

Universidad de la Ciencias Informáticas

Facultad 2



*Trabajo de Diploma para optar por el título de
Ingenieros en Ciencias Informáticas*

Título: *Análisis y diseño de una herramienta que aplique la Métrica de Avance para el seguimiento de los proyectos de la vista arquitectónica de integración del ERP-Cuba.*

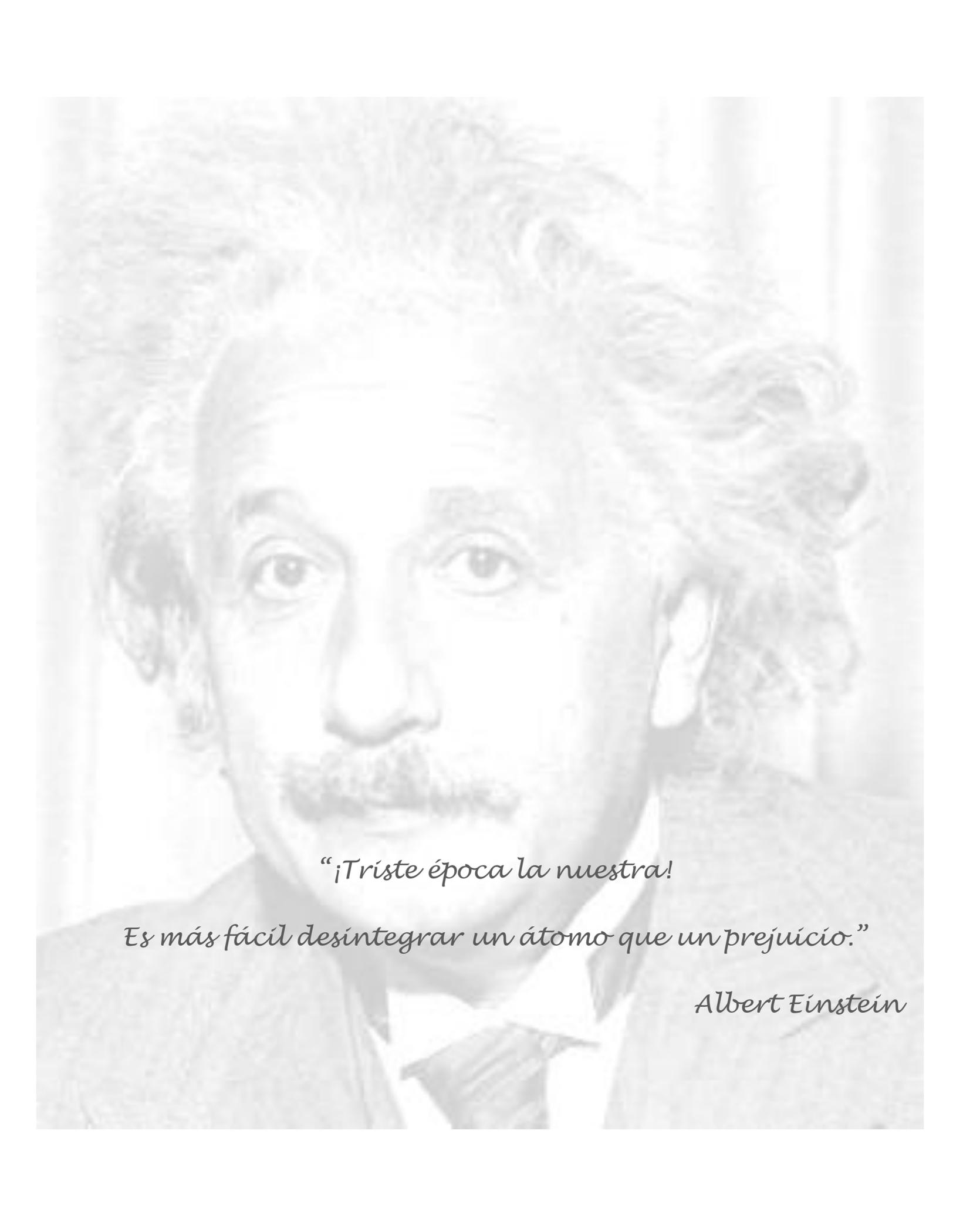
Autores: Diana Rosa Telmo Pacheco

Lisandra Pérez González

Tutor: Ing. Rosario Rodríguez Torres

Ciudad de La Habana, junio del 2010.

“Año 52 de la Revolución. “



“¡Triste época la nuestra!

Es más fácil desintegrar un átomo que un prejuicio.”

Albert Einstein

Declaración de Autoría

Declaramos que _____ y _____ somos los únicos autores de este trabajo y autorizamos a la Universidad de las Ciencias Informáticas (UCI) para que hagan el uso que estimen pertinente con este trabajo.

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

Firma del Autor

Diana Rosa Telmo Pacheco

Firma del Autor

Lisandra Pérez González

Firma del Tutor

Ing. Rosario Rodríguez Torres

Dedicatoria

A mis padres y mis hermanos...

Diana

*A mis padres, mi hermana y
toda mi familia por ser
mi inspiración y mi sostén...*

Lisandra

Resumen

En la Universidad de las Ciencias Informáticas (UCI), existen problemas con el control y seguimiento de proyectos. Existen aplicaciones que permiten definir tareas y las relaciones entre ellas, con las cuales se realiza el seguimiento del progreso de las mismas, pero estas no brindan toda la información necesaria que permita a la dirección de los proyectos tomar las decisiones correctas para que estos sean entregados en la fecha establecida. La información relacionada con el estado de avance de los proyectos productivos no es en ocasiones fiable, ya que no se cuenta con una aplicación que se adapte a las características específicas de la universidad, por lo que la toma de decisiones en cuanto a esfuerzo y tiempo se hace difícil e inexacta. Como consecuencia, los problemas no se detectan a tiempo y los proyectos concluyen casi siempre fuera de fecha. En el presente trabajo de diploma que se realizará bajo la disciplina de desarrollo propuesta por el proyecto ERP-Cuba, se lleva a cabo el análisis y diseño de una herramienta que permite gestionar con mayor eficiencia el estado de avance referente a los proyectos productivos, así como brindar una serie de reportes y gráficas del estado real de los mismos utilizando la Métrica de Avance, la cual toma en cuenta características específicas del marco de trabajo de la universidad. El valor práctico de esta herramienta consiste en disponer de una aplicación web amigable y fácil de usar que aplique la métrica para el seguimiento de proyectos adaptada a las características de la UCI, siendo de gran ayuda a los líderes de proyecto, en el control y seguimiento de los mismos, los resultados de su aplicación servirían de base para futuras estimaciones y planificaciones y los resultados del avance tendrían una base objetiva basada en la medición del software.

Palabras claves

Estado de avance, métrica, Métrica de Avance, estimación, planificación.

Índice de contenido

RESUMEN.....	IV
ÍNDICE DE CONTENIDO	1
ÍNDICE DE FIGURAS.....	3
ÍNDICE DE TABLAS	4
INTRODUCCIÓN	5
CAPÍTULO #1. FUNDAMENTACIÓN TEÓRICA.....	8
1.1. Introducción.....	8
1.1.1. Métrica	8
1.1.2. Métrica de Avance	8
1.1.3. Seguimiento de proyectos mediante la aplicación de métricas.....	9
1.1.4. Integración arquitectónica	10
1.1.5. Vista arquitectónica de integración.....	12
1.2. Estudio del arte	12
1.2.1. Dotproject	12
1.2.2. OpenProj.....	13
1.2.3. GanttProject.....	13
1.2.4. Redmine	14
1.3. Metodologías y herramientas propuestas para el Análisis y Diseño del sistema.....	15
1.3.1. Metodología de desarrollo de Software	15
1.3.2. Herramientas CASE.....	18
1.3.3. Aplicaciones Web	20
1.4. Ingeniería de requisitos.....	21
1.4.1. Actividades de la Ingeniería de requisitos	22
1.5. Arquitectura utilizada	23
1.6. Patrones de diseño.....	25
1.7. Conclusiones del capítulo.....	27
CAPÍTULO #2. CARACTERÍSTICAS DEL SISTEMA	28
2.1. Introducción.....	28

2.2. Métrica de Avance.....	28
2.3. Mapa de procesos.....	34
2.3.1 Descripción de procesos.....	35
2.4. Captura y descripción de requisitos.....	36
2.4.1. Requisitos Funcionales.....	37
2.4.2 Requisitos no Funcionales.....	39
2.4.3 Descripción de los requisitos funcionales.....	41
2.5. Modelo conceptual.....	53
2.6. Diagramas de clases del diseño.....	53
2.6.1. Utilización de Patrones de Diseño.....	54
2.7. Modelo Entidad-Relación.....	57
2.8. Diagrama de Componentes.....	59
2.9. Conclusiones del capítulo.....	60
CAPÍTULO #3. VALIDACIÓN DE LA SOLUCIÓN.....	61
3.1. Introducción.....	61
3.2. Validación de requisitos.....	61
3.3. Métricas para evaluar el diseño.....	65
3.4. Prototipos de Interfaz de usuario.....	67
3.5. Resultados a nivel UCI que ha tenido la utilización de la Métrica de avance.....	68
3.6. Conclusiones del capítulo.....	68
CONCLUSIONES.....	69
RECOMENDACIONES.....	70
REFERENCIAS BIBLIOGRÁFICAS.....	71
BIBLIOGRAFÍA.....	73

Índice de figuras

Figura 1 Fases e iteraciones de la Metodología RUP	17
Figura 2 Patrón MVC	25
Figura 3 Resumen del avance por módulos para una semana dada.....	31
Figura 4 Estado de avance por módulos.....	32
Figura 5 Histórico del avance por semanas	33
Figura 6 Histórico del avance del proyecto.....	33
Figura 7 Mapa de procesos del negocio	35
Figura 8 Modelo Conceptual	53
Figura 9 Diagrama de clases del diseño del requisito Gestionar proyecto.....	54
Figura 10 Modelo Entidad-Relación	58
Figura 11 Diagrama de Componentes	59

Índice de tablas

Tabla 1 Relación de los Pesos de la complejidad del requisito según las combinaciones posibles	30
Tabla 2 Especificación del requisito Gestionar proyecto	42
Tabla 3 Especificación del requisito Gestionar usuario	43
Tabla 4 Especificación del requisito Gestionar tarea.....	45
Tabla 5 Especificación del requisito Gestionar módulo.	47
Tabla 6 Especificación del requisito Gestionar requisito.....	48
Tabla 7 Especificación del requisito Mostrar reportes	50
Tabla 8 Especificación del requisito Generar gráficos	51
Tabla 9 Lista de chequeo.....	61
Tabla 10 Resultados de la métrica TOC	66
Tabla 11 Resultados de la métrica RC.....	67

Introducción

En la actualidad, un equipo de desarrollo de software compuesto por estudiantes y profesionales de la Universidad de las Ciencias Informáticas (UCI), en conjunto con especialistas de Organismos de la Administración Central del Estado (OACE), se encuentran inmersos en la creación de un Software Integral de Gestión que responda a la actual necesidad económica que presenta el país. Para el desarrollo de dicho sistema, cuyo nombre es CedruX, este ha sido dividido en varios subsistemas, la subdirección de Tecnología del Proyecto y dentro de la misma el Grupo de Arquitectura de Sistema atiende la integración de los proyectos de la vista arquitectónica.

Durante la primera etapa del proyecto los líderes de los proyectos, analistas y arquitectos comprobaron que los proyectos están interrelacionados, repercutiendo grandemente un cambio en los requisitos así como el impacto del mismo en la implementación. Teniendo en cuenta que el equipo de desarrollo es muy numeroso y el proyecto es extenso, la transmisión de información se hace difícil, por lo que la manera óptima para gestionar los subsistemas es a través de una herramienta automatizada que permita mantener el seguimiento de los mismos.

Luego de estudiar algunas de las herramientas que pudieran resolver esta situación, se pudo verificar que estas no satisfacen las necesidades reales del proyecto ERP-Cuba para el seguimiento de los subsistemas, puesto que presentan serios problemas de usabilidad dado que los usuarios a menudo no encuentran el flujo de las acciones mientras trabajan con la herramienta, además de que el proyecto trabaja en base a fechas fijas de entrega de los subsistemas, su seguimiento en el tiempo es limitado y no ofrecen todas las funcionalidades que se necesitan para darle un riguroso seguimiento. Una explicación más exhaustiva del estudio se puede encontrar en el Capítulo 1 del presente trabajo de diploma.

Frente a las limitaciones que ofrecen las herramientas estudiadas, se hace necesario crear una nueva, basada en software libre, que sea capaz de facilitar el seguimiento de los subsistemas de la vista arquitectónica de integración; así como permitir conocer el impacto que sobre otros subsistemas tiene, la realización de un cambio en uno de los subsistemas, y brindar dicha información inmediatamente a las personas afectadas. Por este motivo, se plantea la necesidad de realizar el análisis y el diseño, que servirá de base para la posterior implementación, de una nueva herramienta que supla las necesidades reales del proyecto ERP-Cuba.

Dada la situación anteriormente expuesta se plantea como **problema científico** de la investigación: ¿Cómo facilitar el seguimiento de los proyectos de la vista arquitectónica de integración mediante la aplicación de la Métrica de Avance en el proyecto ERP-Cuba?

Partiendo del problema planteado se identificó como **objeto de estudio**: Arquitectura de integración; y como **campo de acción**: Seguimiento de proyectos de la vista arquitectónica de integración del ERP-Cuba.

La presente investigación tiene como **objetivo general** Realizar el análisis y diseño de una herramienta que aplique la Métrica de Avance para el seguimiento de los proyectos de la vista arquitectónica de integración del ERP-Cuba.

Los **objetivos específicos** definidos para este estudio son:

Realizar el marco teórico de la investigación.

Realizar el análisis y diseño de la herramienta.

Evaluar la especificación de requisitos y el modelo de diseño de la herramienta.

Idea a defender

Con la realización de un correcto análisis y diseño de una herramienta que aplique la Métrica de Avance se podrá facilitar el seguimiento a los proyectos de la vista arquitectónica de integración del ERP-Cuba.

Para dar cumplimiento a los objetivos trazados se plantearon las siguientes **tareas de la investigación**:

- ✓ Estudio del estado del arte de la gestión de proyectos, así como los sistemas que existen para la gestión de los mismos, tanto propietarios como libres, para analizar sus características e inconvenientes.
- ✓ Estudio del fenómeno y comprender el problema.
- ✓ Estudio y fundamentación de la elección de la metodología de software usada para realizar el análisis y el diseño.

- ✓ Estudio de las distintas herramientas, técnicas y buenas prácticas que existen para modelar el análisis y el diseño.
- ✓ Análisis y diseño del sistema a partir de las herramientas, las técnicas y buenas prácticas estudiadas.
- ✓ Definición y aplicación de métricas para evaluar los resultados.
- ✓ Redacción del informe final de la investigación.

Capítulo #1. Fundamentación Teórica

1.1. Introducción

En este capítulo se aborda el contexto del objeto de investigación, además el estado del arte realizando un análisis del seguimiento de proyectos basado en métricas. Se caracterizan las tecnologías, herramientas y se analiza la metodología de desarrollo de software que se utilizarán para el análisis y diseño del sistema.

1.1.1. Métrica

El proceso de planificación del desarrollo de cualquier sistema debe hacerse partiendo de una estimación del trabajo a realizar. Sólo a partir de ello es factible conocer los recursos necesarios y el tiempo necesario para su realización. La definición y aplicación de una métrica no es un objetivo en sí mismo sino un medio para controlar el desarrollo de un sistema de software. A continuación se da una explicación sobre las métricas y sus características.

Las métricas no son más que medidas o colecciones de datos de las actividades de los proyectos y los recursos. Ayudan a entender, monitorizar, controlar, predecir y probar un proyecto en desarrollo. Estas deben ser simples, objetivas, fáciles de coleccionar, fáciles de interpretar y difíciles de malinterpretar. Las mismas producen indicadores a partir de los cuales se pueden tomar decisiones importantes, definen un método y una escala cuantitativa que pueden ser usados para determinar el valor que toma cierta característica en un producto de software creado y su objetivo principal está enfocado en medir un producto para aumentar su calidad.(1)(2)

1.1.2. Métrica de Avance

La Métrica de Avance toma en consideración los requisitos a desarrollar, las tareas para desarrollar de acuerdo con el flujo de trabajo establecido por la arquitectura del proyecto y valora su complejidad, tiene en cuenta que las tareas tienen un peso específico, dentro de un tiempo total necesario para implementar los requisitos. Mediante la misma se obtiene cuantitativamente el estado de avance del proyecto, así como su evolución en el tiempo, dando como resultado información precisa y necesaria para la toma de

decisiones gerenciales del proyecto. También aporta el registro histórico de datos que la soportan y constituye una base objetiva para futuras estimaciones.

Características de la Métrica de Avance

- ✓ Permite trazar las estrategias necesarias para cumplir con la planificación.
- ✓ Brinda una base para futuras estimaciones y planificaciones.
- ✓ Permite obtener cuantitativamente el estado de avance de una etapa de un software.
- ✓ Constituye en sí un registro histórico de la productividad del trabajo.
- ✓ Determina qué cantidad de requisitos, puede realizar un equipo de desarrollo en un tiempo determinado.
- ✓ Determina en tiempo real el estado de una etapa de software.

La Métrica de Avance evidencia un sistema de medición y evaluación en tiempo real de una etapa de un software, a diferencia de otros tipos de métricas que no muestran sus resultados con una información precisa y definida.

1.1.3. Seguimiento de proyectos mediante la aplicación de métricas

El seguimiento de proyectos es un proceso de análisis y recopilación sistemática de información a medida que avanza un proyecto. Su principal objetivo es mejorar la eficacia, efectividad y organización de los mismos. Se basa en metas establecidas y actividades planificadas durante las distintas fases del trabajo de la planificación. Ayuda a que se siga una línea de trabajo y permite a la administración conocer cuando algo no está funcionando. Si se lleva a cabo adecuadamente, es una herramienta de incalculable valor para una buena administración y proporciona la base para la evaluación. Permite determinar si los recursos disponibles son suficientes, si están bien administrados, si la capacidad de trabajo es adecuada, y si se está haciendo lo que se había planificado.

Las métricas constituyen una base objetiva para el seguimiento de proyectos, son fundamentales para poder planificar y controlar de forma realista la línea de trabajo de estos. Permiten gestionar y medir una adecuada visibilidad para la administración sobre la situación del proyecto, ya que identifican oportunamente cualquier desviación contra lo planeado, con el objetivo de tomar decisiones oportunas para corregirlas. Especifican el estado actual del avance de las actividades, la desviación de las actividades, avance según la importancia de las actividades, la desviación en el tiempo de las actividades, el costo, los cambios en el alcance original y la eficiencia en los objetivos planteados.(4)(5)(11)

Técnicas de seguimiento basadas en métricas

Las técnicas básicas que normalmente se utilizan para este fin:

- ✓ **Reuniones:** Reuniones con el equipo de trabajo de manera grupal y/o individual para revisar ,controlar y medir el progreso de su trabajo.
- ✓ **Revisiones:** Revisiones de los productos elaborados de acuerdo con el plan de trabajo para validar que los avances sean reales y los productos tengan la calidad suficiente como para considerarlos completados.
- ✓ **Reportes:** Reportes individuales de los integrantes del equipo de acuerdo con una frecuencia especificada (por ejemplo: semanal o diaria).
- ✓ **Software de Administración:** Reportes de los avances y el trabajo realizado por medio de alguna herramienta de planeación y administración de proyectos.

1.1.4. Integración arquitectónica

La integración arquitectónica en el ERP-Cuba se realiza mediante el protocolo Inversión de Control (IoC), éste es el encargado de publicar los servicios de los diferentes subsistemas para que los que necesiten utilizarlos puedan tener acceso a los mismos. Cada subsistema contiene un XML con diferentes funciones y en él se publican los servicios de ese módulo con los elementos que brinda de cada componente. El componente que necesite un determinado servicio de otro componente puede utilizarlo sin tener que escribir todo este código nuevamente, o sea, facilita la reutilización de código. Esto funciona tanto para los

componentes que utilizan funcionalidades de otros componentes dentro de los módulos como para los módulos que utilizan servicios de otros módulos.

Esta integración se realiza en tres niveles:

1er nivel: Es la integración que se realiza entre componentes de un determinado módulo, es un proceso que ocurre internamente ya que estos componentes están relacionados y necesitan recursos unos de otros. Para esto se usa el patrón IoC interno que es el encargado de integrar los componentes a un nivel de módulo. Existe uno para cada módulo.

2do nivel: Es la integración que se realiza entre todos los módulos del proyecto ERP-Cuba. Este es un proceso externo. Los diferentes módulos del proyecto general también están estrechamente relacionados y necesitan recursos de otros módulos, para su funcionamiento. Para realizar esta integración se usa el patrón IoC externo, que hay uno para la integración de todos los módulos del proyecto.

3er nivel: Es la integración del proyecto ERP-Cuba con cualquier sistema o proyecto que necesite sus servicios. Este nivel de integración todavía no se ha implementado, aunque se está haciendo un profundo estudio para poder ponerlo en práctica en un futuro no muy lejano.

Principales problemas a la hora de integrar los subsistemas en el proyecto ERP-Cuba:

- ✓ Falta de experiencia del personal involucrado en el proyecto.
- ✓ Falta de comunicación y de información entre el personal involucrado.
- ✓ No se tiene una definición de los servicios que le brinda un subsistema a otro.
- ✓ Incumplimiento en la terminación de los servicios y estabilidad de los mismos.

Algunos de estos problemas de integración son más frecuentes a la hora de integrar los módulos a nivel de sistema por ser un proyecto muy grande, lo cual no sucede en la integración de los componentes dentro de los módulos ya que la comunicación es mayor y el personal está más compenetrado en el trabajo.

Luego de llevar a cabo la integración entre los subsistemas o módulos se generan varios artefactos que se conoce como la vista arquitectónica de integración.

1.1.5. Vista arquitectónica de integración

Se conoce como vista arquitectónica de integración en el ERP- Cuba a la integración que se realiza a nivel de sistema, es decir, entre los módulos que componen el proyecto en general y los artefactos que se generan al realizar esta integración.

Entre los artefactos generados se encuentra la matriz de trazabilidad que aunque no es un modelo físico, es usado para ver la dependencia o los servicios que brinda un módulo a otro, y los diagramas de componentes donde se muestra la relación entre los componentes.

Una **vista arquitectónica** es "una descripción simplificada (una abstracción) de un sistema desde una perspectiva particular o punto de vista, que cubre particularidades y omite entidades que no son relevantes a esta perspectiva" [Booch 1998].

1.2. Estudio del arte

La utilización de las herramientas basadas en métricas en los proyectos, ha tenido una repercusión importante en la producción, no sólo en lo que se refiere a la información, sino también en el resultado . A nivel mundial existen varios software que utilizan métricas para el seguimiento de proyectos.

Se centra el estudio del estado del arte en los software: Dotproject, OpenProj y GanttProject.

Herramientas que utilizan métricas para el seguimiento de proyectos

1.2.1. Dotproject

Dotproject es una herramienta que permite gestionar las distintas fases y tareas que componen un proyecto. A menudo, esta gestión implica un control en recursos humanos, materiales, que hacen que esta labor se torne compleja y prácticamente inabordable sin la ayuda de determinadas herramientas que den soporte a la tarea de planificación y gestión de proyectos.

Dotproject se perfila como una interesante herramienta para trabajar en entornos colaborativos, permitiendo a los integrantes del equipo trabajar compartiendo información relativa a los proyectos. (12)

1.2.2. OpenProj

Es un software libre, y constituye una alternativa sólida frente a Microsoft Project, la herramienta de Microsoft para la creación, gestión y mantenimiento de proyectos. Permite hacer casi cualquier actividad que realice su símil de Microsoft, tiene soporte para abrir y tratar los archivos de Microsoft Project (compatibilidad importantísima) y permite crear y gestionar tantos proyectos como se necesiten.

Esta aplicación esta disponible para Sistemas Operativos: Windows, Linux, Unix, entre otros. Sus funciones principales, se concentran en el diseño de gráficas de Gantt, diagramas de redes PERT, y muchas otras gráficas más.

OpenProj controla todos los aspectos referentes a la gestión de proyectos, como la planificación y programación, la gestión y asignación de recursos, la simulación de alternativas en procesos críticos, etc. Asimismo, proporciona la funcionalidad necesaria para trabajar con entornos multiproyecto (incluso relacionarlos). También permite hacer un seguimiento de las distintas etapas de un proyecto: presupuesto, definición, estimación, etc.

A pesar de no ser un producto comercial, OpenProj, ofrece todas las características necesarias y suficientes para trabajar en proyectos de gran envergadura. (14)(15)

1.2.3. GanttProject

Es una aplicación que permite organizar y planificar proyectos a través de diagramas Gantt, así como también la carga horaria de los integrantes dentro del mismo.

GanttProject es una aplicación open source, por lo que se puedes descargar de forma gratuita, además es una aplicación íntegramente escrita en Java y la comunidad que existe alrededor, creando plugins es bastante grande por lo que está en continuo desarrollo.

Después de analizar estas herramientas se concluyó que las mismas no brindan todas las funcionalidades que se necesitan teniendo en cuenta las particularidades del proyecto ERP-Cuba. Además de no adaptarse al modelo de la universidad, las métricas que utilizan estiman tiempo y no permiten tener el estado real del proyecto (atrasado o en tiempo) y no permiten cambios en los parámetros de entrada. Por estas razones se decidió diseñar una herramienta que utilice la Métrica de Avance y que resuelva las necesidades reales del proyecto.

1.2.4. Redmine

Redmine es un gestor y planificador de proyectos con interfaz web, orientado a la coordinación de tareas, comunicación de participantes, y que puede especializarse en proyectos de desarrollo gracias a herramientas como la integración en un repositorio de código.

Permite gestionar múltiples proyectos desde una sola interfaz con una ventana de navegador. La navegación es muy sencilla y se puede saltar y cambiar de proyecto en cualquier momento. También dentro de cada proyecto pueden definirse varios subproyectos.

Una de las mecánicas más útiles para el desarrollo de un proyecto en son las peticiones y su visualización. Estas peticiones se dividen en 3 tipos: errores, tareas y soporte las mismas pueden asignarse a un miembro del proyecto. Se puede indicar una fecha de inicio y fin para esa petición, e incluso llevar un control del tiempo y porcentaje realizado. Con todos estos datos, pueden visualizarse las peticiones de manera personalizada estableciendo filtros, y servir así de informes de tareas o incidencias.

Incluye un calendario para visualizar todas las peticiones a lo largo de un mes elegido, marcando claramente el día de inicio y de fin de cada petición. Igualmente ocurre con la vista en diagrama de Gantt, que va marcando el porcentaje completado conforme avanzan los días.

Algunas otras funcionalidades que habría que destacar son la página personal de cada usuario, que ofrece una vista personalizable con información de todos los proyectos donde está participando, como un calendario global, o peticiones asignadas.

Este gestor es el utilizado por el proyecto ERP-Cuba pero el mismo no satisface todas las necesidades del seguimiento del proyecto ya que éste permite definir tareas y visualizar su seguimiento pero no solo

referentes a la producción, permite todo tipo de tareas incluyendo planificación de clases entre otras. Además no aplica métricas y no permite tener un estado real del avance del proyecto teniendo en cuenta las tareas que cada requisito debe cumplir para darse por terminado, teniendo en cuenta la fecha establecida para la entrega.

1.3. Metodologías y herramientas propuestas para el Análisis y Diseño del sistema

El Proyecto Sistema Integral de Gestión CedruX precisó el marco de trabajo, en estas condiciones, a la herramienta para el seguimiento de proyectos de la vista arquitectónica de integración se le realizará el análisis y diseño siguiendo el marco de trabajo propuesto, el cual define las metodologías y herramientas a utilizar.

A continuación se hará un pequeño estudio acerca de las diferentes tecnologías, metodologías y herramientas, centrando particular interés en las más adecuadas para dar solución a la problemática planteada y que se ajustarán en la construcción de dicho sistema.

1.3.1. Metodología de desarrollo de Software

La metodología de desarrollo de software es un marco de trabajo usado para estructurar, planificar y controlar el proceso de desarrollo en sistemas de información.

Para la realización de este sistema se estructuró un Modelo de Desarrollo, pues para llevar a cabo un proyecto es necesario que cada uno de las personas involucradas en él, posean un modelo estandarizado, una definición clara y precisa de las responsabilidades de cada uno en el desarrollo de la solución.(7)

Teniendo en cuenta el Modelo de Desarrollo establecido y el alcance de esta investigación, los artefactos a generar durante la realización de este trabajo son:

- ✓ Mapa de procesos del negocio.
- ✓ Descripción de procesos del negocio.
- ✓ Especificación de requisitos de software.

- ✓ Prototipo de interfaz de usuario.
- ✓ Modelo conceptual.

RUP (Rational Unified Process)

El Proceso Unificado es un marco de trabajo genérico que puede especializarse para una gran variedad de sistemas de software, para diferentes áreas de aplicación, diferentes tipos de organizaciones y diferentes tamaños de proyecto.

RUP propone cuatro fases para todo el proceso, dentro de estas se realizan varios flujos de trabajo, en dependencia del esfuerzo en cada fase. Esta metodología cuenta con nueve flujos de trabajo, seis de ingeniería y tres de soporte. En la figura siguiente se muestra cómo varía el esfuerzo asociado a las disciplinas según la fase en la que se encuentre el proyecto. Los flujos de trabajo propuesto son los siguientes:

Flujos de ingeniería:

- ✓ Modelamiento del negocio: Se describe el negocio, define las actividades a automatizar y el personal que participa en ellas.
- ✓ Levantamiento de requisitos: Se define que debe hacer el sistema, identificando las funcionalidades que desea el usuario y las restricciones que esto impone.
- ✓ Análisis y diseño: describe como el sistema debe cumplir las funcionalidades previstas y las restricciones, que detalladamente indican que es lo que se va a programar.
- ✓ Implementación: se comienza a desarrollar el sistema en términos de clases y objetos.
- ✓ Pruebas: se identifican y eliminan los errores y defectos surgidos durante el proceso de desarrollo.
- ✓ Despliegue: se pone un reléase del producto en manos de los usuarios finales.

Flujos de soporte:

- ✓ Gestión de configuración y cambios: se describe como llevar a cabo el control de los artefactos generados en el proyecto, de manera que se mantenga al tanto a todo el equipo con las últimas versiones.
- ✓ Gestión de proyectos: engloba las actividades encaminadas a lograr un producto que realmente satisfaga las expectativas del cliente.
- ✓ Gestión de entornos: contiene las actividades que describen procesos y herramientas que empleará el equipo de trabajo y el proceso para llevar a cabo dichas actividades en la organización. (Ivar Jacobson).

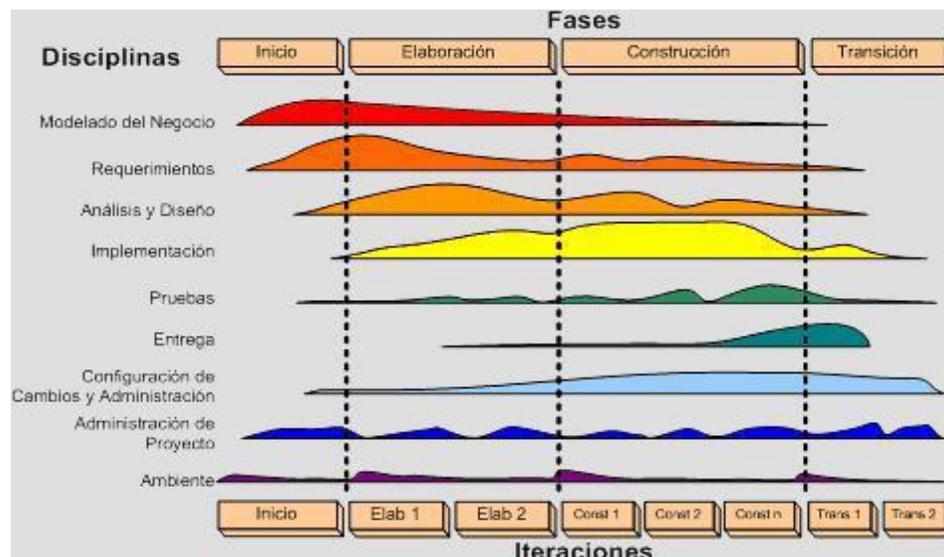


Figura 1 Fases e iteraciones de la Metodología RUP

Notación para el modelado de procesos del negocio (BPMN)

El modelado del negocio es la técnica para alinear los desarrollos con las metas y objetivos de las empresas e instituciones.

BPMN es un estándar que contiene los elementos requeridos para el modelado de procesos de negocio. Actualmente es aceptado como la principal notación estándar, ya que puede capturar todos los detalles relevantes de un proceso de negocio según su perspectiva, comunicarse con facilidad y comprensión.

Es importante por la capacidad de estandarización que puede ejercer en las organizaciones, proporcionando normas para acelerar la transición entre el modelado, y la automatización de procesos de negocio.

Ventajas

- ✓ La notación gráfica BPMN es simple, fácil, y comprensible.
- ✓ Contiene una representación gráfica que puede manejar ambos contextos: modelado y automatización.
- ✓ Apoya el contexto ejecutable, sin modificar el modelo de procesos.
- ✓ Fue creado con énfasis en SOA (Arquitectura Orientada en Servicios).
- ✓ Permite expresar con precisión excepciones de negocio, subprocessos, participantes, etc.
- ✓ Se ha convertido en el estándar de proveedores como Lombardi, Savvion, Apia, TIBCO, Oracle, IBM, SAP, etc.
- ✓ BPMN es mucho más que un diagrama de flujo, ya que puede ser ejecutado por las áreas técnicas.

La herramienta a desarrollar se realizará bajo el marco de trabajo del proyecto ERP-Cuba el cual no define una metodología de desarrollo de software específica pero si utiliza las bases de RUP en cuanto a fases y flujos de trabajo y BPMN para el modelado de procesos.

1.3.2. Herramientas CASE

Las Herramientas CASE (Computer Aided Software Engineering) son diversas aplicaciones informáticas destinadas a aumentar la productividad en el desarrollo de software y reducir el coste de las mismas en términos de tiempo y dinero. Estas herramientas permiten modelar y documentar los artefactos, cubriendo el ciclo de vida del proceso de desarrollo de software. Algunas herramientas Case conocidas son el ArgoUML, Rational Rose, Visual Paradigm, Easy CASE, CASE Studio 2, CASEWise entre otras.

Visual Paradigm

Es una herramienta case que utiliza “UML”: como lenguaje de modelado. Esta herramienta ha sido diseñada para un gran número de usuarios, incluyendo ingenieros de sistemas, analistas de sistemas, analistas de negocio y arquitectos de sistemas, tiene unas características gráficas muy cómodas que facilitan la realización de los diagramas de modelado que sigue el estándar de UML. Permite dibujar todos los tipos de diagramas de clases, código inverso, generar código desde diagramas y generar documentación (PARADIGM, 2007). Además se integra con diversos IDE`s (Integrated Development Environment) como: NetBeans (de Sun), JDeveloper (de Oracle), Eclipse (de IBM) y JBuilder (de Borland). Es desarrollada como software libre, es compatible con las metodologías existentes para el desarrollo de software y muy sencilla de usar, fácil de instalar y actualizar. (6)

Otros de los beneficios del Visual Paradigm son:

- ✓ Soporta UML.
- ✓ Interoperabilidad con modelos UML2 (meta modelos UML 2.x para plataforma Eclipse) a través de XMI.
- ✓ Editor de Detalles de Casos de Uso.
- ✓ Generación de objetos Java desde la base de datos.
- ✓ Importación y exportación de ficheros XMI.
- ✓ Navegación intuitiva entre el modelo visual y el código.
- ✓ Poderosa herramienta de generación de PDF/HTML a partir de diagramas UML.
- ✓ Sincronización entre el código fuente y el modelo en tiempo real.

El marco de trabajo del ERP-Cuba define Visual Paradigm como la herramienta case, por lo que será la utilizada para la realización de los diagramas de la herramienta a desarrollar.

Framework para el diseño de la interfaz: ExtJS

ExtJS es una biblioteca de JavaScript para el desarrollo de aplicaciones web interactivas usando tecnologías como AJAX, DHTML y DOM. [8] Originalmente construida como una extensión de la biblioteca YUI (Yahoo User Interface), en la actualidad puede usarse como extensión para las bibliotecas jQuery y Prototype. Con el tiempo se convirtió en un Framework independiente y a principios de 2007 se creó una compañía para comercializar y dar soporte del Framework Ext. De esta forma Ext tiene dos tipos de licencias, LGPL y comercial. Extjs soporta:

- ✓ Internet Explorer 6+.
- ✓ FireFox 1.5+ (PC, Mac).
- ✓ Safari 3+ (PC, Mac).
- ✓ Opera 9+ (PC, Mac).

Algunas de las cualidades de Extjs son:

- ✓ Alto rendimiento.
- ✓ Interfaces personalizables.
- ✓ Muy buena arquitectura que permite extender los componentes gracias a su buen modelado.
- ✓ API fácil de utilizar.
- ✓ Licencia comercial y open source (gratis).

La herramienta a desarrollar se realizará bajo el marco de trabajo del proyecto ERP-Cuba el cual define como herramienta case el Visual Paradigm y para el diseño de la interfaz ExtJS.

1.3.3. Aplicaciones Web

Una aplicación web es una aplicación informática basada en la arquitectura Cliente-Servidor, que los usuarios pueden utilizar desde cualquier lugar mediante un navegador, accediendo a un servidor web a través de internet o de una intranet, permitiendo utilizar varios servicios de la red lo cual no sucede con

aplicaciones de escritorio. Estas funcionan independientemente de la versión del sistema operativo instalada en el cliente.

A continuación se muestran algunas de las ventajas y desventajas que presentan las mismas:

Ventajas (Montalvo, 2008-2009)

- ✓ No se necesita instalar programas (No depende de algún Software).
- ✓ Puedes acceder a ella desde cualquier ordenador con conexión a Internet.
- ✓ Brindan privacidad con acceso (usuario y contraseña) para acceder a tus datos.
- ✓ No necesitas actualizarlo.
- ✓ Fácil utilización.
- ✓ Comunican las empresas.

Desventajas (Franulic, 2007)

- ✓ La seguridad de datos confidenciales.
- ✓ La conexión a Internet.
- ✓ Versatilidad y potencia (limitaciones de HTML) y necesidad de buenas comunicaciones.
- ✓ Cambios en las condiciones del servicio.

1.4. Ingeniería de requisitos

La Ingeniería de requisitos tiene lugar durante todo el ciclo de vida del software, fundamentalmente en las primeras etapas cuando es necesario encontrar y comunicar las necesidades de clientes y usuarios.

La obtención correcta de los requisitos puede llegar a describir con claridad el comportamiento de un sistema. De tal manera que, basarse en la extracción de requisitos y sobre todo en que sean correctos,

proporciona minimizar los problemas relacionados al desarrollo de sistemas, logrando la reducción de tiempo en la construcción, así como la reducción de errores.

“La Ingeniería de requisitos facilita el mecanismo apropiado para comprender lo que quiere el cliente, analizando necesidades, confirmando su viabilidad, negociando una solución razonable, especificando la solución sin ambigüedad, validando la especificación, y gestionando los requisitos para que se transformen en un sistema operacional.” (Pressman R. S., 2005)

1.4.1. Actividades de la Ingeniería de requisitos

En la actualidad persisten problemas en el desarrollo de software, entre ellos, un inadecuado entendimiento de las necesidades de los usuarios, incapacidad de absorber cambios en los requisitos e insatisfacciones de los clientes por inaceptable o bajo desempeño del software. Los requisitos son cada vez más complejos debido a su grado de proliferación, diversificación y conectividad.

Para la realización de este proceso se realizan varias actividades entre las que se encuentran: elicitación, análisis y negociación, especificación, validación y gestión de requisitos; encontrándose vinculadas pues se utiliza la actividad precedente para la realización de la próxima, aplicándose de manera continua e iterativa. Estas actividades consisten en:

- ✓ Elicitación de Requisitos.
- ✓ Gestión de requisitos.
- ✓ Análisis y negociación.
- ✓ Especificación de Requisitos.
- ✓ Validación de Requisitos.

Validación de Requisitos: Se comprueba que la especificación de requisitos se ajuste a las necesidades del cliente verificando que las necesidades hayan sido adecuadamente interpretadas. Realizándose con el fin de comprobar la consistencia, completitud, corrección, precisión del documento, así como el

descubrimiento de problemas en él, antes de comprometer recursos en su implementación. (Dra. García L., Ing. Fernández L. y MSc. Aguillón E., 2008)

La validación de requisitos es la etapa final de la Ingeniería de Requisitos y tiene como objetivo ratificar los requerimientos, es decir, verificar que todos los requerimientos que aparecen en el documento especificado representan una descripción, por lo menos, aceptable del sistema que se debe implementar. Esto implica verificar que los requerimientos sean consistentes y que estén completos.

Existen varias técnicas de validación de requisitos, algunas de ellas son:

- ✓ **Listas de chequeo:** Son frecuentemente usadas en inspecciones o revisiones de artefactos generados en el proceso de producción de software; son listas de aspectos que deben ser completados o verificados.
- ✓ **Auditorías:** La revisión de la documentación con esta técnica consiste en un chequeo de los resultados contra una checklist (Listas de Chequeo), predefinida o definida a comienzos del proceso.
- ✓ **Prototipos:** Algunas propuestas se basan en obtener prototipos de la definición de requisitos que, sin tener la totalidad de las funcionalidades, permitan al usuario hacerse una idea de la estructura de la interfaz del sistema.

1.5. Arquitectura utilizada

Existen varias definiciones de Arquitectura de Software pero ninguna es respaldada unánimemente por la totalidad de los arquitectos. Una de las más reconocidas es la de Clements: La Arquitectura de Software es, a grandes rasgos, una vista del sistema que incluye los componentes principales del mismo, la conducta de esos componentes según se la percibe desde el resto del sistema y las formas en que los componentes interactúan y se coordinan para alcanzar la misión del sistema [Cle96a].

Una Arquitectura de Software, consiste en un conjunto de patrones y abstracciones coherentes que proporcionan el marco de referencia necesario para guiar la construcción del software para un sistema de

información, establece los fundamentos para que analistas, diseñadores, programadores, etc. trabajen en una línea común que permita alcanzar los objetivos del sistema, cubriendo todas las necesidades.

La arquitectura de software define, de manera abstracta, los componentes que llevan a cabo alguna tarea de computación, sus interfaces y la comunicación entre ellos.

Patrón de Arquitectura MVC (Modelo-Vista-Controlador)

Los patrones expresan el esquema fundamental de organización para sistemas de software. Proveen un conjunto de subsistemas predefinidos; especifican sus responsabilidades e incluyen reglas y guías para organizar las relaciones entre ellos; así como ayudan a especificar la estructura fundamental de una aplicación.

MVC es un patrón de arquitectura de software clásico que está formado por tres niveles:

- ✓ **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) es decir, su lógica de negocio.
- ✓ **Vista:** Maneja la visualización de la información.
- ✓ **Controlador:** Controla el flujo entre la vista y el modelo (los datos).

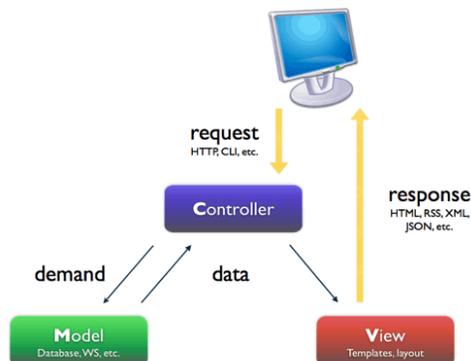


Figura 2 Patrón MVC

Este modelo de arquitectura presenta varias ventajas y desventajas:

Ventajas

- ✓ Soporte de múltiples vistas: Dado que la vista se halla separada del modelo y no hay dependencia directa del modelo con respecto a la vista, la interfaz de usuario puede mostrar múltiples vistas de los mismos datos simultáneamente.
- ✓ Adaptación al cambio: 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 o PDAs. Dado que el modelo no depende de las vistas, agregar nuevas opciones de presentación generalmente no afecta al modelo.
- ✓ La conexión entre el Modelo y sus Vistas es dinámica; se produce en tiempo de ejecución, no en tiempo de compilación.

Desventajas

- ✓ Una desventaja que tiene este modelo es el costo de actualizaciones frecuentes: Si el modelo experimenta cambios frecuentes, por ejemplo, podría desbordar las vistas con una lluvia de requerimientos de actualización.

El patrón de arquitectura “Modelo Vista Controlador” provee las características apropiadas para darle uso en el diseño y desarrollo de la herramienta propuesta por las características antes expuestas.

1.6. Patrones de diseño

Los patrones de diseño surgen ante la necesidad de la reutilización de diseños y de soluciones a problemas frecuentes encontrados por los ingenieros. Con la reutilización se consigue la reducción de tiempos y la disminución del esfuerzo de mantenimiento, esto trae consigo mayor eficiencia y consistencia

en el diseño de la solución. Las características que presentan los patrones son las siguientes: son soluciones concretas, proponen soluciones a problemas concretos, no son teorías genéricas.

Son soluciones técnicas. Indican resoluciones técnicas basadas en Programación Orientada a Objetos (POO). En ocasiones tienen más utilidad con algunos lenguajes de programación y en otras son aplicables a cualquier lenguaje.

Se utilizan en situaciones frecuentes, ya que se basan en la experiencia acumulada al resolver problemas reiterativos.

Favorecen la reutilización de código. Ayudan a construir software basado en la reutilización, a construir clases reutilizables. Los propios patrones se reutilizan cada vez que se vuelven a aplicar. (9)

El uso de un patrón no se refleja en el código. Al aplicar un patrón, el código resultante no tiene por qué delatar el patrón o patrones que lo inspiró.

Los patrones GRASP son los siguientes:

- ✓ Experto.
- ✓ Creador.
- ✓ Controlador.
- ✓ Bajo Acoplamiento.
- ✓ Alta Cohesión

Bajo Acoplamiento: Promueve la baja dependencia y una alta reutilización de las clases. Asigna responsabilidades manteniendo un bajo acoplamiento. Soporta el diseño de clases más independientes, que reducen el impacto de cambios que acrecientan la reutilización. Uno de los principales síntomas de un mal diseño y alto acoplamiento es una herencia muy profunda. Siempre hay que considerar las ventajas de la delegación respecto de la herencia.

Alta Cohesión: La alta cohesión al igual que el bajo acoplamiento logran un diseño fácil de reutilizar y adaptar, más legibilidad para los programadores y diseñadores en general; la extensibilidad y flexibilidad del diseño aumentan, ya que las clases usan solo lo que necesitan y en su mayoría son recursos propios, por otro lado, cada cual hace lo que le corresponde hacer según la parte del problema que esta modelando (LARMAN 1998).

1.7. Conclusiones del capítulo.

En el presente capítulo se abordó acerca de las métricas, sus características e importancia así como técnicas de seguimiento basadas en métricas, parámetros de seguimiento, etc. Se definió la Métrica de Avance para el análisis y diseño de la herramienta. Se estudiaron los software a nivel mundial que aplican métricas para el seguimiento de proyectos, concluyendo que no es factible la utilización de ninguno de ellos. Se realizó un estudio de los patrones de diseño, se definió la metodología a utilizar siendo la propuesta por el marco de trabajo del proyecto ERP-Cuba y se definieron las actividades a realizar en la Ingeniería de requisitos.

Capítulo #2. Características del sistema

2.1. Introducción

Es importante detenerse a analizar las características y organización de los procesos que se desarrollan en el área que se pretende automatizar, se hace con el fin de lograr una mayor comprensión del problema que se desea resolver. Con este propósito se realiza la modelación del negocio y también con la intención de establecer el entendimiento entre clientes y desarrolladores. En el presente capítulo se describe la Métrica de avance, se realiza la modelación del negocio, y además se analizan tanto los requerimientos funcionales como los no funcionales.

2.2. Métrica de Avance

Para conocer el estado de un software es necesario medir el avance durante su desarrollo. Este puede expresarse como el por ciento de cumplimiento de las tareas de la fase en que se encuentra. Por esta razón, para medir el avance es necesario considerar las tareas de la fase de acuerdo a su complejidad y los requisitos a desarrollar.

De acuerdo a la arquitectura definida para desarrollar una solución de software se definen los roles y actividades que se requieren para obtener el producto. En una arquitectura por capas podría ser necesarios roles específicos para realizar cada una de las capas arquitectónicas y en función de la complejidad de estas tareas el tiempo que se requiere para terminar un requisito puede variar.

A cada una de las tareas se le asignó un peso, que representa el por ciento del tiempo total que toma su ejecución como parte de la culminación de un requisito. Por tanto, la suma de todos los pesos resultaría un 100%. Estos pesos se fueron ajustando a lo largo de las iteraciones del desarrollo para que expresaran con mayor fidelidad la complejidad de las tareas y el esfuerzo en tiempo requerido para ejecutarlas.

La realización de estas tareas hace posible que se realicen los propósitos de la fase. Desde el punto de vista de los requisitos, ninguno habrá concluido hasta tanto no hayan sido realizadas cada una de las actividades anteriores.

Por este motivo a cada una de las tareas definidas se le otorga, para cada requisito, un Estado de completamiento entre 0 y 1: 0 si aún no se ha realizado y 1 si ya la tarea se encuentra terminada; un valor decimal que se encuentre entre estos dos números representa el progreso de la tarea dado por el responsable de ejecutarla y basado en las actividades definidas dentro de su flujo de trabajo que aún le restan por terminar.

Así el estado de avance de un requisito puede definirse como:

$$n$$

Avance por Requisito (AR_i) = $\sum_{j=1}^n$ (Estado de completamiento_j × Peso de la tarea_j)

$$j=1$$

Donde: Peso de la tarea corresponde al valor asignado a la tarea (j).

Otra dimensión a analizar para establecer el avance de la fase es la complejidad de los requisitos a desarrollar. La existencia de requisitos de diferente complejidad implica que el tiempo de realización de una tarea no sea el mismo para todos los casos. Por ello, para medir el avance de un módulo conformado por varios requisitos, sería erróneo calcularlo como el promedio del avance de estos.

Los requisitos se clasifican según su complejidad en: alto, medio o bajo. Esta clasificación es concedida por el Diseñador en función de los siguientes aspectos:

- Número de elementos de datos.
- Lógica de negocio.
- Interfaces externas, dígame dispositivos u otro sistema.

En un módulo solo se presenta una de las combinaciones posibles de complejidad de los requisitos que lo conforman. Para cada combinación se estableció un peso específico, valor llamado Peso de la complejidad del requisito (P_{cr}) y es el que permite ponderar la diferencia de esfuerzo para desarrollar los requisitos de diferente complejidad.

Tabla 1 Relación de los Pesos de la complejidad del requisito según las combinaciones posibles

Caso	Combinación posible	Peso R complejidad alto (Pcr alta)	Peso R complejidad medio (Pcr media)	Peso R complejidad bajo (Pcr baja)
1	Existen requisitos con complejidad alta, media y baja.	0,5	0,3	0,2
2	Todos los requisitos tienen complejidad alta.	1	0	0
3	Existen requisitos con complejidad alta y media.	0,6	0,4	0
4	Existen requisitos con complejidad alta y baja.	0,8	0	0,2
5	Todos los requisitos tienen complejidad media.	0	1	0
6	Existen requisitos con complejidad media y baja.	0	0,6	0,4
7	Todos los requisitos tienen complejidad baja.	0	0	1

Utilizando las posibles combinaciones de complejidad de los requisitos que componen un módulo y partiendo del valor del avance de cada requisito, es posible determinar el avance total del módulo. Esto se obtendría mediante la suma del producto entre el Peso de complejidad y el promedio de avance de los requisitos para cada uno de los niveles de complejidad, lo que quedaría representado por la siguiente expresión:

$$\text{Avance por Módulo (AM)} = \text{Pcr alto} \times \text{PAr alto} + \text{Pcr medio} \times \text{PAr medio} + \text{Pcr bajo} \times \text{PAr bajo}$$

Donde PAr alto es el promedio de avance de los requisitos de complejidad alta, PAr medio es el promedio de avance de los requisitos de complejidad media y PAr bajo es el promedio de avance de los requisitos de complejidad baja.

Para determinar el avance de la fase, se puede considerar aceptable, el promedio de avance de cada módulo.

Avance del Proyecto = $\frac{\sum_{i=1}^n \text{Avance del Módulo}}{n}$, donde n representa la cantidad de módulos.

La siguiente tabla muestra un resumen, para una semana dada, de la situación del proyecto, donde se puede observar la lista de los módulos que corresponden a la iteración y la cantidad de funcionalidades (requisitos) a implementar en cada uno de ellos.

Estado Actual del Desarrollo					
Total de semanas	8				
Semana actual	7				
Semanas que quedan	1				
Módulos	Funcionalidades	Responsable	Avance	Semanas para terminar	Estado del desarrollo
Importador	1		100%	0,0	En tiempo
Editor de Ecuaciones	6		97%	0,2	En tiempo
Vistas	4		92%	0,6	En tiempo
Soporte	2		91%	0,7	En tiempo
Omisiones	2		59%	4,9	Atrasado
Muestreo	2		92%	0,6	En tiempo
Historial	1		37%	11,9	Atrasado
Gráficos	1		92%	0,6	En tiempo
Generales	8		65%	3,7	Atrasado
Extracciones	2		37%	11,8	Atrasado
Extratificaciones	4		33%	14,2	Atrasado
Duplicados	1		81%	1,6	Atrasado
Carpetas	4		79%	1,9	Atrasado
Buscar	3		61%	4,5	Atrasado
Antigüedad	1		42%	9,7	Atrasado
Análisis estadístico	2		92%	0,6	En tiempo
Grid dinámico	1		95%	0,4	En tiempo
Persistidor de datos	1		95%	0,4	En tiempo

Figura 3 Resumen del avance por módulos para una semana dada

Se puede observar que la tabla expresa las semanas que necesita el módulo para terminar. De los datos anteriores se puede establecer la siguiente relación:

$$\frac{\text{Total Semanas}}{100\%} = \frac{\text{Semana Actual}}{\text{Avance por Módulo}}$$

Para obtener las semanas que quedan para terminar, suponiendo que se mantiene el ritmo de trabajo, se puede realizar la siguiente consideración:

$$\text{Semanas para Terminar} = \frac{\text{Semana Actual} \times 100\%}{\text{Avance por Módulo}} - \text{Semana Actual}$$

Nótese que existe otro campo en la tabla cuyo nombre es Estado de Desarrollo. El valor por módulo de este campo depende totalmente de los resultados arrojados en el cálculo anterior. Si el resultado obtenido en Semanas para Terminar supera las semanas que quedan para la fecha de entrega establecida por el cliente, el módulo está *Atrasado*, esto significa que deben tomarse decisiones derivadas de esta información que permitan efectuar la entrega en la fecha acordada. En el caso contrario, es decir, si el resultado obtenido es menor que las semanas que quedan para la fecha de entrega establecida entonces el módulo está *En Tiempo*.

Análisis de los resultados

La forma más cómoda y visual de lograr un seguimiento del avance por módulos, es a través del análisis del gráfico del estado de avance. Este gráfico muestra claramente en qué estado de avance se encuentran los módulos y por tanto advierte al jefe del proyecto en cuáles tiene que volcar los mayores esfuerzos.

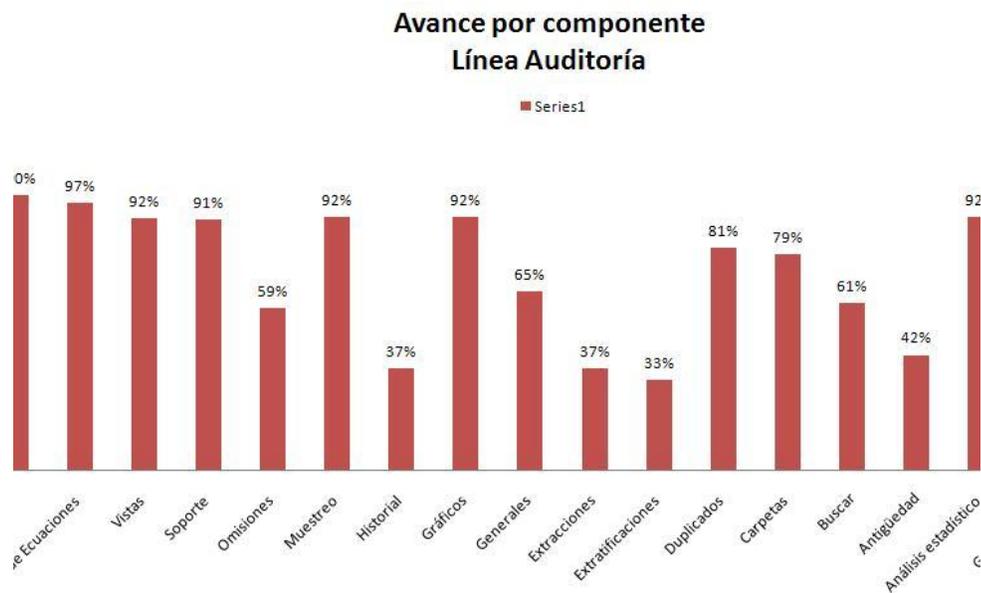


Figura 4 Estado de avance por módulos

En la siguiente figura se muestra un histórico por semanas del progreso de los diferentes módulos y total del proyecto durante el proceso de implementación del software.

		semana1	semana2	semana3	semana4	semana5	semana6	semana7	semana8	semana 9
No	Módulos	7-3-2008	14-3-2008	21-3-2008	28-3-2008	4-4-2008	11-4-2008	18-4-2008	25-4-2008	2-5-2008
1	Importador	24%	34%	57%	62%	81%	89%	92%	92%	98%
2	Editor de Ecuaciones	13%	41%	42%	42%	79%	91%	93%	93%	98%
3	Vistas	48%	60%	77%	81%	93%	95%	96%	96%	96%
4	Soporte	44%	46%	53%	57%	73%	77%	85%	86%	97%
5	Omisiones	30%	30%	41%	41%	50%	66%	82%	86%	90%
6	Muestreo	25%	47%	67%	75%	84%	92%	93%	93%	97%
7	Historial	7%	18%	18%	19%	34%	58%	71%	86%	85%
8	Gráficos	11%	32%	55%	56%	93%	94%	95%	95%	95%
9	Generales	18%	48%	61%	76%	89%	89%	91%	92%	92%
10	Extracciones	15%	49%	49%	55%	77%	80%	86%	86%	86%
11	Extratificaciones	37%	61%	76%	79%	80%	85%	85%	91%	91%
12	Duplicados	26%	53%	78%	83%	88%	91%	91%	91%	97%
13	Carpetas					26%	31%	46%	65%	65%
		25%	43%	56%	61%	73%	80%	85%	89%	91%

Figura 5 Histórico del avance por semanas

El avance general del proyecto a partir de los datos de cada semana se representaría gráficamente de la siguiente manera.

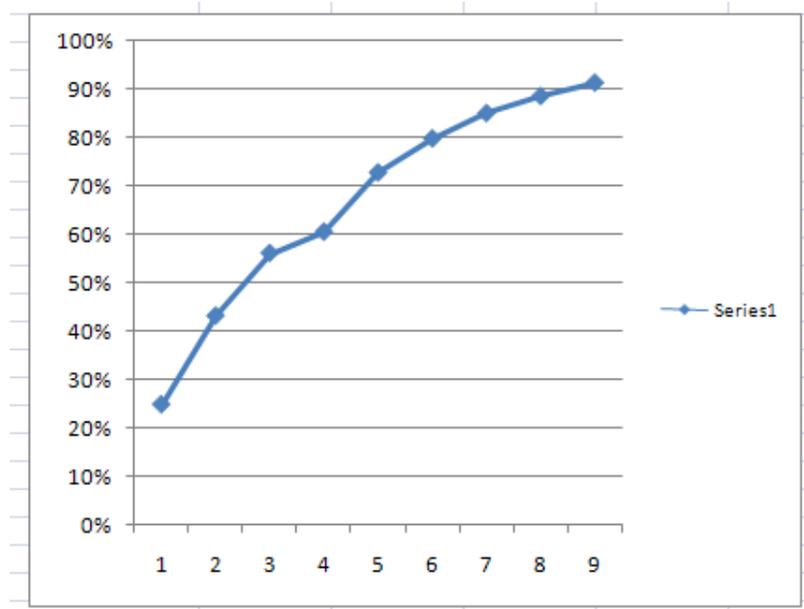


Figura 6 Histórico del avance del proyecto

Esta gráfica permite observar las tendencias a lo largo del tiempo. Nótese que durante las primeras semanas del desarrollo existe un avance relativo alto. Luego se observa un menor avance en la semana 4 respecto a la semana 3, período durante el cual se efectuaban en la Universidad los IV Juegos Deportivos, los que afectaron como mostró la métrica, el proceso de desarrollo, debido a que la mayoría de los miembros del equipo de proyecto son estudiantes. Luego se observa nuevamente un avance relativamente alto hasta aproximadamente el 80% punto en el cual comienza la etapa de pruebas y donde el avance fue más lento.

Un comportamiento ideal estaría representado por una línea recta en la gráfica del avance general del proyecto; pero esto solo es posible bajo condiciones ideales, las cuales son muy difíciles de obtener debido entre múltiples factores, a que los miembros del grupo de desarrollo no tienen la experiencia suficiente y no se dedican al trabajo a tiempo completo. Además las variables de ajustes introducidas al método deben ser evaluadas para ganar en exactitud. Un ejemplo de esto lo demuestra la desaceleración del avance durante la etapa de pruebas, lo que podría indicar que el peso dado a esta fase debería ser mayor con respecto al otorgado al resto de las tareas.

Lo anterior determina que el método propuesto no es exacto, ya que en él influyen múltiples factores y las variables son cambiantes; sin embargo, su aplicación arrojó resultados que en la práctica no se alejaron de la realidad y el proyecto cumplió en el tiempo previsto sus compromisos productivos, en parte porque pudieron ser tomadas a tiempo las medidas gerenciales pertinentes cuando se detectaron desviaciones.

2.3. Mapa de procesos

El mapa de procesos permite conocer cómo se entrelazan los distintos procesos por los que pasa una determinada institución, para lograr un objetivo específico. En este artefacto se muestran las entradas que necesita cada proceso para desarrollarse, así como las salidas generadas por los mismos. Con un mapa de procesos se puede explicar de manera global el funcionamiento de los procesos existentes, pues este se realiza en función de las relaciones entre ellos mediante los elementos considerados como entradas y salidas.(10)

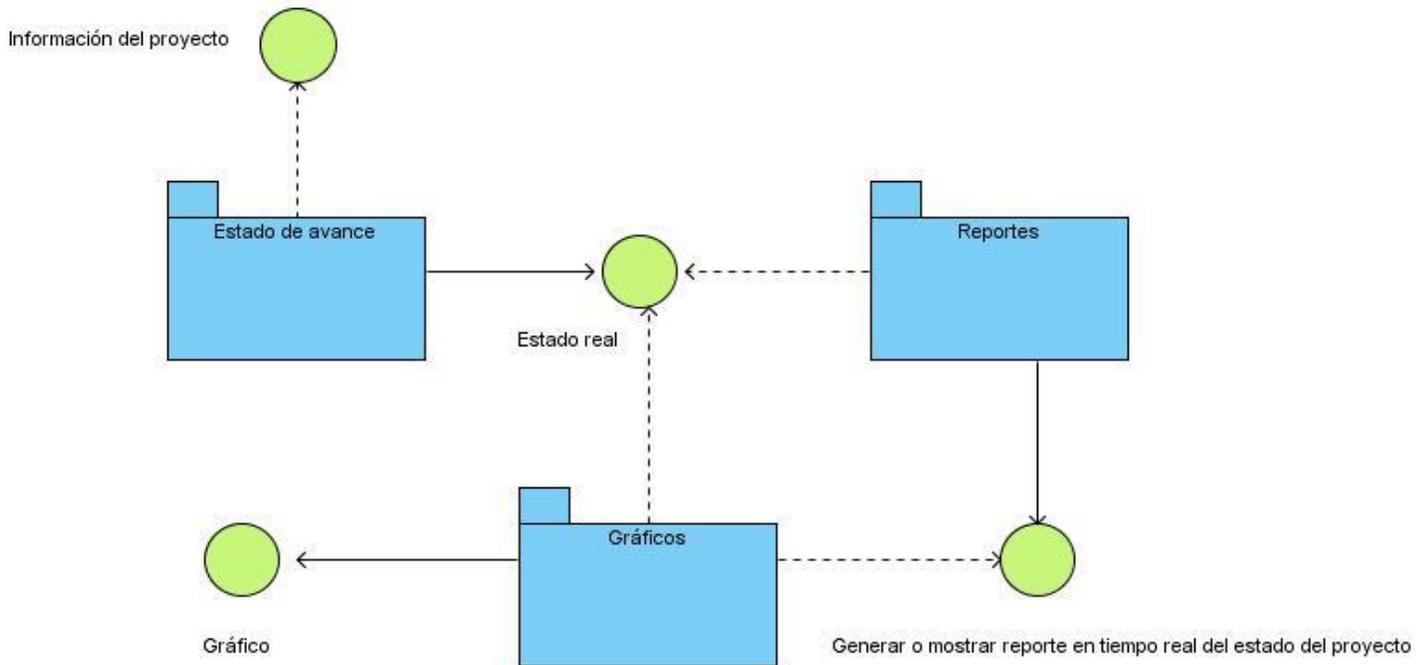


Figura 7 Mapa de procesos del negocio

El proceso Estado de avance tiene como entrada toda la información referente al proyecto lo que genera como salida un resultado del estado en tiempo real de las funcionalidades definidas a desarrollar, este resultado es consultado por el proceso Reportes el cual es el encargado de mostrar la información obtenida por el estado de avance, a partir de estos reportes o del estado real se generan Gráficos en los cuales se evidencia una mayor visibilidad del resultado.

2.3.1 Descripción de procesos

La descripción de los procesos del negocio son técnicas por excelencia para alinear el desarrollo con las empresas y entidades, son la base para comprender mejor la forma de operar de una organización y documentar con eficiencia los procesos; la documentación es una de las tareas más importantes cuando se habla de procesos del negocio, pues con esta se logra una mejora en el entendimiento de los mismos, quedando persistencia del análisis realizado.

Descripción del proceso de negocio: Estado de avance

Mediante este proceso se permite acceder a toda la información necesaria para conocer el estado de avance de los proyectos, los módulos, las tareas, en el mismo se gestiona toda la información referente al proyecto en cuestión, se evidencia en dos definiciones tomando como punto de partida la fecha inicial y final de terminación del proyecto, la primera se define En tiempo cuando el cálculo de la cantidad de semanas para terminar arroja un resultado menor que las semanas establecidas para la fecha de entrega y el segundo se define como Atrasado cuando el cálculo de la cantidad de semanas para terminar arroja un resultado mayor que las semanas establecidas para la fecha de entrega.

Descripción del proceso de negocio: Reportes

Mediante este proceso se permite mostrar en tiempo real los resultados de las operaciones realizadas a través de reportes. Este permite mostrar el estado de avance en que se encuentra el proyecto, el por ciento de avance que tienen los módulos por semanas, el por ciento de avance del proyecto, el estado de avance por módulos del proyecto así como el por ciento de avance de los módulos. Con esto se obtiene una información precisa de la situación real de los proyectos.

Descripción del proceso de negocio: Generar gráficos

Mediante este proceso se permite graficar los reportes del por ciento de avance en el proyecto, para poder obtener una mayor visibilidad del comportamiento de cada uno de los resultados que se obtuvieron y se mostraron anteriormente.

2.4. Captura y descripción de requisitos

Con la especificación de requisitos del software se pueden definir los requisitos capturados y analizados, de manera que estos puedan ser fuente de información para los desarrolladores, clientes, líderes etc., garantizándose que su legibilidad posibilite un fácil entendimiento para todos los interesados. La especificación de requisitos contiene la descripción del requisito funcional en cuestión, en el Proyecto ERP-Cuba se definió una plantilla para la descripción de requisitos la cual es factible utilizar para la realización de los mismos.

2.4.1. Requisitos Funcionales

Los requerimientos funcionales son aquellas capacidades o condiciones que el sistema debe cumplir. Ellos definen qué es lo que el sistema debe hacer y permiten identificar las funcionalidades requeridas.

A continuación se enumeran las funcionalidades que el sistema debe cumplir:

✓ **RF 1 Autenticar Usuario.**

- Autenticarse (usuario, contraseña).

✓ **RF 2 Gestionar Usuario.**

- 2.1 Crear usuario (Nombre del usuario, Rol, Proyecto al que pertenece).
- 2.2 Modificar usuario.
- 2.3 Eliminar usuario.

✓ **RF 3 Gestionar Proyecto.**

- 1.1 Crear proyecto (Nombre del proyecto, Facultad, Cantidad de semanas en que se debe realizar, Líder del proyecto, Cantidad de recursos humanos).
- 1.2 Modificar proyecto.
- 1.3 Eliminar proyecto.

✓ **RF 4 Gestionar Tarea.**

- 3.1 Crear Tarea (nombre de la fase, nombre de la tarea, peso de la tarea).
- 3.2 Modificar Tarea.
- 3.3 Eliminar Tarea.

✓ **RF 5 Gestionar Módulos.**

- 4.1 Crear módulo (nombre del módulo, nombre de la fase a la que pertenece, proyecto al que pertenece).
- 4.2 Modificar módulo.
- 4.3 Eliminar módulo.

✓ **RF 6 Gestionar Requisito.**

- 5.1 Crear requisito (nombre del requisito, Responsable, Complejidad, % de avance, nombre de la tarea, nombre de la fase a la que pertenece, módulo al que pertenece).
- 5.2 Modificar requisito.
- 5.3 Eliminar requisito.

✓ **RF 7 Reportes.**

- 6.1 Mostrar estado de avance del proyecto.
- 6.2 Mostrar por ciento de avance del proyecto.
- 6.3 Mostrar por ciento de avance del módulo por semanas.
- 6.4 Mostrar estado de avance por módulos del proyecto.
- 6.5 Mostrar por ciento de avance por módulo.

✓ **RF 8 Generar gráfico.**

- 7.1 Generar gráficos del por ciento de avance del proyecto por semanas.
- 7.2 Generar gráficos del por ciento de avance por módulos del proyecto.

Se definieron los siguientes conceptos:

Tarea: Son las actividades que se definen en dependencia de la fase en la que se desarrolle un proyecto.

Fase: Son las definidas por RUP para el ciclo de realización de un proyecto: inicio, elaboración, construcción y transición.

Módulo: Divisiones que se hacen a un proyecto, en subproyectos o módulos que definen el proyecto en cuestión.

Requisitos: Son las actividades o funcionalidades que se realizan en cada uno de los módulos que se definen, están determinados por el cliente.

Estos conceptos pueden ser adaptados en dependencia de la metodología a usar por cada proyecto.

2.4.2 Requisitos no Funcionales

Los requerimientos no funcionales son propiedades o cualidades que el producto debe tener. Son esas características que posibilitan que el producto sea atractivo, usable, rápido, confiable, etc. En muchos casos los requerimientos no funcionales son fundamentales en el éxito del producto. Normalmente están vinculados a requerimientos funcionales, es decir una vez que se conozca lo que el sistema debe hacer, se puede determinar cómo ha de comportarse, qué cualidades debe tener o cuán rápido o grande debe ser.

✓ Usabilidad

RNF1: La herramienta podrá ser usada por cualquier persona que posea conocimientos básicos de computación y trabajo en la web.

✓ Apariencia o interfaz externa

RNF2: La interfaz debe ser agradable para el usuario, que combine correctamente los colores, tipo de letra y tamaño, no debe contener muchas imágenes que demoren las respuestas al usuario. La navegabilidad debe ser sencilla, permitiendo el reconocimiento visual de las funcionalidades.

✓ **Rendimiento**

RNF3: La herramienta deberá ser capaz de gestionar la información y dar respuesta a las solicitudes lo mas rapido posible.

✓ **Seguridad**

RNF4: Se podrá acceder a la herramienta solamente después de autenticarse.

✓ **Portabilidad**

RNF5: La herramienta deberá ser multiplataforma, haciendo énfasis en Windows y Linux.

✓ **Soporte**

RNF6: La herramienta contará antes de su puesta en marcha con un período de pruebas.

✓ **Políticos Culturales**

RNF7: La herramienta solo podrá ser utilizada en territorio cubano y por las entidades autorizadas. No debe contener palabras en otros idiomas.

✓ **Software**

RNF8: Sistema operativo utilizando software libre o superior a Windows 98 y navegador Internet Explorer v6.0 o superior, Mozilla Firefox v2.x.

✓ **Hardware**

Para el cliente:

RNF9: Requerimientos mínimos: Procesador Pentium IV, 128 Mb de memoria RAM y capacidad de disco duro preferiblemente mayor de 20 GB.

2.4.3 Descripción de los requisitos funcionales

Tabla 1 Especificación del requisito Autenticar usuario

Precondiciones	El usuario debe pertenecer al dominio UCI, y estar previamente insertado en la base de datos.	
Flujo de eventos		
Flujo básico Autenticar usuario.		
1	El usuario accede a la página principal de la aplicación.	
2	El sistema muestra el formulario en el cual el usuario debe introducir los datos necesarios para ser identificado.	
3	El usuario introduce los datos que ha solicitado el sistema y presiona el botón Aceptar.	
4	El sistema verifica usuario y contraseña introducidos.	
5	Concluye el requisito.	
Poscondiciones:		
1	El sistema actualiza la base de datos.	
Pos-condiciones:		
1	El sistema almacena la información de las operaciones realizadas.	
Validaciones		
1	N/A	
Relaciones	Requisitos Incluidos	N/A
	Extensiones	N/A
Conceptos	Autenticar usuario	
	Autenticar usuario	Visibles en la interfaz: usuario, contraseña.
Requisitos especiales	N/A	
Asuntos	N/A	

pendientes

Tabla 2 Especificación del requisito Gestionar proyecto

Precondiciones	Tiene que haberse autenticado correctamente con los permisos necesarios.
Flujo de eventos	
Flujo básico Gestionar proyecto.	
6	Seleccionar la opción deseada para gestionar el proyecto.
7	Concluye el requisito.
Sección: Crear proyecto	
1	Seleccionar la opción crear proyecto.
2	El sistema muestra el formulario para introducir los datos necesarios para adicionar un nuevo proyecto.
3	El Líder de proyecto introduce los datos correspondientes y selecciona la opción Aceptar.
Poscondiciones:	
1	El sistema verifica los datos.
2	El sistema almacena los datos creados en la base de datos.
Sección: Modificar proyecto	
1	Seleccionar la opción modificar proyecto.
2	El sistema muestra el formulario con los datos necesarios para modificar un proyecto.
3	El Líder de proyecto busca el proyecto a modificar.
4	El Líder del proyecto modifica los datos y selecciona la opción Aceptar.
Poscondiciones:	
1	El sistema actualiza la base datos con los datos modificados.
Sección: Eliminar proyecto	
1	Seleccionar la opción eliminar proyecto.

2 El sistema muestra el formulario con los datos necesarios para eliminar un proyecto.		
3 El Líder de proyecto busca el proyecto a eliminar.		
4 El Líder del proyecto elimina el proyecto y selecciona la opción Aceptar.		
Poscondiciones:		
1 El sistema actualiza la base de datos.		
Pos-condiciones:		
1 El sistema almacena la información de las operaciones realizadas.		
Validaciones		
2 N/A		
Relaciones	Requisitos	N/A
	Incluidos	
	Extensiones	N/A
Conceptos	Gestionar proyecto	
	Gestionar proyecto	Visibles en la interfaz: Nombre del proyecto, Facultad, Cantidad de semanas en que se debe realizar, Líder del proyecto, Cantidad de recursos humanos.
Requisitos especiales	N/A	
Asuntos pendientes	N/A	

Tabla 3 Especificación del requisito Gestionar usuario

Precondiciones	Tiene que haberse autenticado correctamente con los permisos necesarios.
Flujo de eventos	
Flujo básico Gestionar usuario.	

1 Seleccionar la opción deseada para gestionar usuario.

2 Concluye el requisito.

Sección: Crear usuario

1 Seleccionar la opción crear usuario.

2 El sistema muestra el formulario para introducir los datos necesarios para adicionar un nuevo usuario.

3 El Líder de proyecto introduce los datos correspondientes y selecciona la opción aceptar.

Poscondiciones: El sistema verifica los datos.

El sistema almacena los datos creados en la base de datos.

Sección: Modificar usuario

1 Seleccionar la opción modificar usuario.

2 El sistema muestra el formulario con los datos necesarios para modificar un usuario.

3 El Líder de proyecto busca el usuario a modificar.

4 El Líder del proyecto modifica los datos y selecciona la opción aceptar.

Poscondiciones: el sistema actualiza la base datos con los datos modificados.

Sección: Eliminar usuario

1 Seleccionar la opción eliminar usuario.

2 El sistema muestra el formulario con los datos necesarios para eliminar un usuario.

3 El Líder de proyecto busca el usuario a eliminar.

4 El Líder del proyecto elimina el usuario y selecciona la opción aceptar.

Poscondiciones:

1 El sistema actualiza la bade datos.

Pos-condiciones:

1 El sistema almacena la información de las operaciones realizadas.

Validaciones N/A

1 N/A

Relaciones **Requisitos** N/A

Incluidos

	Extensiones	N/A
Conceptos	Gestionar usuario	
	Gestionar usuario	Visibles en la interfaz: Nombre del usuario, Rol, Proyecto al que pertenece.
Requisitos especiales	N/A	
Asuntos pendientes	N/A	

Tabla 4 Especificación del requisito **Gestionar tarea.**

Precondiciones	Tiene que haberse autenticado correctamente con los permisos necesarios.
Flujo de eventos	
Flujo básico Gestionar Tarea.	
1	Seleccionar la opción deseada para gestionar tarea.
2	Concluye el requisito.
Sección: Crear usuario	
1	Seleccionar la opción crear tarea.
2	El sistema muestra el formulario para introducir los datos necesarios para adicionar una nueva tarea.
3	El Líder de proyecto introduce los datos correspondientes y selecciona la opción aceptar.
Poscondiciones:	
1	El sistema verifica los datos.
2	El sistema almacena los datos creados en la base de datos.
Sección: Modificar tarea	

1 Seleccionar la opción modificar tarea.

2 El sistema muestra el formulario con los datos necesarios para modificar una tarea.

3 El Líder de proyecto busca la tarea a modificar.

4 El Líder del proyecto modifica los datos y selecciona la opción aceptar.

Poscondiciones:

1 El sistema actualiza la base datos con los datos modificados.

Sección: Eliminar tarea

1 Seleccionar la opción eliminar tarea.

2 El sistema muestra el formulario con los datos necesarios para eliminar una tarea.

3 El Líder de proyecto busca la tarea a eliminar.

3 El Líder del proyecto elimina el tarea y selecciona la opción aceptar.

Poscondiciones:

1 El sistema actualiza la bade datos.

Pos-condiciones

1 El sistema almacena la información de las operaciones realizadas.

Validaciones		N/A
1		N/A
Relaciones	Requisitos	N/A
	Incluidos	
	Extensiones	N/A
Conceptos	Gestionar tarea	
	Gestionar tarea	Visibles en la interfaz: Nombre de fase, Nombre de la tarea, Peso de la tarea.
Requisitos especiales	N/A	
Asuntos pendientes	N/A	

Tabla 5 Especificación del requisito Gestionar módulo.

Precondiciones	Tiene que haberse autenticado correctamente con los permisos necesarios
Flujo de eventos	
Flujo básico Gestionar módulo.	
1	Seleccionar la opción deseada para gestionar módulo.
2	Concluye el requisito.
Sección: Crear módulo	
1	Seleccionar la opción crear módulo.
2	El sistema muestra el formulario para introducir los datos necesarios para adicionar un nuevo módulo.
3	El Líder de proyecto introduce los datos correspondientes y selecciona la opción aceptar.
Poscondiciones:	
1	El sistema verifica los datos.
2	El sistema almacena los datos creados en la base de datos.
Sección: Modificar módulo	
1	Seleccionar la opción modificar módulo.
2	El sistema muestra el formulario con los datos necesarios para modificar un módulo.
3	El Líder de proyecto busca el módulo a modificar.
4	El Líder del proyecto modifica los datos y selecciona la opción aceptar.
Poscondiciones:	
1	El sistema actualiza la base datos con los datos modificados.
Sección: Eliminar módulo	
1	Seleccionar la opción eliminar módulo.
2	El sistema muestra el formulario con los datos necesarios para eliminar un módulo.
3	El Líder de proyecto busca el módulo a eliminar.
4	El Líder del proyecto elimina el módulo y selecciona la opción aceptar.
Poscondiciones:	

1 El sistema actualiza la base de datos.		
Pos-condiciones		
1 El sistema almacena la información de las operaciones realizadas.		
Validaciones	N/A	
1	N/A	
Relaciones	Requisitos	N/A
	Incluidos	
	Extensiones	N/A
Conceptos	Gestionar módulo	
	Gestionar módulo	Visibles en la interfaz: Nombre del módulo, Nombre de la fase a la que pertenece, Proyecto al que pertenece.
Requisitos especiales	N/A	
Asuntos pendientes	N/A	

Tabla 6 Especificación del requisito Gestionar requisito

Precondiciones	Tiene que haberse autenticado correctamente con los permisos necesarios.
Flujo de eventos	
Flujo básico Gestionar requisito.	
1	Seleccionar la opción deseada para gestionar requisito.
2	Concluye el requisito.
Sección: Crear requisito	
1	Seleccionar la opción crear requisito.
2	El sistema muestra el formulario para introducir los datos necesarios para adicionar un

nuevo requisito.

3 El Líder de proyecto introduce los datos correspondientes y selecciona la opción aceptar.

Poscondiciones:

- 1 El sistema verifica los datos.
- 2 El sistema almacena los datos creados en la base de datos.

Sección: Modificar requisito

- 1 Seleccionar la opción modificar requisito.
- 2 El sistema muestra el formulario con los datos necesarios para modificar un requisito.
- 3 El Líder de proyecto busca el requisito a modificar.
- 4 El Líder del proyecto modifica los datos y selecciona la opción aceptar.

Poscondiciones:

- 1 El sistema actualiza la base datos con los datos modificados.

Sección: Eliminar requisito

- 1 Seleccionar la opción eliminar requisito.
- 2 El sistema muestra el formulario con los datos necesarios para eliminar un requisito.
- 3 El Líder de proyecto busca el requisito a eliminar.
- 4 El Líder del proyecto elimina el requisito y selecciona la opción aceptar.

Poscondiciones:

- 1 El sistema actualiza la bade datos.

Pos-condiciones

- 1 El sistema almacena la información de las operaciones realizadas.

Validaciones	N/A
---------------------	-----

1	N/A
---	-----

Relaciones	Requisitos	N/A
-------------------	-------------------	-----

Incluidos

Extensiones	N/A
--------------------	-----

Conceptos	Gestionar requisito
------------------	--------------------------------

Gestionar requisito	Visibles en la interfaz: Nombre , Responsable, Complejidad, % de avance, Nombre de la fase a la que pertenece, Módulo al que pertenece.
Requisitos especiales	N/A
Asuntos pendientes	N/A

Tabla 7 Especificación del requisito **Mostrar reportes**

Precondiciones	Tiene que haberse autenticado correctamente con los permisos necesarios.
Flujo de eventos	
Flujo básico Mostrar reportes.	
1	Seleccionar la opción deseada para mostrar reporte.
2	Concluye el requisito.
Sección: Mostrar estado de avance del proyecto.	
1	Seleccionar la opción para mostrar estado de avance del proyecto.
2	El sistema muestra el estado de avance del proyecto.
Poscondiciones:	
Sección: Mostrar el por ciento de avance del proyecto.	
1	Seleccionar la opción para mostrar por ciento de avance del proyecto.
2	El sistema muestra el por ciento de avance del proyecto.
Poscondiciones:	
Sección: Mostrar por ciento de avance de los módulos por semanas.	
1	Seleccionar la opción para mostrar por ciento de avance por semanas del proyecto.
2	El sistema muestra el por ciento de avance del proyecto por semanas.
Poscondiciones:	

Sección: Mostrar el estado de avance por módulo del proyecto.

1 Seleccionar la opción para mostrar el estado de avance por módulo del proyecto.

2 El sistema muestra el estado de avance por módulo del proyecto.

Poscondiciones:

Sección: Mostrar el por ciento de avance por módulo.

1 Seleccionar la opción para mostrar por ciento de avance por módulo.

2 El sistema muestra el por ciento de avance por módulo.

Poscondiciones

Pos-condiciones El sistema almacena la información de las operaciones realizadas.

Validaciones		N/A
1		N/A
Relaciones	Requisitos	N/A
	Incluidos	
	Extensiones	N/A
Conceptos	Mostrar reportes	
	Mostrar reportes	Visibles en la interfaz: estado de avance,porciento de avance.
Requisitos especiales		N/A
Asuntos pendientes		N/A

Tabla 8 Especificación del requisito Generar gráficos

Precondiciones	Tiene que haberse autenticado correctamente con los permisos necesarios.
Flujo de eventos	

Flujo básico Generar gráficos.

- 1 Seleccionar la opción de Reportes .
- 2 Seleccionar el estado de avance.
- 3 Seleccionar el por ciento de avance.
- 4 Concluye el requisito.

Sección: Generar gráficos del por ciento de avance del proyecto por semanas.

- 1 Seleccionar el por ciento de avance de avance del proyecto por semanas de los reportes.
- 2 El sistema genera el gráfico del por ciento.

Poscondiciones:

Sección: Generar gráficos del por ciento de avance por módulos del proyecto.

- 1 Seleccionar de los reportes el por ciento de avance por módulo del proyecto.
- 2 El sistema genera el gráfico del por ciento.

Poscondiciones:

Pos-condiciones

- 1 El sistema almacena la información de las operaciones realizadas.

Validaciones		N/A
1		N/A
Relaciones	Requisitos	N/A
	Incluidos	
	Extensiones	N/A
Conceptos	Generar gráfico	
	Generar gráfico	Visibles en la interfaz: porcentaje de avance
Requisitos especiales		N/A
Asuntos pendientes		N/A

2.5. Modelo conceptual

En el modelo conceptual se definen los diferentes requisitos funcionales que se van a desarrollar en la herramienta, los atributos a utilizar en cada uno de ellos y las relaciones que se establecen en los mismos, con lo cual se obtiene un mejor entendimiento de cómo se va comportar el sistema.

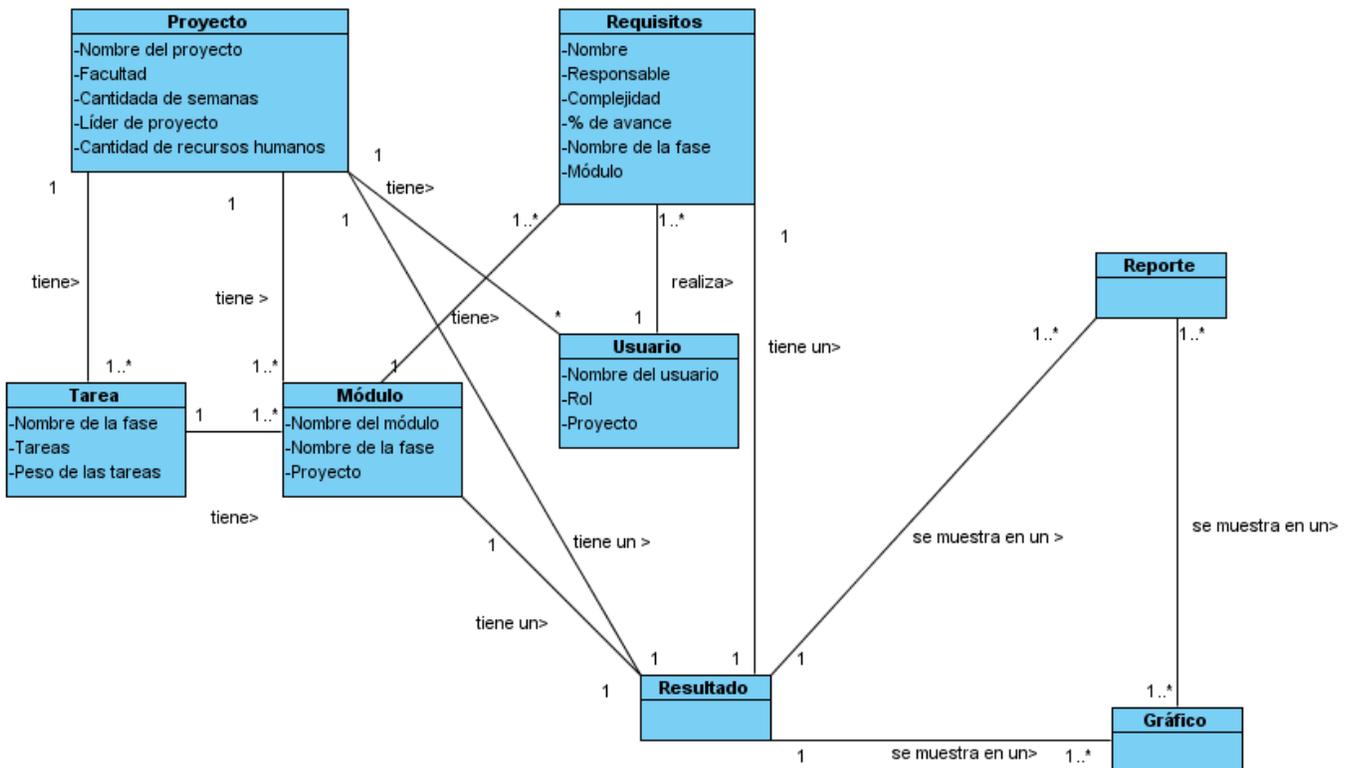


Figura 8 Modelo Conceptual

2.6. Diagramas de clases del diseño

Un diagrama de clases muestra un conjunto de clases, interfaces y colaboraciones con sus relaciones estructurales y de herencia. En el caso de las aplicaciones Web, representa las colaboraciones que ocurren entre las páginas, donde cada página lógica puede ser representada como una clase. En este tipo

de aplicaciones son más importantes la modelación de la lógica y estado del negocio que los detalles de presentación.

2.6.1. Utilización de Patrones de Diseño

De los patrones GRASP estudiados anteriormente, Bajo Acoplamiento y Alta Cohesión, han sido motivo de inspiración a la hora de diseñar la herramienta. También son los patrones que la arquitectura propuesta por el proyecto ERP define para la realización del mismo.

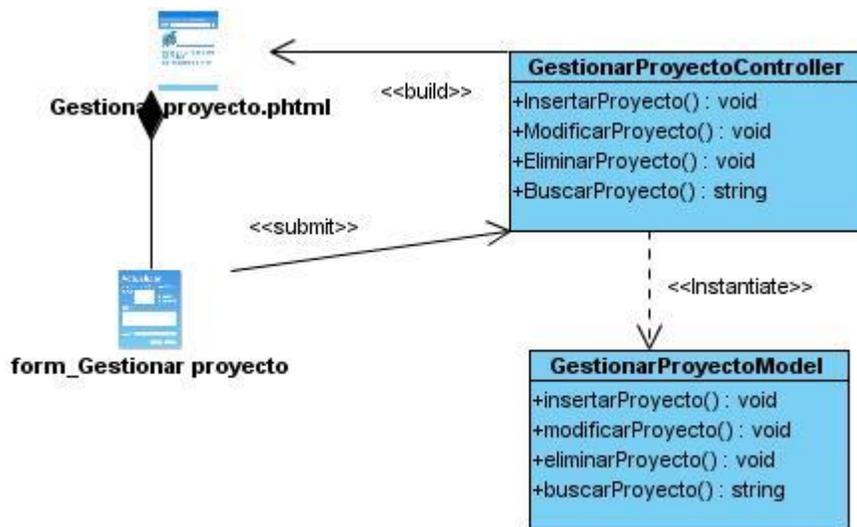


Figura 9 Diagrama de clases del diseño del requisito Gestionar proyecto

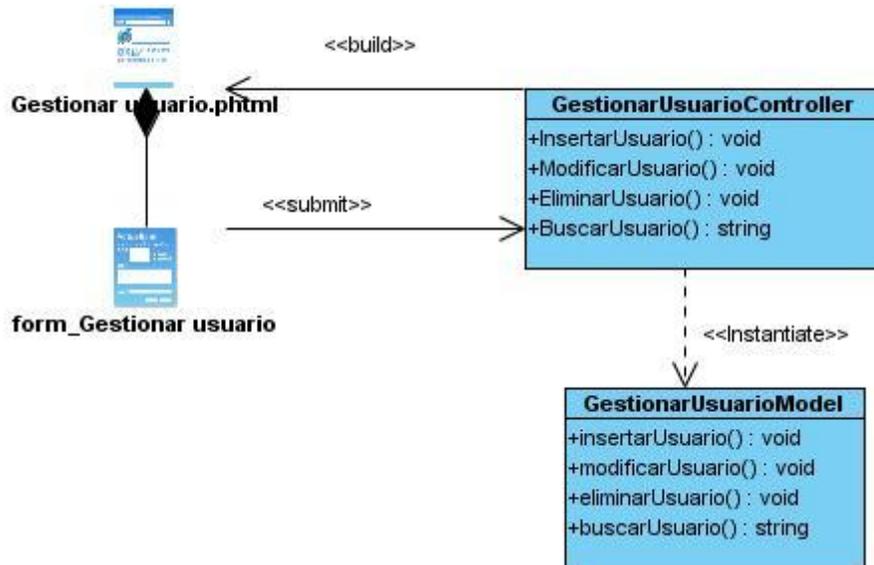


Figura 10 Diagrama de clases del diseño del requisito Gestionar usuario

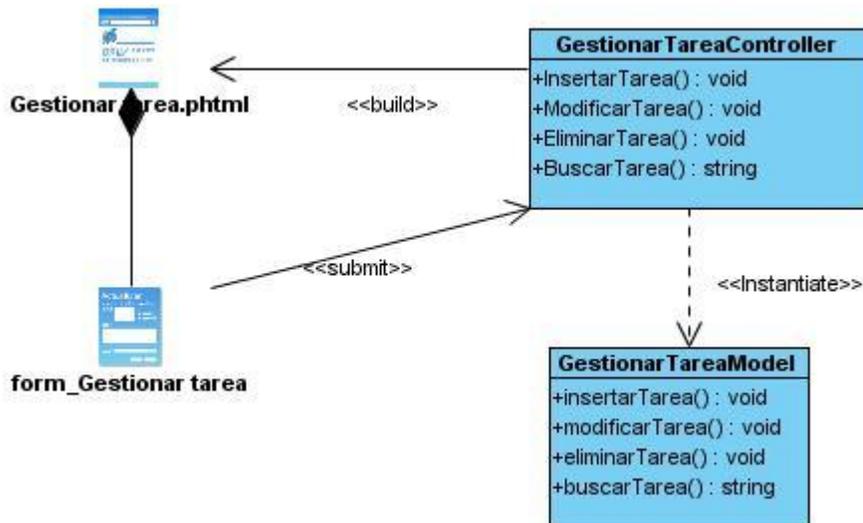


Figura 11 Diagrama de clases del diseño del requisito Gestionar Tarea

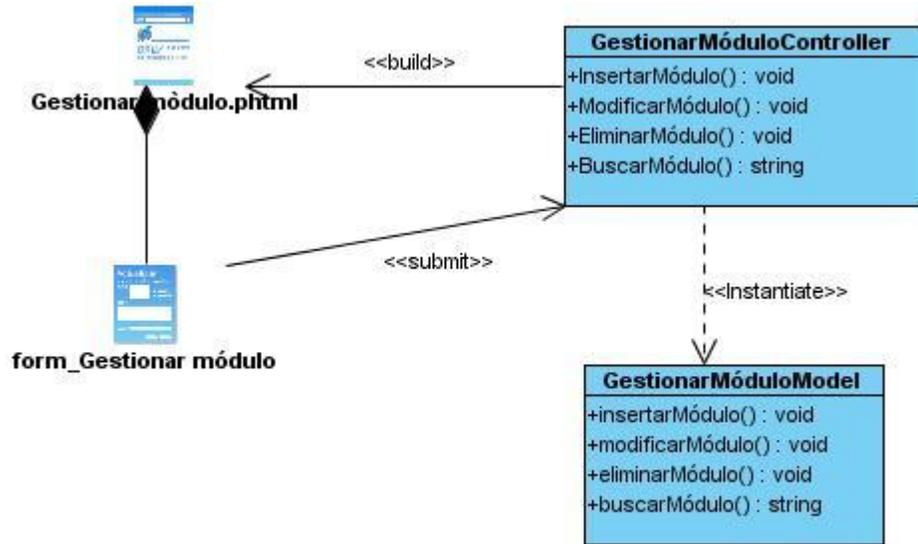


Figura 12 Diagrama de clases del diseño del requisito Gestionar módulo

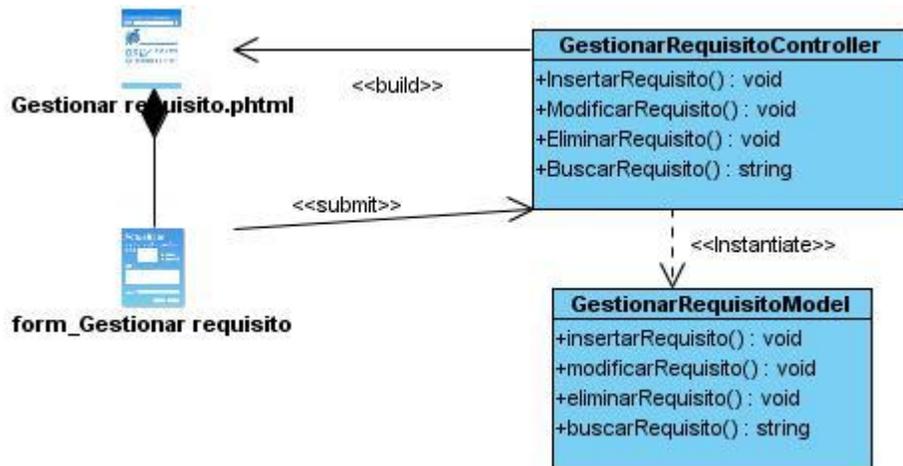


Figura 13 Diagrama de clases del diseño del requisito Gestionar requisitos

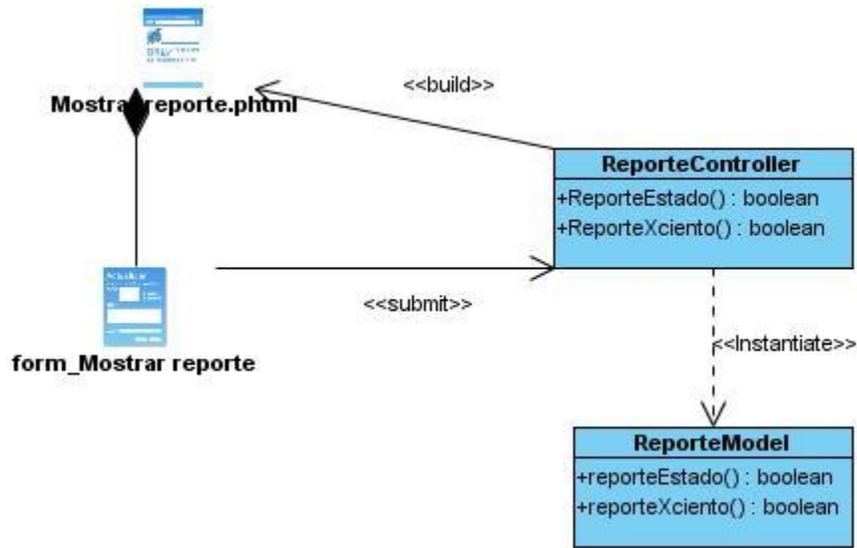


Figura 14 Diagrama de clases del diseño del requisito Mostrar reporte

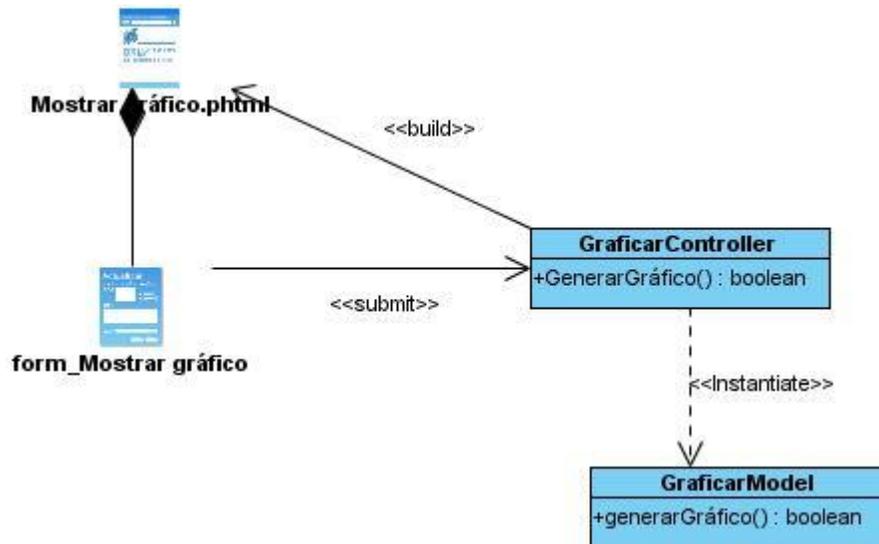


Figura 15 Diagrama de clases del diseño del requisito Graficar

2.7. Modelo Entidad-Relación

El Modelo Entidad-Relación está formado por un conjunto de conceptos que permiten describir la realidad mediante un conjunto de representaciones gráficas.

Originalmente, el modelo entidad-relación sólo incluía los conceptos de entidad, relación y atributo. Más tarde, se añadieron otros conceptos, como los atributos compuestos y las jerarquías de generalización, en lo que se ha denominado modelo entidad-relación extendido.

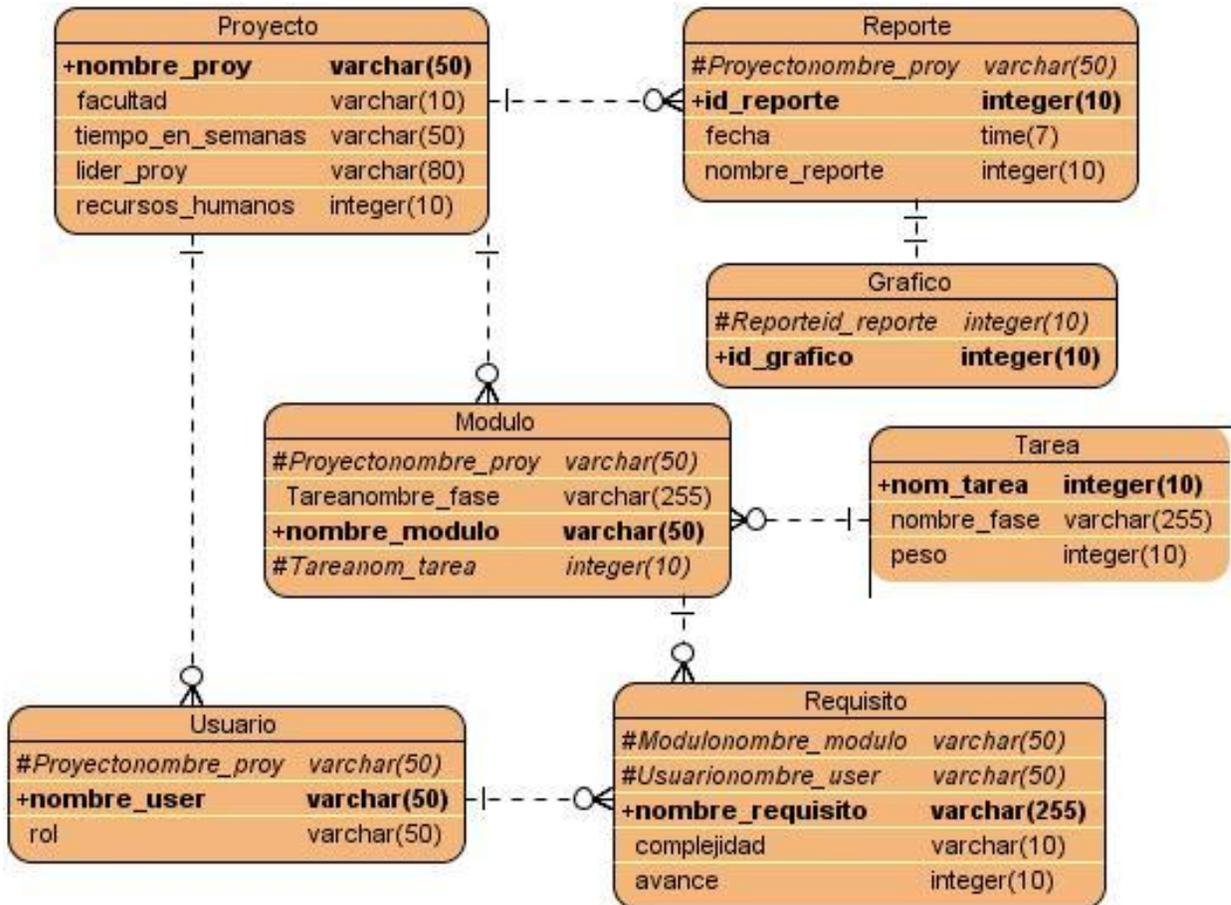


Figura 16 Modelo Entidad-Relación

En este modelo se visualizan las entidades con sus atributos y como éstas se vinculan mediante relaciones. Las entidades son las encargadas de almacenar todos los datos necesarios para que la herramienta ofrezca las funcionalidades definidas en la captura de requisitos.

2.8. Diagrama de Componentes

Los diagramas de componentes describen los elementos físicos del sistema y sus relaciones. Muestran las opciones de realización incluyendo código fuente, binario y ejecutable. Los componentes representan todos los tipos de elementos de software que entran en la fabricación de aplicaciones informáticas. Pueden ser simples archivos, paquetes, bibliotecas cargadas dinámicamente, etc. Los distintos componentes pueden agruparse en paquetes según un criterio lógico y con vistas a simplificar la implementación.

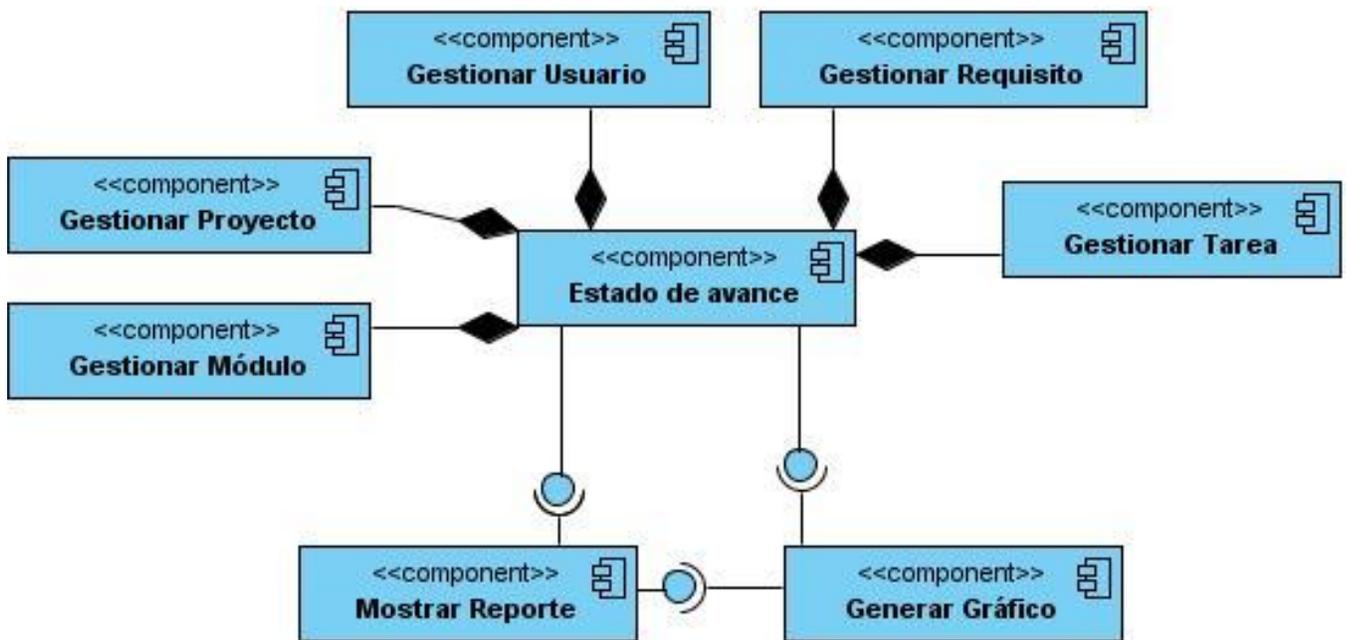


Figura 17 Diagrama de Componentes

El diagrama muestra los componentes a desarrollar en la herramienta y las relaciones entre ellos. El componente Estado de avance está compuesto por los componentes Gestionar Proyecto, Gestionar Módulo, Gestionar Tarea, Gestionar Usuario, Gestionar Requisito y los componentes Mostrar Reportes y Generar Gráficos tienen una relación de consumo de interfaz con Estado de Avance.

2.9. Conclusiones del capítulo

En este capítulo, se expuso la métrica a utilizar profundizando en las principales fórmulas que esta utiliza para conocer el estado de avance. A partir de la comprensión de los procesos de negocio, se definieron las principales funcionalidades que debe tener la propuesta del sistema a desarrollar, estructurándose en requisitos funcionales y no funcionales. Se elaboró el modelo conceptual, el diagrama de clases del diseño, el modelo entidad relación, el diagrama de componentes y se describieron textualmente cada uno de ellos. Con todo lo antes expuesto se tiene una noción de que es lo que se debe hacer y cómo, en la realización de la herramienta propuesta.

Capítulo #3. Validación de la solución

3.1. Introducción

En el presente capítulo se realiza la validación de requisitos con el objetivo de comprobar que estos sean precisos y cumplen con las necesidades del usuario, además se describen las métricas utilizadas para evaluar la calidad del diseño así como las afectaciones de los atributos de calidad con la aplicación de las mismas y se presentan los prototipos de interfaz de usuario.

3.2. Validación de requisitos

Se decidió utilizar la lista de chequeo definida en el proyecto ERP-Cuba para la validación de requisitos, la cual verifica el estado de los artefactos generados en el proceso de desarrollo del software.

Formas de uso:

Peso: Define si el indicador a evaluar es crítico o no (no Crítico se considera aceptable).

Evaluación (Eval): Es la forma de evaluar el indicador en cuestión. El mismo se evalúa de 1 en caso de mal y 0 en caso que el elemento revisado no presente errores.

Cantidad de elementos afectados: Especifica la cantidad de errores encontrados sobre el mismo indicador.

Comentario: Especifica los señalamientos o sugerencias que quiera incluir la persona que aplica la lista de chequeo.

Estructura del Documento: Abarca todos los aspectos definidos por el expediente de proyecto o el formato establecido por el proyecto.

N.P. (No Procede): Se usa para especificar que el indicador a evaluar no se puede aplicar en ese caso.

Tabla 9 Lista de chequeo

Elementos definidos por la metodología

Peso	Indicadores a Evaluar	Eval	(NP)	Cantidad de elementos afectados	Comentarios
No critico	1.¿Están todos los requisitos redactados de forma simple y clara para aquellos que vayan a consultarlo en un futuro?	0		0	
No critico	2. ¿Debería especificarse algún requisito con más detalle?	0		0	
No critico	3. ¿Debería especificarse algún requisito con menos detalles?	0		0	
No critico	4.¿Todos los requisitos identificados se centran en lo que el sistema debe hacer y no como el sistema debe hacerlo?	0		0	
	5. ¿Se han identificado la precondición que declara qué debe estar seguro el sistema antes de autorizar el inicio del caso de uso, está escrita como una afirmación simple en pasado continuo?		NP		
No critico	6. ¿Se ha descrito el flujo básico?	0		0	
No critico	7.¿Se han identificado las Pos-condiciones?	0		0	
Critico	8. ¿Se ha descrito los flujos alternativos y sus respectivas Pos-condiciones?	1		7	
No critico	9. ¿Se han validado los datos según lo establecido en el	0		0	

	Modelo conceptual?			
	10. ¿Se han identificado el listado de los requisitos incluidos por el requisito en el siguiente formato <<Evento en el que se incluye>>: <<Nombre del requisito>>, en la agrupación <<Nombre de la agrupación de requisitos>>.?		NP	
	11. ¿Se han identificado el listado de los requisitos que extienden al requisito en el siguiente formato <<Evento en el que se extiende>>: <<Nombre del requisito>>, en la agrupación <<Nombre de la agrupación de requisitos>>?		NP	
	12. ¿Se han identificado los atributos del concepto que se utilizan en el requisito, tanto internamente como para mostrarlos al usuario , en dos grupo : Visibles en la interfaz: <<Nombre del atributo 1>> Utilizados internamente: <<Nombre del atributo 2>> ?		NP	Se muestran solo los Visibles en la interfaz.
	13. ¿Se ha identificado el listado de requisitos especiales, que son los requisitos no funcionales		NP	

	específicos para el requisito?				
	14. ¿ Se ha identificado los asuntos pendientes , que son las posibles mejoras al requisito?		NP		
	15. ¿Se han especificado los formatos de entrada y salida?		NP		
No critico	16. ¿ No aparece un mismo requisito en más de un lugar del documento de especificación?	0		0	
No critico	17. ¿No existe contradicción entre lo especificado por un requisito y lo especificado por otro?	0		0	
No critico	18. ¿No existe contradicción entre lo especificado por un requisito y lo especificado por otro?	0		0	
	19. ¿Se puede verificar cada requisito? (Un requisito se dice que es verificable si existe algún proceso no excesivamente costoso por el cual una persona o una máquina pueda chequear que el software satisface dicho requerimiento?)		NP		

Con la realización de la validación usando la lista de chequeo a los requerimientos se obtuvo un buen resultado en los aspectos medidos, ya que la evaluación tuvo una gran aceptación al igual que el peso de los mismos, y la cantidad de elementos afectados tuvo un mínimo porcentaje de efectividad, por lo tanto la validación de los requisitos fue positiva y satisfactoria ya que los mismos están descritos correctamente y cumplen con el resultado esperado.

3.3. Métricas para evaluar el diseño

Son varios los puntos de vista relacionados con la calidad del software. Desde metodologías hasta las distintas normas de calidad, que pueden estar orientados tanto a los procesos de desarrollo como a los productos de software. No es objetivo abundar sobre los temas de calidad, pero si desarrollar una evaluación del diseño propuesto para el desarrollo de la herramienta.

Un aspecto importante a tener en cuenta en la evaluación del diseño es la creación de métricas básicas inspiradas en el estudio de la calidad del diseño orientado a objetos referenciadas por Pressman en (PRESSMAN 1998).

Atributos de calidad que se abarcan:

1. **Complejidad de implementación:** Consiste en el grado de dificultad que tiene implementar un diseño de clases determinado.
2. **Reutilización:** Consiste en el grado de reutilización presente en una clase o estructura de clase, dentro de un diseño de software.
3. **Acoplamiento:** Consiste en el grado de dependencia o interconexión de una clase o estructura de clase, con otras, está muy ligada a la característica de Reutilización.
4. **Complejidad del mantenimiento:** Consiste en el grado de esfuerzo necesario a realizar para desarrollar un arreglo, una mejora o una rectificación de algún error de un diseño de software. Puede influir indirecta, pero fuertemente en los costes y la planificación del proyecto.

5. **Cantidad de pruebas:** Consiste en el número o el grado de esfuerzo para realizar las pruebas de calidad (Unidad) del producto (Componente, modulo, clase, conjunto de clases, etc.) diseñado.

6. **Responsabilidad:** Consiste en la responsabilidad asignada a una clase en un marco de modelado de un dominio o concepto, de la problemática propuesta.

Las métricas concebidas como instrumento para evaluar la calidad del diseño y su relación con los atributos de calidad definidos en este trabajo son las siguientes:

1. **Tamaño operacional de clase (TOC):** Está dado por el número de métodos asignados a una clase. Los atributos de calidad que afecta son: Responsabilidad, Complejidad de implementación y Reutilización de manera que un aumento del TOC implica que aumente la Responsabilidad y la Complejidad de mantenimiento mientras que la Reutilización disminuye.

Luego de realizar un estudio de las clases utilizadas la evaluación de la métrica reflejó que el 71% de las clases tienen entre 1 y 5 procedimientos lo cual conduce a los resultados mostrados a continuación.

Tabla 10 Resultados de la métrica TOC

Atributos	TOC
Responsabilidad	71% (Baja)
Complejidad de implementación	71% (Baja)
Reutilización	71% (Alta)

Ver gráficos en Anexo 1 de la versión digital de este documento.

Haciendo un análisis de los resultados obtenidos en la evaluación del instrumento de medición de la métrica TOC, se puede concluir que el diseño de la herramienta a implementar tiene una calidad aceptable teniendo en cuenta que el 71 % de las clases posee menos cantidad de operaciones que la mitad del valor máximo registrado en las mediciones. Además el 100% de las clases poseen evaluaciones positivas en los atributos de calidad (Cohesión, Complejidad de Implementación y Reutilización).

2. **Relaciones entre clases (RC):** Está dado por el número de relaciones de uso de una clase con otras. Los atributos de calidad que esta afecta son: Acoplamiento, Complejidad de mantenimiento, Cantidad de

pruebas y Reutilización de manera tal que un aumento del RC implica que aumente el Acoplamiento, Complejidad de mantenimiento y la Cantidad de pruebas mientras que la Reutilización disminuye.

Luego de realizar un estudio de las clases utilizadas la evaluación de la métrica reflejó que un 93 % de las clases tienen menos de 3 dependencias lo cual conduce a los resultados mostrados a continuación.

Tabla 11 Resultados de la métrica RC

Atributos	RC
Acoplamiento	62%(Bajo)
Complejidad de mantenimiento	70%(Baja)
Cantidad de pruebas	70% (Baja)
Reutilización	78%(Alta)

Ver gráficos en Anexo 2 de la versión digital de este documento.

Haciendo un análisis de los resultados obtenidos en la evaluación del instrumento de medición de la métrica RC, se puede concluir que el diseño de la herramienta a implementar tiene una calidad aceptable teniendo en cuenta que el 93 % de las clases incluidas en este subsistema posee menos de 3 dependencias de otras clases. Así mismo los atributos de calidad Acoplamiento, Complejidad de Mantenimiento, Cantidad de Pruebas y Reutilización se comportan satisfactoriamente en un 100 % de las clases.

Analizando los resultados obtenidos después de aplicar las métricas de evaluación del diseño, se puede concluir que los patrones Bajo Acoplamiento y Alta Cohesión fueron utilizados correctamente ya que el 22% de las clases no poseen acoplamiento con otras clases y el 100% posee índices aceptables en cuanto a Acoplamiento y a Cohesión.

3.4. Prototipos de Interfaz de usuario

Para el diseño de las interfaces se tuvo en cuenta la organización de los elementos a mostrar, de forma tal que en todas las interfaces que tienen elementos comunes, estos ocupen la misma posición en cada una de ellas. Se conservaron los mismos patrones de colores, tamaño y forma.

Ver prototipos de interfaz en Anexo 3 de la versión digital de este documento.

3.5. Resultados a nivel UCI que ha tenido la utilización de la Métrica de avance

La Métrica de avance fue utilizada por diferentes proyectos productivos en la UCI entre los que se encuentran el Proyecto Sistema de Gestión Penitenciaria (SIGEP), CICPC y CedruX en la línea de Auditoría. Hasta el momento solo fue posible emplearla utilizando el Excel como herramienta para realizar los cálculos y llegar a obtener algunos gráficos de comportamiento del estado de los proyectos antes mencionados.

Luego de su aplicación además de mostrar los líderes gran aceptación de la misma permitió obtener cuantitativamente el estado de avance de la implementación del software y constituyó en sí un registro histórico de la productividad del trabajo del equipo de proyecto, se pudo determinar hasta qué cantidad de requisitos un equipo de desarrollo puede implementar en un tiempo determinado resultado para el equipo de desarrollo de la línea Auditoría del CEDRUX que en 11 semanas de trabajo se pudo implementar de 130 a 140 funcionalidades y aseguró que la métrica puede ser utilizada en cualquier proyecto de desarrollo de software, redefiniendo las tareas y su peso específico dentro del tiempo total de desarrollo.

3.6. Conclusiones del capítulo

En este capítulo se realizó la validación de requisitos obteniendo un resultado satisfactorio de la descripción de los mismos. Se realizó la evaluación del diseño utilizando las métricas Tamaño operacional de clases (TOC) y Relaciones entre clases (RC) obteniendo como resultado una calidad aceptable del diseño realizado. Además se elaboraron los prototipos de interfaz de usuario para cada requisito con los cuales se pretende dar una idea de la estructura de la interfaz del sistema.

Conclusiones

Se realizó el análisis y diseño de la herramienta que aplica la Métrica de avance. Se creó un punto de partida para desarrollar una herramienta de fabricación nacional que servirá para la planificación y seguimiento de proyectos productivos dentro del modelo universidad de la UCI. Se realizó una validación del diseño mediante la utilización de instrumentos de medición obteniéndose buenos resultados de calidad. Se sentaron las bases para la implementación de la herramienta.

Recomendaciones

Se recomienda consultar la documentación obtenida ya que esta constituye una guía para futuros desarrollos, la base para futuras mejoras y extensiones de la herramienta propuesta, además realizar la implementación de la herramienta propuesta y continuar agregando funcionalidades que gestionen otros procesos comunes a cada módulo dentro del proyecto.

Referencias Bibliográficas

1. [1] Métricas de Software. Disponible en: [En línea] <http://www.monografias.com>.
2. [2] Propuesta de técnicas de estimación y métricas para la metodología ágil. Disponible en: [En línea] http://bibliodoc.uci.cu/TD/TD_2442_09.pdf.
3. [3] SIMETSE-Sistema de METricas para evaluar el Software Educativo. Disponible en: [En línea] 2007. http://bibliodoc.uci.cu/TD/TD_0803_07.pdf.
4. [4] Métricas, estimación y planificación de proyectos de software. Disponible en: Métricas, estimación y planificación de proyectos de software. Disponible en: [En línea] http://74.125.47.132/search?q=cache:eH3auKiR5_YJ:www.willydev.net/Insit.
5. [5] Seguimiento y Evaluación de proyectos. [En línea] www.civicus.org.
6. [6] Visual Paradigm International. [En línea] Disponible en: [En línea] 2005. <http://www.visual-paradigm.com/product/vpuml/>.
7. [7] Metodologías del desarrollo del software. Disponible en: [En línea] 2004. <http://www.informatizate.net>.
8. [8] Rational Rose Enterprise, Disponible en: <http://www-142.ibm.com/software/products/es/es/enterprise/>.
9. [9] Patrones. Disponible en: <http://www.librospdf.org/72024/craig-larman-uml-y-patrones/>.
10. [10] Mapa de procesos. Disponible en: <http://www.portalcalidad.com/articulos/73-Doc-mapa>.
11. [11] Seguimiento de proyectos. [Online] <http://www.monografias.com/trabajos10/soft/soft.shtml#ges>.
12. [12] DotProject. [Online] <http://es.wikipedia.org/wiki/DotProject>.
13. [13] DotProject. [Online] <http://www.abartiateam.com/dotproject>.
14. [14] OpenProject. [Online] <http://sentidoweb.com/2007/09/27/openproj-gestion-de-proyectos-open-source.php>

15.[15]OpenProject. [Online] <http://www.aplicacionesempresariales.com/openproj-herramienta-para-el-diseno-libre-de-proyectos.html>

16.[16]GanttProject. [Online] <http://www.aplicacionesempresariales.com/ganttproject-herramienta-para-el-diseno-libre-de-proyectos>.

Bibliografía

Ing. Rosario Rodríguez Torres, Ing. Dailys Díaz Fuentes. Métrica para establecer el estado de avance de proyectos en soluciones de gestión Sistema Integral de Gestión CedruX.

Jacobson, Ivar, Booch, Grady and Rumbaugh, James. El Proceso Unificado de Desarrollo de Software. Madrid : PEARSON EDUCACION, 2000.

Monroe, Robert, Melton, Ralph and Garlan, David. Architectural Styles, design patterns, and objects. s.l. : IEEE Software, 1997.

Molpeceres, Alberto. Procesos de desarrollo: RUP, XP y FDD. 2002

RUP, Rational Unified Process. 2003. P.

Rumbaugh, James. The Unified Modeling Language Reference Manual. s.l. : Addison Wesley, 1999.