epígrafe 2.2) se trató de capturar todos los posibles usos del sistema, así como todos los atributos de calidad que se le atribuyen.

Paso II Descripción de la Arquitectura

En este paso se presentan (Epígrafe 2.2) los escenarios propuestos de forma clara y concisa. Se describe en lenguaje natural la representación del sistema y su comportamiento.

Paso III Clasificación y Priorización de los Escenarios

En este punto del análisis se clasifican los escenarios en Directos o Indirectos y se les da un orden de prioridad (del 1-33) donde el más significativo es el primero y el menos significativo es el último. Para ello se tuvo en cuenta el resultado arrojado por las encuestas realizadas al personal de los equipos conformados anteriormente donde cada uno otorgó una prioridad (ver Tabla 2):

ESTIMACIÓN DE PRIORIDAD										
NO.	ESCENARIOS	E1		E2		E3				PrF
Escenarios Directos										
1	Escenario de Seguridad	1	3	1	1	1	30	1	8	6
2	Escenario de Aspectos	1	4	7	22	2	25	23	5	11
3	Escenario de Caché	3	4	7	16	10	15	20	20	12
4	Escenario de Persistencia de datos	1	6	6	14	1	20	10	1	7
5	Escenario de Validaciones	4	5	8	5	1	18	2	17	8
6	Escenario de Excepciones	2	1	10	6	15	28	19	18	12
7	Escenario de Portal Frontal	4	2	2	9	1	10	22	15	8
8	Escenario de Conceptos y Atributos Dinámicos	3	5	14	17	2	20	8	12	10
9	Escenario de Instalación	6	1	9	2	1	20	15	1	7
10	Escenario de Interoperabilidad	2	1	3	15	1	30	13	7	9
11	Escenario de Concurrencia	1	5	15	3	3	17	11	19	9
12	Escenario de Transacciones	1	3	4	4	1	32	6	2	7
13	Escenario de Piscina de Conexiones	7	6	5	21	10	20	3	11	10
14	Escenario de Procesamiento en Lote	2	5	6	10	15	20	9	3	9
15	Escenario de Trazas	6	3	5	11	1	30	16	14	11
16	Escenario de Objetos a nivel de Datos	3	6	18	18	10	10	24	20	14
17	Escenario de Configuraciones	5	4	19	20	1	15	4	16	11

18	Escenario de Variables Globales	4	4	20	8	1	20	5	10	**9**
19	Escenario de Integración	6	4	10	7	1	30	12	4	**9**
20	Escenario de Mantenimiento y Operación de BD	6	6	9	23	15	15	18	29	**15**
21	Escenario de Importación y Exportación	1	2	10	13	1	10	21	13	**9**
22	Escenario de estructura y Composición	7	9	9	12	1	30	7	6	**10**
Escenarios Indirectos										
23	Escenario de Ayuda	5	2	10	26	10	15	4	27	**12**
24	Escenario de Réplica de Datos	1	4	12	25	1	30	9	9	**11**
25	Escenario de Reportes	1	2	11	28	3	34	3	23	**13**
26	Escenario de Flujos de Trabajo	1	7	10	30	1	25	15	28	**15**
27	Escenario de Gráficos	3	8	11	32	7	30	20	24	**17**
28	Escenario de Mapas	7	8	23	31	20	15	26	30	**20**
29	Escenario de Alertas y Avisos	2	5	14	27	3	20	17	24	**14**
30	Escenario de Complementos Office	3	2	25	33	15	5	4	26	**14**
31	Escenario de Impresión	4	8	26	29	25	25	19	25	**20**
32	Escenario de Gestión Documental	5	7	13	19	6	25	17	22	**14**
33	Escenario de Historiales	4	5	5	24	1	15	14	21	**11**

Tabla 2 Cálculo del índice de prioridad de los escenarios

E1: Equipo de Arquitectura

E2: Equipo de Tecnología

E3: Equipo Externo

PF: Prioridad Final

Rango de Prioridad: 1-8: **Alta**, 9-16: **Medio-Alta**, 17-25: **Medio-Baja**, 26-33: **Baja**

En este paso se evaluaron las prioridades emitidas por cada equipo y se promedió una prioridad final (PF) para cada escenario, quedando como se muestra en la tabla anterior. Teniendo en cuenta que las prioridades de todos los escenarios se encuentran entre 6-20, donde solo tres de ellos están en el rango Medio-bajo puede concluirse que los escenario propuestos tienen una prioridad Medio-Alta en la arquitectura tecnológica de sistemas de gestión empresarial.

Paso IV Evaluación Individual de los Escenarios Indirectos

En este paso se demuestra si pueden ser acomodados en la arquitectura los escenarios indirectos teniendo en cuenta el esfuerzo estimado para su implementación y otros factores, según el criterio de los equipos entrevistados.

1- Escenario de Réplica de Datos

Impacto de riesgo: Alto

Nivel de experticia de los Recursos Humanos: Alto

Costo de Desarrollo: Alto

2- Escenario de Reportes Dinámicos

Impacto de riesgo: Alto

Nivel de experticia de los Recursos Humanos: Medio

Costo de Desarrollo: Medio

3- Escenario de Flujos de Trabajo

Impacto de riesgo: Medio-Alto

Nivel de experticia de los Recursos Humanos: Alto

Costo de Desarrollo: Medio-Alto

4- Escenario de Gestión Documental

Impacto de riesgo: Medio

Nivel de experticia de los Recursos Humanos: Medio

Costo de Desarrollo: Medio

5- Escenario de Historiales

Impacto de riesgo: Medio

Nivel de experticia de los Recursos Humanos: Medio

Costo de Desarrollo: Bajo

6- Escenario de Alertas y Avisos

Impacto de riesgo: Medio

Nivel de experticia de los Recursos Humanos: Alto

Costo de Desarrollo: Medio-Alto

7- Escenario de Complementos Open Office

Impacto de riesgo: Bajo

Nivel de experticia de los Recursos Humanos: Medio-Alto

Costo de Desarrollo: Medio-Bajo

8- Escenario de Ayuda

Impacto de riesgo: Medio

Nivel de experticia de los Recursos Humanos: Medio

Costo de Desarrollo: Medio-Bajo

9- Escenario de Mapas

Impacto de riesgo: Bajo

Nivel de experticia de los Recursos Humanos: Alto

Costo de Desarrollo: Alto

10- Escenario de Gráficos

Impacto de riesgo: Medio

Nivel de experticia de los Recursos Humanos: Alto

Costo de Desarrollo: Medio

11- Escenario de Impresión

Impacto de riesgo: Medio

Nivel de experticia de los Recursos Humanos: Medio

Costo de Desarrollo: Medio

En la Tabla 3 se definen los valores generales de los factores analizados.

Factores	e1	e2	e3	e4	e5	e6	e7	e8	e9	e10	e11	General
Impacto de riesgo	A	A	M-A	M	M	M	B	M	B	M	M	**M**
Nivel de experticia	A	M	A	M	M	A	M-A	M	A	A	M	**M-A**
Costo de Desarrollo	A	M	M-A	M	B	M-A	M-B	M-B	A	M	M	**M**

Tabla 3 Determinación de los valores generales de los factores.

e*n*: escenario número *n* (donde n es un valor entre 1-11)

El impacto de riesgo de estos escenarios en la Arquitectura Tecnológica del sistema Cedrux es Medio, pues de 11, 7 tienen riesgo Medio, 2 Bajo y 2 Alto. El nivel de experticia requerido para su inclusión en la arquitectura tecnológica del sistema es Medio-Alto pues 5 escenarios requieren un nivel de experticia Alto y el costo de desarrollo estimado es Medio pues 9 de los 11 escenarios tienen costo de desarrollo Medio. Por lo que puede concluirse que la inclusión de estos escenarios en la arquitectura tecnológica del sistema Cedrux requeriría un esfuerzo Medio, ya que el impacto de riesgo en la arquitectura tecnológica del sistema es Medio, así como el costo y el nivel de experticia requeridos para su implementación.

Paso V Evaluación de la Interacción entre Escenarios

En este paso se analizó la interacción existente entre los diferentes escenarios donde se observó que casi todos interactúan con el escenario de seguridad, validaciones, portal frontal, excepciones y trazas.

Paso VI Evaluación Global

En este último paso se le asigna un peso (1-10) en la escala de importancia que tienen para la arquitectura tecnológica de un sistema de gestión empresarial. Para ello se tuvo en cuenta el resultado arrojado por las encuestas realizadas al personal de los equipos conformados donde cada uno le otorgó un peso (ver Tabla 4):

ESTIMACIÓN DE PRIORIDAD										
NO.	ESCENARIOS	E1		E2		E3				PeF
Escenarios Directos										
1	Escenario de Seguridad	10	8	10	10	10	10	1	9	9
2	Escenario de Aspectos	10	8	8	4	9	7	7	9	8
3	Escenario de Caché	8	8	8	6	5	5	4	7	6
4	Escenario de Persistencia de datos	10	7	9	7	10	5	3	10	8
5	Escenario de Validaciones	7	8	7	8	10	5	2	6	7
6	Escenario de Excepciones	9	9	7	8	4	10	5	5	7
7	Escenario de Portal Frontal	7	10	10	8	10	4	7	7	8
8	Escenario de Conceptos y Atributos Dinámicos	8	7	5	6	9	6	3	8	7
9	Escenario de Instalación	5	9	5	9	10	5	2	10	7
10	Escenario de Interoperabilidad	9	10	9	8	10	10	2	9	8
11	Escenario de Concurrencia	10	8	5	9	9	6	4	6	7
12	Escenario de Transacciones	10	7	9	9	10	10	1	10	8

13	Escenario de Piscina de Conexiones	4	6	9	6	5	4	1	8	**5**
14	Escenario de Procesamiento en Lote	9	5	9	8	4	4	4	10	**7**
15	Escenario de Trazas	5	9	7	8	10	8	5	7	**7**
16	Escenario de Objetos a nivel de Datos	8	6	6	6	4	3	10	4	**6**
17	Escenario de Configuraciones	6	7	5	5	10	4	3	7	**6**
18	Escenario de Variables Globales	7	4	4	8	10	5	1	9	**6**
19	Escenario de Integración	5	6	6	8	10	8	6	10	**7**
20	Escenario de Mantenimiento y Operación de BD	5	6	7	5	3	5	6	4	**5**
21	Escenario de Importación y Exportación	10	9	7	7	10	4	7	8	**8**
22	Escenario de estructura y Composición	4	1	7	8	10	8	4	9	**6**
Escenarios Indirectos										
23	Escenario de Ayuda	6	9	6	7	4	4	3	3	**6**
24	Escenario de Réplica de Datos	10	8	4	7	10	10	8	9	**8**
25	Escenario de Reportes	10	8	7	6	8	10	7	5	**8**
26	Escenario de Flujos de Trabajo	10	6	9	5	10	8	7	3	**7**
27	Escenario de Gráficos	8	5	7	4	3	9	6	5	**6**
28	Escenario de Mapas	4	4	4	4	1	5	3	3	**4**
29	Escenario de Alertas y Avisos	9	6	5	6	6	5	7	4	**6**
30	Escenario de Complementos Office	8	8	4	4	4	1	6	5	**5**
31	Escenario de Impresión	7	4	4	6	1	8	5	6	**5**
32	Escenario de Gestión Documental	6	7	8	6	8	8	5	4	**7**
33	Escenario de Historiales	7	6	7	5	10	3	4	4	**6**

Tabla 4 Cálculo del peso de los escenarios

Teniendo en cuenta el peso asignado puede decirse de forma general que los escenarios propuestos tienen un peso significativo en la arquitectura de un sistema de gestión empresarial ya que casi todos se encuentran por encima de la media (5), solo uno es poco significativo (4).

Si se tiene en cuenta el análisis realizado en el paso IV, donde los escenarios indirectos revelaban de forma general un esfuerzo y un impacto de riesgo medio y el análisis de este propio paso, que revela que casi todos se encuentran por encima de la media en la escala de importancia para la arquitectura de un sistema de gestión empresarial, se puede señalar que estos pueden ser incluidos en la arquitectura tecnológica del sistema Cedrux. Incluso aquellos escenarios que presentan un impacto de riesgo alto como es el escenario de réplica de datos, el de reportes y el de flujos de trabajo, pues estos tienen un peso elevado en la escala de importancia para un sistema de gestión empresarial según revela el resultado de la encuesta, con valores de 8,8 y 7 respectivamente, siendo estos los valores más altos de

todos los escenarios indirectos. El caso particular del escenario indirecto que se queda por debajo (4) de la media, indica que este no es esencial para la arquitectura tecnológica del sistema, sin embargo puede tomarse en cuenta en caso de que algún cliente tenga particularidades en su negocio que requieran ese escenario.

3.5 Conclusiones parciales

- El análisis realizado a los escenarios propuestos basado en método SAAM ha arrojado resultados significativos. La mayoría de los especialistas consultados coinciden en que los escenarios tienen una prioridad y un peso significativo en la arquitectura tecnológica de sistemas de gestión empresarial, por lo que puede decirse es válida la propuesta realizada. Con esta conclusión se da cumplimiento al objetivo principal de la investigación: identificar y describir los escenarios tecnológicos de un sistema de gestión empresarial.
- La evaluación de los escenarios indirectos revela que pueden ser incluidos en la arquitectura tecnológica del sistema Cedrux ya que su desarrollo puede ser afrontado por el equipo de tecnología pues conlleva un esfuerzo e impacto de riesgo medio, de forma general, y aquellos casos particulares donde el impacto de riesgo en la arquitectura es alto, tienen pesos significativos en la escala de importancia dentro de un sistema de gestión empresarial y los pesos más altos de los escenarios indirectos. Además de forma general casi todos tienen un peso significativo para la arquitectura de un sistema de gestión empresarial. Con lo que se resuelve la problemática de la investigación.

CONCLUSIONES

- El estudio realizado de los principales sistemas de gestión empresarial permitió identificar los escenarios que debe cubrir la arquitectura tecnológica de un sistema de este tipo. Pudiendo definir de este modo los escenarios no cubiertos por el sistema Cedrux.
- Se definieron los aspectos a describir de cada escenario y se realizó una propuesta de herramientas, tecnologías, estilos y patrones a utilizar por el sistema Cedrux y cualquier otro que reutilice estas tecnologías.
- Para asegurar la validez de la propuesta efectuada se realizó un análisis basado en el método SAAM, confeccionándose para ello una encuesta con preguntas precisas, encaminadas a determinar los aspectos requeridos para demostrar la validez de los escenarios. Dicha encuesta arrojó resultados positivos, evidenciando el cumplimiento de todos los objetivos planteados.

RECOMENDACIONES

Al concluir el presente trabajo de diploma, y luego de considerar cumplidos los objetivos trazados en el mismo, se recomienda:

- Profundizar en el estudio de los escenarios tecnológicos para el desarrollo de sistemas de gestión empresarial de manera que sea posible proponer y especificar para cada uno herramientas, tecnologías y patrones a utilizar.
- Realizar una guía que defina los roles, pasos y procesos que intervienen en el desarrollo de una arquitectura tecnológica y desarrollarla tomando como referencia los escenarios identificados.

BIBLIOGRAFÍA

1. **Clements, Paul C. and Northrop, Linda M.** Software Architecture: An Executive Overview. Pittsburgh, Pennsylvania 15213 : s.n., Febrero 1996.
2. **Camacho, Erika, Cardeso, Fabio and Núñez, Gabriel.** Arquitecturas de Software: guía de estudio. abril 2004.
3. **Perry, Dewayne E. and Wolf, Alexander L. .** Foundations for the Study of Software Architecture. *SOFTWARE ENGINEERING NOTES.* Octubre 1992. Vol. XVII, 4.
4. **Garlan, David and Shaw, Mary.** An Introduction To Software Architecture. [ed.] Ambriola V. and Tortora G. New Jersey : s.n., Junio 1994.
5. **Kazman, Rick, et al., et al.** Scenario-Based Analysis of Software Architecture. [Documento]. Octubre 29, 1995.
6. **Jacobson, Ivar, Booch, Grady and Rumbaugh, James.** *El Proceso Unificado de Desarrollo de Software.* [trans.] Salvador Sánchez, et al., et al. Madrid : Addison Wesley, Pearson Education S.A, 2000. p. 464. ISBN/84-7829-036-2.
7. **Pressman, Roger S.** *Ingeniería de Software: un enfoque práctico.* [ed.] Profesores del Departamento de Lenguajes y Sistemas Informáticos e Ingeniería del Software de la Facultad de Informática de la Universidad Pontificada de Salamanca. s.l. : Mc Graw Hill, 2001.
8. **Bachmann, Felix, et al., et al.** *Documenting Software Architecture: Views and Beyond.* Quinta. Boston : Addison Weslwy, Pearson Education Inc., 2004. ISBN/0-201-70372-6.
9. **Bass, Len, Clements, Paul and Kazman, Rick.** *Software Architecture in Practice.* Segunda. Boston : Addison Wesley, Pearson Education, Inc., 2003. p. 560. ISBN/0-321-15495-9.
10. **Kruchten, Philippe .** Planos Arquitectónicos: El Modelo de "4+1" Vistas de la Arquitectura del Software. [trans.] María Cecilia Bastarrica. [Documento]. Noviembre 1995.
11. **Reynoso, Carlos Billy and Kicillof, Nicolás.** Estilos y Patrones en la Estrategia de Arquitectonica de Microsoft. [Documento]. Universidad de Buenos Aires : s.n., Marzo 2004.
12. **The Open Group Architecture Framework.** TOGAF Version 8 Enterprise Edition. [Online] [Cited: Febrero 22, 2011.] http://www.opengroup.org/architecture/togaf8-doc/arch.
13. **Platt, Michael.** Microsoft Architecture Overview. [Online] Julio 2002. [Cited: Febrero 22, 2011.] http://msdn.microsoft.com/en-us/library/ms978007.aspx.
14. **Young, Ralph R.** *The Requirements Engineering Handbook.* Boston : ARTECH HOUSE, INC., 2004. p. 244. ISBN/1-58053-266-7.
15. **Sommerville, Ian.** Ingeniería del Software. Séptima Madrid : Pearson Education SA (Addison Wesley), 2005. p. 712. ISBN/84-7829-074-5.
16. **Lazo Ochoa, René.** Modelo de referencia para el desarrollo arquitectónico de sistemas de software en dominios de gestión. [Documento]. Ciudad de La Habana : s.n., Febrero 7, 2011.
17. **Bosch , Jan and Molin, Peter .** Software Architecture Design: Evaluation and Transformation. s.l. : Psilander Grafiska, Karlskrona, 1997. ISSN 1103-1581.
18. **Buschmann, Frank, et al., et al.** *Pattern-Oriented Software Architecture: A system of Patterns.* Primera. s.l. : JOHN WlLEY & SONS, 1996. Vol. I. ISBN/ 0-471- 95889-7.

19. **Laudon, K. and Laudon, J.** Information Systems Management: Organization and technology. 7 s.l. : Prentice Hall, 2001.
20. **López T., Marcelo.** Herramientas colaborativas utilizadasen el comercio electrónico: ERP, CRM, SCM. Septiembre 2007. Vol. II.
21. **Montalvo, Erika, Tapia, David and Plancarte Sánchez, Federico.** Administración de la función informática. [Documento (Maestría en administración de tecnologías)]. Monterrey : s.n., 2005.
22. **Santos Martín, José Ignacio and Del Olmo Martínez, Ricardo.** Adaptación de los sistemas ERP al modelo E-Business. [Documento]. Septiembre 9 y 10, 2004. VII Congreso de Ingeniería de Organizaciones Leganés.
23. [Online] http://personales.alumno.upv.es/~leatata/paquetes.htm.
24. **Alania Vilchez, James and Tapia Conozco, Wilfredo.** Scribd. *Tecnología ERP-SAP.* [Online] 2007. http://www.scribd.com/doc/923993/Tecnologia-ERPSAP.
25. **Hernández, José Antonio.** *SAP R/3 HandBook.* Segunda. s.l. : McGraw-Hill.
26. **DAA Contenidos Digitales, S.L.** CMS-SPAIN.com. *Oracle lanza en España Fusion Middleware, una solución única de base de datos, middleware y aplicaciones.* [Online] Abril 2009. http://www.ecm-spain.com/noticia.asp?IdItem=6338.
27. **Oracle.** Oracle Technology Foundation for JD Edwards EnterpriseOne. [Online] http://www.oracle.com/us/products/applications/jd-edwards-enterpriseone/tools-and-technology/053313.html.
28. —. JD Edwards EnterpriseOne. [Online] http://www.oracle.com/us/products/applications/jd-edwards-enterpriseone/index.html.
29. —. Aplicaciones PeopleSoft Enterprise. [Online] http://www.oracle.com/lad/products/applications/peoplesoft-enterprise/index.html.
30. —. PeopleSoft Enterprise PeopleTools – Tools and Technology. [Online] http://www.oracle.com/us/products/applications/peoplesoft-enterprise/tools-tech/053978.html?ssSourceSiteId=ocomlad.
31. —. PeopleSoft Applications Portal. [Online] http://www.oracle.com/us/products/applications/peoplesoft-enterprise/tools-tech/061863.html.
32. —. PeopleTools Integration & Service Oriented Architecture. [Online] http://www.oracle.com/us/products/applications/peoplesoft-enterprise/tools-tech/054003.html.
33. —. PeopleTools Administration and Infrastructure. [Online] http://www.oracle.com/us/products/applications/peoplesoft-enterprise/tools-tech/053980.html.
34. **Sage.** La tecnología de Sage ERP X3 . [Online] 2008. http://www.sageerpx3.com/sp/page/technology/.
35. —. La solución ERP todoterreno de Sage para medianas y grandes empresas. [Online] 2008. http://granempresa.sage.es/hojasproducto/sageERPX3_Folleto.pdf.
36. **Ramirez Hernández, Javier, Bermúdez Pérez, Henry Ernesto and Barrueco Barreto, Dayron Jesús.** Solución base para el desarrollo del sistema de control logistico de Cedrux. Cuidad de La Habana : s.n., Septiembre 2010.
37. **Piñera Andux, Yadira .** Formalización y estandarización de la documentación técnica de la arquitectura tecnológica del Marco de Trabajo Sauxe versión 2.0. La Habana : s.n., 2010. Universidad de las Ciencias Informáticas.

38. **Vázquez Zambrano, Donel, et al., et al.** Vista de la arquitectura de seguridad del proyecto ERP – Cuba. 2010.
39. **Pfleeger , Charles P. .** *Security in Computing.* Cuarta. s.l. : Prentice Hall, 2006. p. 880. ISBN-10: 0-13-239077-9; ISBN-13: 978-0-13-239077-4.
40. **Mondragón Sotelo, Martin R. .** Seguridad informática. Capítulo 4 Control de acceso. [Online] Julio 2006. http://mygnet.net/articulos/seguridad/763/.
41. **Gómez Baryolo, Oiner, Bauta Camejo, René Rodrigo and Díaz Peña, Omar Antonio.** Tecnologías libres para el desarrollo de aplicaciones web de gestión. [Artículo].
42. **Robaina, Laydisbel Jaime and Blanco Zamora, Yaniris.** Solución Informática para el Módulo Estructura y Composición del sistema Cedrux. La Habana : s.n., Mayo 2009. Universidad de las Ciencias informáticas.
43. **Programación en Castellano.** Programación en castellano. [Online] 2011. http://www.programacion.com/articulo/por_que_elegir_php_143.
44. **Eguíluz Pérez, Javier .** Introducción a JavaScript. *librosweb.es.* [Online] http://www.librosweb.es/javascript/capitulo1.html.
45. —. Introducción a CSS. *librosweb.es.* [Online] http://www.librosweb.es/css/capitulo1.html.
46. **Díaz Peña, Omar Antonio.** Diseño arquitectónico de una plataforma para la arquitectura distribuida en PHP basada en Servicios Web. [Trabajo de Diploma]. La Habana : s.n., Junio 2010.
47. **W3C.** W3C Interaction Domain . *XHTML2 Working Group Home Page.* [Online] 2007. http://www.w3.org/MarkUp/.
48. ENTORNO INTEGRADO DE DESARROLLO SUN FORTE FOR JAVA 3.0. [Online] http://www-gsi.dec.usc.es/~alberto/fdp/practicas/SunForte/ForteSun.pdf.
49. **Sun Campus Ambas s ador.** NetBeans 6.5 el único IDE que necesitas. [Online] http://www.techbloog.com/talks/netbeans65es_cl.pdf.
50. **ABCdatos.com.** ABCdatos. [Online] 2011. http://www.abcdatos.com/tutoriales/tutorial/v587.html.
51. **Hernández Valdés, Mayté and Alejo Ramirez, Omar .** Análisis y evaluación del diseño de una herramienta para la toma de decisiones de la vista arquitectónica de integración del proyecto ERP-Cuba. 2010.
52. **Küng, Stefan , Onken, Lübbe and Large, Simon .** TortoiseSVN Un cliente de Subversion para Win. 2006.
53. **Gómez Baryolo, Oiner, Bauta Camejo, René Rodrigo and Díaz Peña, Omar Antonio .** Tecnologías libres para el desarrollo de aplicaciones Web de gestión. [Artículo].
54. **Pimentel González, Luis Alberto, et al., et al.** Reko Replicador.
55. **Hernández Hernández, Yasmany y otros.** Manual de Usuario V. 1.5.1.
56. **Association for Computing Machinery.** ACM Digital Library. *Business Process Execution Language for Web Services BPEL and BPEL4WS 2nd Edition.* [Online] 2011. http://portal.acm.org/citation.cfm?id=1199048.
57. **Bergmann, Sebastian.** eZ Components Workflow Engine. [Online] Diciembre 06, 2006. http://sebastian-bergmann.de/archives/642-eZ-Components-Workflow-Engine.html.
58. **IDS Scheer AG.** IDS Scheer Named a Leader in the Magic Quadrant for Business Process Analysis Tools. [Online] Marzo 2010. http://www.ids-

scheer.cz/en//IDS_Scheer_Named_a_Leader_in_the_Magic_Quadrant_for_Business_Process_Analysis_Tools/168035.html?referer=38718.
59. **Jean-Damien POGOLOTTI.** pChart. [Online] 2011. http://www.pchart.net/.
60. —. pChart: a PHP class to build charts. [Online] http://pchart.sourceforge.net/.
61. pChart: a chart drawing library. [Online] http://pchart.sourceforge.net/documentation.php.
62. **Developers Challenge (with prizes!!) y otros.** XMPP.org. *About.* [Online] 2010. http://xmpp.org/about-xmpp/ y http://xmpp.org/about-xmpp/technology-overview/.
63. **Scribd Inc.** Openfire sistema de mensajería instantánea GPL. [Online] 2011. http://www.scribd.com/doc/55553397/Openfire-sistema-de-mensajeria-instantanea-GPL.
64. **SBS, Jive Software.** OpenFire. [Online] 2011. http://www.igniterealtime.org/projects/openfire/index.jsp.
65. **Developers Challenge (with prizes!!) y otros.** Servers. [Online] 2011. http://xmpp.org/xmpp-software/servers/.
66. **Google Project Hosting.** xmpphp. *XMPPHP: The PHP XMPP Library.* [Online] 2011. http://code.google.com/p/xmpphp/.
67. **Werdmuller, Ben.** XMPP: powering the real-time, really live web. [Online] Junio 2009. http://benwerd.com/2009/06/xmpp-powering-the-real-time-really-live-web/.
68. Strophe.js. *An XMPP library for JavaScript.* [Online] http://strophe.im/strophejs/.
69. Strophe. *libraries for XMPP poets.* [Online] http://strophe.im/.
70. **ArchivosPC.** Generador de archivos de ayuda. [Online] 2011. http://www.archivospc.com/c/1291/p1/Generador+de+Archivos+de+ayuda.php.
71. **Eichorn, Joshua .** phpDocumentor: The complete documentation solution for PHP. *Information, Development, and Support.* [Online] 2007. http://www.phpdoc.org/.
72. **Antunez, Romanuel Ramón and Hernández Montero, Lidisy .** Sistema para la creación de mapas temáticos (SCMP). Ciudad de La Habana : s.n., 2010.
73. **Joomla.** Gestión Documental: Alfresco. [Online] Mayo 2011. http://www.samtek.es/index2.php?option=com_content&do_pdf=1&id=38.
74. **Alfresco Software, Inc.** *Gestión Documental Alfresco.* [Online] 2011. http://www.alfresco.com/es/products/solutions/ecm/dm/.
75. **Clements, Paul, Kazman, Rick y Klein, Mark.** Evaluating Software Architectures. s.l. : Adisson Wesley, 2002.
76. **Ionita, Mugurel T. , Hammer, Dieter K. and Obbink, Henk .** Scenario-Based Software Architecture Evaluations Methods: An Overview. University Eindhoven.
77. **Ubuntu.** Ubuntu. *PgAdmin III.* [Online] Marzo 2008. http://www.guia-ubuntu.org/index.php?title=PgAdmin_III.
78. **PostgreSQL Global Development Group.** pgAdmin PostgresSQL Tools. *Latest news.* [Online] 2011. http://www.pgadmin.org/.
79. **Zend Technologies Inc.** Zendframework des.com. [Online] 2010. http://manual.zfdes.com/es/zend.cache.html.
80. **Brito González, Danelys, et al., et al.** Componente para la generación de conceptos y atributos dinámicos. [Documento]. La Habana : s.n., 2011. XVIII Fórum de Ciencia y Técnica.

81. **Christensen, Erik (Microsoft), et al., et al.** W3C. *Web Services Description Language (WSDL) 1.1.* [Online] Marzo 15, 2011. http://www.w3.org/TR/wsdl.
82. **Gamma , Erich, et al., et al.** *Design Patterns.* 1997.
83. **Bass, Len, Clements, Paul and Kazman, Rick.** *Software Architecture in Practice* . s.l. : Addison Wesley, 2003. 0-321-15495-9.
84. **Zend Technologies Ltd.** Programmer's Reference Guide. *Zend_Mail.* [Online] 2011. http://zendframework.com/manual/en/zend.mail.html.
85. **Cuenca González, Llanos and Boza García, Andrés.** Estudio comparativo de paquetes ERP. [Documento]. Valencia : s.n., Septiembre 7 y 8, 2006. X Congreso de Ingeniería de Organización.
86. **Eguíluz Pérez, Javier.** Introdución a AJAX. *librosweb.es.* [Online] http://www.librosweb.es/ajax/capitulo1.html.
87. **Simple Machine Community Forum.** Tecnohackers. *Que es un lenguaje de programacion* . [Online] 2009. http://www.tecnohackers.net/area-de-programacion/que-es-un-lenguaje-de-programacion/.
88. **Zend Technologies Ltd.** Zend Studio - the leading PHP IDE. [Online] 2010. http://www.zend.com/en/products/studio/.

ANEXOS

Descripción de las partes de los distintos escenarios tecnológicos para el desarrollo de sistemas de gestión empresarial

Anexo 1: Escenario de Seguridad

ESCENARIO DE SEGURIDAD	
PARTES DEL ESCENARIO	**POSIBLES VALORES**
FUENTE	Puede ser externa, interna y estar autorizada o no.
ESTÍMULO	Intento de ver la información, cambiar o borrar datos, acceder a los servicios/recursos del sistema o, reducir o suprimir la disponibilidad de los servicios del sistema.
ARTEFACTO	Pueden ser los servicios del sistema o la información dentro del mismo.
AMBIENTE	Pueden ser online u offline, conectado o desconectado, con firewall o sin él.
RESPUESTA	El sistema debe detectar el evento y realizar una o varias de las siguientes acciones: autenticar el usuario, bloquear el acceso a los datos y/o servicios, conceder o retirar el permiso de acceso a los datos o servicios, registrar el acceso/modificación o el intento.
MEDIDA DE LA RESPUESTA	Instantánea.
ATRIBUTOS INVOLUCRADOS: Seguridad, Integridad	
Cubierto por la arquitectura	

Anexo 2: Escenario de Aspectos

ESCENARIO DE ASPECTOS	
PARTES DEL ESCENARIO	**POSIBLES VALORES**
FUENTE	Interno: Invocación de un inyector
ESTÍMULO	Invocación directa de un método inyector (pre, pos o fallo).
ARTEFACTO	-
AMBIENTE	Entorno de aplicación.
RESPUESTA	El sistema debe invocar los elementos de los aspectos.
MEDIDA DE LA RESPUESTA	Instantánea.
ATRIBUTOS INVOLUCRADOS: Modificabilidad, Flexibilidad	
Cubierto por la arquitectura	

Anexo 3: Escenario de Caché

ESCENARIO DE CACHÉ	
PARTES DEL ESCENARIO	**POSIBLES VALORES**
FUENTE	Externa
ESTÍMULO	Solicitud de la caché.
ARTEFACTO	Caché
AMBIENTE	Entorno de exploración sistémico.
RESPUESTA	Retorno del recurso solicitado o envío de un mensaje informando la ausencia del mismo en caché.
MEDIDA DE LA RESPUESTA	Instantánea.
ATRIBUTOS INVOLUCRADOS: Accesibilidad, Disponibilidad, Rendimiento	
Cubierto por la arquitectura	

Anexo 4: Escenario de Persistencia de Datos

ESCENARIO DE PERSISTENCIA DE DATOS	
PARTES DEL ESCENARIO	**POSIBLES VALORES**
FUENTE	Disparador de inicio de configuración del marco de trabajo.
ESTÍMULO	Interacción con algún entorno de persistencia.
ARTEFACTO	ORM
AMBIENTE	Entorno de acceso a datos.
RESPUESTA	Dato persistido.
MEDIDA DE LA RESPUESTA	Instantánea.
ATRIBUTOS INVOLUCRADOS: Disponibilidad	
Cubierto por la arquitectura	

Anexo 5: Escenario de Validaciones

ESCENARIO DE VALIDACIONES	
PARTES DEL ESCENARIO	**POSIBLES VALORES**
FUENTE	Externa.
ESTÍMULO	Entrada de datos.
ARTEFACTO	Reglas de validación
AMBIENTE	Entorno de entrada de datos a la aplicación.
RESPUESTA	El sistema debe permitir la validación de los datos entrados por el usuario.

MEDIDA DE LA RESPUESTA	Instantánea.
ATRIBUTOS INVOLUCRADOS:	
Cubierto por la arquitectura	

Anexo 6: Escenario de Excepciones

ESCENARIO DE EXCEPCIONES	
PARTES DEL ESCENARIO	**POSIBLES VALORES**
FUENTE	Interna o externa
ESTÍMULO	Una falla en el sistema, ya sea un componente que falla en responder a una entrada o un componente que responde con un valor incorrecto.
ARTEFACTO	Los procesadores de sistema, los canales de comunicación, persistencia de almacenamiento, los procesos.
AMBIENTE	Normal de operación o modo degradado (menos características o una caída tras la solución)
RESPUESTA	El sistema debe detectar el evento y realizar una o varias de las siguientes acciones: registrarlo, notificar a las partes apropiadas, deshabilitar las fuentes del evento que causaron la falla de acuerdo a las reglas definidas, no estar disponible durante un intervalo de tiempo pre-especificado o continuar operando en modo normal o modo degradado.
MEDIDA DE LA RESPUESTA	Instantánea, tiempo de reparación, tiempo en modo degradado, tiempo disponible.
ATRIBUTOS INVOLUCRADOS: Tolerancia a fallos	
Cubierto por la arquitectura	

Anexo 7: Escenario de Portal Frontal

ESCENARIO DE PORTAL FRONTAL	
PARTES DEL ESCENARIO	**POSIBLES VALORES**
FUENTE	Externa
ESTÍMULO	Acceso a las funcionalidades del sistema.
ARTEFACTO	Portal frontal.
AMBIENTE	Normal de operación.
RESPUESTA	El sistema debe permitir al usuario acceder a todas sus funcionalidades.
MEDIDA DE LA RESPUESTA	Instantánea
ATRIBUTOS INVOLUCRADOS: Usabilidad	

Cubierto por la arquitectura

Anexo 8: Escenario de Metadatos

ESCENARIO DE METADATOS	
PARTES DEL ESCENARIO	**POSIBLES VALORES**
FUENTE	Externa
ESTÍMULO	Uso y explotación de escenarios dinámicos
ARTEFACTO	Entidades de datos
AMBIENTE	Negocios hiperdinámicos.
RESPUESTA	El sistema debe permitir que los conceptos identificados crezcan en atributos sin tener que modificar el código fuente o el modelo de datos.
MEDIDA DE LA RESPUESTA	Instantánea.
ATRIBUTOS INVOLUCRADOS: Modificabilidad	
Cubierto por la arquitectura	

Anexo 9: Escenario de Instalación

ESCENARIO DE INSTALACIÓN	
PARTES DEL ESCENARIO	**POSIBLES VALORES**
FUENTE	Externa
ESTÍMULO	Ejecución del instalador del sistema.
ARTEFACTO	Instalador del sistema.
AMBIENTE	Entorno de instalación del sistema con todas las precondiciones cubiertas.
RESPUESTA	El sistema debe permitir su instalación integral o por módulos permitiendo incluir actividades en el flujo de instalación inicial y debe crear la base de datos sobre la cuales se trabajará.
MEDIDA DE LA RESPUESTA	Tiempo de instalación.
ATRIBUTOS INVOLUCRADOS: Disponibilidad, Usabilidad	
Cubierto por la arquitectura	

Anexo 10: Escenario de Interoperabilidad

ESCENARIO DE INTEROPERABILIDAD	
PARTES DEL ESCENARIO	**POSIBLES VALORES**
FUENTE	Externa

ESTÍMULO	Transmisión o recibo de datos.
ARTEFACTO	Información.
AMBIENTE	Plataforma de Interoperabilidad.
RESPUESTA	El sistema debe permitir intercambiar información.
MEDIDA DE LA RESPUESTA	Instantánea
ATRIBUTOS INVOLUCRADOS: Interoperabilidad	
Cubierto por la arquitectura	

Anexo 11: Escenario de Concurrencia

ESCENARIO DE CONCURRENCIA	
PARTES DEL ESCENARIO	**POSIBLES VALORES**
FUENTE	Externa
ESTÍMULO	Intento de modificación de un recurso de forma simultánea.
ARTEFACTO	Recursos del sistema.
AMBIENTE	Entorno de gestión de datos.
RESPUESTA	El sistema debe ejecutar la solución optimista o pesimista.
MEDIDA DE LA RESPUESTA	Instantánea.
ATRIBUTOS INVOLUCRADOS: Disponibilidad, Operabilidad	
Cubierto por la arquitectura	

Anexo 12: Escenario de Transacciones

ESCENARIO DE TRANSACCIONES	
PARTES DEL ESCENARIO	**POSIBLES VALORES**
FUENTE	Externa
ESTÍMULO	Cualquier operación que se realice con las bases de datos.
ARTEFACTO	Registros de la base de datos.
AMBIENTE	Entorno de operación del sistema.
RESPUESTA	El sistema debe ejecutar la transacción y en caso de existir algún fallo notificar al cliente y dejar el sistema como se encontraba en el estado inicial (rollback).
MEDIDA DE LA RESPUESTA	Instantánea.
ATRIBUTOS INVOLUCRADOS: Tolerancia a fallos	
Cubierto por la arquitectura	

Anexo 13: Escenario de Piscina de Conexiones

ESCENARIO DE PISCINA DE CONEXIONES	
PARTES DEL ESCENARIO	**POSIBLES VALORES**
FUENTE	Externa
ESTÍMULO	Conexión con la base de datos.
ARTEFACTO	Gestor de Base de Datos
AMBIENTE	Normal de ejecución del sistema.
RESPUESTA	El sistema debe permitir solo la cantidad de conexiones que esté predeterminada, posibilitando de esta forma que el servidor no colapse.
MEDIDA DE LA RESPUESTA	Instantánea.
ATRIBUTOS INVOLUCRADOS: Rendimiento, Operabilidad	
Cubierto por la arquitectura	

Anexo 14: Escenario de Procesamiento en Lote

ESCENARIO DE PROCESAMIENTO EN LOTE	
PARTES DEL ESCENARIO	**POSIBLES VALORES**
FUENTE	Externa.
ESTÍMULO	Multitud de usuarios accediendo a la misma dirección web o varios usuarios realizando un gran número de operaciones.
ARTEFACTO	Servidor web.
AMBIENTE	Normal de operación.
RESPUESTA	El sistema debe realizar el procesamiento de forma ordenada y distribuida que evite el colapso del servidor web.
MEDIDA DE LA RESPUESTA	Instantánea.
ATRIBUTOS INVOLUCRADOS: Rendimiento, Operabilidad	
Cubierto por la arquitectura	

Anexo 15: Escenario de Trazas

ESCENARIO DE TRAZAS	
PARTES DEL ESCENARIO	**POSIBLES VALORES**
FUENTE	Externa o interna.
ESTÍMULO	Acciones de entidades sobre el sistema.
ARTEFACTO	Cualquier parte del sistema.

AMBIENTE	Normal de operación.
RESPUESTA	El sistema debe permitir seguir las acciones realizadas por cualquier entidad sobre el sistema de manera que se puedan identificar violaciones a la seguridad, ocurrencia de excepciones o cualquier otra acción.
MEDIDA DE LA RESPUESTA	Instantánea.
ATRIBUTOS INVOLUCRADOS: Integridad, Operabilidad	
Cubierto por la arquitectura	

Anexo 16: Escenario de Objetos a Nivel de Datos

ESCENARIO DE OBJETOS A NIVEL DE DATOS	
PARTES DEL ESCENARIO	**POSIBLES VALORES**
FUENTE	Externa.
ESTÍMULO	Acciones sobre la base de datos.
ARTEFACTO	Base de datos.
AMBIENTE	Normal de operación.
RESPUESTA	El sistema debe permitir a los usuarios realizar operaciones con las bases de datos desde el nivel de presentación, sin tener que interactuar con ningún gestor específico.
MEDIDA DE LA RESPUESTA	Instantánea.
ATRIBUTOS INVOLUCRADOS: Usabilidad, Operabilidad	
Cubierto por la arquitectura	

Anexo 17: Escenario de Configuraciones

ESCENARIO DE CONFIGURACIONES	
PARTES DEL ESCENARIO	**POSIBLES VALORES**
FUENTE	Externa.
ESTÍMULO	Acción de configuración del sistema.
ARTEFACTO	Diferentes partes del sistema.
AMBIENTE	Normal de operación.
RESPUESTA	El sistema debe proveer algún mecanismo que permita manejar de forma centralizada las configuraciones de las diferentes aplicaciones.
MEDIDA DE LA RESPUESTA	Instantánea.
ATRIBUTOS INVOLUCRADOS: Modificabilidad, Operabilidad, Flexibilidad	

Cubierto por la arquitectura

Anexo 18: Escenario de Variables Globales

ESCENARIO DE VARIABLES GLOBALES	
PARTES DEL ESCENARIO	**POSIBLES VALORES**
FUENTE	Interna.
ESTÍMULO	Acceso a información común que ofertan otras partes del sistema.
ARTEFACTO	Almacén de datos globales.
AMBIENTE	Normal de operación.
RESPUESTA	El sistema debe permitir que cada parte del sistema pueda acceder de manera fácil y sencilla a datos que oferte otra sin incurrir en códigos engorrosos.
MEDIDA DE LA RESPUESTA	Instantánea.
ATRIBUTOS INVOLUCRADOS: Accesibilidad	
Cubierto por la arquitectura	

Anexo 19: Escenario de Inyegración

ESCENARIO DE INTEGRACIÓN	
PARTES DEL ESCENARIO	**POSIBLES VALORES**
FUENTE	Interna o Externa.
ESTÍMULO	Trasmisión o recibo de datos.
ARTEFACTO	Componentes, subsistemas y sistemas.
AMBIENTE	Entorno de integración.
RESPUESTA	El sistema debe permitir que dos o más aplicaciones se fusionen para responder a una demanda que por sí solas no era posible.
MEDIDA DE LA RESPUESTA	Instantánea.
ATRIBUTOS INVOLUCRADOS: Integrabilidad	
Cubierto por la arquitectura	

Anexo 20: Escenario de Mantenimiento y Operación

ESCENARIO DE MANTENIMIENTO Y OPERACIÓN	
PARTES DEL ESCENARIO	**POSIBLES VALORES**
FUENTE	Externa.
ESTÍMULO	Mantenimiento u operación de las bases de datos.

ARTEFACTO	Bases de datos
AMBIENTE	Entorno de datos.
RESPUESTA	El sistema debe permitir a los usuarios realizar operaciones CRUD sobre la bases de datos, así como monitoreo, mantenimiento y administración de las mismas.
MEDIDA DE LA RESPUESTA	Instantánea.
ATRIBUTOS INVOLUCRADOS: Integridad, Mantenibilidad	
Cubierto por la arquitectura	

Anexo 21: Escenario de Exportar e Importar Datos

ESCENARIO DE EXPORTAR E IMPORTAR DATOS	
PARTES DEL ESCENARIO	**POSIBLES VALORES**
FUENTE	Externa
ESTÍMULO	Solicitud de importar o exportar información
ARTEFACTO	Archivo
AMBIENTE	Normal de operación.
RESPUESTA	El sistema debe permitir a los usuarios la importación y exportación de datos.
MEDIDA DE LA RESPUESTA	Instantánea.
ATRIBUTOS INVOLUCRADOS: Usabilidad	
Cubierto por la arquitectura	

Anexo 22: Escenario de Estructura y Composición

ESCENARIO DE ESTRUCTURA Y COMPOSICIÓN	
PARTES DEL ESCENARIO	**POSIBLES VALORES**
FUENTE	Externa
ESTÍMULO	Conformar la estructura de la organización
ARTEFACTO	-
AMBIENTE	Normal de operación.
RESPUESTA	El sistema debe permitir crear una estructura de entidades que permita establecer el nivel jerárquico que compone la organización.
MEDIDA DE LA RESPUESTA	Instantánea.
ATRIBUTOS INVOLUCRADOS: Usabilidad	

Cubierto por la arquitectura

Anexo 23: Escenario de Ayuda

ESCENARIO DE AYUDA	
PARTES DEL ESCENARIO	**POSIBLES VALORES**
FUENTE	Externa
ESTÍMULO	Acceso a la información referente al funcionamiento de cada una de las partes que conforman el sistema.
ARTEFACTO	Ayuda
AMBIENTE	Normal de operación.
RESPUESTA	El sistema debe permitir crear sistemas de ayuda y documentación profesional que orienten a los usuarios en el uso del mismo.
MEDIDA DE LA RESPUESTA	Instantánea.
ATRIBUTOS INVOLUCRADOS: Cognoscibiilidad, Usabilidad, Comprensibilidad	
No cubierto por la arquitectura	

Anexo 24: Escenario de Réplica de Datos

ESCENARIO DE RÉPLICA DE DATOS	
PARTES DEL ESCENARIO	**POSIBLES VALORES**
FUENTE	Externa o interna.
ESTÍMULO	Operación sobre una base de datos.
ARTEFACTO	Bases de Datos.
AMBIENTE	Con conexión o sin conexión.
RESPUESTA	El sistema debe permitir que los cambios realizados a un objeto se propaguen al resto de las bases de datos para evitar inconsistencia en los mismos.
MEDIDA DE LA RESPUESTA	Instantánea, tiempo que demore la réplica.
ATRIBUTOS INVOLUCRADOS: Tolerancia a fallos, Disponibilidad	
No cubierto por la arquitectura	

Anexo 25: Escenario de Reportes

ESCENARIO DE REPORTES	
PARTES DEL ESCENARIO	**POSIBLES VALORES**
FUENTE	Externa
ESTÍMULO	Puede ser crear/eliminar/modificar/importar/buscar/diseñar un

	reporte.
ARTEFACTO	Reporte
AMBIENTE	Normal de operación.
RESPUESTA	El sistema debe permitir crear/eliminar/modificar/importar/buscar /diseñar un reporte.
MEDIDA DE LA RESPUESTA	Instantánea, tiempo que se demora en importar un reporte.
ATRIBUTOS INVOLUCRADOS: Operabilidad	
No cubierto por la arquitectura	

Anexo 26: Escenario de Flujos de Trabajo

ESCENARIO DE FLUJOS DE TRABAJO	
PARTES DEL ESCENARIO	**POSIBLES VALORES**
FUENTE	Externa
ESTÍMULO	Puede ser modelar/modificar/eliminar algún flujo de trabajo.
ARTEFACTO	Diagrama de Flujo de Trabajo.
AMBIENTE	Normal de operación.
RESPUESTA	El sistema debe permitir modelar/modificar/eliminar algún flujo de trabajo sin incurrir en cambios globales en el sistema.
MEDIDA DE LA RESPUESTA	Instantánea.
ATRIBUTOS INVOLUCRADOS: Operabilidad, Modificabilidad	
No cubierto por la arquitectura	

Anexo 27: Escenario de Gráficos

ESCENARIO DE GRÁFICOS	
PARTES DEL ESCENARIO	**POSIBLES VALORES**
FUENTE	Externa
ESTÍMULO	Solicitud de generación de gráficos para analizar datos.
ARTEFACTO	-
AMBIENTE	Normal de operación.
RESPUESTA	El sistema debe permitir la generación de gráficos para la visualización de información.
MEDIDA DE LA RESPUESTA	Instantánea.
ATRIBUTOS INVOLUCRADOS: Usabilidad	
No cubierto por la arquitectura	

Anexo 28: Escenario de Mapas

ESCENARIO DE MAPAS	
PARTES DEL ESCENARIO	**POSIBLES VALORES**
FUENTE	Externa
ESTÍMULO	Solicitud de generación de un mapa.
ARTEFACTO	-
AMBIENTE	Normal de operación.
RESPUESTA	El sistema debe permitir la generación de mapas temáticos que permitan al usuario el análisis de diferentes datos de manera geoespacial.
MEDIDA DE LA RESPUESTA	Instantánea.
ATRIBUTOS INVOLUCRADOS: Usabilidad	
No cubierto por la arquitectura	

Anexo 29: Escenario de Alertas y Avisos

ESCENARIO DE ALERTAS Y AVISOS	
PARTES DEL ESCENARIO	**POSIBLES VALORES**
FUENTE	Externa o Interna
ESTÍMULO	Subscripción de componentes/subsistemas/usuario a eventos que disparan otros componentes/subsistemas.
ARTEFACTO	Componentes/subsistemas/sistemas implicados.
AMBIENTE	Normal de operación.
RESPUESTA	El sistema debe permitir la subscripción de componentes/subsistemas a eventos que disparan otros componentes/subsistemas y que una vez que el evento es disparado debe avisar del hecho a los elementos subscritos.
MEDIDA DE LA RESPUESTA	Tiempo que demore en disparase el evento.
ATRIBUTOS INVOLUCRADOS: Usabilidad	
No cubierto por la arquitectura	

Anexo 30: Escenario de Complementos Open Office

ESCENARIO DE COMPLEMENTOS OPEN OFFICE	
PARTES DEL ESCENARIO	**POSIBLES VALORES**
FUENTE	Externa

ESTÍMULO	Selección de una función tipo de los módulos del sistema.
ARTEFACTO	Hoja de Cálculo de Open Office.
AMBIENTE	Hoja de Cálculo de Open Office.
RESPUESTA	El sistema debe permitir que una gama de funciones tipo de los módulos contables y financieros mediante la interacción con la hoja de cálculo del paquete de ofimática de Open Office utilicen sus facilidades de diseño.
MEDIDA DE LA RESPUESTA	Instantánea.
ATRIBUTOS INVOLUCRADOS: Interoperabilidad	
No cubierto por la arquitectura	

Anexo 31: Escenario de Gestión de Historiales

ESCENARIO DE GESTIÓN HISTORIALES	
PARTES DEL ESCENARIO	**POSIBLES VALORES**
FUENTE	Externa
ESTÍMULO	Acciones de usuarios sobre los sistemas.
ARTEFACTO	-
AMBIENTE	Normal de operación.
RESPUESTA	El sistema debe permitir conocer el rastro de las acciones que un usuario realiza sobre el sistema, los lugares visitos, etc. Debe permitir determinar a qué subsistemas se le gestionará esta información y que datos serán registrados en la base de datos.
MEDIDA DE LA RESPUESTA	Instantánea
ATRIBUTOS INVOLUCRADOS: Modificabilidad, Accesibilidad	
No cubierto por la arquitectura	

Anexo 32: Escenario de Gestión Documental

ESCENARIO DE GESTIÓN DOCUMENTAL	
PARTES DEL ESCENARIO	**POSIBLES VALORES**
FUENTE	Externa
ESTÍMULO	Puede ser búsqueda/almacenamiento/procesamiento/eliminación o distribución de documentos.
ARTEFACTO	-
AMBIENTE	Normal de operación.
RESPUESTA	El sistema debe guardar/modificar/almacenar/buscar/procesar /recuperar o distribuir documentos.

MEDIDA DE LA RESPUESTA	Asincrónica
ATRIBUTOS INVOLUCRADOS: Modificabilidad, Accesibilidad	
No cubierto por la arquitectura	

Anexo 33: Escenario de impresión

ESCENARIO DE IMPRESIÓN	
PARTES DEL ESCENARIO	**POSIBLES VALORES**
FUENTE	Externa
ESTÍMULO	Petición de impresión
ARTEFACTO	-
AMBIENTE	Normal de operación
RESPUESTA	El sistema debe permitir al usuario monitorizar y manejar las tareas de impresión.
MEDIDA DE LA RESPUESTA	Instantánea.
ATRIBUTOS INVOLUCRADOS: Interoperabilidad	
No cubierto por la arquitectura	

Encuesta aplicada para el desarrollo del método de SAAM

Anexo 34: Encuesta aplicada para el desarrollo del método SAAM

ENCUESTA PARA VALIDACIÓN DE ESCENARIOS DE LA ARQUITECTURA TECNOLÓGICA DE UN SISTEMA DE GESTIÓN EMPRESARIAL.

Cargo:
Centro:

Basado en su experiencia y criterio personal responda las siguientes preguntas de los escenarios arquitectónico-tecnológicos que se presentan a continuación:

a) Asigne un **orden de prioridad** (1-34) a los escenarios según la importancia que le concede dentro de la arquitectura tecnológica de un sistema ERP. (Donde 1 es la máxima prioridad y 34 la mínima). (Prioridad:_____)
b) Asigne un **peso** (1-10) en la escala de importancia que tienen los escenarios propuestos para el éxito de un sistema de gestión empresarial. (Peso:____)
c) ¿Qué impacto tendría la inclusión de los escenarios indirectos (son aquellos escenarios que en mayor o menor medida requieren cambios en la arquitectura) en la arquitectura del sistema Cedrux en cuanto a los riesgos? (Medio, Alto, Bajo)
d) Estime el **tiempo** y la **cantidad de recursos humanos** que requiere la implementación de cada escenario indirecto, así como el nivel de experticia de estos últimos. (Meses: _____, Recursos Humanos:_____ y Nivel

de experticia: Alto_____ Medio_____ Bajo_____)

e) Estime el **costo de desarrollo** de cada escenario indirecto. (Alto, Medio, Bajo)

f) ¿Considera que la **identificación proactiva** de los escenarios indirectos **contribuye a agilizar** el **trabajo del equipo de arquitectura** del sistema Cedrux?
Si____ No____ Parcialmente____

g) **Clasifique la guía** propuesta para construir arquitecturas de referencia para sistemas de gestión empresarial de acuerdo a la medida en que satisface los escenarios de un sistema de gestión empresarial en Excelente, Muy buena, Buena, Aceptable, Cuestionable, Mala, Pésima.

Excelente___ Muy buena___ Buena___ Aceptable___ Cuestionable___ Mala, Pésima___

Escenarios Directos

1- La arquitectura tecnológica debe proveer algún mecanismo de autenticación, autorización y auditoría, de forma que se detecte cualquier acción indebida de un usuario, proceso o sistema que y que se mantengan protegidos los datos contra intentos de acceso sin previa autenticación o permisos requeridos.

- Prioridad:_____ Peso:_____

2- La arquitectura tecnológica debe proveer alguna solución para la configuración dinámica y centralizada de aspectos, permitiendo activar o desactivar aspectos como la seguridad, las trazas, los historiales, las validaciones u otros que pueden o no ser usados en el desarrollo o ejecución de un sistema.

- Prioridad:_____ Peso:_____

3- La arquitectura tecnológica debe proveer algún mecanismo que permita gestionar de manera dinámica la caché para facilitar el manejo y accesibilidad a información importante de forma rápida.

- Prioridad:_____ Peso:_____

4- La arquitectura tecnológica de la plataforma debe proveer un mecanismo de abstracción de la capa relacional de persistencia.

- Prioridad:_____ Peso:_____

5-La arquitectura tecnológica debe proveer algún mecanismo de gestión, configuración y administración de validaciones para sucesos del sistema.

- Prioridad:_____ Peso:_____

6- La arquitectura tecnológica debe proveer algún mecanismo de gestión, configuración y administración de excepciones de manera dinámica y declarativa.

- Prioridad:_____ Peso:_____

7- La arquitectura tecnológica debe proveer algún mecanismo que brinde la posibilidad de integrar y visualizar todas las representaciones o respuestas del sistema que naveguen hasta la capa cliente de la arquitectura en un área de trabajo única, taxonómicamente ordenada.

- Prioridad:_____ Peso:_____

8- La arquitectura tecnológica debe proveer algún mecanismo para la gestión de conceptos y atributos dinámicos, de manera que el usuario defina los conceptos que se ajusten a sus características, permitiendo que estos crezcan en atributos sin tener que modificar el código fuente o el modelo de datos.

- Prioridad:_____ Peso:_____

9- La arquitectura tecnológica debe proveer algún mecanismo que le permita la instalación integral del o

por módulos teniendo en cuenta la dependencia entre los mismos, permitiendo insertar tareas en el flujo de instalación para la configuración inicial. Además debe permitir que el sistema se pueda actualizar a partir de y hasta una versión dada, así como dar la posibilidad de instalar en entornos donde la app y la base de datos puedan encontrarse en un mismo servidor o no.

- Prioridad:_____ Peso:_____

10- La Arquitectura tecnológica debe proveer algún mecanismo que permita comunicarse con otros sistemas externos para transmitir y recibir información, notificar o gestionar sucesos, así como para exportar e importar datos.

- Prioridad:_____ Peso:_____

11- La arquitectura tecnológica debe proveer algún mecanismo de configuración y gestión dinámica de las características de concurrencia sobre entidades o dominios de la solución lógica que se modela, permitiendo implementar el mecanismo pesimista u optimista.

- Prioridad:_____ Peso:_____

12- La arquitectura tecnológica de la plataforma debe proveer un mecanismo de administración de transacciones, de manera que las operaciones de modificación sobre el modelo de dominio sean transaccionales por defecto. Y que ante algún error en los datos se pueda traer de vuelta la base de datos al tiempo y estado en que se encontraba (en estado consistente) antes de que el daño se causara.

- Prioridad:_____ Peso:_____

13- La arquitectura tecnológica debe proveer algún mecanismo de administración y configuración de la piscina de conexiones con los gestores más utilizados.

- Prioridad:_____ Peso:_____

14- La arquitectura tecnológica debe proveer algún mecanismo de gestión, configuración y administración de procesamiento en lote para evitar que el servidor colapse.

- Prioridad:_____ Peso:_____

15- La arquitectura tecnológica de la plataforma debe proveer algún mecanismo de administración y configuración dinámica de trazas de la solución.

- Prioridad:_____ Peso:_____

16-La arquitectura tecnológica de la plataforma debe proveer algún mecanismo para la gestión de objetos a nivel de datos (bases de datos, esquemas, secuencias, disparadores, funciones, roles, tablas, atributos, etc.) desde la capa de aplicación, sin necesidad de que el usuario interactúe directamente con un cliente de algún gestor en específico.

- Prioridad:_____ Peso:_____

17- La arquitectura tecnológica de la plataforma debe proveer un mecanismo para la configuración dinámica y centralizada de las configuraciones de las aplicaciones que se instancien sobre la plataforma tecnológica.

- Prioridad:_____ Peso:_____

18- La arquitectura tecnológica de la plataforma debe proveer algún mecanismo para la configuración dinámica y centralizada variables globales que permita agrupar información común conceptualizada que se maneja en todo el sistema.

- Prioridad:_____ Peso:_____

19- La arquitectura tecnológica debe proveer la capacidad para configurar y gestionar de manera dinámica

la integración de las aplicaciones o dominios de soluciones que se instancien o construyan con la misma.

- Prioridad:_____ Peso:_____

20- La arquitectura tecnológica debe proveer algún mecanismo que permita el monitoreo, mantenimiento y administración de base de datos.

- Prioridad:_____ Peso:_____

21- La arquitectura tecnológica de la plataforma debe proveer algún mecanismo que permita importar y exportar datos en diferentes formatos.

- Prioridad:_____ Peso:_____

22- La arquitectura tecnológica debe proveer algún mecanismo que brinde la posibilidad al usuario de definir la estructura organizativa en la cual se va a ubicar su entidad y la estructura dentro de dicha entidad, así como permitir que se puedan especificar los cargos por la cuales van a estar compuestas las diferentes áreas dentro de las unidades y ser configurable para que los usuarios no estén obligados a usar los mismos conceptos estructurales.

- Prioridad:_____ Peso:_____

Escenarios Indirectos

23- La arquitectura tecnológica debe proveer algún mecanismo generador de ayuda para brindar la información referente al funcionamiento de cada una de las partes que conforman el sistema y facilitar y mejorar el trabajo de los usuarios con el mismo.

- Prioridad:_____ Peso:_____
- Impacto de Riesgo: Alto_____ Medio_____ Bajo_____
- Meses: _____, R.Humanos:_____ y Nivel de experticia: Alto_____ Medio_____ Bajo_____
- Costo de desarrollo: Alto_____ Medio_____ Bajo_____

24- La Arquitectura tecnológica debe proveer algún mecanismo que permita compartir datos entre bases de datos en diversas localizaciones donde las modificaciones que se hagan en una BD sean actualizadas en las demás, de forma bidireccional, con o sin conexión.

- Prioridad:_____ Peso:_____
- Impacto de Riesgo: Alto_____ Medio_____ Bajo_____
- Meses: _____, R.Humanos:_____ y Nivel de experticia: Alto_____ Medio_____ Bajo_____
- Costo de desarrollo: Alto_____ Medio_____ Bajo_____

25- La arquitectura tecnológica debe proveer algún mecanismo que permita de forma dinámica la generación, administración, configuración y visualización de reportes.

- Prioridad:_____ Peso:_____
- Impacto de Riesgo: Alto_____ Medio_____ Bajo_____
- Meses: _____, R.Humanos:_____ y Nivel de experticia: Alto_____ Medio_____ Bajo_____
- Costo de desarrollo: Alto_____ Medio_____ Bajo_____

26- La arquitectura tecnológica debe proveer algún mecanismo de gestión, configuración y administración para flujos de trabajo de forma dinámica y declarativa.

- Prioridad:_____ Peso:_____
- Impacto de Riesgo: Alto_____ Medio_____ Bajo_____
- Meses: _____, R.Humanos:_____ y Nivel de experticia: Alto_____ Medio_____ Bajo_____
- Costo de desarrollo: Alto_____ Medio_____ Bajo_____

27- La arquitectura tecnológica debe proveer algún mecanismo que permita la generación de diferentes tipos de gráficos para realizar análisis estadístico.

- Prioridad:_____ Peso:_____
- Impacto de Riesgo: Alto_____ Medio_____ Bajo_____
- Meses: _____, R.Humanos:_____ y Nivel de experticia: Alto_____ Medio_____ Bajo_____
- Costo de desarrollo: Alto_____ Medio_____ Bajo_____

28- La arquitectura tecnológica debe proveer algún mecanismo que permita la generación de mapas para la representación geoespacial de la información.

- Prioridad:_____ Peso:_____
- Impacto de Riesgo: Alto_____ Medio_____ Bajo_____
- Meses: _____, R.Humanos:_____ y Nivel de experticia: Alto_____ Medio_____ Bajo_____
- Costo de desarrollo: Alto_____ Medio_____ Bajo_____

29- La arquitectura tecnológica debe proveer algún mecanismo de alertas y avisos que permita la subscripción de los componentes a eventos disparados por otros componentes de manera que se envíe una alerta del momento en que cierto servicio o recurso ha sido liberado y puede comenzar a usarse. Estas alertas pueden ser enviadas por diferentes vías: telefónica, mensajería instantánea, correo, bíper o por el mismo sistema.

- Prioridad:_____ Peso:_____
- Impacto de Riesgo: Alto_____ Medio_____ Bajo_____
- Meses: _____, R.Humanos:_____ y Nivel de experticia: Alto_____ Medio_____ Bajo_____
- Costo de desarrollo: Alto_____ Medio_____ Bajo_____

30- La arquitectura tecnológica debe proveer algún mecanismo que permita integrar funciones tipo del ERP con las funcionalidades brindadas por el Office, de manera que los usuarios puedan utilizar las facilidades brindadas por la hoja de cálculo para el trabajo con funciones tipo de los módulos contables y financieros.

- Prioridad:_____ Peso:_____
- Impacto de Riesgo: Alto_____ Medio_____ Bajo_____
- Meses: _____, R.Humanos:_____ y Nivel de experticia: Alto_____ Medio_____ Bajo_____
- Costo de desarrollo: Alto_____ Medio_____ Bajo_____

31- La arquitectura tecnológica de la plataforma debe proveer un mecanismo para la configuración dinámica y centralizada de la gestión de historiales.

- Prioridad:_____ Peso:_____
- Impacto de Riesgo: Alto_____ Medio_____ Bajo_____
- Meses: _____, R.Humanos:_____ y Nivel de experticia: Alto_____ Medio_____ Bajo_____
- Costo de desarrollo: Alto_____ Medio_____ Bajo_____

32- La arquitectura tecnológica debe proveer algún mecanismo permita la gestión documental, llevando a cabo el procesado, almacenamiento, búsqueda, recuperación y distribución de documentos al conjunto de usuarios que operen sobre el mismo.

- Prioridad:_____ Peso:_____
- Impacto de Riesgo: Alto_____ Medio_____ Bajo_____
- Meses: _____, R.Humanos:_____ y Nivel de experticia: Alto_____ Medio_____ Bajo_____

- Costo de desarrollo: Alto_____ Medio_____ Bajo_____

33- La arquitectura tecnológica debe proveer algún mecanismo que permita la configuración administración y monitorización de las actividades de impresión, de forma que un usuario pueda programar las actividades de impresión (tiempo de comienzo, impresora que realizará el trabajo) y continuar trabajando en otras labores, pudiendo desactivar o monitorizar el servicio de impresión cuando desee.

- Prioridad:_____ Peso:_____
- Impacto de Riesgo: Alto_____ Medio_____ Bajo_____
- Meses: _____, R.Humanos:_____ y Nivel de experticia: Alto_____ Medio_____ Bajo_____
- Costo de desarrollo: Alto_____ Medio_____ Bajo_____

Si considera que deba ser agregado algún otro escenario puede hacer su propuesta a continuación:

_____
_____
_____
_____
_____
_____
_____
_____

_____ _____

Nombre y Apellidos Firma

GLOSARIO

API (Application Programing Interface)

Interfaz generalmente especificada como un conjunto de operaciones, la cual es definida por un programa de aplicación que permite el acceso a las funcionalidades del programa. Esto significa que esta funcionalidad puede ser llamada directamente por otros programas y no solamente accedida a través de la interfaz de usuario.

Bases de Datos

Data Bases. Colección de información, tablas y otros objetos organizados y presentados para un propósito específico como, por ejemplo, búsqueda, clasificación y nueva combinación de datos. Las bases de datos se almacenan en archivos

Componentes

Un componente es una pieza de software con una interfaz o un conjunto de interfaces bien definido que proporciona el acceso a sus servicios. Los componentes sirven como bloques de construcción para la estructura de un sistema y han de poder ser desarrollados, adquiridos, incorporados al sistema y compuestos con otros componentes de forma independiente, en tiempo y espacio. A nivel de lenguajes de programación, pueden ser representados como módulos, clases, objetos o conjunto de funciones relacionadas. Un componente que no implementa todos los elementos de su Interfaz es llamado componente abstracto. Un ejemplo de componente pueden ser objetos, procesos o colecciones de estos.

Disparador o Trigger

Un trigger o disparador en una Base de datos, es un procedimiento que se ejecuta cuando se cumple una condición establecida al realizar una operación de inserción, actualización o borrado.

DOM

Document Object Model (Modelo en Objetos para la representación de Documentos), provee un conjunto estándar de objetos para representar documentos HTML y XML, permitiendo su combinación y ofreciendo una interfaz estándar para el acceso y manipulación.

Framework

Es una estructura de soporte definida, representa una arquitectura de software que modela las relaciones generales de las entidades del dominio. Provee una estructura y una metodología de trabajo la cual extiende o utiliza las aplicaciones del dominio.

Grid Computing

Grid computing da la idea de una gran potencia de cálculo y almacenamiento. Un grid computacional es una infraestructura hardware y software que suministra al que la utiliza acceso seguro y barato, a unas elevadas capacidades computacionales. El concepto infraestructura se utiliza porque un grid es un conjunto de recursos (ciclos de CPU, datos, sensores, etc.), y todos esos recursos necesitan una interconexión hardware y un control software para que estén ensambladas en un grid. Esta infraestructura debe proporcionar a los usuarios un servicio seguro a todos los niveles: capacidad de cómputo, de integridad de datos, seguridad de acceso, etc.

GPL

General Public License es una licencia pública general de GNU

Hardware

Corresponde a todas las partes físicas y tangibles de una computadora, sus componentes eléctricos, electrónicos, electromecánicos y mecánicos. El término se ha extendido a cualquier sistema que tenga una parte programable (contenido en algún tipo de memoria, no necesariamente eléctrica), para diferenciar esa parte intangible (el programa) de la parte tangible

Infraestructura Tecnológica

Se encuentra integrada por un conjunto de elementos de hardware (servidores, puestos de trabajo, redes, enlaces de telecomunicaciones, etc.), software (sistemas operativos, bases de datos, lenguajes de programación, herramientas de administración, etc.) y servicios (soporte técnico, seguros, comunicaciones, etc.) que en un conjunto dan soporte a las aplicaciones (sistemas informáticos) de una empresa.

Interfaz

Especificación de atributos y operaciones asociadas con un componente de software. La interfaz es usada como medio de acceso a acceder.

Layouts

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

LDAP

Siglas del inglés Lightweight Directory Access Protocol, en español Protocolo Ligero de Acceso a Directorios, hacen referencia a un protocolo a nivel de aplicación el cual permite el acceso a un servicio de directorio ordenado y distribuido para buscar diversa información en un entorno de red.

Middleware

Es un software de conectividad que permite ofrecer un conjunto de servicios que hacen posible el funcionamiento de aplicaciones distribuidas sobre plataformas heterogéneas. Funciona como una capa de abstracción de software distribuida que se sitúa en la capa de aplicaciones y las capas inferiores (sistema operativo y red). El Middleware sirve para la integración de aplicaciones y para soportar la integración de procesos.

Objeto mapeador relacional (ORM)

Es una técnica de programación para convertir datos entre el sistema de tipos utilizado en un lenguaje de programación orientado a objetos y el utilizado en una base de datos relacional.

Plataforma

En informática, es el principio en el cual se constituye un hardware, sobre el cual un software puede ejecutarse/desarrollarse. La plataforma define un estándar alrededor del cual un sistema puede ser desarrollado. Una vez que la plataforma ha sido definida, los desarrolladores de software pueden producir el software apropiado y los gerentes pueden adquirir el hardware apropiado para su uso

Publicación/Subscripción

Este es un paradigma de mensajería asíncrona donde los remitentes (publicadores que envían) no están programados para enviar sus mensajes a receptores específicos (suscriptores). Por el contrario, los mensajes publicados se organizan en especies de clases, sin conocimiento de quiénes puedan ser los suscriptores. Éstos últimos expresan el interés por una o más clases y sólo reciben los mensajes, sin conocimiento de cuáles son todos los publicadores.

Return Of Investment (ROI)

ROI son las siglas en inglés de Return On Investment y es una medida de rendimiento usada para evaluar la eficiencia de una inversión o para comparar la eficiencia de un número de inversiones. Se calcula a través de la división de los beneficios (retorno) de una inversión divido por el costo y el resultado se expresa como un porcentaje o una proporción.

Rollback

Operación que ante la ocurrencia de algún error devuelve a la base de datos a algún estado previo.

Servidor de aplicaciones

Se trata de un dispositivo de software que proporciona servicios de aplicación a las computadoras cliente. Generalmente gestiona la mayor parte de las funciones de lógica de negocio y de acceso a los datos de la

aplicación. Los principales beneficios son la centralización y la disminución de la complejidad en el desarrollo de aplicaciones

Servidor de Bases de Datos

Database Server. Un servidor de base de datos es un programa que provee servicios de base de datos a otros programas u otras computadoras, como es definido por el modelo cliente-servidor. También puede hacer referencia a aquellas computadoras (servidores) dedicadas a ejecutar esos programas, prestando el servicio.

Script

Conjunto de instrucciones.

Sistemas Legados

Legacy Systems. Es un tipo de activo de TI basado en la productividad, como hardware, software o algún servicio que está cercano a terminar su vida útil. Se sabe que un sistema es legado y está llegando al final de su vida útil porque se convierte en una barrera para la innovación y necesita ser remplazado

Sistemas Operativos

Operating Systems. Sistema tipo software que controla la computador y administra los servicios y sus funciones como así también la ejecución de otros programas compatibles con éste. Ejemplos de familias de sistemas operativos: Windows, Unix, Linux, DOS, Mac OS, etc.

Un sistema operativo permite interactuar con el hardware de computadores, teléfonos celulares, PDAs, etc. Y ejecutar programas compatibles con éstos. Permite controlar las asignaciones de memoria, ordenar solicitudes al sistema, controlar los dispositivos de entrada y salida, facilitar la conexión a redes y el manejo de archivos

Sitio Web

Es un conjunto de páginas Web, típicamente comunes a un dominio de Internet o subdominio en la World Wide Web en Internet. Una página Web es un documento HTML/XHTML accesible generalmente mediante el protocolo HTTP de Internet. A las páginas de un sitio Web se accede desde un URL raíz común llamado portada, que normalmente reside en el mismo servidor físico. Los URL organizan las páginas en una jerarquía, aunque los hiperenlaces entre ellas controlan cómo el lector percibe la estructura general y cómo el tráfico Web fluye entre las diferentes partes de los sitios.

SOA

Service Oriented Architecture (Arquitectura orientada a servicios). Es un marco de trabajo conceptual que permite a las organizaciones unir los objetivos del negocio con la infraestructura de TI integrando los datos y la lógica de negocio de sus sistemas separados. Desarrollada a finales de los ’90, SOA establece un marco de trabajo para servicios de red – o tareas comunes de negocios – para identificar el uno al otro y comunicarlo.

SOAP

Simple Object Access Protocol (Protocolo de acceso simple). Es un protocolo estándar creado por Microsoft, IBM y otros, está actualmente bajo el auspicio de la W3C que define cómo dos objetos en diferentes procesos pueden comunicarse por medio de intercambio de datos XML. SOAP es uno de los protocolos utilizados en los servicios Web

Software

Software es un término genérico que designa al conjunto de programas de distinto tipo (sistema operativo y aplicaciones diversas) que hacen posible operar con el ordenador. Son aquellos programas que nos ayudan a tareas específicas como edición de textos, imágenes, cálculos, etc. también conocidos como aplicaciones

Swing

Es una librería gráfica para Java.

URL

Uniform Resource Locator (Localizador Uniforme de Recursos). Permite localizar o acceder de forma sencilla cualquier recurso de la red desde el navegador de la WWW. Las URLs se utilizan para definir el documento de destino de los hiperenlaces, para referenciar los gráficos y cualquier otro fichero que se desee incluir dentro de un documento HTML. Cada elemento de Internet tendrá una URL que lo defina, ya sea que se encuentre en un servidor de la WWW, FTP, gopher o las News. Una direcciónURL tiene tres partes: Siglas del Protocolo, Ruta en el Servidor y Nombre de Domino del ordenador servidor. Las Siglas del Protocolo indican qué protocolo se usará para la transmisión de datos; la Ruta en el Servidor indica la ubicación del archivo o fichero que se ha solicitado por parte del usuario dentro del ordenador o servidor; todos los ficheros o archivos están ordenados jerárquicamente en el servidor en una estructura de directorios o carpetas

WSDL

Web Services Description Language (Lenguaje para la Descripción de servicios Web). Es una plantilla o interfaz que permite a las aplicaciones el describirle a otras aplicaciones, las reglas para interactuar entre sí.

W3C

World Wide Web Consortium, un consorcio internacional que produce estándares para la World Wide Web. Desarrolla tecnologías interoperables (especificaciones, guías, software y herramientas) para llevar a la Web a alcanzar su máximo potencial. W3C es un foro de información, comercio, comunicación y entendimiento colectivo.