Universidad de las Ciencias Informáticas
Facultad 3

Estrategia de implantación del Modelo de Factoría de Software aplicando Inteligencia para el Polo Productivo Gestión de Proyecto.

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

Autor(es): Niurbelis Cisnero Martínez

Tutor(es): Ing. Moisés Alain Mayet Solano

Ing. Yaimí Trujillo Casañola

Co-tutor: Lic. Gaspar Melchor González Font

Asesor: Dr. Pedro Y. Piñero Pérez

Junio 2007

DEDICATORIA

A mis padres, son mi ejemplo, sin ustedes no hubiera llegado hasta aquí.

A mis hermanos, con su amor me han hecho sentir segura.

A mi prima Clary, siempre ha estado cuando la he necesitado.

A mi familia en general, por ser una familia excelente y muy unida.

AGRADECIMIENTOS

A la Revolución y a Fidel, que han hecho posible el sueño de muchos jóvenes.

A la UCI y sus profesores, han permitido forjarme como una mejor persona.

A Yaimí por haberme ayudado y guiado en el desempeño de la tesis. Gracias por el tiempo que me ha dedicado.

A las personas que aportaron sus ideas para el desarrollo de esta tesis.

A mis amigos que alegraron mi vida a cada momento haciendo más agradable mi estancia en la escuela.

A todas las personas que de una forma u otra se han preocupado por mí y han ayudado a mi formación personal y profesional,

Mis agradecimientos

DECLARACIÓN DE AUTORÍA

Declaro que soy el único autor de este trabajo y autorizo a la Universidad de las Ciencias Informáticas a hacer uso del mismo en su beneficio.

Para que así conste firmo la presente a los ____ días del mes de Junio del año 2007.

_____	_____	_____
Niurbelis Cisnero Martínez	Ing. Moisés Alain Mayet Solano	Ing. Yaimí Trujillo Casañola
Autor	Tutor	Tutor

RESUMEN

El Gobierno Cubano con la ayuda de la Universidad de las Ciencias Informáticas está tratando de llevar la informatización a todas las esferas de la sociedad, realizando software para diferentes empresas. El Polo Productivo Gestión de Proyecto de la Facultad 3 se dedica a crear portales y aplicaciones Web para satisfacer la demanda de estas empresas.

La producción actual de aplicaciones Web en este Polo es aún artesanal, no se cuenta con un proceso de producción de software organizado que permita obtener productos de calidad. Esto ha llevado a la necesidad de buscar alternativas que permitan mejorar el proceso productivo. Una solución son las factorías de software, las cuales permiten mejorar la producción. La forma en que se lleva a la práctica este enfoque es a través de modelos. Varias empresas en el mundo han aplicado algunos modelos obteniendo resultados importantes.

En el presente trabajo se propone una estrategia para implantar el Modelo de Factoría de Software aplicando Inteligencia en el Polo Gestión de Proyecto. La estrategia se basa en las tendencias actuales de la producción de aplicaciones Web en el Polo y la experiencia adquirida en el desarrollo de las mismas.

Para probar la estrategia se tomó un proyecto piloto. A pesar de que aún no se han obtenido todos los resultados que se esperan, sirvió de base para mejorar algunos de los procesos realizados en el proyecto. Se definió un flujo de procesos de acuerdo a la metodología RUP con los roles organizados estructuralmente, se creó el repositorio de componentes.

PALABRAS CLAVE
Factoría de Software, Modelo de Factoría, estrategia de implantación

TABLA DE CONTENIDOS

Índice

INDICE DE FIGURAS

INDICE DE TABLAS

INTRODUCCIÓN

El desarrollo de software constituye un sector de esencial importancia a nivel mundial, se encuentra en el centro de todas las grandes transformaciones. La economía digital, la evolución de las empresas y la administración del conocimiento se resuelven actualmente con software.

El desarrollo de Software en Cuba es aún incipiente. El Gobierno Cubano se ha propuesto como una de sus principales tareas realizar la informatización masiva y ordenada de todas las esferas de la sociedad. Un recurso necesario para realizar esta tarea lo constituyen los técnicos y profesionales relacionados con la informática. Con el fin de contar con más personal capacitado en el uso de las tecnologías, se llevaron a cabo una serie de transformaciones en todo el país. Se crearon nuevos Joven Club de computación, se renovaron laboratorios en centros educacionales y se abrieron nuevos centros especializados en la preparación de futuros ingenieros informáticos. Como parte de este plan, surge la Universidad de las Ciencias Informáticas (UCI).

La UCI es la primera universidad creada en la batalla de ideas, tiene entre sus principales objetivos formar profesionales altamente calificados capaces de llevar la informatización a todas las esferas de la sociedad: salud, educación, cultura, deporte, etc.

La universidad está estructurada por facultades, cada una de ellas asociada con un perfil. La facultad 3 se especializa en el perfil de turismo y negocio, la cual cuenta con tres polos productivos: Sistemas Legales, Gestión de Recursos y Gestión de Proyectos.

El Polo Productivo Gestión de Proyecto se dedica a la creación de software para la gestión de proyectos, la informatización de la infraestructura productiva y del proceso de producción de software. Este polo se dedica a crear Portales y Aplicaciones Web para la Universidad y otras empresas del país.

En los proyectos productivos de este Polo se presenta la siguiente **situación problémica**. La planificación del trabajo no es la mejor, no se siguen estándares establecidos en la Ingeniería de

Software, afectándose la efectividad del equipo de desarrollo. No existe un flujo de procesos bien definidos que le indique a cada persona involucrada en el proyecto cuando tiene que interactuar. No se tiene una metodología de estimación y gestión del tiempo de entrega y el costo de un trabajo determinado basado en el conocimiento real y en la capacidad productiva. Los componentes realizados no se han almacenado en un repositorio donde se encuentren clasificados y documentados, lo que dificulta su reutilización en futuros proyectos.

“Una solución a estos problemas es la producción organizada y estructurada de software, en entidades responsables, que ofrezcan servicios diferenciados orientados a incrementar la calidad del producto final. Las factorías de software, buscan industrializar el desarrollo de sistemas, rescatar ese proceso del ámbito del cliente y trasladarlo a un medio donde se establezcan unos procedimientos y métodos definidos que cuenten con herramientas que ayuden en su implantación”[1].

A pesar de que existen varios modelos de factoría de software probados y aceptados por varias empresas en el mundo con resultados importantes, en el Polo Gestión de Proyecto no se cuenta con un modelo adaptado a sus características que permita organizar la producción de software en el mismo.

A partir de lo antes expuesto se plantea el siguiente **problema científico**:

Las deficiencias en el modelo de producción de software del Polo Productivo Gestión de Proyecto afecta la producción de software.

Determinándose como **objeto de estudio** la actividad productiva en el Polo Productivo Gestión de Proyecto.

El **objetivo general** es: proponer una estrategia de implantación de un Modelo de Factoría de software para el Polo Productivo Gestión de Proyecto.

Y el **campo de acción** es el proceso de producción de software en el Polo Productivo Gestión de Proyecto.

A partir de aquí se plantea la siguiente **Hipótesis**:

Si se estudia el proceso de producción de software existente en el Polo Productivo Gestión de Proyecto y se elabora una estrategia de implantación de un modelo de factoría de software se obtendrá una definición del proceso productivo de software en el polo que contribuya a eliminar las deficiencias existentes.

Entre las **tareas** propuestas para conseguir el objetivo se encuentran:

- Caracterizar el enfoque de factoría.
- Caracterizar los modelos de factoría que se han aplicado en el mundo.
- Identificar los problemas existentes en la producción de software en el Polo Productivo Gestión Proyecto y el potencial con que se cuenta.
- Diseñar y ejecutar el proceso de implantación del Modelo de Factoría de Software para el Polo Productivo Gestión de Proyecto.
- Evaluar la aplicación del modelo en proyectos pilotos.

Se tomó como **población** los proyectos de producción de software del Polo Productivo Gestión Proyecto y como **unidad de estudio** el proceso de producción de software en el mismo.

Durante el desarrollo de la investigación se utilizan métodos **teóricos** y **empíricos**. Dentro de los teóricos: el histórico y el enfoque sistémico. Entre los empíricos: la observación, la encuesta y la entrevista.

Introducción

La revisión bibliográfica constituyó un método importante para la concepción del proyecto de investigación y para apropiarse de los conocimientos relacionados con el tema.

Se usa el Método de Observación para observar todos los procesos y pasos que se llevan a cabo para la realización de aplicaciones Web con el fin de entender este proceso y poder proponer un proceso estandarizado para confeccionar las mismas.

Se realizaron entrevistas a jefes de proyecto para conocer el funcionamiento del proceso productivo en el Polo Gestión de Proyecto. Además de entrevistas a profesionales con conocimiento del tema, a profesores con conocimientos de metodología y otras personas con conocimientos sobre el desarrollo de tesis.

Se realizaron encuestas para determinar los problemas existentes en el polo y las principales herramientas utilizadas.

Este trabajo tiene como principal destinatario a empresas que pretendan implantar el enfoque de factoría o se propongan estandarizar parte de su proceso de producción.

El trabajo fue dividido en tres capítulos, a continuación se muestra el nombre de cada uno con una breve explicación de su contenido:

Capítulo 1. Fundamentación Teórica: Se estudian los fundamentos teóricos relacionados con las Factorías de Software. Se hace un estudio de los Modelos de Factoría existentes definiendo sus características más importantes. Además, se hace referencia a estándares de calidad, así como a algunas metodologías, herramientas y lenguajes de programación afines con las necesidades del Polo.

Capítulo 2. Estrategia de implantación: A partir de encuestas y entrevistas se definieron las características de la producción de software en el Polo Gestión de Proyecto y la necesidad del modelo de factoría para organizar la producción en el mismo. Se define el proceso de

implantación del Modelo de Factoría aplicando Inteligencia para el Polo Gestión de Proyecto, definiendo cada una de sus partes.

Capítulo 3: Resultados de la implantación: Se muestran los resultados obtenidos con la implantación de la estrategia en un proyecto piloto.

CAPÍTULO 1: FUNDAMENTACIÓN TEÓRICA

1.1 Introducción

En este capítulo se abordan temas relacionados con el modelo de factoría para el Polo de Gestión de Proyecto. Se analizan algunos modelos de Factoría de Software existentes en el mundo y estándares de calidad como CMMI. Además de las principales metodologías, herramientas y lenguajes de programación utilizados en el Polo para el desarrollo de software.

1.2 ¿Qué es Factoría de Software?

“Una factoría es cualquier tipo de fábrica o industria, es decir, cualquier tipo de instalación en la cual se produce la transformación de materias primas o productos semiterminados en otros productos, bien para otras industrias, bien para su uso o consumo final. 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”[2].

Las factorías se utilizan para definir cualquier tipo de industria en la que se obtenga un producto a través de la estandarización de su proceso de desarrollo. Ya sea una factoría de conocimiento o una factoría de software en la que el resultado es un producto software creado a partir de un proceso estandarizado cumpliendo con principios de ingeniería y que se tiene en cuenta la reutilización de componentes.

“A fines de los años 60`s e inicios de los 70`s, surge en la industria del software el concepto de Factoría de Software. Este nace como una respuesta a la necesidad de aliviar la incertidumbre que se tenía en el desarrollo de proyectos de software en aspectos como:

- Confiabilidad de los productos.

- Mantener en presupuesto y calendario los proyectos de desarrollo de software.
- Falta de una definición y seguimiento adecuado a los procesos de producción, así como un medio efectivo de medir su desempeño y la productividad de las personas que lo ejecutan.
- Falta de estandarización en los métodos y herramientas empleados en los procesos.
- Falta de herramientas para ser rastreables los productos (requerimientos, especificaciones de productos, etc.) que generan los procesos" [3].

El concepto de Factoría de Software 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:

"Bermer en 1969 define una factoría de software como un ambiente en el cual se construyen programas y se efectúan pruebas. En este ambiente deben existir herramientas para realizar las acciones de construir y probar. Una factoría posee medidas de productividad y calidad, los registros financieros son mantenidos por costo de la producción y la forma de administración debe dar subsidios para prever o estimar datos de futuros proyectos"[4].

Una factoría de software no se centra solamente en construir programas y efectuar pruebas, debe contar además con un repositorio de componentes donde se almacenen los componentes creados durante el desarrollo de software y que permitan reducir el tiempo de desarrollo a partir de su reutilización. Debe tener organizadas a las personas que se encargan de construir y probar los producto, definiendo roles y responsabilidades. La factoría de software debe estar enfocada a un segmento del mercado, ya que la producción de software es un campo bastante grande y no es posible abarcar la producción de todos ellos.

"Según Cusumano (1989), 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" [1].

Una factoría de software debe producir software a gran escala, pero estar centrada en un área especifica. Como bien plantea Cusumano 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 practicas de ingeniería de software.

"En 1992 Cantone plantea que 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" [1].

Una factoría puede producir varios tipos de productos en dependencia de sus facilidades, 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 1992 plantea que 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" [1].

Para la implementación de una factoría no basta con contar con una factoría de componentes, es necesario que los mismos se guarden en un repositorio y sean reutilizados en la construcción de otros productos.

"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 et. al., 1992]" [1].

"Una factoría de software debe poseer un conjunto de herramientas estandarizadas para la construcción de software, bases históricas para ser usadas en la dirección de proyectos, y principalmente, poseer un alto grado de reutilización de código en el proceso de desarrollo de un determinado software, apoyado en una base de componentes reutilizables [Li et. al., 2001]" [1].

Una factoría de software debe además, especializar a sus profesionales en un área determinada, definiendo roles y responsabilidades.

"Según Fernández y Teixeira 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" [1].

Una factoría de software es una organización que cuenta con herramientas para soportar el proceso de desarrollo y que divide el trabajo entre los implicados en dicho proceso, especializando a cada uno en un área determinada. Como dicen Fernández y Teixeira debe producir software a través de un proceso estandarizado aplicando conceptos de ingeniería de software y que tenga en cuenta en todo momento la demanda de los usuarios y clientes.

"Una Factoría de Software es una línea de productos software que configura herramientas extensibles, procesos y contenido para automatizar el desarrollo y mantenimiento de variantes de un producto arquetípico mediante la adaptación, ensamblaje y configuración de componentes basados en frameworks. Por lo tanto, las Factorías de Software se centran en el desarrollo de sistemas similares promoviendo la reutilización de arquitecturas, componentes software y conocimiento" [3].

Las factorías de software tienen definida la arquitectura de los productos que desarrollan y cuentan con herramientas estandarizadas para soportar el proceso de desarrollo. Hacen uso de la reutilización de componentes para reducir el tiempo de desarrollo y aumentar la calidad de los productos.

“Según Fernández y Teixeira una factoría de software puede tener varios dominios de actuación en dependencia de las fases de desarrollo del software en las que opere, desde un proyecto de software completo, hasta un proyecto físico o solo codificación del software” [1].

Dadas las características que posee la producción de software en la UCI no es posible clasificarla de acuerdo a un tipo de factoría según las definen Fernández y Texeira, ya que se realizan varios tipos de productos. La Factoría en el Polo Gestión de Proyecto, se puede clasificar como una Factoría de Proyectos Físicos, en sus proyectos generalmente se realiza diseño, implementación y prueba, aunque se tiene un dominio en muchos casos del negocio a automatizar.

Estos autores enumeran una serie de características inherentes a una estructura fabril para software:

- Proceso definido y estandarizado para el desarrollo de software.
- Control y almacenamiento en bibliotecas de componentes de software (documentos, código, métodos, etc.)
- Mejora continua del proceso.
- La producción de software debe estar fuertemente basada en métodos y técnicas estandarizadas.
- Interacción controlada con el cliente, alto poder de entendimiento.
- Planificación y control de la producción.

- Hardware y software ajustado a las necesidades del usuario.
- Estimación de costos y tiempo basados en el conocimiento real de la capacidad productiva, mediante métodos de obtención basados en datos históricos.
- Control del estado de ejecución de todas las demandas.
- Garantía de la calidad del producto.
- Control de los recursos humanos involucrados.
- Capacitación de los recursos humanos.
- Mecanismos de control de costos.

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. 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. La factoría de software debe 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.

“El enfoque de factoría de software viene a formalizar todos los procesos (etapas de producción) y sus productos, trabajando en líneas de producción, con etapas y tareas perfectamente definidas para cada tipo de profesional involucrado en el proceso, yendo de la productividad en la línea de producción a las rutinas de control de la calidad. Se busca la especialización de los profesionales, para que cada uno garantice la productividad de la fase en la que está ocupado.

Entre los principales objetivos trazados por una factoría de software están:

- Industrializar el desarrollo de sistemas de software.
- Producción de software a gran escala.
- Lograr una alta productividad en el desarrollo de software.
- Establecer líneas 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" [1].

1.3 Modelos de factoría existentes

En este epígrafe se hace un análisis de los modelos de Factoría de Software más significativos encontrados en la bibliografía consultada, de los cuales será seleccionado uno para implantarse en el Polo de Gestión de Proyecto. Estos modelos son la forma en que varias empresas y entidades han llevado a la práctica el enfoque de Factoría de Software.

1.3.1 Modelo basado en la norma ISO 9001 y CMM

“Este modelo divide la Factoría de Software en cinco entidades bien definidas (Ver Anexo 1). 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.

La Entidad Técnicas provee el soporte técnico y conceptual para la definición del proceso (representado en la Entidad Proceso), el cual es guiado por el estándar de calidad CMM. La gestión de proyecto definido en la Entidad Gestión de la Factoría se tiene en cuenta durante todo el ciclo de vida del proyecto. Los requisitos de calidad para la organización de la factoría son definidos por la norma ISO 9001.

La Entidad Gestión de la Factoría define, a través de las sub-entidades Gestión de Calidad y Organización del Modelo de Proceso, los trabajadores involucrados en el proceso de desarrollo de software y sus roles, los cuales son guiados por PSP y TSP.

Los activos del proceso, las herramientas y los componentes de código dan soporte al proceso de desarrollo de software” [5].

1.3.2 Modelo Eureka

“El modelo fabril propuesto por el proyecto Eureka está compuesto de proceso, reglas, herramientas, información, equipamiento (computadoras) y trabajadores (Ver Anexo 2).

El proceso de desarrollo de este modelo 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 almacenada, soportan la automatización del proceso de desarrollo. El modelo posee características giradas al proceso de desarrollo de software distribuido, en el mismo se sigue el enfoque software bus” [5].

1.3.3 Modelo Clasificatorio

“El Modelo Clasificatorio propuesto por Fernández y Teixeira está dirigido a clasificar las factorías de acuerdo al alcance o ámbito de funcionamiento que tienen a lo largo del proceso de desarrollo de software (Ver Anexo 3). Una Fábrica de Software puede ser clasificada como:

Factoría de proyecto ampliada: comprende el concepto de arquitectura de solución. La arquitectura de solución es una etapa anterior al diseño conceptual del software, la cual se ocupa en proyectar una solución en la que el software está formado por los componentes más significativos arquitectónicamente, se definen los principios que orientan el diseño y evolución del software.

Factoría de Proyectos de Software: abarca todo el ciclo de vida sistémico para la realización del software. Se tiene un conocimiento al detalle del negocio a automatizar.

Factoría de Proyectos Físicos: se dedica al diseño, implementación y prueba. No se tiene un pleno conocimiento del negocio.

Factoría de Programas: desarrolla componentes de código para la construcción del software” [5].

1.3.4 Modelo propuesto por Basili.

El presente modelo divide una factoría de software en dos grandes entidades: organización basada en proyectos y factoría de componentes (Ver Anexo 4).

“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" [5].

1.3.5 Modelo Replicable

"El presente modelo fue desarrollado para ser replicado en una factoría de software. El mismo 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, las cuales poseen un proceso guiado por un modelo de calidad de software. El proceso puede ser aplicado al desarrollo de software o al desarrollo de un componente. (Ver Anexo 5)

La organización de la producción esta dividida en cinco áreas: área de producción de análisis de sistema o modelado de negocio, área de producción de diseño de software, área de construcción de software, área de producción de componentes de infraestructura o activos del proceso, área de producción de componentes de código.

La unión de las áreas de análisis de sistemas, diseño de software y construcción de software forman el ámbito de negocio del modelo de producción. El ámbito de negocio incluye la interacción entre el cliente y la fábrica de software. Las áreas de producción de componentes de infraestructura y componentes de código forman el ámbito interno del modelo, el que es transparente a los ojos del cliente de la factoría. Este ámbito es el responsable de los subproductos creados, componentes para la construcción del sistema" [1].

Este modelo define una serie de actividades para implementar el proceso, roles de trabajo que ejecutan estas actividades y una serie de herramientas, métodos y mecanismos para automatizar el proceso.

1.3.6 Modelo aplicando Inteligencia

"El modelo aplicando inteligencia hace una división de la factoría de software en seis entidades y sus relaciones (Ver Anexo 6):

- Bases tecnológicas: Comprende el contexto de las bases tecnologías y herramientas para dar soporte y automatización al proceso de desarrollo.

- Proceso de desarrollo: 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 sub-entidades: Gestores de la Factoría y Grupo de desarrollo.

- Repositorio de componentes: 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.

- Gestión de la Factoría: Comprende todos las áreas de la gestión de proyecto.

- Centro de Inteligencia: Tiene el objetivo de utilizar herramientas de Vigilancia Tecnológica, Inteligencia Empresarial, Prospectiva que permitan la orientación estratégica.

La descripción de la arquitectura y composición del modelo se basa en que el resultado final de un proyecto es un producto, que toma forma durante su desarrollo gracias a la intervención de varios tipos de personas, que utilizan PSP y TSP para la planificación personal y en equipo. Estas personas están representadas en el modelo por la Entidad Personas, la que se divide en Grupo de desarrollo y Gestores de la Factoría. El Grupo de Desarrollo lo conforman las personas

involucradas directamente en la producción, los cuales son quienes ejecutan las actividades o flujos de trabajo, guiados por el proceso de desarrollo de software, representado en el modelo mediante la Entidad Proceso de desarrollo. La sub-entidad Gestores de la factoría está compuesta por el equipo de dirección de la factoría, encargados del control y gestión del Grupo de Desarrollo.

El proceso es automatizado y soportado por diversas tecnologías y herramientas, representados en el modelo mediante la Entidad Bases tecnológicas. El proceso de desarrollo es regulado por CMMI, modelo de calidad integrado para la industria del software que provee áreas y prácticas importantes para el desarrollo y evaluación del proceso de desarrollo y la gestión de proyectos.

La reutilización tiene efectos muy positivos en el desarrollo de software, entre estos efectos está el aumento en la productividad y calidad así como la reducción del tiempo de desarrollo. Para dar soporte al proceso en este sentido la factoría cuenta con una base de componentes reutilizables, representada en la entidad “Repositorio de Componentes”. Todo esto es gestionado desde la Entidad Gestión de la Factoría la cual recibe la orientación estratégica de la Entidad Centro de Inteligencia” [5].

1.3.7 Comparación de los modelos.

El Modelo basado en la norma ISO 9001 y CMM define las entidades que conforman el modelo de factoría y las relaciones que se establecen entre ellas. Usa estándares de calidad reconocidos a nivel mundial como ISO 9001 y CMM. Hace uso de PSP y TSP para la planificación personal y en equipo. Este modelo tiene algunas características negativas, no desglosa las actividades, objetivos y características de cada una de las entidades que conforman el modelo.

El aporte del Modelo Eureka está en el desarrollo distribuido de software, da una visión de cómo se puede distribuir la construcción de un producto software entre diferentes factorías, y después realizar la unión de los componentes elaborados por cada una para conformar el producto final. Hace uso de herramientas para soportar el proceso de desarrollo. Este modelo no usa

estándares de calidad, aspecto importante a tener en cuenta en la producción de software actual. No divide el modelo en entidades que permitan describir el modelo y facilitar la aplicación del mismo por otras empresas que quieran adoptar el enfoque de factoría. Tampoco define como se organiza la producción, el proceso y los desarrolladores, ni como se gestiona el proyecto.

El Modelo Clasificatorio permite clasificar las Factorías de Software de acuerdo a las actividades que se realizan durante el proceso de desarrollo de software, ayuda a identificar de qué tipo es la factoría de software y sus avances en este enfoque. Se puede construir un producto completo o parte de él. Solo se enmarca en el proceso de desarrollo, no define la demás áreas involucradas en el proceso. No tiene en cuenta los estándares de calidad, la organización de la producción y de las personas, y el uso de herramientas para la automatización de los procesos de desarrollo, soporte y gestión.

El Modelo propuesto por Basili divide la factoría en dos unidades aumentando la eficiencia y especialización en la producción. Se basa en la reutilización de componentes durante el desarrollo, los cuales se encuentran en la base de componentes reutilizables. Este modelo solo se enmarca en el área de producción, dejando sin definir las demás áreas involucradas en la producción. No tiene en cuenta el uso de herramientas para la automatización del proceso y no define la organización del proceso y las personas involucradas en el mismo.

El Modelo Replicable reúne las características más importantes de los modelos anteriores y puede ser adaptado a cualquier factoría de software. Es mucho más explícito a como aplicarse que los modelos anteriores. Presenta una propuesta de roles a asignar a cada una de las actividades aunque no los organiza estructuralmente. Describe las técnicas y herramientas a utilizar y define mejor el proceso de producción pero no se enmarca en el uso de alguna de las metodologías estandarizadas, ni utiliza estándares de calidad. No tiene en cuenta la gestión de proyecto.

El Modelo de Factoría aplicando Inteligencia es un modelo que propone la reutilización de componentes, aspecto a tener en cuenta en cualquier factoría de software. Permite definir el proceso a partir de la metodología usada para la construcción del producto. Las personas que participan se organizan por roles y planifican sus actividades tanto personales como en equipo, usan TSP y PSP. Hace uso de estándares reconocidos a nivel mundial como lo son ISO y CMMI. Propone el uso de métodos y herramientas para soportar el proceso de desarrollo. Tiene en cuenta la gestión del proyecto y la calidad del producto y proceso. Además usa inteligencia organizacional y empresarial para la orientación estratégica.

El Modelo de Factoría de Software aplicando Inteligencia es el más completo de todos, reúne las características más importantes de los modelos anteriores. Estas características lo hacen propicio para ser implantado en la factoría de software del Polo de Gestión de Proyecto.

1.4 Técnicas utilizadas para replicar un modelo y proceso industrial

El proceso industrial puede ser considerado un producto, el mismo puede configurarse en paquetes que pueden ser replicados por las organizaciones que quieren desarrollar las técnicas industriales para la producción del software.

La unidad industrial replicada puede operar de dos maneras: orientada a objetos y orientada a procesos. Si la unidad industrial replicada es orientada a objetos, las herramientas de diagramas, las IDE (Entorno Integrado de Desarrollo), y los lenguajes de programación deben corresponder al concepto de orientado a objetos. Se pueden configurar paquetes orientados a objetos y orientados a procesos.

Otra forma de constituir los paquetes es tener como base el modelo de organización de una factoría de software. Los paquetes pueden configurarse tomando en cuenta las características del modelo y los procesos que se realizan en la factoría.

1.4.1 Modelo y Técnicas de Replicación

El modelo de replicación usa las técnicas del traslado de conocimiento para transferir el conocimiento sobre la factoría de software. Este modelo está dividido en 7 entidades diferentes. El diseño del mismo se puede ver en la siguiente figura.

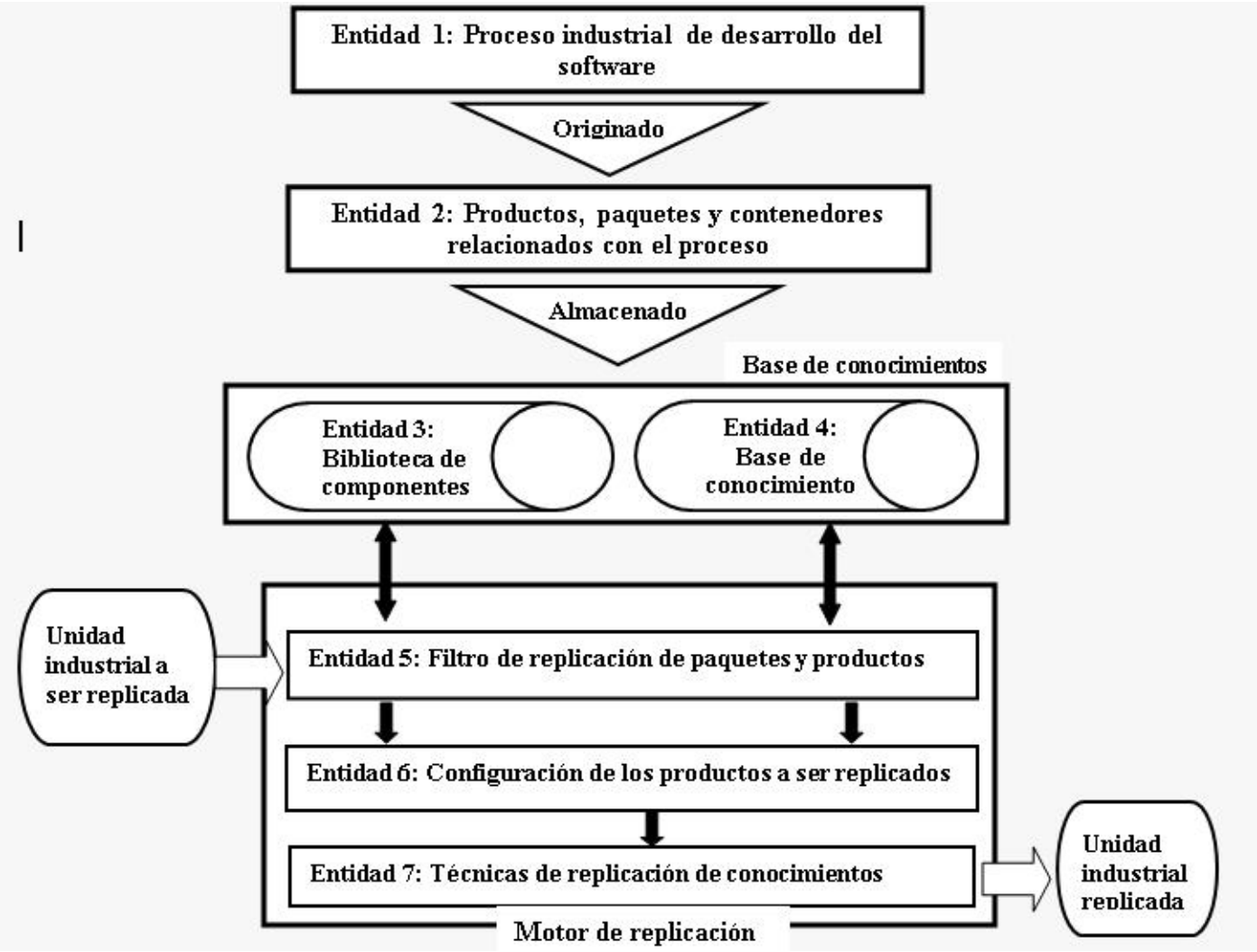

Figura 1. 1 Modelo de replicación para una Factoría de Software.

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 conocimiento sobre los productos (Entidad 4). En la biblioteca de componentes, se guardan los componentes del código e infraestructura.

En la entidad 4 se guarda todo lo referente al proceso industrial, los productos, los paquetes, contenedores, y las técnicas de representación de conocimiento.

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. Finalmente, en la entidad 7 se usa una de las técnicas de replicación de conocimiento para transferir el conocimiento.

1.5 Estándares de Calidad

1.5.1 Modelo de Madurez de Capacidad Integrado (CMMI)

CMMI es un modelo de calidad del software que clasifica las empresas en niveles de madurez. Estos niveles sirven para conocer la madurez de los procesos que se realizan para producir software. CMMI tiene 5 niveles:

Inicial o Nivel 1 CMM – CMMI: Este es el nivel en donde están todas las empresas que no tienen procesos. Los presupuestos se disparan, no es posible entregar el proyecto en fechas, te tienes que quedar durante noches y fines de semana para terminar un proyecto. No hay control sobre el estado del proyecto, el desarrollo del proyecto es completamente opaco, no sabes lo que pasa en él.

Repetible o Nivel 2 CMM – CMMI: Quiere decir que el éxito de los resultados obtenidos se puede repetir. La principal diferencia entre este nivel y el anterior es que el proyecto es gestionado y

controlado durante el desarrollo del mismo. El desarrollo no es opaco y se puede saber el estado del proyecto en todo momento.

Los procesos que hay que implantar para alcanzar este nivel son:

- Gestión de requisitos
- Planificación del proyecto
- Seguimiento y control del proyecto
- Gestión de acuerdos con proveedores
- Medición y análisis
- Aseguramiento de calidad de procesos y productos
- Gestión de configuración.

Definido o Nivel 3 CMM – CMMI: Resumiéndolo mucho, alcanzar este nivel significa que la forma de desarrollar proyectos (gestión e ingeniería) está definida, por definida quiere decir que está establecida, documentada y que existen métricas (obtención de datos objetivos) para la consecución de objetivos concretos.

Los procesos que hay que implantar para alcanzar este nivel son:

- Desarrollo de requisitos
- Solución Técnica
- Integración del producto
- Verificación

- Validación
- Desarrollo y mejora de los procesos de la organización
- Definición de los procesos de la organización
- Planificación de la formación
- Gestión de riesgos
- Análisis y resolución de toma de decisiones

Cuantitativamente Gestionado o Nivel 4 CMM – CMMI: Los proyectos usan objetivos medibles para alcanzar las necesidades de los clientes y la organización. Se usan métricas para gestionar la organización.

Los procesos que hay que implantar para alcanzar este nivel son:

- Gestión cuantitativa de proyectos
- Mejora de los procesos de la organización.

Optimizado o Nivel 5 CMM – CMMI: Los procesos de los proyectos y de la organización están orientados a la mejora de las actividades. Mejoras incrementales e innovadoras de los procesos que mediante métricas son identificadas, evaluadas y puestas en práctica.

Los procesos que hay que implantar para alcanzar este nivel son:

- Innovación organizacional
- Análisis y resolución de las causas

La implantación de un modelo de estas características es un proceso largo y costoso que puede costar varios años de esfuerzo. Aún así, el beneficio obtenido para la empresa es mucho mayor que lo invertido.

1.5.1.1 Evaluaciones basadas en CMMI (SCAMPI)

"En paralelo con el desarrollo de CMMI, el SEI elaboró un método para la evaluación formal del modelo denominado SCAMPI (*Standard CMMI Appraisal Method for Process Improvement*). El método define una serie de reglas para la evaluación del modelo, las cuales deben utilizarse para valorar las distintas partes del mismo durante una evaluación formal"[6].

SCAMPI permite:

- Comprender mejor el nivel de competencia en ingeniería de una organización, identificando los puntos fuertes y débiles de sus procesos actuales.
- Relacionar esos puntos fuertes y débiles con el modelo CMMI.
- Priorizar planes de mejora.
- Centrarse en las mejoras más importantes que haya que acometer según el nivel de madurez de la organización y de los recursos de que disponga.
- Obtener para la organización su clasificación en uno de los niveles del modelo.
- Identificar riesgos de desarrollo y adquisición relativos a las limitaciones de la organización.

El método SCAMPI consta de tres fases:

Fase 1: Planeamiento y Preparación para el Appraisal

Fase 2: Conducción del Appraisal.

Fase 3: Resultados de la evaluación.

1.5.2 ISO (International Standard Organization)

ISO ofrece un enfoque sistemático para la calidad total, presionando a las empresas a documentar, implantar y mantener un sistema contable detallado de sus procedimientos y especificaciones de trabajo.

ISO 9000 es una norma acordada internacionalmente para asegurar un sistema gerencial de calidad. La norma desarrolla una serie de guías que apoyan a los proveedores y a los fabricantes para desarrollar un sistema de calidad.

Las normas de la familia ISO 9000 se han elaborado para asistir a las organizaciones, de todo tipo y tamaño, en la implementación y la operación de sistemas de calidad eficaces. Brindan el marco para documentar en forma efectiva los distintos elementos de un sistema de calidad y mantener la eficiencia del mismo dentro de la organización.

ISO 9000 desarrolla una serie de requerimientos que son mucho más amplios que el control y/o inspección. Busca que todo aspecto relacionado con la producción, la administración o el proceso de servicios sea adecuadamente planificado y operado, que se tengan registros y que se tomen acciones con relación a problemas.

La serie ISO 9000 está formada por cinco documentos, tres de ellos son modelos de aseguramiento de la calidad, específicamente el 9001, el 9002 y el 9003. Los otros dos son simples lineamientos que sirven de apoyo.

- ISO 9000: Principios y conceptos, lineamientos para su selección y utilización.

- ISO 9001: Modelo de aseguramiento de la calidad, aplicable al diseño, desarrollo, fabricación, instalación y servicio.
- ISO 9002: Modelo de aseguramiento de la calidad, aplicable a la fabricación y a la instalación.
- ISO 9003: Modelo de aseguramiento de la calidad, aplicable a la inspección y ensayos finales.
- ISO 9004: Principios y conceptos, lineamientos para la gestión de calidad y elementos del sistema de calidad.

ISO 9001 es la norma que sirve de modelo a las empresas que desean desarrollar un sistema de calidad que cubra las actividades de: diseño, desarrollo, producción, instalación y servicios posventa. Actualmente, es la norma más completa y más exigente de la familia ISO 9000, y exige el cumplimiento de veinte requisitos.

“ISO 9001 especifica los requisitos para un sistema de gestión de la calidad que pueden utilizarse para su aplicación interna por las organizaciones, para certificación o con fines contractuales. Se centra en la eficacia del sistema de gestión de la calidad para dar cumplimiento a los requisitos del cliente” [7].

ISO 9001 es un conjunto de reglas de carácter social y organizativo para mejorar y potenciar las relaciones entre los miembros de una organización. Pretenden mejorar las capacidades y rendimiento de la organización, y conseguir aumentar la calidad final del producto.

“ISO 9001 requiere que la política de calidad esté definida, documentada, sea comprendida, implementada, y mantenida; Esas responsabilidades y autoridades para todos los empleados especificando, alcanzando, y la calidad de monitoreo esté definida; y esos recursos de verificación de la misma organización estén definidos, entrenados, y financiados. Un gerente designado asegura que el programa de calidad es implementado y mantenido” [5].

Adoptar la norma ISO 9001 en la Factoría trae beneficios para la misma, se logra que la información esté actualizada y sea efectiva. Además genera confianza en la capacidad de sus procesos, en la calidad de sus productos y proporciona las bases para la mejora continua. Se logra una mejor aceptación por parte de los clientes. Usar la norma ISO 9001 es una carta de presentación para abrir nuevos mercados.

1.5.3 PSP y TSP

“El Proceso de Software Personal (PSP) ayuda a los ingenieros del software a hacer bien su trabajo. Muestra cómo aplicar métodos avanzados de ingeniería a sus tareas diarias, proporciona métodos detallados de planificación y estimación, muestra cómo controlar su rendimiento y explica cómo los procesos definidos guían su trabajo”[8].

PSP es un proceso que permite mantener un control de todas las actividades que se realizan en el día tanto en el trabajo como fuera de él, llevando un control de las mismas. Si es usado apropiadamente, brinda los datos históricos necesarios para trabajar mejor y lograr que los elementos rutinarios del trabajo sean más predecibles y eficientes.

El Proceso de Software en Equipo (TSP) es un proceso que al igual que el PSP, está basado en el modelo CMM. Es un modelo o proceso de trabajo en equipo enfocado a aminorar varios de los problemas, tanto técnicos como administrativos, que se presentan en el desarrollo de software.

TSP está diseñado para ayudar a controlar, administrar y mejorar la forma en que trabaja un equipo de software. Está estructurado por formularios, guías y procedimientos para desarrollar software.

“TSP provee un esquema de trabajo donde cada desarrollador tiene perfectamente definido sus roles, sus actividades, y sus responsabilidades. Incluye procedimientos para la mejora continua del proceso de desarrollo, para mejorar la calidad del software producido, para mejorar la estimación del tiempo de desarrollo, para la disminución de defectos en el producto y para

promover la integración del equipo de desarrollo. Es decir, el TSP apoya tanto al equipo de desarrollo como a los administradores del proyecto para la culminación a tiempo y dentro del presupuesto de los proyectos de desarrollo de software" [5].

1.6 Metodologías de desarrollo de software

Las metodologías imponen un proceso disciplinado sobre el desarrollo de software con el fin de hacerlo más predecible y eficiente. Lo hacen desarrollando un proceso detallado con un fuerte énfasis en planificar inspirado por otras disciplinas de la ingeniería.

"A pesar de que se conocen varias metodologías para controlar el proceso de producción de software, existe un grupo de características que no pueden faltar para lograr llevar a buen término el trabajo. 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.

No existe una metodología universal para hacer frente con existo a cualquier proyecto de desarrollo. Toda metodología debe ser adaptada al contexto del proyecto: recursos técnicos y humanos, tiempo de desarrollo, tipo de sistema, etc." [5].

Actualmente existen dos clasificaciones para las metodologías: Tradicionales o Fuertes y Ligeras o Ágiles. Las metodologías tradicionales se centran especialmente en el control del proceso, estableciendo rigurosamente las actividades involucradas, los artefactos que se crean y las herramientas y notaciones que se usarán. Dentro de esta metodología se destaca RUP (Rational Unified Process), como la metodología líder en el desarrollo de software. Las metodologías ágiles dan mayor valor al individuo, a la colaboración con el cliente y al desarrollo incremental del software con iteraciones muy cortas. Dentro de estas metodologías se encuentran XP (eXtreme Programming), Crystal Methodologies, SCRUM, entre otras.

Para el desarrollo de sistemas en el Polo de Gestión de proyectos se usa una combinación entre RUP y XP.

1.6.1 Proceso Unificado de Rational (RUP)

El Proceso Unificado de Rational o RUP (Rational Unified Process), es un proceso de desarrollo de software que utiliza como notación UML (Lenguaje Unificado de Modelado). Constituye la metodología estándar más utilizada para el análisis, implementación y documentación de sistemas orientados a objetos.

“RUP es más que un simple proceso; es un marco de trabajo genérico que puede especializarse para una gran variedad de sistemas software, para diferentes áreas de aplicación, diferentes tipos de organizaciones, diferentes niveles de aptitud y diferentes tamaños de proyectos”[9].

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). Presenta una forma disciplinada de asignar tareas y responsabilidades (quién hace qué, cuándo y cómo). Se caracteriza por estar guiado por los casos de uso, estar centrado en la arquitectura y ser iterativo e incremental. Esto es lo que hace único al Proceso Unificado.

RUP pretende implementar las mejores prácticas actuales en ingeniería de software:

- Desarrollo iterativo del software
- Administración de requerimientos
- Uso de arquitecturas basadas en componentes
- Modelamiento visual del software
- Verificación de la calidad del software

- Control de cambios

1.6.2 Programación eXtrema (eXtreme Programming, XP)

“XP es una metodología ágil 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 como especialmente adecuada para proyectos con requisitos imprecisos y muy cambiantes, y donde existe un alto riesgo técnico. Los principios y prácticas son de sentido común pero llevadas al extremo, de ahí proviene su nombre” [10].

Las actividades que se realizan en XP son las siguientes:

- Planificación
- Diseño
- Codificación
- Pruebas

1.7 Análisis RUP-CMMI

“Implementar RUP en el desarrollo de un proyecto implica que ciertas áreas de proceso de CMMI sean alcanzadas y otras no. A continuación se presenta un análisis de las áreas de proceso de CMMI en relación a RUP (Ver tabla resumen en Anexo 7):

Nivel 2: Repetible (Managed)

- Gestión de requisitos (Requirements Management)

RUP define claramente el proceso de administración de requerimientos y aporta herramientas como los casos de uso. La gestión de requisitos es una de las bases de RUP.

- Planificación del proyecto (Project Planning)

RUP habla de la planeación del proyecto de manera iterativa y del control de riesgos.

- Seguimiento y control de proyecto (Project Monitoring and Control)

RUP define cómo debe ser el control del proyecto.

- Gestión de acuerdo con proveedores (Supplier Agreement Management)

RUP no menciona nada sobre administración de acuerdos, es algo no considerado.

- Medición y análisis (Measurement and Analysis)

La medición y análisis no están contemplados detalladamente en RUP.

- Aseguramiento de la calidad de procesos y productos (Process and Product Quality Ass.)

En la etapa de transición se realiza la verificación de la calidad aunque no tan detallada como lo exige CMMI. La verificación de calidad del producto está bien definida (Planes de pruebas, ejecución, evaluación de las pruebas) pero la evaluación de calidad del proceso no está considerada.

- Gestión de configuración (Configuration Management)

RUP es muy claro cuando se habla de administración de la configuración, incluso es una de las mejores prácticas recomendada.

Nivel 3: Definido (Defined)

- Desarrollo de requisitos (Requirements Development)

RUP define el proceso de levantamiento de requisitos desde que se identifica lo que el sistema debe hacer hasta que se elaboran prototipos.

- Solución técnica (Technical Solution)

La solución técnica se refiere a todo el proceso de construcción del producto, algo así como la etapa de construcción de RUP.

- Integración del producto (Product Integration)

Está muy relacionado con la solución técnica, consiste en la integración del producto con otros sistemas, es equivalente a lo que define RUP como etapa de transición.

- Verificación (Verification)

CMMI es más amplio que RUP en cuanto a este aspecto que consiste en ir verificando que cada parte del proceso se hace bien.

- Validación (Validation)

Es equivalente a cuando en la etapa de transición en RUP se valida la satisfacción del usuario.

- Desarrollo y mejora de los procesos de la organización (Organizational Process Focus)

Está relacionado con los objetivos y metas de la compañía, algo que en RUP no está considerado ya que RUP se refiere al proceso de desarrollo mientras CMMI se enfoca a la organización.

- Definición de los procesos de la organización (Organizational Process Definition)

Aunque RUP define cómo deben ser las partes del proceso de desarrollo, CMMI es mucho más amplio y exige la definición de procesos más generales de las organizaciones.

- Planificación de la formación (Organizational Training)

Se refiere al entrenamiento de personal que no está considerado en RUP.

- Gestión integrada de proyecto (Integrated Project Management)

Se refiere a la coordinación del equipo de un proyecto con otros grupos y otras partes de la organización. RUP define roles y actividades pero no es tan específico como lo requiere CMMI.

- Gestión de riesgos (Risk Management)

RUP considera al igual que CMMI la evaluación de riesgos como parte fundamental del proceso.

- Análisis y resolución de la toma de decisiones (Decision Analysis and Resolution)

CMMI exige la definición de mecanismos claros para cuantificar y evaluar la toma de decisiones que puedan afectar al proceso, lo que no hace RUP.

Nivel 4: Cuantitativamente Gestionado **(**Quantitatively manager)

- Mejora de los procesos de la organización (Organizational process performance)

Se trata de la manera de cuantificar y evaluar el desempeño de la empresa, el logro de objetivos y metas. RUP no es tan específico en esta materia aunque define evaluación de partes del proceso.

- Gestión cuantitativa de proyectos (Quantitative Project Management)

Se refiere a la administración cuantitativa del desempeño de los procesos y la calidad. CMMI es explícito en la exigencia del control estadístico de procesos.

Nivel 5: Optimizado (Optimizing)

- Innovación organizacional (Organizational Innovation and Deployment)

Se enfoca a los cambios en cuanto a tecnología y a la administración de cambios en el proceso. RUP define como hacer las cosas pero no cómo mejorarlas.

- Análisis y resolución de las causas (Causal Analysis and Resolution)

Se trata de la parte de prevención de defectos, el estudio de las causas y la generación de posibles soluciones y mecanismos para evitarlos. RUP no trata nada de esto" [11].

CMMI y RUP son compatibles, la implementación de RUP apoya ciertas prácticas requeridas por el modelo CMMI, pero el hecho de seguir la metodología de desarrollo unificado no implica que se cumplan con las áreas de proceso de CMMI. Es necesario que en el desarrollo de los productos se establezca mecanismos para controlar que las áreas de proceso de CMMI se están cumpliendo realmente.

1.8 Tecnologías para software de Gestión de Proyecto.

1.8.1 Aplicaciones Web

Como resultado del desarrollo y evolución del mundo, desde hace ya varios años, surgió una red gigante que agrupa miles de redes de computadoras distribuidas por toda la superficie del globo conocida como Internet.

Uno de los servicios más importantes y más usados de Internet es el World Wide Web, (telaraña de alcance mundial), o simplemente Web, que constituye el universo de información accesible a través de Internet, fuente inagotable del conocimiento humano. Esta información se visualiza de manera gráfica e interactiva haciendo uso del sistema de hipertexto.

Este servicio, de igual forma, ha evolucionado. Estos últimos se conocen como aplicaciones Web, y son implementados por grupos de desarrollo de software, tal como en las aplicaciones de escritorio.

Con las aplicaciones Web el usuario no sólo recibe páginas del servidor como respuesta a su solicitud, sino que puede también enviar información de regreso a través de formularios. Están diseñadas para interactuar con bases de datos con el fin de recoger, almacenar, organizar y distribuir información, creando herramientas poderosas a ser utilizadas en la administración consistente de la información.

1.8.2 Portales Web

Un Portal Web es una aplicación Web que gestiona de forma uniforme y centralizada contenidos provenientes de diversas fuentes, implementa mecanismos de navegación sobre los contenidos, integra aplicaciones e incluye mecanismos de colaboración para el conjunto de usuarios (comunidad) a los que sirve de marco de trabajo.

1.9 Herramientas propuestas

Dado que en el Polo se trabaja por Windows y Linux, se proponen herramientas que generalmente permiten el trabajo en ambos sistemas operativos.

1.9.1 Plone

Plone es un Sistema de Gestión de Contenidos (CMS) basado en Zope y programado en Python. Es Open Source. Puede utilizarse como servidor intranet o extranet, un Sistema de Publicación de documentos y una herramienta de trabajo en grupo para colaborar entre entidades distantes. Plone es escalable, estable, evolutivo y basado en estándares del mercado. Está disponible en numerosas plataformas: Linux, Windows, Sun Solaris, Mac OS X y numerosas versiones Unix. Plone tiene soporte internacional extendido y su interface está traducido a más de treinta idiomas.

1.9.2 Zope (Framework de gestión de Contenidos, Server.)

Zope es un servidor de aplicaciones Web escrito en Python. Zope es empleado para desarrollar sistemas de administración de contenido como Plone y Silva, Intranets, Portales, ERP y aplicaciones personalizadas. Provee base de datos propia orientada a objetos, conectores a base de datos relacionales como MySql, mecanismo para generación páginas dinámicas, entre otros.

1.9.3 ArgoUML (Modelador de diagrama)

ArgoUML es una herramienta profesional de Modelado con UML, es de uso libre. Incluye todos los diagramas UML 1.3. Permite ingeniería inversa de código Java y generación de código. Disponible para las plataforma más importantes e integración con Eclipse. Se ha desarrollado con los criterios de código abierto y licencia libre.

1.9.4 Eclipse

Eclipse es una plataforma de software de Código abierto independiente de una plataforma. La plataforma Eclipse está diseñada para la construcción de aplicaciones Web, aplicaciones Java de todo tipo, Enterprise JavaBeans (EJBs) y programas C++.

La Plataforma Eclipse permite descubrir e invocar funcionalidad implementada en componentes llamados plugins. Un plug-in es la unidad mínima de funcionalidad de Eclipse que puede ser distribuida de manera separada.

La Plataforma Eclipse está diseñada para afrontar las siguientes necesidades:

- Soportar la construcción de gran variedad de herramientas de desarrollo.
- Soportar las herramientas proporcionadas por diferentes fabricantes de software independientes (ISV’s)

- Soportar herramientas que permitan manipular diferentes contenidos (ejemplo HTML, Java, C, JSP, EJB, XML, y GIF).
- Facilitar una integración transparente entre todas las herramientas y tipos de contenidos sin tener en cuenta al proveedor.
- Proporcionar entornos de desarrollo gráfico (GUI) o no gráficos.
- Ejecutarse en una gran variedad de sistemas operativos, incluyendo Windows y Linux.

Eclipse es una herramienta para todo, y para nada en particular.

1.10 Lenguajes de programación

1.10.1 Python

Python es un lenguaje interpretado, interactivo y orientado a objeto, preparado para realizar cualquier tipo de programa, desde aplicaciones Windows a servidores de red o incluso, páginas Web. Es extremadamente poderoso y posee una sintaxis muy clara. Las implementaciones de Python son multiplataformas, actualmente corre en Linux, Unixs, Windows, Mac, entre otras. Python contiene gran cantidad de librerías, tipos de datos y funciones incorporadas al lenguaje que ayudan a realizar muchas tareas habituales sin necesidad de tener que programarlas desde cero.

1.10.2 HTML

HTML es un lenguaje diseñado para estructurar textos para generar páginas Web. Gracias a Internet y a los navegadores Web, el HTML se ha convertido en el formato más fácil para la creación de páginas Web debido a su sencillez. El lenguaje HTML es extensible, se le pueden

añadir características, etiquetas y funciones adicionales para el diseño de páginas Web, generando un producto vistoso, rápido y sencillo.

1.10.3 XML

XML (Extensible Markup Language) es el lenguaje universal de marcado para documentos estructurados y datos en la Web. No solo es un lenguaje de marcado, sino también un metalenguaje que permite describir otros lenguajes de marcado. Es más amplio, más rico y más dinámico que HTML, presenta todas las ventajas del HTML añadiendo características propias como la capacidad de describir de forma estructurada el contenido del documento de manera independiente a la definición del formato de presentación de dicha información. Permite el uso ilimitado de los tipos de datos que pueden utilizarse en Internet. Es con toda probabilidad el futuro estándar no sólo de Internet sino también del intercambio de información.

1.10.4 CSS

CSS (Hojas de Estilo en Cascada) es un lenguaje formal usado para definir la presentación de un documento estructurado escrito en HTML o XML (y por extensión en XHTML). Las CSS son un lenguaje de programación Web usado para dar mayor personalización a las páginas Web y mantener el mismo estilo en múltiples páginas. Usando CSS se puede definir los colores, fondos, bordes, margen, alineaciones, letras, tamaño y muchas cosas más para casi todas las partes de una página Web. Las hojas de estilo en cascada resultan muy prácticas, ya que permiten ahorrar líneas de código y gestionar mejor la presentación de texto.

1.11 Conclusiones

En este capitulo se han definido conceptos fundamentales para la investigación, a partir de su análisis se puede llegar a la conclusión de que las Factorías de Software son las encargadas de industrializar el proceso de producción de software, llevando ese proceso del ámbito artesanal al

industrial. Se analizaron algunos modelos de factoría usados actualmente con el fin de seleccionar el adecuado para ser implantado en la Factoría de Software del Polo Gestión de Proyecto. Se presenta un modelo de replicación de un modelo de factoría de software a través del cual se puede realizar la estrategia de implantación. Se describen, además, algunos estándares de calidad usados actualmente y que ayudan a mejorar el desempeño de la factoría, uno de ellos es el estándar CMMI, modelo de calidad integrado que provee áreas y practicas importantes para el desarrollo y evaluación del proceso de desarrollo y la gestión de proyectos. Otro aspecto tratado fueron las principales metodologías, herramientas y lenguajes de programación utilizados en el Polo para el desarrollo de software.

CAPÍTULO 2: ESTRATEGIA DE IMPLANTACIÓN

2.1 Introducción

La transferencia tecnológica permite asimilar nuevas tecnologías que mejoren la forma de trabajar de las empresas. La implantación de la factoría de software en el Polo Gestión de Proyecto se puede hacer a través de la transferencia tecnológica.

Los aspectos que se deben tener en cuenta para hacer la transferencia tecnológica de la Factoría Software son los siguientes:

- Recopilar la información necesaria que permita implantar la Factoría.
- Diseñar los elementos de la factoría a implantar.
- Gestionar los recursos necesarios para la puesta en marcha de la estrategia.
- Implantar la Factoría de Software en el Polo Gestión de Proyecto.
- Hacer un seguimiento del funcionamiento de la Factoría de Software implantada.
- Emitir un reporte de los resultados obtenidos.

2.2 Recopilar información para implantar la Factoría

Uno de los primeros pasos a la hora de hacer una transferencia tecnológica es recoger toda la información necesaria sobre la nueva tecnológica y los beneficios que trae la misma, además se debe tener un dominio pleno sobre el área donde va a ser implantada y las características de la misma que llevaron al cambio.

La nueva tecnología en este caso está relacionada con las factorías de software. Con el fin de recoger la información necesaria sobre este enfoque, en el capitulo anterior se definieron una serie de aspectos relacionados con el tema.

La estrategia de implantación se propone para el Polo Gestión de Proyecto, por lo que, antes de definir la estrategia para implantar la factoría es necesario conocer como funciona el mismo.

2.2.1 Polo Productivo Gestión de Proyecto.

El concepto de polo aglutina un conjunto de ideas que combinan la producción, investigación y docencia en la Universidad de la Ciencias Informáticas.

El Polo Productivo Gestión de Proyecto perteneciente a la Facultad 3 tiene la siguiente misión:

- Desarrollar software para la gestión de proyectos, la informatización de la infraestructura productiva y del proceso de producción de software.
- Desarrollo sostenido de herramientas para la ayuda a la toma de decisiones.

Este polo cuenta con una línea de desarrollo, el desarrollo de software para la gestión, control y seguimiento de proyectos, la cual tiene tres proyectos:

- Informatización del Convenio Cuba-Venezuela.
- Informatización de la Gestión de la Infraestructura Productiva.
- Plataforma para la gestión integral de un proyecto productivo.

Se realizaron entrevistas a Jefes de proyectos e integrantes del Polo Productivo Gestión de Proyecto para conocer el estado actual del mismo.

Este polo se dedica a la creación de aplicaciones y portales Web. Para la creación de las aplicaciones Web se sigue el flujo de trabajo del Proceso Unificado de Rational (RUP). A continuación se presenta como se realiza este proceso en el Polo.

El proceso de producción de las aplicaciones Web comienza a partir del levantamiento de requisitos y modelamiento del negocio. En esta fase todo depende del trabajo de los analistas, que son los encargados de modelar el sistema y el negocio con el fin de entender lo que se quiere lograr. Las personas a cargo de este proceso se entrevistan con los clientes para obtener toda la información que necesiten, muchas veces esta información no se entiende de la mejor manera y se crea un producto que no cumple todo lo que el cliente espera.

En la fase de análisis y diseño se transforman los requisitos del cliente modelados en el diagrama de casos de uso del sistema a un diagrama de clases teniendo en cuenta el lenguaje de programación. En esta fase se debe ser muy cuidadosos, ya que lo que se implemente depende completamente de la definición del diseño.

En la fase de implementación se debe tener un Plan de integración de construcciones para indicarle a cada programador lo que tiene que hacer en cada iteración, pero esto no se exige, cada programador trabaja por si solo. Los productos son creados casi desde cero, en la mayoría de los casos no se tiene en cuenta la reutilización de componentes, aspecto importante ya que permite reducir el tiempo de desarrollo y aumentar la calidad del producto. Una vez concluida esta fase, las pruebas que se realizan no son tan minuciosas y en muchas ocasiones no se detectan los errores que tiene el producto.

Las personas involucradas con la producción no tiene una buena planificación de sus actividades, tanto personal como en equipo y no tienen bien definidos los roles y responsabilidades que deben cumplir en cada momento.

Por estas razones surge la necesidad de buscar alternativas que permitan mejorar la forma de trabajar del polo y obtener productos de mejor calidad. Una solución es implantar una factoría de

software acorde a las características del Polo Gestión de Proyecto. Aplicar este enfoque aporta beneficios para la producción de software, ya que permite organizar todos los factores implicados en la producción.

En una Factoría existe un grupo de herramientas estandarizadas para soportar el proceso de desarrollo, cada persona tiene definido cual es su rol y las tareas que debe realizar. Aplicar este enfoque permite reducir el tiempo de desarrollo y aumentar la calidad de los productos gracias a la reutilización de componentes.

2.3 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 Polo Productivo Gestión de Proyecto.

Definir una estrategia de implantación del Modelo de Factoría de Software aplicando Inteligencia para el Polo Productivo Gestión de Proyecto implica poner al mismo en condiciones de realizar su trabajo de manera eficaz y eficiente, a través de la organización de los factores implicados en el proceso. Se define el flujo de trabajo, se organizan los roles y se asignan tareas y responsabilidades, se crea un repositorio de componentes y se realiza la gestión de proyecto.

Para implantar la factoría de software se necesita realizar una serie de actividades y sobre todo debe existir motivación por parte de los integrantes de los equipos de desarrollo y de la dirección del polo. Es necesario tener un proceso de producción estandarizado que brinde como resultado un producto.

Como primer paso los procesos que pueden existir en la factoría se pueden agrupar en paquetes según su funcionalidad. El modelo de replicación propuesto en el capitulo anterior se puede usar para transferir todos los paquetes de la factoría (Ver epígrafe 1.7.1).

El proceso industrial de desarrollo de software, correspondiente a la Factoría de Software a ser implantada, origina los paquetes y procesos que se replicarán. Los paquetes son guardados en la base de conocimiento. Esta entidad almacena todo lo referente a la factoría de software a ser implantada.

En el motor de replicación se filtran los paquetes que serán replicados para analizar si sus procesos cumplen con las funcionalidades previstas para la factoría de software. Hecho esto, se configuran los paquetes a ser replicados de forma tal que la implantación de la factoría se haga de una manera más simple. Finalmente se usa una de las técnicas de replicación de conocimiento para transferir todo lo relacionado con la factoría de software a ser replicada (en este caso los paquetes).

Como parte de la estrategia, las entidades que conforman el Modelo de Factoría aplicando Inteligencia se configuran en paquetes que podrían ser replicados por empresas o entidades que deseen aplicar este enfoque. Estos paquetes contendrían procesos definidos teniendo en cuenta las características y objetivos principales de una Factoría de Software.

En la siguiente tabla se pueden ver los paquetes (entidades) correspondientes al Modelo de Factoría de Software aplicando Inteligencia que serán replicados.

Tabla 2. 1 Paquetes que serán replicados en la Factoría de software.

Paquetes	**Proceso**
Proceso de desarrollo	1. Definir la metodología de desarrollo 2. Levantamiento de requisitos y modelamiento del negocio 3. Análisis y Diseño 4. Implementación 5. Soporte y Prueba.

	6. Gestión de la configuración (durante todo el proceso).
Personas	1. Definir la cantidad de personas que tendrá la factoría 2. Definir la estructura organizativa, formar grupos de trabajos y asignar roles y responsabilidades. 3. Impartir cursos de capacitación para que cada persona en la factoría tenga los conocimientos necesarios para un buen desempeño laboral en la misma.
Gestión de Proyecto.	1. Definir los objetivos de los proyectos que lleguen a la factoría 2. Definir alcance, tiempo y costo. 3. Planificar el trabajo de la Factoría 4. Prever posibles riesgos y la forma de mitigarlos. 5. Gestionar la calidad del proyecto.
Bases tecnológicas	1. Identificar las herramientas y tecnologías necesarias para el desarrollo de las aplicaciones. 2. Establecer técnicas de configuración de las herramientas. 3. Verificar el funcionamiento correcto del Hardware y software de la factoría. 4. Definir mecanismos de seguridad de la información.
Repositorio de componentes	1. Hacer catálogo de componentes reutilizables de cada proyecto con su descripción, objetivo, características tecnológicas y ubicación. 2. Crear repositorio de componentes 3. Actualizar repositorio según requerimientos.

Centro de Inteligencia	1. Definir las personas encargadas de realizar la gestión empresarial y del conocimiento, la vigilancia tecnológica, la inteligencia empresarial y la prospectiva.

Una vez definidos los paquetes que serán replicados y teniendo en cuenta que sus procesos cumplen lo que se espera que tenga la factoría, se pasa a configurar cada uno de ellos definiendo sus características más significativas.

2.3.1 Paquete Proceso

Las factorías de software se pueden clasificar de acuerdo al alcance o ámbito de funcionamiento que tienen a lo largo del proceso de desarrollo de software (Ver Anexo 3). El proceso de desarrollo que se realiza en el Polo de Gestión de Proyecto comienza a partir de la entrega de los requisitos para el diseño del sistema e implementación, por lo que, para el Polo de Gestión de proyecto la factoría se clasifica como una Factoría de proyectos físicos, ya que se centra principalmente en el diseño, implementación y prueba, aunque se tiene conocimiento del negocio a automatizar.

Actualmente el proceso de desarrollo de aplicaciones Web en el Polo Productivo Gestión de Proyecto sigue el ciclo de vida del Proceso Unificado de Rational (RUP). Se propone seguir utilizando esta metodología ya que es la metodología líder en el desarrollo de software y puede ser adaptada a las características de cada proyecto. Para organizar el proceso de desarrollo se dividió el mismo en cuatro entidades:

- Levantamiento de requisitos y modelamiento del negocio
- Análisis y diseño
- Implementación

- Soporte y Prueba.

En la entidad Levantamiento de requisitos y modelamiento del negocio se captan las características principales de la aplicación y del entorno donde se va a implantar, además de identificar a quien está dirigida la misma. Luego en el Análisis y diseño se hace una maqueta de cómo quedarían esos requisitos tomando en cuenta el lenguaje de programación que se usará. Una vez hecho esto, los programadores en la Entidad Implementación se encargarían de llevar el diseño a la implementación quedando conformado el producto. Finalmente se le realizarían pruebas para comprobar que el producto cumple con las funcionalidades descritas por el cliente.

Este proceso debe estar coordinado por el Jefe del equipo de desarrollo quien es el encargado dirigir y guiar al Equipo durante la confección del producto y velar por el cumplimiento del trabajo en el tiempo establecido, además de atender sus necesidades. El proceso de desarrollo debe estar definido por el área organización del proceso de la Gestión de proyecto y controlada desde esta.

A continuación se detallará cada Entidad.

2.3.1.1 Levantamiento de requisitos y Modelamiento del negocio

Esta es la fase inicial del proceso productivo en la factoría de software, donde el cliente solicita la creación de una aplicación Web. El Jefe de proyecto junto con los analistas se entrevista con los clientes para determinar el entorno del negocio donde va a funcionar la aplicación, lo que se espera que haga y a quienes está dirigida. Es decir, se hace 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 realizadas en esta entidad son:

- *Capturar requisitos:* a través de las técnicas de recopilación de información (Entrevistas y encuestas) se obtienen los requisitos del sistema.

- *Modelar diagrama de casos de uso del negocio:* Una vez identificado los actores del negocio y los procesos que se realizan en el mismo, se confecciona el modelo de casos de uso representando la relación entre ellos.

- *Modelar sistema:* A partir de la captura de requisitos se obtienen los actores y casos de uso del sistema. Una vez identificados se pasa a modelar el sistema a partir de la relación de estos.

- *Análisis de la arquitectura:* Se analiza cada caso de uso para identificar los que son necesarios para una primera iteración, es decir, se identifican los casos de uso críticos más importantes a tener en cuenta para realizar la aplicación.

Los roles que intervienen son:

- *Analista de procesos de negocio:* es el responsable de la arquitectura del negocio por lo que dirige y coordina el proceso de modelado del negocio. Identifica los actores y procesos del negocio y las reacciones que se establecen entre ellos. Define, además, las reglas del negocio a tener en cuenta.

- *Analista del sistema:* es el responsable del levantamiento de requisito y de modelar el sistema. Delimita el sistema, encuentra los actores y casos de uso y asegura que el modelo de casos de uso del sistema es completo y consistente. Es el que dirige el modelado y la captura de requisitos.

- *Arquitecto:* describe la vista de la arquitectura del modelo de casos de uso definiendo la prioridad de cada caso de uso para decidir en que iteración será desarrollado cada uno. Esta es una entrada importante para planificar las iteraciones.

Los principales artefactos que se obtienen son:

- *Plan de desarrollo del software:* Recoge toda la información requería para dirigir el proyecto. Adjunta varios artefactos desarrollados al inicio y se mantiene durante todo el proyecto. Contempla de forma general los siguientes elementos:
  - Visión del proyecto
    - Propósito y objetivos
    - Entregables
  - Organización del proyecto
    - Estructura
    - Roles y responsabilidades
  - Administración del Proceso
    - Estimaciones del proyecto.
    - Plan del Proyecto.
    - Plan de Iteraciones.
- *Documento Visión:* Especifica las características propias del sistema, funcionalidad y resultados esperados por la Factoría, acuerdos iniciales, trabajadores y sus roles, los objetivos del producto, propósitos, planificación y la captura de requisitos, estudio de factibilidad, etcétera.

- *Modelo de casos de uso del negocio:* Describe los procesos de negocio de una empresa en términos de casos de uso y actores del negocio, que se corresponden con los procesos del negocio y los clientes, respectivamente.

- *Modelo de casos de uso del sistema:* permite que los desarrolladores y los clientes lleguen a un acuerdo sobre las condiciones y posibilidades que debe cumplir el sistema (requisitos).

- *Glosario de términos:* es una lista de términos asociados al negocio y al sistema que son comúnmente usados por los analistas (y otros desarrolladores). Deben ser del dominio del equipo de desarrollo, ya que son importantes para poder modelar el negocio y el sistema.

- *Descripción de la arquitectura (vista del modelo de casos de uso del sistema):* Contiene una vista de la arquitectura del modelo de casos de uso, que representa los casos de uso más significativos para la arquitectura.

- *Prototipo de interfaz de usuario:* Ayudan a comprender y especificar las interacciones entre actores humanos y el sistema durante la captura de requisitos. Ayuda a desarrollar una interfaz grafica mejor y comprender mejor los casos de uso.

En el Anexo 8 se puede ver una tabla descriptiva con el flujo anteriormente mencionado.

2.3.1.2 Análisis y Diseño

Inicialmente el arquitecto debe elaborar el modelo de diseño. Identifica que subsistemas y componentes reutilizables pueden ser esbozados, dejando especificado sus interfaces. Los ingenieros de casos de uso realizan cada caso de uso en términos de clases y/o subsistemas del diseño y sus interfaces. Las realizaciones de los casos de uso resultantes establecen los requisitos de comportamiento para cada clase o subsistema que participe en alguna realización de caso de uso. Los ingenieros de componentes especifican a continuación los requisitos y los

integran dentro de una clase, puede ser mediante la creación de operaciones, atributos y relaciones consistentes sobre una clase o mediante la creación de operaciones consistentes en cada interfaz que proporcione el subsistema.

Las actividades realizadas son las siguientes:

- *Diseñar la arquitectura:* El principal objetivo de la arquitectura en esta etapa inicial del desarrollo es la creación del modelo de diseño. En esta etapa es donde se decide que componentes o librerías brindan mejor servicios en el desarrollo, considerando los servicios que brinda el repositorio. Además, se debe especificar que artefactos se deben utilizar para la modelación en la fase, así como los elementos que se deben usar.

- *Identificar clases de diseño:* en esta tarea se debe identificar las clases del diseño que se necesitan para realizar el caso de uso; para esto se deben estudiar la funcionalidad y los requisitos descritos.

- *Creación del modelo de diseño:* a partir de las clases identificadas y tomando en cuenta el lenguaje de programación a usar se crea el modelo de diseño.

Los roles para el análisis y diseño son los siguientes:

- *Jefe de análisis y diseño:* Es el que lleva la dirección del grupo de análisis y diseño, debe tener una visión global de toda la funcionalidad y forma del sistema. Debe apoyar todas las tareas con el fin de conseguir el máximo de creatividad en esta etapa.

- *Arquitecto:* En esta etapa, el arquitecto es responsable de la estructura del modelo de diseño. Debido a que el trabajo en las factorías es regulado por líneas de desarrollo y tecnologías de implementación bien definidas, la estructura de la arquitectura es estándar en la mayoría de los subsistemas. El trabajo en ese sentido es enfocado principalmente al control de los artefactos utilizados en las diferentes partes de la estructura, teniendo en cuenta los componentes reutilizables existentes en el repositorio, los cuales exigen de un

lugar en esa estructura dada su clasificación, su estilo de adaptación e integración. El modelo de diseño debe adaptarse a los cambios que sufre la tecnología de implementación.

- *Ingeniero de casos de uso:* es el responsable de la integridad de una o más realizaciones de casos de uso y debe garantizar que se cumplen los requisitos que se esperan de ellos. Esto incluye hacer legibles y adecuadas para su propósito todas las descripciones textuales y todos los diagramas que describen la realización del caso de uso.

- *Ingeniero de componente:* Define y mantiene las operaciones, métodos, atributos, relaciones y requisitos de implementación de una o más clases, garantizando que cada una cumple los requisitos que se esperan de ella según las realizaciones de casos de uso en las que participa. Puede mantener también la integridad de uno o más subsistemas, asegurando que sus contenidos son correctos al igual que las dependencias de otros subsistemas y/o interfaces son correctas.

Los artefactos que se generan son los siguientes:

- *Modelo de diseño:* es un modelo de objetos que describe la realización física de los casos de uso centrándose en como los requisitos funcionales y no funcionales, junto con otras restricciones relacionadas con el entorno de implementación, tienen impacto en el sistema a considerar.

- *Realizaciones de casos de uso:* es una colaboración en el modelo de diseño que describe como se realiza un caso de uso específico, y como se ejecuta en términos de casos de uso del diseño. Una realización de caso de uso del diseño tiene una descripción de flujos de eventos textual, diagramas de clases que muestra sus clases de diseño participantes y diagramas de interacción que muestran la realización de un flujo o escenarios concretos de un caso de uso en términos de interacción entre objetos del diseño. La realización de

casos de uso del diseño se efectúa a través de una colaboración en la cual intervienen el diagrama de clases y los diagramas de interacción.

- *Diagrama de clases:* muestra un conjunto de clases, interfaces y colaboraciones, así como sus relaciones. Los diagramas de clases se utilizan para modelar la vista de diseño estática de un sistema. Principalmente, esto incluye modelar el vocabulario del sistema, modelar las colaboraciones o modelar esquemas. Se deben dividir y detallar cuidadosamente las clases según su rol en la aplicación, ya sean clases de interfaz, control o entidad.

- *Diagramas de interacción:* Un diagrama de interacción muestra una interacción que consiste en un conjunto de objetos y sus relaciones, incluyendo los mensajes que se pueden enviar entre ellos. Los diagramas de secuencia y los diagramas de colaboración son diagramas de interacción. Los diagramas de interacción se utilizan para modelar los aspectos dinámicos de un sistema.

- *Subsistema del Diseño:* es una forma de organizar los artefactos del modelo de diseño en piezas más manejables. Puede constar de clases del diseño, realizaciones de casos de uso, interfaces y otros subsistemas. Pueden proporcionar interfaces que representan la funcionalidad que exportan en términos de operaciones.

En el Anexo 9 se puede ver una tabla descriptiva con la Entidad anteriormente mencionado.

2.3.1.3 Implementación

Esta es la entidad que más carga de trabajo tiene, es donde concretamente se construye el producto.

El flujo de trabajo se inicia implementando la arquitectura básica por el arquitecto del proyecto, dejando esbozado los componentes claves del modelo de diseño, si es necesario incluir

componentes del repositorio, este se le solicita al Administrador del repositorio, colocándolo a la vez en la arquitectura. A continuación el Jefe de implementación 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 programadores, estos deben implementar y controlar la calidad de la parte en que trabajen. Al finalizar la tarea correspondiente solicitan la revisión por parte del jefe de programación y del responsable de calidad. Finalmente el Jefe de implementación (junto a algunos programadores en caso necesario) es responsable de la unión de las partes para conformar el producto final. Integrado el producto, se envía a la entidad de soporte y prueba.

Las actividades que se realizan son las siguientes:

- *Implementación de la arquitectura:* Primeramente se deben identificar los componentes significativos dentro de la arquitectura, tales como componentes ejecutables o reutilizables.

- *Planificar implementación:* En la planificación se debe verificar que los componentes reutilizables estén bien diseñados. Se deben tener en cuenta todos los requisitos modelados a través de casos de uso de diseño. Los resultados deben estar especificados en el plan de ensamblado y comunicados a los programadores encargados de cada parte.

- *Implementar clase:* La implementación de una clase consiste en implementar una clase de diseño en un componente de fichero. Esto incluye lo siguiente:

  - Esbozo de un componente de fichero que contendrá el código fuente.

  - Generación del código fuente a partir de la clase de diseño y de las relaciones en que participa.

  - Implementación de las operaciones en las clases de diseño en forma de métodos.

- Comprobación de que el componente proporciona las mismas interfaces que las clases de diseño.

- Reparación de defectos de la clase, durante la pruebas de la clase.

- *Generación de código a partir del modelo de diseño:* al elaborar la estructura de un modelo de código, esta debe estar sincronizado con el modelo de diseño. El código fuente debe generarse a partir de la herramienta de modelado que se esté utilizando.

- *Realizar pruebas de unidad:* En esta fase es fundamental realizar pruebas de unidad, las cuales tienen como propósito probar los componentes implementados como unidades independientes. Dentro de estas pruebas están las pruebas de especificación y las pruebas de estructura.

  - Las pruebas de especificación o "pruebas de caja negra" verifican el comportamiento del componente externamente. Para esto es necesaria la implementación de clases de pruebas que contengan un conjunto de valores de entrada, estado o salida; para los que se supone que un componente se debe comportar.

  - Las pruebas de estructura o "pruebas de caja blanca" se realizan con el propósito de verificar que un componente funciona internamente como se quería. O sea, se debe intentar que en cada algoritmo se evalúe todos los caminos posibles a recorrer.

- *Integrar producto:* Se integran los componentes realizados por los programadores al producto final.

- *Controlar Calidad:* El control de la calidad en el proceso productivo estará enfocado al producto final. Cada vez que se implemente un componente, antes de sincronizar con el

SVN se debe chequear la calidad y registrar en el cuaderno de registro de defectos los errores encontrados.

Los roles que intervienen son los siguientes:

- *Jefe de implementación:* El Jefe de implementación necesita garantizar que los contenidos y elementos dentro de los subsistemas de diseño se corresponden con los elementos físicos de la estructura de implementación; que sus dependencias con otros subsistemas o interfaces son correctos, y que implementan correctamente las interfaces que proporcionan. Debe planificar la implementación y dejarla registrada en el Plan de integración de construcciones. Deben tener participación en la fase de análisis y diseño para en la fase de implementación tener una visión de cada artefacto de implementación a utilizar y poder dirigir la fase con mejor claridad.

- *Arquitecto:* Es responsable de la integridad del modelo de implementación y asegura que el modelo como un todo es correcto, completo y legible. Se considera que la estructura es correcta cuando implementa la funcionalidad descrita en el modelo de diseño y en los requisitos adicionales especificados en esa fase. Además debe colocar en la estructura de directorio los componentes reutilizables especificados en el diseño. El arquitecto debe garantizar la sincronización entre el modelo de diseño y la estructura de implementación, asegurando que cada cambio o nueva idea arquitectónica generada y aprobada en la implementación, sea correspondida en el diseño.

- *Responsable de calidad:* Es responsable de controlar la calidad en ciertos momentos del proceso de desarrollo. Se encarga de controlar que cada desarrollador cumpla con las normas establecidas, estableciendo mecanismo que logren que la calidad sea una tarea de todos.

- *Programadores:* Responsables de los componentes que le sean asignados por el jefe de implementación, llevando a código fuente las especificaciones del diseño correspondiente.

Los artefactos que se deben crear son los siguientes:

- *Modelo de implementación:* describe cómo los elementos del modelo de diseño se implementan en términos de componentes, como se organizan estos de acuerdo con los mecanismos de estructuración y modularización disponibles en el entorno de implementación y en el lenguaje o lenguajes de implementación utilizados y como dependen los componentes unos de otros.

- *Plan de integración de construcciones:* Describe la secuencia de construcciones necesarias en una iteración. Describe la funcionalidad que se espera sea implementada en dicha construcción y las partes del modelo de implementación que están afectadas por la construcción.

- *Solicitud de Repositorio:* consiste en la descripción formal del recurso del repositorio a utilizar, ya sean componentes de código, patrones, algoritmos, formularios o documentación.

En el Anexo 10 se puede ver una tabla descriptiva con el flujo anteriormente mencionado.

2.3.1.4 Soporte y Prueba.

En las factorías de software, a pesar de que se busca la especialización de los trabajadores en un rol especifico, la posibilidad de que se comentan errores es significativa, por lo que se hace necesario comprobar que los productos cumplen con lo especificado.

Las pruebas juegan un papel fundamental en la garantía de la calidad, representan una revisión de las especificaciones, del diseño y la codificación. Para reducir los errores en la construcción de software se hace necesario crear pruebas minuciosas y bien planificadas.

El flujo de trabajo de soporte y prueba comienza cuando llega al equipo de prueba la lista de los requisitos funcionales, el modelo de diseño, el resultado de la implementación del producto a

probar y la descripción de la arquitectura, elementos necesarios para que el ingeniero de prueba inicie la planificación de las pruebas dejando elaborado el Plan de prueba. Después de esto diseña los casos de prueba y procedimientos necesarios para realizar las pruebas. El probador utiliza los casos de prueba y los procedimientos para ejecutar las pruebas, detectando errores que pueden existir. Los errores que se detecten pasan al ingeniero de pruebas para registrarlo en el seguimiento y reporte de defectos, y enviarlos a flujos anteriores (diseño e implementación) para que sean corregidos. El ingeniero de prueba hace una valoración general del resultado de las pruebas realizadas.

Se realizan las siguientes actividades:

- *Planificar prueba:* en esta actividad el ingeniero de prueba realiza una planificación de las pruebas que se realizarán. Los requerimientos adicionales, el modelo de diseño, el modelo de implementación y la descripción de la arquitectura, son los elementos de entrada para elaborar el plan de prueba.

- *Diseñar prueba:* En el diseño de las pruebas el ingeniero de prueba se encarga de elaborar los casos de prueba y los procedimientos de prueba que especifican como realizar estos casos de prueba. Para realizar el diseño de la pruebas el ingeniero de prueba se basa en los requisitos adicionales, en el modelo de diseño, en el modelo de implementación, en la descripción de la arquitectura y en el plan de prueba.

- *Ejecutar pruebas:* En esta actividad se realizan las pruebas planificadas y diseñadas por el ingeniero de prueba. El probador ejecuta las pruebas realizando los procedimientos correspondientes a cada caso de prueba, se comparan los resultados de las pruebas con los resultados esperados y se analizan los que no coincidan, además de informarle de los defectos encontrados al ingeniero de prueba para que este los registre y los envié a flujos anteriores para ser corregidos.

- *Evaluar pruebas:* El ingeniero de prueba evalúa los resultados de las pruebas comparando los resultados con los objetivos esbozados en el plan de prueba, documenta los resultados obtenidos y sugiere nuevas acciones si fuera necesario en el documento de evaluación de pruebas.

- *Reportar defectos:* en esta actividad el ingeniero de prueba realiza el reporte de los defectos encontrados a los flujos anteriores para que sean corregidos.

Con solo dos roles se pueden realizar estas tareas. Estos roles pueden ser desempeñado por una o más personas en dependencia de la magnitud de la aplicación, estos son:

- *Ingeniero de prueba:* es el encargado de planificar y diseñar las pruebas a realizar, además de evaluar el resultado de las mismas.

- *Probador:* es el encargado de ejecutar las pruebas diseñadas por el ingeniero de prueba, guiándose por el procedimiento de prueba.

Los artefactos que se proponen que sean creados en esta entidad son los siguientes:

- *Modelo de prueba:* es una colección de casos de prueba, procedimientos de prueba y componentes de prueba que describe como deben ejecutarse las pruebas a aspectos específicos del software.

- *Plan de prueba:* Describe las estrategias, recursos y planificación de la prueba. La estrategia de prueba incluye la definición del tipo de prueba a realizar para cada iteración y sus objetivos, el nivel de cobertura de prueba y de código necesario y el porcentaje de pruebas que deberían ejecutarse con un resultado específico.

- *Casos de prueba:* especifican la forma de probar los componentes, incluyendo la entrada o resultados con lo que ha de probarse y las condiciones bajo las que ha de probarse.

- *Procedimiento de prueba:* especifica como realizar uno o más casos de prueba. Es útil reutilizar un procedimiento de prueba para varios casos de prueba y varios procedimientos para un mismo caso de prueba.

- *Evaluación de prueba*: es la evaluación del resultado de las pruebas, tales como la cobertura del caso de prueba, la cobertura de código y el estado de los defectos.

- *Reporte de defectos:* En el reporte de defectos es donde el ingeniero de prueba registra los defectos encontrados para enviarlos a los flujos anteriores (diseño o implementación).

En el Anexo 11 se puede ver una tabla descriptiva con el flujo anteriormente mencionado.

Concluida la revisión, el equipo de desarrollo corrige los errores reflejados en el reporte de defectos. Hecho esto, el producto es enviado nuevamente al equipo de prueba para una nueva revisión. El equipo de prueba analiza nuevamente el producto para encontrar nuevas inconformidades. Esto se hace mientras se encuentren errores.

Cuando se ha concluido el producto y se han hecho varias revisiones para verificar que no contenga errores, se envía el producto al equipo de calidad de la línea correspondiente para una nueva revisión, en caso de encontrarse errores se regresa para su corrección. Si se determina que está listo para su salida, se le entrega una versión al cliente para que lo evalúe, en caso de tener inconformidad con el producto se regresa para su corrección. En caso contrario, el cliente firma los documentes pertinentes donde expresa su conformidad con el producto y se cierra el contrato. En este proceso participan el Gestor de la Factoría, el Jefe de la línea y el Jefe del Equipo de desarrollo correspondiente, además de los clientes.

2.3.1.5 Gestión de configuración

La gestión de configuración es el arte de identificar, organizar y controlar las modificaciones que sufre el software que construye un equipo de desarrollo. Tiene como objetivo:

- Establecer y mantener la integridad de los productos generados durante un proyecto software y a lo largo de todo el ciclo de vida del mismo.
- Evaluar y controlar los cambios que se producen sobre el producto.
- Facilitar la visibilidad sobre el producto.

Para conseguir estos objetivos se realizan las siguientes actividades:

- Identificación de la configuración: consiste en identificar la estructura del producto, sus componentes y el tipo de estos, y en hacerlos únicos y accesibles de alguna forma.
- Control de cambio en la configuración: consiste en controlar las versiones y entregas de un producto y los cambios que se producen en él a lo largo del ciclo de vida. Para el control de cambio se recomienda trabajar con Subversion.
- Generación de informes de estado: consiste en informar acerca del estado de los componentes de un producto y de las solicitudes de cambio, recogiendo estadísticas acerca de la evolución del producto.
- Auditorías de la configuración: consiste en validar la completitud de un producto y la consistencia entre sus componentes, asegurando que el producto es lo que el usuario quiere.

El administrador del repositorio es el encargado de crear y configurar el ambiente para la gestión de la configuración, ejecutar las auditorias y elaborar el plan de Gestión de la configuración. Además de controlar los cambios que se producen a los elementos de configuración durante el desarrollo del producto.

2.3.2 Paquete Personas

Las personas constituyen un factor importante en el éxito de un proyecto de software por lo que su organización es fundamental. La Factoría presenta una estructura organizativa donde cada persona involucrada ocupa un rol en dependencia de sus conocimientos. La misma se divide en Grupo de desarrollo y Gestores de la factoría.

2.3.2.1 Grupo de desarrollo

Cuando se inicia un nuevo proyecto en la Factoría, como primer paso se debe hacer una estimación de la cantidad de personas que tendrá el equipo de desarrollo. La cantidad de integrantes estará en dependencia de la carga de trabajo inicial y el tamaño del producto a desarrollar. Para la selección de las personas involucradas en la producción en cada equipo de desarrollo se propone utilizar el proceso definido por la Facultad 3 (Ver Anexo 12). Una vez seleccionados los integrantes de cada proyecto, se firma el compromiso de ética mediante el cual se asegura el compromiso de cada uno a la protección de la información, recursos y demás medios puestos al servicio del desarrollo del proyecto.

Estas personas deben estar organizadas en grupos de trabajo que permitan integrar esfuerzo por encima de proyectos individuales para el logro de un objetivo común, mejorando las operaciones relativas al software.

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[12] hace un análisis de varios organigramas para equipos de desarrollo de software, estos son: descentralizado democrático (DD), descentralizado controlado(DC) y centralizado controlado(CC).

Tabla 2. 2 Tipos de Organigramas básicos para el trabajo en equipo.

	Descentralizado democrático	**Descentralizado controlado**	**Centralizado controlado**
Descripción	No existe un jefe permanente, se asignan coordinadores de tareas a corto plazo. Las decisiones se toman por consenso del grupo	Existe un jefe bien definido y jefes de equipos con responsabilidades sobre subtareas. Las decisiones se toman por consenso del grupo pero las actividades y responsabilidades se asignan a los subgrupos	Existe un jefe bien definido que controla los problemas a alto nivel y también las decisiones internas.
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

Capítulo II: Estrategia de Implantación

Teniendo en cuenta que la factoría de software será aplicada al Polo Productivo Gestión de Proyecto se propone utilizar el organigrama DC ya permite dividir el trabajo entre pequeños equipos, conformando subgrupos. En este tipo de organización la comunicación fluye tanto horizontal como vertical, algo importante en la factoría de software, ya que los miembros del equipo de desarrollo necesitan comunicarse y compartir experiencias y que sus criterios se tengan en cuenta por la dirección de la factoría. Ofrece mejores resultados para equipos que estarán gran cantidad de tiempo trabajando juntos, como es el caso de los equipos de desarrollo que se crean en el polo.

La factoría puede trabajar en varias líneas de desarrollo a la vez (actualmente una sola), por lo que en cada línea existirán tantos equipos de desarrollo como sean necesarios según la cantidad de trabajo que posea la factoría en esa línea. Para que 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.

El estilo de dirección adoptado para el control de las normas de funcionamiento basadas en trabajos investigativos realizados se propone que sea autocrático, pues las mismas son las que guían a los participantes de la factoría a cumplir los objetivos estratégicos planteados.

Los participantes deben tener libertad de generar ideas e intercambiar opiniones, por lo que en la realización de las tareas por parte de cada rol definido en la factoría se debe implantar un estilo democrático. Este estilo aumenta el compromiso del equipo y posibilita la toma de buenas decisiones, ya que refleja varios puntos de vista.

El estilo de dirección empleado tiene una gran influencia sobre los resultados a obtener, la clave está en saber cual aplicar según las circunstancias.

2.3.2.2 Gestores de la factoría

La factoría debe contar con un equipo de dirección encargado de dirigir todas las acciones que se realicen en la misma.

La estructura de dirección estaría compuesta por: un equipo de asesoría y servicio a la producción; líneas de desarrollo definidas, un grupo de calidad y equipos de desarrollo software, donde cada equipo tiene un responsable. El responsable del equipo de asesoría y servicio, junto a los responsables de las líneas de producción se subordinaría a la dirección general de la factoría; el responsable del equipo de desarrollo junto al responsable del grupo de calidad se subordinaría al jefe de su línea correspondiente. Podrían existir tantos equipos de desarrollo como demanda de trabajo haya por parte del cliente.

Se identifican los siguientes roles:

- *Gestor de la Factoría:* Es el encargado de dirigir y apoyar las distintas líneas de desarrollo de la factoría. Debe ser capaz de en conjunto con la dirección de cada línea de desarrollo estimar plazos, costos y riesgos para la realización de un determinado producto software, estableciendo las relaciones necesarias con el cliente. Además controlar el desarrollo de los distintos productos, debe verificar y tener noción de la fase en que se encuentran. Administrará con economía y eficacia los distintos recursos y procesos de la factoría, debe organizar el trabajo de forma tal que las personas bajo su mando cumplan su rol adecuadamente.

Por cada línea de desarrollo dentro de la factoría habrá un equipo de dirección de la misma donde se identifican los siguientes roles:

- *Jefe de línea de desarrollo:* Es el encargado de dirigir el desarrollo de productos software en una línea determinada dentro de la factoría. Tiene la tarea de velar por que los productos en su línea de desarrollo se realicen a tiempo y según las especificaciones

establecidas por el cliente, realizando la recepción de un producto a desarrollar, controlando su estado de ejecución, enviando el entregable al cliente y almacenándolo según las reglas establecidas. Tiene bajo su mando a uno o varios equipos de desarrollo, es el intermediario entre estos y el jefe de la factoría, constituye el canal a través del cual fluye la información desde los directivos hacia el equipo de desarrollo. Este cargo lo realizan uno o varias personas en dependencia de la cantidad de líneas de desarrollo que tenga la factoría.

- *Jefe del grupo de calidad:* Es el encargado de dirigir la gestión de la calidad en una línea determinada de la factoría. Tiene la tarea de velar por que los productos en su línea de desarrollo se realicen con la calidad requerida coordinando la realización de pruebas y revisiones a los diferentes productos. Tiene bajo su mando a varias personas encargadas de gestionar la calidad de los productos que se hacen en la línea. Este cargo lo realizan una o varias personas en dependencia de la cantidad de líneas de desarrollo que tenga la factoría.

El equipo de asesoría estaría compuesto por:

- *Jefe de grupo de asesoría y servicio:* Guía y apoya al grupo de soporte de la Factoría, responde por este grupo ante la dirección de la misma. Es el encargado de identificar las necesidades de cada grupo de desarrollo con el fin de poner a los especialistas en función de las mismas.

- *Administrador del repositorio de componentes reutilizables*: Es el responsable de gestionar el repositorio de componentes reutilizables de la Factoría.

- *Investigador:* es el encargado de 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.

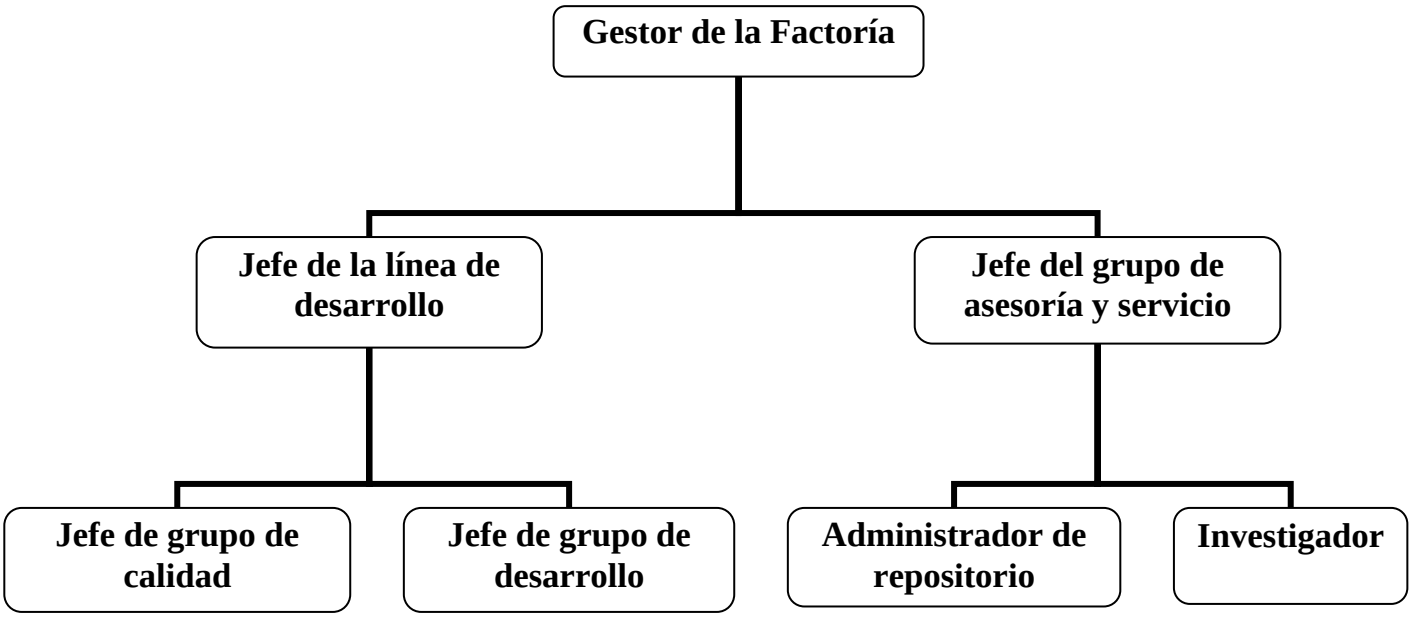

Figura 2. 1 Estructura de dirección de la Factoría

En el Anexo 13 se puede ver una tabla con la responsabilidad, habilidades, conocimientos y valores que debe tener cada rol en la factoría.

2.3.2.3 Uso de PSP

Existen instrumentos de PSP que pueden ayudar en gran magnitud al trabajo organizativo de cada individuo en la factoría; teniendo en cuenta que el personal de la misma es joven e inexperto en desarrollo de software. Además de tener otras responsabilidades adicionales al desarrollo de software. 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. El cuaderno del ingeniero es una agenda de trabajo que todo

ingeniero debería tener para controlar las actividades que realiza en su horario laboral. El diseño del cuaderno se puede observar en el Anexo 14.

- *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, y dejar almacenado el historial de las mismas. Ayuda al individuo a controlar la duración y frecuencia de las interrupciones, permitiendo mejorar la calidad y eficiencia de su trabajo. El diseño del cuaderno se puede ver en el Anexo 15.

- *Resumen semanal de actividades***:** agrupa las actividades por días, obteniendo totales de tiempo relacionados con los trabajos efectuados diariamente y de los trabajos efectuados en la semana en general (Ver Anexo 16).

- *Cuaderno de trabajo:* sirve para registrar datos de tiempos estimados y reales. Con los datos almacenados los trabajadores de la factoría pueden comenzar a estimar su velocidad mínima, media y máxima ante los tareas a elaborar en dependencia de su tipo. Esto le permite al Jefe de Equipo poder asignar las tareas de una forma más organizada, ya que tendría un control del tiempo aproximado que le lleva a cada cual realizar una tarea especifica. El diseño del cuaderno se puede ver en el Anexo 17.

- *Cuaderno de registro de defectos:* ayuda a reunir datos de defectos. Este cuaderno es importante para saber los errores que más se cometen y en que fase de desarrollo se introducen, permitiendo en un futuro tener en cuenta los mismos. El diseño del mismo se puede ver en el Anexo 18.

Estos cuadernos ayudan a cada ingeniero a organizar y controlar el trabajo que realiza durante su jornada laboral, permitiéndole tener un control de las principales tareas que debe realizar para desarrollar un determinado producto de la mejor forma posible.

2.3.2.4 Capacitación del personal

La factoría de software debe garantizar que sus trabajadores cuenten con los conocimientos necesarios para realizar su trabajo eficientemente. Con este fin, se deben impartir cursos de capacitación que abarquen desde el uso de PSP hasta el trabajo con las herramientas y metodologías de desarrollo.

2.3.3 Paquete Bases tecnológicas

La entidad Bases tecnológicas dentro de una factoría de software comprende el contexto de las bases tecnologías y herramientas, las técnicas y mecanismos para construir, soportar y gestionar el proceso de desarrollo. Este proceso es automatizado y soportado por estas herramientas y técnicas representadas en la entidad.

En la organización de una factoría de software debe conocerse fundamentalmente que utilidad presenta cada herramienta, y consumo de recursos en la estación de trabajo. Además se debe crear una cultura de cumplimiento a totalidad de todas las normas disciplinarias y técnicas especificadas por los directivos.

Cada vez que a la factoría se incorpore un nuevo integrante, deberá estar ya relacionado con el uso de las herramientas y las normas establecidas en el proceso. Se deben implantar mecanismos para evaluar el verdadero conocimiento de los trabajadores de la factoría y así evitar falsas suposiciones que pueden traer inconsistencias en los tiempos de entrega, por diferencias en la velocidad de aprendizaje.

2.3.3.1 Tecnologías y herramientas.

La factoría debe tener establecido cual es la plataforma, lenguaje de programación y el grupo de herramientas a utilizar. Dado que en el Polo se trabaja por Windows y Linux, se proponen herramientas que generalmente permiten el trabajo en ambos sistemas operativos. Para la

Factoría del Polo Productivo Gestión de Proyecto las herramientas que se proponen son las siguientes:

Herramienta de gestión de proyecto: Trac.

Trac es una herramienta open source con una interfaz Web muy simple que integra herramientas básicas para comunicación, gestión y seguimiento de proyectos. Posee una wiki que es un sitio Web colaborativo que puede ser editado por varios usuarios. Los usuarios de una wiki pueden crear, editar, borrar o modificar el contenido de una página Web de una forma interactiva, fácil y rápida; dichas facilidades hacen de una wiki una herramienta efectiva para la escritura colaborativa. Su interfaz de Subversion permite la gestión de versiones. Cuenta con opciones para seguimiento de hitos, eventos y evolución del desarrollo del proyecto de software y un sistema de tickets para gestión de bugs, tareas y otras incidencias.

Herramientas de Gestión de Configuración: Subversion.

Subversion, también conocido como SVN, es un controlador de versiones empleado en la administración de archivos utilizados en el desarrollo de software o contenido, diseñado como sustituto competitivo de CVS en la comunidad de código abierto. Subversion gestiona los cambios que sufren directorios y ficheros en el tiempo, lo que permite recuperar o volver a versiones antiguas fácilmente, visualizar las modificaciones hechas en el tiempo y realizar copias de seguridad. Permite, además, acceder al repositorio donde se encuentran los directorios y ficheros a través de Internet, esto permite que varias personas puedan trabajar en un mismo fichero de forma colaborativa.

Herramienta de modelado: ArgoUML

ArgoUML es una herramienta profesional de Modelado con UML. Definida en el Capítulo 1.

Requerimientos mínimos de sistema:

- Cualquier sistema operativo que soporte Java
- 10 MB de espacio libre en el disco duro.
- Java 2 JRE o JDK versión 1.4 o superior.

Herramientas para el desarrollo de componentes de código: Eclipse

La plataforma Eclipse es un entorno integrado de desarrollo (IDE). Definida en el capitulo 1.

Requerimientos mínimos de sistema:

- Procesador de 200 MHz o superior
- 256 MB de memoria RAM

Servidor de aplicaciones: Zope

Zope es un servidor de aplicaciones Web, definida en el capitulo 1.

Requerimientos del Sistema:

- 16 MB free RAM
- Procesador de 166 MHz o superior
- 20 MB de espacio en disco

2.3.3.2 Mecanismos de seguridad de la información

En toda organización es importante establecer medidas que posibiliten la seguridad de la información que fluye tanto interna como externamente a la factoría.

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 (Ver Anexo 19).

2.3.3.3 Mecanismos de control de la calidad

Los productos terminados deben cumplir las especificaciones de calidad, para esto son útiles las listas de chequeo. Estas listas especifican una serie de puntos a evaluar durante todas las fases del desarrollo con el objetivo de hacer el menor número de correcciones posibles al final (Ver Anexo 20).

2.3.4 Paquete Gestión de proyecto.

La gestión de proyectos es la disciplina encargada de organizar y administrar recursos de manera tal que se pueda culminar todo el trabajo requerido en el proyecto dentro del alcance, el tiempo, y los costos definidos.

2.3.4.1 Gestión del alcance

La gestión del alcance se ocupa de que en el proyecto se realice el trabajo necesario para cumplir los objetivos planteados al inicio del proyecto. Para esto se debe definir y planear el alcance del proyecto, para verificar que se está cumpliendo y eventualmente cambiarlo.

Los procesos que se deben realizar son las siguientes:

- **Planificación del Alcance:** crear un plan de gestión del alcance del proyecto que refleje cómo se definirá, verificará y controlará el alcance del proyecto, y cómo se creará y definirá la Estructura de Desglose del Trabajo (EDT).

- **Definición del Alcance**: desarrollar un enunciado del alcance del proyecto detallado como base para futuras decisiones del proyecto.
- **Crear Estructura de Desglose del Trabajo (EDT):** subdividir los principales productos entregables del proyecto y el trabajo del proyecto en componentes más pequeños y más fáciles de manejar.
- **Verificación del Alcance:** formalizar la aceptación de los productos entregables completados del proyecto.
- **Control del Alcance**: controlar los cambios en el alcance del proyecto.

Estos procesos interaccionan entre sí y también con los procesos de las demás Áreas de Conocimiento.

En el Anexo 21 se puede ver una tabla descriptiva con estos procesos.

2.3.4.2 Gestión del tiempo

La gestión del tiempo tiene como objetivo asegurar que el proyecto se realice en el tiempo planificado, logrando el alcance del proyecto en tiempo, costo y calidad requerida por el cliente, sin rebasar los riesgos inherentes al 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 las cantidades de recursos necesarios para realizar cada actividad del cronograma.
- **Estimación de la Duración de las Actividades:** estima la cantidad de períodos laborables que serán necesarios para completar cada actividad del cronograma.
- **Desarrollo del Cronograma:** analiza las secuencias de las actividades, la duración de las actividades, los requisitos de recursos y las restricciones del cronograma para crear el cronograma del proyecto.
- **Control del Cronograma:** controla los cambios del cronograma del proyecto.

En el Anexo 22 se puede ver una tabla descriptiva con estos procesos.

Para hacer una buena estimación del tiempo, se debe tener un control de todas las actividades que se realizan en el proyecto y el tiempo en el que estas se deben cumplir, estableciendo hitos que marquen el fin de cada actividad. Se propone trabajar con Trac para planificar y tener un control de las actividades en el proyecto.

2.3.4.3 Gestión del costo

La gestión del costo tiene como objetivo que el proyecto se complete con el presupuesto inicialmente aprobado.

Los procesos que hay que realizar son los siguientes:

- **Estimación de Costo:** desarrollar una aproximación de los costos de los recursos necesarios para completar las actividades del proyecto.
- **Preparación del Presupuesto de Costo**: sumar los costos estimados de actividades individuales o paquetes de trabajo a fin de establecer una línea base de costo.

- **Control de Costo:** influir sobre los factores que crean variaciones del costo y controlar los cambios en el presupuesto del proyecto.

En el Anexo 23 se puede ver una tabla descriptiva con 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.

2.3.4.4 Control de la calidad

La gestión de la calidad tiene como objetivo que el proyecto satisfaga las necesidades para las que fue inicialmente diseñado. Para ello se debe planear, asegurar y controlar la calidad del proyecto en todo momento, respecto a esas necesidades.

Los procesos que hay que realizar son los siguientes:

- **Planificación de Calidad**: identificar qué normas de calidad son relevantes para el proyecto y determinando cómo satisfacerlas.

- **Realizar Aseguramiento de Calidad:** aplicar las actividades planificadas y sistemáticas relativas a la calidad, para asegurar que el proyecto utilice todos los procesos necesarios para cumplir con los requisitos.

- **Realizar Control de Calidad:** supervisar los resultados específicos del proyecto, para determinar si cumplen con las normas de calidad relevantes e identificar modos de eliminar las causas de un rendimiento insatisfactorio.

En el Anexo 24 se puede ver una tabla descriptiva con estos procesos.

La Factoría debe tener un equipo de calidad y un jefe que serían los encargados de implantar la norma ISO 9001, ya que es un modelo que cubre todas las áreas de desarrollo de un producto software. Esta norma exige que las políticas de calidad estén definidas y sean cumplidas por todos los trabajadores de la factoría (Ver anexo 25).

Definida la política de calidad, el equipo de calidad se encargaría de verificar que las mismas se cumplan durante todo el ciclo de desarrollo. Se deben realizar pruebas para verificar que se cumplen los estándares de calidad establecidos y que los productos cumplen los requerimientos del cliente.

2.3.4.5 Gestión de riesgo

La gestión de riesgo se encarga de identificar, analizar y dar respuesta a los riesgos que ponen en peligro el desempeño del proyecto.

Los procesos que hay que realizar son los siguientes:

- **Planificación de la Gestión de Riesgos:** decidir cómo enfocar, planificar y ejecutar las actividades de gestión de riesgos para un proyecto.
- **Identificación de Riesgos**: determinar qué riesgos pueden afectar al proyecto y documentar sus características.
- **Análisis Cualitativo de Riesgos**: priorizar los riesgos para realizar otros análisis o acciones posteriores, evaluando y combinando su probabilidad de ocurrencia y su impacto.
- **Análisis Cuantitativo de Riesgos:** analizar numéricamente el efecto de los riesgos identificados en los objetivos generales del proyecto.
- **Planificación de la Respuesta a los Riesgos:** desarrollar opciones y acciones para mejorar las oportunidades y reducir las amenazas a los objetivos del proyecto.

- **Seguimiento y Control de Riesgos:** realizar el seguimiento de los riesgos identificados, supervisar los riesgos residuales, identificar nuevos riesgos, ejecutar planes de respuesta a los riesgos y evaluar su efectividad a lo largo del ciclo de vida del proyecto.

En el Anexo 26 se puede ver una tabla descriptiva con estos procesos.

Los riesgos tienen que estar correctamente identificados, así como los mecanismos de respuestas pertinentes. Para mitigar estos riesgos se crean los planes de contingencia y mitigación. Algunos riesgos que pueden existir se pueden ver en el Anexo 27.

La gestión de tiempo y costo la debe realizar el planificador, el que estaría a cargo además de gestionar los riesgos en conjunto con el Jefe de proyecto.

2.3.4.6 Jefe de proyecto.

Para que un proyecto pueda ser dirigido y gestionado adecuadamente es imprescindible designar un jefe de proyecto con las capacidades y características idóneas para garantizar el logro de los objetivos establecidos. El jefe de proyecto tiene un papel decisivo en la planificación, ejecución y control del proyecto, debe impulsar el avance del mismo mediante la toma de decisiones orientadas a cumplir los objetivos.

2.3.5 Paquete Repositorio de componentes

En el enfoque de factoría de software la reutilización tiene un papel fundamental en el desarrollo de software dado las ventajas que trae consigo:

- Aumento de la productividad
- Incremento de la calidad.
- Reducción del riesgo

El repositorio de componentes constituye el almacén de componentes reutilizables de la Factoría, por lo que debe ser mantenido y gestionado constantemente.

El repositorio de componentes reutilizables puede contener dos grandes grupos:

- Componentes de código: puede ser una clase, un procedimiento o función, un módulo, un subsistema o una aplicación.
- Activos del proceso: pueden ser patrones de diseño, algoritmos, un esquema de una base da datos, manuales y documentación o un modelo.

Cuando se guarda un componente es necesario que estén bien documentados y clasificados, esto es importante a la hora de buscar un componente de acuerdo a necesidades específicas.

Para reducir el costo de encontrar los componentes adecuados en el repositorio existen las técnicas de clasificación y recuperación de componentes en un repositorio, las cuales son llevadas a un sistema que automatice los procesos.

Las técnicas de clasificación y recuperación de componentes en un repositorio son: hojear (Browsing), hypermedia o por recuperación lineal.

Debe existir un repositorio central del Polo y otro ubicado en el mismo laboratorio donde se esté trabajando con el fin de evitar pérdidas y tener la información en varios lugares por si existen problemas de conectividad. Dado que la estrategia se enmarcará principalmente en la fase inicial de la factoría, se propone usar el método hojear para la organización de los componentes en el repositorio si la cantidad de componentes es pequeña. A medida que se vayan elaborando nuevos elementos hay que elaborar un estudio para mejorar el sistema, se podrá usar el método hypermedia o recuperación lineal en dependencia de la necesidad de la factoría en ese momento.

Como primer paso se debe hacer un catálogo con los componentes reutilizables que tiene cada proyecto existente en el polo, con su descripción, objetivo, características tecnológicas y ubicación, de forma que permita tener un control de estos elementos. Hecho esto, se debe crear el repositorio y guardar los componentes con que se cuenta hasta ese momento. A la hora de guardar un nuevo componente se debe verificar antes que el componente es el más actualizado y no contiene errores, además de contar con los datos que permitan localizarlo en otro momento. Para esto, cada vez que se cree un nuevo componente y se necesite guardar en el repositorio, debe contener adjunto un documento donde se especifiquen datos como nombre, descripción, uso, lenguaje de programación y ubicación, de forma tal que permita encontrar el componente adecuado de acuerdo a necesidades específicas. (Ver plantilla en Anexo 28).

Para tener mejor organizado el repositorio, en cada proyecto se deben guardar los componentes teniendo en cuenta el flujo de trabajo en que se realizó. Ejemplo, todos los artefactos generados durante el flujo de trabajo de análisis y diseño, se guardarán en una carpeta con el nombre del flujo.

Para evitar perdidas en el repositorio de componentes, solo se permitirá modificar los componentes que se encuentren en el repositorio al Jefe de la entidad correspondiente, al jefe del proyecto y al administrador del repositorio tomando en cuenta el proceso de control de cambio. Se deben establecer mecanismos para gestionar el repositorio de manera eficiente.

2.3.6 Paquete Centro de inteligencia.

Este paquete tiene la responsabilidad de la orientación estratégica de la Factoría, puede ser interna o externa a la factoría. El Centro de inteligencia es el encargado de la gestión del conocimiento y la información, la prospectiva y la vigilancia tecnológica de la factoría.

La Gestión del conocimiento es un recurso imprescindible para las empresas actuales. Constituye la gestión de los activos intangibles que generan valor para la organización. Se relaciona con los procesos de capacitación, estructuración y transmisión del conocimiento. Con la Gestión del

conocimiento se pretende transmitir el conocimiento y experiencia entre todos los miembros de la factoría con el fin de que cada persona tenga los conocimientos necesarios para realizar su trabajo.

La Vigilancia Tecnológica consiste en la captación continua y el análisis sistemático de información relacionada con las tecnologías y sus tendencias. Tiene como objetivo proporcionarles a todos los trabajadores de la factoría información útil para la toma de decisiones. Debe ser capaz de orientar sobre la tendencia del mercado respecto a las tecnologías usadas en la factoría y los productos que se crean en la misma. La vigilancia tecnológica permite conocer el presente y orientar el futuro.

“La prospectiva (del inglés "prospect", significa esperanza) es una disciplina que estudia el futuro desde un punto social, científico y tecnológico con la intención de comprenderlo y de poder influir en él. La Organización para la Cooperación y el Desarrollo Económico (OCDE) define la prospectiva como el conjunto de tentativas sistemáticas para observar a largo plazo el futuro de la ciencia, la tecnología, la economía y la sociedad con el propósito de identificar las tecnologías emergentes que probablemente produzcan los mayores beneficios económicos y sociales. La prospectiva es una disciplina y un conjunto de metodologías orientadas a la previsión del futuro. Básicamente se trata de imaginar escenarios futuros posibles, denominados futuribles, y en ocasiones de determinar su probabilidad, con el fin último de planificar las acciones necesarias para evitar o acelerar su ocurrencia” [13]. La prospectiva estudia el mercado y orientar a la factoría de software hacia una mejora de sus procesos, con el objetivo de producir mejores productos.

La orientación estrategia puede ser interna o externa a la factoría. Se necesita un gestor para la investigación interna y otro para la externa, los que se encargaría de llevar la orientación estratégica para la toma de decisiones a cada miembro de la factoría.

La gestión del conocimiento debe ser realizada por el gestor de información interna encargado de promover el conocimiento para todos los miembros de la organización y almacenar todos los documentos que puedan ser utilizados con este fin. Es el encargado de capacitar a las personas en el uso de las herramientas y metodologías de trabajo que se usan en la factoría, impartiendo cursos de capacitación. En caso de no tener los conocimientos necesarios, se encarga de encontrar una persona capacitada para impartir los cursos.

La vigilancia tecnológica la realiza el gestor de información externa. Este se encarga de investigar como se encuentra el mundo con respecto a las tecnologías, lenguajes de programación y plataforma de desarrollo con las que trabaja la factoría, con el fin de definir hacia donde se avanza en este sentido. Hace un análisis de las ventajas que ofrecen las nuevas tecnologías usadas en el mundo, valorando si es necesario hacer el cambio de tecnología o no.

Estas personas realizan la prospectiva, ya que tienen en cuenta el futuro desde su punto de vista.

La factoría se encuentra en su fase inicial, por lo que en un primer momento no es necesario implantar el centro de inteligencia. La implantación de la factoría se centrará principalmente en las partes de la estrategia que más se necesitan actualmente en el Polo Gestión de Proyecto. El centro de inteligencia se deja para una fase más avanzada de la factoría, en la que estén incluidos varios proyectos del polo y sea necesario hacer una orientación estratégica.

2.4 Aspectos a tener en cuenta para la implantación de la factoría

Definida la estrategia a seguir para implantar la factoría, es necesario reunirse con todos los miembros de la misma para explicarles en que consiste la factoría y las ventajas que trae consigo este cambio de tecnología. Se gestionan todos los recursos humanos, tecnológicos y financieros que hagan falta para poner en marcha la Factoría. Se crea el equipo de dirección y la organización del proceso de desarrollo.

Un recurso importante para implantar la factoría satisfactoriamente es el interés y el compromiso de cada uno de los miembros del proyecto, principalmente de la dirección del mismo.

Una vez implantada la factoría se deben hacer inspecciones sistemáticas para analizar el funcionamiento de la factoría y el desempeño de cada uno de sus miembros. Se hace un seguimiento estricto de cada paquete definido en la estrategia, así como de cada uno de los procesos que engloba, de esta forma se obtienen los resultados de la implantación de la factoría y se puede mejorar el proceso de implantación para una segunda fase del desarrollo de la factoría de software en el Polo Productivo Gestión de Proyecto.

La implantación de la factoría de software podría fallar si no existe un entendimiento común entre los miembros de la factoría y sus coordinadores, si no existe un compromiso personal de cada persona involucrada en el proceso de implantación con el éxito de la misma, si no se cuenta con los recursos necesarios para la puesta en marcha de la factoría de software en el polo.

2.5 Conclusiones

La estrategia propuesta para la implantación de la factoría en el Polo Gestión de proyecto hace uso de la transferencia tecnológica. Para esto se configuraron las entidades del Modelo de Factoría aplicando Inteligencia en paquetes, los que facilitarían la replicación del mismo. Se hace una descripción de los aspectos más importantes de cada paquete, los que se tendrán en cuenta a la hora de implantar la factoría en un proyecto o en el polo en sentido general.

CAPÍTULO 3: RESULTADOS DE LA IMPLANTACIÓN

3.1 Introducción

Para implantar la factoría de software en el Polo Productivo Gestión de Proyecto, en el capítulo anterior se definió una estrategia. Se tomó el proyecto Informatización del convenio Cuba-Venezuela como proyecto piloto para probar la misma. Dado que la estrategia será probada en un solo proyecto, solo se replicarán los paquetes más importantes contenidos en la misma que mejoren la situación actual del proyecto.

Como primer paso se realizó una reunión para explicar en que consiste la factoría de software, los beneficios de este enfoque y las características del modelo de factoría de software propuesto a ser implantado. Se estructuró el equipo de desarrollo definiendo la estructura organizativa y los roles que desempeñará cada integrante del mismo.

3.2 Proyecto piloto

Para probar la estrategia de implantación se tomó entre los proyectos del Polo Productivo Gestión de Proyecto, el proyecto Informatización del Convenio Cuba – Venezuela.

Este proyecto está a cargo de la creación de una aplicación Web para informatizar la gestión de los proyectos entre Cuba y Venezuela. Surge como una necesidad de la Secretaria Técnica Venezolana (MEMPET) y la Secretaria Técnica Cubana (MINVEC) para gestionar las mixtas que se están haciendo como convenio entre los dos países.

El proyecto cuenta con 26 estudiantes de tercer año. Dispone de 14 estaciones de trabajo y un servidor ubicado en el mismo laboratorio (Laboratorio 12). El proyecto está trabajando en un área con acceso restringido, por lo que las políticas de seguridad se vuelven un poco más flexibles, ya

que no todo el personal tiene acceso a las máquinas y demás recursos puestos al servicio del proyecto.

3.3 Organización del proceso

El proyecto Informatización del Convenio Cuba-Venezuela definió las mismas cuatro entidades que se proponen en la estrategia para organizar el proceso de desarrollo. Se designó para cada entidad un jefe encargado de dirigir todas las operaciones que se realizan en la misma.

Entidad Levantamiento de Requisitos y modelamiento del negocio: Se encarga de la fase inicial del proyecto. En esta fase se realiza el levantamiento de requisitos, para esto se realizaron entrevistas a clientes para determinar lo que se esperaba con la aplicación. Se elaboró el documento levantamiento de requisitos, se identificaron satisfactoriamente todos los actores y casos de usos necesarios para comprender el sistema y el negocio. Se logró, de una manera concisa y clara, recoger toda la información necesaria para la puesta en marcha del proyecto en una primera fase.

En esta etapa se necesitaron cuatro personas, dos analistas de negocio, uno de sistema y un arquitecto. Se crearon los diagramas de casos de uso del negocio y del sistema, el documento visión y demás artefactos propuestos para esta fase en la estrategia de implantación.

Entidad Análisis y diseño: A partir de los diferentes diagramas creados en la entidad anterior, se confeccionaron en esta fase los diagramas de clases de diseño correspondientes. No se tomaron en cuenta los demás artefactos que se proponen para esta fase y que también son necesarios para realizar la implementación.

En esta etapa se asignaron tres personas, el Jefe del equipo de análisis y diseño, y dos arquitectos, el arquitecto principal y el de seguridad. Actualmente solo está trabajando en esta fase el jefe de la entidad, los demás arquitectos se encuentran desempeñando otras tareas en la fase de implementación.

Entidad Implementación: En esta entidad no se creó ningún artefacto propuesto en la estrategia, se pasó directamente de las especificaciones en el diagrama de clases del diseño a programar las diferentes partes del producto. Participaron en esta etapa el jefe de implementación y seis programadores más encargados de generar el código de la aplicación.

Entidad Soporte y prueba: En esta etapa se necesitaron cuatro personas, el jefe de la entidad y tres probadores más, los que se encargaron de realizar las pruebas al producto. No se crearon los artefactos propuestos en la estrategia como el plan de prueba y los casos de prueba. Las pruebas se realizaron tomando en cuenta los requisitos del cliente y chequeando que la aplicación cumpliera todo lo especificado. Esta tarea estuvo a cargo de todos los integrantes de la entidad.

Gestión de configuración: La gestión de la configuración solo se enmarcó en el control de cambio, no se realizaron las demás actividades que también son necesarias para no tener inconsistencias a la hora de construir el producto. Este proceso estuvo a cargo de un administrador, encargado además de mantener actualizado el repositorio de componentes.

Cada integrante del proyecto tiene acceso al repositorio de componentes. Cuando necesita hacer algún cambio a un producto, entra al repositorio y hace una copia en su estación de trabajo, una vez modificado lo actualiza en el repositorio. El control de cambio no se realiza a través de la revisión por parte de un comité de control de cambio.

3.4 Personas

No fue necesario crear el equipo de desarrollo para el proyecto Informatización del convenio Cuba-Venezuela, para realizar el mismo se tomó un grupo de estudiantes que anteriormente estaban desarrollando otro tipo de software.

Estas personas se organizaron siguiendo el organigrama Descentralizado controlado. Se designó un Jefe de proyecto y jefes de equipos de desarrollo encargados de la dirección de los equipos

que trabajarán en las diferentes entidades. La responsabilidad del producto estuvo a cargo del Jefe del proyecto, pero la creación del mismo se dividió entre los diferentes miembros del proyecto desempeñando cada persona involucrada en el proceso un rol determinado (Ver Anexo 29). Actualmente se encuentran trabajando en el proyecto 22 estudiantes, el resto pasaron a cumplir otras tareas en el desarrollo del Proyecto Futuro.

Se hizo una distribución de máquinas asignando entre dos y tres personas por cada una. Actualmente existen 4 computadoras compartidas con estudiantes de 5to año que se encuentran desarrollando su trabajo de tesis. La máquina servidora se dejó libre, solo se utiliza para gestionar el repositorio o en caso se existir mucha carga de trabajo y haga falta su uso. El horario de trabajo se organizó teniendo en cuenta el horario de clases de cada persona. En caso de trabajar en una misma computadora personas del mismo grupo, el horario de trabajo se distribuyó tomando en cuenta el tiempo restante, poniéndose de acuerdo en el tiempo que cada uno estaría trabajando. Los estudiantes de 5to año generalmente trabajan en los horarios de clases de sus compañeros u otro tiempo que no afecte el desarrollo del proyecto.

Dado que la factoría de software es aplicada a un solo proyecto, no se cubren todos los roles de dirección que se propone en la estrategia de implantación. El equipo de dirección estuvo compuesto por el Gestor de la factoría (Líder de proyecto), Jefe del grupo de calidad (encargado de la Entidad de Soporte y pruebas) y un administrador del repositorio de componentes reutilizables (desempeña además el rol de programador en la fase de implementación). No se designó ningún Investigador, cada integrante del equipo de desarrollo realiza la investigación en dependencia de sus necesidades.

El proyecto lo conforman estudiantes de 3er año, que no tienen dominio del trabajo con PSP (Proceso de Software Personal), por lo que fue necesario en un primer momento impartir un curso sobre el uso de los diferentes cuadernos que propone PSP. A pesar de que estos cuadernos son muy importantes para el desempeño de la factoría, los mismos no se tomaron en cuenta, ya que no se contaba con el tiempo suficiente para llevar un control de tal magnitud.

No fue necesario impartir cursos sobre las herramientas y lenguajes de programación, ya que anteriormente se encontraban desarrollando software sobre la plataforma Zope, por lo que no fue necesario capacitarlos en el uso de la misma. Tienen conocimientos sobre los diferentes lenguajes de programación que se proponen para la factoría. Además poseen conocimientos básicos sobre la creación de los diferentes modelos y diagramas que propone RUP, adquiridos por el desempeño en otros proyectos y la asignatura ingeniería de software que se imparte en el tercer año de la carrera en la UCI.

No tienen muchos conocimientos sobre el uso de las herramientas de gestión de proyecto y de gestión de configuración, pero no fue posible capacitarlos en el uso de estas herramientas, ya que el uso de las mismas en el polo se encuentra en su fase inicial y no se cuenta con una persona con los conocimientos suficientes para impartir estos cursos. Las personas encargadas de la planificación y la gestión de configuración, para apropiarse de algunos conocimientos sobre el trabajo con las herramientas utilizaron cursos online que en la mayoría de los casos no cubre todo lo que referente al trabajo con estas herramientas.

El proceso de capacitación en este sentido solo se enfocó al uso de los diferentes cuadernos que propone PSP para la planificación y desempeño personal.

3.5 Bases tecnológicas

Cada persona posee en su estación de trabajo las herramientas necesarias para desempeñar bien su trabajo. Durante el desarrollo del proyecto se usaron algunas de las herramientas propuestas en la estrategia, además de otras como ArchGenXML y el plugin de Python para Eclipse (PyDev).

Para crear los diferentes diagramas se trabajó con ArgoUML, a partir de los mismos y haciendo uso de ArchGenXML se generaron script para Python. Se utilizó Eclipse para generar todo el código de la aplicación. Se utilizaron, además, Photoshop y Gimp para el trabajo con las

imágenes y Dreamweaver y Quanta para el trabajo con la Web. Además de Tortoise y RapidSVN para el trabajo con Subversion.

Para proteger la información se restringió el acceso al laboratorio de trabajo al personal ajeno al mismo. En cada estación solo se permite el acceso a las torres de trabajo a los administradores, permitiendo que la información esté protegida de personas no autorizadas. Además se habilitó el firewall y se instaló el NOD32 Antivirus System para proteger las estaciones de trabajo del acceso no autorizado y de los virus que tanto daño hacen a la información.

3.6 Gestión de proyecto

La gestión de proyecto estuvo a cargo de una planificadora, la que se centró principalmente en planificar cada una las tareas que debía cumplir cada integrante del equipo de desarrollo para la primera fase del proyecto, estimando el tiempo que demora cada una de ellas. Para planificar las actividades se propuso en la estrategia utilizar Trac, pero la misma no se usó debido a que la planificadora no cuenta con los conocimientos necesarios para el uso de esta herramienta. La planificación se realizó de forma manual, enviando por correo a cada persona lo que tenía que hacer y el tiempo que disponía para cumplir cada una de las actividades.

No se hizo una gestión de riesgos y de costo del proyecto. La gestión de costo la realiza la facultad en dependencia de la cantidad de profesores asociados al proyecto y los recursos materiales involucrados en el proceso. Para el desarrollo de este proyecto no fue necesario gestionar el costo de los recursos materiales, ya que estos recursos estaban asignados al grupo de desarrollo desde que se creó el mismo. 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.

La gestión de la calidad estuvo a cargo de un asesor, 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.

El Jefe de Proyecto (Gestor de la factoría en el proyecto) se encargó de planificar y administrar los recursos del proyecto. Además de coordinar las relaciones con los clientes y mantener al equipo de proyecto enfocado en la realización de sus tareas con el fin de obtener el producto en el tiempo establecido cumpliendo con las características que exigía el cliente.

3.7 Repositorio de Componentes

El proyecto cuenta con un repositorio de componentes ubicado en un servidor en el mismo laboratorio donde se trabaja (Laboratorio 12). En este servidor se guardan además las herramientas de trabajo que se utilizan en el proyecto, así como los documentos y demás cosas comunes del proyecto, permitiendo contar con un lugar común al cual dirigirse en caso de existir la necesidad de alguna herramienta, documento o componente reutilizable.

Dado que la factoría se encuentra en una fase inicial y el proyecto no cuenta con muchos componentes en el repositorio, se usó el método hojear para organizar los componentes en el repositorio. El uso de este método por el momento resuelve el problema, en una fase más avanzada de la factoría se debe hacer un estudio para determinar que otro método es adecuado según las circunstancias.

Los componentes fueron guardados con un nombre descriptivo que permite encontrarlo según las necesidades. Estos generalmente son productos (equivalente a clases en otros lenguajes) creados durante la fase de implementación.

Mantener el repositorio en forma estuvo a cargo de un administrador, el cual usó Subversion para gestionar todos los componentes creados por el proyecto y que necesitaban estar guardados en el repositorio.

Existe además un repositorio de componentes central ubicado en la dirección de producción de la facultad, en el cual se tiene una copia de las versiones de productos hechas por cada proyecto de la facultad, permitiendo tener la información guardada en varios lugares para evitar perdidas.

3.8 Conclusiones

A pesar de que en un primer momento todo se organizó para trabajar como una factoría de software, esto no se cumplió. Se hizo una división de roles y responsabilidades y se tomaron en cuenta varios de los aspectos propuestos en la estrategia, pero no se logró organizar el proceso de desarrollo de la manera propuesta ya que no se contaba con el tiempo necesario para el desarrollo del producto y los requisitos cambiaban constantemente. El proceso de desarrollo estuvo centrado principalmente en la generación del código, no se crearon en cada entidad los artefactos propuestos en la estrategia.

No basta con establecer una buena estrategia para implantar una Factoría de software y contar el compromiso de cada persona involucrada en el proceso, se necesita además, hacer una estimación de los recursos necesarios para la puesta en marcha de la misma y el del tiempo que demora realizar cada proyecto en la factoría de software para evitar inconsistencia a la hora de construir un producto.

CONCLUSIONES

Las factorías de software ayudan a las empresas a obtener productos de calidad a través de la organización de todos los factores implicados en la producción. Los modelos de factoría se basan en las características principales que debe tener toda factoría de software. La estrategia que se propone en este trabajo se basa en adaptar el Modelo de Factoría de Software aplicando Inteligencia a las características del Polo Productivo Gestión de Proyecto.

Para realizar este trabajo se recolectó la información necesaria para definir la estrategia de implantación del Modelo de Factoría. A continuación se mencionan las etapas por la que transitó la investigación.

Primeramente se hizo un estudio de los diferentes conceptos relacionados con las factorías de software y los objetivos que cumplen las mismas. A esto siguió un análisis de los modelos que varias empresas en el mundo han usado para llevar a cabo el enfoque de factoría, de los mismos fue escogido uno para ser implantado en el Polo Gestión de Proyecto. Además se hizo un análisis de diferentes conceptos que serán usados por la factoría de software.

En el capítulo 2 se propone una estrategia para implantar el Modelo de Factoría aplicando Inteligencia en el Polo Productivo Gestión de Proyecto. Como parte de la misma se configuraron los paquetes a ser replicados, explicando las partes que componen cada uno de ellos.

En el capítulo 3 se presentan los resultados obtenidos con la implantación de la estrategia en un proyecto piloto. El mismo sirvió de base para implantar los diferentes paquetes que se proponían ser replicados en la estrategia. Esta práctica resultó positiva a pesar de no cumplir con todo lo que se esperaba al implantar un modelo de factoría de software en el Polo Productivo Gestión de Proyecto.

Por todo lo anteriormente dicho se considera cumplido el objetivo planteado en esta investigación.

RECOMENDACIONES

Los objetivos de este trabajo no abarcan todos los temas relacionados con el enfoque de factoría de software.

Por lo que se propone:

- Continuar profundizando en el estudio de las factorías de software.
- Comprobar la validez de la estrategia propuesta a través de la implantación de la misma en los demás proyectos que se realizan en el Polo Productivo Gestión de Proyecto.
- Utilizar como metodología de desarrollo una metodología Ágil como XP para una fase más avanzada de la factoría, ya que el desarrollo de aplicaciones Web por lo general es un proceso bastante rápido.
- Adaptar la estrategia a los demás polos de la Facultad 3.

Por los aportes que trae el enfoque de factoría en el ciclo de desarrollo de un producto software se propone hacer uso de modelos de factoría para organizar la producción de software en cada Facultad.

REFERENCIA BIBLIOGRÁFICA

1. Rios La Hoz, Y. and M. Marante Valdivia, *Modelo Funcional de la Factoría de Software de la UCI para la linea Carrefour*. 2005, Universidad de las Ciencias Informáticas: Cuidad de la Habana.

2. Wikipedia. *Factoria*. [cited 19 febrero 2007]; Available from: http://es.wikipedia.org/wiki/FactorÃa.

3. Balderas Padrón, A. and A. Díaz Olavarrieta. *Fábrica de software. Un modelo de negocio certificable basado en Estructura y Capacidades*. [cited 2 Abril 2007]; Available from: http://www.certum.com/Publicaciones/FabSoft.pdf.

4. Fabri, J.A., et al., *Techniques for the Development of a Software Factory: Case CEPEIN-FEMA*.

5. Trujillo Casañola, Y. *Evaluación teórica de la adopción del enfoque de Factorías de Software en la Universidad de Ciencias Informáticas.* 2007 [cited 7 marzo 2007]; Available from: http://www.intempres.pco.cu/Intempres2006/Intempres2006/Evaluacion%20de%20trabajos/Yaim%ED%20Trujillo%20Casa%F1ola%20P.pdf.

6. Peralta, M.L. *ASISTENTE PARA LA EVALUACIÓN DE CMMI-SW*. [cited 26 Febrero 2007]; Available from: http://www.itba.edu.ar/capis/webcapis/proyectodetesisdemagister/peralta-anteproyecto.pdf.

7. ISO, *ISO 9001:2000 Sistemas de gestión de la calidad. Requisitos*. 2000.

8. Humphrey, W.S., *Introducción al Proceso de Software Personal.* Addison Wesley ed. 2001, Madrid.

9. Jacobson, I., G. Booch, and J. Rumbaugh, *El Proceso Unificado de desarrollo de Software*. Vol. I. 2004, La Habana: Editorial Félix Varela.

10. Canós, J.H., P. Letelier, and M.C. Penadés. *Metodologías Ágiles en el Desarrollo de Software*. [cited 8 Marzo 2007]; Available from: http://www.willydev.net/descargas/prev/TodoAgil.Pdf.

11. Vallejo, N. and J.D. Orozco E. *ANÁLISIS RUP – CMMI*. 2005 [cited 3 Marzo 2007].

12. Pressman, R.S., *Ingeniería de Software. Un enfoque práctico*. 2002, La Habana: Editorial Felix Varela.

13. Wikipedia. *Prospectiva*. [cited 23 abril 2007]; Available from: http://es.wikipedia.org/wiki/Prospectiva.

BIBLIOGRAFÍA

1. Vallejo, N. and J.D. Orozco E. *ANÁLISIS RUP – CMMI*. 2005 [cited 3 Marzo 2007].

2. Leyva Almira, H.G., *Aplicación Web para Gestionar Antecedentes Penales*. 2005, INSTITUTO SUPERIOR POLITÉCNICO JOSÉ ANTONIO ECHEVERRÍA: Cuidad de la Habana.

3. Rojas Rivero, D.J. *Aproximación a la Industria del Software en el Estado Lara*. 2003 [cited 14 Marzo 2007]; Available from: http://bibcyt.ucla.edu.ve/edocs_bciucla/rojas//software.pdf.

4. Peralta, M.L. *ASISTENTE PARA LA EVALUACIÓN DE CMMI-SW*. [cited 26 Febrero 2007]; Available from: http://www.itba.edu.ar/capis/webcapis/proyectodetesisdemagister/peralta-anteproyecto.pdf.

5. Jacobson, I., G. Booch, and J. Rumbaugh, *El Proceso Unificado de desarrollo de Software*. Vol. I. 2004, La Habana: Editorial Félix Varela.

6. Cordova Besoain, G.J. *Estudio y comparación de las metodologías: ISMS-CMMI*. [cited 13 marzo 2007]; Available from: http://weblogs.udp.cl/gcordova/archivos/(1234)Estudio_y_comparacin_de_las_metodologas_ISMS-CMMI.doc.

7. Trujillo Casañola, Y. *Evaluación teórica de la adopción del enfoque de Factorías de Software en la Universidad de Ciencias Informáticas.* 2007 [cited 7 marzo 2007]; Available from: http://www.intempres.pco.cu/Intempres2006/Intempres2006/Evaluacion%20de%20trabajos/Yaim%ED%20Trujillo%20Casa%F1ola%20P.pdf.

8. Wikipedia. *Factoria*. [cited 19 febrero 2007]; Available from: http://es.wikipedia.org/wiki/FactorÃa.

9. Balderas Padrón, A. and A. Díaz Olavarrieta. *Fábrica de software. Un modelo de negocio certificable basado en Estructura y Capacidades*. [cited 2 Abril 2007]; Available from: http://www.certum.com/Publicaciones/FabSoft.pdf.

10. Castor, E.d.M. *Fábrica de Software: Passado, Presente e Futuro*. [cited 24 Abril 2007]; Available from: http://www.unibratec.com.br/revistacientifica/diretorio/edicao1/artigo_fabricasw_tecnologus_final.pdf.

11. Pressman, R.S., *Ingeniería de Software. Un enfoque práctico*. 2002, La Habana: Editorial Felix Varela.

12. Aguilar Sierra, A. *Introducción a la Programación Extrema*. 2002 [cited 8 marzo 2007]; Available from: http://www.willydev.net/descargas/Articulos/General/IntroXP.PDF.

13. Humphrey, W.S., *Introducción al Proceso de Software Personal.* Addison Wesley ed. 2001, Madrid.

14. ISO, *ISO 9001:2000 Sistemas de gestión de la calidad. Requisitos*. 2000.

15. Muñoz, J. and V. Pelechano. *MDA vs Factorías de Software*. [cited 18 Marzo 2007]; Available from: http://oomethod.dsic.upv.es/anonimo/..%5Cfiles%5CInConferenceArticle%5C01-Munoz.pdf.

16. Febles Estrada, A. and I. Pérez Estévez, *Medir el proceso de control de configuración, ¿una utopía para la Industria Nacional de Software?*, Instituto Superior Politécnico "José Antonio Echeverría": La Habana.

17. Canós, J.H., P. Letelier, and M.C. Penadés. *Metodologías Ágiles en el Desarrollo de Software*. [cited 8 Marzo 2007]; Available from: http://www.willydev.net/descargas/prev/TodoAgil.Pdf.

18. Rios La Hoz, Y. and M. Marante Valdivia, *Modelo Funcional de la Factoría de Software de la UCI para la linea Carrefour*. 2005, Universidad de las Ciencias Informáticas: Cuidad de la Habana.

19. Caceres, P. and E. Marcos. *Procesos Àgiles para el Desarrollo de aplicaciones Web*. [cited 8 Marzo 2007]; Available from: http://www.dlsi.ua.es/webe01/articulos/s112.pdf.

20. Trujillo Casañola, Y. *Propuesta de modelo de producción de software para la universidad de las ciencias informáticas.* [cited 14 marzo 2007]; Available from: http://www.informaticahabana.com/evento_virtual/?q=node/160&ev=III%20Taller%20Internacional%20de%20Calidad%20en%20las%20TICs.

21. Wikipedia. *Prospectiva*. [cited 23 abril 2007]; Available from: http://es.wikipedia.org/wiki/Prospectiva.

22. Palacio, J. *Sinopsis de los modelos SW-CMM y CMMI*. 2006 [cited 8 marzo 2007]; Available from: http://www.qualitatis.org/files/1/sinopsis_cmm.pdf.

23. Fabri, J.A., et al., *Techniques for the Development of a Software Factory: Case CEPEIN-FEMA*.

24. Basili et. al., 1992]. Basili, V. R.; Caldiera, G.; Cantone, G.; *A Reference Archiecture for the Component Factory*. ACM Transaction on Software Engineering and Methodology. Vol 1. nº 1. pp 53-80. January 1992.

25. [Cantone, 1992]. Cantone, G. *Software Factory: Modeling the Improvement*. 'Competitive Performance Through Advanced Technology'. Third International Conference on (Conf. Publ. No. 359). Pages: 124 – 129. 27-29 Jul 1992.

26. [Cusumano, 1989]. Cusumano, Michael A. *Software Factory: A Historical Interpretation*. IEEE Software, (Vol. 6, No. 2). pp. 23-30. March/April 1989.

27. [Fernstrom et. al., 1992]. FERNSTROM, C.; NARFELT, K. H; OHLSSON, L. *Software Factory Principles, Architecture and Experiments*. IEEE Software. (Vol. 9, No. 2) pp. 36-44. March/April 1992.

28. [Fernández y Teixeira, 2004]. Fernandes, Aguinaldo Aragon; Teixeira, Descartes de Souza. *Fábrica de Software:Implementação e Gestão de Operações*. Editora Atlas, 2004.

29. [Paulk93a], M.C. Paulk, B. Curtis, M.B. Chrissis, and C.V. Weber, Capability Maturity Model for Software, Version 1.1, Software Engineering Institute, CMU/SEI-93-TR-24, February 1993.

30. Fernström, C. *The Eureka Software Factory: Concepts and Accomplishment*. Proceedings Third European Software Engineering Conference. Berlin, 1991.

31. Li, C.; Li, H.; Li, M. *A Software Factory Model Based on ISO 9000 e CMM for Chinese Small Organization*. Second Asia-Pacific Conference on Quality Software (APAQS'01).Hong Kong. December, 2001.

32. VERRAL , M. S. *Software Bus. Architectures for Distributed Development Support Environments*. IEEE Colloquium on. Pages:4/1 - 4/3.

GLOSARIO DE TÈRMINOS

Actor: Alguien o algo, fuera del sistema o negocio que interactúa con el sistema o negocio.

Artefacto: Una parte de la información que (1) es producida, modificada, o usada por un proceso, (2) define un área de responsabilidad, y (3) está sujeta al control de versión. Un artefacto puede ser un modelo, un elemento del modelo, o un documento. Un documento puede adjuntar otros documentos. Una parte de la información que es usada o producida por un proceso de desarrollo.

CMM: El Modelo de Capacidad y Madurez o CMM (Capability Maturity Model), es un método de definir y gestionar los procesos a realizar por una organización.

CMMI: Modelo de Capacidad y Madurez Integrado (Capability Maturity Model Integrated), es un modelo que mide la madurez de las organizaciones.

Cocomo: Constructive Cost Model, es un modelo que permite realizar estimaciones y planificaciones de proyectos de sistemas informáticos.

HTML: “Hyper Text Marckup Language”, lenguaje de marcas, con que se programan las páginas Web. Brinda facilidades para mostrar imágenes, textos hipervínculos, tablas, etc., y es interpretado por los navegadores Web.

IDE: Entorno integrado de desarrollo: un entorno desde el que se pueden editar programas, compilarlos y depurarlos (Integrated Development Environment).

Intranet: Es una adaptación de las mismas tecnologías que existen en Internet, para que sean utilizadas dentro de la red interna de una empresa u organización de forma tal que sus miembros puedan intercambiar información de todo tipo, utilizando el Web como interfaz común.

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

Glosario de Términos

Plugin: Es un programa que interactúa con otro programa para aportarle una función o utilidad específica, este programa adicional es ejecutado por la aplicación principal. Se usan para añadir nuevas funcionalidades sin afectar las ya existentes.

Plataforma: Base, elemento de apoyo.

Proceso: Secuencia de actividades invocadas para producir un producto de software.

Requisito o Requerimiento: Característica o propiedad que se desea para el sistema.

Rol: Papel, cometido o función que tiene, desempeña o interpreta un actor.

UML: "Unified Modeling Language" Lenguaje gráfico que brinda un vocabulario y reglas para especificar, construir, visualizar y documentar los artefactos de un sistema utilizando el enfoque orientado a objetos.

XML: del inglés (eXtensible Markup Language- Lenguaje de Marcado Ampliable o Extensible).