Universidad de las Ciencias Informáticas
Facultad 10

Estrategia de Implantación del Modelo de Factoría de Software aplicando Inteligencia a la Fábrica de Portales de la Facultad 10.

Tesis en opción al título de ingeniero en ciencias informáticas

Autores: Ailet Avila Pérez.

Yadira Caridad Bagarotti Acebo.

Tutores: Msc. Yaimí Trujillo Casañola

Ing. William Santana Méndez

Ciudad de la Habana, Cuba
Mayo, 2009

DECLARACIÓN DE AUTORÍA

Declaramos que somos las únicas autoras del presente trabajo. Autorizamos a la Universidad de las Ciencias Informáticas (UCI) a darle el uso que estimen pertinente a este trabajo.
Para que conste firman el presente documento a los _____ días del mes de _____ de _____.

_____ _____

Ailet Ávila Pérez

Autora

Yadira Caridad Bagarotti Acebo

Autora

_____

MSc. Yaimí Trujillo Casañola

Tutora

_____

Ing. William Santana Méndez

Tutor

"La calidad nunca es un accidente; siempre es el resultado de un esfuerzo de la inteligencia."

John Ruskin

Agradecimientos

A nuestros padres por brindarnos su amor y ser nuestros amigos incondicionales.

A los profesores y amigos que nos acompañaron en este largo viaje que ha sido la UCI, fueron el faro que nos guió cuando alguna vez perdimos el rumbo y alegraron nuestras vidas en los momentos más difíciles.

A Yaimí y William por apoyarnos y darnos su confianza en todo momento.

A todas las personas que de una forma u otra brindaron su apoyo para la realización de esta investigación, les estaremos eternamente agradecidas.

A mi tata por apoyarme en toda mi vida.
A mis tres mosqueteros (mulato, tigre y salomón) por nunca abandonarme y ser mis consejeros.
A mis amigos del complejo 2 de comedores (el negro, el gordo, julito , maikel y angel) por alegrar mi existencia.
A mis amigas luisa, cristi, odalis, mariajulia, lisset y sisley por ser incondicionales.
A mi amor por ayudarme en este período de mi vida y ser mi guía.
En fin a todas las personas que me apoyaron para que yo pudiera realizar mi sueño.

Gracias

Ailet

A mi mamá por no juzgarme nunca, por escucharme cada vez que necesité hablar y por siempre estar ahí para mí. A mi familia por estar siempre cuando los necesité y por confiar en mí, incluso en los momentos más difíciles.
Gracias a mis amigos por hacerme más fácil estos 5 largos años.
A Karel, Yosbel, Albertini, Lucyliu, Aile, Alémany y Yaire por soportarme tanto tiempo, sepan que los quiero mucho.
A todos aquellos que estuvieron, los que están y seguirán a mi lado.
Gracias de todo corazón.

Yady

Dedicatoria

A mis creadores por darme la vida y estar ahí siempre que los necesitaba, por apoyarme y alentarme para poder llegar a realizar el sueño de mi vida.

A mis padres

Ailet

A mis padres que me lo han dado todo sin pedir nada a cambio, a Iche por ser tan especial y a mi abuelito Roberto, siempre lo recordaré.

Yady

RESUMEN

La Universidad de las Ciencias Informáticas (UCI) promueve el desarrollo de software en el ámbito nacional e internacional, mediante proyectos productivos que vinculan a estudiantes y profesores, en este desarrollo la Fábrica de Portales de la Facultad 10 juega un papel primordial, pues tiene la misión de producir portales tanto para el país como para el extranjero.

Este proyecto presenta problemas con la definición del proceso de desarrollo de software, pues las personas se encuentran desorientadas al no conocer sus responsabilidades, ni las actividades que deben realizar, no se utilizan estándares de calidad para mejorar el mismo y tampoco se lleva a cabo la reutilización de componentes. Estas son algunas de las deficiencias encontradas en el modelo de producción que afectan el proceso de desarrollo de portales, las cuales deben ser eliminadas para obtener productos con mayor calidad.

De ahí que el presente trabajo defina como problema científico: ¿Cómo las deficiencias en el modelo de producción de la Fábrica de Portales afectan la producción? Con el objetivo: de diseñar una estrategia de transferencia de un Modelo de Factoría de Software que elimine las deficiencias existentes en este proyecto. Para cumplir con el objetivo propuesto se utilizó la metodología de la investigación científica, asumiendo como hipótesis: Si se estudia el proceso de producción de software existente en el proyecto y se elabora una estrategia de implantación de un modelo de factoría de software se obtendrá una definición del proceso productivo de software que contribuya a eliminar las deficiencias existentes.

Para aplicar la estrategia se definieron 5 paquetes que coinciden con las entidades que conforman el modelo de factoría propuesto. Para su implantación se tomó un proyecto piloto, debido al poco tiempo disponible para efectuar la transferencia no se transfirieron todos los paquetes, pero sirvió para eliminar insuficiencias y mejorar algunos de los procesos realizados en el proyecto. Los resultados más trascendentales obtenidos se reflejan en la organización del proceso y de los roles de acuerdo a la metodología RUP, y en la reorganización del repositorio de componentes.

PALABRAS CLAVE

Modelo de Factoría de Software, Estrategia de transferencia

Contenido

ÍNDICE DE TABLAS

ÍNDICE DE FIGURAS

INTRODUCCIÓN

El desarrollo de software es uno de los sectores de mayor crecimiento en los últimos años y representa, cada vez más, una de las principales actividades económicas tanto en los países desarrollados como para los países en vía de desarrollo. Dentro de este avance se destacan los portales Web, hallándose presente en la actualidad en la mayor parte de los sistemas que resultan vitales para el funcionamiento y el progreso de las sociedades modernas. Estos son muy útiles para organizaciones, empresas, escuelas y otras entidades por brindar acceso a una disímil cantidad de recursos y servicios informáticos de forma integrada y sencilla. Además por resolver necesidades en cuanto al acceso a la información y permitir al usuario una fácil comunicación con otras personas.

Cuba no se encuentra ajena a este desarrollo, pues actualmente en la Universidad de las Ciencias Informáticas (UCI), se desarrollan portales para el ámbito nacional e internacional.

Dentro de esta se halla la Facultad 10, la cual se especializa en Software Libre y cuenta con 4 polos que se dedican a crear software en distintas vertientes, uno de estos es el polo de Gestión de la Información y el Conocimiento y dentro de él se localiza el proyecto Fábrica de Portales, este nace con el objetivo de desarrollar aplicaciones Web a nivel nacional o foráneo, contribuyendo a la informatización de la sociedad. Los proyectos que se realizan en la fábrica son generalmente de corta duración y presentan una arquitectura de la información similar, pero el proceso que utilizan para la construcción no es el más adecuado teniendo en cuenta las características de los mismos, vinculados a este proceso se hallan estudiantes y profesores que cumplen diferentes roles.

En este proyecto se han detectado varias deficiencias en el modelo de producción que dificultan la creación de software, las mismas se exponen a continuación: la mayoría de los integrantes desconocen el procedimiento para desarrollar un portal y no tienen clara la estructura del proyecto. Aunque la mayoría conocen su rol muchos no saben las actividades que deben desarrollar y los artefactos que generan. Existe desconocimiento de la metodología empleada por el proyecto y de los flujos de trabajo establecidos. Si bien la comunicación del equipo de desarrollo en su totalidad es buena, la planificación del trabajo tanto personal como a nivel de equipo no es la mejor, no se siguen estándares de calidad, afectándose la efectividad del equipo de desarrollo. Los componentes realizados no se documentan para facilitar su búsqueda y utilización en productos posteriores y en la gran mayoría de los casos no se lleva a cabo la reutilización de los mismos. El proyecto no cuenta con el material necesario para ofrecerles la

capacitación a los estudiantes, esto trae consigo el insuficiente dominio de las herramientas de trabajo y a su vez la gestión de las dudas se convierte en un tema crítico e imprescindible. No se realiza la firma de un acta como constancia del levantamiento de requisitos y se desconocen las políticas de seguridad de la información establecidas.

Estas deficiencias afectan el proceso de desarrollo de los productos, una solución a estos problemas es el enfoque de factoría de software que tiene como propósitos establecer un proceso estandarizado, repetible y mejorable continuamente, donde las actividades de desarrollo sean predecibles y se reduzca la cantidad de trabajo debido a la reutilización de componentes; mantener capacitados a los recursos humanos y definir roles y responsabilidades, concentrando sus esfuerzos en un área determinada de la producción. Se debe contar con un grupo de herramientas estandarizadas para la construcción de software y la administración y control del proyecto, además debe existir una fuerte comunicación con el cliente.

Para lograr esto se necesita definir una estrategia que transfiera este nuevo enfoque a la práctica, garantizando el cumplimiento de los objetivos y que se obtengan los resultados esperados.

Por lo anteriormente expuesto se plantea el siguiente **problema científico**: *¿Cómo las deficiencias en el modelo de producción de la Fábrica de Portales afectan la producción?*

Se tiene como **objeto de estudio** *el proceso de desarrollo de software* y un **campo de acción** que abarca *el proceso de desarrollo de portales.*

Para darle solución al problema se definió como **objetivo general**: *Diseñar y ejecutar una estrategia de implantación de un modelo de factoría de software en la Fábrica de Portales de la Facultad 10.* Para lograr un desempeño óptimo de la investigación se determinaron los siguientes **objetivos específicos:**

1. Caracterizar y definir el enfoque de factoría de software y los modelos que se han aplicado en el mundo.
2. Caracterizar y definir el proceso productivo en la Fábrica de Portales.
3. Diseñar y ejecutar la estrategia de implantación del modelo de factoría de software.
4. Diseñar y validar en un piloto el modelo de factoría propuesto.

La **hipótesis** que sustenta la investigación es la que sigue: *Si se estudia el proceso de producción de software existente en el proyecto y se elabora una estrategia de implantación de un modelo de factoría de software se obtendrá una definición del proceso productivo de software que contribuya a eliminar las deficiencias existentes.*

Para demostrar la veracidad de la hipótesis planteada y darle cumplimiento a los objetivos propuestos se proponen las siguientes **tareas de investigación.**

- Realizar búsquedas bibliográficas sobre conceptos de portales, así como sus funciones y ventajas.
- Realizar búsquedas bibliográficas sobre conceptos de factoría de software.
- Comparar varias definiciones de factoría de software.
- Estudio de los modelos de factoría de software propuesto por autores y organizaciones especializadas.
- Comparar los modelos de factoría de software propuesto por autores y organizaciones especializadas.
- Describir el modelo a aplicar en el proyecto.
- Realizar búsquedas bibliográficas sobre conceptos de ingeniería de software, procesos y modelos de procesos del software.
- Comparar varios modelos de procesos de desarrollo del software.
- Realizar un estudio de las metodologías de desarrollo de software más utilizadas en la universidad.
- Realizar un estudio de las herramientas que se utilizan para la construcción de portales.
- Realizar búsquedas bibliográficas sobre la aplicación del enfoque en empresas, organizaciones e institutos que desarrollan software en el mundo.
- Elaborar la estrategia de implantación del modelo.
- Describir los procesos de implantación.
- Definir pruebas pilotos.
- Aplicar el proceso a un piloto.
- Evaluar los resultados de aplicar el piloto.

La mejora que se pretende lograr con esta investigación es obtener un modelo de producción de portales que se adecúe a las características del mismo, teniendo en cuenta la organización del proceso y las personas, la gestión de proyecto, la definición de las bases tecnológicas, la reutilización y una óptima comunicación con el cliente.

La investigación realizada transitó por varias etapas:

En la primera etapa se realizó un estudio de los portales, sus funcionalidades y ventajas, se tuvo en cuenta el enfoque de factoría de software y los modelos de factoría propuestos por diferentes autores y organizaciones especializadas y se seleccionó el que se va a implantar. Además se hizo alusión a los elementos necesarios para conformar un proceso para el desarrollo de portales como son: los modelos de procesos del software, las metodologías y herramientas.

En la segunda etapa se realizaron encuestas y entrevistas a los integrantes de la fábrica y a los clientes, con el objetivo de identificar los problemas existentes en el proyecto; los temas a tratar fueron: organización del proceso y las personas, la gestión del proyecto, las bases tecnológicas y la comunicación con el cliente. Además se diseñó le estrategia de implantación del modelo de factoría elegido anteriormente.

En la tercera etapa se analizaron los resultados de las encuestas y entrevistas, se ejecutó el proceso de implantación en un proyecto, demostrándose que contribuye a la industrialización, pues define el flujo de procesos, los roles y las responsabilidades en el proyecto, mejora la planificación e incorpora la reutilización de componentes.

El trabajo fue dividido en tres capítulos, a continuación se da una breve explicación de cada uno.

Capítulo 1: Fundamentación Teórica. Aborda conceptos relacionados con el tema de portales, factoría de software, ingeniería de software y procesos. Se mencionan los modelos de factorías de software más representativos a nivel internacional, las características esenciales de cada uno y la experiencia vista desde las empresas en donde se aplicó este enfoque. Además, se hace referencia a los modelos de procesos del software, las metodologías, herramientas y lenguajes de programación afines con las necesidades de un portal.

Capítulo 2: Estrategia de Implantación. Describe los métodos, procedimientos y técnicas utilizadas para llevar a cabo la investigación científica y propone la estrategia de implantación del modelo seleccionado.

Capítulo 3: Resultados de la implantación. Muestra los resultados obtenidos de las encuestas y entrevistas aplicadas, así como los resultados de la implantación del modelo.

CAPÍTULO 1: FUNDAMENTACIÓN TEÓRICA.

1.1 Introducción

En el presente capítulo se enuncian los principales conceptos relacionados con portales, factoría de software, ingeniería de software y proceso. Se describen los modelos de factoría de software propuestos por diferentes autores y organizaciones especializadas, y los resultados de varias empresas que han adoptado este enfoque. Además se tienen en cuenta los elementos necesarios para conformar un proceso como son: las metodologías, los modelos de procesos del software y las herramientas.

1.2 ¿Qué es un portal ?

En la actualidad los portales han alcanzado una gran demanda debido a todas las facilidades y beneficios que ofrecen. Cada día son más las empresas que se dedican a la producción de los mismos y son utilizados con distintos fines.

En el año 2000 Mendoza plantea: “Portal es un nuevo término, comúnmente sinónimo de puerto, que se utiliza para referirnos a un Sitio Web que sirve de punto de partida para iniciar nuestra actividad de navegación en Internet, al cual visitamos con frecuencia y al que generalmente designamos como página de inicio en nuestro navegador, trátese de Microsoft Internet Explorer o Netscape Communicator”. (MENDOZA, 2000)

Mendoza expresa que un portal es un sitio que se utiliza como medio para navegar en internet y que es visitado con mucha frecuencia y puede ser accedido de cualquier navegador.

En el 2004 Guglielmetti dice: “Es un sitio que brinda acceso a una diversa cantidad de recursos y servicios informáticos de forma integrada y sencilla. En este se suelen encontrar herramientas para compra electrónica, programas, documentos de toda clase, foros de usuarios, y buscadores, entre otros servicios”. (GUGLIELMETTI, 2004)

Guglielmetti tiene un concepto muy diferente de Mendoza, pues para él un portal no es solo un contenedor de información sino que también puede ofrecer servicios de forma completa y sencilla.

Abreu y Colomé expresan en el 2007: “Se puede decir que un sitio Web (website) es un conjunto de páginas Web, típicamente comunes a un dominio de Internet o subdominio en la Word Wide Web en Internet”. (ABREU, COLOME, 2007)

Para Abreu y Colomé un sitio es un conjunto de páginas que se encuentran inmersa en la Word Wide Web en Internet.

En Skynetstudio se puntualiza en el 2008:“Es un sitio web cuyo objetivo es ofrecer al usuario, de forma fácil e integrada, el acceso a una serie de recursos y de servicios, entre los que suelen encontrarse buscadores, foros, documentos, aplicaciones, compra electrónica, etc. Principalmente están dirigidos a resolver necesidades específicas de un grupo de personas o de acceso a la información y servicios de una institución pública o privada”. (SKYNETSTUDIO, 2008)

Skynetstudio concuerda en algo con Guglielmetti que un portal es un sitio que sirve para ofrecer al usuario el acceso a una serie de recursos y servicios como son: buscadores, foros, documentos, aplicaciones, compra electrónica entre otros. Además expone algo que no se había dicho hasta ahora que están destinado a resolver necesidades de un de un grupo de personas.

En Web21 se publicó en el año 2009: “Un portal opcionalmente podría ofrecer servicios de búsqueda: que incluye mecanismos de búsqueda, directorios y páginas amarillas para localizar negocios o servicios. Contenidos: es decir, información de varios temas como noticias, deportes, pronósticos de clima, listas de eventos locales, mapas, opciones de entretenimiento, juegos, ligas a estaciones de radio y a otros sitios con contenido especial en ciertas áreas de interés y facilidades de comercialización como son: anuncios clasificados para trabajos, actuar como un intermediario entre compradores y vendedores, visualizar anuncios, brindar correos electrónicos y ligas a otros sitios que también se dedican a la venta”. (WEB21, 2009)

En Web21 se plantean muchas de las opciones que ofrece un portal, como se puede apreciar no solo se encuentra informacion de diversos temas sino que sirve como fuente de busqueda y como intermediario entre compradores y vendedores.

En Indigo se plantea en el 2009: “Las ventajas que aporta un portal son diversas, ya que el mismo garantiza un sitio más interactivo donde la información está actualizada y sin límites, es de fácil navegación, es sencillo de usar para usuarios sin conocimientos técnicos, cualquiera con conocimientos sobre un procesador de palabras básico puede aprender fácilmente a gestionar información. Permite suscripción de usuarios (con o sin boletín) para ingresar a determinadas zonas protegidas y contacto con los mismos, recepción de noticias de otros usuarios para enriquecer el sitio y ver cuántos usuarios están visitando el sitio en tiempo real”. (INDIGO, 2009)

Los autores anteriores han hablado mucho sobre las informaciones y servicios que brindra un portal pero han hecho muy poca alusión a sus beneficios. Indigo abunda más en este tema y expone las facilidades de navegación, de uso y de administración que ofrece un portal.

Despés de analizar todas las definiciones consultadas anteriormente este trabajo define a un portal como un sitio web cuyo objetivo es ofrecer al usuario de forma fácil e integrada, el acceso a una serie de recursos y de servicios, de fácil navegación, accesible para todos los usuarios, que presente una interfaz gráfica amigable y contribuya a resolver necesidades específicas de un grupo de personas o de acceso a la información y servicios de una institución pública o privada.

1.3 ¿Qué es factoría de software?

Se le denomina factoría a cualquier fábrica o industria, del tipo que sea, en la cual se lleve a cabo la transformación de materias primas en otros productos, ya sea para su autoconsumo o para el consumo de otras industrias. Por extensión se está aplicando esta palabra para designar determinadas actividades en las cuales no se produce consumo y transformación de materias y que tienen como objeto final la obtención de productos intangibles: factoría de comunicación, factoría de cine, factoría de software.

A fines de los años 60´s e inicios de los 70´s, surge en la industria del software el concepto de fábrica de software. Este ha venido madurando y evolucionado a medida que se han definido técnicas y métodos, además de la aparición de diversas herramientas de software. Varios autores a lo largo de la historia han dado su definición del término:

Cusumano en 1989 declara: “Una empresa productora de software que no responda a características como: producción de software en gran escala, estandarización de tareas, estandarización del control, división del trabajo, mecanización y automatización, no puede ser considerada una factoría de software. El desarrollo de una factoría implica que las buenas prácticas de Ingeniería de Software sean aplicadas sistemáticamente”. (CUSUMANO, 1989).

Como bien plantea Cusumano una factoría debe tener definido roles y asignar responsabilidades, tener definido y estandarizado el proceso de producción y controlar en todo momento la producción, haciendo uso de las buenas prácticas de ingeniería de software. Pero el elemento analizable es la condición de tener producción a gran escala, dando a entender erróneamente que las pequeñas y medianas empresas con bajos niveles de producción no puedan utilizar el enfoque.

Cantone en 1992 precisa: “Una factoría de software debe, para ser flexible, ser capaz de producir varios tipos de productos; llevar a cabo conceptos de Ingeniería de Software (metodología, herramientas, gestión de la configuración), y también ser capaz de estudiar, diseñar, implementar y mejorar sus sistemas y procesos”. (CANTONE, 1992)

Una factoría puede tener una línea de producción y ser flexible porque sea capaz de adaptar los productos a diferentes entornos de implantación, con diferentes tecnologías y aplicaciones. Pero debe en todo momento estandarizar su producción a través de la aplicación de conceptos de ingeniería y como dice Cantone, ser capaz de mejorar este proceso de manera sistemática.

Basili en ese mismo año pronuncia: “Una organización con características de factoría de software debe poseer una estructura de construcción de software basada en componentes. Los componentes utilizados en la construcción del software pueden ser desarrollados por una unidad de producción de componentes (factoría de componentes). La factoría de componentes es la base para la implementación de una factoría de software”. (BASILI, 1992).

Fernstrom apoya a Basili cuando enuncia: “Una organización fabril para el desarrollo de software debe tener claro el asunto del ‘software único’, es decir, todo software es único, pero algunas partes de ellos se pueden repetir en varios proyectos. El proceso industrial debe contener el desarrollo, almacenamiento y montaje de partes reutilizables en un producto”. (FERNSTROM, 1992)

Basili y Fernstrom hablan del papel de los componentes reutilizables en la producción de software y centran toda la atención en este concepto, pero en la vida real para que un software tenga calidad, deben definirse los flujos de procesos, los roles, las responsabilidades y las técnicas de reutilización que propicien el uso eficiente de los elementos reutilizables.

Fernández y Teixeira en el 2004 plantean: “Una factoría de software es una organización con procesos estructurados, controlados y mejorados de forma continua, considerando principios de Ingeniería Industrial, orientados a dar respuesta a múltiples demandas de distintas naturaleza y alcance. Dirigida a la creación de productos de software, conforme a los requerimientos documentados de los usuarios y clientes, de la forma más productiva y económica posible”. (FERNÁNDEZ, 2004).

Fernández y Teixeira afirman que las características de los proceso en una factoría: deben ser estructurado, controlado y mejorado continuamente y que deben estar regidos por principios de ingeniería industrial para obtener productos de software de acuerdo a lo especificado por los clientes.

1.3.1 Aspectos importantes sobre las Factorías de Software

Yanosky Ríos La Hoz aborda los principales objetivos de una factoría de software que permiten industrializar su funcionamiento. “Estos objetivos son los siguientes:

- Industrializar el desarrollo de sistemas de software.
- Producción de software a gran escala.
- Lograr una alta productividad en el desarrollo de software.
- Establecer una línea de producción.
- Mejora continua de los procesos.
- Estimación de costos y plazos extremadamente precisa.
- Reducción de los costos de producción.
- Lograr una buena gestión de la calidad.
- Especializar al profesional en una tarea específica del proceso, concentrando sus esfuerzos en dicha tarea”.(RÍOS, 2005)

No existe un definición universal sobre factorías de software, por lo que basado en las definiciones anteriores, este trabajo asume factoría de software como una organización estructurada creada para el desarrollo de software, con procesos estandarizados, repetibles y mejorables continuamente, donde exista una fuerte comunicación con el cliente y las estimaciones se realicen basadas en los datos históricos, fruto de experiencias anteriores.

Debe ser una fábrica en la cual las actividades de desarrollo sean predecibles y se reduzca la cantidad de trabajo debido a la reutilización de componentes. Estar enfocada a un segmento del mercado, garantizar la calidad del software y contar con un grupo de herramientas estandarizadas para la construcción de software y la administración y control del proyecto. Debe, además, mantener capacitados a sus recursos humanos y definir roles y responsabilidades, concentrando sus esfuerzos en un área determinada de la producción.

Por tanto, se puede afirmar que una factoría de software es una organización dedicada a la producción de software pero con sus procesos industrializados, donde se comparten recursos y conocimiento según su línea de producción. Estas líneas agrupan los proyectos que trabajan sobre una misma plataforma tecnológica que posibilitan crear un esquema para el desarrollo y mantenimiento de sistemas similares.

1.3.2 Modelos de Factoría de Software

En este epígrafe se hace referencia a los modelos de factoría de software más significativos encontrados durante la investigación los cuales se encuentran respaldados por la experiencia de empresas y entidades que los han adaptado. Los modelos son los siguientes:

- Modelo basado en la norma ISO 9001 y CMM.
- Modelo Eureka.
- Modelo Clasificatorio.
- Modelo propuesto por Basili.
- Modelo Replicable.
- Modelo aplicando Inteligencia

1.3.2.1 Modelo basado en la norma ISO 9001 y CMM

"El modelo basado en la norma ISO 9001 y CMM es un modelo genérico en el que cualquier factoría de software puede adaptar las entidades que componen el mismo de acuerdo a sus características y necesidades. Este modelo divide la factoría de software en cinco entidades bien definidas: Técnicas, Proceso, Trabajadores involucrados, Gestión de la Factoría y Activos del proceso, herramientas y componentes de código". (TRUJILLO CASAÑOLA, 2007)

En el Anexo 1 se puede observar cada una de las entidades mencionadas y las relaciones entre ellas.

Este modelo es relevante principalmente por los estándares de calidad que definen su nombre, el uso de la norma ISO 9001 y de CMM garantizan que los procesos se desarrollen con la calidad que se necesita. El enfoque principal de este modelo está en el proceso del software en función de la mejora continua para lograr una buena estimación de los costos y tiempo. Otro elemento importante es que engloba las actividades en entidades, pero no las especifica en su totalidad, se queda muy general.

1.3.2.2 Modelo Eureka

El modelo fabril está compuesto de proceso, reglas, herramientas, información, trabajadores y equipamiento.

"El proceso de desarrollo está compuesto por reglas, las que son definidas por las personas involucradas en el ambiente de desarrollo de software y constituyen patrones a seguir, algoritmos, métodos de

desarrollo de software. Las herramientas e información almacenadas, soportan la automatización del proceso de desarrollo". (ROCKWELL, 1993)

En el Anexo 2 se muestra la arquitectura del mismo. El modelo propone un proceso de desarrollo de software distribuido, siguiendo el enfoque software bus (Ver Anexo 3). "A través de las reglas se puedan unir los componentes realizados por distintos equipos para formar el producto". (TRUJILLO CASAÑOLA, 2007)

El valor de este modelo se aprecia a la hora de hacer software distribuido, porque elimina las limitaciones geográficas, cada factoría de software puede trabajar por separado, guiándose por las reglas establecidas, y después realizar la unión de los componentes elaborados por cada una para formar el producto final. Un aspecto negativo es que no abarca temas como la gestión de proyectos, recursos, metodologías de desarrollo o el uso de estándares de calidad.

1.3.2.3 Modelo Clasificatorio

El Modelo Clasificatorio propuesto por Fernández y Teixeira está dirigido a clasificar las factorías de acuerdo al alcance o fases de desarrollo que compone el proceso definido en la factoría y las clasifica en: Factoría de Proyectos Ampliada, Factoría de Proyectos de Software, Factoría de Proyectos Físicos, Factoría de Programas. Estas clasificaciones se pueden observar en el Anexo 4.

"Una Factoría de Proyectos Ampliada comprende el concepto de arquitectura de solución.

La Factoría de Proyectos de Software abarca todo el ciclo de vida sistémico para la realización del software, correspondiente al análisis, diseño, implementación, prueba e implantación.

La Factoría de Proyectos Físicos se abstrae del enfoque sistémico del software, se dedica al diseño, implementación y prueba. No se tiene un pleno conocimiento del negocio.

La Factoría de Programas, considerada la menor de las entidades, tiene como objetivos desarrollar componentes de código para la construcción del software". (FERNÁNDEZ, 2004)

Este modelo aporta una clasificación para las factorías en dependencia de la magnitud del proyecto, sin embargo, solo se limita a clasificar su alcance, y deja sin definir áreas claves en la producción como es la gestión del proyecto, etc.

1.3.2.4 Modelo propuesto por Basili

El Modelo Propuesto por Basili divide una factoría de software en dos grandes entidades. “El modelo se divide en organización basada en proyectos de software (unidad de producción de software), y factoría de componentes (unidad de producción de componentes). La organización basada en proyectos realiza las solicitudes de productos (componentes para la construcción del software), de datos (estadística para la estimación de costo y plazos) y de planos (modelos, métodos para el análisis y diseño de software) a la factoría de componentes. La factoría de componentes posee una base de componentes reutilizables, de la cual se apoya para dar respuesta a las solicitudes hechas por la unidad de producción de software. En respuesta a la solicitud la organización basada en proyectos recibe los modelos y componentes para la construcción del software, además de estadísticas y datos históricos que se encuentran en la base de componentes”. (BASILI, 1992)

El modelo puede verse en el Anexo 5.

Este modelo se basa en la división de la factoría en dos unidades y las relaciones que se establecen entre ellas, esto es importante para aumentar la productividad; así cada unidad se especializa en determinadas actividades, ganando en experiencia, lo que apoyado con la reutilización de componentes, permite ganar en tiempo de desarrollo y en la efectividad de los productos desarrollados. Pero este se encuentra inconcluso pues sólo se enmarca en el área de producción dejando sin definir la gestión del proyecto, la organización de los desarrolladores y el uso de herramientas para la automatización de los procesos.

1.3.2.5 Modelo Replicable

“Este modelo plantea que una factoría de software debe poseer:

- Un modelo de organización de la producción.
- Una unidad de producción de componentes y una unidad de producción de software.
- Tanto la unidad de producción de componentes como la de software poseen un proceso.
- El proceso es guiado por un modelo de calidad de software.
- El proceso es compuesto de actividades compuestas de tareas.
- Las tareas utilizan los componentes, y estos son clasificados en infraestructura (o activos del proceso) y código.
- Las tareas usan un conjunto de herramientas para la automatización de las mismas.

- Por último el proceso puede ser aplicado al desarrollo de software o al desarrollo de un componente". (RÍOS, 2005)

En el Anexo 6 se encuentran las relaciones entre los conceptos.

Además describe las técnicas, herramientas, roles y actividades dentro del proceso. Este modelo define mejor el proceso de producción, está muy detallado por cada actividad pero no rige el proceso por estándares de calidad ni por metodologías de ingeniería de software reconocidas, además de que tampoco abarca completamente el área referida a la gestión de proyecto.

1.3.2.6 Modelo aplicando Inteligencia

"Este modelo de factoría de software fue propuesto por Yaimí Trujillo Casañola para la Universidad de las Ciencias Informática. Este se encuentra dividido en 6 entidades, las cuales se mencionan a continuación: Bases tecnológicas, Proceso, Personas, Repositorio de componentes, Gestión de proyecto e Inteligencia. Las relaciones entre estas entidades se pueden apreciar en el Anexo 7.

- Gestión de Proyecto: Comprende todas las áreas de la gestión de proyecto. Presenta las unidades de gestión, organización del proceso, gestión del capital humano y gestión de la calidad.
- Proceso: Comprende el conjunto de actividades que conforman el flujo de trabajo, el cual depende de la metodología que se utilice para guiar el desarrollo del proyecto.
- Personas: Comprende el capital humano involucrado con el proceso de desarrollo de software, la estructura organizativa y los roles que ocupan, está dividida en dos unidades: Gestores de la Factoría y Grupo de desarrollo.
- Inteligencia: Comprende los métodos que permitan la orientación estratégica de la factoría con el uso de herramientas de Vigilancia Tecnológica, Inteligencia Empresarial, Prospectiva. Presenta dos unidades: la interna de inteligencia organizacional y la externa de inteligencia empresarial.
- Bases tecnológicas: Comprende el contexto de las bases tecnológicas y herramientas, las técnicas y mecanismos para construir, soportar y gestionar el proceso de desarrollo.
- Repositorio de componentes: Comprende el almacenamiento y gestión de los activos del proceso y componentes de código. Entiéndase como activos del proceso formularios, documentos, patrones, algoritmos utilizados como artefactos en el proceso. Los activos del proceso también pueden ser denominados como componentes de infraestructura, componentes de valor en el proceso". (TRUJILLO CASAÑOLA, 2007)

Este modelo recoge las características más trascendentales de los modelos vistos anteriormente, utiliza estándares internacionales, define el proceso por metodologías de desarrollo y apoya la reutilización. Es más abarcador porque además de definir el proceso le brinda gran importancia a las áreas de gestión de proyecto, organización y calidad del proceso de desarrollo del software. También busca la manera de gestionar el conocimiento y la experiencia de la empresa y las características del entorno con la introducción de la entidad centro de inteligencia.

1.3.2.7 Selección del modelo de factoría de software

Después de analizar los diferentes modelos de factoría, así como sus principales características, se concluye que el modelo a utilizar en la Fábrica de Portales es el Modelo de Factoría aplicando Inteligencia, ya que es el más completo de todos y reúne las particularidades más importantes de los modelos anteriores.

Se escoge este modelo porque permite la definición de un proceso a partir de una metodología y el uso de estándares internacionales como ISO, CMMI, PSP y TSP para elevar la calidad del proceso de desarrollo y del producto final. Las personas que participan se organizan por roles y planifican sus actividades tanto personales como en equipo, usan TSP y PSP. Permite la reutilización de componentes lo cual propicia una mayor productividad. Se definen las bases tecnológicas para la gestión, soporte y construcción del producto. Tiene en cuenta la gestión del proyecto y la calidad del producto. Además usa inteligencia organizacional y empresarial para la orientación estratégica.

1.4 Ingeniería de software

La industria del software cada día cobra mayor auge, son miles las entidades incursionando en esta área. Revisando lo sucedido durante los últimos años se puede apreciar resultados asombrosos, de simples programas para cálculos matemáticos se ha transitado a complejos sistemas, con altas velocidades de procesamiento. Sin embargo, al analizar los datos estadísticos que se muestran en la Tabla 1.1 acerca de los fallos de proyectos, tomado de los CHAOS REPORT de Standish Group correspondientes a los años 1994 y 2002, (GROUP, 2006) se hace constar que a pesar de conocer el incremento en número de los proyectos de desarrollo, no pasa igual con la ejecución eficiente y eficaz de los mismos, cuyas mejoras se mantienen en niveles inferiores a los niveles de calidad que desarrolladores y clientes realmente necesitan.

Descripción	1994	2002
Proyectos completados en tiempo y según los costos previstos	16%	34%
Proyectos cancelados	31%	15%
Proyectos con grandes problemas de desarrollo	53%	51%
Promedio de costos sobrepasados	189%	43%
Planificación sobrepasada	222%	82%
Entregas vs. características solicitadas	61%	52%

Tabla 1.1 Resumen del CHAOS REPORT de Standish Group en los años 1994 y 2002

El origen de la ingeniería de software se debe precisamente a estos problemas, el desarrollo de software viene padeciendo de retrasos considerables en la planificación, poca productividad, elevadas cargas de mantenimiento, demandas cada vez más desfasadas con las ofertas, baja calidad y fiabilidad del producto, dependencia de los realizadores. Esto es lo que se ha denominado comúnmente crisis del software o aflicción crónica. Ello ha venido originado por una falta de formalismo, metodologías, herramientas de soporte y administración eficaz de los proyectos de desarrollo de software.

Según Pressman: “Sin tener en cuenta como lo llamemos, el conjunto de problemas encontrados en el desarrollo del software de computadoras no se limitan al software que no funciona correctamente. Es más, el mal abarca los problemas asociados a cómo desarrollar software, como mantener el volumen cada vez mayor de software existente y como poder esperar mantenernos al corriente de la demanda creciente del software”. (PRESSMAN, 2002)

Actualmente está surgiendo una gran expectativa ante la evolución de la Ingeniería del Software, al ir apareciendo nuevos métodos y herramientas formales que van a permitir en el futuro un planteamiento de ingeniería en el proceso de elaboración de software. Dicho planteamiento vendrá a paliar la demanda creciente por parte de los usuarios, permitiendo dar respuesta a los problemas.

Varias son las definiciones dadas sobre ingeniería de software entre ellas la de Bauer en 1972:“Ingeniería del Software trata del establecimiento de los principios y métodos de la ingeniería a fin de obtener software de modo rentable que sea fiable y trabaje en máquinas reales”. (BAUER, 1972)

Según el Bauer se busca aplicar producción de software en gran escala, estandarización de tareas, estandarización del control, división del trabajo, mecanización y automatización para desarrollar software de manera rentable y que sea funcional.

En 1976 enuncia Bohen: “Ingeniería de Software es la aplicación práctica del conocimiento científico en el diseño y construcción de programas de computadora y la documentación asociada requerida para desarrollar, operar (funcionar) y mantenerlos. Se conoce también como desarrollo de software o producción de software”. (BOEHM, 1976)

Bohen pone nuevos elementos cuando menciona aplicar el conocimiento científico, concibe la documentación como elemento importante en el desarrollo y por último no solo se enmarca en el desarrollo del producto hasta su construcción sino las actividades postproducción de implantación y mantenimiento.

Zelkovitz en 1979 puntualiza:”Ingeniería de software es el estudio de los principios y metodologías para el desarrollo y mantenimiento de sistema de software”. (ZELKOVITZ, 1979)

En este caso Zelkovitz coincide con Bauer en que la ingeniería no se enmarca en la producción de los productos sino que llega hasta las etapas de mantenimiento, pero su mayor aporte de es mencionar el papel de las metodologías para guiar el desarrollo del software.

En 1993 la IEEE establece como estándar el concepto: “La aplicación de un enfoque sistemático, disciplinado y cuantificable al desarrollo, operación (funcionamiento) y mantenimiento del software; es decir, la aplicación de ingeniería al software.” (IEEE, 1993)

La IEEE precisa nuevamente que no solo es el desarrollo sino también las etapas de operación y mantenimiento, pero menciona características importantes el enfoque del área de producción donde el trabajo sea sistemático, disciplinado y cuantificable.

Pressman en el 2002, expresa: “La Ingeniería de Software es una tecnología multicapa”. En la que, según él: “El proceso de la ingeniería del software es la unión que mantiene juntas las capas de tecnología y que permite un desarrollo racional y oportuno de la ingeniería del software”. “Los métodos de la ingeniería de software indican cómo construir técnicamente el software”. Y “Las herramientas de la ingeniería del software proporcionan un enfoque automático o semi-automático para el proceso y para los métodos”. (PRESSMAN, 2002)

Pressman es el primero en mencionar la interacción de los elementos: proceso, métodos y herramientas, así como el papel de cada uno de ellos, que son los que transforman el desarrollo de software de una actividad artesanal a una actividad industrial. Donde el enfoque es a aplicar los principios de la era industrial para alcanzar un desarrollo más moderno, estandarizado, repetible y mejorable continuamente, elevando la calidad y la productividad del equipo de desarrollo.

Estos autores concuerdan que la ingeniería de software está llamada a dejar atrás el desarrollo artesanal de sistemas por diferentes vías: el uso de los principios y métodos de la ingeniería, la aplicación práctica del conocimiento científico, el estudio de los principios y metodologías, la aplicación de un enfoque sistemático, disciplinado y cuantificable. Pero solo Pressman menciona el cómo lograrlo y es la fortaleza de definir un proceso acorde a las características del producto, estandarizado, repetible y mejorable continuamente, que para ello se base en los métodos y las herramientas.

En los próximos epígrafes se abordara los fundamentos teóricos para definir el proceso, la metodología y las herramientas.

1.5 Modelo de procesos de software

En la bibliografía consultada no existe una receta de cómo aplicar los elementos de la Ingeniería de Software de manera tal que se tribute al éxito de los proyectos, pero la mayoría de los autores referencian a Pressman en el siguiente planteamiento: “Un modelo de proceso del software es una estrategia de desarrollo que acompaña al proceso, métodos, capas de herramientas y las fases genéricas. Se selecciona un modelo de proceso para la ingeniería del software según la naturaleza del proyecto y de la aplicación, los métodos y las herramientas a utilizarse y los controles y entregas que se requieren”. (PRESSMAN, 2002)

En el análisis de este planteamiento se evidencia el papel del modelo de proceso de software como rector para definir los elementos de la Ingeniería de Software. Está definición no busca imponer un estándar a partir de algún método particular, ni definir un equipo completo de métodos, procesos y herramientas aceptables. Todos los elementos deben ser adaptados al contexto del proyecto (recursos técnicos y humanos, tiempo de desarrollo, tipo de sistema, etc.) Se considera que a partir de las propuestas cada proyecto o línea de desarrollo debe examinar sus necesidades, elegir, definir y justificar la selección dentro de la definición de los requerimientos del software. En este epígrafe se hace referencia a los modelos de proceso del software más importantes encontrados durante la investigación.

1.5.1 Modelo Lineal Secuencial (modelo en cascada)

Al decir de Royce “es el enfoque metodológico que ordena rigurosamente las etapas del ciclo de vida del software, de forma tal que el inicio de cada etapa debe esperar a la finalización de la inmediatamente anterior. Comienza en un nivel de sistemas y progresa con el análisis, diseño, codificación, pruebas y

mantenimiento". (ROYCE, 1970).El modelo lineal secuencial comprende las siguientes fases las fases como se muestra en el Anexo 8.

Sobre este modelo Hanna menciona "El modelo lineal secuencial es el paradigma más antiguo y más extensamente utilizado en la ingeniería del software. Sin embargo, la crítica del paradigma ha puesto en duda su eficacia" (HANNA, 1995)

Los proyectos reales raras veces siguen el modelo secuencial puro que se propone; debido a que el modelo lineal puede acoplar interacción con otros indirectamente. Las debilidades más abordadas de este modelo es debido a que su aplicación en proyectos con requisitos cambiantes puede causar confusión en el equipo de desarrollo sobre la fase en la que se encuentran, y por ende una versión del producto no estará disponible hasta que no haya avanzado el proyecto y puede que el software no cumpla con los requisitos del usuario por el largo tiempo de entrega del producto. Este modelo es de suma importancia, debe usarse solamente si se entienden a plenitud los requisitos y se adecua al desarrollo de productos sencillos y de poca complejidad, se desarrolla en un ciclo convencional de ingeniería donde una fase no empieza hasta que no termina la fase anterior.

1.5.2 Modelo Construcción de Prototipos

Sobre este modelo Brooks plantea "este modelo comienza con la recolección de requisitos. El desarrollador y el cliente encuentran y definen los objetivos globales para el software, identifican los requisitos conocidos y las áreas del esquema en donde es obligatoria más definición. Entonces aparece un diseño rápido que lleva a la construcción de un prototipo. El prototipo es evaluado por el cliente/usuario y se utiliza para refinar los requisitos del software a desarrollar. La iteración ocurre cuando el prototipo se pone a punto para satisfacer las necesidades del cliente, permitiendo al mismo tiempo que el desarrollador comprenda mejor lo que se necesita hacer". (BROOKS, 1975).

Los clientes por lo general definen un conjunto de procesos para la informatización, pero les es muy difícil identificar los requisitos explícitos. En otros casos, los desarrolladores no tienen una clara comprensión de los requisitos. En estas situaciones, un paradigma de construcción de prototipos que permita la validación y verificación de los requisitos desde etapas tempranas puede ofrecer un mejor enfoque y mitigar los riesgos asociados a la gestión y desarrollo de los mismos. Mitiga los efectos de los requisitos cambiantes. Este enfoque entorpece el desarrollo si la herramienta de modelado del prototipo es incompatible con el de desarrollo de la aplicación porque si no se duplicarían los esfuerzos para el desarrollo de la capa

interfaz. Este modelo se puede asociar a cualquiera de los modelos restantes para la etapa de Requerimiento. El modelo anterior se puede apreciar en el Anexo 9.

1.5.3 El modelo DRA (Desarrollo Rápido de Aplicaciones)

A partir de la propuesta del modelo en cascada y de la debilidad de no tener un producto hasta avanzado el proyecto, Martin propone: "El modelo DRA es una adaptación a alta velocidad del modelo lineal secuencial, permite construir sistemas utilizables en poco tiempo, normalmente de 60 a 90 días, frecuentemente con algunas concesiones. Se utiliza una construcción basada en componentes". (MARTIN, 1991) Este modelo se puede apreciar en el Anexo 10.

Como se aprecia en la figura del Anexo este modelo permite el desarrollo en paralelo de varios equipos y la integración posterior del producto o el desarrollo en varios ciclos de vida evolutivos o en espiral. El modelo permite construir sistemas utilizables en poco tiempo, utiliza una construcción basada en componentes y por ende sus ventajas, los entregables pueden ser fácilmente trasladados a otra plataforma y permite visibilidad temprana, una mayor flexibilidad y la codificación manual es menor. Tiene como limitantes para proyectos grandes que requieren recursos humanos suficientes y no es adecuado cuando los riesgos técnicos son altos.

1.5.4 Modelos Evolutivos del proceso del software

En la mayoría de los proyecto grandes mayores de 40 casos de uso, los requisitos del producto cambian conforme el desarrollo procede, haciendo que el camino que lleva al producto final constantemente tenga que ser redefinido; los compromisos y las exigencias del cliente impulsan a que sea imposible finalizar un producto completo, por lo que se debe introducir un enfoque que propicie cumplir la presión competitiva. Muchas veces en estos casos se consigue comprender perfectamente el conjunto de requisitos del producto en etapas avanzadas del desarrollo por lo que se necesita un modelo de proceso que sea diseñado para obtener parte del software en la medida que se vaya comprendiendo.

Una solución a esto es la propuesta de McDermid y Rook donde "Los modelos evolutivos son iterativos. Se caracterizan por la forma en que permiten a los ingenieros del software desarrollar versiones cada vez más completas del software. Cada secuencia lineal produce un incremento del software." (MCDERMID, 1993) Los modelos evolutivos tiene tres vertientes; incremental, en espiral y desarrollo concurrente.

Según estos autores el incremental "Combina elementos del modelo de cascada (aplicados repetitivamente) con la filosofía interactiva de construcción de prototipos. El primer incremento es un producto esencial (núcleo), se afrontan requisitos básicos y muchas funciones extras (conocidas o no) quedan para los siguientes incrementos. El cliente usa el producto central y en base a la utilización y/o evaluación se desarrolla un plan para el incremento siguiente. Este proceso se repite hasta que se elabora el producto completo. El modelo incremental entrega un producto operacional en cada incremento. El desarrollo incremental es un modelo útil cuando no está disponible el personal que se necesita para una implementación completa en la fecha límite que se haya establecido para el proyecto". (MCDERMID, 1993) Este modelo se puede apreciar en el Anexo 11.

El modelo evolutivo en espiral fue propuesto inicialmente por Boehm y a su juicio "Es un modelo que conjuga la naturaleza iterativa de construcción de prototipos con los aspectos controlados y sistemáticos del modelo lineal secuencial. En el modelo espiral, el software se desarrolla en unas series de versiones incrementales, durante las primeras iteraciones, la versión incremental puede ser un modelo en papel o un prototipo y durante las últimas iteraciones, se producen versiones cada vez más completas del sistema. El modelo se divide en un número de actividades estructurales llamadas regiones de tareas que pueden ser de tres a seis". (BOEHM, 1998) Este modelo se puede apreciar en el Anexo 12

Para explicar el modelo concurrente Davis y Sitaram han descrito un ejemplo "un cambio de requisitos durante el último desarrollo implica que se están escribiendo requisitos, diseñando, codificando, haciendo pruebas y probando la integración todo al mismo tiempo."(DAVIS, 1994)

Generalmente este modelo se utiliza en grandes proyectos de desarrollo, con equipos grandes y que los requisitos sean cambiantes, en el caso específico de la vertiente concurrente se utiliza a menudo como el paradigma de desarrollo de aplicaciones cliente/servidor. Al decir de Sheleg "Cuando se aplica a cliente/servidor, el modelo de proceso concurrente define actividades en dos dimensiones una dimensión de sistemas y una dimensión de componentes. Los aspectos del nivel de sistemas se afrontan mediante tres actividades: diseño, ensamblaje y uso. La dimensión de componentes se afronta con dos actividades: diseño y realización. La concurrencia se logra de dos formas: las actividades de sistemas y de componentes ocurren simultáneamente y pueden modelarse con el enfoque orientado a objeto... una aplicación cliente/servidor típica se implementa con muchos componentes, cada uno de los cuales se pueden diseñar y realizar concurrentemente."(SHELEG, 1994)

De manera general los clientes no tienen que esperar hasta que el sistema se entregue completamente para comenzar a hacer uso de él, se pueden aclarar los requisitos que no tengan claros conforme ven las entregas del sistema, se disminuye el riesgo de fracaso de todo el proyecto, puede aplicarse a cualquier proyecto sin importar la complejidad, es apropiado para sistemas orientados a objeto, se puede aplicar el enfoque de construcción de prototipos en cualquier etapa de evolución del producto.
Tiene como limitantes la definición de los requisitos para cada incremento, el cual debe ser pequeño para limitar el riesgo, aumentar la funcionalidad es difícil para establecer las correspondencias de los requisitos contra los incrementos y detectar las unidades o servicios genéricos para todo el sistema los cuales se desarrollan en las primeras etapas, esto incide en que le proyecto tienda a ser incontrolable.

1.5.5 Desarrollo basado en componentes

El uso del paradigma Orientado a Objetos ha creado el marco de trabajo idóneo para introducir el modelo de desarrollo basado en componentes. Según Nierstrasz: “El modelo de desarrollo basado en componentes conduce a la reutilización del software, lo que beneficia a los ingenieros de software. Es evolutivo por naturaleza, y exige un enfoque iterativo para la creación del software. Configura aplicaciones desde componentes preparados de software llamados clases”. (NIERSTRASZ, 1992) La estructura de este modelo se puede apreciar en el Anexo 13.
Este modelo permite alcanzar un alto índice de productividad y la reutilización del software, reduce el tiempo de entrega, el costo y esfuerzo del proyecto, disminuye los riesgos durante el desarrollo. Pero requiere de actividades para identificar los componentes, así como su clasificación y almacenamiento en repositorios, el flujo de procesos demanda del ensamblaje de componentes a través de la actividad de ingeniería. Reúne en él las características más importantes de los modelos anteriores.

1.5.6 Consideraciones sobre los Modelos de Proceso de Software

Después de analizar los distintos modelos de producción y ver las características de cada uno se desarrolló una tabla de comparación entre los modelos basado en las ventajas y desventajas de cada uno, para ello ver el Anexo 14.
Basado en los elementos más significativos de cada uno y teniendo en cuenta las particulares de la línea de producción de ser desarrollos cortos entre uno y dos meses con equipos pequeños de 5 a 10 personas, donde los requisitos pueden ser modelados íntegramente en prototipos, los despliegues deben ser

inmediatos, el estilo arquitectónico es Cliente/Servidor. Se seleccionaron dos para combinarlos y utilizarlos en el proyecto, los cuales son el Modelo de Construcción de Prototipos y el Modelo de Desarrollo Rápido de Aplicaciones.

El segundo combina la fortaleza de un desarrollo basado en componentes, al modelo en cascada, con la idea de línea de producción y montaje del proceso industrial; es posible notar también una fuerte presencia de intercambio e integración de piezas, en el caso de la línea de producción de software, esas piezas son denominadas componentes de software. Permite además el desarrollo evolutivo en cualquiera de sus vertientes incremental, espiral o concurrente incorporando por tanto sus ventajas. Para mitigar sus desventajas se propone combinarlo con el primero que se menciona para así tener una mejor comprensión y compromisos con los requisitos del sistema.

La propuesta incide fuertemente en la reducción del tiempo de desarrollo de los proyectos y por ende en los costos, elevar los niveles de productividad y mitigar los riesgos técnicos, al basarse en componente y validar y verificar los requisitos a través de prototipos. Permite que el sistema en caso necesario pueda ser fácilmente desarrollado en otra plataforma, la codificación manual se disminuye y es más flexible. Si el sistema es grande se pueden desarrollar a través de varios equipos trabajando de manera paralela, o en espiral o con un equipo que trabaje en iteraciones.

Es importante señalar que requiere de desarrolladores y clientes comprometidos con actividades rápidas, cuando el proyecto es grande se necesita de una organización de muchos equipos y demanda de mayor efectividad en la gestión de proyecto; en sistemas en los que no se pueda definir una arquitectura claramente o los componentes que la componen, este enfoque puede que no obtenga los resultados que se esperan.

1.6 Proceso de desarrollo de software

El proceso de desarrollo de software según Jacobson "es aquel en que las necesidades del usuario son traducidas en requerimientos de software, estos requerimientos transformados en diseño y el diseño implementado en código, el código es probado, documentado y certificado para su uso operativo". Concretamente "define quién está haciendo qué, cuándo hacerlo y cómo alcanzar un cierto objetivo". (JACOBSON, 1998)

Jacobson concibe el proceso de desarrollo de software en diferentes etapas en el que las necesidades del cliente se transforman en el desarrollo de diferentes actividades para obtener el producto final y que

establece lo que debe hacer los desarrolladores, cuando y como para obtener el producto final. Estos elementos generalmente se responden al configurar una metodología para el proceso. El proceso debe tener un diagrama de flujo que describa el orden lógico de actividades. Estas son asignadas a un rol, además debe tener tareas para ejecutarlas y artefactos de entrada y salidas. Los artefactos terminados deben cumplir las especificaciones de calidad que se le propone en la lista de chequeo. Mediante la cual se especifican una serie de puntos a evaluar. En todas las fases del desarrollo se deben tener en cuenta estos puntos, en función de que al finalizar cada actividad haya que hacer el menor número de correcciones posibles.

Por su parte Presman expresa “El proceso es un diálogo en el que se reúne el conocimiento y se incluye en el software para convertirse en software. El proceso proporciona una interacción entre los usuarios y los diseñadores, entre los usuarios y las herramientas de desarrollo, y entre los diseñadores y las herramientas de desarrollo.” (PRESSMAN, 2005).

En este caso Pressman incluye que la interacción no solo es entre desarrolladores y clientes sino que a su vez estos interactúan con las herramientas que semiautomatizan o automatizan el proceso. Incluye por tanto un nuevo elemento a tener en cuenta al describir un proceso y es poner las herramientas a utilizar para desarrollar las actividades. En la Figura 1.1 se representan los elementos que componen un proceso y su interrelación.

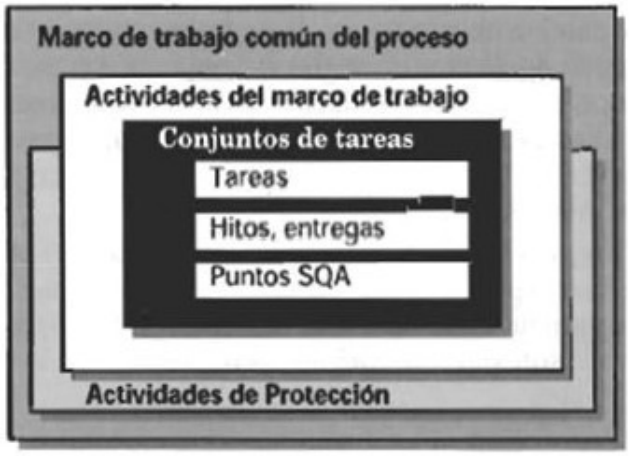


Figura 1.1 El proceso del software.

El proceso juega un papel determinante para el éxito de un proyecto y la satisfacción del cliente, en este sentido dice Margaret Davis: “Si el proceso es débil, el producto final va a sufir indudablemente”. (DAVIS, 1995)

Sobre el mismo tema refuerzan Grady Booch, James Rumbaugh e Ivar Jacobson: “En la ingeniería de software el objetivo es construir un producto de software o mejorar uno existente. Un proceso efectivo proporciona normas para el desarrollo eficiente de software de calidad. Capturar y presentar las mejores prácticas que el estado actual de la tecnología permite. En consecuencia, reduce el riesgo y hace el proyecto más predecible. El efecto global es el fomento de una visión y una cultura global.”(Jacobson, 2000)

El proceso de desarrollo de software no es único; no existe un proceso de software universal que sea efectivo para todos los contextos de proyectos de desarrollo. Debido a esta diversidad, es difícil encontrar en la literatura un proceso que sirva a la medida para los proyectos, por esta razón cada proyecto o línea de producción debe elegir, definir y justificar la metodología que guía su proceso a partir del modelo de producción y los requerimientos del software.

1.7 Metodologías de desarrollo de software

Las metodologías de desarrollo de software surgieron a raíz de la necesidad de controlar y documentar proyectos cada vez más complejos, impulsadas principalmente por instituciones económicamente importantes y con requisitos de seguridad y fiabilidad en sus sistemas sumamente estrictos.

La palabra metodología proviene de método, que quiere decir poner orden, organizar, ordenar, sistematizar. Se puede decir que una metodología es un conjunto de métodos utilizados en una investigación o proceso. Según la norma 1074 de IEEE toda “metodología de desarrollo de software debe incluir la forma en que se va a realizar la captura de requisitos, el diseño, la implementación y prueba”. (IEEE, 1997)

En la situación actual el uso correcto de una metodología adquiere una importancia mayor debido a las tendencias de desarrollo en equipo y con la finalidad de eliminar de una vez el caos existente en el mundo del desarrollo de software.

Las metodologías se basan en una combinación de los modelos de proceso genéricos (cascada, evolutivo, incremental, etc.). Estas son básicamente un conjunto de procedimientos que imponen una serie de pasos sobre el desarrollo del software, permiten producir y mantener un producto garantizando su

fiabilidad y calidad. No existe una metodología aplicable a todos los proyectos, debe ser seleccionada de acuerdo al contexto del mismo: recursos técnicos y humanos, tiempo de desarrollo, tipo de sistema, etc.
En la actualidad las metodologías se clasifican en robustas o tradicionales y ligeras o ágiles.
Las tradicionales describen un proceso de desarrollo de software configurable que se adapta a través de los proyectos a variados tamaños y diferentes niveles de complejidad. Se basa en muchos años de experiencia en el uso de la tecnología orientada a objetos y en el desarrollo de software. Se centran en el control del proceso, establecen de manera rigurosa las actividades involucradas, los artefactos que se crean y las herramientas y notaciones que serán usadas. Dentro de estas metodologías se encuentran: Microsoft Solution Framework (MSF) y Proceso Unificado de Desarrollo (RUP, Rational Unified Process), esta última es un proceso de desarrollo de software que utiliza como notación Lenguaje Unificado de Modelado (UML).
Las metodologías ágiles se basan en iteraciones muy cortas y le dan mayor valor al individuo, dentro de estas metodologías se destacan SCRUM, Adaptive Software Developmen (ASD), Crystal Methodologies, XP (eXtreme Programming), etc. Aunque los creadores e impulsores de las metodologías más populares coinciden con los principios enunciados anteriormente, cada metodología tiene características propias y hace hincapié en algunos aspectos más específicos. En el Anexo 15 se muestra una comparación entre ellas.
A continuación se resumen características de tres de ellas, estas son utilizadas con éxito en proyectos reales pero les faltaba una mayor difusión y reconocimiento.

Proceso Unificado de Desarrollo, RUP

El Proceso Unificado de Desarrollo, RUP por sus siglas en inglés es creado en la compañía Rational, donde confluyen 'los tres amigos' como se llaman a sí mismos o los tres grandes Grady Booch, James Rumbaugh e Ivar Jacobson en tres décadas de trabajo.
Al decir de Letelier “El objetivo de RUP es asegurar la producción de software de alta calidad, que satisfaga los requerimientos de los usuarios finales (respetando el plan y el presupuesto). Se caracteriza por estar guiado por los casos de uso, estar centrado en la arquitectura y ser iterativo e incremental. RUP brinda la posibilidad de elegir el conjunto de componentes de procesos que concuerdan con las necesidades del proyecto. Al ser partidario de la unificación del equipo de trabajo asigna responsabilidades, artefactos y tareas de manera que cada miembro individualmente perciba lo que tiene que hacer y como lo tiene que hacer”. (LETELIER, 2002)

Por otra parte el Grupo Soluciones Innova publica en su sitio: “Si se persigue la línea que propone RUP es viable, entrega un producto a tiempo y con confianza. RUP es una plataforma manejable de procesos de desarrollo que brinda muchísimos beneficios. Solo RUP provee un Framework de proceso configurable mediante el cual se pueden seleccionar e implementar los componente adecuados para lograr que dicho proceso sea sólido y estable. Es capaz de que el proceso sea práctico, con guías para facilitar el despegue de la planificación del proyecto, además de una detallada documentación para cuando se haga la entrega del producto el cliente tenga la posibilidad de consultarlas y entender el software”. (GSI, 2007)

En esencia esta metodología guía a los equipos de proyecto en cómo administrar el desarrollo iterativo de un modo controlado mientras se balancean los requerimientos del negocio, el tiempo al mercado y los riesgos del proyecto. El proceso describe los diversos pasos involucrados en la captura de los requerimientos y en el establecimiento de una guía arquitectónica lo más pronto, para diseñar y probar el sistema hecho de acuerdo a los requerimientos y a la arquitectura. El proceso describe qué entregables producir, cómo desarrollarlos y también provee patrones. El proceso unificado es soportado por herramientas que automatizan entre otras cosas, el modelado visual, la administración de cambios y las pruebas. Se caracteriza básicamente por ser vital la captura de requisitos, iteración actual condicionada por la anterior, se necesita de un buen líder de proyecto para garantizar el trabajo del equipo de desarrollo, se realiza un gran número de artefactos lo que puede provocar retrasos por mala preparación de los analistas, las responsabilidades están divididas y es aplicable a todo tipo de proyecto asumiendo sus extensiones.

Programación Extrema, XP

Programación Extrema, XP por sus siglas en inglés es la metodología ágil más difundida. Sobre ella Manuel Calero Solin expresa: “Germina como nueva disciplina para desarrollar software cerca de unos seis años atrás, pero ha causado una gran conmoción para los desarrolladores a nivel mundial. Su objetivo es muy simple, solo satisfacer al cliente e incrementar colosalmente el trabajo en equipo, donde los líderes del proyecto, clientes y programadores en si forman parte del grupo y están involucrados en todo el desarrollo del software”. (CALERO SOLIS, 2003)

Así mismo Beck explica: “Está centrada en potenciar las relaciones interpersonales como clave para el éxito en desarrollo de software, promoviendo el trabajo en equipo, preocupándose por el aprendizaje de los desarrolladores, y propiciando un buen clima de trabajo. Se basa en realimentación continua entre el cliente y el equipo de desarrollo, comunicación fluida entre todos los participantes, simplicidad en las

soluciones implementadas y coraje para enfrentar los cambios. XP se define especialmente adecuada para proyectos con requisitos imprecisos y muy cambiantes, y donde existe un alto riesgo técnico". (BECK, 2000)

Una de las principales características de esta metodología es que se centra en resolver el problema lo más rápido posible. Cada miembro del equipo debe estar listo para enfrentar cualquier problema y la instrumentación de los comentarios "El cliente se introduce en el equipo de desarrollo", "Hago algo y lo pruebo", "Termino todo y después integro".

Los obstáculos más comunes surgidos en proyectos XP al decir de Larman: "son la "fantasía" de pretender que el cliente se quede en el sitio y la resistencia de muchos programadores a trabajar en pares". (LARMAN, 2004)

Craig Larman señala como factores negativos la ausencia de énfasis en la arquitectura durante las primeras iteraciones (no hay arquitectos en XP) y la consiguiente falta de métodos de diseño arquitectónico. Un estudio de casos de Bernhard Rumpe y Astrid Schröder sobre 45 proyectos reveló que las prácticas menos satisfactorias de XP han sido "la presencia del usuario en el lugar de ejecución y las metáforas, que el 40% de los encuestados no comprende para qué sirven o cómo usarlas". (BERNHARD, 2001)

SCRUM

SCRUM ha sido desarrollada por Ken Schwaber, Jeff Sutherland y Mike Beedle. Define un marco para la gestión de proyectos, que se ha utilizado con éxito durante los últimos 10 años.

Martin en su libro sobre la esencia de la metodología expresa: "Está especialmente indicada para proyectos con un rápido cambio de requisitos. Sus principales características se pueden resumir en dos: El desarrollo de software se realiza mediante iteraciones, denominadas sprints, con una duración de 30 días. El resultado de cada sprint es un incremento ejecutable que se muestra al cliente. La segunda característica importante son las reuniones a lo largo proyecto, entre ellas destaca la reunión diaria de 15 minutos del equipo de desarrollo para coordinación e integración". (SCHWABER, 1999)

Por su parte Juan Palacio puntualiza: "Es una metodología de desarrollo muy simple, que requiere trabajo duro porque no se basa en el seguimiento de un plan, sino en la adaptación continua a las circunstancias de la evolución del proyecto.

- Es un modo de desarrollo de carácter adaptable más que predictivo.
- Orientado a las personas más que a los procesos.

- Emplea la estructura de desarrollo ágil: incremental basada en iteraciones y revisiones". (PALACIO, 2006)

La metodología propicia el desarrollo iterativo e incremental, y por ende asume las ventajas de este modelo de proceso de desarrollo. Conduce a que el cliente no tiene que esperar hasta que el sistema se entregue completo para usarlo, se pueden aclarar los requisitos, disminuye el riesgo de fracaso de todo el proyecto, puede aplicarse a cualquier proyecto sin importar la complejidad. Tiene como limitante la definición de los requisitos para cada Sprint. Al igual que XP no centra su desarrollo en la arquitectura.

1.7.1 Consideraciones sobre las metodologías

Las metodologías seleccionadas no son fáciles de comparar entre sí conforme a un pequeño conjunto de criterios. XP y RUP han definido claramente sus procesos, mientras que los de Scrum son más difusos en ese sentido, limitándose a un conjunto de principios y valores. Lo que tienen en común es su modelo de desarrollo incremental (pequeñas entregas con ciclos rápidos), cooperativo (desarrolladores y usuarios trabajan juntos en estrecha comunicación), directo (el método es simple y fácil de aprender) y adaptativo (capaz de incorporar los cambios). Pero solo RUP establece un desarrollo centrado en la arquitectura y guiado por casos de uso lo que se alinea con la propuesta de la línea de producción que se propone. Estructuralmente se asemejan al modelo DRA (Desarrollo Rápido de Aplicaciones) por el desarrollo en iteraciones y basada en componentes, con un ciclo de ingeniería claro en sus 6 disciplinas, propone una etapa de diseño independiente de la plataforma en el Análisis lo que coincide con que puede ser desarrollado en varias plataforma. Su guiado en caso de uso propone el desarrollo de prototipos evolutivos lo que alinea con el modelo de construcción de prototipos.

Debido a que el software que se pretende desarrollar es de ciclos cortos con equipos pequeños la metodología se debe aligerar. A esto se le suma la comparación de los elementos necesarios para guiar el proceso (roles, artefactos, actividades y flujo de trabajo) que se presenta en el Anexo 15. Donde se puede evidenciar que de los tres es el más descrito y completo para alinearse con los Lineamientos de Calidad y el Expediente de Proyecto establecido de manera obligatoria por la Dirección de Calidad en la Universidad.

1.8 Bases tecnológicas

Contar con aplicaciones desarrolladas en computadoras posibilita un acceso rápido y fácil a la información, una buena gestión de la información conduce a tomar la decisión más acertada en cada momento. Frases como esta son comúnmente usadas para motivar a las entidades a la informatización masiva y ordenada de sus procesos, en la búsqueda de elevar sus niveles de planificación, gestión y producción.

Sobre las herramientas dice Bandinelli: “Se han desarrollado herramientas de tecnologías de procesos para ayudar a las organizaciones de software a analizar los procesos actuales, organizar tareas de trabajo, controlar y supervisar el progreso y gestionar la calidad técnica. (BANDINELLI, 1995)

Los autores Booch, Rumbaugh y Jacobson en su libro Proceso Unificado de Desarrollo plantean: “Las herramientas soportan los procesos de desarrollo de software modernos. Hoy, es impensable desarrollar software sin utilizar un proceso soportado por herramientas. El proceso y las herramientas vienen en el mismo paquete: las herramientas son esenciales en el proceso”. (JACOBSON, 2000)

De la responsabilidad de esta capa en la ingeniería de software expresa Pressman: “Las herramientas de la Ingeniería del software proporcionan un enfoque automático o semi-automático para el proceso y para los métodos”. (PRESSMAN, 2002)

Todos estos autores concuerdan en el papel de las herramientas como parte fundamental de la ingeniería de software y tributa a la efectividad de los procesos, elevan los niveles de productividad y disminuyen los riesgos técnicos.

La amplia variedad de tecnologías existentes para el desarrollo de productos constituye un factor importante para decidir cuales utilizar. En el libro de Proceso Unificado de Desarrollo se mencionan cuales deben ser las herramientas que dan soporte al ciclo de vida completo (JACOBSON, 2000)

A continuación se presentan las herramientas consultadas para definir cuáles serán las utilizadas en la factoría.

Herramientas para la gestión de proyectos o para la coordinación de equipos de trabajo:

DotProject: “Ofrece un marco completo para la planificación, gestión y seguimiento de múltiples proyectos para clientes diferentes, está construido por aplicaciones de código abierto. Es una aplicación basada en web, multiusuario, soporta varios lenguajes y es Software libre. Está programada en PHP, y utiliza MySQL como base de datos (aunque otros motores como Postgre también pueden ser utilizados). La plataforma recomendada para utilizar DotProject se denomina LAMP (Linux + Apache + MySQL + PHP)”. (TID, 2007)

Gforge: "Es una herramienta basada en la web de gestión de proyectos y colaboración de software, creada para el desarrollo de software en forma comunitaria que permite organizar y administrar gran cantidades de proyectos, ofrece alojamiento de proyectos, control de versiones (CVS y Subversion), de seguimiento de fallos, de mensajería y salvas automáticas(Bacula). Está licenciado bajo la licencia GPL (GNU General Public License)". (ALTAMIRANDA, 2009)

Sistemas de gestión de contenido (CMS)

Joomla: "Es un sistema que permite crear sitios web de alta interactividad, profesionalidad y eficiencia. Su interfaz administrativa es sencilla y amigable. Con Joomla se puede crear sitios web de noticias, sitios corporativos, sitios web de presencia, portales comunitarios, e incluso también pueden crearse sistemas que funcionen en redes cerradas (Intranets) para gestionar información interna de compañías o empresas de negocios. Está programado en lenguaje PHP y SQL, utiliza bases de datos relacionales, más específicamente MySQL. Es de uso gratuito, de libre distribución, y de código abierto, se usa y distribuye bajo licencia pública general (GNU/GLP). Del mismo modo tiene como plataformas compatibles GNU/Linux, Windows y Mac OSX". (COMUNIDAD JOOMLA, 2008)

Drupal: "Es un programa de código abierto, con licencia GNU/GPL, escrito en PHP, desarrollado y mantenido por una activa comunidad de usuarios. Poderoso CMS muy conocido por la calidad de su código y por la seguridad que brinda, es estable y de actualización continua, configuración sencilla, instalación ágil, importante cantidad de módulos, temas visuales y excepcional documentación. Permite publicar artículos, imágenes, u otros archivos y servicios añadidos como foros, encuestas, votaciones, blogs y administración de usuarios y permisos". (REYERO, 2006)

Herramientas de diseño de páginas web:

Quanta Plus (Quanta+): "Es una herramienta libre de desarrollo de páginas web diseñado para el proyecto KDE, pero funciona también en el entorno GNOME. Usa asistentes para creación de tablas, enlaces y páginas en blanco, soporta una multitud de lenguajes como HTML, JavaScript, CSS, PHP, SQL y varios más, contiene un analizador que informa acerca de la correcta creación de las páginas y los documentos pueden ser previsualizados dentro de la aplicación usando el motor KHTML, facilita el control de las distintas versiones, presenta soporte de extensiones/plugins para añadir funcionalidades extra y es de instalación muy sencilla". (QUANTA PLUS, 2005)

Dreamweaver: "Puede utilizarse para diseñar páginas profesionales y proporciona una potente combinación de herramientas visuales de diseño, funciones y soporte para la edición del código. Soporta

gran cantidad de tecnologías fáciles de usar, como hojas de estilo y capas, JavaScript para crear efectos e interactividades y permite la inserción de archivos multimedia. Además presenta la función de autocompletar, disponible en varios idiomas y resaltado de la sintaxis para instrucciones en HTML y lenguajes de programación como PHP, JSP o ASP. Plataforma Windows y Mac, incluye Framework para AYAX, administrador CSS, compatibilidad con dispositivos móviles, integración con Adobe Photoshop, Adobe Fireworks". (DE GRACIA, 2008)

Herramientas de modelado:

Visual Paradigm: "Es una herramienta UML profesional que soporta el ciclo de vida completo del desarrollo de software: análisis y diseño orientados a objetos, construcción, pruebas y despliegue. El software de modelado UML ayuda a una más rápida construcción de aplicaciones de calidad, mejores y a un menor coste. Permite dibujar todos los tipos de diagramas de clases, código inverso, generar código desde diagramas. Además, la herramienta es colaborativa, es decir, soporta múltiples usuarios trabajando sobre el mismo proyecto; genera la documentación del proyecto automáticamente en varios formatos como Web o Pdf y permite control de versiones. Cabe destacar igualmente su robustez, usabilidad y portabilidad". (FREE DOWNLOAD MANAGER, 2007)

"Es software propietario y tiene licencia comercial". (GIRALDO, ZAPATA, 2005)

Rational Rose: "Es una herramienta con plataforma independiente que ayuda a la comunicación entre los miembros del equipo, a monitorear el tiempo de desarrollo y a entender el entorno de los sistemas. Utiliza la notación estándar en la arquitectura de Software (UML), la cual permite a los arquitectos de software y desarrolladores visualizar el sistema completo utilizando un lenguaje común. Abarca todo el ciclo de vida de un proyecto: concepción y formalización del modelo, construcción de los componentes, transición a los usuarios y certificación de las distintas fases. Los diseñadores pueden tomar ventajas de esta herramienta, porque permite que la salida de una iteración sea la entrada de la próxima que está por venir. Puede generar código en Java, C++, Visual Basic. Ada, Corba y Oracle". (MENENDEZ, 2007)

Sistemas de gestión de base de datos:

MySQL: "Es un sistema de gestión de base de datos relacional, multihilo y multiusuario. Es muy utilizado en aplicaciones web y en plataformas (Linux/Windows-Apache-MySQL-PHP/Perl/Python), y por herramientas de seguimiento de errores como Bugzilla. Su popularidad como aplicación web está muy ligada a PHP, que a menudo aparece en combinación con MySQL. Presenta características como son:

conectividad segura, transacciones y claves foráneas, replicación, búsqueda e indexación de campos de texto y disponibilidad en gran cantidad de plataformas y sistemas". (PECOS, 2009)

PostgreSQL: "Cuenta con sofisticadas particularidades tales como la versión multi-control de concurrencia (MVCC), replicación asincrónica, transacciones jerarquizadas, en línea, un sofisticado plan de consulta, y un excelente optimizador. Es altamente escalable, tanto en la formidable cantidad de datos que puede administrar como en la cifra de usuarios concurrentes que puede acomodar. Trabaja en varios sistemas operativos como Linux, UNIX y Windows. También resiste el acaparamiento de grandes objetos binarios, ya sean imágenes, sonidos o vídeo. Tiene interfaces de programación originario de C / C + +, Java, Net, Perl, Python, Ruby, Tcl, ODBC, entre otros". (PECOS, 2009)

Entorno de desarrollo integrado:

NetBeans: "Es una herramienta para programadores pensada para escribir, compilar, depurar y ejecutar programas. Está escrito en Java pero puede servir para cualquier otro lenguaje de programación, es un producto libre y gratuito sin restricciones de uso. Es fácil de instalar y de uso instantáneo y se ejecuta en varias plataformas incluyendo Windows, Linux y Mac OSX y Solaris. Puede obtener todas las herramientas que necesite para crear aplicaciones profesionales para el escritorio, la empresa, la web y equipos móviles con el lenguaje Java, C/C++, y Ruby. Todas las funciones del IDE son provistas por módulos". (SUN, 2007)

Zend Studio: "Editor web orientado a la programación de páginas PHP, con ayudas en la gestión de proyectos y depuración de código. Es un programa que aparte de funcionar como editor de texto para páginas PHP, garantiza una serie de ayudas muy útiles para los desarrolladores. Zend Studio implementa además unas interesantes opciones para trabajar en grupo, al integrar el sistema de trabajo conocido como CVS". (ÁLVAREZ, 2003)

Incluye todos los componentes necesarios durante el ciclo de vida de una aplicación en PHP. Incluye editor, análisis, optimizadores de código y herramientas de base de datos. Permite agilizar el desarrollo web y permite simplificar proyectos complejos. Incorpora el Framework de Zend, PHP documentor, manual de PHP. Integración con subversión, los navegadores, integración avanzada con FTP. Soporte para Web Services, PHP4, PHP5 y SQL.

Servidor Web:

Servidor Apache: "Servidor HTTP de código abierto, está patentado por licencia BSD. Muestra entre otras características mensajes de error altamente configurables, bases de datos de autenticación y

negociado de contenido, es altamente adaptable, utiliza una gran suma de Perl, PHP y otros lenguajes de script pero también trabaja con Java. Ventajas que presenta: es modular, open source, multi-plataforma, extensible y popular ya que es muy fácil conseguir ayuda y soporte". (THE APACHE SOFTWARE FOUNDATION, 2009)

1.8.1 Selección de las tecnologías

Después de un profundo estudio investigativo sobre las bases tecnológicas que quedarán establecidas en la factoría de software (Ver Anexo 16) se escogió para la Gestión de proyecto el Gforge por integrar varias funcionalidades, este se utilizará para el control, gestión y seguimiento de errores, para el control de versiones y como salva automática, para la Gestión de Contenido se eligió el CMS Drupal debido a la experiencia que se tiene en la utilización del mismo dentro de la fábrica, por sus facilidades de uso y también por poder utilizar como Gestor de Base de Datos además de MySQL, PostgreSQL, este último será el elegido para trabajar en la fábrica, como entorno de desarrollo integrado se va a utilizar el Zend Studio pues es un completo IDE para el lenguaje de programación PHP, como herramienta de modelado se utilizará el Visual Paradigm por las ventajas que brinda y por ser compatible o correr sobre plataforma libre, para el diseño de páginas web se va a utilizar el Quanta Plus (Quanta+) por ser una herramienta libre y como servidor se seleccionó el Servidor Apache por ser compatible con el sistema operativo GNU/Linux, este no se incluyó en la matriz de comparación debido a la familiarización del equipo de desarrollo con él. Los lenguajes de programación a utilizar son PHP, HTML, CSS y JavaScript los cuales fueron seleccionados por estar muy ligados a las herramientas seleccionadas.

1.9 Transferencia tecnológica y del conocimiento

"Las sociedades contemporáneas se enfrentan al reto de proyectarse y adaptarse a un proceso de cambio que viene avanzando muy rápidamente hacia la construcción de Sociedades del Conocimiento. Este proceso es dinamizado esencialmente por el desarrollo de nuevas tendencias en la generación difusión y utilización del conocimiento, y está demandando la revisión y adecuación de muchas de las empresas y organizaciones sociales y la creación de otras nuevas con capacidad para asumir y orientar el cambio. Una Sociedad del Conocimiento es una sociedad con capacidad para generar, apropiar, y utilizar el conocimiento para atender las necesidades de su desarrollo y así construir su propio futuro, convirtiendo

la creación y trasferencia del conocimiento en herramienta de la sociedad para su propio beneficio". (GEPSEA, 2009)

En Gepsea se habla del surgimiento de una Sociedad del conocimiento y de las ventajas que trae asociada este nuevo término, además este enfoque utiliza la transferencia del conocimiento como una herramienta para un beneficio mayor para la sociedad.

"Según la definición de las Naciones Unidas, la transferencia de tecnología constituye la transferencia de los conocimientos que son necesarios para la fabricación de un producto, la aplicación de un procedimiento o la prestación de un servicio.

La Regulación vigente en Cuba vinculada a los procesos de Transferencia de Tecnología (Resolución No 13/98), define la Transferencia de Tecnología, como el proceso de transmisión, absorción, adaptación, difusión y reproducción de la tecnología hacia una entidad distinta a donde se originó". (TÁPANES ROBAU, 2006)

En ambas definiciones se evidencia el concepto básico de la existencia de un ente emisor del conocimiento y un ente receptor del mismo.

La definición de las Naciones Unidas no solo ve la aplicación de la transferencia hacia la fabricación de un producto, eso va más allá hacia la prestación de un servicio. Sin embargo en la otra definición se evidencia la transferencia como un proceso con varias características que es aplicado a una entidad distinta adonde se originó.

Teniendo en cuenta lo anteriormente dicho este trabajo asume la transferencia tecnológica como un proceso que se lleva a cabo para transferir el conocimiento necesario para la fabricación de un producto, la aplicación de procesos o prestaciones de servicios; con el fin de potenciar la capacidad, aprovechamiento y conservación de los recursos con que se cuenta, así como la comunicación y retroalimentación de los responsables de realizar la transferencia. El éxito o fracaso del proceso de transferencia depende de la propia organización, la importancia que se le brinde y el impacto en sus trabajadores.

1.9.1 Transferencia de un Modelo de Factoría de Software

El enfoque de factoría de software necesita de técnicas y estrategias para transferir el conocimiento adquirido en las investigaciones. Para llevar un modelo de factoría a la práctica, se definen sus entidades en paquetes, estos representan la unión de varias actividades del modelo industrial, en función del

proceso que se desarrolla, las herramientas y el material de apoyo. Los paquetes son analizados y posteriormente replicados a un proceso en desarrollo.

A continuación se explica el proceso de transferencia de un modelo para replicar una factoría de software. El mismo debe usar técnicas de traslado de conocimiento y está dividido en 7 entidades distintas, lo anteriormente expuesto se puede apreciar en el Anexo 17.

"El proceso industrial de desarrollo del software (Entidad 1) origina los productos, los paquetes y contenedores que se replicarán (Entidad 2). Los productos deben ser guardados en una base de conocimientos que se divide en la biblioteca de componentes (Entidad 3) y la base de conocimientos sobre los productos (Entidad 4). En la biblioteca de componentes, se guardan los componentes del código e infraestructura. En la entidad 4 el conocimiento se guarda en el proceso, herramientas y material de apoyo. La base de conocimientos sobre productos propone que se defina una persona con plenos conocimientos sobre todos los procesos que se desarrollan en la factoría para que asesore la implantación de la misma. Se formaliza en este trabajo el proceso industrial, los productos, los paquetes, contenedores, y las técnicas de representación del conocimiento, simplemente quedan por definir las entidades inherentes al motor de replicación. El motor de replicación posee en su estructura las entidades 5, 6 y 7. La entidad 5 tiene como objetivo filtrar los productos, según las necesidades de la unidad a ser replicada. Hecho el filtrado, la entidad 6 configura los productos que se replicarán y la entidad 7 replica las entidades." (FABRI, 2004)

La transferencia del modelo de factoría de software seleccionado debe estar organizada en paquetes que contengan los procesos de la misma, agrupados de acuerdo a sus funcionalidades. Una vez definidos y analizados los paquetes que serán transferidos se configuran para su implantación en la factoría.

1.10 Experiencia Internacional

En la actualidad son muchas las empresas que han adoptado el enfoque de factoría de software, estas se ubican en varios países y han obtenido una serie de resultados que les han reportado grandes beneficios en todos sus sectores.

Aunque para muchos la palabra factoría es muy común ahora, fue en 1969 cuando se escuchó en Japón este término.

"La primera compañía en el mundo que nombró una organización del software como una factoría fue Hitachi Software Works en 1969, la segunda fue implantada entre los años 1975-1976 bajo el nombre de

Corporación de Desarrollo de Sistemas, por un líder americano en el campo del software personalizado". (AAEN, 1997)

"Durante los años setenta y ochenta en Japón se siguieron instalando fábricas de software: NEC en 1976, Toshiba en 1977, Fujitsu en 1979 y 1983, Hitachi en 1985, NTT en 1985 y Mitsubishi en 1987". (GARZÁS PARRA, 2007)

"Vector Software Factory es una compañía de desarrollo de componentes y construcción de soluciones software avanzadas, basadas en componentes reutilizables siempre en entornos de nuevas tecnologías. Este enfoque les permite adaptarse a las necesidades de sus clientes, ahorrando costes, en un menor tiempo y con más garantías de éxito. Vector SF obtiene la certificación CMMI nivel 3 y se sitúa entre las empresas de software con mayor calidad en procesos y productos a nivel mundial". (FACTORY, V. S., 2006)

"El Grupo Gesfor es una multinacional de capital 100% español, fundada en 1985 y dedicada a la consultoría y servicios en el sector de las TI, que desarrolla proyectos y presta servicios de valor añadido, tanto en ámbitos de soporte estratégico como operativo, a las principales compañías españolas en los distintos sectores económicos. Otra de las líneas de negocio con mayor proyección de crecimiento para la compañía es la factoría de software, servicio para el desarrollo de sistemas de información, en diferentes plataformas y con criterios de alta productividad y calidad, mediante técnicas y métodos de ingeniería de software. Esta ha alcanzando en 2006 una facturación de 62 millones de euros". (PASTOR, 2007)

"Actualmente, Ibermática es una de las principales compañías de servicios en TI en el mercado español, surgió en 1973. En el 2005 Ibermática creó el Instituto Ibermática de Innovación, un centro de investigación que dirige la aplicación de modelos de innovación tanto en la propia Ibermática como en otras organizaciones, a través de la implantación de soluciones y la creación de una metodología para medir la innovación. Este instituto basó su funcionamiento en este enfoque, aplicando de forma sistemática los conceptos más avanzados en materia de estandarización e industrialización del software, permitiéndole a Ibermática ganar mayor prestigio a nivel nacional e internacional. Todo este empeño valió para que la factoría de software alcanzara la certificación de nivel 3 de capacidad". (IBERMÁTICA, 2007)

"Grupo SMS Argentina se ha posicionado en el mercado internacional como empresa proveedora de soluciones TI, es especialista en el mercado de Seguros, Banca y Finanzas. Su objetivo es transformar las necesidades de sus clientes en resultados que mejoren sus procesos de negocio y que además estén alineados con su estrategia empresarial. Este cuenta con una factoría de software de profesionales

expertos en tecnología .NET, como así también en las metodologías y entornos propios de compañías líderes en el negocio de seguros.

El conocimiento conseguido a lo largo de los más de 8 años de experiencia con su modelo de factoría de software, permite que las empresas se beneficien de la mejora en la calidad del software, ya que el Grupo SMS reutiliza el knowhow obtenido, utiliza metodologías especializadas y aumenta la productividad al poder asumir y reaccionar ante diferentes volúmenes de trabajo y conseguir reducir errores y re-trabajos, además de una reducción de costes gracias al cumplimiento en los plazos de entrega y poder aplicar economías de escala". (GRUPO SMS, 2008)

"Tata Consultancy Services es la mayor empresa de servicios informáticos de la India. La compañía brinda servicios de TI, soluciones de negocios y outsourcing para clientes regionales y globales en 14 países, entre los que se incluyen México, países de Centro y Sudamérica, España y Portugal, representando en su conjunto un porcentaje de ingresos superior al 5%. (RAVEST, 2008)

TCS ha conseguido unos ingresos consolidados de 2.240 millones de dólares de EE.UU. en el período del 2004-2005. Varios de sus centros tienen certificado CMMI L5". (ANONIMO, 2008)

"Otra muestra es la consultora española de TI con mas 17 años de experiencia especializada en servicios de desarrollo de software bajo el modelo factoría. Con una plantilla global de más de 320 empleados, presta servicios desde España (Madrid, Barcelona y Albacete) y Francia (Paris, Marsella) a empresas y administraciones públicas nacionales de Europa y América del Sur. Es acreedora de la certificación ISO 9001:2000 desde el año 2005, un reconocimiento que garantiza la calidad de todas y cada una de las actividades y procesos de negocio que desarrolla, incluyendo aquellos de índole no-técnica". (ALHAMBRA-EIDOS, 2007)

"Capgemini es uno de los principales proveedores de servicios de Consultoría, Tecnología y Outsourcing del mundo, ofrece una forma única de trabajar con sus clientes que denomina Collaborative Business Experience. Presente en más de 30 países, alcanzó unos ingresos globales de 8.710 millones de euros en 2008, y tiene alrededor de 90.000 empleados. Las factorías se mueven en segmentos como SAP, J2EE y .net y en Business Intelligence". (CAPGEMINI, 2009)

"Grupo Accenture es una compañía española que tiene actividad en todo el ámbito estatal e internacional, presentan oficinas en Madrid, Barcelona, Málaga, Bilbao, Sevilla, Zaragoza y Valencia y cuentan con más de186.000 profesionales comprometidos en la creación de valor para el cliente. Las factorías cuentan con distintas especialidades: tecnología pura, sector consumo, banca, industria, telecos y farmacia. Su centro

de Málaga tiene la certificación CMMI-SW nivel 5, como el de Asturias de Capgemini. La compañía obtuvo una facturación de 23.390 millones de dólares durante el año fiscal finalizado el pasado 31 de agosto de 2008". (ACCENTURE, 2009)

Otros ejemplos es la factoría de software INDRA, T-SYSTEMS, Atos Origin, Matchmind.

Desde la implantación de la primera factoría de software hasta la fecha se puede observar que las empresas que han adoptado este enfoque han obtenido importantes beneficios, sobre todo en el cumplimiento de los costos y tiempos estimados, la calidad de los productos y la satisfacción de sus clientes. Esto se debe en gran medida a que las soluciones de software que proporcionan tienen en cuenta la tecnología más avanzada, adaptadas a los cambios y avances tecnológicos, siempre con una inmejorable relación calidad / precio. Además cuentan con equipos especializados que utilizan metodologías de trabajo y herramientas para la gestión de proyecto, y con repositorios de componentes reutilizables.

1.11 Conclusiones

En el capítulo se han identificado conceptos relacionados con portales, algunas de sus aplicaciones y los beneficios que aportan.

Se analizaron los conceptos principales sobre factorías de software, determinando que son las encargadas de industrializar el proceso de producción de software, transformándolo del ámbito artesanal al industrial.

Se estudiaron algunos modelos de factoría con el objetivo de elegir uno para implantar en el proyecto, escogiéndose el Modelo de Factoría de Software aplicando Inteligencia debido a que contiene los aspectos más significativos del resto de los modelos.

Se analizaron los elementos fundamentales para constituir el proceso como fueron: los modelos de procesos del software y de los mismos se seleccionaron dos para combinarlos y utilizarlos en el proyecto, los cuales son, los cuales son el Modelo de Construcción de Prototipos y el Modelo de Desarrollo Rápido de Aplicaciones.

También se tuvieron en cuenta las metodologías más importantes utilizadas en la universidad y se decidió utilizar un RUP aligerado por las facilidades que brinda, se especificaron las herramientas y lenguajes de programación que se utilizarán en la fábrica y de esta forma se dieron los primeros pasos para obtener el proceso a utilizar en la misma.

Llevar a la práctica un modelo de factoría de software implica también un proceso de transferencia tecnológica y de conocimientos que debe ser apoyado por todos los trabajadores de la organización que lo van a implementar, para poner de manifiesto lo anteriormente dicho se va a utilizar un modelo de replicación de un modelo de factoría de software, mediante el cual se realizará la estrategia de implantación.

También se hizo referencia a algunas experiencias por parte de empresas relacionadas con este enfoque donde se manifestó principalmente la reducción de los costos de producción, tiempo de desarrollo y la satisfacción de sus clientes por el aumento de la calidad del producto.

CAPÍTULO 2: ESTRATEGIA DE IMPLANTACIÓN

2.1 Introducción

En este capítulo se describe el proyecto de Fábrica de Portales de la Facultad 10 de la Universidad de las Ciencias Informáticas (UCI), los métodos, procedimientos y técnicas utilizadas en la investigación. Además se define la estrategia de implantación del Modelo de Factoría de Software aplicando Inteligencia y los riesgos que pueden afectar el funcionamiento correcto de la factoría de software.

2.2 Fábrica de Portales

La Universidad de las Ciencias Informáticas (UCI) es la primera universidad surgida en la Batalla de Ideas, esta tiene como misión:

- Formar profesionales altamente calificados y comprometidos con su patria.
- Producir software y servicios informáticos, a partir de la vinculación estudio-trabajo como modelo de formación.

La UCI está basada en un nuevo concepto de universidad productiva, el cual implica que la producción esté ligada a la docencia. En esta los profesores y estudiantes participan en proyectos que se realizan tanto para el mercado nacional, como internacional, los cuales tienen distintos campos tales como: la educación, la salud, el deporte, gobierno en línea, software libre, sitios y portales web, productos multimedia y otros.

La misma cuenta con una Infraestructura Productiva (IP) que es la encargada de organizar la producción de los proyectos. En esta se encuentran presente un grupo de direcciones vinculadas a diferentes líneas de producción y áreas para la gestión de la producción.

En cada facultad existen diferentes polos productivos, en la Facultad 10 uno de estos polos es el de Gestión de la Información y el Conocimiento, dentro del mismo se encuentra el proyecto de Fábrica de Portales el cual surge con el objetivo de apoyar la producción de portales en el país y el exterior. El mismo se encuentra conformado por 65 personas de ellos 13 profesores y el resto estudiantes de 3er año a 5to año.

En este proyecto se han detectado varias deficiencias en el modelo de producción que dificultan la creación de software, las mismas se exponen a continuación: la mayoría de los integrantes desconocen el

procedimiento para desarrollar un portal y no tienen clara la estructura del proyecto. Aunque la mayoría conocen su rol muchos no saben las actividades que deben desarrollar y los artefactos que generan. Existe desconocimiento de la metodología empleada por el proyecto y de los flujos de trabajo establecidos. Si bien la comunicación del equipo de desarrollo en su totalidad es buena, la planificación del trabajo tanto personal como a nivel de equipo no es la mejor, no se siguen estándares de calidad, afectándose la efectividad del equipo de desarrollo. Los componentes realizados no se documentan para facilitar su búsqueda y utilización en productos posteriores y en la gran mayoría de los casos no se lleva a cabo la reutilización de los mismos. El proyecto no cuenta con el material necesario para ofrecerles la capacitación a los estudiantes, esto trae consigo el insuficiente dominio de las herramientas de trabajo y a su vez la gestión de las dudas se convierte en un tema crítico e imprescindible. No se realiza la firma de un acta como constancia del levantamiento de requisitos y se desconocen las políticas de seguridad de la información establecidas.

Estas deficiencias afectan el proceso de desarrollo de los productos, por lo que una solución a estos problemas puede ser la implantación del enfoque de factoría de software al proyecto de Fábrica de Portales, ya que este busca industrializar el desarrollo de software definiendo los procesos de desarrollo con métodos y técnicas estandarizadas para la mejora continua, donde cada trabajador asume una tarea específica y centra sus esfuerzos en ella. Se debe contar con un grupo de herramientas estandarizadas para la construcción de software y la administración y control del proyecto, además debe existir una fuerte comunicación con el cliente. Es conveniente apoyarse en la reutilización de componentes, ya que permite disminuir el riesgo de cometer errores, el tiempo de desarrollo, el costo de producción y garantiza la calidad del producto final.

Aplicar los principios de factoría de software en este proyecto reporta importantes beneficios a corto y mediano plazo que marcan una diferencia sustancial en el desarrollo de la producción.

2.3 Métodos, procedimientos y técnicas utilizados

“El método científico de investigación es la forma de abordar la realidad, de estudiar la naturaleza, la sociedad y el pensamiento, con el propósito de descubrir su esencia y sus relaciones”. (ROLANDO ALFREDO HERNÁNDEZ LEÓN 2002).

Todo método científico de investigación necesita una estrategia para abordar las características y condiciones específicas del problema a resolver. La estrategia para esta investigación fue descriptiva y se

desarrolló relacionada al proyecto productivo de Fábrica de Portales, dirigida fundamentalmente a las personas involucradas en los procesos de desarrollo de portales.

Se utilizaron métodos teóricos para estudiar las características del objeto que no están observables directamente, elaborar una hipótesis y proponer mejoras. Estos métodos son: Histórico lógico y el Hipotético deductivo.

El método histórico lógico permitió investigar todo lo relacionado con el enfoque de factoría, en cuanto a conceptos, características, funciones, como se está aplicando en otras empresas y que resultados han obtenido. El hipotético deductivo se encuentra presente en el cumplimiento del objetivo y la hipótesis planteados.

Los métodos empíricos permiten describir y explicar las características del fenómeno en estudio. Estos métodos se aplicaron con el objetivo de recolectar los datos necesarios para identificar la problemática e identificar las causas de estas. Las encuestas y entrevistas realizadas al personal involucrado en el proceso de desarrollo de los portales fueron imprescindibles, permitieron conocer el funcionamiento del proyecto y su organización. A continuación se muestra la definición y operacionalización de las variables, y los indicadores que se van a medir mediante los métodos empíricos.

Variables independientes:

Proceso de producción actual.

Estrategia de implantación de un modelo de factoría de software.

Variable dependiente:

Proceso de producción obtenido.

Variables	Dimensión	Indicador	Subindicador	UM
Proceso de producción actual	Fábrica de Portales	Organización del proceso y las personas	Metodologías de desarrollo	---Sí ---No
			Definición del flujo de trabajo	---Sí ---No
			Conocimiento de Roles y Responsabilidades	---Sí ---No
			Conocimiento de las actividades y entregables por rol	---Sí ---No
		Gestión de proyecto	Gestión de tiempo	---Días
			Gestión de costo	---Peso
			Planificación del proyecto	---Sí ---a nivel de personas ---a nivel de equipo ---No
			Recursos asignados	---Cantidad
			Gestión de esfuerzo	---Días/Hombre
		Definición de las Bases Tecnológicas.	Conocimiento de las herramientas	---Sí ---No
			Repositorio de Componentes	---Sí ---No
		Comunicación con el Cliente.	Modelación de las funcionalidades del proyecto a desarrollar	---Sí ---No
			Aceptación de los entregables por parte del cliente	---Sí ---No
			Satisfacción del cliente	---Buena ---Regular ---Mala

Estrategia de implantación de un modelo de factoría de software	Fábrica de Portales	Complejidad del proceso de desarrollo de software.	Cantidad de desarrolladores	Número de personas
			Magnitud del proyecto	---Grande ---Mediano ---Pequeño
		Los paquetes que la conforman.	Proceso de desarrollo	Objetivos. Características. Procesos que se ejecutan. Responsabilidad.
			Personas	Objetivos. Características. Procesos que se ejecutan. Responsabilidad.
			Gestión de proyecto	Objetivos. Características. Procesos que se ejecutan. Responsabilidad.
			Bases tecnológicas	Objetivos. Características. Procesos que se ejecutan. Responsabilidad.
			Repositorio de componentes	Objetivos. Características. Procesos que se ejecutan. Responsabilidad.
Proceso de producción obtenido.	Fábrica de Portales	Organización del proceso y las personas	Metodologías de desarrollo	---Sí ---No
			Definición del flujo de trabajo	---Sí ---No
			Conocimiento de Roles y	---Sí

			Responsabilidades	---No
			Conocimiento de las actividades y entregables por rol	---Sí ---No
		Gestión de proyecto	Gestión del tiempo	---Días
			Gestión del costo	---Peso
			Gestión de esfuerzo	---Días/Hombre
			Recursos asignados	---Cantidad
			Planificación del proyecto	---Sí ---a nivel de personas ---a nivel de equipo ---No
		Definición de las Bases Tecnológicas.	Conocimiento de las herramientas	---Sí ---No
			Repositorio de Componentes	---Sí ---No
		Comunicación con el Cliente.	Modelación de las funcionalidades del proyecto a desarrollar	---Sí ---No
			Aceptación de los entregables por parte del cliente	---Sí ---No
			Satisfacción del cliente	---Buena ---Regular ---Mala

Tabla 1.2 Definición y operacionalización de las variables

2.3.1 Encuestas a los integrantes del proyecto

Se tomó como **población** los integrantes de la fábrica y como **unidad de estudio** los estudiantes.

Para llevar a cabo la selección de la muestra se decidió aplicar la técnica de muestreo no probabilística a una población determinada, ya que ésta técnica permite ahorrar tiempo y recursos. Dentro de esta se seleccionó el muestreo intencional para poder obtener la mayor representatividad e información posible, de acuerdo con los intereses de la investigación.

Las encuestas fueron aplicadas a estudiantes de diferentes proyectos de la Fábrica de Portales, que desempeñan distintos roles, con el objetivo de valorar la situación actual. En estas se evaluaron

indicadores que son importantes para el desarrollo de los portales, como son: Organización del proceso y las personas y Definición de las Bases Tecnológicas.
De un total de 52 estudiantes del proyecto se aplicó encuestas a 26 siendo la muestra un 50 % de toda la población.
Se hicieron preguntas cerradas y semicerradas mayormente para tener un poco más de profundidad en los criterios de los encuestados. (Ver Anexo 18)

2.3.2 Entrevista al Líder del proyecto

La población a estudiar fueron los directivos de la fábrica y la unidad de estudio el líder del proyecto. Se realizó una entrevista formal al Líder del proyecto como se muestra en el Anexo 19 donde se evaluaron los siguientes indicadores: Gestión de proyecto y Comunicación con el Cliente.

2.3.3 Entrevistas a los clientes

La población a estudiar fueron los clientes que actualmente se encuentran colaborando con la fábrica y la unidad de estudio los clientes del proyecto piloto. La selección se realizó con la técnica de muestreo no probabilística, muestreo intencional. Se seleccionaron 6 de estas personas evaluándose el siguiente indicador: Comunicación con el Cliente. (Ver Anexo 20)

2.4 Estrategia de Implantación

El objetivo de la estrategia es obtener un área de producción de software organizada con un modelo de factoría de software cuyo procesos sean definidos, repetibles y mejorables continuamente, capaz de elevar la producción de software en el proyecto de Fábrica de Portales.
Definir una estrategia de implantación del Modelo de Factoría de Software aplicando Inteligencia para el proyecto Fábrica de Portales implica poner al mismo en condiciones de realizar su trabajo de manera eficaz y eficiente. Para implantar la factoría de software se necesita realizar una serie de actividades, tener un proceso de producción estandarizado que brinde como resultado un producto y sobre todo debe existir motivación por parte de los integrantes.
La estrategia para hacer la transferencia del modelo de Factoría de Software aplicando Inteligencia es la siguiente:

- Realizar un proceso de investigación para recopilar la información que permita implantar un modelo de factoría a un proyecto.
- Definir el proceso de transferencia para el proyecto.
- Gestionar todos los recursos necesarios para poner en marcha la estrategia de transferencia.
- Implantar el proceso de transferencia definido al proyecto.
- Hacer un seguimiento del funcionamiento del proceso de transferencia.
- Emitir un reporte de los resultados obtenidos.

El primer paso se realiza para conocer las características del proyecto, y buscar alternativas para solucionar las deficiencias que presente. Se analizan los distintos tipos de modelos que existen y la experiencia de las empresas que respaldan sus beneficios.

El próximo paso es definir el proceso de transferencia, el mismo es orientado a procesos, y éstos son definidos atendiendo a las características y objetivos principales de las factorías de software. Estos procesos se configuran en paquetes, lo que facilita la reutilización para aquellas empresas que deseen aplicar este enfoque.

El proceso depende de la clasificación de las factorías (Ver Anexo 21). Según este modelo, se define un proceso completo al pasar por los flujos de trabajo de Modelo del Negocio, Captura de requisitos, Análisis y Diseño, Implementación, Prueba y Despliegue y en base a los que se lleven a cabo en una factoría se define su alcance. Como en el proyecto Fábrica de Portales se comienza a partir de la entrega de los requisitos, la factoría se clasifica como de modelado, diseño e implementación.

Antes de trazar la estrategia de implantación primeramente se deben definir todos los procesos que existen en la factoría y agruparlos en paquetes teniendo en cuenta criterios de funcionalidad, así se replicarán paquetes que contengan procesos con funcionalidades similares. Los paquetes son definidos de acuerdo al orden de implantación. El modelo de replicación propuesto en el capitulo anterior se puede usar para transferir todos los paquetes de la factoría (Ver epígrafe 1.9.1).

Paquetes	Procesos	Responsables	Participantes

Proceso de desarrollo	-Requisitos. -Arquitectura y Diseño. -Implementación y prueba. -Despliegue.	-Gestor de la factoría. -Gestor de desarrollo. -Director de calidad.	-Líder de proyecto. -Analistas. -Cliente. - Arquitecto. -Diseñador de Base de datos. -Diseñador de componentes. - Implementadores. -Implementador de Base de datos. -Administrador del Repositorio. -Probador -Documentadores -Administrador de despliegue.
Personas	-Definir la estructura organizativa, formar grupos de trabajos, asignar roles y responsabilidades. -Impartir cursos de capacitación. - Instrumentos PSP.	- Gestor de la factoría.	- Gestor de desarrollo. -Planificador.
Gestión de Proyecto	- Definir el proyecto y los objetivos del mismo. -Establecer una planificación del proyecto. -Establecer la estructura organizativa de los equipos. - Definir alcance, tiempo y costo del proyecto. - Prever posibles riesgos y la forma de mitigarlos. - Gestionar la calidad del proyecto. -Coordinar y supervisar las distintas tareas y actividades de las que consta el proyecto.	- Gestor de la factoría.	-Planificador.
Bases tecnológicas	-Conocer las herramientas a utilizar. -Instalar en las máquinas según las necesidades de los roles -Definir mecanismos de seguridad de la	- Especialista en tecnología.	-Administrador de repositorio. -Director de calidad. -Investigador.

	información. -Definir las listas de chequeo. - Capacitación.		
Repositorio de componentes	-Configurar el repositorio de componentes. - Definir la estructura de almacenamiento de la información. -Hacer catálogo de componentes reutilizables de cada proyecto con su descripción, objetivo, características tecnológicas y ubicación.	-Administrador de repositorio.	- Documentador. - Gerente de la factoría.

Tabla 1.3 Paquetes (entidades) correspondientes al Modelo de Factoría de Software aplicando Inteligencia.

2.4.1 Proceso de desarrollo

El proceso en el desarrollo de software define las actividades necesarias para entender, comprender y responder ante los requisitos del usuario, convirtiéndolos en un producto final. Para garantizar una buena organización del proceso de desarrollo se van a tener en cuenta los modelos de procesos del software definidos anteriormente: el Modelo de Construcción de Prototipos y el Modelo de Desarrollo Rápido de Aplicaciones, que serán combinados con el objetivo de aminorar sus desventajas y obtener un proceso bien definido que se ajuste a las características de los portales.

Los flujos de apoyo no se detallan en este epígrafe pues estos se incluyen dentro de otros paquetes. La gestión de configuración se tendrá en cuenta en el repositorio, la gestión de proyecto dentro del paquete gestión de proyecto y ambiente dentro de bases tecnológicas.

A continuación se describen las principales actividades a desarrollar en cada uno de los flujos y los artefactos generados en las mismas y los roles que participan en su desarrollo. Se va a tener en cuenta la participación del documentador a medida que el proyecto avanza, es decir está presente en todos los flujos de trabajo.

2.4.1.1 Requisitos

En este flujo el cliente solicita la creación de un portal Web. El líder del proyecto junto con el analista se entrevista con los clientes para hacer un levantamiento de los requisitos tanto funcionales como no

funcionales. A partir de estos se hace el modelo del sistema, el cual permite llegar a un acuerdo entre los desarrolladores y lo que el cliente quiere. Las actividades más importantes a realizar en este flujo son:

- **Diseñar la arquitectura de la información:** Se define y organiza el contenido que se va a publicar en el portal Web.
- **Captura de requisitos:** El analista extrae, de cualquier fuente de información proporcionada por el cliente, las necesidades de éste. Todos los elementos de información deben ser almacenados e identificados. Pueden provenir de conversaciones con el cliente o estar recogidos en ficheros de datos, gráficos, normativas legales, etc.
- **Especificación de Requisitos:** Se deben definir los requisitos y negociarlos con el cliente, para obtener una descripción de requisitos lo más certera posible, evitando al máximo los problemas que puede traer consigo una mala definición de los mismos.
- **Validación de Requisitos:** Una vez que han sido definidos los requisitos, se debe realizar un proceso de validación de los mismos, para garantizar que la definición planteada de cada uno de ellos es la más adecuada.
- **Encontrar actores y casos de uso:** Se debe esbozar quienes interactuaran con el sistema y que funcionalidades realizan.

  En el Anexo 22 se puede observar una tabla descriptiva de este flujo.

2.4.1.2 Arquitectura y Diseño

Durante este flujo se realiza el diseño para ser utilizado en la implementación. El arquitecto es el responsable de definir la arquitectura y decidir que componentes reutilizables pueden ser utilizados. El diseñador de componentes es el encargado de realizar el diseño de clases y subsistemas, y el de base datos de realizar la confección de la misma.

Las actividades más representativas de este flujo son:

- **Definir la arquitectura:** Se encuentran los patrones de la arquitectura, los principales mecanismos y convenciones de modelado para el sistema.
- **Diseño de componentes:** Se encarga de definir todos los componentes que van a ser implementados.

  Diseñar la Base de datos: Se usa para detallar el comportamiento que debe tener la base de datos.

En el Anexo 23 se puede observar una tabla descriptiva de este flujo.

2.4.1.3 Implementación y prueba

Este es el flujo que más trabajo tiene, pues es donde precisamente se construye el producto. Si es necesario incluir componentes del repositorio, los implementadores se lo solicitan al Administrador del repositorio. El líder de proyecto chequea que el diseño esté elaborado completamente con el nivel de detalle especificado y planifica la implementación, divide el diseño y lo reparte entre los diferentes implementadores. Luego el arquitecto junto con los implementadores es responsable de la unión de las partes para conformar el producto final e inmediatamente después de la integración se envía para que se le realicen las pruebas. Las pruebas están bajo la dirección del Director de calidad de la factoría.
Las actividades más representativas de este flujo son:

- **Estructurar el modelo de implementación:** Se establecen las relaciones entre los subsistemas y se obtiene el modelo de implementación.
- **Desarrollar plan de integración:** Se establece cómo se van a construir los subsistemas definidos.
- **Implementar componentes:** Aquí el implementador escribe código fuente, reutiliza código, compila, realiza prueba unitaria, implementa los elementos del modelo de diseño y si encuentra defectos en el diseño, regresa al diseño y los corrige.
- **Implementar la Base de datos:** Queda definida la base de datos con todas sus funcionalidades implementadas.
- **Integrar subsistemas:** Se integran los subsistemas después de ser construidos.
- **Integrar sistema:** Se integran todos los componentes elaborados.
- **Planificar evaluaciones:** Consistirá en planificar que es lo que hay que probar, aquí es donde se deciden los casos de prueba a efectuar, se realiza su diseño y se monitorea y evalúa el progreso de las pruebas.
- **Ejecutar evaluaciones:** Se evalúan los resultados de las pruebas comparando los resultados con los objetivos esbozados en el plan de prueba y se registran las no conformidades.

En el Anexo 24 se puede observar una tabla descriptiva de este flujo.

2.4.1.4 Despliegue

En este flujo se prueba el producto en su entorno de ejecución final, en caso de detectar inconformidades con el mismo se retorna para su corrección. En caso contrario, el cliente firma los documentos convenientes donde expresa su conformidad con el producto y se cierra el contrato.
La actividad más representativa de este flujo es:

- **Desarrollar el plan de despliegue:** Describe el conjunto de tareas necesarias para instalar y probar el producto desarrollado, de manera tal que pueda ser transferido con eficacia al cliente.

En el Anexo 25 se puede observar una tabla descriptiva de este flujo.

2.4.2 Personas

Las personas involucradas en los procesos de una factoría de software forman un factor importante para el éxito de un proyecto por lo que su organización es fundamental. La factoría cuenta con una estructura organizativa donde cada persona implicada ocupa un rol determinado dentro de la misma en dependencia de sus conocimientos, habilidades y valores, la misma se divide en Grupo de desarrollo y Gestores de la factoría.

2.4.2.1 Grupo de desarrollo

La conformación del equipo de desarrollo es un proceso fundamental para el buen funcionamiento de los proyectos que se inician, por esta razón es imprescindible que se tenga en cuenta la cantidad de personas que van a participar en el mismo y las habilidades de cada uno. Para llevar esta acción a cabo se debe medir la carga de trabajo inicial y la dimensión del producto a desarrollar.
La mejor estructura de equipo depende de las características de la factoría, el número de personas que compondrá el equipo, la preparación que posean sus integrantes y la dificultad de las tareas asignadas al mismo.
“Pressman propone tres organigramas para equipos de desarrollo de software: el Descentralizado democrático (DD), el Descentralizado controlado (DC) y el Centralizado controlado (CC)”. (PRESSMAN, 2002) (Ver Anexo 26)
Teniendo en cuenta que la factoría de software será aplicada al proyecto Fábrica de Portales se propone utilizar el organigrama DC, ya que permite dividir el trabajo entre pequeños equipos, conformando subgrupos, lo que trae como consecuencia que la comunicación sea buena entre todos los miembros del

equipo de desarrollo. También ofrece mejores resultados para equipos que llevan gran cantidad de tiempo trabajando juntos.

En la factoría se van a encontrar los equipos de trabajos divididos por especialidad, es decir un grupo de analistas, otro de implementadores, etc. cada uno de estos equipos va a contar con un jefe que es el encargado de apoyar y controlar el trabajo de sus empleados. Cuando se va a conformar un equipo de desarrollo se seleccionan los integrantes que sean necesarios de cada especialidad, el mismo debe estar integrado por un grupo de 5 a 10 personas porque los productos son de corta duración y no tienen tanta complejidad.

Con esta cantidad de desarrolladores por equipo se logra una comunicación más fluida y se obtienen productos con la calidad requerida en el tiempo estimado. Se gestiona y recolecta toda la información para el desarrollo de las habilidades de los integrantes del proyecto y se realiza un compromiso personal y por colectivos de trabajo a la protección a la no divulgación de la información y hechos a los que tendrán acceso por razón del desarrollo de los proyectos.

Para que una factoría funcione correctamente, además de la organización, se necesitan estilos de dirección. Un estilo de dirección es la forma en que la dirección interactúa con las personas bajo su mando. Existen tres estilos básicos que figuran en la bibliografía sobre gestión de proyectos: el autocrático, el permisivo y el democrático.

En la factoría existen normas de funcionamiento, el estilo de dirección adoptado para el control de las mismas se propone que sea autocrático, pues estas son las que guían a los participantes de la factoría a cumplir los objetivos estratégicos planteados. En esta cada participante tiene un rol definido y tareas que cumplir y para llevar esto a cabo se debe adoptar un estilo democrático, pues este permite generar ideas e intercambiar opiniones y la retroalimentación entre los miembros del equipo.

2.4.2.2 Gestores de la factoría

Para que en una factoría todo funcione exitosamente debe existir un equipo de dirección encargado de regir todas las acciones que se realicen en la misma. El organigrama referente a la estructura organizativa de la misma se puede apreciar en el Anexo 27 y en el Anexo 28 se muestran los datos principales de cada directivo.

Podrán existir tantos equipos de desarrollo como demanda de trabajo haya por parte del cliente, cada equipo tendría un investigador en caso de ser necesario, y un jefe del grupo de desarrollo que respondería

delante del gestor de desarrollo. A continuación se pueden apreciar los roles definidos para llevar a cabo la dirección de la factoría, se mantuvieron algunos de los roles propuestos por el Modelo de Factoría aplicando Inteligencia y los restantes fueron tomados de TSP y RUP.

Gestor de la factoría: Es el encargado de estimar recursos, plazos, costos y riesgos para la realización de un determinado producto software, controlar el desarrollo del mismo, verificar y tener noción de la fase en que se encuentra. Es el responsable de reunirse y contactar con el cliente para las entregas de cada versión realizada. Debe organizar el trabajo de forma tal que las personas bajo su mando cumplan su rol adecuadamente.

Director de calidad: Es el capacitado para dirigir la gestión de la calidad de la factoría y tiene la tarea de vigilar para que los productos se realicen con la calidad requerida, coordinando la realización de pruebas y revisiones a los mismos, contando con el apoyo de varias personas y basándose en las listas de chequeo.

Gestor de desarrollo: Es el que se encarga de dirigir el desarrollo de productos software dentro de la factoría. Tiene bajo su mando a uno o varios equipos de desarrollo, es el intermediario entre estos y el Gestor de la factoría, constituye el canal a través del cual fluye la información desde los directivos hacia el equipo de desarrollo. Tiene la tarea de velar por que los productos se realicen a tiempo y según las especificaciones establecidas por el cliente y verificar directamente el trabajo de los líderes de proyecto.

Administrador de repositorio: Es el responsable de gestionar el repositorio de componentes, establecer las políticas de seguridad de la información y llevar a cabo la gestión de configuración.

Investigador: Es el facultado para realizar cualquier investigación, búsqueda de información en un tema determinado solicitado por cualquier grupo de desarrollo. Este cargo lo pueden realizar varias personas en dependencia de la carga de trabajo que existe en el ámbito investigativo.

Especialista en tecnología: Es el encargado establecer las bases tecnológicas de la factoría y garantizar las instalaciones de las herramientas en cada estación de trabajo, manteniéndolas actualizadas en dependencia de la última versión.

2.5.2.3 Instrumentos de PSP

El trabajo organizativo de los individuos de una factoría depende en gran medida de los instrumentos de PSP, estos son fundamentales cuando el personal de la misma es inexperto en el desarrollo de software. A continuación se proponen algunos elementos que son importantes en la planificación personal,

principalmente para el aprendizaje de la estimación del tiempo y el control de defectos. Estos instrumentos son:

Cuaderno del ingeniero: permite registrar tiempos, guardar cálculos, tomar notas de las fases del desarrollo, y registrar además cualquier otro tipo de información que afecte el horario laboral en la factoría. (Ver Anexo 29)

Cuaderno de registro de tiempo: sirve para registrar diariamente las actividades realizadas en cada instante del día. Es útil para contabilizar y describir las interrupciones ocurridas durante la jornada laboral, permitiendo mejorar la calidad y eficiencia de su trabajo. Les permite a los líderes observar los comportamientos de los integrantes de la factoría y realizar análisis al respecto. (Ver Anexo 30)

Resumen semanal de actividades: las actividades se agrupan por días, obteniéndose los totales de tiempo relacionados con los trabajos efectuados a diario y de los trabajos efectuados en la semana en general. (Ver Anexo 31)

Cuaderno de registro de defectos: ayuda a registrar datos de defectos, en dependencia de su tipo, principalmente los que tienen que ver con la calidad de componentes implementados. Este cuaderno es importante para saber los errores que más se cometen y en qué fase de desarrollo se introducen, permitiendo en un futuro prevenirlos. (Ver Anexo 32)

2.5.3 Repositorio

En el enfoque de factoría de software la reutilización juega un papel excepcional pues ayuda a prolongar la vida útil de los materiales almacenados correspondientes al desarrollo de proyectos anteriores, garantizando la reducción del tiempo de trabajo y la aparición de errores, el aumento de la productividad y la calidad, así como la disminución del riesgo. En el caso de la Fábrica de Portales es de gran utilidad debido a la gran similitud que presentan los portales en cuanto a la arquitectura de la información y en cuanto a su estructura, independientemente del objetivo con el cual sean desarrollados.

El repositorio de componentes reutilizables debe contener dos grupos fundamentales:

- **Componentes de código:** pueden ser clases, procedimientos o funciones, módulos, subsistemas o una aplicación. Ejemplo: los módulos de Drupal que se utilizan en el proyecto.
- **Activos del proceso:** pueden ser patrones de diseño, algoritmos, un esquema de una base de datos, manuales y documentación o un modelo. Ejemplo: documentación sobre los proyectos que se desarrollan y documentación sobre el CMS Drupal.

Para permitir una mayor organización del trabajo y facilitar la búsqueda de cada uno de los componentes, quedará confeccionado un catálogo con los datos principales de los mismos: su nombre, su descripción, su tipo, fecha de creación, y la ubicación. Cada vez que se inserte un nuevo componente al repositorio se debe actualizar la planilla mostrada en el Anexo 33 y llenar los datos que la misma solicita. Este debe ser confeccionado por el documentador.

La información de la fábrica almacenada en el repositorio, debe estar organizada por portales Web, en la carpeta correspondiente a cada uno se deben guardar los artefactos organizados en dependencia del flujo de trabajo en que se realizaron. Ejemplo, todos los artefactos generados durante el flujo de requisitos, se guardarán en una carpeta con el nombre del flujo.

Se debe habilitar además, una carpeta con todos los módulos de Drupal utilizados en el proyecto y de cada módulo se debe tener una descripción que facilite su posterior utilización. Asimismo debe quedar almacenada el resto de la información generada durante el proyecto.

La misión del administrador de repositorio es mantener actualizado el mismo para facilitar la reutilización eficaz de sus componentes. También es el encargado de realizar la gestión de configuración y de elaborar el plan. Este debe configurar el repositorio para insertar, modificar o eliminar componentes, limitando el acceso a los desarrolladores en dependencia del rol que desempeña en la factoría. Debe realizar salvas diarias de su repositorio en el repositorio central a nivel de facultad para evitar problemas de conexión o pérdidas.

2.5.4 Bases Tecnológicas

La entidad Bases tecnológicas dentro de una factoría de software comprende el contexto de las herramientas, las técnicas y mecanismos para construir, soportar y gestionar el proceso de desarrollo. En la organización de una factoría de software debe conocerse fundamentalmente la utilidad que presenta cada herramienta, las normas disciplinarias y técnicas especificadas por los directivos que se deben cumplir. Cada vez que a la factoría se incorpore un nuevo integrante, deberá recibir una capacitación sobre las herramientas que se utilizan en la misma. El Especialista en tecnología es el encargado de decidir de cada herramienta la versión a utilizar y velar porque cada rol tenga en su puesto de trabajo las herramientas con las cuales tiene que trabajar.

2.5.4.1 Tecnologías y herramientas

En la factoría deben estar definidas todas las tecnologías y herramientas que se van a utilizar para el desarrollo de los proyectos. En la Fábrica de Portales las que se van a utilizar fueron definidas en el Capítulo 1, las mismas son:

Gestión de proyecto: Gforge.

Gestión de contenido: Drupal

Modelado: Visual Paradigm.

Diseño de páginas Web: Quanta Plus.

Sistema de Gestión de base de datos: PostgreSQL.

Entorno de desarrollo integrado: Zend Studio.

Servidor Web: Apache.

2.5.4.2 Técnicas y mecanismos

En una factoría es inevitable implantar técnicas y mecanismos que posibiliten la seguridad de la información ya que este es un factor clave para el desempeño de la misma. Para mantener la información es necesario establecer políticas de seguridad de la información (PSI). Estás políticas describen accesos como: la información básica, software que se deben instalar, garantía del producto en uso, restricciones sobre los dispositivos de acceso a la estación de trabajo y del personal ajeno a la factoría. Incluyendo medidas que se deben tomar para traspasar información a través de Internet. En el Anexo 34 se muestra una lista con las mismas.

Todos los productos que se elaboren en la factoría tienen que tener una alta calidad por esta razón es necesario tener listas de chequeo que permitan controlar la calidad de los mismos. En todo el desarrollo de software se debe evaluar el cumplimiento de estas listas con el objetivo de hacer el menor número de correcciones posibles al final. (Ver Anexo 35).

2.5.5 Gestión de Proyecto

“La gestión de proyectos implica la planificación, supervisión y control del personal, del proceso y de los eventos que ocurren mientras evoluciona el software desde la fase preliminar a la implementación operacional”. (PRESSMAN 2002)

Los objetivos principales de la gestión de proyectos son:

- Definir el proyecto y los objetivos del mismo
- Establecer una planificación del proyecto
- Establecer la estructura organizativa de los equipos
- Definir alcance, tiempo y costo del proyecto
- Prever posibles riesgos y la forma de mitigarlos
- Gestionar la calidad del proyecto
- Coordinar y supervisar las distintas tareas y actividades de las que consta el proyecto.

El encargado de realizar todas las actividades comprendidas dentro de la gestión de proyectos es el líder de proyecto junto con el planificador, utilizando las técnicas y herramientas que se explicarán a continuación.

2.5.5.1 Proceso de gestión

El Modelo aplicando Inteligencia propone para la entidad Gestión de proyecto 6 procesos que describen, organizan y efectúan el trabajo del proyecto. Estos procesos se relacionan entre sí realizando distintas actividades, las mismas generan artefactos que sirven como entrada para el siguiente proceso.

A continuación se muestran los procesos por los que debe transitar una correcta gestión de proyecto y en el Anexo 36 se explica las relaciones entre ellos:

- **Proceso de Iniciación:** Comienza con la asignación del proyecto a la fábrica de software. El mismo define los objetivos del proyecto, los requerimientos y le ofrece al cliente la posibilidad de especificar y mejorar los requerimientos que solicita. También determina la factibilidad del proyecto y el alcance del mismo.
- **Proceso de Planificación:** Se encarga de detallar el desarrollo del proyecto, de planificar el costo, tiempo y la cantidad de personal para terminar el mismo. Además coordina la cantidad de integrantes que tendrá el proyecto, establece los roles y asigna jefes para los equipos de trabajo, gestiona los recursos necesarios para el desarrollo de los productos y se encarga de gestionar los riesgos. A medida que vaya surgiendo nueva información del proyecto este proceso se encarga de actualizar todo.
- **Proceso de Ejecución:** Se elaboran los productos según los flujos de trabajo definidos en el paquete proceso de desarrollo.

- **Proceso de Control:** Supervisa y vigila todas las acciones realizadas, con el fin de asegurar que se cumple con las especificaciones y objetivos planificados. Se encarga de observar la ejecución del proyecto, de forma que se puedan identificar los posibles problemas oportunamente y adoptar las acciones correctivas necesarias.
- **Proceso de Cierre:** Una vez concluido el proyecto, este proceso se encarga de probar el producto antes de ser entregado y verificar que se cumplieron todos los requisitos iniciales, luego se le entrega al cliente el producto final.
- **Proceso de Sostenibilidad:** Consiste en chequear el funcionamiento del producto para darle soporte y mantenimiento. Controla los distintos cambios que se pueden producir a lo largo de la ejecución del proyecto y aplica medidas para corregir los errores detectados durante la ejecución. El mismo permite vigilar de manera sistemática si el producto funciona con éxito para evitar insatisfacciones por parte del cliente.

2.5.5.2 Gestión del alcance

La gestión del alcance del proyecto incluye los procesos requeridos para definir y controlar el alcance del proyecto: lo que está y lo que no está incluido en el mismo. Se ocupa de que en el proyecto se realice el trabajo necesario para cumplir los objetivos planteados al inicio del proyecto.

Los procesos que se deben realizar son las siguientes:

- **Planificación del Alcance:** Crear un plan de gestión del alcance del proyecto que documente como será definido, verificado y controlado el alcance y cómo se creará y definirá la Estructura Detallada del Trabajo (EDT).
- **Definición del Alcance:** Desarrollar un enunciado del alcance del proyecto detallado.
- **Crear la Estructura de Desglose del Trabajo**: Subdividir los principales entregables del proyecto en tareas más pequeñas y componentes más manejables.
- **Verificación del Alcance:** Formalizar la aceptación del alcance del proyecto.
- **Control del Alcance:** Controlar los cambios en el alcance del proyecto.

En el Anexo 37 se pueden encontrar argumentados estos procesos.

2.5.5.3 Gestión del tiempo

La gestión del tiempo incluye los procesos necesarios para lograr la construcción del proyecto en el tiempo planificado. Se recomienda hacer un Diagrama de Gantt para hacer una adecuada planificación de las tareas. Se propone trabajar con Gforge para planificar y tener un control de las actividades en el proyecto.

Los procesos que hay que realizar son los siguientes:

- **Definición de las Actividades:** Identifica las actividades específicas del cronograma que deben ser realizadas para producir los diferentes productos entregables del proyecto.
- **Establecimiento de la Secuencia de las Actividades:** Identifica y documenta las dependencias entre las actividades del cronograma.
- **Estimación de Recursos de las Actividades:** Estima el tipo y la cantidad de recursos necesarios para realizar cada actividad.
- **Estimación de la Duración de las Actividades:** Estima el período laboral de las actividades que será necesario para desarrollar el proyecto.
- **Desarrollo del Cronograma:** Desarrollar un cronograma que analice las secuencias de las actividades, la duración de las actividades, los requisitos de recursos y las restricciones para crear el proyecto.
- **Control del Cronograma:** Controla todos los cambios que se realicen en el cronograma del proyecto.

En el Anexo 38 se pueden encontrar argumentados estos procesos.

2.5.5.3 Gestión del costo

La gestión de costos del proyecto incluye los procesos necesarios para asegurar que el proyecto se complete dentro del presupuesto aprobado. Para llevar a cabo el mismo se desarrollan los siguientes procesos:

- **Estimación de Costes**: Desarrolla una aproximación de los costes de los recursos necesarios para completar las actividades del proyecto.
- **Preparación del Presupuesto de Costes:** Suma los costes estimados de actividades individuales o paquetes de trabajo a fin de establecer una línea base de coste.

- **Control de Costes**: Ejerce influencia sobre los factores que crean variaciones del coste y controla los cambios en el presupuesto del proyecto.

En el Anexo 39 se pueden encontrar argumentados estos procesos.

Para realizar la Gestión del costo se debe hacer una planificación de los recursos necesarios para llevar a cabo el proyecto. Concluido esto se debe estimar el costo aproximado de cada recurso, sumando el costo final de los mismos al presupuesto del proyecto. Esta estimación se puede hacer mediante el método COCOMO II, ya que permite estimar el costo de todos los recursos necesarios en el proyecto.

2.5.5.4 Gestión de la calidad

La gestión de la calidad del proyecto incluye los procesos y las actividades de la organización ejecutante que determinan las políticas, los objetivos y las responsabilidades relativos a la calidad, de modo que el proyecto satisfaga las necesidades que motivaron su creación. Implementa el sistema de gestión de calidad a través de políticas y procedimientos, con actividades continuas de mejora de procesos realizadas a lo largo de todo el proyecto, según corresponda.

Los procesos que aquí se definen son los siguientes:

- **Planificación de la Calidad:** Se identifica los estándares de calidad relevantes para el proyecto y se especifica cómo implementarlos.
- **Realizar Aseguramiento de la Calidad:** En esta fase todo el equipo de trabajo de calidad, está a cargo de comprobar que se cumple con las actividades planificadas y sistemáticas relativas a la calidad, para asegurar que el proyecto utilice todos los procesos necesarios para cumplir con los requisitos.
- **Control de Calidad:** Una vez definida la política de seguridad a seguir el equipo de desarrollo es el encargado de efectuar pruebas de calidad que verifiquen que el producto está listo.

En el Anexo 40 se pueden encontrar argumentados estos procesos.

Se determina utilizar el estándar ISO 9001 ya que este modelo comprueba la calidad en todas las etapas de desarrollo de un producto de software. El Director de calidad es el responsable de la implantación de la norma ISO 9001 y CMMI para la mejora de procesos de la organización.

2.5.5.5Gestión de riesgos

La gestión de los riesgos del proyecto incluye los procesos relacionados con la planificación de, la identificación y el análisis de los riesgos, las respuestas a los riesgos, y el seguimiento y control de riesgos de un proyecto. Los objetivos de los mismos son aumentar la probabilidad de eventos positivos, y disminuir la probabilidad de eventos adversos para los objetivos del proyecto.

Los procesos de la gestión de riesgos son:

- **Planificación de la Gestión de Riesgos:** Decide cómo enfocar, planificar y ejecutar las actividades de gestión de riesgos para un proyecto.
- **Identificación de Riesgos:** Determina qué riesgos pueden afectar al proyecto y documenta sus características.
- **Análisis Cualitativo de Riesgos:** Prioriza los riesgos para otros análisis o acciones posteriores, evaluando y combinando su probabilidad de ocurrencia y su impacto.
- **Análisis Cuantitativo de Riesgos:** Analiza numéricamente el efecto de los riesgos identificados en los objetivos generales del proyecto.
- **Planificación de la Respuesta a los Riesgos:** Desarrolla opciones y acciones para mejorar las oportunidades y reducir las amenazas a los objetivos del proyecto.
- **Seguimiento y Control de Riesgos:** Realiza el seguimiento de los riesgos identificados, supervisa los riesgos residuales, identifica nuevos riesgos, ejecuta planes de respuesta a los riesgos y evalúa su efectividad durante todo el ciclo de vida del proyecto.

En el Anexo 41 se pueden encontrar argumentados estos procesos.

En el Anexo 42 se presenta una lista de riesgos para el proyecto y la forma de mitigar cada uno de ellos, además de un plan de contingencia en caso de que el riesgo se convierta en un problema. Esta lista debe ser chequeada constantemente por el líder de proyecto para detectar sus posibles manifestaciones.

2.6 Descripción de la estrategia de implantación

La estrategia propuesta se aplicará al proyecto CESPO bajo condiciones críticas, pues solo se cuenta con un mes para realizar la implantación del modelo, además el inicio de la misma no se corresponde con el inicio del proyecto, esto trae como consecuencia que no se puedan transferir todos los paquetes.

Para la realización de la misma no solo se va a tener en cuenta el procedimiento establecido para llevarlo a cabo, sino que también se hizo referencia al actual proceso de mejora CMMI el cual presenta una serie

de prácticas genéricas las cuales servirán de base para realizar diferentes actividades, las mismas se detallarán a continuación.

- **Establecer políticas de organización**

Se va a establecer y mantener una política de la organización para garantizar la planificación y realización del proceso. Los directivos son los responsables de constituir y comunicar a los integrantes los principios, la dirección, y las expectativas de la factoría.

- **Planeación del proceso**

Se va a realizar un plan para llevar a cabo el proceso, determinando todo lo que se necesita para ejecutarlo y alcanzar los objetivos establecidos. El mismo se discutirá con las partes interesadas, para llegar a un acuerdo final.

- **Proveer recursos**

Asegurar que todos los recursos necesarios para llevar a cabo el proceso estén disponibles en todo momento incluyendo el personal calificado, las herramientas propuestas y las instalaciones físicas adecuadas.

- **Asignar responsabilidades**

A cada persona del equipo de desarrollo del proyecto piloto se le asignará una responsabilidad al igual que a los restantes integrantes de la factoría. De esta forma se garantiza que se obtengan los resultados esperados. A continuación se presenta un organigrama con la estructura organizativa de los participantes en el proyecto piloto.

Figura 1.2 Estructura organizativa del proyecto piloto.

- **Entrenamiento y capacitación del personal**

Con el objetivo de capacitar a los integrantes del proyecto para la transferencia del modelo de la factoría se debe realizar un taller para introducirlos y concientizarlos en la importancia de aplicar este enfoque. Los temas a debatir abarcan la familiarización con los antecedentes y definiciones de factoría, conocimiento sobre las empresas que han utilizado este enfoque y los beneficios que han tenido con su aplicación, los modelos que existen, principalmente el que se desea transferir y sus características.

Se realizará una reunión con los integrantes para que conozcan el proceso en cuestión y estén en condiciones de llevarlo a cabo o apoyar su ejecución.

Se deben efectuar cursos orientados al trabajo con las herramientas donde el personal tenga menos conocimientos y de esta forma garantizar que los mismos puedan desarrollar su trabajo eficazmente.

Para llevar a cabo la capacitación de TSP y PSP, se va a realizar un taller, en el mismo se explicarán las principales herramientas que deben conocer y se les facilitará un ejemplo que contiene las plantillas principales para trabajar, como son: Cuaderno de Registro de Tiempos, Resumen Semanal de Actividades y Cuaderno de Defectos. El nivel de conocimiento que adquieran sobre estas herramientas y su trabajo, depende del interés y desempeño de cada persona.

- **Configuración**

Para mantener la integridad de los datos y del producto se van a establecer diferentes niveles de control, se contará con políticas de seguridad de la información y con un repositorio donde se localizará la última versión del producto de trabajo en uso, al igual que otros documentos de interés para el proyecto.

- **Identificar involucrados relevantes**

Se garantizará la participación de los interesados durante la ejecución del proceso, estos se mantendrán motivados y comprometidos.

- **Monitoreo y control del proceso**

Se debe realizar un seguimiento continuo del desempeño laboral de cada uno de los miembros del equipo y también se debe efectuar un seguimiento preciso de cada paquete definido en la estrategia, así como de cada uno de los procesos que engloba.

Se controlará el proceso de acuerdo al plan trazado, para lograr esto se establecerá una vigilancia diaria y una supervisión estricta del mismo, permitiendo tomar medidas cuando sea necesario.

Se tendrá en cuenta el cumplimiento de la calidad del producto, se contará con personas dentro de la factoría pero ajenas al equipo de desarrollo, que certifiquen que el proceso se desarrolló según lo previsto y de acuerdo a las normas y procedimientos establecidos.

- **Retroalimentar a los directivos**

Se mantendrán informados los directivos de la factoría de lo que se está realizando en cada momento y se consultarán en caso de ser necesario.

- **Evaluar resultados**

Se realizará una evaluación de la implantación de la estrategia, lo cual valdrá para conocer los procesos que hay que perfeccionar y las mejoras que se obtuvieron. Esto servirá de base para una segunda fase de implantación.

2.7 Conclusiones

En este capítulo se describió la Universidad de las Ciencias Informáticas, se puntualizó cómo funciona el proyecto de Fábrica de Portales de la Facultad 10, se conocieron las deficiencias existentes en el mismo y la solución a tomar para erradicarlas. Se detallaron los métodos, procedimientos y técnicas utilizadas para llevar a cabo la investigación. Además, se identificaron y describieron los paquetes de transferencia del Modelo Factoría de Software aplicando Inteligencia al proyecto, donde se especificaron las características y objetivos de cada una de las entidades, sus relaciones y procesos, y los riesgos que pueden surgir durante la implantación de la factoría.

CAPÍTULO 3: RESULTADOS DE LA IMPLANTACIÓN

3.1 Introducción

En el presente capítulo se muestran los resultados de la aplicación de los métodos, los procedimientos, las técnicas y el modelo propuesto. Los resultados obtenidos al probar algunos de los elementos que fueron propuestos en la estrategia de implantación del modelo de Factoría de Software aplicando Inteligencia al proyecto Sistema de Procesamiento de Opiniones para CESPO. Además se evidencian las experiencias adquiridas y las ventajas que puede ofrecer llevar a cabo la aplicación del enfoque factoría de software.

3.2 Resultados de las encuestas realizadas

Se encuestó según lo planificado a un total de 26 personas, que cumplen diferentes roles, siendo la muestra un 50% de toda la población. Estas encuestas arrojaron como resultado que aunque todos los encuestados conocían el objetivo del proyecto Fábrica de Portales el 31% no conocía el procedimiento para desarrollar un portal. El 65% no sabía cómo se encontraba estructurado el proyecto y solo el 27% conocía las deficiencias del mismo. Aunque casi en su totalidad los encuestados tenían conocimientos del rol que desempeñaban en el proyecto un 46% no conocía los artefactos generados por el mismo y el 42% desconocía las actividades que debían desarrollar.

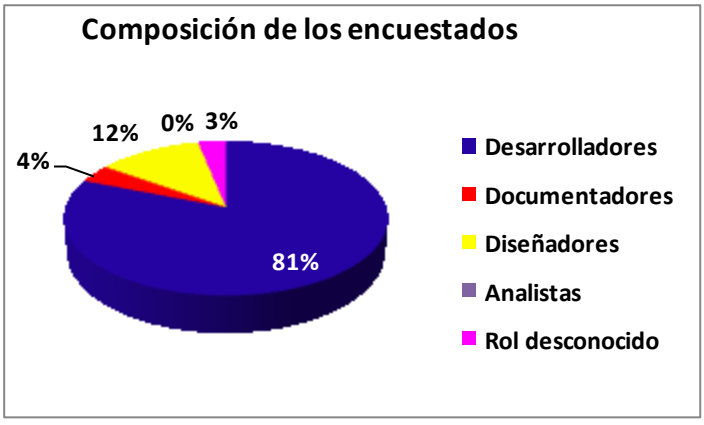


Figura 1.3 Composición de los encuestados.

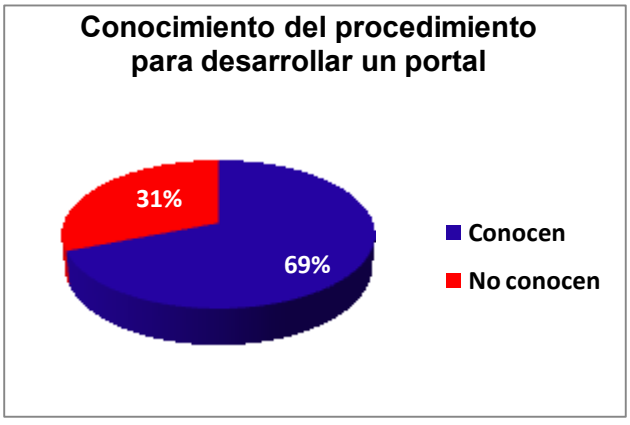


Figura 1.4 Procedimiento para desarrollar un Portal.

Un 38% no sabía que metodología se empleaba en el proyecto ni los flujos de trabajo definidos. El 77% de los encuestados determinó que la comunicación interpersonal era buena y en algunos casos regular, además existía un 77% que desconocía totalmente las técnicas de planificación empleadas en el proyecto. El 54% desconocía totalmente las herramientas a utilizar en el proyecto para el control, gestión y seguimiento de errores, para el modelado y para las salvas automáticas.

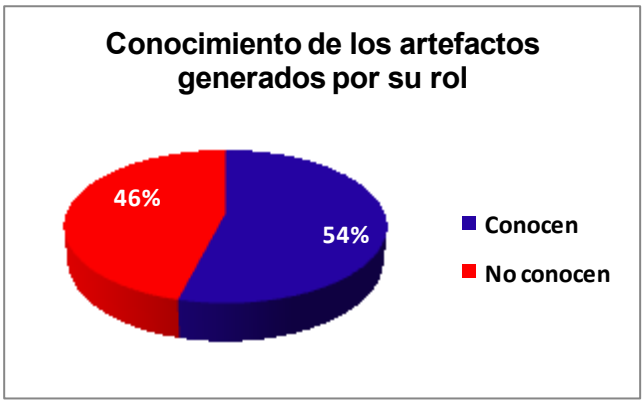


Figura 1.5 Artefactos generados

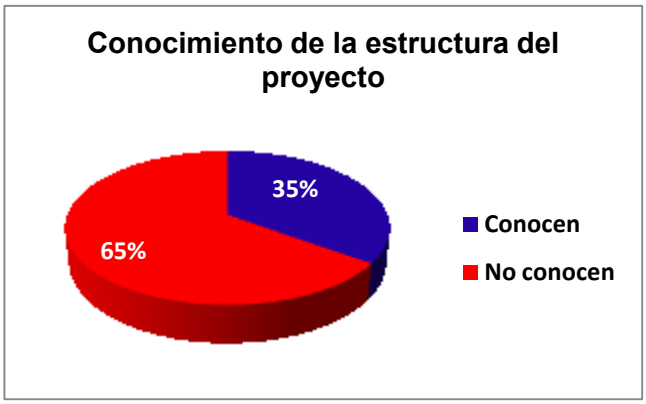


Figura 1.6 Estructura del proyecto

Solo el 27% afirman que los componentes almacenados en el repositorio son reutilizados. El 46% niega la realización de la firma de un acta como constancia del levantamiento de requisitos y el 50% no conocen las políticas de seguridad de la información establecidas en la fábrica. A partir de los resultados se realizó un diagrama Causa-Efecto o de Ishikawa que se encuentra la siguiente figura:

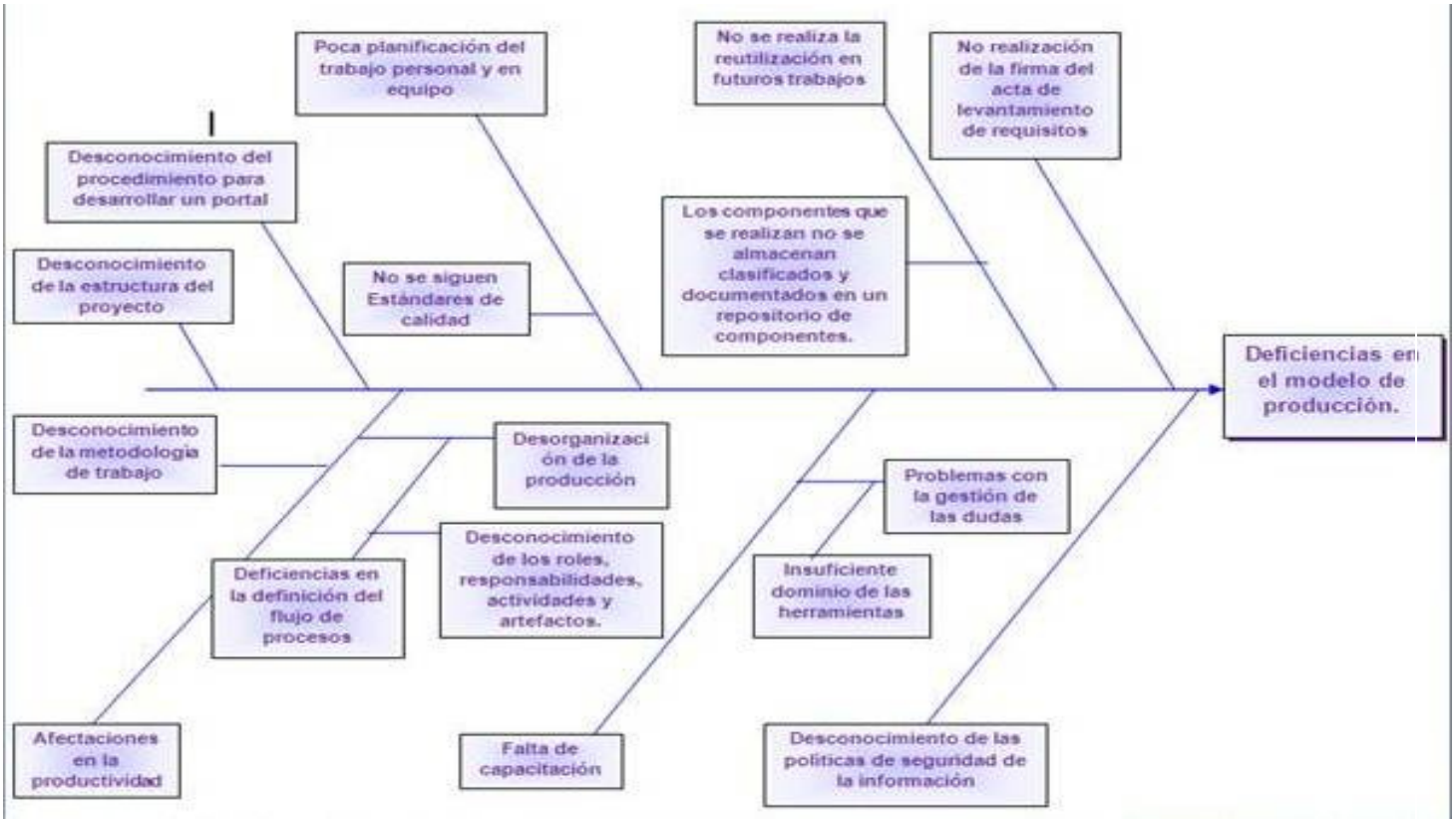


Figura 1.7 Diagrama de causa y efecto

3.3 Resultados de las entrevistas realizadas

La entrevista efectuada al líder del proyecto demostró que no se realiza una buena gestión de proyecto pues no se efectúan estimaciones de costo, tiempo y recursos basadas en procedimientos reales y de acuerdo a las características del proyecto. No existen políticas de seguridad de la información bien definidas y el mismo no cuenta con una estrategia para ejecutar las pruebas.

Las entrevistas realizadas a los clientes demostraron que si se firma un acta como constancia del levantamiento de requisitos y que la comunicación con el equipo de desarrollo y con los directivos de la fábrica es buena.

3.4 Resultados de la implantación de la estrategia

La factoría de software se implantó al proyecto CESPO, siendo ésta la prueba piloto del modelo propuesto y se llevó a cabo durante un mes. La estrategia propuesta se terminó de definir cuando el proyecto llevaba

alrededor de 3 meses de su desarrollo, esto trajo consigo que no se aplicaran muchos de los procesos que esta tenía definidos como son el estudio de la factibilidad, la gestión inicial del alcance, tiempo, entre otras.

3.4.1 Gestión de proyectos

La gestión de proyectos es sumamente importante a la hora de obtener un producto con calidad, por lo tanto es necesario que esta se realice al comienzo del proyecto. Debido a que la estrategia propuesta no se comenzó a implantar desde sus inicios, este paquete no se probó en su totalidad, por lo que solo se explicarán detalladamente aquellos procesos donde se pudieron obtener resultados.

3.4.1.1 Gestión del alcance

El alcance del proyecto no fue determinado según lo establecido por la estrategia, ya que la factoría se puso a funcionar cuando el proyecto tenía cierto tiempo de creado y no desde sus inicios como había sido planeado.

3.4.1.2 Gestión del tiempo

La gestión del tiempo no se realizó de acuerdo a lo establecido por la factoría, sino que el tiempo de desarrollo se definió de común acuerdo entre el cliente y el líder del proyecto, el mismo no es absoluto puesto que cuando se definió no había claridad en la complejidad de los requisitos.

3.4.1.3 Gestión del costo

El costo del proyecto fue otro de los aspectos que no se probó, debido a que este sistema es de producción nacional y en estos casos no se define un precio.

3.4.1.4 Gestión de la calidad

La gestión de la calidad estuvo a cargo del director de calidad, el que se encargó de velar que se cumpliera la política de calidad durante el desarrollo del producto. No se pudo implantar la norma ISO 9001, solo se tomaron en cuenta algunos aspectos que propone la misma como la satisfacción del cliente. No se realizaron las revisiones durante el desarrollo del producto a través de las listas de chequeo.

3.4.1.5 Gestión de riesgos

La gestión de riesgo a pesar de ser unos de los puntos clave para evitar afectaciones al desarrollar el proyecto, no se tomó en cuenta pues los mismos integrantes plantearon que no tenían tiempo para esto.

3.4.2 Proceso

Este paquete solo se pudo probar en el flujo de implementación, debido a las limitaciones de tiempo a la hora de implantar el modelo de factoría. Pese a esto se obtuvo una respuesta positiva por parte de los desarrolladores, pero no se tuvieron en cuenta las listas de chequeo propuestas, pues todavía no se ha concluido este flujo.

3.4.3 Personas

Para transferir este paquete primeramente se realizó un taller donde se les explicó a todos los integrantes en qué consistía factoría de software y los beneficios que aporta su aplicación.

Luego se crearon los equipos por especialidades y se realizaron reuniones independientes con los representantes de cada uno de los equipos donde se les dio a conocer el flujo de procesos, la nueva estructura de trabajo, las actividades y artefactos que debían desarrollar y se explicó que las herramientas que se utilizan son las que define la factoría.

Para la capacitación se realizó un taller para introducirlos en el mundo de factoría, se solicitó que se impartieran cursos sobre las herramientas que menos dominio tenían los miembros y se impartió un taller sobre TSP y PSP.

Se utilizó como organigrama de equipos el descentralizado controlado, pues permitió tener un jefe bien definido y jefes de equipos con responsabilidades sobre cada una de las tareas, estos equipos conformados eran pequeños, lo que trajo como consecuencia que la comunicación fuera buena.

Se asignaron los roles a nivel de factoría y de equipo, así como las tareas que cada uno debía cumplir de acuerdo a su rol de trabajo, según el sistema de conocimientos, habilidades y valores que deben tener para ello, sin embargo no se seleccionaron las personas para integrar el equipo de desarrollo pues cuando se comenzó a implantar el piloto ya el equipo estaba conformado y llevaba varios meses funcionando. Con relación al uso de PSP todo el mundo estuvo de acuerdo en su utilización por las facilidades que brinda, pero pidieron que se realizaran otros talleres que les ayudaran a profundizar en el tema.

3.4.4 Bases tecnológicas

Se definieron todas las herramientas a utilizar en la factoría y los trabajadores que iban a interactuar con cada una, así como las políticas para la seguridad de la información. Para la realización del proyecto piloto el líder del proyecto utilizó como Sistema de Gestión Documental y de Seguimiento de Errores el Gforge, los desarrolladores emplearon el lenguaje de programación PHP y utilizaron como Entorno Integrado de Desarrollo ZendStudio y como Gestor de Base de datos PostgreSql, los analistas y la documentadora utilizaron la herramienta de modelado Visual Paradigm. Además se establecieron políticas de seguridad de la información para las estaciones de trabajo, que fueron circuladas por el gestor de la factoría vía correo electrónico.

3.4.5 Repositorio

Este paquete se transfirió completamente quedando guardados todos los artefactos generados por los flujos de trabajo, como se muestra en la siguiente figura:

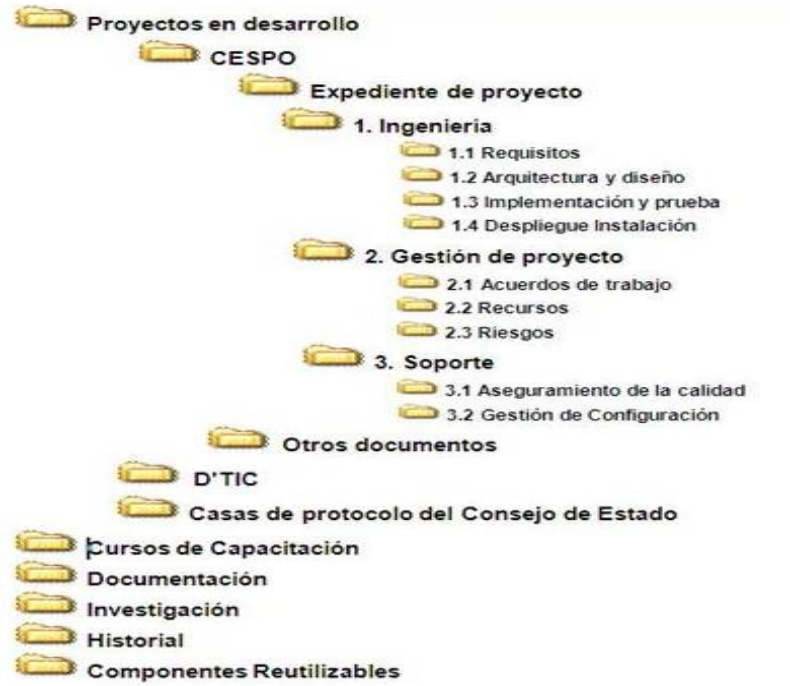


Figura 1.8 Estructura del repositorio

Al existir varios meses de trabajo y componentes reutilizables sin almacenar se previó la realización de un catálogo donde se recogieron los datos de los componentes reutilizables.

3.5 Reporte de los resultados

El reporte de los resultados se realizó aplicando encuestas a los integrantes del proyecto. De los 26 estudiantes encuestados inicialmente se les volvió a repetir la encuesta a solo 10, teniéndose en cuenta que desempeñaran distintos roles en la factoría.

A continuación se reflejan las transformaciones ocurridas después de haber sido implantado el modelo de factoría.

Hubo un aumento del 42% del conocimiento de las actividades a desarrollar por cada rol.

Figura 1.9 Actividades a desarrollar (Antes).

Figura 1.10 Actividades a desarrollar (Después).

Hubo un aumento del 38% del conocimiento de la metodología y flujos empleados en el proyecto.

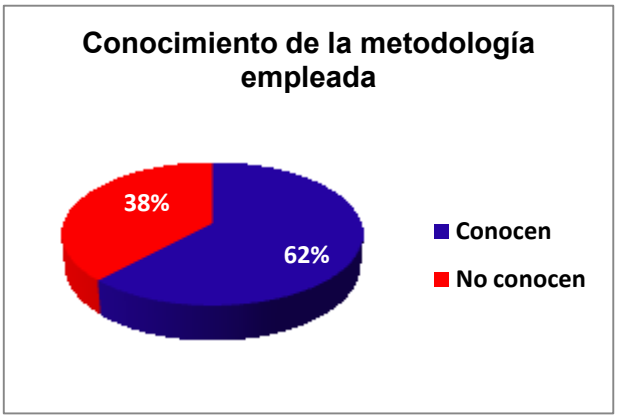


Figura 1.11 Metodología empleada (Antes)

Figura 1.12 Metodología empleada (Después)

Sobre la utilización de PSP/TSP hubo un aumento del 77%, los integrantes encuestados plantearon que les ayuda a planificarse y a detectar errores en iteraciones tempranas.

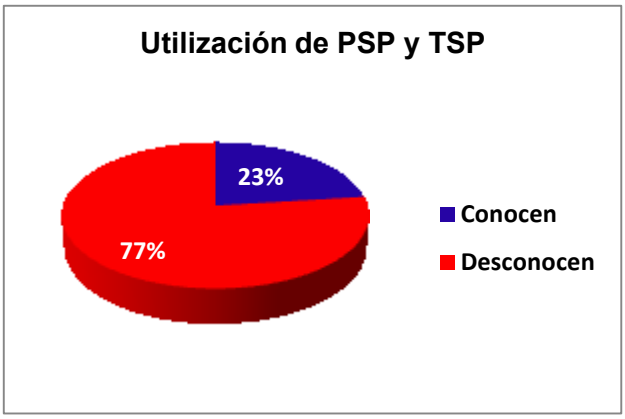


Figura 1.13 Utilización de PSP y TSP (Antes)

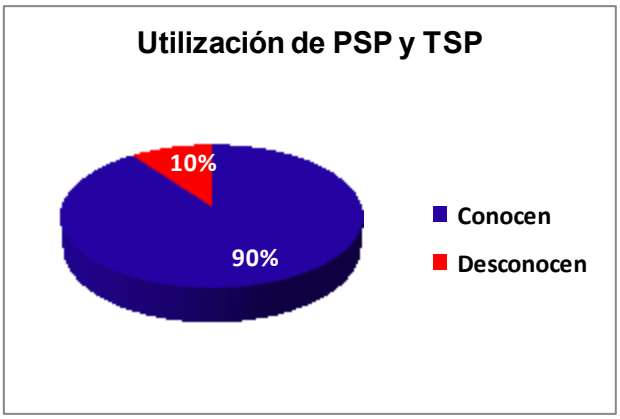


Figura 1.14 Utilización de PSP y TSP (Después)

Se produjo un aumento significativo del 54% en cuanto al conocimiento y trabajo con las herramientas.

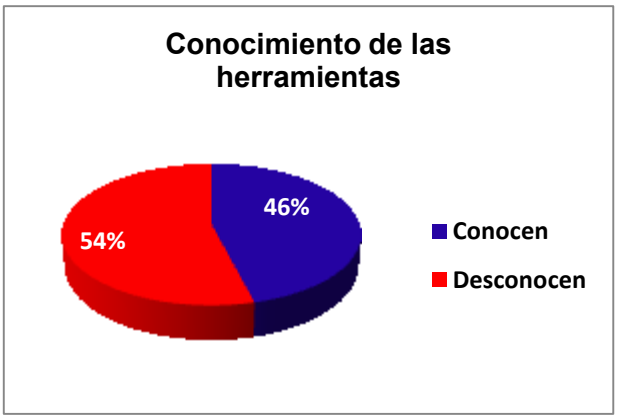


Figura 1.15 Herramientas (Antes)

Figura 1.16 Herramientas (Después)

La reutilización de componentes fue otro de los aspectos que se mejoró considerablemente, aumentando en un 53%.

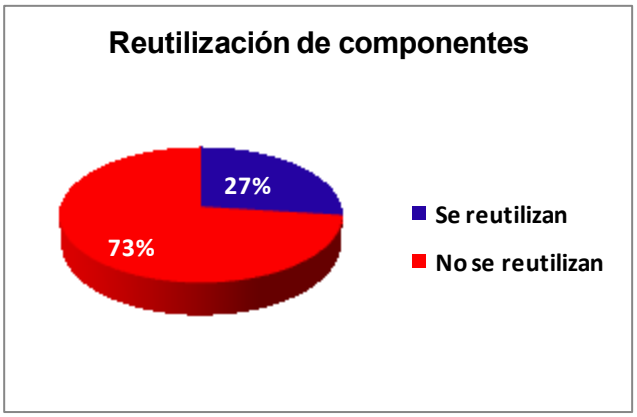


Figura 1.17 Reutilización de componentes (Antes)

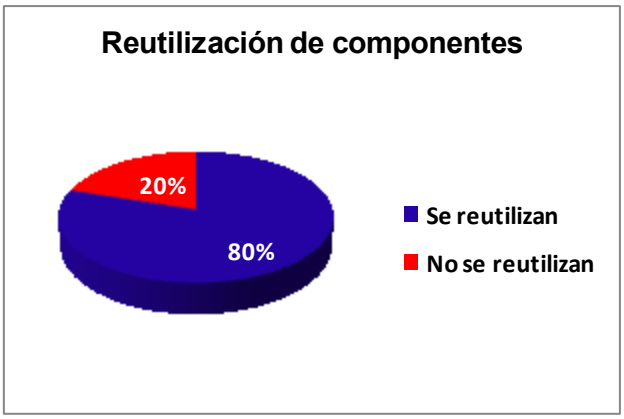


Figura 1.18 Reutilización de componentes (Después)

3.6 Conclusiones

En el presente capítulo se muestran los resultados de la aplicación de los métodos, los procedimientos y las técnicas utilizadas en la investigación.

Se explicaron los resultados obtenidos después de la aplicación de la estrategia de implantación en el proyecto piloto Sistema de Procesamiento de Opiniones para CESPO, este se puso en marcha cuando el proyecto llevaba alrededor de tres meses de desarrollo, por lo que no se pudo implantar en su totalidad todos los paquetes.

Se llegó a la conclusión de que implementar en un proyecto una estrategia para transferir un modelo de factoría de software puede mejorar la organización de las personas durante la producción, ya que determinar una estructura organizativa correcta que funcione por equipos de trabajo, utilizar las herramientas idóneas tanto para el desarrollo de productos como para la organización y planificación personal, lograr la reutilización de componentes y el conocimiento del proceso de trabajo son elementos que demuestran el rendimiento y la efectividad de implantar una factoría de software.

Conclusiones

La estrategia de implantación del modelo de factoría de software:

- Contribuyó a la definición del flujo de proceso, los roles y las responsabilidades a partir de una metodología.
- Mejoró la gestión del tiempo a través del uso de PSP para la planificación a nivel personal.
- Identificó a los desarrolladores con su rol y con los objetivos del proyecto.
- Elevó el sentido de pertenencia de los desarrolladores con su proyecto y con ello el compromiso.
- Favoreció la toma de conciencia en el uso y organización de los componentes reutilizables.

Por todo lo anteriormente dicho se considera cumplido el objetivo planteado en esta investigación pues se llegó a diseñar una estrategia de transferencia del Modelo de Factoría de Software aplicando Inteligencia. A pesar de no probarse en su totalidad y no haber brindado todos los resultados esperados sirvió de gran utilidad para la organización interna del trabajo, elevó el sentido de pertenencia de los desarrolladores con su proyecto y favoreció la toma de conciencia en el uso y organización de los componentes reutilizables.

Recomendaciones

El objetivo de este trabajo investigativo es diseñar la estrategia de transferencia de un modelo de factoría de software. Por lo que se considera necesario implantar completamente esta estrategia, ya que en este trabajo solo se prueban algunos elementos de la misma. Se recomienda:

1. Implantar esta estrategia en otros proyectos dentro de la misma Fábrica de Portales desde sus inicios.
2. Implantar esta estrategia en proyectos similares en otras facultades de la universidad.
3. Redefinir el modelo propuesto para otra línea de producción o proyecto productivo.
4. Utilizar como proceso de desarrollo el propuesto por el Ing. William Santana Méndez.

Trabajos citados

AAEN, IVAN, MATHIASSEN, LARS y BOTCHER, PETER. 1997. Software Factories. [En línea] 1997. [Citado el: 2 de 3 de 2009.] http://www.cin.ufpe.br/~in953/lectures/papers/Software_Factories_17.pdf.

ABREU, BARTOLOMEO, YANEDI y COLOME, DUNIA MARIA. 2007. *Portal de los Institutos Politécnicos de Informática.* La Habana : s.n., 2007.

ACCENTURE. 2009. [En línea] 2009. [Citado el: 27 de 4 de 2009.] http://www.accenture.com/Countries/Spain/About_Accenture/default.htm.

ALHAMBRA-EIDOS. 2007. [En línea] 2007. [Citado el: 2 de 5 de 2009.] http://www.alhambra-eidos.es/index.html.

ALTAMIRANDA, JUNIOR. 2009. *PLATAFORMA GFORGE, Universidad de los Andes.* [En línea] 2009. [Citado el: 3 de 2 de 2009.] http://sistemas.fsl.fundacite-merida.gob.ve/docman/view.php/16/116/gforge.pdf.

ÁLVAREZ, MIGUEL ANGEL. 2003. *Desarrollo Web. Zend Studio.* [En línea] 2003. [Citado el: 10 de 3 de 2009.] http://www.desarrolloweb.com/articulos/1178.php.

ANONIMO. 2008. *Microsoft y Tata Consultancy Services proporcionan servicios a los proveedores de telecomunicaciones.* [En línea] 2008. [Citado el: 2 de 5 de 2009.] http://www.prnewswire.co.uk/cgi/news/release?id=145706.

BANDINELLI, S. et al, {{*Modelingand Improving an Industrial Software Process*,,, IEEE Trans. Software Engi-neering, vol. 21, n.", Mayo 1995, pp. 440-454.

BASILI, VICTOR R, CALDIERA, GIANLUIGI y CANTONE, GIOVANNI. 1992 . *A Reference Archiecture for the Component Factory. ACM Transaction on Software Engineering and Methodology.* 1992 .

BAUER, E. L. 1972. *Software Engineering, Information Processing.* 71, North Holland Publishing Co,Amsterdam : s.n., 1972.

BECK, K. 2000. *“Extreme Programming Explained. Embrace Change”,Pearson Education,1999,Traducido al español como: “Una explicación de la programación extrema. Aceptar el cambio”, Addison Wesley.* 2000.

BOEHM, B. 1976. *Software Engineering, IEEE Transactions on Computers.* 1976. C-25, num. 12, pp. 1226-.1241.

BOEHM, B. 1998. *Usingth e WINWIN Spiral Model: A Case Study, Computer, vol. 31, n."7, pp. 33-44.* 1998.

BERNHARD RUMPE y ASTRID SCHÖDER. *“Quantitative survey on Extreme Programming Projects*”. Informe, Universidad de Tecnología de Munich, 2001.

BROOKS, F. 1975. *The Mytical Man-Month, Addison-Wesley.* 1975.

CALERO SOLIS, MANUEL. 2003. Una explicación de la programación extrema (XP).V Encuentro usuarios xBase. [En línea] 2003. [Citado el: 16 de 2 de 2009.] http://www.willydev.net/InsiteCreation/v1.0/descargas/prev/explicaxp.pdf.

CANTONE, GIOVANNI. 1992. *Software Factory: Modeling the Improvement. 'Competitive Performance Through Advanced Technology'. Third International Conference on (Conf. Publ. No. 359).* 1992.

CAPGEMINI. 2009. [En línea] 2009. [Citado el: 03 de 05 de 2009.] http://www.es.capgemini.com/quienes/.

COMUNIDAD JOOMLA. 2008. [En línea] 2008. [Citado el: 2 de 5 de 2009.] http://comunidadjoomla.org/.

CUSUMANO y A, MICHAEL. 1989. *Software Factory: A Historical Interpretation. IEEE Software.* 1989.

DAVID, A., y P. SITARAM, ccA *Concurrent Process Model for Software Developement*,,, Software Enginee-ring Notes, ACM Press, vol. 19, n." 2, Abril 1994, pp. 38-5 1.

DAVID, M.J., *ccprocess and Product: Dichotomy or Duality*,,, Software Engineering Notes, ACM Press, vol. 20, n." 2, Abril 1995, pp. 17- 18.

DBPEDIA. 2009. [En línea] 2009. [Citado el: 4 de 2 de 2009.] http://dbpedia.org/page/Repository.

DE GRACIA, CARLOS. 2008. *Estudio Comparativo: Dreamweaver CS3 vs. Visual Studio 2008. Enfocado en el manejo de XHTML, CSS y JavaScript.* [En línea] 2008. [Citado el: 13 de 1 de 2009.] http://www.scribd.com/doc/3634510/Dreamweaver-CS3-vs-Visual-Studio-2008 .

DERMID, J. y P. ROOK. 1993. *ccsoftware Development Process Models,en Software Ingineer's Reference Book, CRC Press, pp. 15126-15128.* 1993.

FABRI, JOSÉ AUGUSTO, SCHNECK D, MARCELO y S, M D M. 2004. *ElabTI: Um ambiente real e replicável de produção de software. Brasil Escola Politecnica USP.* 2004.

FACTORY, V. S. 2006. Web Corporativa Vector Software Factory. [En línea] 2006. [Citado el: 25 de 3 de 2009.] http://www.vectorsf.com/h/index2.html?preSet=noticias_2006.html.

FERNANDES, AGUINALDO ARAGON y DESCARTES DE SOUZA, TEIXEIRA. 2004. *Fábrica de Software:Implementação e Gestão de Operações.* 2004.

FERNSTROM, C, NARFELT, K H y OHLSSON, L. 1992. *Software Factory Principles, Architecture and Experiments. IEEE Software.* 1992.

FREE DOWNLOAD MANAGER. 2007. *Visual Paradigm for UML .* [En línea] 2007. [Citado el: 13 de 2 de 2009.] http://www.freedownloadmanager.org/es/downloads/Paradigma_Visual_para_UML_(M%C3%8D)_14720_p.

GARZÁS PARRA, JAVIER y PIATTINI VELTHUIS, MARIO G. 2007. *FABRICAS DEL SOFTWARE: EXPERIENCIAS, TECNOLOGÍAS Y ORGANIZACIÓN.* s.l. : RA-MA, 2007.

GEPSEA. 2009. Grupo de Estudios Prospectivos Sociedad Economía y Ambiente. *La sociedad del Conocimiento.* [En línea] 2009. [Citado el: 1 de 2 de 2009.] http://personales.com/venezuela/merida/gepsea/sc.htm.

GIRALDO, LUIS y ZAPATA, YULIANA. 2005. *Herramientas de desarrollo de ingeniería de SW para Linux .* [En línea] 2005. [Citado el: 29 de 1 de 2009.] http://hugolopez.phi.com.co/docs/download/file=Giraldo-Zapata-Herramientas%20de%20ISW.pdf,_id=17.

GROUP. 2006. *GROUP,S. The Chaos Report.* 2006.

GRUPO SMS. 2008. [En línea] 2008. [Citado el: 14 de 4 de 2009.] http://www.grupo-sms.com/index.asp.

GSI. 2007. *GRUPO SOLUCIONES INNOVA.* [En línea] 2007. [Citado el: 27 de 3 de 2009.] http://www.rational.com.ar/herramientas/rup.html.

GUGLIELMETTI, MARCOS. 2004. Mastermagazine. *Definición de Portal. .* [En línea] 2004. [Citado el: 5 de 5 de 2009.] http://www.mastermagazine.info/termino/6349.php.

HANNA, M, FAREWALL WATERFALLS, SOFTWARE MAGAZINE, Mayo 1995, pp.38-46. McDermid, J. y P. Rook, ccsoftware Development.

HERNÁNDEZ LEÓN, ROLANDO ALFREDO y G., S. C. 2002. *El Paradigma Cuantitativo de la Investigación Científica.* 2002.

IBERMATICA. 2007. [En línea] 2007. [Citado el: 24 de 3 de 2009.] http://www.ibermatica.com/ibermatica.

IEEE. 1993. *IEEE: Standards Collection: Somare Engineering.* 1993. Standard 610.12-1990.

IEEE Std 1074-1997, IEEE Standard for Developing Software Life Cycle Processes

INDIGO. 2009. *¿Qué es un Portal Web?* [En línea] 2009. [Citado el: 13 de 2 de 2009.] http://www.indigo.com.mx/temas-de-ayudausuarios/40-ique-es-un-portal-web.html?9ceb1fcf0e3e09e23cfa5c3094ad1bd9=611e5154602837870d7317005b8b5b23.

JACOBSON, I. y BOOCH, G. y RUMBAUGH, J. 2000. *"El Proceso Unificado de Desarrollo de software".Prólogo, Capítulos 1-5, Apéndice A. Visión General de UML, Apéndice B. .* s.l. : Addison-Wesley, 2000. Páginas 3-104, 407-424.

KEN y J., y R. HUNTER. 1994. *Inside RAD, McGraw-Hill.* 1994.

LETELIER, P. (2002). *Rational Unified Process (RUP).* Recuperado el 2 de 3 de 2009, de Departamento de Sistemas Informáticos y Computación, Universidad Politécnica de Valencia : https://pid.dsic.upv.es/C1/Material/Documentos%20Disponibles/Introducci%C3%B3n%20a%20RUP.doc

MARTIN, J. 1991. *Rapid Aplication Developement, Prentice-Hall.* 1991.

MENDOZA, JORGE, A. 2000. Informaticamilenium. *Los Portales, una nueva dimensión en Internet.* [En línea] 1 de 12 de 2000. [Citado el: 5 de 5 de 2009.] http://www.informaticamilenium.com.mx/Paginas/mn/articulo25.htm.

MENENDEZ, ROSA. 2007. *Rational Software Corporation.* [En línea] 2007. [Citado el: 23 de 1 de 2009.] http://www.usmp.edu.pe/publicaciones/boletin/fia/info36/proyectos.html.

NIERSTRASZ. 1992. *Component-Oriented Software Developement,CACM, vol. 35, n." 9, pp.160-165.* 1992.

PALACIO, JUAN. 2006. Navegapolis. *El modelo Scrum.* [En línea] 2006. [Citado el: 4 de 3 de 2009.] http://www.navegapolis.net/content/view/694 .

PASTOR, PALOMA. 2007. GRUPO GESFOR. [En línea] 2007. [Citado el: 19 de 3 de 2009.] http://www.gesfor.es/htm/08/salaprensa/doc/dosierprensaMarmickVS2.doc.

PECOS, DANIEL. 2009. *PostgreSQL vs. MySQL.* [En línea] 2009. [Citado el: 20 de 3 de 2009.] http://www.netpecos.org/docs/mysql_postgres/index.html.

PRESSMAN, ROGER S. 2002. *Ingeniería del software, un Enfoque Práctico. .* Madrid : s.n., 2002.

PROCESS MODELS,,, en Software Ingineer's Reference Book, CRC Press, 1993, pp. 15126-15128.

QUANTA PLUS. 2005. [En línea] 2005. [Citado el: 25 de 2 de 2009.] http://quanta.kdewebdev.org/.

RAVEST, CECILIA. 2008. [En línea] 2008. [Citado el: 28 de 4 de 2009.] http://www.bnamericas.com/company-profile/tecnologia/Tata_Consultancy_Services_Iberoamerica_S,A,-TCS_Iberoamerica.

REYERO, JOSE A. 2006. DRUPAL. [En línea] 2006. [Citado el: 13 de 1 de 2009.] http://drupal.org.es/drupal.

RÍOS, YANOSKY. 2005. *Modelo funcional de la Factoría de Software de la UCI para la línea Carrefour.* Ciudad Habana : s.n., 2005.

ROCKWELL, R. G., M. H. 1993. *The Eureka Software Factory CoRe: A Conceptual Reference Model for Software Factories. Software Engineering Environments Conference.* 1993.

ROYCE, W.W. 1970. *Managintgh e Developement of Large Software Systems: Concepts and Techniques,Proc. WESCON.* 1970.

SCHWABER K, BEEDLE M., MARTIN R.C. 1999. *"Agile Software Development with SCRUM". Prentice Hall. 2001.Prentice Hall.* 1999.

SHELEG, W., ccConcurrent Engineering: A New Para-dign for CIS Developement,,, Application Development Trends, vol. 1, n." 6, Junio 1994, pp. 28-33.

SKYNETSTUDIO. 2008. *Tecnología.* [En línea] 2008. [Citado el: 5 de 5 de 2009.] http://www.skynet-studio.com.ar/tecnologia.aspx.

SUN. 2007. *Desarrollo de aplicaciones multiplataforma con NetBeans IDE .* [En línea] 2007. [Citado el: 18 de 1 de 2009.] http://www.sun.com/emrkt/innercircle/newsletter/spain/0207spain_feature.html.

TÁPANES ROBAU, DAYSARÍH y RODRIGUEZ BATISTA, ARMANDO. 2006. *La transferencia de tecnología asociada al proceso inversionista en Cuba en el cuatrienio 2002-2005.* [En línea] 2006. [Citado el: 12 de 3 de 2009.] http://www.bibliociencias.cu/gsdl/collect/eventos/index/assoc/HASH0111/380f07a0.dir/doc.pdf.

THE APACHE SOFTWARE FOUNDATION. 2009 . . [En línea] 2009 . [Citado el: 5 de 5 de 2009 .] http://www.apache.org/.

TID. 2007. PROYECTO VULCANO. *Forja de proyectos software de calidad. Estudio de herramientas de certificación.* [En línea] 2007. [Citado el: 24 de 3 de 2009.] http://www.ines.org.es/vulcano/wp-content/uploads/2007/09/d6-estudio-de-herramientas-de-certificacion-tid.pdf.

TRUJILLO CASAÑOLA, YAIMI. 2007. *Modelo de factoría de software aplicando inteligencia.* Ciudad de la Habana : s.n., 2007.

WEB21. 2009. *Portales Web a Medida.* [En línea] 2009. [Citado el: 4 de 5 de 2009.] http://www.web21.es/index.php?opcion=1&id_nodo=136.

ZELKOVITZ M. V., SHAW A. C. y GANNON J. D. 1979. *Principles o Software Engineering and Design.* Prentice Hal : Englewoods Clif, 1979.

Bibliografía

ALTAMIRANDA, JUNIOR. *PLATAFORMA GFORGE, Universidad de los Andes.* [En línea] [Citado el: 2 de 3 de 2009.] http://sistemas.fsl.fundacite-merida.gob.ve/docman/view.php/16/116/gforge.pdf.

ÁLVAREZ, MIGUEL ANGEL. 2003. Desarrollo Web. *Zend Studio.* [En línea] 2003. [Citado el: 10 de 3 de 2009.] http://www.desarrolloweb.com/articulos/1178.php.

ATOSORIGIN. 2009. [En línea] 2009. [Citado el: 6 de 2 de 2009.] http://www.es.atosorigin.com/es-es/.

BANDINELLI, S. et al, {{*Modelingand Improving an Industrial Software Process*,,, IEEE Trans. Software Engi-neering, vol. 21, n.", Mayo 1995, pp. 440-454.

BERNHARD RUMPE y ASTRID SCHÖDER. *"Quantitative survey on Extreme Programming Projects*". Informe, Universidad de Tecnología de Munich, 2001.

CALERO SOLIS, MANUEL. 2003. *Una explicación de la programación extrema (XP).V Encuentro usuarios xBase. MADRID.* [En línea] 2003. [Citado el: 18 de 2 de 2009.] http://www.willydev.net/InsiteCreation/v1.0/descargas/prev/explicaxp.pdf.

CANÓS, JOSE H, LETELIER, PATRICIO y PENADÉS, Mª CARMEN. *Metodologías Ágiles en el Desarrollo de Software.* [En línea] [Citado el: 23 de 2 de 2009.] http://www.willydev.net/descargas/prev/TodoAgil.Pdf.

CANTONE, GIOVANNI. 1992. *Software Factory: Modeling the Improvement. 'Competitive Performance Through Advanced Technology'. Third International Conference on (Conf. Publ. No. 359).* 1992.

CISNEROS MARTÍNEZ, NIURBELIS. 2007. *Estrategia de implantación del Modelo de Factoría de Software aplicando Inteligencia para el Polo Productivo Gestión de Proyecto.* Ciudad Habana : s.n., 2007.

CUSUMANO y A, MICHAEL. 1989. *Software Factory: A Historical Interpretation. IEEE Software.* 1989.

DAVID, A., y P. SITARAM, ccA *Concurrent Process Model for Software Developement*,,, Software Enginee-ring Notes, ACM Press, vol. 19, n." 2, Abril 1994, pp. 38-5 1.

DAVID, M.J., *ccprocess and Product: Dichotomy or Duality*,,, Software Engineering Notes, ACM Press, vol. 20, n." 2, Abril 1995, pp. 17- 18.

DE GRACIA, CARLOS. 2008. Estudio Comparativo: Dreamweaver CS3 vs. Visual Studio 2008. Enfocado en el manejo de XHTML, CSS y JavaScript. [En línea] 2008. [Citado el: 13 de 1 de 2009.] http://www.scribd.com/doc/3634510/Dreamweaver-CS3-vs-Visual-Studio-2008.

DÍAZ FLORES, MIRIAN MILAGROS. Diferencia de RUP y XP, Escuela de Ingeniería de Sistemas. [En línea] [Citado el: 12 de 3 de 2009.] http://www.usmp.edu.pe/publicaciones/boletin/fia/info49/articulos/RUP%20vs.%20XP.pdf.

Drupal. 2005. [En línea] DRUPAL, 2005. [Citado el: 13 de 1 de 2009.] http://drupal.org.es/drupal.

FACTORY, V. S. 2009. Web Corporativa Vector Software Factory. [En línea] 2009. [Citado el: 25 de 3 de 2009.] http://www.vectorsf.com/h/index2.html?preSet=noticias_2006.html.

FERNANDES, AGUINALDO ARAGON y TEIXEIRA, DESCARTES DE SOUZA. 2004. *Fábrica de Software:Implementação e Gestão de Operações.* s.l. : Atlas, 2004.

FERNSTROM, C. 1991. *The Eureka Software Factory: Concepts and Accomplishment. Proceedings Third European Software Engineering Conference.* Berlin : s.n., 1991.

FERNSTROM, C, NARFELT, K H y OHLSSON, L. 1992. *Software Factory Principles, Architecture and Experiments. IEEE Software.* 1992.

GIRALDO, LUIS y ZAPATA, YULIANA. 2005. Herramientas de desarrollo de ingeniería de SW para Linux. [En línea] 2005. [Citado el: 14 de 1 de 2009.] http://hugolopez.phi.com.co/docs/download/file=Giraldo-Zapata-Herramientas%20de%20ISW.pdf,_id=17.

GSI. 2007. GRUPO SOLUCIONES INNOVA. [En línea] 2007. http://www.rational.com.ar/herramientas/rup.html.

2004. *Guía de los Fundamentos de la Dirección de Proyectos (Guía del PMBOK®) Tercera Edición.* EE.UU : Project Management Institute, Four Campus Boulevard, Newtown Square, PA, 2004. 19073-3299.

HANNA, M, FAREWALL WATERFALLS, SOFTWARE MAGAZINE, Mayo 1995, pp.38-46. McDermid, J. y P. Rook, ccsoftware Development.

HERNÁNDEZ LEÓN, ROLANDO ALFREDO y G., S. C. 2002. *El Paradigma Cuantitativo de la Investigación Científica.* 2002.

HERRERO GONZÁLES, LIDISVEY y CORRALES GUERRA, ZILENYS. 2007. *Estrategia de Implantación del Modelo de Factoría de Software aplicando Inteligencia para Contenidos Educativos.* 2007.

IBERMATICA. 2007. [En línea] 2007. [Citado el: 24 de 3 de 2009.] http://www.ibermatica.com/ibermatica.

IEEE Std 1074-1997, IEEE Standard for Developing Software Life Cycle Processes

INDRA. 2008. [En línea] 2008. [Citado el: 1 de 2 de 2009.] http://www.indra.es/servlet/ContentServer?cid=1082008092186&pagename=IndraES%2FPage%2FEstructMenuRastroModulos&Language=es_ES&pid=1082008092186&c=Page.

JOOMLAOS.NET. 2005 . ¿Qué es Joomla? [En línea] 2005 . [Citado el: 27 de 3 de 2009.] http://www.joomlaos.net/-que-es-joomla--4.php.

LETELIER, PATRICIO. 2005. Portal Desarrollo de Software. *Introducción a RUP, Universidad Politécnica de Valencia.* [En línea] 2005. [Citado el: 15 de 2 de 2009.] https://pid.dsic.upv.es/C1/Material/default.aspx.

—. 2002. Rational Unified Process (RUP). *Departamento de Sistemas Informáticos y Computación, Universidad Politécnica de Valencia .* [En línea] 2002. [Citado el: 2 de 3 de 2009.] https://pid.dsic.upv.es/C1/Material/Documentos%20Disponibles/Introducci%C3%B3n%20a%20RUP.doc.

LETELIER, PATRICIO y PENADÉS, Mª CARMEN. 2006. Portal Desarrollo de Software. *Metodologías ágiles para el desarrollo de software: eXtreme Programming (XP), Universidad Politécnica de Valencia.* [En línea] 2006. [Citado el: 14 de 2 de 2009.] http://www.willydev.net/descargas/masyxp.pdf.

MENENDEZ, ROSA. 2007. *Rational Software Corporation.* [En línea] 2007. [Citado el: 23 de 1 de 2009.] http://www.usmp.edu.pe/publicaciones/boletin/fia/info36/proyectos.html.

Metodologías Ágiles en el Desarrollo de Software. [En línea] [Citado el: 24 de 2 de 2009.] http://www.willydev.net/descargas/prev/TodoAgil.Pdf.

PALACIO, JUAN. 2007. Navegapolis. *Flexibilidad con Scrum.* [En línea] 2007. [Citado el: 2 de 3 de 2009.] http://www.navegapolis.net/files/Flexibilidad_con_Scrum.pdf.

—. 2006. Navegapolis. *El modelo Scrum.* [En línea] 2006. [Citado el: 4 de 3 de 2009.] http://www.navegapolis.net/content/view/694.

PASTOR, PALOMA. 2007. GRUPO GESFOR. [En línea] 2007. [Citado el: 19 de 3 de 2009.] http://www.gesfor.es/htm/08/salaprensa/doc/dosierprensaMarmickVS2.doc.

PECOS, DANIEL. 2009. PostgreSQL vs. MySQL. [En línea] 2009. [Citado el: 20 de 3 de 2009.] http://www.netpecos.org/docs/mysql_postgres/index.html.

PÉREZ VALDÉS, DAMIAN. 2008. Editores web que facilitan tu trabajo. [En línea] 2008. [Citado el: 10 de 2 de 2009.] http://www.laisla.cult.cu/index2.php?option=com_content&do_pdf=1&id=19.

PRESSMAN, ROGER S. 2002. *Ingeniería del software, un Enfoque Práctico.* Madrid : s.n., 2002. 84-481-3214-3219 p.

PROCESS MODELS,,, en Software Ingineer's Reference Book, CRC Press, 1993, pp. 15126-15128.

QUANTA PLUS. 2005. [En línea] QUANTA PLUS, 2005. [Citado el: 25 de 2 de 2009.] http://quanta.kdewebdev.org/.

SANTANA MÉNDEZ, WILLIAM. 2009. *Propuesta de procedimiento para el desarrollo de portales Web.* Ciudad Habana : s.n., 2009.

SHELEG, W., ccConcurrent Engineering: A New Para-dign for CIS Developement,,, Application Development Trends, vol. 1, n." 6, Junio 1994, pp. 28-33.

SOFTONIC. 2008. *"El mejor editor HTML existente para Linux".* [En línea] SOFTONIC, 2008. [Citado el: 16 de 1 de 2009.] http://quanta-plus.softonic.com/linux.

SOFTWARE ENGINEERING INSTITUTE. 2006. *CMMI® for Development,Version 1.2.* Pittsburgh : s.n., 2006. PA 15213-3890.

SUN. 2007. *Desarrollo de aplicaciones multiplataforma con NetBeans IDE.* [En línea] SUN, 2007. [Citado el: 18 de 1 de 2009.] http://www.sun.com/emrkt/innercircle/newsletter/spain/0207spain_feature.html.

TÁPANES, ROBAU , DAYSARÍH y RODRIGUEZ, BATISTA, ARMANDO. 2006. La transferencia de tecnología asociada al proceso inversionista en Cuba en el cuatrienio 2002-2005. [En línea] 2006. [Citado el: 12 de 3 de 2009.] http://www.bibliociencias.cu/gsdl/collect/eventos/index/assoc/HASH0111/380f07a0.dir/doc.pdf.

TID. 2007. PROYECTO VULCANO. *Forja de proyectos software de calidad. Estudio de herramientas de certificación.* [En línea] 2007. [Citado el: 24 de 3 de 2009.] http://www.ines.org.es/vulcano/wp-content/uploads/2007/09/d6-estudio-de-herramientas-de-certificacion-tid.pdf.

TOP NOTCH THEMES. 2009. *Drupal vs. Joomla: a frank comparison from an IBM consultant.* [En línea] 2009. [Citado el: 2 de 3 de 2009.] http://www.topnotchthemes.com/blog/090224/drupal-vs-joomla-frank-comparison-ibm-consultant.

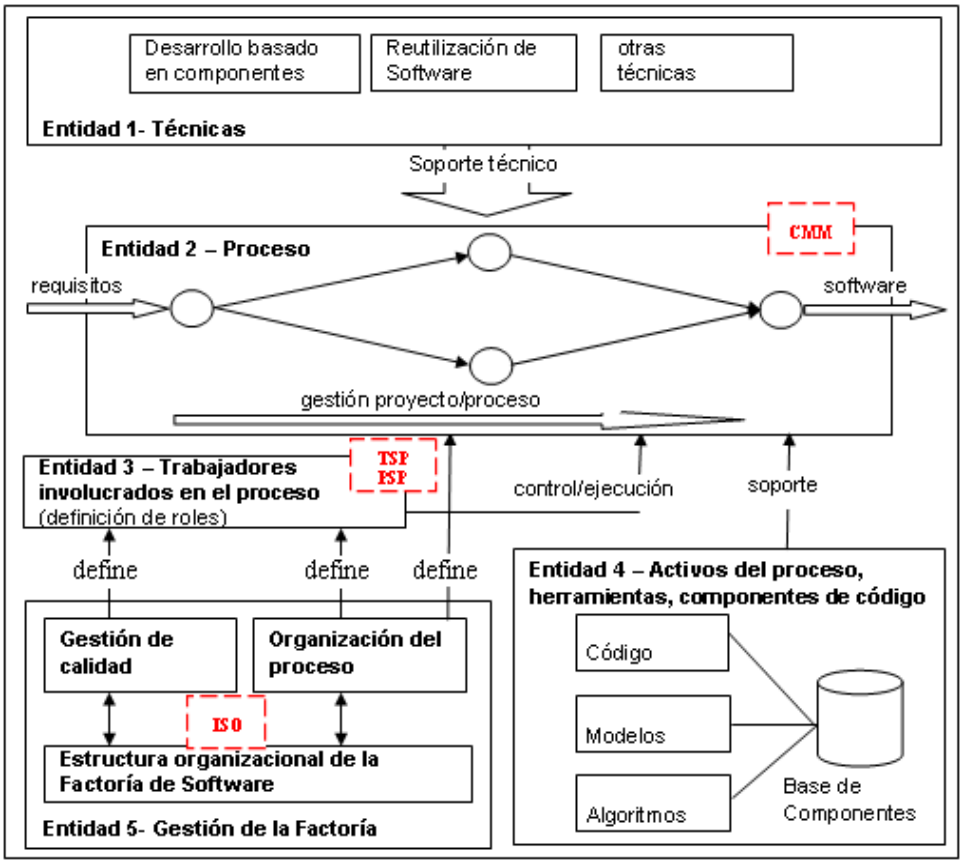
TRUJILLO CASAÑOLA, YAIMI. 2007. Modelo de factoría de software aplicando inteligencia. Ciudad de la Habana : s.n., 19 de 6 de 2007.

T-SYSTEMS. 2009. [En línea] 2009. [Citado el: 4 de 2 de 2009.] http://www.t-systems.es/tsi/es/100100/Homepage.

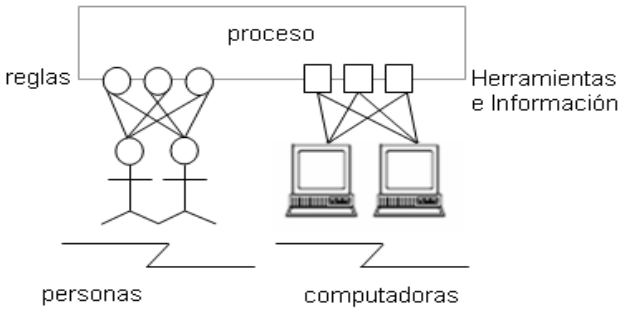
VICTOR R, BASILI, GIANLUIGI, CALDIERA y CANTONE, GIOVANNI. 1992. *A Reference Archiecture for the Component Factory. ACM Transaction on Software Engineering and Methodology.* 1992

Anexos

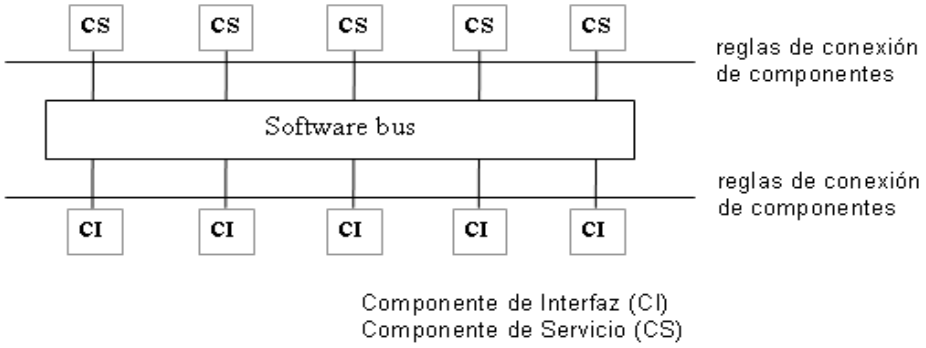
Anexo 1: Modelo basado en la norma ISO 9001 y CMM



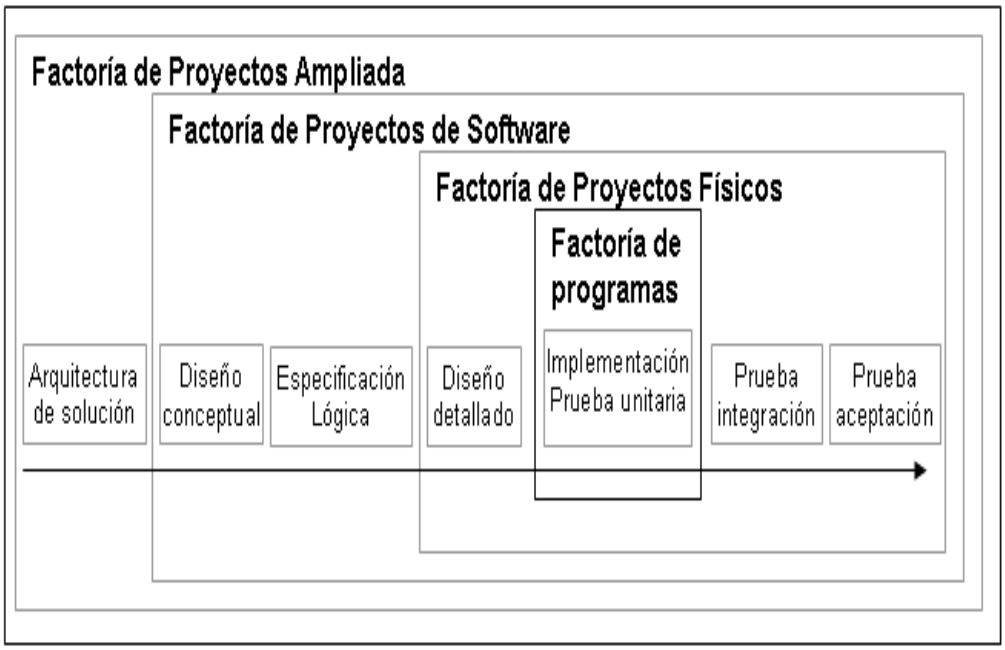
Anexo 2: Modelo Eureka



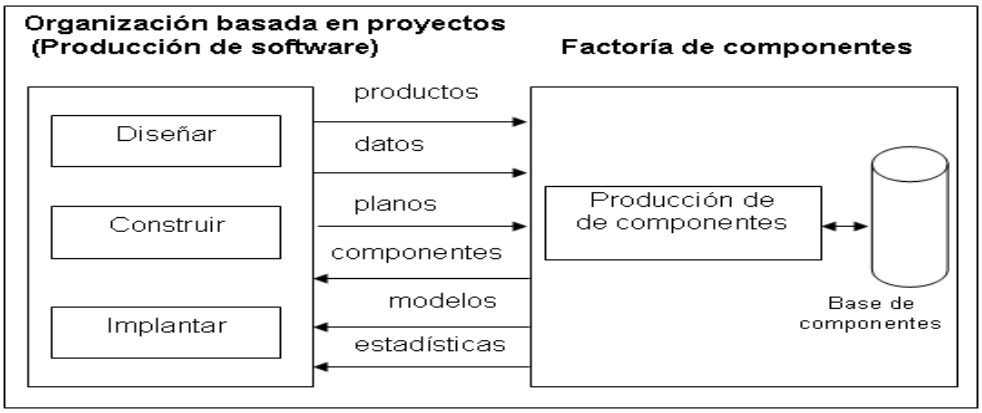
Anexo 3: Enfoque software bus (Modelo Eureka)



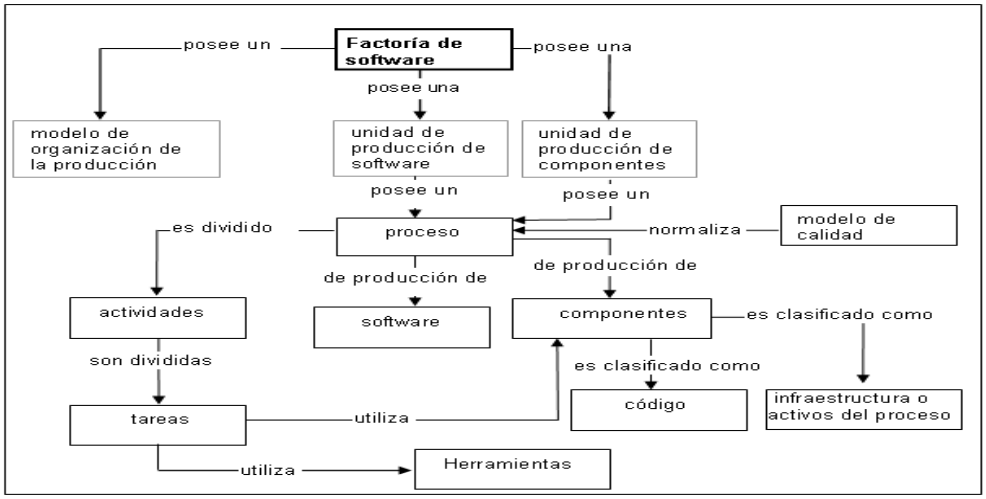
Anexo 4: Modelo Clasificatorio



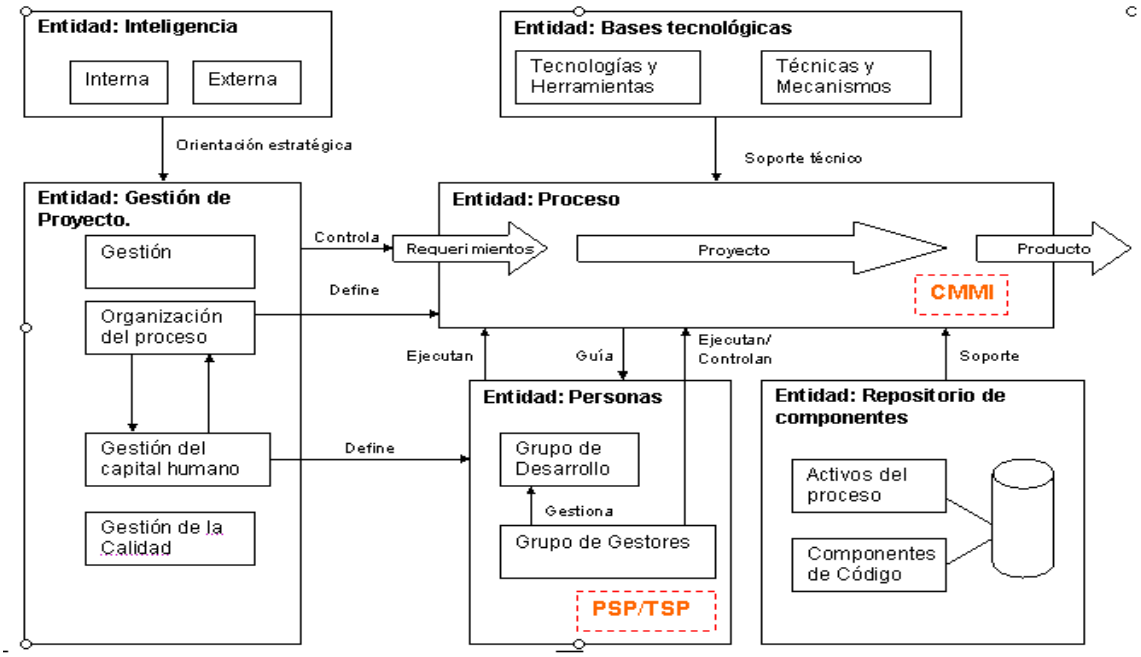
Anexo 5: Modelo propuesto por Basili



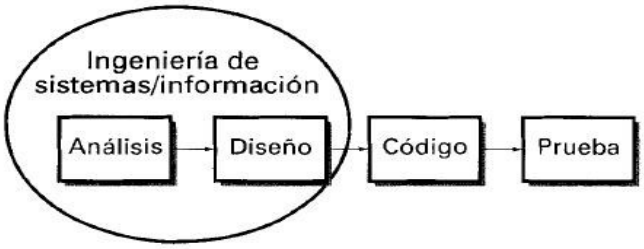
Anexo 6: Modelo Replicable



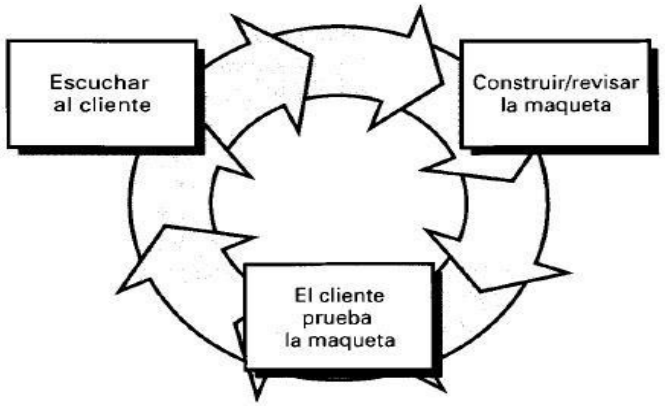
Anexo 7: Modelo aplicando Inteligencia



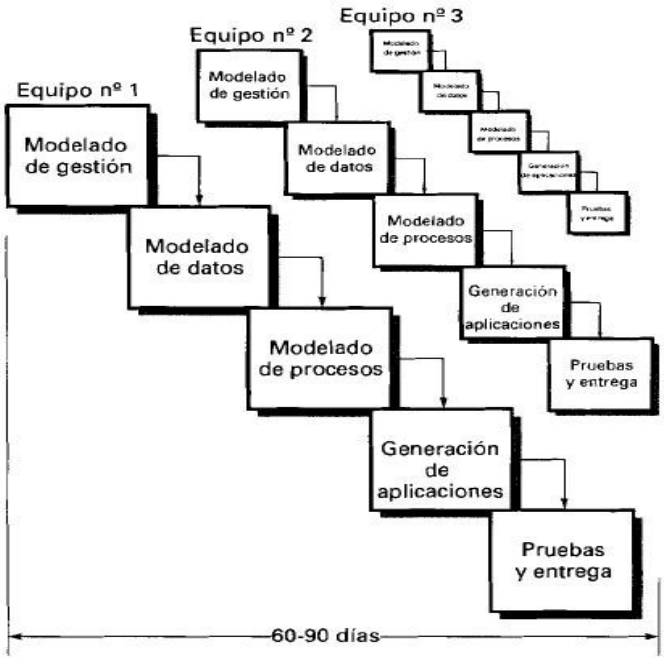
Anexo 8: Modelo lineal secuencial (Modelo en Cascada)



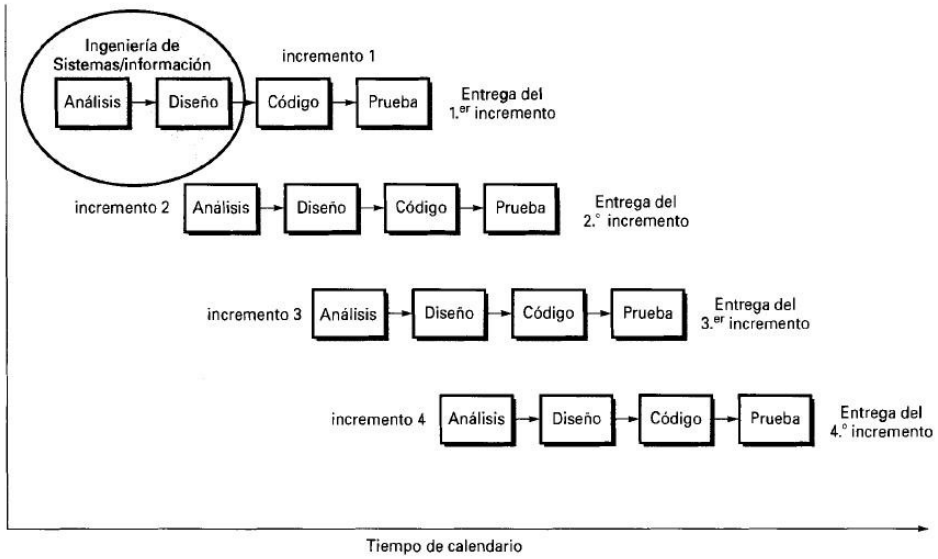
Anexo 9: Modelo construcción de prototipos



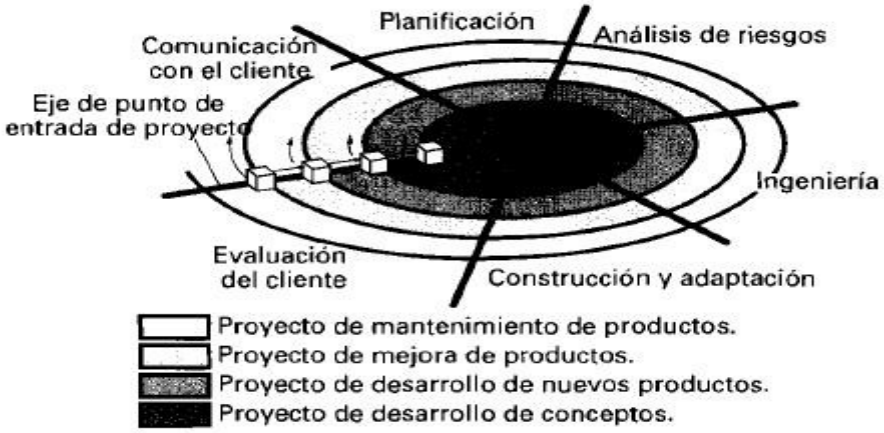
Anexo 10: Modelo DRA (Desarrollo Rápido de Aplicaciones)



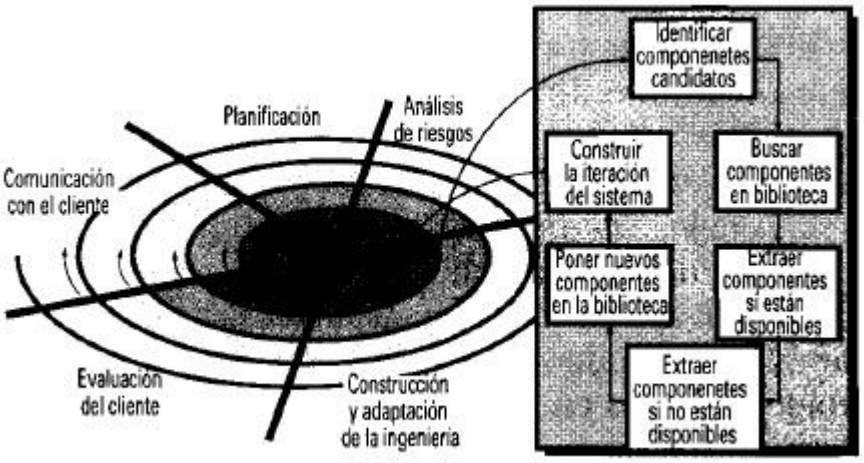
Anexo 11: Modelos Evolutivos del proceso del software: Incremental



Anexo 12: Modelos Evolutivos del proceso del software: Espiral



Anexo 13: Desarrollo basado en componentes



Anexo 14: Matriz de comparación entre modelos de desarrollo del software

Nombre del Modelo	Acrónimo	Tipo de Modelo	Ventajas	Desventajas
Lineal Secuencial	No tiene	Cascada	- Paradigma más antiguo y más extensamente utilizado en la ingeniería del software. -Modelado según el ciclo de ingeniería convencional. - Proporciona una plantilla en la que se encuentran métodos para análisis, diseño, codificación, pruebas y mantenimiento. - Una fase no comienza hasta que termine la fase anterior y generalmente se incluye la corrección de los problemas encontrados en fases previas.	-Requiere que el cliente exponga explícitamente al principio todos los requisitos, lo que comúnmente es difícil que ocurra. - No estará disponible ninguna versión del programa hasta que el proyecto no esté muy avanzado. - Existe una alta probabilidad de que el software no cumpla con los requisitos del usuario por el largo tiempo de entrega del producto.

Construcción de Prototipos	No tiene	Iterativo	- Puede servir como primera versión del sistema. Se construye para servir como un mecanismo de definición de requisitos. -Los usuarios y desarrolladores logran un mejor entendimiento del sistema. Esto se refleja en una mejora de la calidad del software.	-Para el rápido desarrollo se necesitan herramientas que pueden ser incompatibles con otras o que poca gente sabe utilizar. - De forma demasiado frecuente la gestión de desarrollo del software es muy lenta.
Desarrollo Rápido de Aplicaciones	DRA	Cascada	- Permite construir sistemas utilizables en poco tiempo -Se utiliza una construcción basada en componentes. - Los entregables pueden ser fácilmente trasladados a otra plataforma. - Visibilidad temprana, una mayor flexibilidad y la codificación manual es menor.	- Requiere clientes y desarrolladores comprometidos, sino los proyectos fracasarán. - Para proyectos grandes requiere recursos humanos suficientes. - No es adecuado cuando los riesgos técnicos son altos.
Incremental	No tiene	Cascada, Iterativo	-Los clientes no tienen que esperar hasta que el sistema se entregue completamente para comenzar a hacer uso de él. - Los clientes pueden aclarar los requisitos que no tengan claros conforme ven las entregas del sistema. - Se disminuye el riesgo de fracaso de todo el proyecto.	- Cada incremento debe ser pequeño para limitar el riesgo (menos de 20.000 líneas). - Cada incremento debe aumentar la funcionalidad. - Es difícil establecer las correspondencias de los requisitos contra los incrementos. - Es difícil detectar las unidades o servicios genéricos para todo el sistema.
Espiral	No tiene	Cascada, Iterativo	-Puede adaptarse y aplicarse a lo largo de la vida del software de computadora. - Las tareas se aplican a cualquier proyecto de software que se realice, sin tener en cuenta el tamaño ni la	- Su principal desventaja es que es nuevo (1988) y no se ha utilizado tanto como otros modelos de ciclo de vida. - Requiere una considerable habilidad para la evaluación del

			complejidad. - Son apropiados particularmente para el desarrollo de sistemas orientados a objetos. - Toma en consideración explícitamente el riesgo. - Permite a quien lo desarrolla aplicar el enfoque de construcción de prototipos en cualquier etapa de evolución del producto.	riesgo. -Puede resultar difícil convencer a algunos clientes que el proceso es controlable.
Desarrollo basado en componentes	No tiene	Iterativo	-Permite alcanzar un alto índice de productividad y la reutilización del software. -Reduce el tiempo de entrega, el costo y esfuerzo del proyecto. - Disminuye los riesgos durante el desarrollo.	- Los compromisos en los requisitos son inevitables, por lo cual puede que el software no cumpla las expectativas del cliente. - Las actualizaciones de los componentes adquiridos no están en manos de los desarrolladores del sistema.

Anexo 15: Matriz de comparación entre las metodologías de desarrollo de software

Nombre de la metodología	***RUP***	XP	SCRUM
Actividades	Modelar diagrama de casos de uso del negocio y del sistema, Análisis de la arquitectura, Captura de requisitos, Diseñar la arquitectura, Identificar clases de diseño, Creación del modelo de diseño, del análisis, de despliegue y de implementación. Implementación de la arquitectura, Implementar clases, Integrar producto, Planificar prueba, Diseñar prueba, Ejecutar pruebas, Evaluar pruebas,	Planificación, Diseño, Codificación, Pruebas.	Planificación de sprint, Reunión diaria, Revisión de sprint.

	Reportar defectos, Control de cambio en la configuración, Generación de informes de estado, Auditorías de la configuración.		
Flujos de trabajo	Modelado del negocio, Requisitos, Análisis y Diseño, Implementación, Pruebas, Despliegue, Gestión del proyecto, Configuración y control de cambios, Ambiente.	Exploración, Planificación de la Entrega (Release), Iteraciones, Producción, Mantenimiento y Muerte del Proyecto	Todo el trabajo se hace en Sprint. Cada Sprint es una iteración de 30 días de calendario consecutivos.
Artefactos	Modelo de Casos de Uso, Modelo de Análisis y Diseño, Modelo de Despliegue, Modelo de Implementación, Diagrama de clases ,de interacción y subsistema de diseño, Realizaciones de casos de uso, Especificación de Requisitos, Descripción de la arquitectura software, Plan de integración de construcciones, Modelo de prueba, Plan de prueba, Casos de prueba, Evaluación de prueba, Reporte de defectos, Documento de Visión, Glosario de términos, Lista de riesgos y plan de contingencia, Plan del proyecto, Plan de Aseguramiento de la Calidad, Manual de Usuario	Historias del Usuario, Tareas de Ingeniería, Pruebas de Aceptación, Pruebas Unitarias y de Integración, Plan de la Entrega, Código	Pila del producto, Pila del sprint, Incremento.
Roles	Gestores (Líder del proyecto), Analistas, Desarrollador/Diseñador, Especialistas en Prueba, Apoyo, Otros roles.	Jefe de Proyecto, Programador**,** Cliente, Encargado de Pruebas, Rastreador, Entrenador.	Propietario del producto, Equipo de desarrollo, Scrum Manager.

Anexo 16: Matriz de comparación entre las herramientas de desarrollo de software

Herramienta	Funcionalidad	Plataforma	Lenguaje	Licencia	Ventajas	Desventajas	% en la UCI
Gestión de Contenido(CMS)							
Joomla	-Publicaciones en Internet e intranets. - Hacer caché de páginas. - Indexamiento web. - Flash con noticias. - Encuestas, foros, blogs. Administración del diseño y aspecto estético del sitio mediante la utilización de templates/plantillas. Administración de la navegación y del menú del sitio. - Permite instalar, desinstalar y administrar componentes y módulos.	-GNU/Linux -Windows -Mac OSX.	-Hecho en PHP	-GPL	-Más sencillo de usar, y de instalar. -Cuenta con un gran número de comunidades de desarrollo. -Permitir la compatibilidad hacia atrás con versiones previas de componentes, plantillas, módulos y otras extensiones. -Una serie de plugins instalados por defectos.	- Permite nada más que dos niveles de categorías: Secciones y Categorías. - Deficiente planificación de la interfaz administrativa. -Limitación de los permisos a los usuarios. - No cuenta con un conector estable para la utilización de Postgre. -Es lento. -No tiene un sistema de comentarios integrado. -Incompatibili-dad de módulos.	0

						-El contenido es principalmente texto. Orientada a portal de noticias.	
Drupal	-Realización de intranets de compañías, enseñanza en línea, comunidades de arte y administración de proyectos. -Permite publicar artículos, imágenes, u otros archivos y servicios añadidos como foros, encuestas, votaciones, blogs y administración de usuarios y permisos.	-GNU/Linux -Windows -Mac OSX. -Unix -BSD -Solaris	-Hecho en PHP	-GPL	-Flexibilidad y Potencia. -Buen gestor de contenido. -Alto rendimiento, escalabilidad. -Método avanzado de clasificación de artículos. -Acceso a la web es muy rápido. -Permite a los usuarios configurar su propio perfil en dependencia de los permisos otorgados por el administrador.	-El panel de control de la administración no es óptimo. -Los foros son un poco desorganizados. -Es obligatorio cerrar sesión antes de salir del sitio.	1
Gestión de Proyectos /Control, gestión y seguimiento de errores/Salva automáticas/Control de versiones							
GForge	- Permite organizar y administrar gran cantidades de	-Linux	-PHP	-GPL	-Permite centralizar y homogeneizar la gestión de	- No ofrece compatibilidad directa con herramientas	11

	proyectos. - Compartición de la documentación. - Gestión de la planificación de tareas. - Descargas de archivos. - Control de versiones. -Salvas automáticas.				proyectos. -Es una página única. -Se consigue aumento de productividad. -Se tienen herramientas comunes a toda la empresa o departamento.	externas de gestión de proyectos. - No permite dar de alta, configurar, seguir y gestionar procesos propios de las empresas clientes. -No presenta herramienta para la certificación de calidad.	
DotProject	- Visualización de informes y estadísticas sobre los proyectos Registrados. - Foros de discusión para los usuarios registrados. - Sistema de gestión de incidencias. - Sistema de registro de ficheros. - Sistema de administración y configuración del portal	-LAMP -Windows -Mac OSX.	- Hecho en PHP.	-BSD -GPL	-Su instalación es muy sencilla y está bien documentada. -Existen filtros (por usuario y empresa) para visualizar las diversas tareas y proyectos registrados. -Es posible crear plantillas de proyecto, de forma que cuando se crea un nuevo proyecto se	- No es posible anidar proyectos. - No se incluye ningún tipo de soporte para llevar a cabo la gestión de los riesgos del proyecto. - Las diversas actualizaciones de versiones no garantizan la compatibilidad con las anteriores. -Mantenimiento de los planes de proyecto es	47

					copien todas las tareas de la plantilla anteriormente creada.	costoso.	
Modelado							
Rational Rose	- Permite Especificar, Analizar, Diseñar el sistema antes de Codificarlo. - Permite que la salida de una iteración sea la entrada de la próxima que está por venir. - Permite el Modelado Visual mediante UML	-Windows	-Puede generar código en Java, C++, Visual Basic, Ada, Corba y Oracle.		- Ingeniería Inversa. - Generador de Código. - Trabajo en Grupo. - Desarrollo Iterativo. - Genera los documentos automáticamente.		10
Visual Paradigm	- Permite realizar todo tipo de diagramas de clases. - Permite el Modelado Visual mediante UML. -Permite control de versiones. - Generar código desde diagramas.	-GNU/Linux -Windows	Es capaz de resistir varios lenguajes tanto de código como de ingeniería inversa tales como: Java, C + +, CORBA IDL, PHP, XML Schema,	-Gratuita y Comer-cial	- Soporta completamente el ciclo de vida del desarrollo de software. - Código inverso. - Generar código desde diagramas. - Generar documentación. - Cabe destacar		57

			Ada y Python		igualmente su robustez, usabilidad y portabilidad		
Gestor de Base de Datos							
PostgreSQL	-Permite la - Replicación asincrónica. -Soporta el uso de índices, reglas y vistas. -Permite la gestión de diferentes usuarios, como también los permisos asignados a cada uno de ellos. -Permite la declaración de funciones propias, así como la definición de disparadores. -Incluye herencia entre tablas (aunque no entre objetos, ya que no existen).	-Linux -Windows -Unix	- Tiene interfaces de programación originario de C / C + +, Java, Net, Perl, Python, Ruby, Tcl, ODBC, entre otros. - Se encuentra mayormente escrito en C.	-BSD	- Un excelente optimizador. -Incorpora una estructura de datos array. -Es altamente escalable. - Resiste el acaparamiento de grandes objetos binarios. -Implementa el uso de rollback's, subconsultas y transacciones, haciendo su funcionamiento mucho más eficaz. -Tiene la capacidad de comprobar la integridad referencial, así como también la de almacenar	-La velocidad de respuesta que ofrece con base de datos relativamente pequeñas puede parecer un poco deficiente. -Consume gran cantidad de recursos. -Es de 2 a 3 veces más lento que MySQL.	49

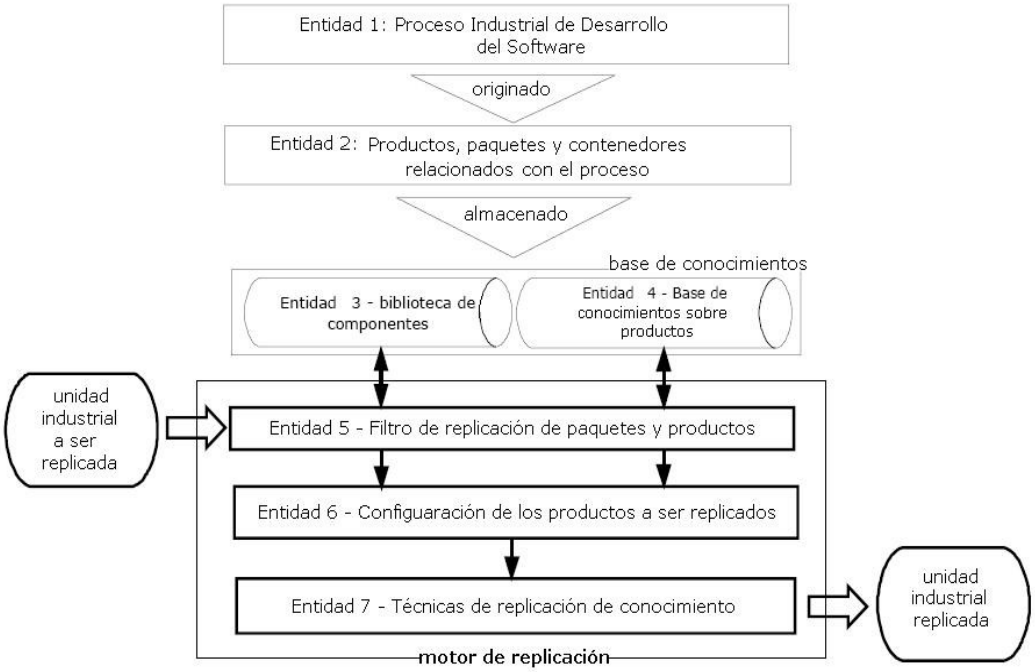
					procedimientos en la propia base de datos		
MySQL	-Permite administrar base de datos. -Gestión de usuarios y passwords, manteniendo un muy buen nivel de seguridad en los datos.	-Linux -Windows -Mac OS X	-Está escrito en una mezcla de C y C++. -Permite acceder a la base de datos a través de varios lenguajes, C, C++, C#, Pascal, Delphi, etc.	-GPL	-Conexión segura. - Multihilo y multiusuario. -Rapidez y facilidad de uso. -Excelente rendimiento -Gran portabilidad entre sistemas. -Facilidad de configuración e instalación. -Tiene una probabilidad muy reducida de corromper los datos.	-Carece de soporte para transacciones, rollback's y subconsultas. -No es viable para su uso con grandes bases de datos, a las que se acceda continuamente, ya que no implementa una buena escalabilidad. -El hecho de que no maneje la integridad referencial, hace de este gestor una solución pobre para muchos campos de aplicación.	16
Entorno de Desarrollo Integrado							
Net Beans	- Permite escribir, compilar, depurar y ejecutar programas.	-GNU/Linux -Windows -Mac OS X -Solaris	- Está escrito en Java. -Puede manejar múltiples lenguajes	- CDDL	- Contiene todos los módulos necesarios para el desarrollo de aplicaciones	-El proceso de compilación es un tanto más engorroso y lento.	8

			de programación Java, C++, PHP, Python, Ruby entre otros.		en Java. -Es multiplataforma. -Permite completamiento de código. -Facilidad de uso y de instalación.		
Zend Studio	-Permite agilizar el desarrollo web y permite simplificar proyectos complejos. -Funciones de depuración. -Seguimiento de variables y mensajes de error del intérprete de PHP. -Permite el autocompletado de código, ayuda de código y lista de parámetros de funciones y métodos de clase. -Permite Inserción automática de paréntesis y corchetes de cierre. -Permite el sangrado	-GNU/Linux -Windows -Mac OS X	-Está escrito en Java. -Es un completo IDE para el lenguaje de programación PHP	Comercial	-No requiere la instalación previa de PHP. -Detección de errores de sintaxis en tiempo real. -Soporte para control de versiones. -Ofrece soporte básico para lenguajes Web. -Soporte para navegación en bases de datos y ejecución de consultas SQL. -Soporte para gestión de grandes proyectos de desarrollo.	-Es propietario. -Requiere de conexión a Internet para registrar el producto. -Costo de la licencia. -Un poco complejo.	30

	automático y otras ayudas de formato de código.				-Cuenta con un buen Depurador		
Diseño Web							
Dreamweaver	- Desarrollo de páginas web.	- Windows -Mac	Lenguajes soportados : -HTML -XHTML -CSS -XML -JavaScript -AJAX -PHP -Adobe ColdFusion -ASP.Net -JSP	No libre	- Excelente manejo de tablas. - Excelente integración con PHP y apache. - Excelente administración de sitios. - Múltiples formas de editar el código HTML. -Genera código bastante limpio. -Creación de plantillas. -Permite crear páginas en varios lenguajes.	-Manejo de bases de datos lento, algo deficiente y a veces confuso -Poco intuitivo -Su costo es alto -Nada sencillo de utilizar para principiantes. Se torna lentísimo después de varias horas de uso. -Se Puede llegar a dañar el programa por las extensiones que se instalan.	8
Quanta Plus	- Desarrollo de páginas web	-GNU/Linux	-Es un Editor HTML basado en el código de Bluefish.	-GPL	- Coloreado de sintaxis para todos los lenguajes soportados y completamiento del código.	-Dificulta la rapidez de desarrollo. - No permite editar mientras se guarda un archivo.	-

			-Soporta una multitud de lenguajes como : - HTML -JavaScript -CSS -PHP -SQL y varios más		- Validador HTML integrado en la propia aplicación. - Soporte de extensiones/plugins para añadir funcionalidades extra. - La instalación es muy sencilla y es rápido. -Fácil de utilizar.		

Anexo 17: Modelo de replicación para una Factoría de Software



Anexo 18: Diseño de la encuesta realizada a los integrantes del proyecto Fábrica de Portales

Con este cuestionario se pretende identificar potencialidades y deficiencias en el proceso productivo.

Responde las siguientes preguntas y marcar con una x en el caso que haga falta:

1. Que sabes acerca del proyecto Fábrica de Portales:

_____
_____
_____
_____

2. Conoces el procedimiento que se sigue para elaborar un Portal:

Si ___ No ___

a) En caso afirmativo: explíquelo.

_____
_____
_____
_____

3. Diga cómo está estructurado el proyecto Portales:

_____

_____

4. Diga que deficiencias le encuentra al proyecto Portales:

_____
_____
_____
_____

5. Conoces el rol que desempeñas en el proyecto:

Si ___ No ___

a) En caso afirmativo responde las siguientes preguntas:

- ¿Conoces las funciones de tu rol? ¿Cuáles son?

_____

_____

- ¿Asumes la responsabilidad que te corresponde?
  Sí __ No ___ A veces ___ Nunca ____
- ¿Conoces las actividades asociadas a tu rol?
  Sí __ No ___
- ¿Conoces los artefactos generados por tu rol?
  Sí __ No ___

6. Se basan en alguna metodología para llevar a cabo el desarrollo de portales:

Sí __ No ___ No se ___

a) En caso afirmativo responde las siguientes preguntas:

- ¿Cuál es?

_____

- ¿Conoces los flujos de trabajo que están definidos dentro de tu proyecto?
  Sí __ No ___

7. Cuantas personas integran tu equipo de trabajo: _____

a) ¿Cómo es la comunicación entre los integrantes del equipo?

Buena ___ Mala___ Regular___ No se ___

b) Utilizan alguna técnica para planificar el trabajo tanto a nivel personal como en equipo.

No __ A veces __ Si__ Individual __ Colectivo __

c) ¿Cuáles son?

TSP___ PSP___ Ambas___

8. Qué herramientas se utilizan para el desarrollo de los portales:

Áreas	**¿Con cuáles herramientas trabajan?**	**¿La sabes utilizar?**	
		Si	**No**

Gestión de Proyectos			
Gestión de Contenido			
Diseño de páginas Web			
Control, gestión y seguimiento de errores			
Modelado			
Control de Versiones			
Gestión de las Base de Datos			
Salvas automáticas			
Entorno de desarrollo integrado			

9. ¿Los componentes realizados se almacenan en un Repositorio?

Sí____ No____

a) En caso afirmativo responde la siguiente pregunta:

- Son reutilizados estos componentes para el desarrollo de otros portales.

Sí____ No____ A veces____

10. ¿Se firma un acta como constancia del levantamiento de requisitos?

Sí_____ No_____ A veces____ Nunca____

11. Diga si tienen establecidas políticas de seguridad de la información

Si ___ No ___ No se ___

En caso afirmativo menciónelas: _____

Anexo 19: Diseño de la entrevista al Líder del Proyecto de Portales

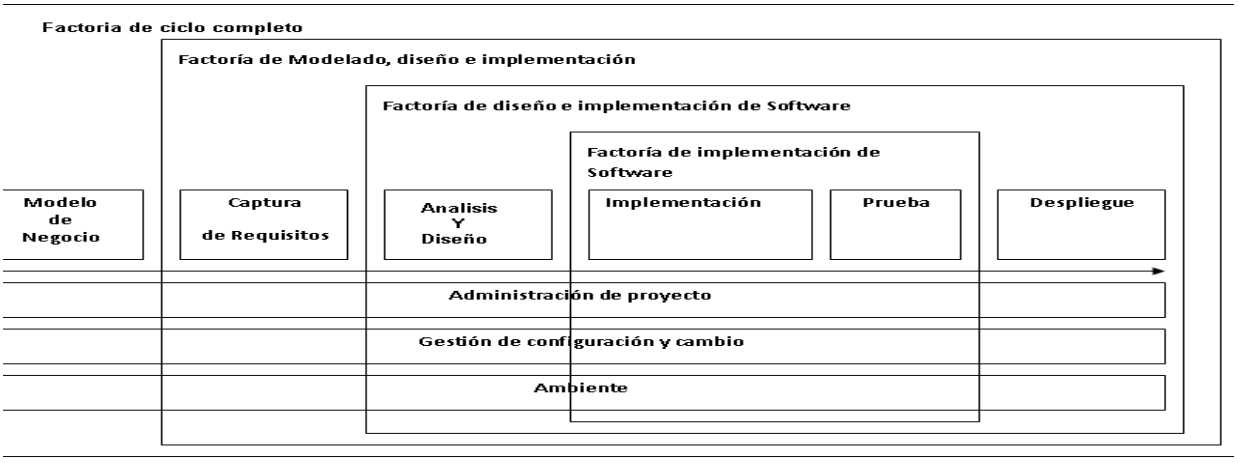
1. ¿Cuál es la función del proyecto?
2. ¿Qué deficiencias presenta actualmente el proyecto?
3. ¿Cómo están definidos los portales? Ejemplo: portal de noticias, portal de información etc.
4. ¿Cuál es la estructura directiva del proyecto?
5. ¿Cómo está diseñado la frecuencia de reuniones entre los directivos y los desarrolladores?
6. ¿Existe un cronograma de programación? Cual
7. ¿Con qué cantidad de recursos técnicos cuenta el proyecto?

8. ¿Con qué cantidad de recursos humanos cuenta el proyecto? Identificados por la actividad que realizan: diseñadores, programadores, etc.
9. ¿Se realiza un estudio de factibilidad al empezar cada proyecto?
10. ¿Se gestionan los costos, tiempo y esfuerzos?
11. ¿Se establece una fecha fija para la revisión y control de los plazos de entrega del trabajo del proyecto?
12. ¿Se firma un acta como constancia del levantamiento de requisitos?
13. ¿Cómo es la comunicación entre los directivos y los clientes?
14. ¿Se cumple con todas las funcionalidades del sistema definidas inicialmente?
15. ¿Se firman un acta memorando la aceptación de la documentación entregada al cliente?
16. ¿Se utiliza alguna metodología para la captura de requisitos y modelado del producto?
17. ¿Se tiene en cuenta la necesidad de la capacitación del personal?
18. ¿Qué políticas de seguridad tiene definido el proyecto?
19. ¿El proyecto tiene definida alguna estrategia de pruebas? Cual

Anexo 20: Diseño de la entrevista a los clientes

1. ¿Se firma algún documento que avale la captura de requisitos?
2. ¿Se cumple con todas las funcionalidades del sistema definidas inicialmente?
3. ¿Se sienten satisfecho con el trabajo realizado y los productos entregados?
4. ¿Califique la comunicación con el personal del proyecto?

Anexo 21: Clasificación de la factoría



Anexo 22: Tabla descriptiva del flujo de trabajo de Requisitos

Actividades	Actividades complementarias	Descripción	Roles	Artefactos
Diseñar la arquitectura de la información.	-Formular el proyecto.	-Se identifican las metas, objetivos y justificación de la aplicación Web.	-Líder de proyecto. -Analista. -Cliente.	-Arquitectura de la información.
	-Estudiar audiencia.	-Se identifica la audiencia, sus necesidades de información.		
	-Estudiar homólogos.	-Se realiza un estado del arte del producto, se investigan los sistemas similares y se identifica quien puede ser una competencia.		
	-Definir inventario de contenidos.	-Se definen los contenidos que se desean publicar en el portal y se identifican las relaciones entre los mismos.		
	-Definir taxonomía.	-Se agrupan los contenidos y se clasifican.		
	-Analizar procesos y servicios.	- Se analizan los procesos y servicios que deben estar presentes en el portal, identificando su funcionamiento y necesidades.		
	-Realizar mapa de navegación.	-Se realiza el sistema de navegación.		
	-Diseñar diagramas visuales.	-Se realizan las pantallas tipo.		
	-Definir etiquetas.	-Se describe cada contenido del portal.		
Captura de requisitos	-Análisis del entorno	-Se debe realizar un estudio detallado sobre el entorno en el que funcionará el sistema, las condiciones bajo las cuales se ejecutará el mismo, el alcance, los objetivos y lograr un entendimiento y/o familiarización por parte del equipo de desarrollo en cuanto a estos aspectos.		- Plan de gestión de Requisitos. -Proyecto técnico. - Glosario de términos.
	-Recolectar y clasificar los	- Lograr una especificación detallada de los requerimientos del portal, teniendo en cuenta		

	requerimientos	la clasificación de los requisitos que se plantea a continuación: - Requisitos de datos. - Requisitos de interfaz. - Requisitos navegacionales. - Requisitos de personalización. - Requisitos transaccionales o funcionales internos. - Requisitos no funcionales.		
Especificación de Requisitos	-Encontrar dependencias, etiquetar y priorizar los requerimientos	-Se debe analizar la definición inicial o informal de los requisitos para poder establecer las dependencias que existen entre ellos; asignar un identificador único que permita rastrearlo en todo momento y establecer niveles de prioridad en función de las peticiones del cliente.		- Documento de Especificación de requisitos.
	- Especificar los requerimientos según su clasificación	-Elaborar una descripción formal de los requisitos, clasificándolos según los tipos de requisitos vistos con anterioridad; realizarle revisiones al documento en conjunto con el equipo de desarrollo y el cliente, en caso de ser necesario, para obtener una especificación formal.		
Validación de Requisitos	-Verificar la calidad en la especificación de los requerimientos	-A través de la aplicación de técnicas realizar revisiones constantes a la definición de los requisitos.		-Solicitud de cambio. -Pedido de cambio.
	-Realizar la trazabilidad de los requerimientos	-Llevar la matriz de trazabilidad. Se puede evaluar a través de la revisión del cumplimiento de los objetivos del portal.		
	-Resolver conflictos de cambios en los requerimientos	- Se deberá registrar la petición de cambio. Llevar un documento de Control de cambios.		
Encontrar actores y casos de uso	-Encontrar los actores.	- Esbozar quienes interactuaran con el sistema.		- Casos de uso. -Modelo de dominio. -Modelo de casos
	-Encontrar los casos de uso.	-Se identifican los casos de uso que darán soporte al trabajo de cada actor.		

	- Realizar Modelo de casos de uso.	-Se establece qué actores interactuarán con cada caso de uso.		de uso del Sistema.
	-Agrupar en paquetes actores y casos de uso.	-Utilizar los paquetes para simplificar el modelo de casos de uso. (Opcional).		
	-Detallar casos de uso.	-Se describe el flujo de sucesos en detalle.		
	-Priorizar los casos de uso.	- Se priorizan los casos de uso de acuerdo a las necesidades del cliente.		
	-Capturar y definir un glosario de términos.	-Los términos deben ser comunes para todos.		

Anexo 23: Tabla descriptiva del flujo de trabajo de Arquitectura y Diseño

Actividades	Actividades complementarias	Descripciones	Roles	Artefactos
Definir la arquitectura.			Arquitecto.	-Arquitectura de Software. -Modelo de despliegue.
Diseño de componentes	-Diseñar clases de diseño.	-En esta tarea se debe identificar las clases del diseño, para esto se deben estudiar la funcionalidad y los requisitos descritos.	Diseñador de componentes.	-Modelo de diseño - Subsistema del Diseño.
	-Diseñar subsistemas.	-Se utiliza para encapsular comportamientos similares.		
Diseñar la Base de datos.			Diseñador de Base de datos.	-Modelo de datos.

Anexo 24: Tabla descriptiva del flujo de trabajo de Implementación y prueba

Actividades	Actividades complementarias	Descripciones	Roles	Artefactos
Estructurar el modelo de implementación.			-Arquitecto.	-Modelo de implementación. -Subsistemas de implementación.
Desarrollar plan de integración.			-Arquitecto.	-Plan de Integración.
Implementar componentes.			-Implementador.	-Código fuente. -Manual de usuario.
Implementar la Base de datos.			-Implementador de Base de datos.	-Base de datos.
Integrar subsistemas.			-Arquitecto.	-Producto.
Integrar sistema.			-Arquitecto. -Implementador.	
Planificar evaluaciones.	-Realizar el Plan de prueba.	- Contiene información sobre los objetivos generales y específicos de las pruebas en el proyecto, así como las estrategias y recursos con que se dotará a esta tarea.	-Probador	-Plan de prueba. -Listas de chequeo. -Plan aseguramiento de la calidad.
	-Realizar casos de prueba.	-Se ejecutan con el objetivo de verificar si el producto satisface los requerimientos del usuario, tal y como se describe en las especificación de los requerimientos y si el producto se comporta como se desea, según lo descrito en el diseño.		-Casos de prueba.
	-Realizar el diseño de prueba.	-Se diseñan las pruebas a realizarse.		-Diseño de casos de prueba.
Ejecutar evaluaciones.	-Realizar pruebas de integración y unitarias.	- Realizar pruebas de integración y unitarias.		- Registro de prueba Unitaria o Integración.
	-Evaluar prueba.	- Se evalúan los resultados de las pruebas comparando los resultados con los objetivos esbozados en el plan de prueba.		- No Conformidades.

Anexo 25: Tabla descriptiva del flujo de trabajo de Despliegue

Actividades	Actividades complementarias	Descripciones	Roles	Artefactos
Desarrollar el plan de despliegue.			-Administrador de despliegue.	-Plan de despliegue. -Acta de Aceptación. -Acta de Entrega. -Acta de Terminación de Proyecto. -Producto.

Anexo 26 Comparación entre los organigramas propuestos por Pressman

Aspectos	Organigramas para equipos de desarrollo		
	Descentralizado democrático	Descentralizado controlado	Centralizado controlado
Descripción	No existe un jefe permanente, se asignan coordinadores de tareas a corto plazo	Existe un jefe bien definido y jefes de equipos con responsabilidades sobre subtareas.	Existe un jefe bien definido que controla los problemas a alto nivel y también las decisiones internas.
	Las decisiones se toman por consenso del grupo	Las decisiones se toman por consenso del grupo pero las actividades y responsabilidades se asignan a los subgrupos	
Complejidad del problema	Buenos resultados en problemas complejos	Buenos resultados en problemas complejos	Buenos resultados en problemas sencillos
Tamaño	Proyectos pequeños	Proyectos grandes	Proyectos grandes
Tiempo de trabajo en equipo	Mejores resultados	Mejores resultados	Con frecuencia aparece cansancio y otros elementos que aumentan la presión y disminuyen la productividad.
Modularidad	Baja	Alta	Alta
Calidad	Baja	Mejor	Mejor

Anexo 27 Estructura de dirección de la factoría

Anexo 28 Conocimientos, habilidades y valores de los roles

Nombre del rol	Responsabilidades	Competencias		
		Habilidades	Sistema de conocimientos.	Valores
Gestor de la factoría	-Estima plazos, costos y riesgos para la realización de un determinado producto software. -Controla el desarrollo de los distintos productos, verifica y tiene noción	-Capacidad para coordinar y dirigir a los integrantes de la factoría. -Habilidades de mando y buenas relaciones con el personal. - Motivar a los trabajadores.	-Hardware y software utilizado en la factoría. -Técnicas de dirección y negociación.	-Tolerancia -Honestidad -Responsabilidad -Organización -Disciplina

	de la fase en que se encuentran. -Establece relaciones con los clientes. -Administra con los distintos recursos y procesos de la factoría. -Organiza el trabajo de forma tal que las personas bajo su mando cumplan su rol apropiadamente.	-Administrar y planificar recursos.		
Director de calidad	-Dirige la gestión de la calidad. -Vela porque los productos se desarrollen con la calidad requerida. -Coordina la realización de pruebas, revisiones y auditorias a los diferentes productos.	-Comunicación. -Dotes de mando y de relaciones interpersonales. -Capacidad para dirigir.	-Normas de calidad existentes. -Lenguajes de programación utilizados. -Herramientas.	-Responsabilidad -Exigencia -Disciplina -Buenas relaciones personales. -Tolerancia -Empatía -Honestidad

Gestor de desarrollo	-Controla el desarrollo de los distintos productos, verifica y tiene noción de la fase en que se encuentran. -Organiza el trabajo de forma tal que las personas bajo su mando cumplan su rol apropiadamente. - Tiene la tarea de velar por que los productos se realicen a tiempo y según las especificaciones establecidas por el cliente.	-Capacidad para coordinar y dirigir al personal bajo su mando. - Motivar a los trabajadores. -Comunicación. -Habilidades de mando y buenas relaciones con el personal.	-Técnicas de dirección y negociación. -Herramientas de desarrollo. -Lenguajes de programación.	-Tolerancia -Honestidad -Responsabilidad -Organización -Disciplina -Exigencia
Administrador de repositorio	-Crear y configurar el ambiente para la gestión de la gestión de configuración. -Administrar el repositorio del proyecto. -Establecer los mecanismos de control de cambio.	-Administrar y organizar los componentes del repositorio.	-Herramientas de gestión de configuración. -Proceso de control de cambio.	-Organización. -Responsabilidad. -Disciplina. -Honestidad. -Exigencia.
Investigador	-Realizar búsqueda sobre temas de interés para la factoría. -Investigar sobre las tecnologías nuevas que pueda utilizar la	-Poder de análisis. -Habilidades de comunicación. -Poder de solución.	-Metodologías de la investigación. - Herramientas de trabajo de la factoría. -Desarrollo de SW en el	-Responsabilidad. -Organización. -Exigencia.

	factoría. -Observar el desarrollo de software a nivel mundial.		Mundo.	
Especialista de tecnología	-Instalar las herramientas de desarrollo. - Verificar el funcionamiento de los puestos de trabajo. -Controlar tanto el hardware como el software.	-Habilidades con las herramientas. -Conocimientos amplios sobre la instalación de las herramientas.	-Conocimientos sobre todas las herramientas utilizadas en la factoría.	-Responsabilidad -Organización -Disciplina.

Anexo 29 Cuaderno del ingeniero

Se debe numerar cada página, para tenerlas en orden y garantizar el registro legal del trabajo. Se deben registrar las notas en orden cronológico y no se podrá insertar o eliminar páginas. En la portada se debe etiquetar el cuaderno con un número para ordenarlos cronológicamente y referenciarlos al ser numerosos. Debe tener además nombre, número de teléfono, dirección de correo electrónico, fecha de comienzo de escritura de los datos y fecha del cierre. Por ejemplo:

Cuaderno número: 1

Cuaderno de Ingeniería

Fábrica de Portales de la Facultad 10

Nombre del Ingeniero: Ailet Bagarotti Pérez

Teléfono/ correo electrónico: abperez@estudiantes.uci.cu

Fecha de Apertura: 24/03/2009 **Fecha de Cierre:**

Dentro del cuaderno se deben utilizar las dos primeras páginas como índice de contenidos y cada página debe ser numerada. A continuación se muestra el ejemplo del cuaderno contenido de ingeniería

Contenido del cuaderno de ingeniería		**1**
Página	**Tema**	**Fechas**

Anexo 30 Cuaderno de registro de tiempo

El cuaderno en su cabecera debe suministrar los datos referentes al nombre del trabajador de la factoría ya sea estudiante o profesor, la fecha de inicio, y el nombre o número de clase. Cada período de tiempo se introduce una línea de la siguiente forma.

- Fecha. La fecha de realización de alguna actividad.
- Comienzo. La hora de comienzo de alguna actividad.
- Fin. La hora en que terminas de hacer una actividad.
- Interrupción. Cualquier pérdida de tiempo, debido a interrupciones.
- Δ Tiempo. El tiempo dedicado a cada actividad en minutos.
- Actividad. Nombre descriptivo para la actividad.
- Comentarios. Una descripción más completa de lo que se está haciendo.
- C (Completado). Rellena esta columna cuando termines una tarea.
- U (Unidades). El número de unidades de una tarea acabada.

A continuación se puede observar un ejemplo:

Estudiante: _____ Fecha: _____

Profesor: _____

Ejemplo de cuaderno de registro de tiempo.

Fecha	Comienzo	Fin	Interrupción	∆Tipo	Actividad	Comentario	C	U

Anexo 31 Resumen semanal de actividades

El cuaderno en su cabecera debe tener el nombre del trabajador y el rango de fecha de la semana. Dentro de las actividades se debe encontrar el número secuencial de cada una de ellas y la descripción de la

actividad. Además de la definición de las horas usadas en los días de la semana. Finalmente los totales de tiempos por actividad y de tiempo trabajados diariamente.

Nombre: _____ Fecha: _____

Actividades		**Número de horas**							**I**
No.	**Descripción**	**L**	**M.**	**Mi.**	**J.**	**V.**	**S.**	**D.**	**Total**
Total de horas.									

Anexo 32 Cuaderno de registro de defectos

En la cabecera introduce los siguientes datos: nombre del trabajador de la factoría, fecha y numeración del componente.

Dentro de las instrucciones para el cuaderno de registro de defectos: se encuentra:

- Fecha: Anotar la fecha en la que se encontró el defecto.
- Número: Número para cada defecto.
- Tipo: Anota el tipo de defectos según la lista de tipos de defectos de la tabla.
- Introducido: Anotar la fase en la que se introdujo el defecto.
- Eliminado: Fecha en la que se eliminó el defecto.
- Tipo de corrección: Estima o mide el tiempo necesario para encontrar y corregir el defecto.
- Defecto Corregido: Se puede ignorar esta casilla la primera vez.
- Descripción: Escribe brevemente la descripción de un defecto.

En la siguiente tabla, se muestra el Cuaderno de Registros de Defectos.

Nombre del trabajador: _____

Fecha: _____

Del Componente: _____

Fecha	Numero	Tipo	Introducido	Eliminado	Tiempo de corrección	Defecto corregido
Descripción:						
Fecha	**Numero**	**Tipo**	**Introducido**	**Eliminado**	**Tiempo de corrección**	**Defecto corregido**
Descripción:						

Anexo 33 Catálogo de componentes

Plantilla del componente			
Nombre:			
Descripción:			
Grupo	**Código _____**	**Activos_____**	
Tipo:			
Fecha de creación	**Día:**	**Mes:**	**Año:**
Ubicación:			

Anexo 34 Políticas de Seguridad de la Información (PSI)

- Instalar un antivirus y mantener el mismo actualizado.
- Instalar programas diseñados para prevenir el acceso no autorizado como Firewall o cualquier sistema seguro para controlar los puertos de su sistema.
- Usar claves de acceso que no estén asociadas a datos comunes del usuario y tengan más de 6 caracteres incluyendo números y símbolos.
- Cambiar la clave de acceso constantemente.
- No ejecutar en las estaciones de trabajo ningún archivo contenido en un mensaje de correo no solicitado o enviado por un remitente desconocido.
- Verificar frecuentemente los softwares instalados.

- Configurar los navegadores desactivando la ejecución automática de estos contenidos.
- Se debe dar acceso a la información teniendo en cuenta las tareas que le corresponden a casa usuario.
- En el caso del producto en desarrollo se tendría una copia local en todas las estaciones de trabajo de las personas que tengan que ver con el mismo y una copia centralizada en el Subversion (SVN).
- En el traspaso de información al cliente es a través de un servidor FTP.

Anexo 35 Listas de Chequeo

Listas de chequeo para Requisitos

1. ¿Existe correspondencia entre el planteamiento del problema y los requerimientos?
2. ¿Los requisitos están escritos en lenguaje no técnico, comprensible para el usuario?
3. ¿Todos los requisitos que describen el mismo objeto utilizan la misma terminología?
4. ¿Cada requisito tiene una única interpretación?
5. ¿Todos los requisitos no funcionales están bien definidos?
6. ¿Los nombres de los casos de uso están en infinitivo y reflejan de manera clara el objetivo del usuario sobre el sistema?
7. ¿Están clasificado los casos de uso que definen la arquitectura básica del sistema?
8. ¿Se ha identificado los casos de uso que darán soporte y mantenimiento al sistema?
9. ¿Se ha descrito con precisión todas las alternativas o excepciones?
10. ¿El nombre del caso de uso es único?
11. ¿El resumen dice como se inicia, como termina y las operaciones principales que realiza el caso de uso?
12. ¿Se escribe una precondición si y solo si a partir de la ocurrencia de un suceso determinado comienza el caso de uso?
13. ¿La poscondición plasma cambios que suceden en el sistema al terminarse de ejecutar el caso de uso?
14. ¿Se especifica la complejidad del caso de uso?
15. ¿Está descrito el caso de uso en presente?
16. ¿El Caso de Uso está relacionado con al menos un actor?

17. ¿Si el caso de uso es abstracto (include, extend, generalización-especialización), no lo inicializa ningún actor?
18. ¿No existen abreviaturas?
19. ¿En todos los CU que se introducen datos tienen un flujo alterno donde el sistema valida la integridad de los datos que se introducen y muestra un mensaje en caso de que los datos estén incompletos?
20. ¿En la sección flujos alternativos se describen todas las excepciones que existan por muy evidentes que parezcan?
21. ¿Al describir el caso de uso base se mencionan todos los casos de Uso que Extienden, se incluyen o se generalizan del Caso de Uso?
22. ¿La navegabilidad en los caso de uso de inclusión se inicia desde el caso uso base hasta el caso de uso incluido?
23. ¿La navegabilidad en los caso de uso de extensión se inicia desde el caso uso extendido hasta el caso de uso base?
24. ¿La navegabilidad en la generalización/especialización se inicia desde el caso de uso especializado a al generalizado y se representa con una relación de herencia?
25. ¿Las relaciones de inclusión y extensión entre los caso de uso se han representado con línea discontinua?
26. ¿El diagrama de casos de uso expresa en detalles y claramente lo que debe hacer el sistema?
27. Si la modelación de las interacciones con el sistema es muy extensa ¿ha empleado los paquetes de caso de uso?

Listas de chequeo para la Arquitectura y Diseño

1. ¿Se identifican todas las clases necesarias para el diseño?
2. ¿Se definen las clases con un nombre identificativo?
3. ¿Se relacionan correctamente todas las clases del diseño?
4. ¿En el diagrama de clases del diseño se tiene en cuenta el lenguaje de programación a utilizar?
5. ¿Están definidos todos los métodos de las clases?
6. ¿Están bien declarados los atributos de las clases?
7. ¿El modelo de diseño describe la realización física de los casos de uso del sistema?

8. ¿Se utilizan los subsistemas de diseño para organizar los artefactos del modelo de diseño en piezas más manejables?
9. ¿Se encuentran bien definidas las clases persistentes?

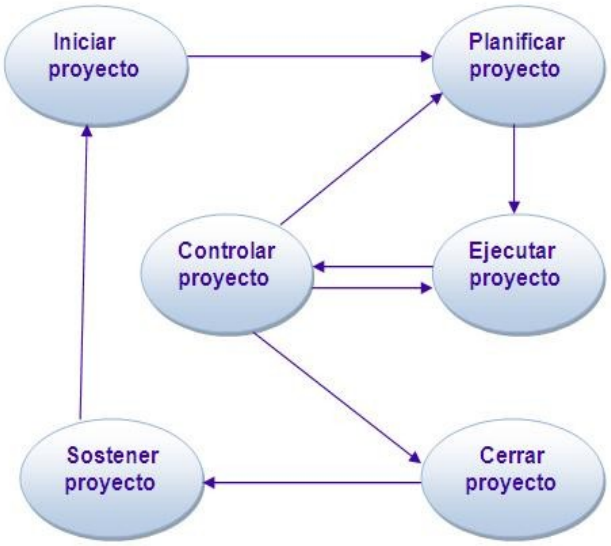
Listas de chequeo para Implementación y prueba

1. ¿Se encuentran definidos los componentes significativos para el desarrollo del producto?
2. ¿Se reutilizan los componentes?
3. ¿Los componentes están bien diseñados?
4. ¿Está bien definido el orden en el que se van a integrar los componentes?
5. ¿Están codificadas todas las clases del diseño?
6. ¿El código está debidamente comentado?
7. ¿Se generan todos los artefactos correspondientes a este flujo?
8. ¿Funciona correctamente el proceso de autenticación de usuario?
9. ¿Se impide la entrada en la aplicación cuando se introduce un usuario que no tiene permiso de acceso para esa aplicación?
10. ¿Aparece un mensaje de error descriptivo al introducir un usuario o una contraseña incorrecta?
11. ¿Funciona correctamente la navegación por los diferentes módulos de la aplicación?
12. ¿Funciona correctamente la tabulación?
13. ¿Los campos deshabilitados no pueden tomar el cursor ni con el ratón ni por tabulación?
14. ¿Se produce un mensaje de error al no rellenar un campo obligatorio?
15. Si la aplicación es para Internet, ¿se puede navegar en ella utilizando Mozilla Firefox?
16. ¿Se evitan las faltas de ortografía, incluidos los acentos?
17. ¿Se probaron todas las entradas a la aplicación?
18. ¿En la aplicación aparecen todas las funcionalidades especificadas en la documentación?
19. ¿Se entienden la interfaz y el contenido de la aplicación?
20. ¿Se presenta al usuario solo la información que necesita?

Listas de chequeo para el Despliegue

1. ¿Se desarrolla correctamente el plan de despliegue?
2. ¿Se tienen en cuenta los errores encontrados en la instalación y prueba del producto?
3. ¿Se le entrega a los clientes el manual de usuario?
4. ¿El producto entregado satisface todas las necesidades del cliente?

Anexo 36 Flujos de procesos de la Gestión de proyectos



Anexo 37: Descripción general de la Gestión del Alcance del proyecto

Procesos	Entradas	Herramientas y Técnicas.	Salidas
Planificación del Alcance	-Factores ambientales de la empresa. -Activos de los procesos de la organización. -Acta de constitución del proyecto. -Enunciado del alcance preliminar. -Plan de gestión del proyecto.	-Juicio de expertos. -Plantillas, formularios y normas.	-Plan de gestión del alcance del proyecto.
Definición del Alcance	- Acta de constitución del proyecto. -Activos de los procesos de la organización.	-Análisis de los interesados. - Análisis del producto. -Juicio de expertos -Identificación de	-Enunciado del alcance del proyecto. -Cambios solicitados. -Plan de gestión del alcance del

	-Enunciado del alcance preliminar. -Plan de gestión del alcance del proyecto. -Solicitudes de cambio aprobadas.	alternativas	proyecto (actualizado).
Crear la Estructura de Desglose del Trabajo (EDT)	-Activos de los procesos de la organización. -Enunciado del alcance del proyecto. -Plan de gestión del alcance del proyecto. -Solicitudes de cambio aprobadas.	- Descomposición. -Plantillas de la estructura de desglose del trabajo.	-Plan de gestión del alcance del proyecto (actualizado). -Enunciado del alcance del proyecto. (actualizado) -Estructura de desglose del trabajo. -Diccionario de la EDT. -Línea base del alcance. -Solicitudes de cambio.
Verificación del Alcance	-Enunciado del alcance del proyecto. -Diccionario de la EDT. -Plan de gestión del alcance del proyecto. - Productos entregables.	-Inspección.	- Productos entregables aceptados. -Acciones correctivas recomendadas. -Cambios solicitados.
Control del Alcance	-Enunciado del alcance del proyecto. -Estructura de desglose del trabajo. -Diccionario de la EDT. -Plan de gestión del alcance del proyecto. -Reportes de desempeño. -Informes de rendimiento. -Solicitudes de cambio aprobadas. -Información sobre el rendimiento del trabajo.	-Sistema de control de cambios. - Análisis de variación. - Replanificación. - Sistema de gestión de la configuración.	-Plan de gestión del proyecto (actualizado). -Enunciado del alcance del proyecto (actualizado) -Estructura de desglose del trabajo (actualizado). -Diccionario de la EDT (actualizado). -Línea base del alcance (actualizado). -Acciones correctivas recomendadas. -Cambios solicitados. -Activos de los procesos de la organización

Anexo 38: Descripción general de la Gestión de Tiempo del proyecto

Procesos	Entradas	Herramientas y Técnicas	Salidas
Definición de las Actividades	-Factores ambientales de la empresa. -Activos de los procesos de la organización. -Enunciado del alcance del proyecto. -Estructura de desglose del trabajo. -Diccionario de la EDT -Plan de gestión del proyecto.	- Descomposición. - Juicio de expertos. - Plantillas. -Planificación gradual. -Componente de planificación.	- Lista de actividades. -Atributos de la actividad. -Lista de hitos. -Cambios solicitados.
Establecimiento de la Secuencia de las Actividades	-Enunciado del alcance del proyecto. - Lista de actividades. -Atributos de la actividad. -Lista de hitos. -Solicitudes de cambio aprobadas.	-Plantillas de red del cronograma. -Determinación de dependencias. -Método de diagramación por precedencia (PDM). -Método de diagramación por Flechas (ADM). -Aplicación de adelantos y retrasos.	- Diagramas de red del cronograma del proyecto. -Lista de actividades (actualizado). - Atributos de la actividad (actualizado). -Cambios solicitados (actualizado).
Estimación de Recursos de las Actividades	-Factores ambientales de la empresa. -Activos de los procesos de la organización. -Lista de actividades. -Atributos de la actividad. -Disponibilidad de recursos. -Plan de gestión del	- Juicio de expertos. -Análisis de alternativas. -Datos de estimaciones publicados. -Software de gestión de proyectos. -Estimación ascendente.	- Requisitos de recursos de las actividades. - Atributos de la actividad (actualizado). -Estructura de desglose de recursos. -Calendarios de recursos (actualizado) -Cambios solicitados.

	proyecto.		
Estimación de la Duración de las Actividades	-Factores ambientales de la empresa. -Activos de los procesos de la organización. -Enunciado del alcance del proyecto. - Lista de actividades. -Atributos de la actividad. - Requisitos de recursos de las actividades. -Calendarios de recursos. -Plan de gestión del proyecto. --Registro de riesgos. --Estimaciones de costes de las actividades.	- Juicio de expertos. - Estimación por analogía. -Estimación paramétrica. - Estimaciones por tres valores. -Análisis de reserva.	- Estimativos de la duración de la actividad. - Atributos de la actividad (actualizado).
Desarrollo del Cronograma	-Activos de los procesos de la organización. -Enunciado del alcance del proyecto. - Lista de actividades. -Atributos de la actividad. - Diagramas de red del cronograma del Proyecto. -Requisitos de recursos de las actividades. -Calendarios de recursos.	-Análisis de la red del cronograma. -Compresión del cronograma. -Aplicación de Calendarios. -Método del camino critico. -Nivelación de recursos. -Método de cadena critica. -Software de gestión de proyectos. -Aplicación de	-Cronograma del proyecto. -Datos del modelo de cronograma. -Línea base del cronograma. - Requisitos de recursos (actualizado). - Atributos de la actividad (actualizado). - Calendario del proyecto (actualizado). -Cambios solicitados. -Plan de gestión del proyecto (actualizado). --Plan de gestión del cronograma (actualizado).

	- Estimaciones de la duración de la actividad. -Plan de gestión del proyecto. --Registro de riesgos.	adelantos y retrasos. -Modelo de cronograma.	
Control del Cronograma	-Plan de gestión del cronograma. -Línea base del cronograma. -Informes de rendimiento. -Solicitudes de cambio aprobadas.	-Medición del Rendimiento. -Diagramas de barras de comparación del cronograma. -Sistema de control de cambios del cronograma. -Informe del avance. -Software de gestión de proyectos. -Análisis de variación.	-Datos del modelo de cronograma (actualizado). -Línea base del cronograma (actualizado). -Mediciones del rendimiento. -Acciones correctivas recomendadas. -Cambios solicitados. -Activos de los procesos de la organización (actualizado). -Lista de actividades (actualizado). -Atributos de la actividad (actualizado). -Plan de gestión del proyecto (actualizado).

Anexo 39: Descripción general de la Gestión de Costos del proyecto

Procesos	Entradas	Herramientas y Técnicas	Salidas
Estimación de Costes	-Factores ambientales de la empresa. -Activos de los procesos de la organización. -Enunciado del alcance del proyecto. -Estructura de desglose del trabajo. -Diccionario de la EDT -Plan de gestión del proyecto. --Plan de gestión del cronograma. -- Plan de gestión de personal. --Registro de riesgos.	-Estimación de costos. -Estimación por analogía. -Estimación paramétrica. -Estimación ascendente. -Software de gestión de proyectos. -Análisis de reserva. -Coste de calidad. -Análisis de propuestas para licitaciones.	-Estimación de costes de las actividades. -Información de respaldo de la estimación de costes de las actividades. -Plan de gestión de costes (actualizado). -Cambios solicitados.
Preparación del Presupuesto de Costes	-Enunciado del alcance del proyecto. -Estructura de desglose del trabajo. -Diccionario de la EDT -Estimación de costes de las actividades. -Información de respaldo de la estimación de costes de las actividades. -Cronograma del proyecto. -Calendarios de recursos. - Contratos. -Plan de gestión de costes.	-Suma de costes. -Estimación paramétrica. -Análisis de reserva. -Conciliación del límite de la financiación.	- Línea base de coste. -Plan de gestión de costes (actualizado). - Cambios solicitados. -Requisitos para la financiación del proyecto.

Control de Costes	- Línea base de coste. -Requisitos para la financiación del proyecto. -Informes de rendimiento. -Información sobre el rendimiento del trabajo. -Plan de gestión del proyecto. -Solicitudes de cambio aprobadas.	-Sistema de control de cambios del coste. -Análisis de medición del rendimiento. -Proyecciones. -Revisiones del rendimiento del proyecto. -Software de gestión de proyectos. -Gestión de variación.	-Estimación de costes (actualizado). - Línea base de coste (actualizado). -Mediciones del rendimiento. -Conclusión proyectada. -Cambios solicitados. -Acciones correctivas recomendadas. -Activos de los procesos de la organización (actualizado). -Plan de gestión del proyecto (actualizado).

Anexo 40: Descripción general de la Gestión de Calidad del proyecto

Procesos	Entradas	Herramientas y Técnicas	Salidas
Planificación de la Calidad	-Factores ambientales de la empresa. -Activos de los procesos de la organización. -Enunciado del alcance del proyecto. -Plan de gestión del proyecto.	-Análisis coste-beneficio. -Estudios comparativos. -Diseño de experimentos. -Coste de la calidad. -Herramientas adicionales de la planificación de la calidad.	- Plan de gestión de la calidad. -Métricas de calidad. -Listas de control de calidad. -Plan de mejoras del proceso. -Línea base de calidad. -Plan de gestión del proyecto (actualizado).
Realizar Aseguramiento de la Calidad	- Plan de gestión de la calidad. -Métricas de calidad. -Plan de mejoras del proceso. -Información sobre el rendimiento del trabajo. -Solicitudes de cambio aprobadas.	-Herramientas y técnicas para la planificación de la calidad. -Auditorías de calidad. -Análisis del proceso. -Herramientas y técnicas para el control de calidad.	-Cambios solicitados. -Acciones correctivas recomendadas. -Activos de los procesos de la organización (actualizado). -Plan de gestión del proyecto (actualizado).

	-Mediciones de control de calidad. -Solicitudes de cambio implementadas. -Acciones correctivas implementadas. -Reparación de defectos implementada. -Acciones preventivas implementadas.		
Realizar Control de Calidad	- Plan de gestión de la calidad. -Métricas de calidad. -Listas de control de calidad. -Activos de los procesos de la organización. -Solicitudes de cambio aprobadas. -Información sobre el rendimiento del trabajo. -Productos entregables.	-Diagrama de causa y efecto. - Diagramas de control. - Diagramas de flujo. -Histograma. - Diagrama de Pareto. - Diagrama de comportamiento. - Diagrama de dispersión. -Muestreo estadístico. -Inspección. -Revisión de reparación de defectos.	-Mediciones de control de calidad. - Reparación de defectos validada. -Línea base de calidad (actualizado). -Acciones preventivas recomendadas. -Acciones correctivas recomendadas. -Reparación de defectos recomendada. -Cambios solicitados. -Activos de los procesos de la organización (actualizado). -Productos entregables validados. -Plan de gestión del proyecto (actualizado).

Anexo 41: Descripción general de la Gestión de Riesgo del proyecto

Procesos	Entradas	Herramientas y Técnicas	Salidas
Planificación de la Gestión de	-Factores ambientales de	-Reuniones y análisis de	-Plan de gestión de riesgos.

Riesgos	la empresa. -Activos de los procesos de la organización. -Enunciado del alcance del proyecto. -Plan de gestión del proyecto.	planificación.	
Identificación de Riesgos	-Factores ambientales de la empresa. -Activos de los procesos de la organización. -Enunciado del alcance del proyecto. -Plan de gestión de riesgos. -Plan de gestión del proyecto.	-Revisiones de documentación. -Técnicas de recopilación de información. -Análisis de listas de control. -Análisis de asunciones. -Técnicas de diagramación.	-Registro de riesgos.
Análisis Cualitativos de Riesgos	-Activos de los procesos de la organización. -Enunciado del alcance del proyecto. -Plan de gestión de riesgos. -Registro de riesgos.	-Evaluación de probabilidad e impacto de los riesgos. -Matriz de probabilidad e impacto. -Evaluación de la calidad de los datos sobre riesgos. -Categorización de riesgos. -Evaluación de la urgencia del riesgo.	-Registro de riesgos (Actualizado).
Análisis Cuantitativo de Riesgos	-Activos de los procesos de la organización. -Enunciado del alcance del proyecto. -Plan de gestión de riesgos. -Registro de riesgos.	-Técnica de recopilación y representación de datos. -Técnicas de análisis cuantitativo de riesgos y de modelado.	-Registro de riesgos (Actualizado).

	-Plan de gestión del proyecto. -- Plan de gestión del cronograma del proyecto. -- Plan de gestión de los costes del proyecto.		
Planificación de la Respuesta a los Riesgos	-Plan de gestión de riesgos. -Registro de riesgos.	-Estrategias para riesgos negativos o amenazas. -Estrategias para riesgos positivos u oportunidades. -Estrategia común ante amenazas y oportunidades. -Estrategia de respuesta para contingencias.	-Registro de riesgos (Actualizado). -Plan de gestión del proyecto (actualizado). -Acuerdos contractuales relacionados con el riesgo.
Seguimiento y Control de Riesgos	-Plan de gestión de riesgos. -Registro de riesgos. -Solicitudes de cambio aprobadas. -Información sobre el rendimiento del trabajo. -Informes de rendimiento.	-Reevaluación de los riesgos. -Auditorías de los riesgos. -Análisis de variación y de tendencias. -Medición del rendimiento técnico. -Análisis de reserva. -Reuniones sobre el estado de la situación.	-Registro de riesgos (Actualizado). -Plan de gestión del proyecto (actualizado). -Activos de los procesos de la organización (actualizado). -Acciones preventivas recomendadas. -Acciones correctivas recomendadas. -Cambios solicitados.

Anexo 42: Lista de riesgos

Riesgos	Estrategia de mitigación	Plan de contingencia
Falta de entendimiento entre el líder y los miembros de la factoría.	-Realizar reuniones sistemáticas con los desarrolladores donde el líder les explique y les chequee las tareas que cada uno debe desarrollar.	-Asignar jefes de equipos capacitados para dirigir y orientar a los desarrolladores en ausencia del líder del proyecto.
Falta de entendimiento entre los clientes y los miembros de la factoría.	-Realizar reuniones sistemáticas con los clientes para que sean partícipes del desarrollo del producto.	-Asignar un responsable encargado de coordinar las reuniones y atender a los clientes.
Cambio constante de los requerimientos.	-Realizar un buen proceso de captura de requisitos. -Verificar la viabilidad de los requerimientos con el cliente.	-Realizar reuniones más a menudo con los clientes.
Inexperiencia del trabajo en equipo.	-Impartir cursos de capacitación sobre la herramienta TSP. -Crear un buen ambiente de comunicación para el trabajo en equipo.	-Organizar reuniones para que el equipo se conozca y orientarlos sobre lo que les corresponde hacer.
Falta de preparación en las herramientas y tecnologías de trabajo.	-Impartir cursos de capacitación que brinden a todos los miembros del proyecto los conocimientos necesarios sobre las herramientas y tecnologías que van a utilizar.	-Seleccionar las personas con más práctica laboral, para que apoyen la capacitación de los desarrolladores.
Cumplimiento del horario de trabajo.	-Aprovechar al máximo el tiempo disponible para trabajar en el proyecto.	-Realizar un chequeo sistemático del cumplimiento del horario de trabajo.
Gestión de riesgo insuficiente.	-Para esto es necesario llevar a cabo reuniones e informes constantes sobre el estado de los riesgos. -Elaborar un plan de gestión de	-Implementar una cultura en la gestión de los riesgos en todo el personal del proyecto.

	riesgos que permita identificar los principales riesgos a los cuales está expuesto el equipo de desarrollo y controlarlos.	
Escasez de recursos para trabajar.	-Elevar el problema a instancias superiores.	-Realizar una distribución bien óptima de los horarios y recursos existentes, así como un control efectivo del cumplimiento de los mismos de manera que puedan aprovecharse al máximo los recursos disponibles.

Glosario de Términos y Siglas

Términos

Artefacto: Es un término general, para cualquier tipo de información creada, producida, cambiada o utilizada por los trabajadores en el desarrollo del sistema.

Bases Tecnológicas: Abarcan el conocimiento sobre las tecnologías para la construcción del software, la gestión y el soporte del mismo.

Calidad de software: Grado con que el que un sistema, componente o proceso cumple con los requerimientos especificados y las necesidades o expectativas del cliente o usuario.

COCOMO: Es un modelo que permite realizar estimaciones y planificaciones de proyectos de sistemas informáticos.

Componentes de código: Es una parte física y reemplazable del sistema, que cumple y proporciona la realización de un conjunto de interfaces.

Estrategia: Conjunto de acciones que se llevan a cabo para lograr un determinado fin.

Factoría: Se le denomina factoría a cualquier fábrica o industria, del tipo que sea, en la cual se lleve a cabo la transformación de materias primas en otros productos, ya sea para su autoconsumo o para el consumo de otras industrias.

Factoría de software: Es una organización estructurada creada para el desarrollo de software, con procesos estandarizados, repetibles y mejorables continuamente, donde exista una fuerte comunicación con el cliente y las estimaciones se realicen basadas en los datos históricos, fruto de experiencias anteriores. Incorpora técnicas, metodologías y herramientas en el desarrollo del software, que mantiene una mejora continua de procesos y trae como resultados la industrialización en la producción de software.

Framework: Es una estructura de soporte definida en la cual otro proyecto de software puede ser organizado y desarrollado. Típicamente, un framework puede incluir soporte de programas, librerías y un

lenguaje de scripting entre otros software para ayudar a desarrollar y unir los diferentes componentes de un proyecto.

Gestión de proyecto: La gestión de proyectos implica la planificación, supervisión y control del personal, del proceso y de los eventos que ocurren mientras evoluciona el software desde la fase preliminar a la implementación operacional.

Gestores de Factoría: Definen la estructura organizacional de la factoría de software, el proceso y la gestión de calidad en la factoría.

Herramientas CASE: Se puede definir a las Herramientas CASE como un conjunto de programas y ayudas que dan asistencia a los analistas, ingenieros de software y desarrolladores, durante todos los pasos del ciclo de vida de desarrollo de un software (Investigación Preliminar, Análisis, Diseño, Implementación e Instalación.).

Metodologías de desarrollo de software: Conjunto de procedimientos que imponen una serie de pasos sobre el desarrollo del software que permiten producir y mantener un producto garantizando su fiabilidad y calidad.

Modelo de Factoría de software: Es una forma de representar el enfoque de factoría de software.

Modelos de procesos del software: Es una estrategia de desarrollo que acompaña al proceso.

Persona: Las personas son seres humanos que intervienen en el proceso de desarrollo, a diferencia del término abstracto trabajadores.

Portal: Sitio web cuyo objetivo es ofrecer al usuario, de forma fácil e integrada, el acceso a una serie de recursos y de servicios.

Proceso: Es un conjunto de pasos parcialmente ordenados con el propósito de alcanzar una meta.

Proceso de software: Es la definición del conjunto de actividades que guían los esfuerzos de las personas implicadas en el proyecto para transformar los requisitos de usuario en un producto.

Repositorio: Es un sitio centralizado donde se almacena y mantiene información digital, como bases de datos o archivos informáticos.

Repositorio de componentes: Biblioteca de componentes software reutilizables. Los componentes almacenados en el repositorio deben tener una representación estándar y estar bien documentados, siendo el sistema gestor de la biblioteca el encargado de organizar, proteger y gestionar dichos componentes.

Reutilización: Reutilizar es la acción de volver a utilizar los bienes o productos ya elaborados y probados. Puede venir propiciada por una mejora o restauración o sin modificarse, usarlo en la creación de un nuevo producto.

Técnicas: Sucesión ordenada de acciones que se dirigen a un fin concreto, conocido y que conduce a unos resultados precisos.

Tecnología: Conjunto de teorías y de técnicas que permiten el aprovechamiento práctico del conocimiento científico.

Transferencia tecnológica: Consiste en transferir el conocimiento adquirido para construir determinado producto, mediante mecanismos que establecen una estrategia de transferencia de conocimientos.

Prototipo: Puede ser cualquier cosa, desde un trozo de papel con sencillos dibujos a un complejo software.

Siglas

ASD: Adaptive Software Developmen.
CESPO: Centro de Estudios Sociopolíticos y de Opiniones.
CMM: El Modelo de Capacidad y Madurez o Capability Maturity Model.
CMMI: Modelo de Madurez de Capacidades Integrado.
CMS: Sistemas de gestión de contenido.
CSS: Hojas de Estilo en Cascada o Cascading Style Sheets
CVS: Sistema Concurrente de Versiones
GPL: La Licencia Pública General de GNU.

HTML: Lenguaje de Marcas de Hipertexto o HyperText Markup Language.

IP: Infraestructura Productiva.

ISO: Organización Internacional de Estándares o International Organization for Standarization.

ISO 9001: La norma ISO 9001, es un método de trabajo, con el fin de mejorar la calidad y satisfacción de cara al consumidor. Está dirigido a mejorar los aspectos organizativos de una empresa.

PHP: Hypertext Preprocessor.

PSP: Proceso de Software Personal o Personal Software Process.

RUP: Proceso Unificado de Racional o Rational Unified Process.

TSP: Proceso de Software en Equipo o Team Software Process.

XP: Programación Extrema o eXtreme Programming.

UCI: Universidad de Ciencias Informáticas.

UML: Lenguaje Unificado de Modelado