ULACIT
UNIVERSIDAD LATINOAMERICANA DE CIENCIA Y TECNOLOGIA

LICENCIATURA EN INGENIERIA INFORMATICA
CON ENFASIS EN DESARROLLO DE SOFTWARE

"ASEGURAMIENTO DE LA CALIDAD DEL SOFTWARE APLICANDO PATRONES DE DISEÑO DURANTE EL PROCESO DE DESARROLLO"

Sustentante: José Pablo López Umaña

PROYECTO DE GRADUACION PARA OPTAR POR EL GRADO DE
LICENCIADO EN INFORMATICA

San José – Costa Rica
ABRIL 2005

DECLARACION JURADA

Yo José Pablo López Umaña, alumno de la Universidad Latinoamericana de Ciencia y Tecnología (ULACIT), declaro bajo la fe de juramento y consciente de la responsabilidad penal de este acto, que soy el autor intelectual de la Tesis de Grado titulada: "ASEGURAMIENTO DE LA CALIDAD DEL SOFTWARE APLICANDO PATRONES DE DISEÑO DURANTE EL PROCESO DE DESARROLLO", por lo que libero a la ULACIT, de cualquier responsabilidad en caso de que mi declaración sea falsa.

Brindada en San José - Costa Rica en el día 12 del mes de Abril del año dos mil cinco.

Firma del estudiante: José Pablo López Umaña
Cédula de Identidad: 1-1082-0301

Tabla de Contenidos

CAPITULO I

Introducción

La presente investigación ha sido realizada en InfoMasters S.A., empresa dedicada a la capacitación, consultoría y desarrollo en las áreas de tecnología de entidades financieras. Los resultados del presente trabajo son específicos para esta Empresa, sin pretender generalizar a otros contextos.

Con el afán de mejorar su proceso de desarrollo y la calidad del mismo, InfoMasters S.A. piensa incorporar nuevas técnicas de gestión en sus planes, para garantizar, a bajo costo, la producción de software de alta calidad.

Bajo este esquema, en el Capitulo I, se realiza una inducción al proceso y la necesidad manifestada por la Empresa.

El Capitulo II de la investigación, proporciona el marco teórico; los conceptos y la relación necesaria entre ellos, para la compresión de los objetivos específicos.

El capitulo III define la metodología y aspectos generales relativos al proceso de la investigación, que permiten contextualizar los resultados producidos.

El capitulo IV contiene dos esquemas. El primero es un diagnóstico que pretende establecer los elementos necesarios del contexto de la Empresa; así como una evaluación de su proceso de desarrollo con base en el planteamiento mencionado anteriormente. El segundo esquema de propuesta, permite establecer nuevos procesos, técnicas y medidas utilizables con el fin de asegurar la calidad del software producido por la Empresa, con base en las carencias.

Adicionalmente, se incluyen los formatos de los documentos utilizados durante el proceso de desarrollo de InfoMasters S.A., con el fin de ubicar al lector en el proceso y en los informes entregados de cada fase.

El desarrollo del software, beneficiando a las partes involucradas, pretende solventar, como una necesidad implícita en el proceso: reducciones en la insatisfacción del cliente, el costo, el tiempo y el uso de recursos adicionales; estos son elementos en el aseguramiento de la calidad.

Justificación

La industria de software en Costa Rica, desde la perspectiva de proceso que este involucra, se ha caracterizado, desde sus inicios, como una actividad meramente artesanal. El tiempo para concluir un proyecto de software, el presupuesto, la preparación académica, etc. son algunos factores que inciden en esta realidad.

Adicional a estos factores, se une la escasa recolección de requisitos para el sistema, problema que desemboca en un producto que no satisface las necesidades planteadas por los usuarios directos.

Un factor adicional de importancia es que el proceso de revisión y pruebas, al final del proyecto, resulta ser nada más que una reconstrucción del software, llevado a cabo con el fin de corregir los defectos que el mismo presentaba al cliente. De una manera generalizada, todos estos elementos integrados incrementan, directamente, los costos del proyecto. Este incremento será mayor, entre más temprana sea la fase en donde se cometen los mismos.

Bajo este escenario, se puede deducir que cada error cometido en las fases tempranas del proyecto, se traducirá, al final, en un defecto del ciclo de desarrollo que significará un incremento del costo asociado. De esta manera, el costo de un

error en el proceso de desarrollo, será inversamente proporcional a la fase en la cual se encuentre el proyecto.

Por esta necesidad nace la idea de refinar los procesos de software existentes, con el fin de incrementar la productividad a través de una correcta Gestión de Calidad del Diseño. Esto, en busca, verdaderamente, de un producto software, que conlleva el manejo de estándares relativos al mismo. En el caso particular del software, el término calidad del software se define como "concordancia con los requisitos funcionales y de rendimiento explícitamente establecidos, con los estándares de desarrollo explícitamente documentados y con las características implícitas que se espera de todo software desarrollado profesionalmente" (Pressman, 2002).

El concepto de calidad del software, va creciendo de la mano de las metodologías y las herramientas de trabajo utilizadas por los desarrolladores actuales. El impacto que las tecnologías han tenido, en nuestro medio, se ve reflejado en la creciente necesidad de las empresas desarrolladoras de incrementar sus niveles de calidad mediante procesos específicos; con el fin de garantizar al comprador el producto que desea, en el tiempo estimado y libre de errores.

Con base en lo anterior, las fases de un proyecto, en donde se debería concentrar el esfuerzo intelectual serían, básicamente, el análisis y diseño. De esta manera, los errores producidos en el proceso de desarrollo serían minimizados a tiempo, evitando, de esta manera, su evolución a defecto del software. Según la Real Academia Española, el término error es aplicado a una "acción desacertada o equivocada. Cosa hecha erradamente." (RAE, 2001). Por su parte, el mismo Organismo define defecto como la "carencia de alguna cualidad propia de algo. Imperfección en algo o en alguien." (RAE, 2001). Estos mismos conceptos son trasladables al software como producto.

Planteamiento del problema

InfoMasters S.A. nace como una empresa consultora y capacitadora, específicamente en el área de modelos de decisión, enfocada hacia el sector financiero costarricense. Bajo este esquema, y liderada por el MBA Fernando Hernández Brenes, InfoMasters evoluciona, a solicitud de sus clientes, y adopta un esquema de trabajo, de triple enfoque, incluyendo, dentro de su planeamiento, su función como desarrolladora de software.

El proceso es normal dentro del ámbito de los negocios, en donde una consultaría abre la puerta hacia una capacitación y por último, hacia un desarrollo de software financiero, hecho a la medida, apoyados en la tecnología de software de riesgo financiero de Palisade Corporation.

En el año 2003, se presenta el proyecto MAC (Modelo de Análisis Crediticio) que es adoptado, con mucho éxito, como herramienta de trabajo, de Gestión de Riesgo, en un importante banco privado nacional, para su área de Crédito Corporativo. Durante su desarrollo, el equipo de trabajo puso a la luz, varias carencias de planificación y de conceptos de ingeniería de software que no eran aplicados correctamente durante la implementación del mismo.

La Empresa cuenta con un equipo de dos Analistas Programadores, y como Gerente de Proyectos el Lic. Hernández. Adicionalmente, el Administrador de los Proyectos es el Ing. Alejandro Carballo Herrera, Ingeniero Industrial y Especialista en Tecnologías de Software, egresado de Cenfotec. Por otra parte, el rol de Analista de Sistemas, es desempeñado por dos Ingenieros que tienen bajo su coordinación la labor de dos programadores cada uno, para un total de 7 personas en el área.

Las herramientas de desarrollo, InfoMasters S.A. son Microsoft .NET (Visual Basic.Net, CSharp.Net ASP.NET) y como motor de base de datos SQL Server 2000. En la parte de modelado, actualmente se utiliza Rational Rose 2003.

Bajo este perfil empresarial, y tomando en consideración limitantes: el tipo de negocio, el presupuesto, etc., se requiere un proceso de gestión óptimo del diseño que permita controlar el avance del software, no solo desde la perspectiva de la Administración de Proyectos, en el cumplimiento de los objetivos, sino también evaluando que sean llevados a la práctica en un ambiente de calidad. Esto con el afán de reducir los costos que significaría un mal diseño de los Sistemas y por ende la insatisfacción del Cliente.

Formulación del problema

- ¿Cómo utilizar patrones de diseño para asegurar la calidad del software durante el proceso de desarrollo en InfoMasters S.A.?

Matriz básica de diseño de investigación

TEMA	PROBLEMA	OBJETIVOS	
		GENERALES	**ESPECÍFICOS**
Aseguramiento de la calidad del Software aplicando patrones de diseño durante el proceso de desarrollo	¿Cómo asegurar la calidad del diseño del software en InfoMasters mediante la aplicación de patrones de diseño durante el proceso de desarrollo?	Determinar la situación actual del proceso de desarrollo de aplicaciones de InfoMasters en términos de calidad.	1. Analizar del proceso de negocio y del proceso actual de desarrollo de software.
			2. Identificar los modelos y arquitecturas utilizados para el desarrollo de software financiero que realiza InfoMasters.
			3. Medir la calidad del proceso de desarrollo de software de InfoMasters.
		Establecer un modelo de desarrollo que permita la aplicación de patrones de diseño en la fase de diseño con el fin de mejorar la calidad del producto de software.	4. Establecer un proceso de gestión de patrones de diseño paralelo al proceso de desarrollo de InfoMasters S.A.
			5. Definir un plan de aseguramiento de la calidad del diseño para el área de Desarrollo de software del InfoMasters S.A.
			6. Incorporar al proceso de desarrollo una métrica de aseguramiento de la calidad para la fase de diseño de cada ciclo de desarrollo.

Matriz de operacionalización de variables

Variables	Definición Conceptual	Definición Operacional	Indicadores	Instrumentos de Recolección de datos
Productividad del Personal	Relación entre lo producido y los medios empleados	Nivel de aprovechamiento del factor humano en términos de los recursos utilizados	• Tiempo de desarrollo	• Observación • Análisis de datos existentes
Creación de tecnología	Capacidad del proceso para agregar tecnología para su uso posterior	Porcentaje de componentes y/o paquetes utilizables en proyectos futuros	• Porcentaje de reutilización	• Observación • Análisis de datos existentes
Calidad del proceso	Grado en que un conjunto de características inherentes cumple con los requisitos establecidos	Grado de completitud de los indicadores a evaluar.	• Complejidad • Costo de desarrollo del software • Confiabilidad del sistema • Facilidad de mantenimiento • Eficiencia del proceso	• Observación • Análisis de datos existentes • Entrevistas no estructuradas

Capitulo II

Marco Teórico

Patrones de diseño de software

Se puede definir *patrón* como “una descripción de un problema y su solución que recibe un nombre y que puede emplearse en otros contextos” (Larman, 1999). Toda esa experticia que ha sido acumulada por los diseñadores de software permite extraer un conjunto de patrones.

El software se construye con base en representaciones abstractas de características similares entre ellas, sea en funcionalidad, en interfaz, o bien en su proceso de desarrollo. Es en el último ítem donde los patrones de diseño son aplicables; debido a que, utilizando cualquier metodología, durante el proceso de desarrollo de software, se aporta de común denominador una fase de diseño del producto.

Es menester recordar que el software posee un carácter evolutivo, aún en nuestros días; porque el cambio es una constante implícita del software. Los procesos, las metodologías, las técnicas y herramientas se encuentran en constante cambio y revisión, lo que hace del software un elemento variable por naturaleza. Con base en este panorama se puede deducir que, conforme la experiencia, se incrementan nuevos patrones, reconocidos y reutilizados por los desarrolladores. Esto crea un valioso nivel de conocimiento a nivel empresarial, que puede ser aprovechado por los analistas, diseñadores de software, en busca de la construcción de modelos más estables y con una claridad superior. Entre mayor uso haya tenido un patrón, mayor aplicabilidad y funcionalidad aporta al modelo que se esté formulando.

Clasificación de Patrones de Diseño según su propósito

Los patrones de diseño pueden ser clasificados en patrones creacionales, estructurales y de comportamiento; pueden ser distribuidos, como en la figura #1, la cual aporta algunos ejemplos de los mismos.

		Ámbito	
		Clase	**Objeto**
Propósito	**Creacional**	Método Fábrica	Fábrica Abstracta, Constructor, Prototipo, Singleton
	Estructural		Adaptador, Puente, Compuesto, Decorador, Fachada, Peso Mosca, Apoderado
	De Compor-tamiento	Intérprete, Método Plantilla	Cadena de Responsabilidad, Comando, Iterador, Mediador, Memento Observador, Estado Estrategia, Visitante

Figura #1: Clasificación de los patrones de diseño más comunes

Fuente: (Ramírez, 2004)

Patrones creacionales: son los patrones aplicados a la creación de objetos del sistema. Esta creación puede ser llevada a acabo de diferentes maneras, las cuales principalmente se definen por el uso posterior de los objetos creados y su naturaleza arquitectónica.

Patrones estructurales: se ocupan de la combinación de objetos para crear estructuras complejas. Algunos se encargan de la relación de herencia entre las clases y otros se encargan de la composición de las mismas (instancias de otras clases que componen una tercera).

Patrones de comportamiento: son los encargados de definir el modo en que las clases y los objetos interactúan entre sí. Describen no solamente patrones de objetos, o clases si no también patrones de comunicación entre ellos.

Adicional a esta clasificación, Larman (1999), en su libro, adiciona dos patrones más que no caben dentro de la clasificación anteriormente descrita; sino más bien, son clasificados como *patrones evaluativos*: el patrón de bajo acoplamiento y el patrón de alta cohesión. Pressman (2002) y Kan (2003) tratan estos patrones como métricas de diseño, lo que demuestra su aplicabilidad desde diferentes perspectivas de la ingeniería de software.

Para el efecto de esta investigación, ambos serán considerados como patrones de diseño, adoptando el modelo de Larman; sin embargo, se enfocarán, también, como indicadores numéricos, completamente medibles para el proceso de investigación, con el fin de establecer su aplicación dentro de una métrica de diseño de software.

El detalle de cada patrón de diseño, su definición, estructura y usos particulares, debido a su extensión, pueden ser consultados en (Larman, 1999) o bien en diferentes sitios de Internet dedicados al tema.

Calidad como producto de los patrones de diseño

García (2002) hace referencia en su texto a una frase propia de Booch: “Una arquitectura orientada a objetos bien estructurada esta llena de patrones. La calidad de un sistema orientado a objetos se mide por la atención que los diseñadores han prestado a las colaboraciones entre sus objetos”.

Singh (1998) por su parte declara abiertamente que “los patrones de diseño son capaces de […] mejorar la calidad de los sistemas de software”

Larman (1999) expresa en su libro que “los patrones […] sirven para mejorar la calidad del diseño”.

Muchos autores reconocidos a nivel mundial, concuerdan con la visión de los antes citados. Los patrones de diseño mejoran la calidad del software, y adicionalmente, mejora el proceso de desarrollo. Esto debido a que facilitan la comunicación entre los diseñadores de un sistema complejo (Larman, 1999), brindando la posibilidad de poder referirse a toda una estructura de colaboración que interactúa para resolver un problema de diseño, por medio de un nombre común.

Todo este panorama nos lleva a cuestionar la dimensión del término calidad dentro del contexto de un producto software.

Calidad de Software

Como se mencionó en el Capitulo I, Pressman (2002) define el término calidad del software como la “concordancia con los requisitos funcionales y de rendimiento explícitamente establecidos, con los estándares de desarrollo, explícitamente documentados, y con las características implícitas que se espera de todo software desarrollado profesionalmente”.

Por su parte, De Antonio (1999) establece una divergencia del concepto citado por Pressman, al afirmar que la calidad del software puede ser aplicada a dos áreas específicas: el producto como tal y el proceso con el cual se elabora el producto.

Es importante mencionar que las metas en cuando a la calidad del producto, establecidas al inicio de su elaboración, están directamente relacionadas con el proceso de desarrollo de software y las medidas de aseguramiento de calidad que se establezcan dentro del mismo.

Esta idea es respaldada por Pooley (2002) que sugiere que "para construir sistemas de alta calidad una organización necesita tanto técnicas que se centren en la calidad del producto, como técnicas que se centren en la calidad del proceso".

Algunos de los aspectos que De Antonio (1999) menciona como justificantes para esta realidad, son por ejemplo el software mismo, como entidad abstracta, su proceso de desarrollo (se desarrolla, no se fabrica), su carácter artesanal, y su explosividad al realizar cambios sobre él mismo. También hace referencia a que como disciplina el desarrollo de software es verdaderamente joven, al punto que los defectos presentes en este, se asumen y no se rechazan.

Para esta investigación en particular, el enfoque de calidad utilizado será la calidad del software, desde la perspectiva del proceso de desarrollo.

Garantía de calidad en el proceso de desarrollo

La garantía la calidad es el proceso de "convencerse a uno mismo y al cliente de que cualquier producto que se entregue es de alta calidad" (Pooley, 2002). También agrega que "esto se hace mediante la monitorización y la mejora del proceso de desarrollo de software con el objetivo de aumentar las posibilidades de que las personas que siguen el proceso produzcan un software de alta calidad".

Este proceso de garantía de calidad se ve apoyado directamente por un sistema de gestión de calidad, que básicamente es un documento que especifica cuales estructuras y procesos tiene la organización para asegurar que, cada proyecto sigue un proceso de calidad apropiado, en busca de la mejora continua.

Dentro del marco del sistema de gestión de calidad (QMS) se pueden identificar algunos elementos tangibles, bastante comunes dentro del proceso, a nivel generalizado:

- Plan de Calidad: de manera muy general, es el hito del proyecto que enumera las tareas a realizar como parte del aseguramiento de calidad, sobre el proyecto en curso.

- Auditorias de Calidad: cumplen una función fiscalizadora del Plan de Calidad propuesto.

El aspecto más importante de la garantía de calidad es la *documentación*, que permitirá madurar los procesos siguientes (mejora continua); además de permitir aprender de los éxitos y fracasos obtenidos en el pasado. Surgen muchos mecanismos que permiten estandarizar la documentación y, por ende los procesos. El SEI, IEEE entre otras, son entidades no lucrativas que promueven un entorno de gestión de calidad para el proceso del software, que tienen como común denominador una metodología bien establecida para la documentación de cada fase del proceso de desarrollo.

Desde esta perspectiva, los patrones de diseño proveen al diseñador de software una clara herramienta para definir estructuras documentables y bien establecidas, que pudieran ser utilizadas, a posteriori, en el desarrollo de otro proyecto. Todo esto facilita la conceptualización del nuevo modelo y su implementación.

A este punto, es donde podemos ubicar los patrones de diseño dentro del proceso de gestión de calidad. No solamente para definir la arquitectura del sistema, sino también para estandarizar las técnicas, desde el punto de vista del programador, como del equipo de pruebas del sistema, facilitando el entendimiento entre las partes y permitiendo una mejora sustancial en la calidad implícita del producto.

Capitulo III

Tipo de investigación

El enfoque cualitativo es el que mejor caracteriza esta investigación. Esto por cuanto se basa en las actividades cotidianas de la empresa, objeto de estudio. No busca utilizar modelos matemáticos complejos para describir el entorno, más bien, analiza el proceso de desarrollo, desde una perspectiva de conocimiento colectivo.

La investigación posee carácter exploratorio, debido al poco estudio que se ha realizado del proceso de desarrollo, además de ser un área de negocios, relativamente nueva, dentro del contexto de InfoMasters S.A.

Sujetos y fuentes de información

La fuente primaria de datos será el mismo factor humano que tiene relación directa con InfoMasters S.A., dentro del área de desarrollo de software.

Algunos elementos, dentro de este contexto, a los que se tienen acceso, son principalmente las experiencias individuales, materiales escritos, tales como los documentos de planificación de los proyectos en los que ha participado InfoMasters S.A.; además de los manuales técnicos de las aplicaciones generadas durante este proceso.

Las conversaciones personales con los directores de proyecto, analistas, así como los mismos programadores y hasta el propio gerente de la empresa, serán recolectadas con el afán de diagnosticar el estado del proceso de desarrollo, en términos de calidad del proceso.

Población Objeto de Investigación

Para esta investigación, la unidad de análisis estará compuesta por el grupo de analistas, desarrolladores y gerentes de proyecto involucrados, directamente, en el proceso de desarrollo de software de InfoMasters S.A. Por ser una empresa de reducido tamaño la necesidad de realizar un muestreo es irrelevante; más bien, los datos serán extraídos de todos los participantes del proceso, que formen parte del equipo de trabajo de InfoMasters S.A.

Instrumentos de recolección de datos

A continuación se listan cada uno de los instrumentos de recolección de datos a utilizar durante la presente investigación:

- Observaciones: Los analistas, desarrolladores y administradores de proyecto serán observados con el fin de evaluar su comportamiento del proceso de desarrollo.

- Análisis de registros y datos existentes: Se estudiarán los escritos generados sobre las aplicaciones creadas anteriormente en la empresa, las cuales incluyen las planificaciones de proyectos y los manuales técnicos de cada sistema.

- Entrevistas: se entrevistarán a algunos clientes seleccionados, no aleatoriamente, con el fin de consultar la opinión, tanto del usuario experto del sistema, como de la contraparte en el área de tecnología, sobre el proceso de desarrollo utilizado y la interacción de ellos dentro de ese mismo proceso. Además se consultarán los resultados obtenidos del proyecto realizado.

Alcances y limitaciones de la investigación

Esta investigación exploratoria pretende, como corolario, alcanzar un mejoramiento del proceso de desarrollo de software, mediante un sistema de aseguramiento de la calidad del diseño. Esta mejora en la calidad se realizará mediante la identificación de las carencias del proceso actual, y aplicando las recomendaciones generadas.

Capitulo IV

Objetivo Específico de Diagnóstico 1

Efectuar un análisis del proceso actual de desarrollo de InfoMasters

Generalidades de InfoMasters S.A.

InfoMasters S.A. cuenta con un grupo interdisciplinario, con el cual lleva a acabo sus proyectos. Liderada por el MBA Fernando Hernández Brenes, ha realizado diversos negocios en sus áreas de acción, con diferentes clientes del entorno financiero, foco actual de acción de La Empresa. Por ser una empresa pequeña, en crecimiento constante, día a día, se sobrecarga de funciones. Aun cuando los puestos están bien definidos, toda la organización colabora en la consecución de los objetivos y metas planteados, provocando así un sentimiento de pertenencia y un compromiso con la visión.

A continuación se detalla un organigrama parcial de InfoMasters S.A., mostrando solamente el área de Desarrollo de Sistemas de Información:

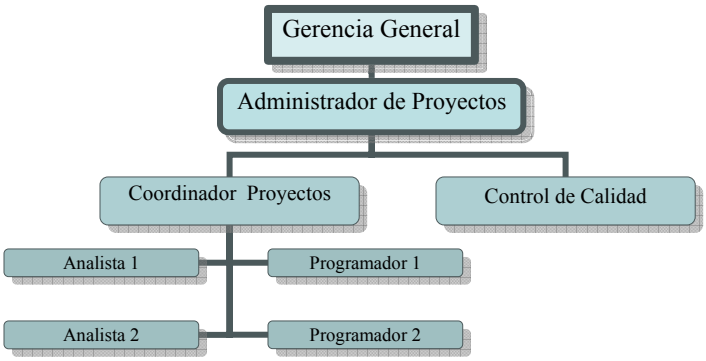

Figura #2: Organigrama del Área de Desarrollo de InfoMasters S.A.
Fuente: Elaboración propia

InfoMasters ha incursionado en el campo del desarrollo del software, producto de su estrategia de negocios. Esta se basa, principalmente, en tres áreas citadas a continuación:

1. Capacitación del Factor Humano
2. Consultoría de los procesos financieros
3. Desarrollo de Sistemas de Información

Detalle del Proceso de Negocio

Capacitación del Factor Humano

El primer elemento de contacto con el cliente es básicamente la capacitación en herramientas tecnológicas, enfocándose sobre la gestión de riesgo; e.g., desde la perspectiva de proyectos, o desde la administración financiera.

Principalmente, se han desarrollado cursos especializados de @*Risk*[1], utilizando *Project* o bien *Excel* como herramientas medulares de trabajo.

La Empresa centra sus esfuerzos atrayendo nuevos clientes de empresas relativas al entorno financiero nacional e internacional.

Consultaría de los Procesos Financieros

Durante los procesos de capacitación ocurre muy frecuentemente que algún jefe, gerente o administrador específico, de un área en particular, observa la necesidad de realizar un estudio más profundo de la situación actual de su empresa. Lo que a la luz de los nuevos conocimientos adquiridos durante el curso de Evaluación de Riesgo Financiero recibido, es un acierto.

[1] @Risk es producido por Palisade Corporation. Se distribuye como un Plug-in para Microsoft Project y Microsoft Excel, para el pronóstico del riesgo basado en modelos matemáticos y estadísticos, principalmente de la Simulación Montecarlo, para predicción de escenarios con base en datos históricos.

Bajo esta cadena de trabajo, en algunas ocasiones, las empresas que envían a sus colaboradores a capacitarse en InfoMasters, solicitan, a esta entidad, los servicios como ente consultor.

De esta manera, se realiza un análisis exhaustivo de la situación actual en el área financiera, y de los posibles factores de riesgo que inciden en este diagnostico, utilizando herramientas tecnológicas propias de la Gestión de Riesgo adicionales.

Desarrollo de Sistemas de Información

Las aplicaciones del software *@Risk* son tan variables, que su fabricante provee a InfoMasters con los componentes *ActiveX,* gestores del análisis de riesgo, para una personalización de los mismos a un modelo especifico de su negocio.

El enlace entre la consultaría y el desarrollo es simple: la consultoría genera problemas que podrían, en algunos casos, ser mejorados utilizando herramientas de riesgo, desarrolladas por InfoMasters S.A.

Este flujo de negocios no es invariable, sino flexible. En ocasiones se podría negociar una consultaría que genere un contrato de capacitación; de esta manera, el enfoque es multidisciplinario y adaptable a las necesidades planteadas por el cliente, en un momento determinado.

El flujo que se definió anteriormente se basa en la visión personalizada del Gerente de la Empresa.

El proceso de desarrollo que se utiliza en InfoMasters S.A. se encuentra debidamente documentado, al igual que lo sucedido dentro de la creación del software que producen.

El proceso de desarrollo de InfoMasters S.A.

El proceso utilizado por InfoMasters S.A. pretende adaptarse al propuesto por Larman (1999), el cual posee las características de ser iterativo e incremental. Sin embargo, en InfoMasters S.A. el mismo ha sido personalizado por los analistas, de tal forma que, se aplique el ciclo de desarrollo de software en cascada con el proceso de Larman. Esto debido a que el mismo posee características iterativas, no así características incrementales, ya a que todos los requerimientos son gestionados desde el inicio del análisis y diseño del sistema completo. Los mismos son tratados exhaustivamente por los analistas al inicio del proyecto.

Las principales fases del proceso de desarrollo se detallan a continuación:

I. Concepción del Sistema

Esta etapa del proceso involucra tareas tan importantes como la creación del estudio de viabilidad del sistema (EVS, Anexo #1) y el cronograma de actividades.

II. Elaboración del Sistema

Está compuesta por los documentos de análisis y diseño del sistema, entre los cuales se pueden enumerar: especificación de requisitos del sistema (ERS, Anexo #2), el de especificación de casos de uso (ECU, Anexo #3) y la especificación del diseño y arquitectura del sistema (DAS, Anexo #4).

III. Construcción del Sistema

El sistema como producto, el Manual de Usuario del Sistema (MUS), cuyo formato es dependiente de la especificación del cliente, y el informe de ciclo de desarrollo (ICD, Anexo #5) son los productos de esta

fase del proceso actual. El sistema debe incluir anotaciones incrustadas en el código fuente, sin embargo las mismas carecen de algún estándar definido por InfoMasters S.A. Adicionalmente, los objetos de las tres etapas utilizadas carecen de una notación prefija definida para su identificación.

IV. Pruebas del Sistema

Las pruebas del sistema se realizan por medio de los casos de prueba que define el analista y la contraparte del proceso. El documento de Casos de Prueba del Sistema (DCP) posee un formado estándar para cada caso de prueba. Sin embargo, el método de definición de los casos de prueba, así como su aplicación, son elementos dependientes del analista que lo realice. Adicionalmente, estas pruebas se hacen, únicamente, para verificar la funcionalidad de los casos de uso; sin embargo, las pruebas actuales no se realizan a nivel del proceso, o bien a nivel de diseño y/o código.

V. Correcciones del Sistema

Esta etapa pretende la corrección de los casos de uso que no satisfagan el 100% de su funcionalidad, especificada contractualmente. La retroalimentación del usuario es relevante en este proceso, ya que basado en su opinión se aprobará o rechazara un caso de uso. En la mayoría de las veces, esta fase marca un hito importante dentro del cronograma del proyecto, en términos de tiempo y costo; ya que se arrastran los errores cometidos desde un principio y se tratan de corregir hasta el final del proyecto.

Objetivo Especifico de Diagnóstico 2

Identificar los modelos y arquitecturas utilizados para el desarrollo de software por InfoMasters S.A.

La observación ha sido utilizada como instrumento de recolección de datos, gracias a la documentación provista por el equipo de trabajo de InfoMasters, en donde se detalla las arquitecturas utilizadas en el desarrollo de sus sistemas. El conjunto de aplicaciones usadas por la Empresa, para esta investigación, se limita a tres únicamente, debido a los contratos de confidencialidad que han firmado con varios de sus Clientes; porque estos no permiten la divulgación de la información a terceros, aún cuando esta conserve un carácter académico.

Arquitectura de los Sistemas desarrollados

InfoMasters S.A., por ser una empresa de reciente creación, adolece fundamentalmente de estándares de desarrollo bien definidos; sin embargo, sí se han documentado los procesos de software realizados en proyectos anteriores.

Aún cuando estas limitantes existan, los analistas de InfoMasters S.A. han definido, al menos en su forma teórica, la arquitectura y el proceso de desarrollo de software para sus proyectos. A continuación se brinda una descripción de la arquitectura y los procesos.

La arquitectura de desarrollo de tres capas consiste en un modelo que separa los diferentes componentes de la aplicación según su objetivo, manteniendo su independencia, y gestionados a través de interfaces bien definidas por los analistas. A continuación se detalla en la siguiente figura:

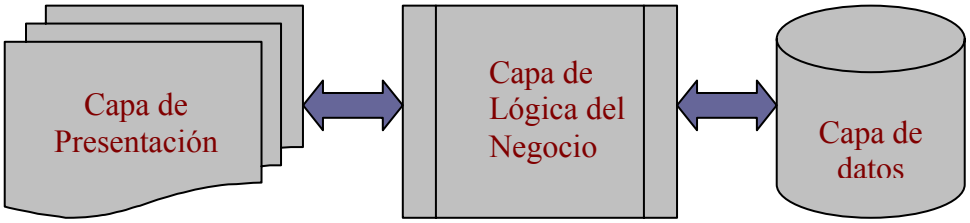

Figura #3: Arquitectura Estándar de InfoMasters S.A.
Fuente: Elaboración propia

La capa de presentación de este modelo representa la interfaz con el usuario, que es la responsable de capturar los datos que serán ingresados al sistema, y la validación de su integridad.

La capa de lógica del negocio recibe los datos ingresados por la capa anterior y los transforma, utilizando un conjunto de componentes especializados de software, acorde a las reglas del negocio. Los componentes son independientes en su lógica y forma de construcción; sin embargo, contienen interfases que permiten su comunicación e intercambio de datos.

La capa de datos provee el almacenamiento de los datos procesados en un repositorio, manteniendo la independencia de esta con respecto a las anteriores. Las interfases, entre esta capa y la de lógica de negocio, son independientes de la implementación, lo que hace flexible el modelo.

Arquitectura basada en Excel y Risk Developers Kit

Parte de la estrategia de negocio de InfoMasters S.A. es crear software basado en Microsoft Excel y @Risk, con el fin de obtener una mayor aceptación por parte del usuario experto.

Debido al tipo de mercado en el cual incursiona InfoMasters S.A., la idea de sistemas basados en Excel tiende a agradar a todos aquellos usuarios del sector financiero; los cuales encuentran peculiar esta herramienta que flexibilizan el desarrollo de sus tareas específicas.

Adicional a Excel, los usuarios desean potenciar las habilidades de las aplicaciones, utilizando las herramientas que @Risk les presenta siempre; esto por cuanto @Risk utiliza Excel como herramienta de hospedaje.

Dado este esquema, la capa de objetos de dominio será enriquecida con los paquetes de interoperabilidad para cada una de estas aplicaciones, según se detalla en la figura a continuación:

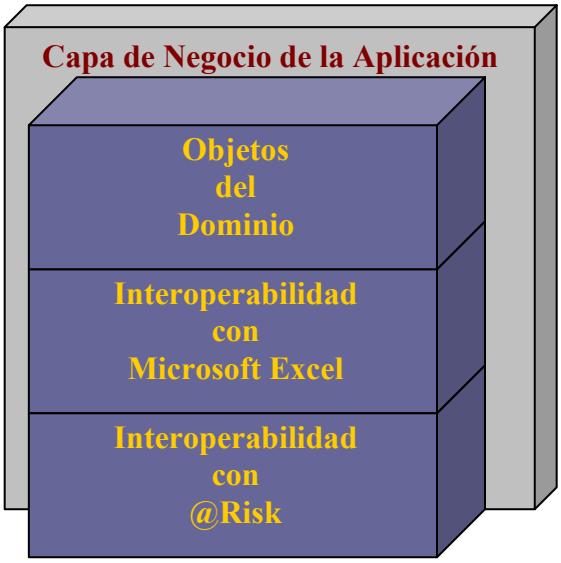

Figura #4: Objetos de la Capa de Negocio
Fuente: Elaboración propia

Esta modificación a la arquitectura de los sistemas es utilizada actualmente por los analistas solamente en los casos en que sea requerida. En algunas ocasiones la misma puede variar, dependiendo de la negociación que se realice con el cliente y del propósito y tipo de aplicación que se desee crear.

Objetivo Especifico de Diagnóstico 3

Medir la calidad del proceso de desarrollo de la Empresa.

Resultados de las mediciones obtenidas

Para medir cada variable del proceso de desarrollo de InfoMasters S.A. se tuvo acceso a tres proyectos de sistemas de información de riesgo bancario, cuya arquitectura era similar a la planteada anteriormente, permitiendo así, un punto adecuado de comparación.

La información respectiva fue recopilada a través de la observación y el análisis de datos existentes. Además, se contó con la participación activa del Gerente de la Empresa, el MBA. Fernando Hernández Brenes, quien a través de una entrevista no estructurada, reveló algunos de los datos que se resumen a continuación. Otros, tales como el nombre del proyecto, la institución financiera que adquiere el producto, etc., no fueron revelados por ser contratos de privacidad firmados con sus respectivos clientes.

Originalmente, los tres proyectos se estimaron para tres, siete y ocho meses respectivamente. Los dos primeros se llevaron a cabo en paralelo; sin embargo y como era de esperarse, no iniciaron al mismo tiempo. El primer proyecto inició tres meses antes que el segundo.

Variable: Creación de tecnología

La variable “Creación de tecnología” hace referencia al proceso de construir elementos de software reutilizables, pero con la más alta calidad y, sobre todo, con la generalidad que se esperaría de un componente de propósito general.

Aún cuando el campo de aplicación es el software de riesgo bancario, la ingeniería de software orientado a objetos permite crear paquetes y/o componentes que

cuenten con la generalidad necesaria para poder reutilizarlos de un proyecto a otro y que tengan sentido dentro del concepto del negocio.

Tomando como referencia los tres proyectos anteriormente citados, se presentaron los siguientes índices de reutilización:

Capa de Negocio	% de Reutilización
Objetos del Dominio	0%
Objetos de Conectividad con Excel	100%
Objetos de Gestión de @Risk	25%
Otros controles y/o utilitarios	60%

Tabla #1: Indicador de la Variable "Creación de tecnología"
Fuente: Elaboración propia

Aún cuando las aplicaciones se encuentran enfocadas hacia un mercado y un dominio de acción en particular, la reutilización que se hace entre los objetos del dominio es nula, debido a la incorrecta estructuración de los mismos. Por su parte, los objetos de conexión con Excel fueron elaborados, utilizando un diseño correcto, lo suficientemente generales para poder ser reutilizados, transparentemente, en las tres aplicaciones.

Variable: Productividad del Personal

Durante la creación de estos proyectos, los programadores tuvieron que esforzarse para dividir su carga de trabajo entre los proyectos que lo requirieran. De esta manera era normal que los mismos se encontraran desarrollando, paralelamente, dos o más casos de uso de proyectos diferentes.

La tabla a continuación resume el indicador de productividad presentado por los proyectos objeto de estudio:

Factor humano	Proyecto	Casos de Uso	Tiempo utilizado	Índice de productividad
Programador 1	Proyecto #1	15 CU	8 meses	1.875 CU/Mes
	Proyecto #2	4 CU	3 meses	1.333 CU/Mes
	Proyecto #3	13 CU	7 meses	1.857 CU/Mes
Programador 2	Proyecto #1	13	8 meses	1.625 CU/Mes
	Proyecto #2	7	3 meses	2.333 CU/Mes
	Proyecto #3	12	7 meses	1.714 CU/Mes

Tabla #2: Indicador para la variable "Productividad del personal"
Fuente: Elaboración propia

Variable: Calidad del Proceso

A continuación se detalla, en la tabla #1, un resumen de los datos obtenidos por cada uno de los índices de la variable de calidad seleccionados para su evaluación:

Indicador	Proyecto #1	Proyecto #2	Proyecto #3
Complejidad	28 Casos de Uso	11 Casos de Uso	25 Casos de Uso
Costo de desarrollo del software	$43000	$17000	$38500
Costo por CU	$1535.71	$1545.45	$1540
Confiabilidad del sistema	3.5 reportes de errores por mes	2 reportes de errores por mes	3 reportes de errores por mes
Facilidad de mantenimiento	7 correcciones de errores por mes 2 nuevos requerimientos por mes	9 correcciones de errores por mes 3 nuevos requerimientos por mes	5 correcciones de errores por mes 2.5 nuevos requerimientos por mes
Eficiencia del proceso	3.50 CU por mes	3.66 CU por mes	3.57 CU por mes

Tabla #3: Indicadores de la variable "Calidad del proceso"
Fuente: Elaboración propia

En forma general el éxito alcanzado por los tres proyectos no fue el óptimo, en términos de la consecución de los objetivos y requerimientos planteados. Particularmente, el proyecto #1, presentó serias deficiencias en el acopio de los requisitos, los cuales variaron significativamente al final del proyecto.

De los veintiocho casos concebidos en uso, el 68%, de ellos fue afectado, por los cambios que solicitó el cliente. Dado este panorama y considerando que, esta reestructuración del producto se presentó al final del mismo, se incrementaron, considerablemente, factores como el tiempo y el costo.

El sistema creado, en el primer proyecto, presentaba no solo carencias en la parte de recolección de requerimientos, sino también problemas importantes de diseño. Al final de la evaluación del impacto en los cambios, se obtuvo que los objetos del dominio tuvieron cambios cercanos al 80%, en términos de cantidad de clases, y que las relaciones existentes se verían modificadas en un 65% de la capa de lógica de negocios.

Evaluación del Proceso de Desarrollo Actual

InfoMasters S.A. ha adoptado, en teoría, el proceso que Larman (1999) había sugerido en su libro, con algún nivel de personalización, según el criterio de los analistas de la empresa y tomando en consideración su realidad. Bajo este esquema es de esperarse el establecimiento de un proceso bien definido, iterativo incremental manejado por casos de uso. Sin embargo, este panorama no es precisamente la norma dentro de InfoMasters S.A.; esto por cuanto el proceso ha sido transformado en cascada, en donde la recolección de los requisitos del sistema a construir y su modelado se realizan al iniciar el proyecto.

Precisamente este escenario ha sido discutido al inicio de este documento, en donde se enfatiza la importancia de la minimización de errores causados en las fases tempranas de un proyecto de software; ya que los mismos son arrastrados

inminentemente hasta las fases finales en donde el cliente verifica que el producto cumple o no con su expectativa.

La incorrecta aplicación del proceso de desarrollo iterativo incremental se ejemplifica en el momento en que se concibe, el mismo, como una serie de actividades consecutivas, sin retroalimentación del usuario experto, sobre dicha creación.

El usuario experto toma un rol pasivo y no activo como el proceso de desarrollo que Larman (1999) promueve; i.e., no involucra la retroalimentación del usuario experto en el proceso de recolección y absorción de requisitos que desea el cliente del Sistema. En realidad la gestión, a base de casos de uso, se limita a una aproximación general de la descripción de sus procesos y no a un análisis exhaustivo y bidireccional entre el analista y el usuario experto.

El concepto de Ciclo, o Iteración tampoco ha sido implementado dentro de la planeación de cada proyecto realizado, lo que ha traído como consecuencia la falta de puntos de control para los casos realizados en uso.

Fase de Concepción

La fase de Concepción lleva a cabo el Estudio de Viabilidad del Sistema (EVS), cuya función es verificar que el proyecto iniciado tenga un objetivo, sea realizable desde el punto de vista de recursos económicos y tecnológicos. Además, esta evaluación se ve enriquecida con el Plan del Proyecto, para ubicar cronológicamente al cliente. Dicha tarea la realizan en conjunto el Administrador de Proyectos y la gerencia general. Ellos son los responsables directos de transmitir al cliente las ideas de la Empresa, para solucionar y/o optimizar su modelo de negocios a través de la herramienta tecnológica propuesta.

En los tres proyectos analizados se encontró suficiente documentación que permite corroborar, positivamente, el seguimiento del proceso y de los estándares utilizados.

Fase de Elaboración

La fase de Elaboración posee un nivel óptimo de congruencia entre la teoría y la práctica. Durante esta fase se definen, como productos principales, la Especificación de Requisitos del Sistema (ERS), la Especificación de Casos de Uso (ECU) en donde se detallan los de alto nivel y esenciales, y el Diseño de la Arquitectura del Sistema (DAS) que detalla el modelo conceptual preliminar y la arquitectura propuesta en cuestión.

El glosario de términos se realiza implícitamente en cada uno de los documentos respectivos, agrupando los conceptos que requieran aclaración utilizando referencias dentro del mismo escrito.

En caso de presentar al cliente un prototipo, en Excel, él mismo lo evalúa a su satisfacción, y decide si la idea está siendo captada con claridad por parte del analista.

Fase de Construcción

En la teoría, la fase de construcción esta dividida en iteraciones; cada una con un conjunto de tareas propias del análisis, diseño, construcción y pruebas de los casos de uso implementados en el mismo. Es aquí en donde la labor del Analista se incrementa de forma significativa, debido a la importancia y necesidad de absorber, completamente, los conceptos del dominio que deberá implementar en esa fase.

A su vez, esta fase debe producir el Informe de Ciclo de Desarrollo (ICD), en donde se detallan los siguientes artefactos: Diagramas de Casos de Uso, el

Modelo Conceptual Parcial, los Diagramas de Secuencia, los Diagramas de Colaboración, el Diagrama de Clases Parcial y por último, el Modelo de Datos Parcial. Algunos artefactos tendrán carácter acumulativo, e.g. en una fase determinada del desarrollo del proyecto se incluirá el diagrama de clases y el modelo de datos de la fase anterior.

Sin embargo la teoría contrasta con la realidad, ya que en la mayoría de los casos se incumple, no solo con la eliminación de las iteraciones, sino también de los artefactos respectivos, los cuales se definen al inicio del proyecto, suprimiendo detalles importantes, que deberían ser captados por el analista a lo largo del proceso. En algunos casos los errores que se comenten son detectados y se intentan corregir; sin embargo estas correcciones no son documentadas debidamente, por lo cual al final la implementación no mantiene su paralelismo con lo realizado.

Para ilustrar esto, el documento de Estudio de Requisitos del Sistema (ERS) incluía artefactos propios de la elaboración del sistema, tales como los Diagramas de Casos de Uso, los Diagramas de Interacción, el Diagrama de Clases total del Sistema y el Modelo de Datos Final. De esta manera, la frontera entre las fases de Concepción y Elaboración carece de forma según el proceso adoptado por la Empresa.

Como un ejemplo de esta perspectiva, en un proyecto de riesgo bancario, realizado durante el 2003, en un banco del sector privado de nuestro país, se utilizó este tipo de implementación con resultados bastante desalentadores. El analista que cumplía la contraparte, expresó brevemente que “el proyecto tuvo un manejo desordenado, carecía de retroalimentación y al final el software no cumplía con los requisitos planteados”.

Fase de Pruebas

El proceso explicado por Larman (1999) incluía dentro de la Fase de Construcción, una subfase de Pruebas, en donde se incluyen las tareas que el desarrollador realiza como control del producto que está desarrollando. Ante esta situación, InfoMasters S.A. promueve el establecimiento de una Fase de Pruebas que trabaje en paralelo a los analistas y desarrolladores.

Las objetivos de esta fase serán el aseguramiento de la calidad del producto desde sus inicios; es decir, desde la recolección de requerimientos hasta las pruebas post implantación.

Sin embargo, como parte de las limitaciones de índole económico de una empresa desarrolladora en sus inicios, InfoMasters S.A. no contaba con personal exclusivo para coordinar estas tareas a lo largo del desarrollo de los proyectos.

Dada la última experiencia negativa, el software de riesgo bancario desarrollado no cumplía con los requisitos ni con el control de calidad debido, a nivel de diseño, la Gerencia decide incluir dentro del equipo de trabajo a un analista desarrollador que cumpliera con estas funciones en la corrección y reestructuración del software en cuestión.

De esta manera, a partir de ese momento se incluye como entregable el Documento de Casos de Prueba (DCP), en donde se reúnen los escenarios a evaluar por parte del encargado de control de calidad. Cada escenario evalúa precisamente los procesos definidos al inicio del proyecto y que han sido depurados por cada iteración, además de evaluar las condiciones, antes y después del proceso y el nivel de satisfacción del cliente en elementos de Interfaz.

Dentro de esta fase se omite un factor importante dentro de la ingeniería de software como lo es la prueba de caja blanca (Pressman, 2002), en donde el software no solamente es evaluado en términos de satisfacción de requerimientos

sino también que los algoritmos y la complejidad del mismo sea aceptable. Para determinar esto Pressman (2002) hace referencia, en su libro, a varias técnicas de evaluación para modelos orientados a objetos que se podrían utilizar.

Fase de Correcciones

Originalmente la fase de correcciones nace como un complemento a la fase de pruebas, en donde los resultados serán corregidos por el equipo de trabajo.

Su ejecución deberá ser llevada a cabo una vez finalizada una iteración de la fase de construcción y las pruebas correspondientes a ese ciclo de desarrollo.

Hasta hace poco tiempo estas se realizaban una única vez al final del proyecto, a continuación de la fase de pruebas única que se realizaba al Sistema antes de su entrega al cliente. Debido a esto, esta fase consumía un importante porcentaje del tiempo total del proyecto, ya que era aquí el punto donde se intentaba reestructurar el software de los defectos encontrados.

Esta reestructuración del sistema significaba en promedio un incremento del 25% del tiempo estimado por el administrador de proyectos al inicio, y una disminución significativa proporcional en la utilidad del proyecto.

En el caso del software de riesgo bancario tomado como ejemplo para esta investigación el incremento real sobre la estimación fue de un 30%, aproximadamente, en donde se intentó rescatar el software hasta el momento desarrollado pero con mal suceso. El mismo al final del proyecto no reunía las cualidades técnicas y funcionales que se esperaban de el, por lo cual, a complacencia y requerimiento del cliente se decidió replantear, desde el inicio, el desarrollo de una segunda versión que captara realmente los requerimientos y cumpliera con las expectativas del cliente.

Objetivo Específico de Propuesta 1

Establecer un proceso de gestión de patrones de diseño paralelo al proceso de desarrollo de InfoMasters S.A.

Aspectos Generales

La aplicación de patrones de diseño durante el proceso de desarrollo involucra la administración de los mismos. Bajo este esquema, la selección y administración de los mismos permite un mejor manejo durante el proceso de desarrollo, enriqueciendo el dominio del conocimiento de los analistas y/o verificadores de calidad.

Todo este panorama nos hace reflexionar sobre la creación de un proceso de gestión de patrones de diseño, que pueda ser utilizado cuando un nuevo patrón de diseño es descubierto dentro de un contexto definido por los analistas de sistemas.

Descripción del proceso

El proceso a describir comienza en el momento en que un posible patrón de diseño ha sido encontrado por los analistas, en un contexto definido. El mismo deberá ser comprobado con la documentación existente, o bien ser probado para asegurar su generalidad de uso en posibles implementaciones. La figura #5 nos muestra gráficamente la interacción de las fases:

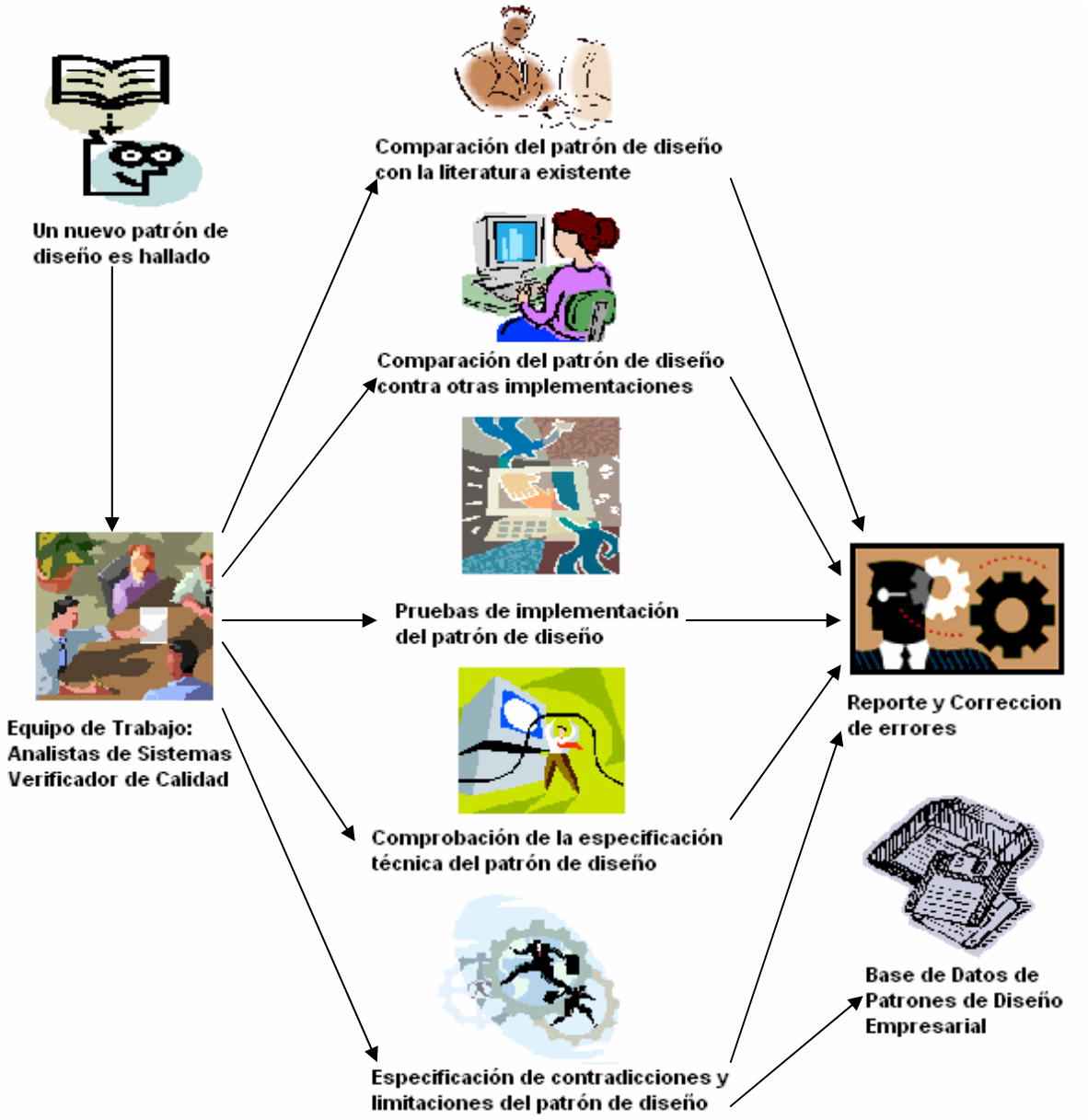

Figura #5: Proceso de gestión de patrones de diseño para InfoMasters S.A.
Fuente: Elaboración propia

A continuación se detallan las fases correspondientes de este proceso:

Fase 1: Comparación del patrón de diseño con la literatura existente.

En algunas ocasiones un posible patrón de diseño puede existir de antemano en un catálogo o base de datos de patrones empresariales. Esta base de datos, cuya

implementación es de carácter genérica y abierta a su libre formato. Luego InfoMasters S.A., registra los patrones de diseño que ha acumulado durante su experiencia en el desarrollo de software. La claridad y generalidad de cada patrón son características claves para su correcto funcionamiento.

Si el patrón de diseño es hallado en el catálogo, el proceso termina; caso contrario, se intentará ubicar en un catálogo de patrones de diseño externo. En caso de no existir la documentación requerida para el nuevo patrón, el proceso continúa su curso.

Fase 2: Comparación del patrón de diseño contra otras implementaciones

Para poder avalar el nuevo patrón de diseño como tal, el analista deberá de demostrar con evidencia propia del proyecto, o de proyectos anteriores, que el mismo existe y que posee congruencia entre su definición y su utilización.

Fase 3: Pruebas de implementación del patrón de diseño

Un patrón de diseño deberá ser objeto de implementación, de lo contrario carecería de utilidad. Además de su implementación, la eficacia y eficiencia del mismo deberán ser expuestas por el analista, demostrando que mantiene sus características de generalidad, modularidad, adaptabilidad y de reutilización.

Fase 4: Comprobación de la especificación técnica del patrón de diseño

Cuando el patrón de diseño ha sido avalado por el verificador de calidad, el analista deberá redactar la documentación respectiva del patrón, con el fin de incluirlo en el catalogo empresarial de patrones. Para dicho efecto se sugiere el formato del Anexo #7, el cual incluye la información mínima de registro de un nuevo patrón de diseño.

Fase 5: Especificación de contradicciones y limitaciones del patrón de diseño

Antes de la inclusión del patrón de diseño en el catálogo empresarial, es recomendable definir las limitaciones que presente ese patrón en conjunto con las contradicciones que este presente, si existieran. Para realizar esta tarea el Anexo #7 incluye la definición de las mismas para su uso posterior.

Objetivo Específico de Propuesta 2

Definir un plan de aseguramiento de la calidad del diseño para el área de Desarrollo de software del InfoMasters S.A.

Aspectos Generales

El proceso de aseguramiento de calidad del diseño propuesto a continuación, parte de la premisa que los requisitos funcionales y no funcionales han sido evaluados correctamente, según el plan de calidad que InfoMasters S.A. posea.

Dicho proceso se basa fundamentalmente en cinco fases, incluyendo como actores principales a un analista de sistemas designado por InfoMasters S.A., al verificador de calidad, al usuario experto y, por último, al analista de sistemas designado por el cliente. El proceso se detalla en la figura a continuación:

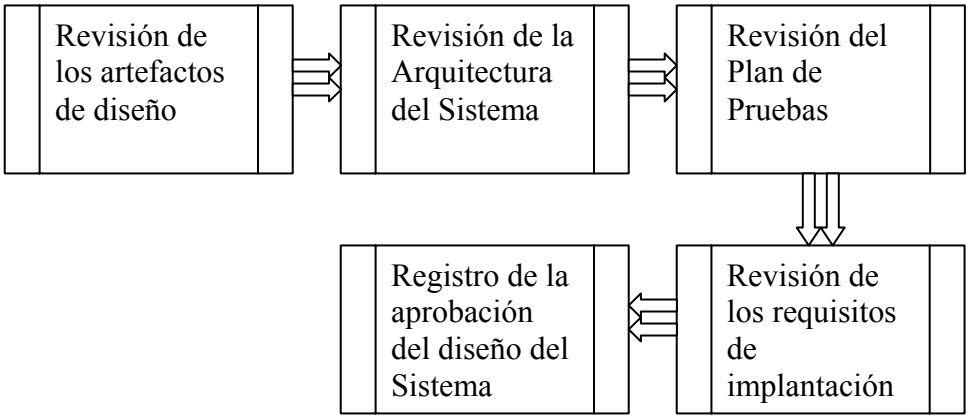

Figura #6: Plan de aseguramiento de la calidad del diseño.
Fuente: Elaboración propia

Fases del Plan de Aseguramiento de la Calidad del Diseño

Fase 1: Revisión y Verificación de los Artefactos de Diseño

Fase 1.1: Ejecución de la Métrica de Aseguramiento de la Calidad

Esta actividad comprende la aplicación de la métrica de aseguramiento de calidad a los artefactos generados durante la fase de diseño por los analistas.

La métrica de aseguramiento de la calidad es independiente del proceso de aseguramiento de la calidad del diseño, lo cual permite flexibilizar el mismo ante los diversos cambios tecnológicos.

Para esta fase se podrá contar como mínimo con los diagramas de casos de uso, los diagramas de secuencia del sistema, los diagramas de colaboración y/o diagramas de secuencia de cada interacción con el sistema y el diagrama de clases.

Fase 1.2: Revisión de los Patrones de Diseño Aplicados

Esta tarea deberá ser ejecutada por el Verificador de Calidad en conjunto con el analista designado.

El primero deberá verificar la existencia de patrones de diseño presentes en los artefactos entregados. Registrará su existencia además del contexto en el que fue utilizado.

El segundo justificará el uso de este patrón en el contexto definido por el verificador, de esta forma queda suficiente evidencia de la correcta aplicación de los patrones de diseño en el ciclo de desarrollo actual.

La herramienta de registro de esta información podrá seguir el formato que se encuentra en el Anexo #7; sin embargo, podrá ser reemplazado libremente por cualquier otra herramienta que realice la misma función.

Fase 1.3 Registro de la Aprobación de los Artefactos de Diseño

En caso de no existir alguna corrección sobre los artefactos evaluados, se procederá a la formalización del proceso. Tanto el analista como el verificador de calidad firmarán el documento respectivo.

Fase 2: Revisión y Verificación de la Arquitectura del Sistema

Fase 2.1: Revisión de la consistencia de la arquitectura diseñada

Se deberá comprobar que la arquitectura obtenida durante la fase de diseño del ciclo actual se ajusta a las normas y estándares establecidos en el Plan de Aseguramiento de la Calidad de la Empresa. La misma deberá responder a los requisitos especificados durante la fase de análisis.

Los cambios en la arquitectura deberán ser justificados por parte del analista encargado; el verificador de calidad evaluará que los mismos no afecten el futuro desempeño de la aplicación.

Fase 2.2: Registro de la Aprobación de la Arquitectura del Sistema

Una vez finalizada la revisión por parte del Verificador de Calidad y del Analista de Sistemas, los responsables del sistema llamados usualmente “contraparte” de la Empresa, durante el proceso de desarrollo, deberán aprobar su arquitectura formalmente, tomando así en consideración las variantes del ambiente tecnológico en que deberá desempeñarse el producto.

La aprobación, o rechazo de la arquitectura, deberá ser registrada, tomando en consideración los criterios establecidos en el plan de aseguramiento de calidad. En caso de ser rechazada, se deberán registrar los puntos negativos encontrados en la misma durante su revisión.

Fase 3: Revisión del Plan de Pruebas

Fase 3.1: Revisión del diseño de los casos de pruebas

Se deberá comprobar que el diseño de los casos de prueba cumpla con las especificaciones correspondientes; tanto los casos de uso como del entorno operacional. Las pruebas unitarias y/o de implantación deberán ser revisadas en esta etapa.

Fase 3.2: Revisión del Plan de Pruebas

Esta fase busca la comprobación de la inclusión en el plan de pruebas tanto de las pruebas de implantación como de aceptación. Las pruebas de implantación deben tomar en consideración los requisitos no funcionales relacionados con las propiedades de calidad. Por su parte, las pruebas de aceptación buscan la verificación y satisfacción del usuario experto del sistema.

Fase 4: Revisión de los requerimientos de implantación

Fase 4.1: Revisión de los requerimientos de documentación

Se deberá coordinar con el cliente el formato de cada informe de ciclo que le sea entregado, así como aspectos relativos a la distribución de los mismos, tales como el número de copias, destinatarios, etc.

Cualquier consideración adicional sobre la documentación que el cliente haga, durante esta fase, deberá ser tomada en consideración de inmediato para hacerla efectiva durante el ciclo actual.

Fase 4.2: Revisión de los requerimientos de la plataforma

Los requerimientos específicos de la plataforma de trabajo para el producto en proceso de creación, deberán ser evaluados de tal forma que se logre identificar cualquier cambio en la configuración del mismo. Dichos cambios eventuales deberán ser notificados al Cliente en búsqueda de su aprobación.

Fase 5: Registro de la aprobación del diseño del sistema

Fase 5.1 Registro de la aprobación del Verificador de Calidad

El verificador de calidad registrará la aprobación del diseño del sistema, con el fin de guardar evidencia suficiente durante el proceso de gestión de calidad.

El formato a utilizar queda a la libre elección de la Empresa; sin embargo, el registro de dicha aprobación será el respaldo para el analista. El mismo deberá recibir copia de la aprobación.

Fase 5.2 Registro de la aprobación del usuario experto y técnico

La aprobación del diseño del sistema deberá ser aprobada por la contraparte y el usuario experto, garantizando así que el mismo cumple con todas las especificaciones correspondientes. En caso de no cumplir con alguna directriz de diseño en particular, deberá ser informada para su inmediata corrección.

Objetivo Específico de Propuesta 3

Incorporar al proceso de desarrollo una métrica de aseguramiento de la calidad para la fase de diseño de cada ciclo de desarrollo.

Aspectos Generales

Kan (2003) en su libro hace referencia a diferentes métricas creadas por autores tales como Lorenz, Chidamber y Kemerer (CK), IBM, entre otros. Sin embargo, con base en los estudios realizados por sus autores respectivos, la métrica CK reúne dos factores importantes: brevedad de aplicación y efectividad relativa.

Debido a esto, se ha elegido dicha métrica para su aplicación dentro del proceso de aseguramiento de calidad del diseño de InfoMasters S.A. La tabla presentada a continuación brinda una referencia más clara de los indicadores:

Indicador	Comentario
Número de métodos por clase (WMC)	Se calcula simplemente por el número de métodos que implemente la clase. El promedio de WMC es el número promedio de métodos por clase.
Profundidad de la Jerarquía de clases (DIT)	Deberá ser menor que 6, iniciando por las clases del *framework* o la clase base.
Numero de hijos por clase (NOC)	Número inmediato de subclases en una jerarquía
Acoplamiento entre las clases (CBO)	Debe ser lo más bajo posible pero mayor a cero. Es el número de clases a las cuales la clase se encuentra acoplada.
Falta de Cohesión entre Métodos (LCOM)	Deberá ser lo mas alto posible. La cohesión de una clase esta indicada por que tan relacionados se encuentran los métodos a las variables de instancia

	de la clase. LCOM está medido como el número de conjuntos disjuntos de métodos locales. La falta de cohesión incrementa la complejidad y los posibles errores durante el proceso de desarrollo.
Respuesta de una Clase (RFC)	Es el número de métodos que pueden ser ejecutados en respuesta a un mensaje recibido por un objeto de esa clase. Entre mayor sea el número de métodos llamados, mayor será la complejidad de clase.

Validación y Justificación del Modelo

Según Kan (2003) se realizó un estudio en el año de 1996 en la Universidad de Maryland, USA, en donde participaron estudiantes de ingeniería de software, específicamente del curso de Análisis y Diseño Orientado a Objetos. El estudio involucró a 8 equipos de trabajo, los cuales evaluaron alrededor de 180 clases sobre diferentes sistemas.

La hipótesis buscaba enlazar valores altos de los indicadores de la Métrica CK con la probabilidad de encontrar clases propensas a error. Algunos hallazgos del estudio se muestran a continuación:

- Los seis indicadores de la métrica son independientes entre si.
- La falta de uso de la herencia fue efectivamente confirmada por los bajos valores obtenidos en los indicadores DIT y NOC.
- Los índices DIT, RFC, NOC y CBO estaban significativamente correlacionados con las clases defectuosas en un análisis estadístico multivariable.
- Estos indicadores fueron superiores a otros de medición de código propios del análisis y diseño estructurado.

Un estudio de mayor relevancia es realizado en el año de 1997, en donde los creadores de la Métrica CK, la aplicaron sobre tres instituciones financieras para comprobar la utilidad de la misma en un entorno empresarial de desarrollo. Los tres sistemas fueron ejecutados por la misma empresa. Los resultados de esta investigación pueden consultarse en (Kan, 2003); sin embargo, algunos de los hallazgos significativos se analizan a continuación.

Uno de los primeros valores observados fue la baja profundidad de la jerarquía de herencia (DIT) y del número de hijos de la clase (NOC); indicando así, que los desarrolladores no estaban aprovechando la reutilización como una característica del diseño orientado a objetos.

En segundo lugar, la correlación entre el WMC, RFC y CBO era de un 85%. Esto significa que los tres indicadores estaban midiendo algo similar.

Además, altos niveles de acoplamiento (CBO) y del indicador LCOM (cohesión) fueron asociados con la baja productividad, un esfuerzo mayor para crear clases reutilizables, y un mayor esfuerzo de diseño.

Los autores usaron esta métrica para señalar las clases con problemas potenciales. Cualquier clase que reúna, al menos, dos de los siguientes criterios, sería señalada para su investigación posterior:

- RFC > 100.
- RFC > 5 veces el número de métodos de la clase.
- CBO > 5.
- WMC > 40.

Consideraciones y recomendaciones finales

- El proceso de desarrollo en InfoMasters S.A. deberá ser llevado a cabo correctamente, según lo definido en su proceso de desarrollo. El equipo, como tal, deberá hacer énfasis en su correcta ejecución con el afán de obtener mejores y más rápidos resultados a nivel de proceso y de producto.

- El rol del verificador de calidad debe ser tomado ampliamente en consideración, ya que el mismo será el responsable de validar el trabajo realizado por el equipo de desarrolladores.

- La incorporación de patrones de diseño será un factor de mejoramiento del sistema, permitiendo realizar diseños de mejor calidad y con mayor confiabilidad.

- El proceso de creación de tecnología, mediante el establecimiento de patrones de diseño de software, dependerá exclusivamente de la seriedad con que el proceso se lleve a cabo. La administración del catálogo de patrones existente resultará muy útil en el proceso de desarrollo del proceso respectivo.

- El proceso de aseguramiento de calidad del diseño es independiente de la métrica a utilizar; sin embargo, se hace énfasis en la utilización de la misma, en la búsqueda de factores que determinen correctamente si el diseño se encuentra en una fase crítica o por el contrario sea un punto fuerte durante el desarrollo del sistema.

BIBLIOGRAFIA

De Antonio, Angélica. (1999). Calidad del Software. Material didáctico no publicado. Universidad Politécnica de Madrid, Madrid, España.

Kan, Stephen. (2003). Metrics and models in software quality engineering. Addison Wesley, USA.

Larman, Craig. (1999). UML y Patrones. Introducción al análisis y diseño orientado a objetos. México: Prentice Hall.

Pooley, Rob. (2002). "Utilización de UML en Ingeniería del Software con Objetos y Componentes". México: Addison Wesley.

Pressman, Roger. (2002). Ingeniería de Software. Madrid, España: Mc Graw Hill.

Ramírez, Elizabeth. (2004). Uso de Patrones de Diseño en .NET - Patrón Decorador. *Revista MTJ .NET Online*. Mayo. Recuperado el 28 de setiembre de 2004 de www.microsoft.com/spanish/msdn/comunidad/mtj.net/

Real Academia Española. (2001). Diccionario de la Real Academia Española. Recuperado el 28 de setiembre de 2004, de buscon.rae.es/diccionario/drae.htm

Singh, Ajit. (1998). Impact of Design Patterns on Quality of Software Systems. Department of Electrical & Computer Engineering. University of Waterloo, Waterloo. Ontario, Canada.

Anexos de la Investigación

Anexo #1: Estudio de Viabilidad del Sistema

Tabla de contenidos

Descripción General del Documento

Actividades del Estudio

- **Definición del Alcance del Sistema**

  <Definición de la frontera de acción de la propuesta>

- **Estudio de la Situación Actual**

  <Detalle de los hallazgos relevantes relativos al o los procesos en evaluación>

- **Definición de los Requisitos Generales del Sistema**

  <Listado general de los requisitos del sistema>

- **Estudio de las Alternativas de Solución**

  <Se plantean las soluciones del caso para resolver el problema>

- **Valoración de las Alternativas**

  <Se evalúan las opciones en términos numéricos para ubicar al cliente dentro de sus posibilidades y realidad económica>

- **Selección de la Solución**

  <Se elije la mejor opción de la evaluación previa. Sin embargo, la evaluación final queda en manos del cliente, quien elige su opción mas adecuada>

- **Cronograma de Ejecución (Si aplica)**

  <En caso de aplicar, se realiza un cronograma de actividades base para ubicar al cliente cronológicamente de forma tentativa>

Anexos

<Quedan a criterio de la gerencia los anexos del documento>

Anexo #2: Especificación de Requisitos del Sistema

Tabla de Contenidos

Descripción General del Documento

Actividades del Estudio

- **Definición del Alcance del Sistema (Refinamiento)**

  <Definición de la frontera de acción del sistema>

- **Identificación del Entorno Tecnológico**

  <Detalle de los hallazgos relevantes relativos al entorno de la aplicación>

- **Especificación de Estándares y Normas**

  <En caso de que la Empresa requiera la aplicabilidad de una norma o estándar en particular>

- **Definición de Requisitos Funcionales**

  <Matriz de requisitos funcionales identificados para el Sistema>

- **Definición de Requisitos No Funcionales**

  <Matriz de requisitos no funcionales identificados para el Sistema>

- **Análisis de Requisitos**

  <Detecta inconsistencias, ambigüedades, duplicidad o escasez de información en los requisitos identificados>

Anexos

<Quedan a criterio del analista los anexos del documento>

Anexo #3: Especificación de Casos de Uso

Tabla de Contenidos

Descripción General del Documento

Actividades del Estudio

- **Listado general de casos de uso**

  <Lista de todos los casos de uso a incluir en el Sistema>

- **Matriz de Seguimiento de Requisitos**

  <Matriz de relación de los requisitos con los casos de uso planteados>

- **Casos de uso de alto nivel**

  <Descripción de cada caso de uso en formato de alto nivel>

- **Casos de uso en formato expandido**

  <En caso de que la Empresa requiera la aplicabilidad de una norma o estándar en particular>

- **Asignación de los casos de uso**

  <Matriz de la asignación de casos de uso según ciclo y recurso responsable>

Anexos

<Quedan a criterio del analista los anexos del documento>

Anexo #4: Diseño y Arquitectura del Sistema

Tabla de Contenidos

Descripción General del Documento

Actividades del Estudio

- **Definición de la Arquitectura del Sistema**

  <Descripción de la arquitectura lógica y física del sistema>

- **Diseño Conceptual del Sistema**

  <Conceptos encontrados y relación entre los mismos>

- **Interacciones del Sistema**

  <Definición de los diagramas de interacción del sistema para cada operación que reciba>

- **Diseño de Clases del Sistema**

  <Detalle de las clases, atributos y métodos según su visibilidad, implementables por el modelo>

- **Diseño de Implementación**

  <Diagramas de despliegue y de paquetes para el ciclo actual>

- **Diseño del Modelo de Datos**

  <Modelo relacional de la base de datos del Sistema>

Anexos

<Quedan a criterio del analista los anexos del documento>

Anexo #5: Informe de Ciclo de Desarrollo

Tabla de Contenidos

Descripción General del Documento

Actividades del Estudio

- **Requisitos Previos de Implantación**

  <Requisitos de instalación para la nueva versión>

- **Especificación de la Base de Datos**

  <Diccionario de la base de datos hasta el momento>

- **Detalles de Implantación de la Base de Datos**

  <Recomendaciones y/o requisitos de implantación de la base de datos>

- **Especificación de los Componentes**

  <Detalle de los componentes y su significado dentro del sistema>

- **Detalles de Implantación de los Componentes**

  <Recomendaciones y/o requisitos de implantación de los componentes>

- **Anotaciones Finales de Implantación**

  <Cualquier anotación de implantación adicional no contemplada en este documento>

Anexos

<Quedan a criterio del analista los anexos del documento>

Anexo #6: Especificación de Casos de Prueba

Tabla de Contenidos

Descripción General del Documento

Actividades del Estudio

- **Definición de los Casos de Prueba**

  <Listado de los Casos de Prueba a ejecutar>

- **Relación de Casos de Prueba con los Casos de Uso**

  <Matriz de relación entre los Casos de Prueba y los Casos de Uso implementados>

- **Resultados de las Pruebas Unitarias**

  <Detalle de los resultados de las pruebas unitarias>

- **Resultados de las Pruebas de Integración**

  <Detalle de los resultados de las pruebas de integración>

- **Resultados de las Pruebas del Sistema**

  <Detalle de la ejecución de los casos de prueba planteados>

- **Anotaciones Finales de los Casos de Prueba**

  <Cualquier anotación adicional no contemplada en este documento>

Anexos

<Quedan a criterio del analista los anexos del documento>

Anexo #7: Especificación de Patrones de Diseño

- **Nombre del Patrón**

  <Palabra o frase corta para referirse al patrón, y el conocimiento y la estructura que describe>

- **Problema**

  <Describe su intención, las metas y objetivos que quiere alcanzar dentro del contexto y problemática encontrada, donde por lo general, los problemas se oponen a los objetivos>

- **Contexto**

  <Describe las precondiciones bajo las cuales el problema y su solución parecen recurrir, y para lo que es deseable la solución, y con esto se puede observar si se puede aplicar el patrón o no a cierto problema>

- **Solución**

  <Representa a las reglas de relación tanto estáticas como dinámicas, que describen cómo obtener el resultado deseado. La descripción puede utilizar figuras y diagramas para identificar la estructura del patrón, sus participantes y sus colaboraciones para mostrar cómo se resuelve el problema>

- **Razonamiento**

  <Nos dice cómo trabaja el patrón actualmente y por qué es bueno. A diferencia de la solución, el razonamiento proporciona el conocimiento de la naturaleza interna de la estructura, y los mecanismos clave que están bajo la superficie del sistema>

- **Fuerzas o problemática**

  <Describen las restricciones relevantes y cómo interactúan aún en conflicto con cada otra y con las metas que se desean>

- **Patrones relacionados**

  <Muestra las relaciones estáticas y dinámicas entre un patrón y otros. Existen patrones que comparten las fuerzas, otros comparten los contextos iniciales y finales, unos dan una solución alterna al mismo problema, incluso algunos pueden aplicarse simultáneamente.>

- **Ejemplos**

  <Describen las condiciones posteriores al uso del patrón y los efectos laterales, buenos y malos, del patrón>

Guadalupe, abril del año 2005.
Srs. Centro de Investigación y Desarrollo Empresarial.
Universidad Latinoamericana de Ciencia y Tecnología.
ULACIT

Estimados señores:

El suscrito, con licencia del Colegio de Licenciados y Profesores, con cédula 1 276 287, hace constar lo siguiente:
que revisó la tesis **"ASEGURAMIENTO DE LA CALIDAD DEL SOFTWARE APLICANDO PATRONES DE DISEÑO DURANTE EL PROCESO DE DESARROLLO";** realizada por el estudiante **José Pablo López Umaña** , de la Carrera de **INGENIERIA INFORMATICA CON ENFASIS EN DESARROLLO DE SOFTWARE** , para optar el grado de Licenciatura en esa disciplina.
Se examinaron los aspectos formales: ortografía, sintaxis, vocabulario, etc y corregidas ya las sugerencias de estilo, dicho estudio queda apto para ser defendido ante el tribunal respectivo.

Muy atentamente,

_____
Lic. Esaú Molina Mena
Colegiado N° 671

San José, 12 de abril de 2005
Centro de Investigación y Desarrollo Empresarial.
Universidad Latinoamericana de Ciencia y Tecnología.
ULACIT

Estimados señores:

El suscrito, con cédula 1 0571 0421, hace constar que revisó la tesis **"ASEGURAMIENTO DE LA CALIDAD DEL SOFTWARE APLICANDO PATRONES DE DISEÑO DURANTE EL PROCESO DE DESARROLLO";** realizada por el estudiante **José Pablo López Umaña** , de la Carrera de **INGENIERIA INFORMATICA CON ENFASIS EN DESARROLLO DE SOFTWARE** , para optar el grado de Licenciatura en esa disciplina. La misma cumple con los requisitos inherentes y está lista para continuar con su proceso de defensa.

Muy atentamente,

_____
MSc. Georges Alfaro Salazar
Ced. 1 0571 0421