CONSTRUCCIÓN DE UNA HERRAMIENTA SOFTWARE PARA SOPORTAR UN PROCESO DISTRIBUIDO DE DESARROLLO UTILIZADO POR UNA COMUNIDAD (I+D)

Trabajo de Grado

Sandra Isabel Yanza Hernández

Daniel Fernando Muñoz Muñoz

DIRECTOR: Ms© Ing. Jorge Jair Moreno Chaustre

UNIVERSIDAD DEL CAUCA

Facultad de Ingeniería de Electrónica y Telecomunicaciones

Departamento de Sistemas

Popayán, Noviembre de 2006

TABLA DE CONTENIDO

LISTA DE TABLAS

LISTA DE FIGURAS

CAPITULO 1: INTRODUCCIÓN

En esta sección primero se describe y plantea el problema, el cual está relacionado con la ausencia de un soporte informático, a nivel nacional, que apoye el desarrollo de software bajo un ambiente de dispersión geográfica, para ser utilizado por una comunidad; posteriormente se presenta la justificación del proyecto. Se finaliza con el objetivo general y objetivos específicos que se cumplieron con el desarrollo del proyecto.

1.1 DESCRIPCION DEL PROYECTO

Este trabajo de grado se enmarca desde dos perspectivas afines pero que provienen de necesidades de innovación, desarrollo e investigación originadas en dos grupos de investigación (GTI [1] y SIMON [2]) interesados sobre una misma área de conocimiento.

SIMON ha asumido el liderazgo de un proyecto de desarrollo de un Entorno Software de Modelamiento y Simulación (ESMS), el cuál agrega a una red de investigadores[1] pertenecientes a varias universidades colombianas[2]. Este proyecto será desarrollado por equipos (I+D) heterogéneos[3] y distribuidos geográficamente. Para apoyar esta misión este trabajo de grado pretende soportar informáticamente un proceso de desarrollo que lidie con los aspectos distribuidos de la comunidad, facilitando el desarrollo de proyectos de software en comunidad.

El grupo GTI de la Universidad del Cauca, está ejecutando el proyecto denominado SIMEP-SW [3], que propende por el mejoramiento de los procesos de desarrollo de software. De esta manera este trabajo de grado pretende constituirse en un aporte significativo al ofrecer estrategias de desarrollo de proyectos de software entre redes de investigadores geográficamente dispersas.

En general este proyecto ofrece una alternativa para el desarrollo de software en comunidad, que resulte en beneficio para equipos geográficamente distantes o comunidades que emprenden un proyecto en común.

[1] En dinámica de sistemas y otras áreas del modelamiento – simulación.
[2] Universidad del Magdalena, Universidad del Cauca, Universidad Nacional y EAFIT.
[3] En Capacidad, paradigma de trabajo, recursos disponibles, experiencia entre otros.

1.1.1 Planteamiento del problema

De acuerdo con Marques (2004), en el mercado dinámico y competitivo del mundo de hoy, las soluciones informáticas se han convertido en un producto de primera necesidad en nuestra sociedad, donde la eficiencia del desarrollo, la alta calidad y el bajo costo son características deseables para el software generado por esta industria. Sin embargo, según Pressman (2001), dicha industria ha encontrado problemas que dificultan la creación de software con las características deseadas, entre los cuáles se encuentran: en primer lugar, el ritmo de la demanda de software supera al ritmo de la oferta, lo que significa que actualmente se crea software de una manera más lenta de lo que el mercado exige; de otra parte, la actual sociedad de la información ha fundamentado todas sus actividades en la confiabilidad de los sistemas informáticos, por lo cual una falla en el desempeño de estos sistemas podría causar sufrimiento humano o grandes perdidas económicas; por otro lado muchos desarrolladores de software no poseen un pensamiento ingenieril y arquitectónico sólido implicando una concepción de arquitecturas software carentes de mantenibilidad, adaptabilidad, escalabilidad, portabilidad y robustez. En consecuencia la industria de software ha buscado mitigar estas preocupaciones mediante el constante repensamiento de procesos de desarrollo, que estén bien definidos y soportados con tecnología de punta, de acuerdo con lo afirmado por Marques (2004). Durante este proceso de repensamiento, actualmente, está emergiendo, una marcada tendencia (a nivel global) relativa al paradigma de desarrollo de software. Esta tendencia, sitúa el desarrollo de software, desde un ambiente centralizado hacia un ambiente geográficamente disperso, como respuesta a los complejos desafíos del mercado, de acuerdo al reporte del Yankee Group [4], “La compañía de producción monolítica con la capacidad de concebir, diseñar, hacer ingeniería, manufacturar y distribuir sus productos, es cosa del pasado, y así mismo aquella que asume la creación, prueba, y distribución de nuevas aplicaciones propietarias desde un solo lugar”. Entre las dificultades que se presentan para el desarrollo de software en ambientes distribuidos se tienen:

- La dispersión geográfica del equipo de desarrollo, implica que las personas involucradas se encuentran en diferentes ciudades a nivel regional, nacional e incluso global; en otras palabras el equipo de desarrollo, trasciende las fronteras geográficas.
- Por otra parte, la dispersión geográfica implica la dispersión temporal, afectando la disponibilidad y sincronización de actividades, reuniones y tareas transversales al equipo de desarrollo.
- Por último, la dispersión geográfica implica, diferencias culturales y de conocimiento, debido a que cada región tiene sus propias costumbres, creencias y patrones de pensamiento que influyen definitivamente en las formas de trabajo (paradigmas) de los individuos y los equipos.

En el marco de la academia-investigación-industria nacional, existen necesidades que convocan a los profesionales del software a asumirlas con diligencia y responsabilidad con el propósito de convertir las crisis en oportunidades. Y es precisamente en este sentido que una necesidad–demanda ha sido identificada relativa al desarrollo de un Entorno Software de Modelamiento y Simulación de Modelos Integrados[4] (ESMS), que compita en pertinencia, funcionalidad, calidad, fiabilidad y coste con otros ESMS propietarios[5] disponibles en el mercado. Esta necesidad ha sido detectada por SIMON [2], quien desde hace tiempo ha impulsado proyectos (I+D) de software. SIMON reconoce que atender en solitario tal necesidad, demandaría la puesta en marcha de un proyecto, cuyo tamaño y esfuerzo implicarían un alto nivel de riesgo. Para SIMON, es evidente que antes que nada, se requiere un re-pensamiento en la forma de desarrollar software, con el objeto de atender satisfactoriamente a la necesidad detectada. SIMON y SIMEP-SW [3] del GTI [1], reconocen en el paradigma de desarrollo distribuido[6] una alternativa interesante para superar el desafío de un proyecto de desarrollo para un ESMS de calidad industrial en el mediano plazo y la propuesta para una comunidad (I+D) distribuida.

SIMON ha asumido el liderazgo del proyecto de desarrollo de un ESMS de modelos integrados, el cuál agrega a una red de investigadores[7] pertenecientes a varias universidades colombianas[8]. Este proyecto será desarrollado por equipos (I+D) heterogéneos[9] y distribuidos geográficamente.

Además, SIMON requiere apoyo al momento de concebir una arquitectura base para el ESMS, sobre la cuál se conectarán unidades de implementación independientes, con interfaces bien definidas y desarrolladas por equipos geográficamente distantes. Sin embargo, este grupo aún no cuenta con un proceso (bien definido y soportado informáticamente) que apoye el desarrollo distribuido de software, puesto que la experiencia que cada universidad tiene se relaciona con desarrollo de software de manera centralizada.

En consecuencia, si se empieza a trabajar sin mitigar las causas de los problemas arriba descritos, el proyecto de desarrollo, sufrirá entre otros los siguientes síntomas: pérdida de control sobre la sincronización de las tareas y personas, fallas en la comunicación, falencias en la gestión, seguimiento y control del proceso.

[4] Los cuáles representarían realidades complejas, utilizando D.S., Fuzzy, Redes Neuronales, Agentes, Análisis Estadístico, Investigación Operacional y Representación Espacio Estado, entre otros.
[5] Por Ejemplo: MatLab.
[6] En el desarrollo distribuido, los equipos trabajan desde varias ubicaciones geográficamente remotas, pero se comportan como un solo gran equipo.
[7] En dinámica de sistemas y otras áreas del modelamiento – simulación.
[8] Universidad del Magdalena, Universidad del Cauca, Universidad Nacional y EAFIT.
[9] En Capacidad, paradigma de trabajo, recursos disponibles, experiencia entre otros.

1.1.2 Justificación

Con el fin de afrontar la nueva tendencia relativa al desarrollo de software, el paradigma de desarrollo distribuido, este trabajo de grado pretende constituirse en un aporte significativo para ofrecer alternativas de desarrollo que fomenten la cooperación, el tele-trabajo y la comunicación entre redes de desarrolladores geográficamente dispersas, para facilitar el desarrollo de proyectos de software.

Como se ha mencionado en la sección anterior, este trabajo de grado proviene de necesidades de dos grupos de investigación (GTI [1] y SIMON [2]) interesados sobre una misma área de conocimiento.

En consecuencia y con el propósito de apoyar el compromiso de SIMON al mismo tiempo de constituirse en un logro en la presentación de resultados del proyecto SIMEP-SW [3] del grupo GTI, este trabajo de grado pretende:

- Para SIMON: poner a disposición una plataforma software, cuyos servicios de valor agregado, apoyen, las actividades relacionadas con la gestión, comunicación y seguimiento de un proyecto de software, que favorezcan las condiciones mínimas para apoyar a una comunidad de investigadores (I+D) geográficamente dispersa en varias universidades del país, en el contexto de la dinámica de sistemas y pensamiento sistémico, cuyo interés consiste en el desarrollo conjunto de ESMS de modelos integrados.
- Para SIMEP-SW de GTI: aportar significativamente a los resultados del proyecto SIMEP-SW presentando un avance hacia el futuro en la forma como las organizaciones de desarrollo de software en Colombia, enfrentarán el desarrollo distribuido basado en comunidades virtuales de (I+D).

En síntesis, este trabajo de grado de ingeniería propende por ofrecer las condiciones para gestionar la comunicación, la coordinación y el control efectivos para el desarrollo de un proyecto de software que se ejecute en condiciones distribuidas, tomando como referencia algunos requisitos y prácticas de un proceso distribuido de desarrollo [7] y el estudio de algunas herramientas de trabajo colaborativo para el desarrollo de proyectos.

Finalmente, la pregunta fundamental de este trabajo consiste en:

¿Cómo apoyar el desarrollo de software en un ambiente geográficamente disperso?

1.1.3 Objetivos

Este trabajo de grado ofrece una alternativa para el desarrollo distribuido de software, a través de la ejecución de los siguientes objetivos:

Objetivo General

- Desarrollar un entorno software orientado al soporte de un modelo de proceso[10] adecuado para el desarrollo distribuido de software para una comunidad de (I+D) geográficamente dispersa.

Objetivos Específicos

- Apoyar las responsabilidades del proceso, relacionadas con la gestión del ciclo de vida (fases, disciplinas e iteraciones), trabajadores (roles, responsabilidades y conformación de equipos de desarrollo) y documentación (modelos y artefactos), mediante una plataforma software.

- Apoyar las actividades del proceso relacionadas con la comunicación y sociabilidad[11] de los equipos de desarrollo distribuido (programación de reuniones, asignación, seguimiento y entrega de tareas, publicación de calendarios e informes del avance y estado del proyecto entre otros) mediante una plataforma software.

[10] Inspirado en el espíritu de RUP.
[11] Modelo de normas o protocolos de comportamiento en una comunidad.

CAPITULO 2: MARCO TEÓRICO Y ESTADO DEL ARTE

Los conceptos fundamentales que guían el desarrollo de este trabajo de grado son:

- El auge de las comunidades virtuales, fenómeno que se ha extendido de manera considerable a través de la red, posibilitando el contacto entre personas distantes y ofreciendo servicios diversos a todos aquellos que interactúan en un espacio (virtual) determinado, y los elementos fundamentales que caracterizan a una comunidad virtual.
- El trabajo colaborativo como área de investigación interdisciplinaria y gestora de sistemas interactivos para el trabajo en grupo, que tienen como metas principales facilitar la comunicación, promover la colaboración y mejorar la coordinación de tareas, aspectos y desafíos a superar en el desarrollo distribuido de software por una comunidad. Además, el desarrollo de software es una actividad eminentemente colaborativa. De esta manera el modo de colaboración tiene que ser establecido dentro del equipo para asegurar un entendimiento compartido del desarrollo del proceso y que el sistema entregado reúna las necesidades del cliente.
- El Proceso Unificado de Desarrollo (UP) como referencia para concebir una plataforma software de soporte al desarrollo distribuido de software.
- La evolución del pensamiento relacionado con los procesos de desarrollo de software cuando se realizan bajo ambientes distribuidos geográficamente y cómo esta distribución afecta aspectos claves de la interacción humana, tales como: la gestión, la disponibilidad, la coordinación y la comunicación.
- La definición de gestión de proyectos que va ligada con el concepto de administración de actividades que conforman un proyecto y las características de un equipo de trabajo dentro de un proyecto.

En la sección del estado del arte se mencionan investigaciones previas relacionadas con el tema de desarrollo de software en ambientes distribuidos y algunas herramientas de gestión de proyectos estudiadas y analizadas.

2.1 COMUNIDADES VIRTUALES

Desde el nacimiento hasta la muerte, damos forma y somos formados por las comunidades a las que pertenecemos. Para bien o para mal esas comunidades influencian nuestro lenguaje, cómo pasamos el tiempo, lo que consideramos importante, con quienes interactuamos y qué naturaleza tiene esa interacción. Algunas comunidades crean condiciones favorables para interacciones sociales fuertes, mientras que otras son todo lo contrario, algunas son constructivas y algunas no. Desarrollar una comunidad virtual es una actividad compleja, puesto que el concepto de las comunidades virtuales tiene diversos significados para diferentes personas, según lo afirma Preece (2003). De otra parte, la revolución de la información y el Internet han mudado la forma en que las organizaciones e individuos se relacionan. Este fenómeno es causado por aspectos como la globalización, los avances tecnológicos y la agresiva competencia industrial. Un nuevo tipo de organización descentralizada, geográficamente dispersa y con un propósito centrado, consiste en la ***Comunidad Virtual,*** la cuál consiste en *una colección de diversas entidades distribuidas geográfica, funcional o culturalmente, enlazadas por medios electrónicos para la comunicación y que establece relaciones laterales y dinámicas para la coordinación, todo esto para satisfacer un propósito bien definido,* de acuerdo con Desanctis (1998). Según Galbraith (1995), a pesar de la naturaleza difusa de este tipo de organización, una identidad común impera en la mente de todos sus miembros, clientes y participantes, dentro de una compañía sin paredes que se comporta como una red de personas que colaboran entre si, trabajando juntas para la satisfacción del propósito fundamental de la comunidad. Una comunidad virtual está caracterizada por los siguientes elementos fundamentales:

- **Las personas.** Interactúan socialmente para satisfacer sus propias necesidades o las de otros, desempeñando diversos roles en la comunidad.
- **Un Propósito en común.** El interés, la necesidad, el intercambio de información o servicios de valor agregado que dan sentido a la existencia y continuidad de la comunidad.
- **Las Políticas.** Los rituales, protocolos, reglas y leyes que guían la interacción y regulan el comportamiento entre los miembros de la comunidad.
- **Los Sistemas de Computadoras.** Soportan, facilitan y median la interacción social y el sentido de pertenencia a la comunidad, mediante el establecimiento de lazos electrónicos.
- **Limites permeables.** Una entidad puede pertenecer a varias organizaciones, las fronteras físicas no existen.
- **Una estructura reconfigurable**. Se pueden redefinir procesos, roles, asignaciones y carga de trabajo según las necesidades.

En cualquier organización, la comunicación es considerada como factor fundamental de éxito. Sin la comunicación vía electrónica las comunidades virtuales simplemente no podrían ser y menos aún tratar con los inconvenientes causados por la distribución geográfica, temporal y cultural. Este tipo de comunicación, no solo permite a los miembros de una comunidad mantener relaciones virtuales con contactos ya establecidos, sino que provee una plataforma para fomentar nuevas relaciones de este tipo. En otras palabras, el verdadero poder de una comunidad virtual, ocurre cuando, las relaciones entre gente conectada electrónicamente producen cuantitativamente nuevos y diversos tipos de comunicación que permite hacer innovación en productos, servicios y procesos, como lo menciona Ring (1994). Finalmente, la concepción de comunidad virtual como organización se relaciona estrechamente con cuatro conceptos, según Osoy (2005): el groupware, los equipos virtuales, el tele-trabajo y la oficina virtual. Este trabajo de grado se concentrará en las preocupaciones derivadas del funcionamiento eficiente y eficaz del equipo virtual. Los equipos virtuales se pueden comunicar sincrónicamente o asincrónicamente por medio de tecnologías como el e-mail, grupos de discusión, conferencias de audio/video/data, votación electrónica y el trabajo colaborativo. Algunos de los beneficios de los equipos virtuales soportados en herramientas de comunicación electrónicas, es que facilitan compartir información y conocimiento sin el costo que implican los desplazamientos.

2.2 EL TRABAJO COLABORATIVO

De acuerdo con Sosa (2003), el desarrollo de sistemas interactivos tiene como principal objetivo facilitar a los usuarios finales alcanzar sus objetivos y llevar a cabo sus tareas a través del sistema en forma eficaz, efectiva y eficiente. En particular, los sistemas interactivos para el trabajo en grupo, tienen como metas principales facilitar la comunicación, promover la colaboración, mejorar la coordinación de tareas y permitir el seguimiento del proceso de construcción del trabajo común. Por tanto el desarrollo de este tipo de sistemas, exige no solo la identificación de aspectos que definan la naturaleza del problema a resolver, sino también abordar aspectos relacionados al trabajo grupal, tales como conciencia de grupo (group awareness), protocolos sociales, comportamiento y dinámica propia del trabajo grupal. Las nuevas tecnologías de la información y de la comunicación introducen diferentes formas de abordar el trabajo grupal mejorando la comunicación y colaboración entre las personas para conseguir metas comunes. Los sistemas que soportan el trabajo grupal en entornos de trabajo compartidos se desarrollan bajo el paradigma denominado *trabajo colaborativo soportado por computador* (en ingles *Computer Support Cooperative Work* o CSCW). La tecnología que soporta el CSCW se denomina *Groupware*. La aplicación de las tecnologías de la información y la comunicación (TIC) a la colaboración es un resultado natural del aprovechamiento de las posibilidades que estas tecnologías han llegado a ofrecer. Sin

embargo, esta aplicación de las TIC constituye un avance trascendental, porque supone la incorporación de un potente soporte técnico para una actividad esencial de la condición humana que siempre ha estado ligada al progreso de la especie.

2.2.1 Definición de CSCW

Valdez (2004) realiza una recopilación de definiciones de trabajo colaborativo como se presenta a continuación.

El concepto de "Trabajo Colaborativo Soportado por Computador" (*Computer Support Cooperative Work* o CSCW*)*, según Grudin (1991), tiene su origen en el año 1984, año en el cual, en respuesta a una iniciativa de la Corporación de Equipos Digitales y el Instituto Tecnológico de Massachussets, se reunió a un grupo de veinte personas, entre los que se incluían desarrolladores e investigadores de distintas áreas para explorar el rol de la tecnología en los entornos de trabajo. En esa reunión se acuñó el término de "Trabajo Cooperativo Asistido por Computadora" para describir dicho rol.

Según Greif (1998), CSCW ha surgido como un campo de la investigación enfocado sobre el rol del computador en el trabajo en grupo. Las preguntas que se hacen relacionan todos los aspectos del cómo los grupos grandes y pequeños pueden colaborar usando la tecnología del computador: ¿Cómo las personas deben planificar el trabajo en conjunto para aprovecharse de este poderoso medio? ¿Qué tipo de software debe desarrollarse? ¿Cómo el trabajo en grupo se definirá para extraer el potencial de las personas y tecnología?. Las respuestas se encontrarán en la investigación de disciplinas que incluyen a la informática, la inteligencia artificial, la psicología, la sociología, la teoría de la organización y la antropología. Por esta razón, se define CSCW como área de investigación interdisciplinaria.

De forma muy similar, Ellis y Gibbs (1989), definen CSCW como un campo nuevo y multidisciplinario, que utiliza la experiencia y colaboración de muchos especialistas, incluidos profesionales de la computación y de las ciencias sociales. CSCW observa cómo trabajan los grupos y cómo la tecnología puede ayudarlos a realizar mejor su trabajo.

De forma contraria, Greenberg (1991), se enfoca más directamente a las tecnologías o software usados en CSCW y comenta que puede verse como una disciplina científica emergente que guía el diseño y desarrollo de groupware.

Según Bannon (1993), CSCW puede ser concebida como un esfuerzo para entender la naturaleza y las características del trabajo cooperativo con el objetivo de diseñar tecnologías adecuadas basadas en el computador.

Beacker y Posner (1993), definen CSCW como la actividad coordinada asistida por computador (tal como la comunicación y la resolución de problemas) llevada a cabo por un grupo de individuos.

Según la teoría mas reciente, el trabajo colaborativo persigue el desarrollo de conocimiento compartido, la aceleración de los flujos de información, y la coordinación de los flujos de recursos para producir economías de costos y tiempos (2000).

2.2.2 Áreas de Investigación en CSCW

En la tarea de apoyar la interacción de los grupos, las áreas de investigación más importantes en CSCW son: la comunicación, la colaboración, y la coordinación, según lo confirma Ellis (1991).

- **La comunicación**. De acuerdo con Sosa (2003), la comunicación es una actividad humana que permite el intercambio de información entre personas. Se intenta que la comunicación sea eficaz, es decir, que quien envía y quien recibe la información perciban el mismo concepto, y eficiente en cuanto al consumo mínimo de recursos. En el proceso de comunicación se identifican distintos elementos: participantes, información que se transmite y medio. De igual manera, en un sistema informático es posible reconocer a cada elemento: la información contenida en documentos, los artefactos y protocolos de interacción que posibilitan el intercambio, y los distintos modos y tipos de comunicación, como por ejemplo comunicación cara a cara, síncrona, etc.
- **La Colaboración.** Exige a las personas, además de comunicarse, un grado mayor de participación para alcanzar un determinado fin. Dicho de otro modo, colaboración implica la participación “intencionada” y coordinada de los miembros de un grupo. En el área de la colaboración, de acuerdo con Antillanca (2004), el problema más importante es dar facilidades a los integrantes de un grupo para que compartan información. Se necesitan ambientes compartidos que ofrezcan, sin obstruir, contexto actualizado de grupo y, cuando sea apropiado, notificación explícita a cada usuario de las acciones de los demás o de los cambios ocurridos en la información compartida.
- **La coordinación.** Es la actividad orientada a gestionar las dependencias entre actividades realizadas en grupo para alcanzar un objetivo, como lo confirma Sosa (2003). Para modelar la coordinación se deben identificar principalmente las leyes y normas que rigen el funcionamiento de la organización, y las herramientas tecnológicas que soportan el trabajo distribuido.

2.2.3 Definición de Groupware

El término groupware es una contracción de las palabras group (grupo) y software (programas). De acuerdo con Antillanca (2004), esta tecnología también incluye el hardware, el que en algunos casos no solo es equipamiento computacional sino también muebles y espacios arquitectónicos especialmente diseñados y considerados fundamentales para la correcta utilización de una aplicación de software dada. Por lo tanto, el groupware es el software y hardware que soporta y ayuda al trabajo en grupo.

Goldring (1994), define groupware como “software que ayuda a los grupos de personas a comunicarse electrónicamente”. Otros conceptos se orientan a definirlo como un “proceso de trabajo en grupo que tiende a un objetivo preciso y aplicaciones concebidas para facilitar este trabajo en grupo” (1978).

Teniendo en cuenta los anteriores conceptos, el groupware es la tecnología que soporta el CSCW o trabajo colaborativo. Términos alternativos usados para designar groupware son: sistemas colaborativos y sistemas cscw.

Según Ellis (1991), los sistemas colaborativos son sistemas basados en computador que ayudan a un grupo de gente empeñadas en una tarea u objetivo común, otorgándoles una interfaz a un ambiente compartido. Esta definición excluye a cualquier sistema en que los usuarios no comparten una tarea en común, por ejemplo un sistema de tiempo compartido convencional.

Uno de los beneficios, identificado por Sosa (2003), aportado por el groupware es ayudar a disminuir o eliminar la burocracia y la jerarquía vertical en la empresa, es decir, “aplana” la estructura jerárquica de la organización en términos de colaboración, comunicación, espíritu de equipo y refuerza las interacciones humanas.

De acuerdo con Sosa (2003), un groupware sirve para aumentar la eficacia del trabajo en tres niveles claves que dan soporte a la interacción grupal: comunicación, colaboración y coordinación. Sin estos niveles cualquier grupo de personas no puede prosperar en su trabajo. De allí, otro concepto que define el groupware como un conjunto de métodos, medios y herramientas que permiten a un grupo de personas mejorar en los tres aspectos mencionados anteriormente.

De igual manera, se definen las tecnologías de trabajo colaborativo como aquellas herramientas que permiten el trabajo en grupo, mediante las cuales distintas personas, situadas en diferentes localizaciones geográficas, pueden trabajar en un mismo proyecto, comunicarse, cooperar, plantear y resolver problemas, coordinarse para la concepción de una tarea común y compartir información y

documentos de trabajo. Normalmente estas herramientas se utilizan por medio de computadores vía Internet e Intranet.

2.2.4 Taxonomía de las Aplicaciones Groupware

De acuerdo con Ortega (2001), las tecnologías de groupware se pueden clasificar de forma general según dos dimensiones, retomando lo expresado por Johansen (1988):

1. Si los usuarios de los grupos están trabajando juntos al mismo tiempo, modo "síncrono", o lo hacen sin coincidir en el tiempo, modo "asíncrono".
2. Si los usuarios están trabajando en grupo en el mismo lugar, "cara a cara", o en diferentes lugares, "distribuidos".

Taxonomía Espacio-Temporal	Mismo Tiempo	Diferente Tiempo
Mismo Lugar	Interacción cara a cara	Interacción asíncrona
Diferente Lugar	Interacción distribuida sincrónica	Interacción distribuida asincrónica

Tabla 1: Taxonomía Espacio-Temporal de las aplicaciones groupware

Algunas de las aplicaciones y servicios de groupware más importantes se resumen en la siguiente tabla:

Taxonomía Espacio-Temporal	Mismo Tiempo	Diferente Tiempo
Mismo Lugar	• Soporte de presentación: proporcionan el soporte necesario para presentar información a un grupo de usuario. • Soporte de Decisiones: facilitan soporte para facilitar la toma de decisiones a un grupo de usuarios. • Equipos audiovisuales: proyector multimedia, proyector de diapositivas, TV, VHS, pantalla	• Computadores compartidos por diferentes usuarios. • Uso de aplicaciones automatizadas generales (procesadores de texto, hojas electrónicas de cálculo, graficadores, etc.).

	electrónica.	
Diferente Lugar	• Chat: permite la interacción escrita entre un grupo de usuarios. • Video-conferencia: conversación en línea en modo video interactivo a través de Internet utilizando software especial orientado al envío-recepción de voz y video. • Escritura colaborativa: permite la colaboración de los miembros de un grupo para la preparación de un documento, facilitando a los usuarios la anotación y modificación de los mismos. • Pizarra compartida: permite a un grupo de usuarios compartir una superficie de dibujo. • Mensajería instantánea: permite el intercambio inmediato de mensajes entre usuarios.	• Correo electrónico (e-mail): la más común de las aplicaciones de groupware. Esencialmente es una tecnología que permite el envío de mensajes entre usuarios, aunque actualmente permite hacer operaciones más sofisticadas, como reenviar mensajes, adjuntar ficheros, crear grupos de correo, etc. • Listas de correo (mailing list): hace posible enviar mensajes de correo electrónico a grupos de usuarios. • Foros de noticias (newsgroups): permite publicar mensajes de correo electrónico clasificados en temas para que los usuarios autorizados puedan acceder a ellos bajo demanda. • Flujo de trabajo (workflow): la mayor parte de los sistemas groupware presentan la opción de organización de tareas mediante funciones workflow. Esta técnica permite su automatización mediante un conjunto de reglas previamente establecidas que encaminan la información gracias a la fusión de formularios, correo electrónico y bases de datos. • Calendario y agenda: soporte para coordinación de recursos, planificación, gestión de proyectos, etc.

Tabla 2: Aplicaciones y servicios de groupware

2.3 PROCESOS DE DESARROLLO DE SOFTWARE

Este trabajo de grado tomará como referencia al Proceso Unificado de Desarrollo (UP) con el propósito de concebir una plataforma software de soporte al desarrollo distribuido de software. El UP en su versión de Rational Corporation, puede adecuarse a proyectos de alta o baja complejidad (tecnológica y de gestión), grandes o pequeños presupuestos y variedad de organizaciones. En consecuencia, en primer lugar se proporcionará una breve descripción del proceso UP y más adelante se enunciarán los desafíos relacionados con el desarrollo distribuido que debería intentar mitigar la plataforma que se desarrolle en este trabajo de grado.

2.3.1 El Proceso Unificado de Desarrollo

De acuerdo con Jacobson (2000), el proceso unificado de desarrollo (UP) es un modelo de proceso concebido para el desarrollo iterativo e incremental del software. Sus creadores Booch, Jacobson y Rumbaugh han concebido UP para proyectos de desarrollo cuya complejidad técnica y de gestión es significativa. UP es adaptable a cualquier proyecto y organización de desarrollo. El UP propone un marco de trabajo genérico que puede especializarse para diferentes sistemas de software, áreas de aplicación, niveles de aptitud y diferentes tamaños de proyectos. Está basado en componentes, lo cual quiere decir que el sistema software en construcción está formado por componentes software interconectados a través de interfaces bien definidas.

El espíritu fundamental del UP consiste en tres (3) principios:

- **Dirigido por casos de uso.** Los casos de uso representan historias de usuario, su conjunto constituye el modelo de casos de uso, el cual describe la funcionalidad total del sistema.
- **Centrado en la arquitectura.** En síntesis la arquitectura comprende la estructura de cómo los componentes de software (lógicos o físicos) se organizan, interrelacionan y colaboran para satisfacer los requisitos críticos del sistema. La realización de los casos de uso críticos darán forma a la arquitectura.
- **Iterativo e Incremental.** El UP es adaptativo en su naturaleza más íntima, al contrario de otros enfoques (predictivos)[12]. En UP un gran esfuerzo de desarrollo (que puede durar varios meses o

[12] Por ejemplo el Ciclo de vida clásico o Desarrollo en Cascada

años) se divide en partes más pequeñas o mini proyectos. Cada mini proyecto es una iteración que resulta en un incremento de software[13].

El proceso puede ser descrito en dos dimensiones o dos ejes. La primera dimensión representa el tiempo y muestra el aspecto dinámico del proceso. Se expresa en términos de fases e iteraciones. La segunda dimensión o eje, representa el aspecto estático del proceso, es decir las actividades y los artefactos.

2.3.1.1 Estructura del Proceso Unificado

- **Las Fases.** Se entienden como los momentos que vive el software durante su ciclo de vida de desarrollo. El proceso unificado propone cuatro: *Inicio, Elaboración, Construcción y Transición*. Cada una tiene unos hitos que determinan cuando la fase termina, dando paso a la siguiente.

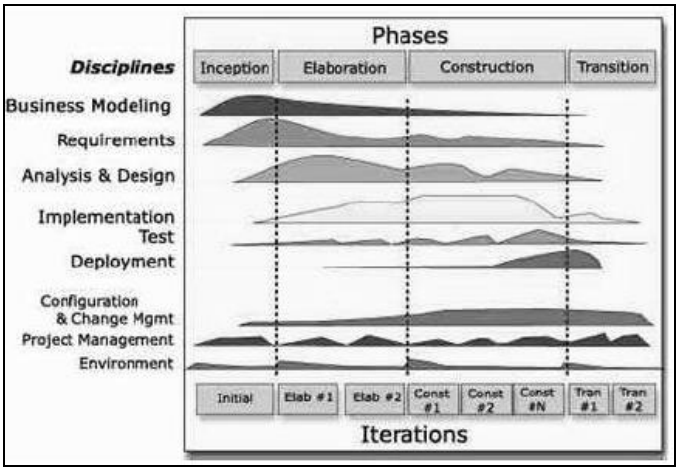


Figura 1: Estructura del Proceso Unificado

- **Las Disciplinas.** Las disciplinas de UP consisten en flujos de trabajo fundamentales que se ejecutan durante todo el ciclo de vida del proceso de desarrollo, pero variando su énfasis dependiendo de la fase en la cuál se encuentre el proceso en un momento determinado. (Ver Figura 1: Estructura del Proceso Unificado).

[13] El cuál resulta ser un sistema de calidad de producción pero cuya funcionalidad es parcial.

- **Las Iteraciones.** El UP incorpora mini-proyectos llamados iteraciones, los cuáles son responsables de implementar una pequeña porción de la funcionalidad del sistema (representada como casos de uso) en una escala de tiempo de semanas a meses. Esta filosofía, permite al equipo de desarrollo, tratar con una complejidad parcial, al mismo tiempo que constantemente se re-orienta el rumbo del proceso con el propósito de verificar que se este desarrollando el sistema correcto. Cada mini-proyecto o iteración es semejante a un proceso en cascada de corta duración. UP contempla las siguientes disciplinas: Requerimientos, Análisis/Diseño, Implementación/ Pruebas, Gestión de Configuración, Gestión del Cambio, Gestión del Proyecto y Entorno.

2.3.2 Desarrollo Distribuido VS. Desarrollo Disperso

Según Ambler (2002), en el desarrollo distribuido, los sub-equipos trabajan desde varias ubicaciones geográficamente remotas, pero se comportan como un solo gran equipo. Este tipo de desarrollo ocurre, por ejemplo, cuando, un equipo externo construye parte del sistema mediante OutSourcing, mientras que otros equipos trabajan juntos dentro de la misma organización. En el caso del desarrollo disperso, ocurre:

- Cada miembro del equipo trabaja en un lugar remoto, por ejemplo, su propia casa.
- Los individuos trabajan en ubicaciones físicas diferentes dentro de las instalaciones de una misma compañía.
- Los individuos trabajan bajo el esquema de código abierto.

En síntesis, los dos enfoques difieren radicalmente, mientras que en el desarrollo distribuido, son los sub-equipos quienes trabajan desde diferentes ubicaciones remotas, en el desarrollo disperso son los individuos quienes trabajan remotamente. El interés de este trabajo de investigación se concentra en el desarrollo distribuido, debido a que es la configuración organizativa que representa fielmente la realidad de distribución de la red de investigadores de I+D en dinámica de sistemas que está involucrada activamente en el proyecto de desarrollo liderado por SIMON [2] y la oportunidad de proponer un logro para el proyecto SIMEP-SW [3] de la Universidad del Cauca.

2.3.3 Evolución de los Procesos Distribuidos de Desarrollo

La evolución del concepto del desarrollo distribuido, inicia con las reflexiones suscitadas por la necesidad de superar con éxito los esfuerzos de desarrollo que no imponen restricciones respecto a la dispersión geográfica de todos los implicados. Estas reflexiones, se agrupan en tres categorías a saber: La gente,

los Procesos y la Tecnología. A este respecto, se presenta a continuación, la revisión de algunos planteamientos y experiencias sobre los procesos de desarrollo distribuidos, basado en el SOFTWARE PRODUCTIVITY CONSORTIUM [5]:

"El desarrollo distribuido establece desafíos significativos relativos a la cultura organizacional y la filosofía de la gestión, las arquitecturas de software, la comunicación, satisfacción de los clientes, los procedimientos de integración y prueba y muchas otras áreas" dice Lockheed Martin Fred Palmer [5], quien se ha desempeñado como moderador del Open Forum Session durante el "The Executive Round Table". "The Executive Round Table", facilita muchas lecciones útiles que ayudan a las personas a implementar exitosamente programas de desarrollo distribuido. Las siguientes reflexiones, sintetizan el trabajo que respecto al desarrollo distribuido ha elaborado el Executive Round Table. A continuación se enuncian tres categorías de reflexiones: Gente, Procesos y Tecnología.

- **La Gente.** La postura más relevante identificada para el desarrollo geográficamente distribuido es la comunicación. Dada la necesidad de crear una visión compartida del producto y proceso software en un ambiente de desarrollo distribuido, el valor de la comunicación cara a cara es vital, particularmente cuando se presenta un diseño complejo que represente una significativa dificultad técnica.

- **El Proceso.** La necesidad de procesos maduros en común a través de los proyectos de desarrollo geográficamente distribuidos es uno de los desafíos identificados más difíciles de superar. Sin embargo, el simple hecho de tener procesos en común no es suficiente, el verdadero reto de la gestión consiste en inculcar y hacer respetar rigurosamente esos procesos comunes entre todos los miembros del equipo y la gestión, y en asegurarse de que las herramientas seleccionadas para el proceso encajen todas en su lugar.

- **La Tecnología.** El groupware y las herramientas de comunicación son requisitos tecnológicos de apoyo a los procesos distribuidos de desarrollo. Así mismo, aplicaciones Intranet, servidores del proyecto, y video conferencia se han consolidado como las tecnologías más útiles en la mayoría de esfuerzos de desarrollo distribuido. Debe tenerse cuidado sin embargo en el uso de herramientas genéricas las cuáles frecuentemente están pobremente documentadas y soportadas y en ocasiones hacen el uso del proceso algo poco amigable. De igual forma, se destaca la importancia de negociar y acordar un conjunto de herramientas comunes de apoyo al proceso siempre que sea posible, aunque en ocasiones es más crítico aún el establecimiento de procesos comunes y arquitecturas abiertas para garantizar el éxito del desarrollo distribuido. Por último, la

necesidad de compartir información privada en un entorno geográficamente distribuido, motiva la seguridad de la red distribuida.

De otra parte, mediante un continuo trabajo de la comunidad del software, las anteriores reflexiones esclarecen nuevos **desafíos** que traen consigo los ambientes distribuidos de desarrollo los cuáles fundamentalmente se concentran en formas alternativas para tratar los aspectos distribuidos de actividades clave tales como: La comunicación, la infraestructura, la coordinación, la disponibilidad y la gestión. El esclarecimiento de estos nuevos desafíos, provoca una reacción, cuyo propósito fundamental consiste en la concepción de soluciones que mitiguen los inconvenientes causados por las soluciones tradicionales al tratar con los aspectos distribuidos de las actividades clave arriba mencionadas. Sin embargo, estas **soluciones** son meras aproximaciones y son constantemente cuestionadas a la luz de su eficacia, en consecuencia, se pueden aprender muchas **lecciones** de su apropiación, uso y desempeño. A este respecto, se presenta la revisión de algunos de los paralelos **Desafíos vs. Soluciones** que implican los esfuerzos de desarrollo distribuido, así como las lecciones aprendidas de la implementación e implantación de las mismas en entornos distribuidos. Estos aportes fundamentalmente pertenecen a Michael Kircher, Prashant Jain, Angelo Corsaro y David Levine (2001), quienes lideran la iniciativa del Distributed Extreme Programming o DXP. De otra parte, se exploran las diferencias entre el desarrollo distribuido y el desarrollo disperso, destacadas por Scott W. Ambler (2002), con el objeto de encuadrar el universo temático objetivo de este trabajo de investigación. La información relativa a los desafíos y soluciones inherentes a los procesos distribuidos de software se encuentra descrita a continuación:

2.3.3.1 Desafíos Encontrados y Soluciones Propuestas

- **La Comunicación.** Un aspecto importante en la comunicación para los humanos, es conocer como reaccionan las personas a las cosas que uno mismo dice. Para juzgar esa reacción, típicamente una persona podría leer el lenguaje corporal o facial y hasta la entonación de la voz. En el desarrollo distribuido las personas generalmente están en ubicaciones geográficamente distantes, ¿como podría alguien conocer las reacciones que otros tienen acerca de sus comentarios?
    - Iniciar la creación del equipo de trabajo en una ubicación física única, con el propósito de permitir a las personas el tiempo suficiente para conocerse y constituir una cultura en común.
    - Luego, es suficiente con una pequeña ventana de video, para percibir, las mutuas reacciones que en la comunicación remota tienen las otras personas acerca de los

comentarios de otros. Al respecto podrían acordarse formas de comunicación remota basadas en video, teléfono, chat, email o teleconferencia.

- Más tarde, deben acordarse reuniones periódicas para incrementar las relaciones interpersonales de todos los miembros del equipo, de tal manera que la cooperación remota se facilite gracias a los lazos de confianza creados.
- Finalmente la capacidad y habilidad para compartir documentos mejora enormemente la comunicación y cooperación remotas.

- **La Coordinación.** Cuando dos o más miembros del equipo que trabajan juntos en un proyecto desde ubicaciones físicas diferentes, la coordinación del trabajo entre ellos se convierte en todo un desafío. Sincronizar la disponibilidad ajustando las diferencias en los tiempos y cronogramas personales, con el propósito de coordinar la distribución, integrar las actividades, y al mismo tiempo compartir documentos y aplicaciones es un desafío a superar.
  - La apropiada coordinación entre los miembros del equipo requiere un mínimo de planeación. Sin embargo hacer un uso extensivo de varias líneas de comunicación puede facilitarlo. Por ejemplo: 2 miembros del proyecto ubicados remotamente podrían intercambiar sus agendas de actividades diarias vía email, en ellas podrían asignar de común acuerdo algunas franjas para dedicarlas al proyecto. Lo anterior les obligaría a tomar en cuenta las diferencias de tiempo disponible existentes para cada uno.

- **La Infraestructura.** Tanto la comunicación como la coordinación, en el desarrollo distribuido dependen de la infraestructura disponible. Esto incluye tanto el hardware como el software, así como el ancho de banda de la red. Una infraestructura pobre puede dificultar seriamente estos aspectos en un proceso distribuido. La disponibilidad de una infraestructura suficiente es vital.
  - Todos los miembros del equipo deben tener el software y hardware adecuado. Se constituyen como criterios importantes de selección del software los siguientes: El software debe ser seleccionado, teniendo en cuenta su: Facilidad de uso, interoperabilidad con otras herramientas y disponibilidad en diferentes plataformas. Igualmente, el hardware debe ser seleccionado cuidadosamente. De hecho cada desarrollador necesitará el poder de una estación de trabajo clásica con características multimedia.

- **La Disponibilidad.** Los miembros de un equipo distribuido de desarrollo pueden estar disponibles pero no necesariamente de forma síncrona. Algunos de ellos incluso podrían estar trabajando en múltiples proyectos y tener su dedicación restringida, mientras que otros simplemente tendrán una disponibilidad limitada debido a asuntos de índole personal. En otras ocasiones podrían presentarse incluso limitantes de disponibilidad relacionadas con diferentes zonas horarias. A continuación, se plantean varias estrategias para superar el componente distribuido de la disponibilidad.
    - No negar ayuda a nadie que la esté solicitando especialmente si es desde una ubicación remota.
    - Publicar diaria o semanalmente las agendas de disponibilidad para cada miembro del equipo.

- **La Gestión.** El gestor del equipo, necesita confiar mucho en sus subordinados y más aún si ellos frecuentemente son remotos. Deben definirse nuevas estrategias alternativas al control directo que realiza el gestor.
    - Los líderes del proyecto necesitan aprender a manejar equipos distribuidos. En particular, debe aprenderse como administrar a los miembros ubicados en locaciones remotas esto podría incluir: Requerir diaria o semanalmente reportes de todos los miembros del equipo ya sean locales o remotos. Proporcionar realimentación frecuente a todos los miembros del equipo para crearles un sentimiento de conexión y pertenencia. Por último, eventos regulares pueden ayudar a construir confianza y motivación entre todos los miembros del equipo.

Finalmente, nuevas **soluciones** llevan a nuevas **lecciones** y a partir de estas, nuevas **reflexiones** son planteadas, no para finalizar el ciclo, sino para iniciar uno nuevo, incrementando el conocimiento y estabilizando el concepto siempre evolutivo del desarrollo distribuido.

2.3.4 Entornos de Soporte a los Procesos de Desarrollo Distribuidos

Según IBM [6], una plataforma que soporte el desarrollo geográficamente distribuido de software, debería satisfacer por lo menos algunas de las siguientes características:

1. **Operar** eficientemente sobre la infraestructura de red disponible (preferiblemente sobre Internet) permitiendo el desarrollo concurrente 24X7.
2. **Garantizar** seguridad y control del acceso autorizado a datos (procesos, proyectos, etc).

3. **Garantizar** la comunicación y colaboración entre el personal geográficamente distante mediante componentes integrados como e-mail, calendarios, agendas, e-conference, e-learning, libretas de contactos, etc.
4. **Garantizar** la coordinación del equipo geográficamente distribuido a pesar de las barreras del lenguaje, cultura y geográficas.
5. **Soportar** la funcionalidad suficiente para permitir la definición de uno o varios modelos de proceso, aplicables como plantillas a los proyectos que se definan.
6. **Garantizar** la administración de los requerimientos y artefactos elaborados para un proyecto cualquiera, incorporando servicios de búsqueda y recuperación mediante las relaciones de trazabilidad entre artefactos.
7. **Capacidad** para valorar mediante métricas, el estado actual del proyecto, la variación en presupuesto y cronograma, así como detección de las áreas más problemáticas.
8. **Capacidad** de almacenaje de modelos de proceso, proyectos, estadísticas de fracasos/éxito y componentes de software en repositorios geográficamente distribuidos como estrategia para el fomento de la reutilización de conocimiento y el intercambio de experiencias.

2.4 ADMINISTRACIÓN DE PROYECTOS

2.4.1 Definición

La definición de administración de proyectos va directamente ligada a la plataforma a desarrollar, porque con esta herramienta se persigue integrar el concepto de administración de actividades de un proceso de ingeniería de software utilizado en un proyecto y la forma de comunicación, colaboración y coordinación entre los integrantes del equipo que las realizan.

Cada proyecto de desarrollo de software está basado en una metodología de desarrollo de software. Las tareas que se pretenden gestionar están ligadas al Proceso Unificado de Desarrollo (UP), relacionadas con la gestión del ciclo de vida del proceso (fases, iteraciones y actividades), trabajadores (asignación de roles, asignación y seguimiento de actividades, conformación de equipos), documentación (asignación y disponibilidad de artefactos) y comunicación entre los integrantes del equipo.

Según la guía PMBOK [8], un proyecto se define como el esfuerzo temporal que se realiza para crear un producto o servicio único. De acuerdo con Valdez (2004), el término temporal de la definición, se refiere a que cada proyecto tiene una fecha de inicio y una fecha de término, y, el concepto único, se refiere a

que el producto o servicio contienen elementos o rasgos que los distinguen de los productos o servicios ya existentes. Un proyecto también se define como un grupo de tareas orientadas a crear un producto o servicio único. Estas tareas son ejecutas por un equipo en un periodo de tiempo definido y organizadas bajo la dirección de un Gerente de proyectos [9].

Los atributos de un proyecto son:

- Propósito único: crea un solo entregable (producto, servicio o resultado).
- Temporal: cada proyecto tiene definido un principio y un fin.
- Elaboración progresiva: incrementos desarrollados en pasos y continuamente.
- Requiere recursos de diferentes áreas.
- Incertidumbre: al ser único introduce el concepto de incertidumbre.

Todo proyecto tiene aspectos implícitos:

- Involucra tareas que son planificadas, ejecutadas y controladas. (Tiempo)
- Requiere gente para realizar esas tareas. (Personas)
- Está sometido a restricciones. (Recursos)

Según Cruz (2004), otra definición indica que “la administración de un proyecto, se define como un sistema de procedimientos, prácticas, tecnologías y conocimientos del tema que permiten la planificación, organización, designación de personal, dirección y control necesarios, para poder manejar con éxito un proyecto de ingeniería de software”.

El resultado final de un proyecto de software es un producto que toma forma durante su desarrollo gracias a la intervención de muchos tipos distintos de personas. Un proceso de desarrollo de software guía los esfuerzos de las personas implicadas en el proyecto, a modo de plantilla que explica los pasos necesarios para terminar el proyecto. Típicamente, el proceso está automatizado por medio de una herramienta o de un conjunto de ellas.

De acuerdo con Jacobson (2000), en el desarrollo de software se distinguen cuatro elementos predominantes, conocidos como las cuatro “P”:

- **Personas.** Los principales autores de un proyecto software son los arquitectos, desarrolladores, ingenieros de prueba, y el personal de gestión que les da soporte, además de los usuarios,

clientes, y otros interesados. Las personas son realmente seres humanos, a diferencia del término abstracto trabajadores.

- **Proyecto.** Elemento organizativo a través del cual se gestiona el desarrollo de software. El resultado de un proyecto es una versión de un producto.
- **Producto.** Artefactos que se crean durante la vida del proyecto, como los modelos, código fuente, ejecutables y documentación.
- **Proceso.** Un proceso de ingeniería de software es una definición del conjunto completo de actividades necesarias para transformar los requisitos de usuario en un producto. Un proceso es una plantilla para crear proyectos.

Además existe otro elemento esencial en el proceso, las herramientas. Las herramientas son software que se utiliza para automatizar las actividades definidas en el proceso. Las herramientas soportan los procesos de desarrollo de software modernos. Hoy, es impensable desarrollar software sin utilizar un proceso soportado por herramientas. Las herramientas son buenas para automatizar procesos repetitivos, mantener las cosas estructuradas, gestionar grandes cantidades de información y para guiarnos a lo largo de un camino de desarrollo concreto.

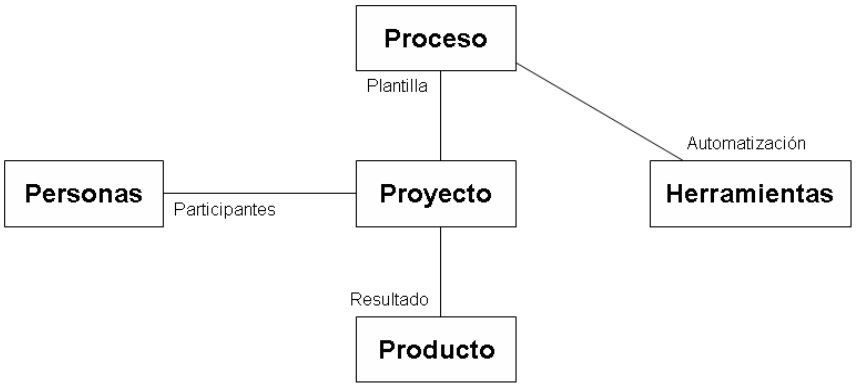


Figura 2: Las cuatro "P" en el desarrollo de software

2.4.2 Equipos de trabajo en el desarrollo de software

Según McConnell (1997), un equipo de trabajo es algo más que un conjunto de personas que desean trabajar juntas para formar un equipo, por lo cual para la definición de equipo se basa en la siguiente "*un equipo es un número de personas pequeño, con habilidades complementarias, que están comprometidas en un propósito, objetivos de rendimiento y con un enfoque comunes, en el que todos*

sean responsables ante todos". Un buen equipo existe siempre que dos cabezas funcionan mejor que cada una por separado. No todos los grupos son equipos. Algunos proyectos se pueden llevar a cabo por grupos de personas que cooperan, pero que no forman un equipo. Algunos grupos no llegan al nivel de compromiso que supone un equipo.

McConnell (1997) explica que un equipo productivo cuenta con las siguientes características:

- *Una alta visión u objetivos compartidos.* Una visión común crea confianza entre los miembros del equipo, porque saben que todos trabajan para llegar al mismo objetivo.
- *Un sentido de identidad del equipo.* En este caso los objetivos comunes se consideran más importantes que los personales y se entiende que se tiene la oportunidad de conseguir con el equipo algo que individualmente no podrían.
- *Una estructura dirigida por resultados.* Dentro del equipo las tareas deben ser claras y todos deben ser responsables de su trabajo en todo momento. El equipo debe tener algún medio de control sobre el rendimiento individual y la realimentación, y las decisiones se deben tomar, si es posible, basándose en hechos más que en opiniones subjetivas.
- *Un compromiso con el equipo.* El compromiso de los miembros se refleja en los sacrificios personales por el bien del equipo. Cuando el equipo se compromete, tiene que haber algo por lo que comprometerse.
- *Confianza mutua.* La confianza consta de cuatro componentes, la honestidad, la franqueza, la firmeza y el respeto. Si se infringen algunos de estos elementos, aunque sólo sea una vez, la confianza se rompe.
- *Comunicación efectiva.* Los miembros de los equipos unidos están al corriente constantemente de lo que les sucede a los demás. Tiene cuidado de comprobar que todos les entiendan cuando hablan, y su comunicación se beneficia del hecho de que muestran una visión común y un sentido de identidad.
- *Un sentido de autonomía.* Los equipos efectivos tienen la sensación de que son libres para poder realizar todo lo que crean necesario, para que el proyecto tenga éxito.

McConnell (1997) retoma un conjunto de directrices básicas para los responsables y miembros del equipo. Si el equipo desea adoptar un conjunto de reglas, las directrices de la Tabla 3, son una buena forma de comenzar.

Responsable del equipo	Miembro del equipo
1. Mostrar el compromiso personal con el objetivo	1. Demostrar una comprensión realista de mi función

del equipo.	y posibilidades.
2. No diluir los esfuerzos del equipo con demasiadas prioridades.	2. Demostrar las opiniones objetivas y basadas en hechos.
3. Ser justo e imparcial con todos los miembros del equipo.	3. Colaborar eficientemente con los demás miembros del equipo.
4. Estar dispuesto a hacer frente y resolver temas asociados con el rendimiento inadecuado por parte de los miembros del equipo.	4. Poner los objetivos del equipo por encima de cualquier objetivo personal.
5. Estar abierto a nuevas ideas e información por parte de los miembros del equipo.	5. Mostrar la voluntad de desarrollar el esfuerzo necesario para conseguir el éxito del equipo.
	6. Estar dispuesto a compartir información, percepción y realimentación apropiadamente.
	7. Demostrar altos niveles de excelencia.
	8. Apoyar y admitir las decisiones del equipo.
	9. Demostrar aspectos de responsabilidad de forma que contribuya al éxito del equipo.

Tabla 3: Directrices básicas para miembros y responsables de equipos

2.5 LA PLATAFORMA .NET

La plataforma de desarrollo seleccionada para el proyecto es la Plataforma .NET. La plataforma .NET en realidad no es algo radicalmente nuevo, es un conjunto de tecnologías dispersas, que en muchos casos ya existían, y Microsoft ha integrado en una plataforma común con el objetivo de facilitar el desarrollo de servicios que se pueden acceder a través de la Web; éstos servicios pueden ser utilizados por personas o por otros sistemas, que a su vez pueden proporcionar servicios mas elaborados.

Microsoft .NET es una plataforma para construir, ejecutar y experimentar la tercera generación de aplicaciones distribuidas, que constan de los siguientes elementos [10]:

- Un modelo de programación basado en XML.
- Un conjunto de servicios Web XML, como Microsoft .NET My Services para facilitar a los desarrolladores integrar estos servicios.
- Un conjunto de servidores que permiten ejecutar estos servicios (como .NET Enterprise Servers).
- Software en el cliente para poder utilizar estos servicios (como Windows XP, agendas electrónicas, etc.)
- Herramientas para el desarrollo como Visual Studio.NET.

2.5.1 Arquitectura .NET

Una definición general de la arquitectura .NET podría ser la siguiente: "Una plataforma independiente del lenguaje para el desarrollo de servicios Web" [10]. La arquitectura .NET (.NET Framework) es el modelo de programación de la plataforma .NET para construir y ejecutar los servicios .NET. El objetivo de esta arquitectura es la de reducir la complejidad en el desarrollo de este tipo de aplicaciones, permitiendo a los desarrolladores centrarse en escribir la lógica específica del servicio a desarrollar. Los componentes de la arquitectura se observan en la siguiente Figura 3.

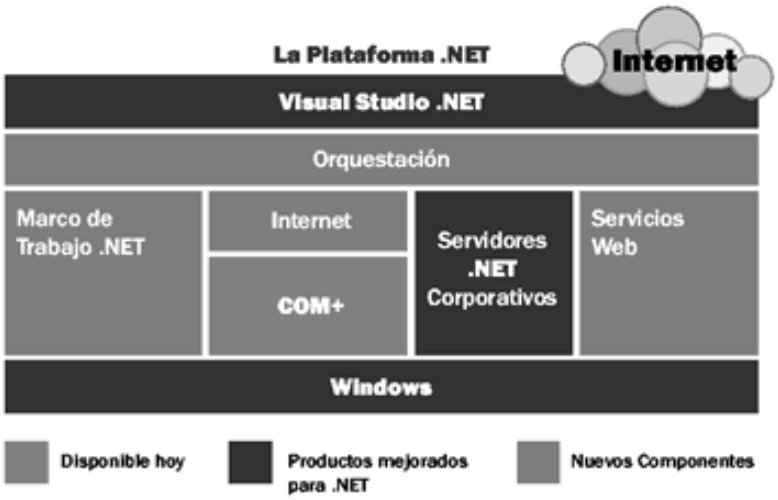


Figura 3: Componentes de la arquitectura .Net

La plataforma .NET provee las herramientas y tecnologías para transformar a Internet en una plataforma de computación distribuida en gran escala. Esta plataforma además soporta los estándares sobre los cuales se basan los servicios Web.

2.5.2 Visual Studio .NET

Para poder desarrollar servicios para este tipo de arquitectura, Microsoft ha lanzado el entorno de desarrollo denominado Visual Studio .NET. El objetivo principal de este entorno de desarrollo es simplificar el desarrollo de aplicaciones Windows y servicios Web permitiendo la elección del lenguaje de programación más adecuado (Visual Basic, C++ o C#) [10].

Microsoft ha desarrollado un lenguaje nuevo de programación basado en C++ denominado C# (C sharp). El concepto de C# es muy parecido a Java, un lenguaje que elimina las complicaciones innecesarias del C++ pero manteniendo su potencia. El principal objetivo de C# es eliminar el uso de Java y C++, con el

objetivo de reducir el coste de desarrollo de los servicios .NET.

2.5.3 Servicios Web

Simplificando, un servicio Web es un programa que se puede acceder a través de Internet utilizando protocolos estándar [10]. Estos servicios se ejecutarán en un servidor Web, no en los PCs clientes, permitiendo que los dispositivos que los utilicen sean más simples (simplemente se necesita un navegador Web). Para implementar un servicio Web es necesario resolver varios problemas:

- Representación de los datos: Para poder compartir datos entre distintas organizaciones se necesita un estándar de representación de datos. Este estándar es XML.
- Utilización del servicio: Se necesita un protocolo para definir cómo acceder y utilizar el servicio. Para ello se utiliza SOAP.
- Definición del servicio: Dado un servicio, para poder utilizarlo se necesita saber qué operaciones ofrece y cómo utilizarlas. Para esto, se utiliza el protocolo WSDL
- Publicación del servicio: Las empresas que proveen servicios y los clientes que quieran utilizarlos necesitan un mecanismo para que se conozcan, es decir, algo parecido a las páginas amarillas. Este es el objetivo del protocolo UDDI.

Todo estos mecanismos son los que utiliza la plataforma .NET para implementar sus servicios Web, que en muchos casos denomina "Servicios Web XML".

2.5.4 Soluciones basadas en servicios

- A medida que crece Internet y las tecnologías relacionadas, las organizaciones buscan integrar sus sistemas entre departamentos y organizaciones, ha evolucionado un enfoque de generación de soluciones basado en servicios.

- Los servicios son conceptualmente similares a los componentes tradicionales, salvo que los servicios encapsulan sus propios datos y no forman parte, de la aplicación sino que son utilizados por ésta. Aplicaciones y servicios que necesitan integrarse se pueden generar en distintas plataformas, por distintos equipos, en diferentes programas y se pueden mantener y actualizar de forma independiente, por tanto, es esencial implementar la comunicación entre ellos con el mínimo acoplamiento.

- Para implementar la comunicación entre los servicios se emplean técnicas basadas en mensajes, que proporcionan altos niveles de solidez y escalabilidad. Se puede implementar la comunicación de mensajes de forma explícita (por ejemplo, escribiendo código para enviar y recibir mensajes de Message Queue Server), o se pueden utilizar componentes que administran la comunicación de forma implícita (por ejemplo, con un servidor Proxy de servicios Web generado por Microsoft Visual Studio® .NET).

- Desde un punto de vista de alto nivel, se puede considerar que la solución basada en servicios está formada por varios servicios, los cuales se comunican entre sí pasando mensajes. Desde el punto de vista conceptual, los servicios se pueden considerar como componentes de la solución global. Sin embargo, internamente el servicio está formado por componentes de software, los cuales se pueden agrupar de forma lógica en servicios de presentación, empresariales y de datos, tal y como se muestra en la Figura 4: Arquitectura de tres capas basada en servicios.

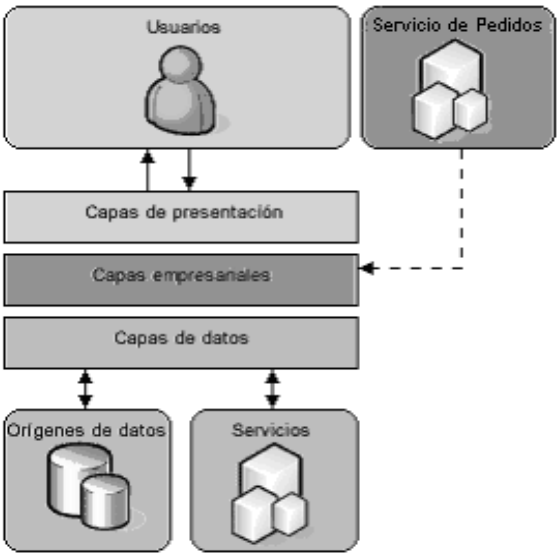


Figura 4: Arquitectura de tres capas basada en servicios

2.6 APLICACIÓN DEL MARCO TEÓRICO

A continuación se presenta un paralelo entre los temas del marco teórico y la aplicación de cada uno dentro del trabajo de investigación desarrollado:

Tema marco teórico	Aplicación en la tesis
Comunidades virtuales	Propender por la creación de una comunidad virtual

	interesada en una misma área de conocimiento, el desarrollo de software. Facilitar el contacto entre investigadores geográficamente dispersos y ofrecer servicios de gestión de proyectos de software.
Trabajo colaborativo (colaboración, coordinación y comunicación)	La herramienta intenta: • Promover la colaboración: el desarrollo de software es una actividad colaborativa, puesto que los trabajadores deben apoyarse entre si, para alcanzar una meta en común. El modo de colaboración tiene que ser establecido dentro del equipo para asegurar un entendimiento compartido del desarrollo del proceso y que los productos generados satisfagan las necesidades detectadas. La herramienta juega el papel de un espacio de trabajo común para los integrantes de la comunidad. La herramienta debe brindar facilidades para compartir información, notificación explicita de acciones de los demás y notificación de asignación de responsabilidades. • Mejorar la coordinación: es necesario coordinar gente y proyectos de software que hacen uso de procesos de desarrollo de software, aplicables como plantillas, en condiciones de dispersión geográfica. Las personas que trabajan juntas tienen que saber que es lo que tienen asignado para hacer y conocer la disponibilidad laboral de sus compañeros de trabajo que permitan planear horas de trabajo conjunto. • Facilitar la comunicación: la herramienta intenta afrontar los desafíos impuestos por el desarrollo distribuido de software por una comunidad, uno de ellos la comunicación. De esta manera la herramienta estimula la interacción distribuida asincrónica (diferente lugar, diferente tiempo), por medio de tecnologías como el correo electrónico enviando mensajes de notificación, el calendario y los mensajes públicos dentro del proyecto ó mensajes privados entre integrantes.
El proceso unificado de desarrollo (UP)	UP es un proceso genérico con naturaleza adaptativa que permite adecuarse a proyectos de alta o baja

	complejidad y variedad de organizaciones. El entorno software a desarrollar intenta soportar un modelo de proceso, inspirado en el espíritu de UP, adecuado para el desarrollo distribuido de software para una comunidad geográficamente dispersa.
Desafíos encontrados en el desarrollo distribuido de software	Se toman como referencia los desafíos en el desarrollo distribuido de software para el desarrollo de la herramienta y las soluciones propuestas para afrontar de mejor manera los desafíos de comunicación, coordinación, infraestructura, disponibilidad y gestión.
Administración de proyectos	La herramienta permitirá la definición de proyectos que hacen uso de plantillas de procesos de desarrollo de software. Para cada proyecto se debe gestionar tiempo, en cuanto a planificación de fases, iteraciones y actividades; gestionar personas, en cuanto a asignación de roles y tareas, y gestionar productos, en cuanto a artefactos y versiones de artefactos a generar.
La plataforma .NET	La plataforma de desarrollo seleccionada para el proyecto es .NET, por poseer experiencia previa en el desarrollo de aplicaciones Windows, y conociendo que la plataforma .NET posee una librería de clases base reutilizable para cualquier tipo de aplicación (Windows, Web, Mobile, etc), permitiendo que se maneje un modelo de programación común. Además el entorno de desarrollo que ofrece la plataforma .NET es el mismo para cualquier tipo de aplicación y la curva de aprendizaje es relativamente corta. También se posee un interés personal por aprender esta tecnología.

Tabla 4: Aspectos tomados del marco teórico para aplicar en la tesis

2.7 ESTADO DEL ARTE

2.7.1 Investigaciones previas

En la actualidad se pueden encontrar trabajos de investigación y desarrollo con contenidos similares al tema que se quiere desarrollar en este proyecto (I+D), el desarrollo distribuido, los cuales sirven como

referentes para el mismo. A continuación se mencionan algunos de estos trabajos realizados y se resaltan las similitudes, diferencias y reflexiones respecto del trabajo de grado que se quiere desarrollar:

- **Modelo de Administración de Proyectos: propuesta para la ejecución en un entorno de desarrollo de software físicamente distribuido**

  Zanoni (2004), propone un modelo de administración de proyectos que incluya el UP y usando UML, para el desarrollo de software de e-business, en un entorno físicamente distribuido. El modelo esta dividido en seis fases: definición de requerimientos, exploración y definición del proyecto, producción de procesos, evaluación, transición e integración. En el futuro, la intención es desarrollar un soporte software para el modelo y aplicar este software dentro del entorno en estudio.

  Resultado Obtenido. El modelo propuesto difiere de los presentes en que busca la unión entre el desarrollo de procesos en entornos distribuidos y la administración de procesos, incorporando un ciclo de vida en espiral, la orientación a objetos y el lenguaje UML. Un énfasis especial se ha dado en la etapa de Definición de Requerimientos y la integración de fases, como los resultados más afectados en el entorno de desarrollo distribuido de software. El modelo propuesto fue el resultado de la necesidad de encontrar respuestas a un problema crítico que estaba presente en los entornos de trabajo. Este problema esta centrado en las dificultades de comunicación debido a la distancia física y cultural entre grupos de usuarios y desarrolladores. Debido al poco tiempo no fue posible aplicar el modelo propuesto.

  Reflexión. Esta investigación presenta similitud con el trabajo de grado que se quiere realizar. Por lo tanto los problemas que se encontraron en este trabajo son casi los mismos que se enfrentarán para el desarrollo de este proyecto de grado. Sin embargo esta investigación solo busca el desarrollo de software de e-business mientras que en el trabajo de grado se quiere construir una plataforma que soporte el desarrollo de software en general. Una carencia de la investigación anterior es que no se puso en práctica por no contar con el tiempo necesario, obteniendo únicamente la información de la investigación. Este proyecto de grado aportará como valor agregado, el soporte informático para un proceso distribuido que beneficiará a una comunidad de I+D geográficamente dispersa en varias universidades del país interesadas en el emprendimiento de proyectos de desarrollo de entornos de modelamiento y simulación de modelos integrados y de otra parte constituirá un logro de valor agregado al proyecto SIMEP-SW [3] de la Universidad del Cauca.

❖ **Aplicación de un proceso ligero en entornos distribuidos: Distributed eXtreme Programming (DXP)**

XP enfatiza la necesidad de tener los miembros del equipo de trabajo físicamente cercanos. Sin embargo, por varias razones, esto no siempre puede ser posible. Para mejorar esta situación, un grupo de personas ha propuesto una idea llamada "Distributed eXtreme Programming" (DXP), la cual involucra los méritos de XP y lo aplica en un entorno de un equipo distribuido. La experiencia obtenida por este grupo de trabajo muestra que DXP puede ser efectiva y que merece la pena en proyectos con equipos que están distribuidos geográficamente. DXP ofrece muchos desafíos, los cuales pueden superarse. Según Kircher (2001), DXP consigue afectar cuatro prácticas de XP, para las cuales se proponen las siguientes soluciones:

1. **Planificar.** Cuando existe dispersión geográfica entre los desarrolladores y los clientes, el proceso de planificación puede apoyarse mediante video conferencia y las aplicaciones compartidas remotamente.
2. **Programación en parejas.** La Programación en Parejas Remota (RPP[14]) debe ser usada, cuando los miembros del equipo no se encuentran en la misma ubicación física. Esto requiere videoconferencia y soporte para aplicaciones compartidas, que compartan el Entorno de Desarrollo Integrado (IDE).
3. **Integración Continua.** Si un miembro del equipo esta trabajando en un sitio central, el/ella puede invitar a otro miembro remoto para realizar integración en la máquina. Si ambos miembros son remotos, esto no es posible.
4. **Cliente en el sitio.** La videoconferencia deberá ser usada para involucrar al cliente remoto. El cliente necesita cumplir un cierto conjunto de reglas tales como coordinación y disponibilidad.

Experiencia Reportada

- **El Proyecto**: Software llamado "Web-Desktop" el cual proporcionará el entorno de trabajo para DXP. El software es accesible vía Web. Todas las aplicaciones pertenecientes al software se ejecutan en la máquina donde fue descargado. Tal software permitiría a un miembro del equipo usar cualquier PC conectado a la Internet, logearse y tener la misma sensación y el mismo entorno de trabajo. Un miembro del quipo de trabajo móvil podría ir ahora a un café Internet y conectarse con su Web cam y/o su micrófono y conseguir conectarse al resto del equipo. No seria necesario descargar e instalar software en cada máquina que el miembro usa.

[14] Remote Pair Programming

- **Experiencia Vivida**
    - **Planificar.** Se ejecutaron varias sesiones de videoconferencia con el cliente. Se uso un editor regular y una aplicación compartida de software.
    - **Programación en Parejas.** Se asignaron historias de usuarios a las parejas y se comenzó el proceso de desarrollo. Se utilizo videoconferencia y aplicaciones compartidas. Se utilizo el e-mail para asignar citas según el horario.
    - **Integración Continua.** Se integraron los cambios desde el lugar de desarrollo hasta el lugar principal.
    - **Cliente en el sitio.** Se utilizo videoconferencia, también el email para comunicar la hora y el canal para las sesiones de videoconferencia.

Reflexión.

- Usando una combinación de comunicación síncrona, como la videoconferencia, y comunicación asíncrona, como el email, resulta ser más efectivo. Sin embargo, lo que se pierde es la presencia física lo cual nunca puede ser completamente sustituido con una herramienta de videoconferencia.
- El desarrollo paralelo lanza el problema de la integridad del código fuente. Para eso se utilizó un email token para serializar el acceso a cambios a los equipos trabajando en una sección de código común.
- Este proyecto ofrece una buena fuente de información debido a que la investigación realizada en cuanto a DXP fue colocada en práctica. Sin embargo es solo una extensión de XP para procesos de desarrollo distribuido por lo que presenta mayor aplicación para procesos de desarrollo pequeños. Lamentablemente la herramienta software desarrollada en esta investigación no está disponible para el público en general, ya que podría ser un gran punto de apoyo para la herramienta que dará soporte a este trabajo de grado.

❖ **Antecedente de un entorno de soporte para un proceso de desarrollo distribuido [11]: PROYECTO GÉNESIS**

Propone el desarrollo de un entorno distribuido que soporta la ingeniería del software cooperativa, permitiendo mantener coordinadas y controladas las actividades relativas al desarrollo de software aunque se realicen desde ubicaciones geográficamente distantes o simplemente UD. Cada UD podría ejecutar instancias de un proceso (o sub-proceso) de desarrollo, las entradas y/o salidas de esos procesos (o sub-procesos) son en realidad artefactos de software (ejemplo: módulos ejecutables, documentos de diseño, pruebas etc). Cada UD mantiene la autonomía de sus procesos internos y sus tecnologías de soporte (por ejemplo: sistemas de gestión de artefactos y sistemas de

gestión del proceso). La definición de la organización distribuida del proceso de desarrollo GENESIS propone la necesidad de distinguir entre dos niveles:

- **El primer nivel:** consiste en la definición del proceso global (también conocido como el proyecto), el cual describe el flujo de trabajo del proceso de desarrollo de software, los artefactos elaborados y como los sub-procesos son asignados a las diferentes UD.
- **El segundo nivel:** permite a una única UD, refinar el subproceso asignado a esta, mediante una definición del proceso local compuesto por actividades. Con el propósito de definir tanto el proceso local como el global, un lenguaje de definición de procesos debe ser provisto, con instrucciones específicas para construir y gestionar las tareas (y sus actividades) a nivel local y global.

Reflexión.

Una de las propuestas de GENESIS, es la capacidad de gestionar los aspectos dinámicos del proceso de software, específicamente las desviaciones que pueden ocurrir en cualquier momento durante la ejecución del proyecto. Este proyecto es semejante al que se plantea en el presente trabajo de grado, con la diferencia que génesis esta liderado por una comunidad de empresas europeas, por lo que su alcance y objetivos son mas ambiciosos, mientras que el actual trabajo plantea una solución al paradigma de desarrollo distribuido adecuado para una comunidad (I+D) geográficamente dispersa en varias universidades del país, que participan en desarrollos conjuntos de entornos de simulación y constituye un logro para el proyecto SIMEP-SW [3] de la Universidad del Cauca. Sin embargo en el portal de este proyecto [11] solo muestran algunos resultados hasta 2002, no pudiéndose determinar el éxito y/o culminación de este proyecto.

❖ **Síntesis de las investigaciones previas**

Teniendo en cuenta las investigaciones previas analizadas, se puede constatar que está emergiendo una marcada tendencia (a nivel global) relativa al paradigma de desarrollo de software. Esta tendencia, sitúa el desarrollo de software, desde un ambiente centralizado hacia un ambiente geográficamente disperso. Las investigaciones tienen como objetivo afrontar esta nueva tendencia proponiendo un nuevo modelo, adaptando una metodología para superar los desafíos que se presentan en entornos distribuidos y proponiendo la construcción de un entorno distribuido que soporte la ingeniería del software cooperativa. Estos esfuerzos se han realizado a nivel internacional, pero la necesidad empieza a ser evidente a nivel nacional. Como ejemplo se presenta la necesidad detectada por el grupo SIMON [2], de ejecutar un proyecto de software con la colaboración de equipos de investigación que se encuentran distribuidos geográficamente, en el país. SIMON reconoce que el paradigma de desarrollo

distribuido es una alternativa interesante para superar este desafío, pero aún no cuenta con una herramienta que apoye este tipo de desarrollo. Para apoyar la necesidad detectada a nivel nacional, se pretende construir una herramienta que fomente la cooperación, el trabajo colaborativo y la comunicación entre grupos de desarrolladores geográficamente dispersos, para facilitar el desarrollo de proyectos de software.

2.7.2 Herramientas

La exploración realizada de herramientas de gestión de proyectos, ha permitido identificar aspectos positivos y aspectos en contra, relacionados con las características de la herramienta construida en este trabajo de grado.

- **Guía de proyectos RUP**

La Guía de Proyectos RUP (RUP Project Guide) integra el Rational Unified Process (RUP) [13] con Microsoft Project 2002 para proporcionar al gerente de proyectos una manera fácil de usar un framework para planear proyectos RUP. Esta herramienta está disponible para descargar exclusivamente en la Rational Developer Network (www.rational.net). Para usar esta herramienta se necesita tener instalado Microsoft Project 2002 y la plantilla de proyectos RUP (RUP Project Guide.mpt). La interfaz de la herramienta se observa en la Figura 5: Interfaz de Project usando la plantilla proyectos RUP.

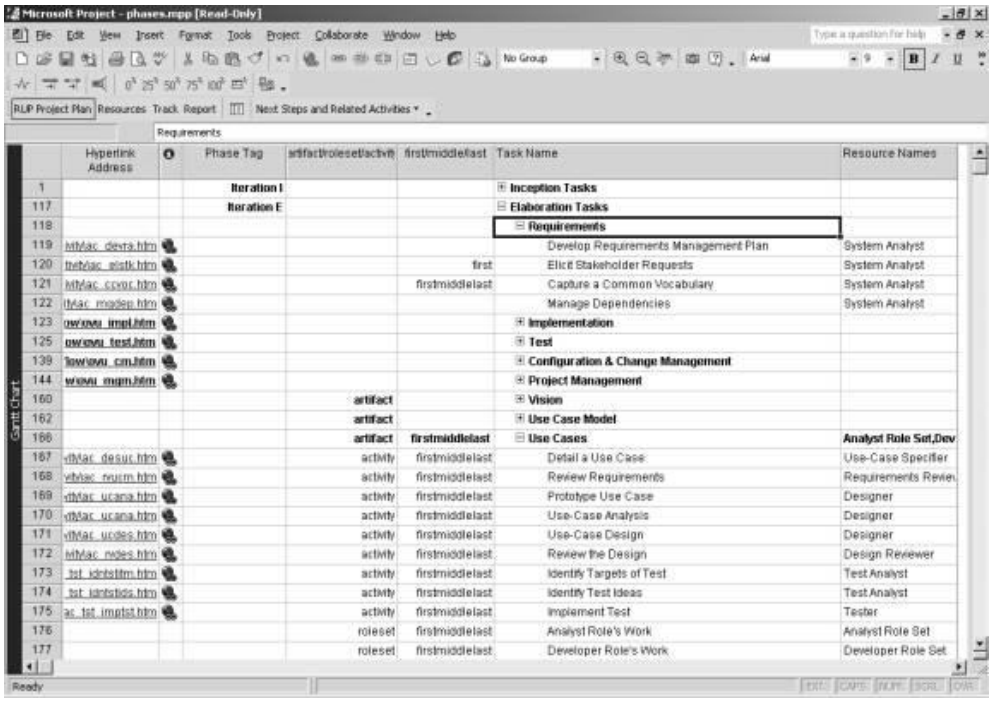


Figura 5: Interfaz de Project usando la plantilla de proyectos RUP

Aspectos a favor

- Esta herramienta está orientada a soportar la planeación de proyectos de software, usando el ciclo de vida de la metodología de desarrollo RUP [13] . La herramienta cuenta con dos planes en los proyectos RUP: un plan de proyecto y un plan de iteración por cada iteración. El plan de proyecto sugiere definir los hitos de las fases y el número de iteraciones. El plan de iteración sugiere definir ítems específicos de la iteración tales como entregables, duración de la iteración, riesgos a ser mitigados, actividades y asignación de responsabilidades. Utiliza como plantilla el RUP (Rational Unified Process), para seleccionar, solamente, las fases, los entregables y actividades que se necesitan para planear el proyecto iterativo, y como soporte de conocimiento al tener acceso a la documentación de descripción de RUP y plantillas para entregables.
- Se cuenta con toda la funcionalidad que brinda Microsoft Project, como herramienta de gestión de proyectos.

Aspectos en contra

- La herramienta Microsoft Project al presentar inter-operabilidad con Project Server, es posible exportar y compartir información para todo el equipo de desarrollo en el Web, pero no soporta la funcionalidad para permitir el trabajo en comunidad basado en el plan de proyecto que se genera, al

no soportar gestión de usuarios, gestión de equipos de trabajo de los miembros de la comunidad, gestión de documentación (repositorio de artefactos) obtenida en el proyecto y comunicación entre miembros de la comunidad, a través de Internet.

- **Microsoft Project**

Es una aplicación de Microsoft que ayuda al usuario a crear planes de proyectos, comunicarlos a otros usuarios y adaptarse a los cambios a medida que estos se van produciendo. Funciona, en muchos sentidos, en forma similar a otras aplicaciones de Microsoft. Las barras de menús, los comandos, las barras de herramientas, los menús contextuales y los cuadros de dialogo tienen mucho en común con Microsoft Excel, Word y Power Point, lo cual facilita los primeros pasos de su uso. La interfaz de la herramienta se puede observar en la Figura 6: Interfaz de Microsoft Project.

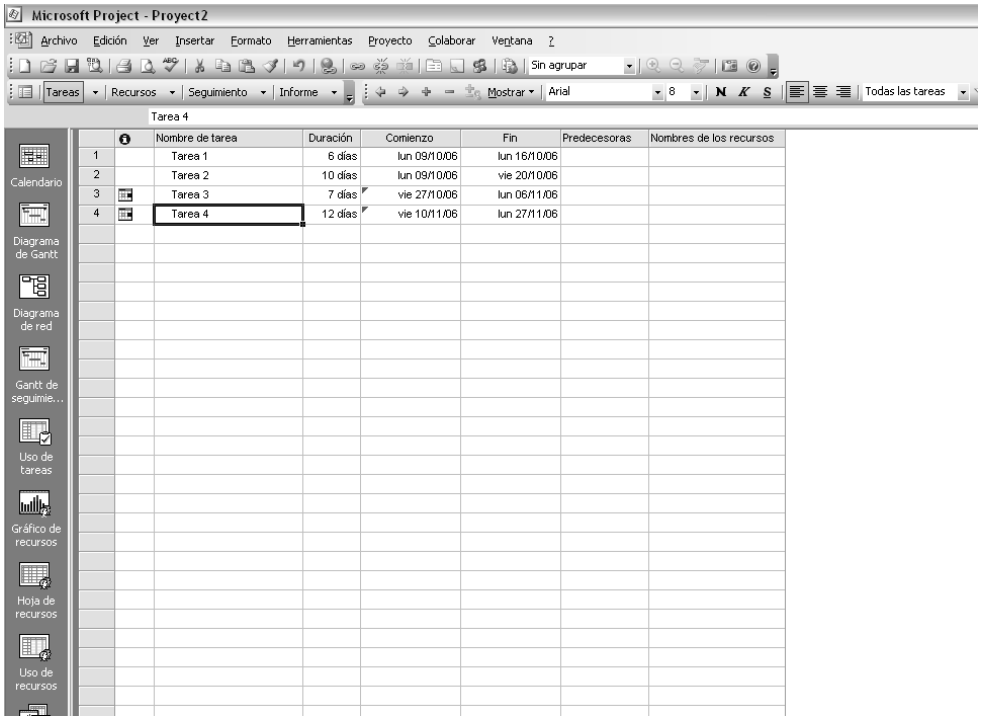


Figura 6: Interfaz de Microsoft Project

Aspectos a favor

- Utiliza el idioma español.
- Permite trabajar con múltiples proyectos.
- Permite elaborar un proyecto desde la fase preliminar hasta la fase final.
- Además de facilitar la programación, también se puede seguir el progreso del proyecto para:

- Identificar y resolver los problemas que se produzcan.
- Generar informes de estado para la administración y los participantes en el proyecto, en formato html.
- Conservar datos históricos que ayuden a planificar proyectos futuros de una forma mas precisa.
- Actualizar periódicamente la programación para reflejar el progreso del proyecto.
- Utilizar diagramas de Gantt para la visualización y control de las actividades del proyecto.
- Hacer uso de un calendario de las tareas de mayor importancia.

Aspectos en contra

- No soporta la funcionalidad para permitir el trabajo de la comunidad de desarrollo, en torno a la planificación del proyecto, al no facilitar la gestión de usuarios, gestión de equipos de trabajo de la comunidad, gestión de documentación (repositorio de artefactos) obtenida en el proyecto, comunicación y coordinación entre miembros de la comunidad, a través de Internet.
- Ofrece soporte para proyectos en general, pero no brinda la funcionalidad adicional requerida para soportar proyectos de desarrollo de software basados en una metodología de ciclo de vida del proyecto, ya que se tiene en cuenta la relación entre tareas (vista en el Diagrama de Gantt) pero no se pueden agrupar en fases e iteraciones, para la gestión del ciclo de vida del proceso instanciado por el proyecto. Además no existe la posibilidad de manejar distintos roles, asociado al proceso de desarrollo de software, para la asignación de responsabilidades (actividades y artefactos).
- El costo de la licencia de esta herramienta comercial es muy alto, lo que constituye una limitante para algunos usuarios con necesidad de usarla.

♦ **PHPCollab**

Es una aplicación que permite la colaboración a través de Internet a miembros de proyectos. Permite a cada uno publicar información y compartirla con los demás, pero también la administración de proyectos y de las tareas. Es una aplicación Open Source con licencia libre GNU. La interfaz de la herramienta se presenta en la Figura 7: Interfaz de PHPCollab-Adicionando una tarea.

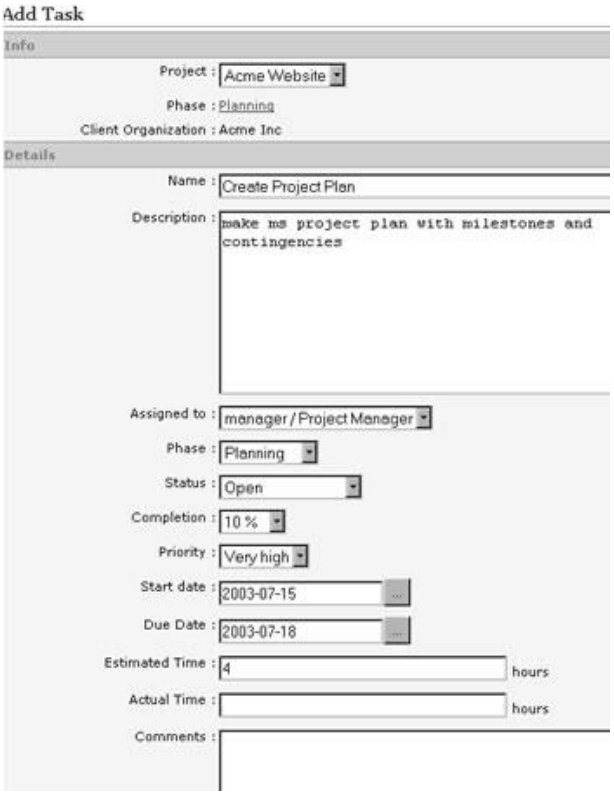


Figura 7: Interfaz de PHPCollab-Adicionando una tarea

Aspectos a favor

- Todos los proyectos en PhpCollab siguen una metodología de ciclo de vida del proyecto, lo que significa que los proyectos están divididos en fases, luego tareas y luego subtareas. Existen unas fases predeterminadas. Las tareas y subtareas se crean. El administrador del sistema puede personalizar estas fases en términos de orden y nombres.
- Sencilla administración con una interfaz gráfica simple.
- Posibilita dar diferentes permisos para cada usuario.
- Realiza control de los accesos que realiza un usuario.
- Se puede realizar la configuración del idioma y del tamaño de los ficheros permitido.
- Soporta notificaciones automáticas vía e-mail de eventos y modificaciones en el proyecto.
- Permite la asignación de tareas al personal del proyecto.
- Visualiza gráficas del progreso de las tareas.
- Permite la creación de foros para los proyectos.
- Tiene un sistema de búsqueda basado en palabras claves.
- Exporta informes en formato pdf.

- Permite la grabación del proyecto en archivos CVS (Concurrent Version System).

Aspectos en contra

- No permite la conformación y gestión de equipos trabajo entre los miembros del proyecto.
- No existe la posibilidad de manejar distintos roles, asociado al proceso de desarrollo de software, para la asignación de responsabilidades (actividades y artefactos).

♦ **FastTrack**

Es una herramienta de gestión de proyectos que permite organizar, realizar seguimiento y gestionar los detalles de un proyecto. Es software propietario. Se utilizó la versión 9.0.1 de prueba que no tiene habilitada la funcionalidad de salvar en archivos. La interfaz se puede ver en la Figura 8: Interfaz de FastTrack.

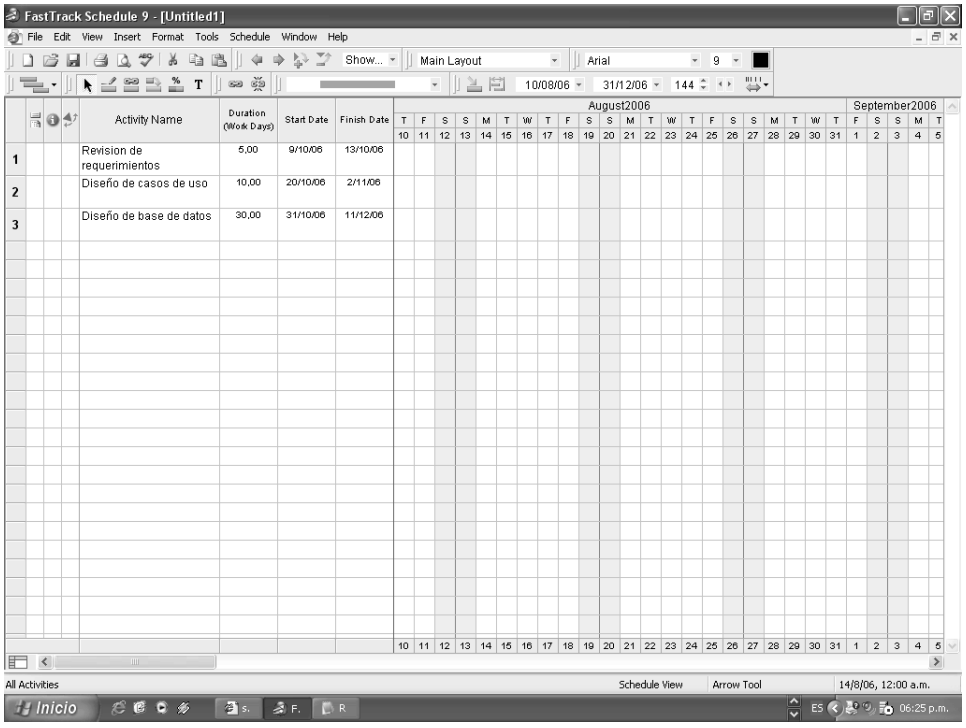


Figura 8: Interfaz de FastTrack

Aspectos a favor

- Es multiplataforma (Mac OS, Windows, e incluso una versión Palm OS).
- Capaz de exportar y recibir información desde Project.

- Permite definir tareas.
- Visualiza las tareas en una vista Gantt que permite dar de alta a las tareas.
- Permite definir y comentar hitos.
- Permite establecer dependencias entre tareas e hitos.
- Facilita la definición y uso de recursos.
- Se puede tener un calendario de días laborales para cada recurso (persona o equipo).
- Realiza seguimiento de tareas, establece ruta crítica.
- Tiene flexibilidad en el diseño grafico, los colores y aspectos de hitos y barras de tareas son llamativos y personalizables.

Aspectos en contra

- No soporta la funcionalidad para permitir el trabajo de la comunidad de desarrollo, en torno a la planificación del proyecto, al no facilitar la gestión de usuarios, gestión de equipos de trabajo de la comunidad, gestión de documentación (repositorio de artefactos) obtenida en el proyecto, comunicación y coordinación entre miembros de la comunidad, a través de Internet.
- Ofrece soporte para proyectos en general, pero no brinda la funcionalidad adicional requerida para soportar proyectos de desarrollo de software basados en una metodología de ciclo de vida del proyecto, ya que se tiene en cuenta la relación entre tareas (vista en el Diagrama de Gantt) pero no se pueden agrupar en fases e iteraciones, para la gestión del ciclo de vida del proceso instanciado por el proyecto. Además no existe la posibilidad de manejar distintos roles, asociado al proceso de desarrollo de software, para la asignación de responsabilidades (actividades y artefactos).
- El costo de la licencia de esta herramienta comercial es muy alto, lo que constituye una limitante para algunos usuarios con necesidad de usarla.

♦ **BaseCamp**

Basecamp es una herramienta para colaboración en proyectos. Se sustenta en la idea de que los proyectos no fallan por la carencia de gráficos o informes sino por la falta de comunicación y colaboración entre los participantes. Para probar su funcionalidad se utilizó la herramienta que ofrece el plan gratuito, la cual permite gestionar un solo proyecto, obteniendo una cuenta libre. La interfaz se puede ver en la Figura 9: Interfaz de BaseCamp.

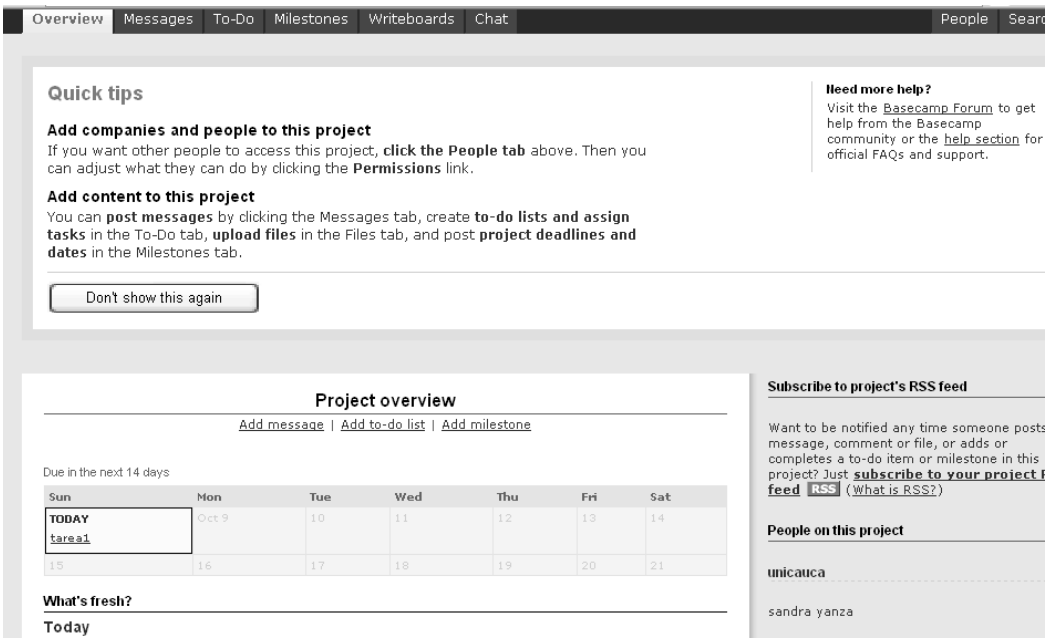


Figura 9: Interfaz de BaseCamp

Aspectos a favor

- Diseño simple y elegante basado en Web.
- No se necesita descargarlo, instalarlo o configurarlo, todo lo que necesita es un navegador Web y una conexión a Internet.
- Permite crear tareas e hitos, y asignar tareas.
- Maneja un calendario simple para eventos.
- Permite compartir archivos.
- Permite crear reportes para ver las tareas de una persona en una fecha específica.
- Permite cambiar los colores de la interfaz.
- Permite gestionar el acceso limitado a las personas.
- Dispone una pizarra para comentarios.

Aspectos en contra

- Ofrece planes con diferentes costos, dependiendo de la funcionalidad ofrecida. El costo cubre el alojamiento de la información, encriptación de datos, servicio de Chat, entre otros.
- No soporta la funcionalidad para permitir la gestión de equipos de trabajo de la comunidad.
- Ofrece soporte para proyectos en general, pero no brinda la funcionalidad adicional requerida para soportar proyectos de desarrollo de software basados en una metodología de ciclo de vida del

proyecto, ya que se tiene en cuenta la relación entre tareas (vista en el Diagrama de Gantt) pero no se pueden agrupar en fases e iteraciones, para la gestión del ciclo de vida del proceso instanciado por el proyecto.

- No existe la posibilidad de manejar distintos roles, asociado al proceso de desarrollo de software, para la asignación de responsabilidades (actividades y artefactos).

♦ **SourceForge.net**

Es el mayor repositorio de código abierto de la Red, una red comunitaria donde se desarrolla, almacena y centraliza el trabajo de más de un millón de programadores. Ofrece un espacio Web gratuito para todos los proyectos de código abierto y software libre que se deseen llevar a cabo. Permite crear y manejar proyectos, organizarlos en categorías comunes, compartir recursos y retomar viejos proyectos o redireccionar trabajos ajenos. El proceso de obtener una cuenta de usuario fue bastante demorado y tedioso al tener que completar una larga serie de formularios. Además la aprobación de la cuenta se extendió a varios días. La interfaz de esta herramienta se puede ver en la Figura 10: Interfaz de SourceForge.net.

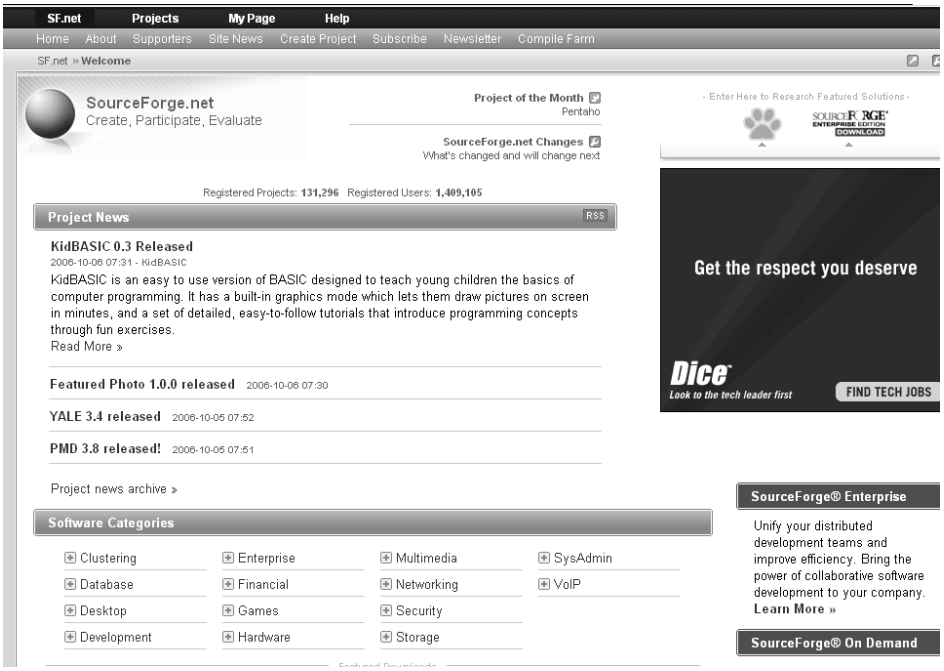


Figura 10: Interfaz de SourceForge.net

♦ **BSCW (Basic Support for Cooperative Work)**

La plataforma BSCW, es una herramienta que permite definir espacios de trabajo compartido y permite usar este espacio de trabajo para almacenar y compartir documentos, que traten sobre un proyecto o sobre un grupo de trabajo en particular.

Aspectos a favor

- Aporta una gran variedad de utilidades en lo referente al desarrollo de tareas en grupo como, por ejemplo, creación de zonas para el trabajo colaborativo, entrada restringida a esas áreas y limitar sus posibilidades para la manipulación de la información mediante derechos otorgados a los usuarios sobre los documentos. Estos pueden ser textos, imágenes, sonidos, anotaciones sobre los mismos, comentarios, listas de direcciones, etc. y desde estas zonas se permite compartirlos y almacenarlos, siempre disponibles para el resto de los miembros.

- Para usar BSCW se requiere el uso de un navegador Web normal y se necesitan tres condiciones básicas: dirección de correo electrónico, nombre del usuario y palabra clave, los dos últimos se establecen en el momento de registrarse como cliente y son el método de asegurar el acceso controlado a las áreas.

- Las zonas de trabajo pueden contener diferentes tipos de objetos:

    - Carpetas. Son las contenedoras de los documentos.
    - Documentos. BSCW admite la manipulación de numerosos tipos de ficheros y soporta diferentes formatos de texto, bases de datos e incluso archivos comprimidos.
    - Enlaces a otras paginas web.
    - Citas y reuniones. BSCW aporta una serie de mecanismos que admiten la posibilidad de establecer reuniones o encuentros virtuales entre sus miembros para la solucion de todo tipo de problemas. De este modo, es posible organizarlas mediante videoconferencia Mbone o chat a través de un sencillo sistema de calendario y planificación.

- El sistema contiene un sofisticado modelo de derechos de acceso que, por ejemplo, permite que un usuario tenga un control total sobre los objetos contenidos en una determinada carpeta, mientras que el resto de usuarios sólo tenga acceso de lectura o que no tenga ningun acceso al contenido de esa misma carpeta. De hecho, es necesaria la figura de un gestor de cada espacio para controlar los miembros de ese espacio de trabajo compartido.

- Cuenta con un sistema de notificación de eventos, que informa a los usuarios sobre las actividades llevadas a cabo en las diversas zonas. Esta información es posible obtenerla de diversas formas, siendo la más cómoda el envío de mensajes de correo electrónico que el sistema realiza automáticamente.

Aspectos en contra

- No brinda la funcionalidad requerida para soportar una metodología de ciclo de vida del proyecto de desarrollo de software, ya que no es posible programar actividades, agrupadas en iteraciones y en fases, para la gestión del ciclo de vida del proceso instanciado por el proyecto.

- No existe la posibilidad de manejar distintos roles, asociados al proceso de desarrollo de software, para la asignación de responsabilidades (actividades y artefactos).

♦ **Síntesis del análisis de las herramientas utilizadas para gestión de proyectos**

A continuación se presentan las principales características de las herramientas:

	Herramientas para gestión de proyectos					
	Guía de proyectos de RUP	**Microsoft Project**	**PhPCollab**	**FastTrack**	**BaseCamp**	**SourceForge.net**
Características						
Programación dinámica de tareas	√	√	√	√	√	√
Diagramas de Gantt, calendario y hojas de tareas	√	√		√		
Guía de proyectos, ciclo de vida	√		√			
Multiplataforma				√		
Informes de estado basados en Web	√	√			√	√
Seguimiento de problemas	√	√		√		
Informes en tiempo real	√	√			√	
Gestión de cuentas de usuario			√		√	√

Repositorio de archivos			√		√	√
Costo	√	√		√	√	
Gestión de equipos de trabajo						
Mecanismos de comunicación entre participantes			√	√	√	
Idioma español	√	√				
Asignación de responsabilidades	√	√	√	√	√	√
Notificaciones automáticas via e-mail			√			
Interfaz Web			√		√	√

Tabla 5: Herramientas utilizadas para gestión de proyectos

Las herramientas analizadas presentan aspectos positivos que se tienen en cuenta para la construcción de la herramienta propuesta, pero los aspectos en contra resultan ser elementos faltantes para proporcionar una alternativa adecuada para la puesta en marcha de proyectos de software por equipos que se encuentren dispersos geográficamente.

- Todas las herramientas analizadas permiten definir tareas que hacen parte de un proyecto, pero solamente dos herramientas soportan la planeación de proyectos, usando una metodología de ciclo de vida del proyecto, lo que significa que los proyectos están divididos en fases y tareas. Estas dos herramientas son la Guía de Proyectos de RUP y PHP-Collab.

- Respecto a los diagramas de Gantt y hojas de tareas, se destacan Microsoft Project y FastTrack, los cuales ofrecen estos servicios para el control de las actividades de un proyecto.

- La capacidad de ofrecer informes de estado basados en Web, es proporcionada por la Guía de Proyectos de RUP, BaseCamp y SourceForge.net.

- Con relación al soporte del trabajo en equipos bajo un ambiente de dispersión geográfica, ninguna de las herramientas analizadas permite gestionar equipos de trabajo. Sólo tres

herramientas manejan la gestión de cuentas de usuario, que son PHP-Collab, BaseCamp y SourceForge.net.

- Con respecto al soporte de la comunicación y coordinación entre los miembros que ejecutan un proyecto, PHP-Collab soporta notificaciones automáticas vía e-mail de eventos y permite la creación de foros; FastTrack permite tener un calendario de días laborales para cada recurso; y BaseCamp maneja un calendario simple para eventos, permite crear reportes para ver las tareas de una persona en una fecha especifica y pone a disposición una pizarra para comentarios.

- Todas las herramientas analizadas permiten la asignación de responsabilidades a usuarios de un proyecto, en este caso, asignación de tareas.

- Solo tres herramientas permiten que los usuarios accedan a la funcionalidad a través de Internet, desde cualquier lugar y en cualquier momento, las cuales son PHP-Collab, BaseCamp y SourceForge.net, facilitando el trabajo desde ubicaciones geográficamente remotas.

- Una herramienta que facilite el desarrollo de un proyecto software que se ejecute en condiciones de dispersión geográfica, necesita incluir un mecanismo de organización, publicación y descarga de las versiones que sean generadas por los miembros del proyecto. En este caso, PHP-Collab, BaseCamp y SourceForge.net dan soporte a esta funcionalidad.

- Las herramientas analizadas presentan los siguientes aspectos en general, para satisfacer las necesidades de comunicación, coordinación y colaboración, los cuales serán tomados en cuenta para apoyar las actividades de un proyecto de software, que se caracteriza por ser una actividad colaborativa:

    - Notificaciones automáticas via e-mail de eventos.
    - Creación de foros.
    - Calendario para dias laborales.
    - Calendario para eventos.
    - Reportes de las tareas.
    - Pizarra para comentarios.
    - Creación de zonas para el trabajo colaborativo entre varios miembros (que pueden, o no, estar lejanos geográficamente o pertenecer a diferentes organizaciones) desde donde compartir documentos para llevar a cabo proyectos comunes.

- Entrada restringida a esas areas, mediante un acceso controlado, permitiendo solamente a miembros previamente registrados y, en los casos que sea necesario, limitar sus posibilidades de manipulación de la información.
- Documentos siempre disponibles para todos los miembros del proyecto en cualquier momento, pues se encuentran almacenados en un servidor central y no en computadores locales.

- Las herramientas analizadas posee características interesantes con respecto al trabajo colaborativo y la gestión de proyectos, sin embargo, ninguna de estas herramientas se especializa en desarrollo de software con un proceso iterativo e incremental, por lo que se decidió implementar una nueva herramienta, teniendo en cuenta que las características que se encontraron en las herramientas analizadas se aprovecharán para construir parte de la concepción de lo que será la nueva herramienta.

- La herramienta a construir busca integrar aspectos de trabajo colaborativo con la gestión de proyectos de software, incluyendo la funcionalidad para soportar proyectos de desarrollo de software basados en una metodología de ciclo de vida del proyecto, con la posibilidad de agrupar tareas en iteraciones y en fases; manejar distintos roles, asociados al proceso de desarrollo de software, para la asignación de responsabilidades (actividades y artefactos); gestionar equipos de trabajo de la comunidad; gestionar la documentación (repositorio de artefactos) obtenida en el proyecto; y promover la comunicación y coordinación entre los miembros de la comunidad, a través de Internet.

A continuación, en el siguiente capitulo, se empezará a explicar la herramienta construida en este proyecto, como resultado principal, brindando una descripción de los motivos de su creación, la funcionalidad que soporta y la arquitectura y metodología empleada para su construcción.

CAPITULO 3: HERRAMIENTA DE SOPORTE DEL PROCESO DISTRIBUIDO DE DESARROLLO

Teniendo como referencia la descripción inicial del proyecto, los fundamentos de marco teórico y los antecedentes del estado del arte, detallados en los dos capítulos anteriores, se procede a explicar la herramienta construida y obtenida como resultado principal de este proyecto.

En este capitulo se realiza una descripción de la herramienta desde el punto de vista de los propósitos por la cual fue construida, la metodología de desarrollo de software utilizada, los actores que se distinguen en el sistema, la arquitectura lógica que representa la división modular de la funcionalidad que soporta, la arquitectura de implementación de la herramienta, y detalles de implementación.

3.1 DESCRIPCION DE LA HERRAMIENTA

El presente trabajo de grado se titula “Construcción de una herramienta software para soportar un proceso distribuido de desarrollo utilizado por una comunidad (I+D)”, y como tal tiene de resultado principal la herramienta software que ha adquirido el acrónimo “SEDISE” representación de las siglas en inglés de la siguiente descripción de la herramienta “**S**upport **E**nvironment for **Di**stributed **S**oftware **E**ngineering” (Entorno de Soporte para la Ingeniería de Software Distribuida).

El propósito fundamental de la herramienta construida consiste en apoyar la gestión de proyectos de desarrollo de software, al mismo tiempo que favorece la comunicación y coordinación de los grupos (i+d) geográficamente dispersos, en el marco de un proceso de software comunitario.

La Figura 11 muestra el esquema general que plantea la herramienta.

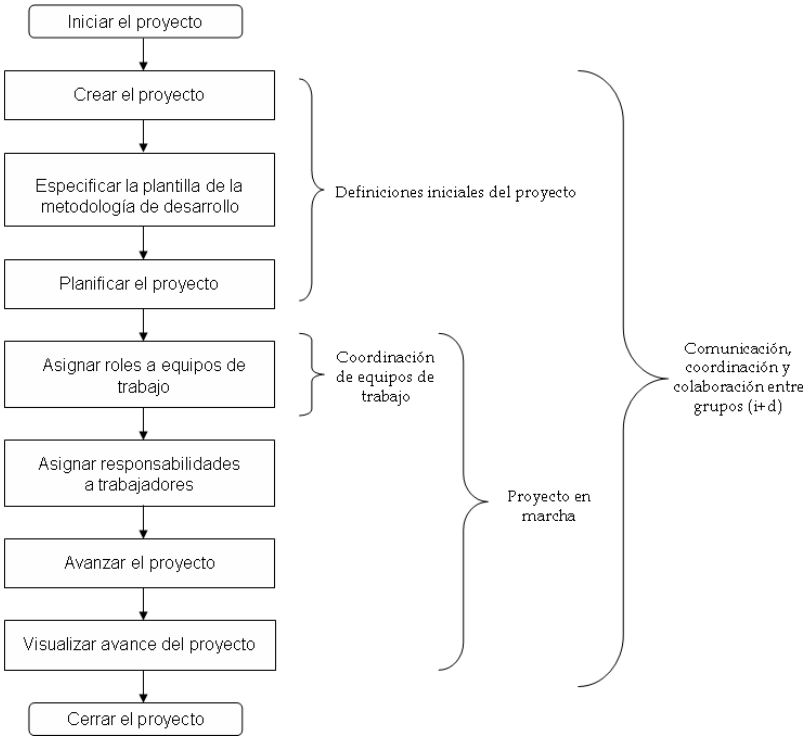


Figura 11: Esquema general de la herramienta

En la figura anterior se muestra en forma general el flujo de trabajo propuesto para manejar los proyectos de desarrollo de software. Todo proyecto de desarrollo tiene una etapa primaria en la cual se realizan definiciones iniciales, las cuales normalmente sólo se hacen antes de comenzar con un proyecto de desarrollo. Durante la ejecución de un proyecto se realizan procesos de gestión y coordinación, y los integrantes del proyecto empiezan a cumplir con las responsabilidades asignadas, entregando resultados, con el fin de lograr los objetivos planeados. Este flujo de trabajo se genera bajo un ambiente de dispersión geográfica de los equipos (I+D) que pertenecen a una comunidad de desarrollo, por lo cual la herramienta intenta soportar las necesidades de comunicación, colaboración y coordinación entre estos equipos.

La construcción de la herramienta se ha realizado teniendo en cuenta los siguientes propósitos:

- **Dar soporte al proceso distribuido de desarrollo utilizado por una comunidad (I+D)**

La herramienta construida tiene el propósito de apoyar el desarrollo de software por equipos que se encuentran dispersos geográficamente, tratando de mitigar los riesgos que se presentan tales como perdida de control sobre la sincronización de las tareas y las personas, fallas en la comunicación, falencias en la gestión y seguimiento de un proyecto de software.

- **Permitir la personalización de plantillas de proceso**

Teniendo en cuenta el marco teórico (ver numeral 2.3.4), según IBM [6], una plataforma que soporte el desarrollo geográficamente distribuido de software debería satisfacer la funcionalidad suficiente para permitir la personalización de plantillas de procesos para ser aplicadas por los proyectos que se definan. Por lo anterior, la herramienta pone a disposición dos plantillas basadas en el Proceso Unificado de Rational (RUP) [13], proceso compuesto por los siguientes elementos que se encuentran lógicamente relacionados: Fases, Disciplinas, Actividades, Pasos, Roles, Tipos de artefactos.

El usuario haciendo uso de este proceso, personaliza una plantilla para ser aplicada en el proyecto. La personalización de la plantilla consiste en:

- Seleccionar las fases que el usuario considere necesarias para ser aplicadas en el proyecto.
- Indicar el número de iteraciones a ejecutar en cada fase.
- Seleccionar las actividades a realizar en cada una de las iteraciones de cada fase.
- Definir el hito de cada fase, compuesto de tipos de artefactos que se deben cumplir al finalizar la fase.

- **Propiciar aspectos colaborativos en torno a la comunidad (I+D)**

La herramienta intenta promover la colaboración, mejorar la coordinación y facilitar la comunicación entre los miembros de una comunidad que trabaja en un proyecto software, de tal manera que se promueva el trabajo en equipo, se valoren los aportes de cada miembro, se forme y se estimule un sentimiento de pertenencia en la comunidad y se alcancen las metas propuestas. Para lo anterior se tienen en cuenta los fundamentos de trabajo colaborativo consignados en el marco teórico (ver numeral 2.2). A continuación se resume la influencia de los principales aspectos del trabajo colaborativo en la herramienta:

- **La colaboración presente en la herramienta:** el desarrollo de software es una actividad colaborativa, puesto que los trabajadores deben apoyarse entre si, para alcanzar una meta en común. El modo de colaboración tiene que ser establecido dentro del equipo para asegurar un entendimiento compartido del desarrollo del proceso y que los productos generados satisfagan las necesidades detectadas. La herramienta juega el papel de un espacio de trabajo común para los integrantes de una comunidad. La herramienta debe brindar facilidades para compartir información, notificación explicita de acciones de los demás y notificación de asignación de responsabilidades.

- **La coordinación presente en la herramienta:** es necesario coordinar gente y proyectos de software que hacen uso de procesos de desarrollo de software, aplicables como plantillas, en condiciones de dispersión geográfica. Las personas que trabajan juntas tienen que saber que es lo que tienen asignado para hacer y conocer la disponibilidad laboral de sus compañeros de trabajo que permitan planear horas de trabajo conjunto.

- **La comunicación presente en la herramienta:** la herramienta intenta afrontar los desafíos impuestos por el desarrollo distribuido de software por una comunidad, uno de ellos la comunicación. De esta manera la herramienta estimula la interacción distribuida asincrónica (diferente lugar, diferente tiempo), por medio de tecnologías como el correo electrónico enviando mensajes de notificación, el calendario y los mensajes públicos dentro del proyecto ó mensajes privados entre integrantes.

- **Permitir la gestión de proyectos software**

La exploración de las seis (6) herramientas tecnológicas para la gerencia de proyectos, presente en el estado del arte (numeral 2.7.2), ha permitido deducir que estas herramientas están orientadas a dar soporte a proyectos en general, debido a que permiten definir actividades y establecer relación entre ellas (vista en el Diagrama de Gantt) pero las actividades no se pueden agrupar en fases e iteraciones, conformando un ciclo de vida del proceso de software que sea administrable. Solo dos (2) de ellas, permiten soportar parte de un ciclo de vida del proceso de software, al guiar al usuario en la definición de actividades dentro de fases. Sin embargo ninguna de las herramientas analizadas permite manejar distintos roles para la asignación de actividades. Teniendo en cuenta estas ausencias, la herramienta soporta la funcionalidad de gestionar el ciclo de vida del proceso de software que guía al proyecto que se desarrolle en comunidad. De esta manera, la herramienta permite la definición de proyectos que hacen uso de una plantilla de proceso de desarrollo de software. Para cada proyecto se puede gestionar tiempo, en cuanto a planificación de fases, iteraciones y actividades; gestionar personas, en cuanto a asignación de roles y tareas; gestionar productos, en cuanto a artefactos y versiones de artefactos, con la posibilidad de disponer de un repositorio de artefactos que pueden ser descargados por los miembros del proyecto, y gestionar el avance del proyecto, visualizando el estado de las actividades e iteraciones, para ofrecer una idea del progreso del proyecto.

3.1.1 Metodología utilizada para la construcción de la herramienta

La metodología de trabajo, para la construcción de la herramienta, está guiada por el Proceso Unificado de Desarrollo (UP) y se usa la notación establecida por el lenguaje UML (Unified Modeling Language).

Para el desarrollo de esta herramienta fueron necesarias cinco (5) iteraciones y dentro de cada iteración se encuentran los artefactos relacionados con las disciplinas de Requerimientos y Análisis, Diseño, Implementación y Pruebas.

La descripción detallada de las iteraciones se puede encontrar en el ANEXO B: Desarrollo de la herramienta en iteraciones.

3.1.2 Actores del Sistema

En el desarrollo del presente trabajo de grado se identificaron los siguientes actores, como esenciales para el funcionamiento y gestión de la herramienta, de un proyecto software y de los equipos de la comunidad:

1. **Usuario Anónimo:** persona que tiene la posibilidad de ver la información general pública (listado de proyectos activos en la herramienta) y la opción de registrarse en el sistema.
2. **Usuario Registrado:** usuario que ha seguido el proceso de registro en el sistema y ha recibido la notificación de aceptación, vía correo electrónico, como nuevo usuario del sistema. Puede ser un Usuario Normal ó un Administrador (usuario con todos los permisos).
    - **Usuario Normal:** usuario registrado en el sistema, que tiene la posibilidad de consultar los proyectos activos en la herramienta, asociarse a un proyecto, proponer la creación de un nuevo proyecto ó modificar su información personal.
    - **Administrador:** usuario encargado de la creación de proyectos y asignación de un director para cada proyecto. Posee todos los derechos con la capacidad de agregar o expulsar a otros. Incluye la funcionalidad del Usuario Normal.
3. **Usuario del Proyecto:** Usuario Registrado asociado a un proyecto como Integrante de un Equipo, Jefe de Equipo, ó Director del proyecto.
    - **Integrante de Equipo:** Tiene la posibilidad de consultar la información del proyecto, y ser parte activa del proyecto, como integrante de un equipo de trabajo, ya que se le pueden asignar roles, puede usar la pizarra de mensajes para los compañeros del proyecto, enviar un mensaje privado a un compañero ó usar el calendario de eventos del equipo.
    - **Jefe de Equipo:** es el usuario encargado de gestionar la asignación del rol, asignado al equipo en una iteración, a los trabajadores que hacen parte del equipo del cual es jefe. Además tiene la posibilidad de cambiar el estado de los trabajadores del equipo. Incluye la funcionalidad del Integrante de Equipo.

- **Director:** como director del proyecto debe gestionar el plan (plantilla de proceso) del proyecto, la temporalidad, la asignación de roles a los equipos de trabajo del proyecto por iteración dentro de una fase, la conformación de equipos de trabajo y la asignación de un jefe a cada equipo. Incluye la funcionalidad del Jefe de Equipo.

4. **Usuario con Rol:** Usuario del Proyecto a quien se le ha asignado un rol, dentro de una iteración de una fase del proyecto.
   - **Subordinado:** tiene la posibilidad de consultar y ejecutar las actividades que se la han asignado así como realizar y adicionar las versiones de artefactos (archivos) que se le han asignado, dentro de las actividades asignadas. También puede hacer uso de la pizarra de mensajes o el envío de mensajes privados a los compañeros de actividad y de rol.
   - **Jefe de Rol:** como Jefe de un rol dentro de un proyecto, debe gestionar la asignación de actividades, el estado de las actividades y la gestión de los artefactos que se deben generar en cada actividad. Incluye la funcionalidad del Subordinado.

En la Figura 12 se representa la relación entre los tipos de usuario definidos:

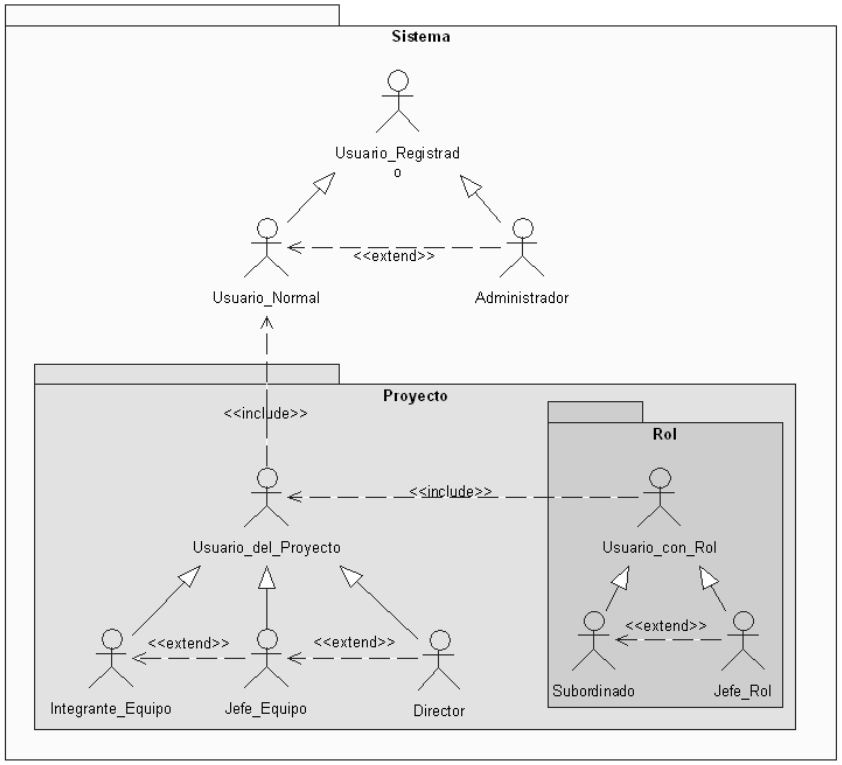


Figura 12: Relaciones entre tipos de usuarios

3.1.3 Arquitectura lógica

La arquitectura lógica de la herramienta se ha realizado teniendo en cuenta los objetivos planteados, por lo que por cada objetivo específico (ver numeral 1.1.3) se ha manejado un módulo de funcionalidad, teniendo en cuenta un tercer módulo de control para la herramienta. Es bueno aclarar que cuando se dice que la herramienta está compuesta por tres módulos, los tres se complementan entre si, por lo que ninguno debe ser visto como parte separada. A continuación se presenta la Figura 13 que describe la arquitectura lógica base de la herramienta y luego se describe la función de cada módulo. Cada módulo se encuentra enumerado, dando el orden en el cual serán explicados en los siguientes tres capítulos de este documento:

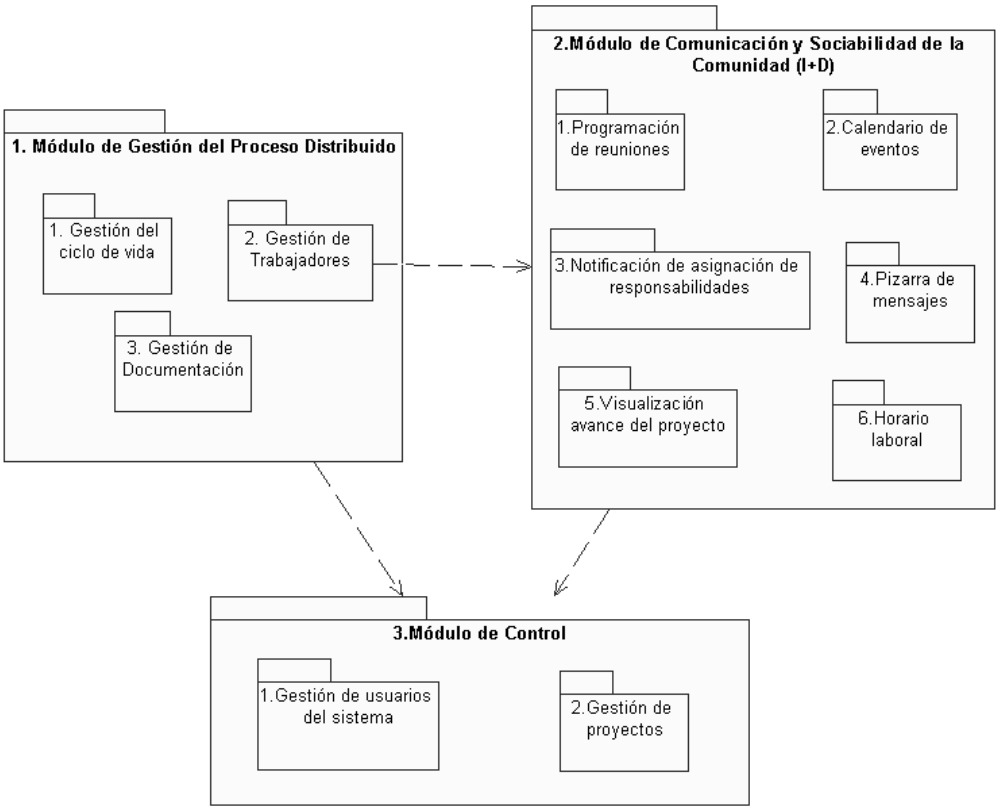


Figura 13: Arquitectura lógica de la herramienta

1. **Módulo de Gestión del Proceso distribuido:** su propósito consiste en apoyar las responsabilidades del proceso de software utilizado por el proyecto en comunidad.

2. **Módulo de Comunicación y Sociabilidad de la Comunidad (I+D):** su propósito consiste en soportar las necesidades de comunicación, coordinación y gestión de la disponibilidad para personas y equipo (I+D), bajo un ambiente de dispersión geográfica, que pertenecen a una comunidad de desarrollo, y que se encuentran ejecutando un proyecto de software.
3. **Módulo de Control:** su propósito consiste en soportar la gestión de usuarios de la herramienta, la adecuación, puesta en marcha y mantenimiento de los proyectos de software en comunidad.

3.1.4 Arquitectura de implementación

La herramienta desarrollada es una aplicación Web ya que este tipo de aplicación tiene las siguientes ventajas:

- **Compatibilidad multiplataforma.** Las aplicaciones basadas en Web no necesitan ser descargadas, instaladas y configuradas. Se puede acceder a través de un navegador Web sin importar cuál sea su configuración, su hardware o su sistema operativo.
- **Actualización.** La actualización de una aplicación Web se puede realizar sin la intervención de los usuarios de la aplicación. Es decir, no se necesita que el usuario se tome el trabajo de realizar descargas o procedimientos de instalación.
- **Menos requerimientos de hardware por parte del usuario.** Las aplicaciones basadas en Web no necesitan mayor capacidad de memoria RAM de parte del usuario final que los programas instalados localmente.
- **Precio.** Las aplicaciones Web no necesitan que el usuario para utilizarla necesite ningún tipo de software instalado en su maquina, aparte del navegador, las licencias que se requieren para que la aplicación funcione solo deben estar en el servidor donde se encuentra la aplicación. Por lo que los costos se ven notablemente reducidos.
- **Descentralización del trabajo.** Los usuarios no necesitan almacenar información en los computadores locales, ya que acceden a un repositorio de información central, de esta manera pueden acceder a la aplicación desde cualquier lugar.
- **Múltiples usuarios concurrentes.** Las aplicaciones basadas en web pueden ser utilizadas por múltiples usuarios al mismo tiempo.
- **Desarrollar aplicaciones en el lenguaje que usted quiera.** Las aplicaciones Web pueden ser separadas de computadores locales y sistemas operativos específicos, y también ser escritas en prácticamente cualquier lenguaje de programación. Mientras que para el software de escritorio se está limitado a usar el mismo lenguaje que el sistema operativo subyacente.

El diseño de una aplicación Web implica la toma de decisiones sobre su arquitectura física, así como sobre la tecnología e infraestructura que se emplearán para implementar su funcionalidad. Para tomar estas decisiones, se debe tener un conocimiento claro de los procesos empresariales que realizará la aplicación (sus requisitos funcionales), así como los niveles de escalabilidad, disponibilidad, seguridad y mantenimiento necesarios (sus requisitos no funcionales, funcionales u operativos).

Para el desarrollo del proyecto se concibió una arquitectura basada en tres capas la cual brinda grandes ventajas como:

- Se caracteriza por la descomposición de las aplicaciones
- Proporciona una escalabilidad, capacidad de administración y utilización de recursos mejorados
- Cada capa es un grupo de componentes que realiza una función específica y que se puede administrar o escalar de manera independiente.
- Cada capa interactúa sólo con las capas directamente debajo o encima y se puede ubicar en servidores físicamente diferentes. La comunicación entre las capas se realiza a través de protocolos estándar como HTTP o SOAP.
- Es posible cambiar o actualizar una capa sin volver a compilar o modificar otras capas.

A continuación se visualiza la arquitectura de implementación de la herramienta construida, la cual tiene cada capa enumerada para su correspondiente explicación.

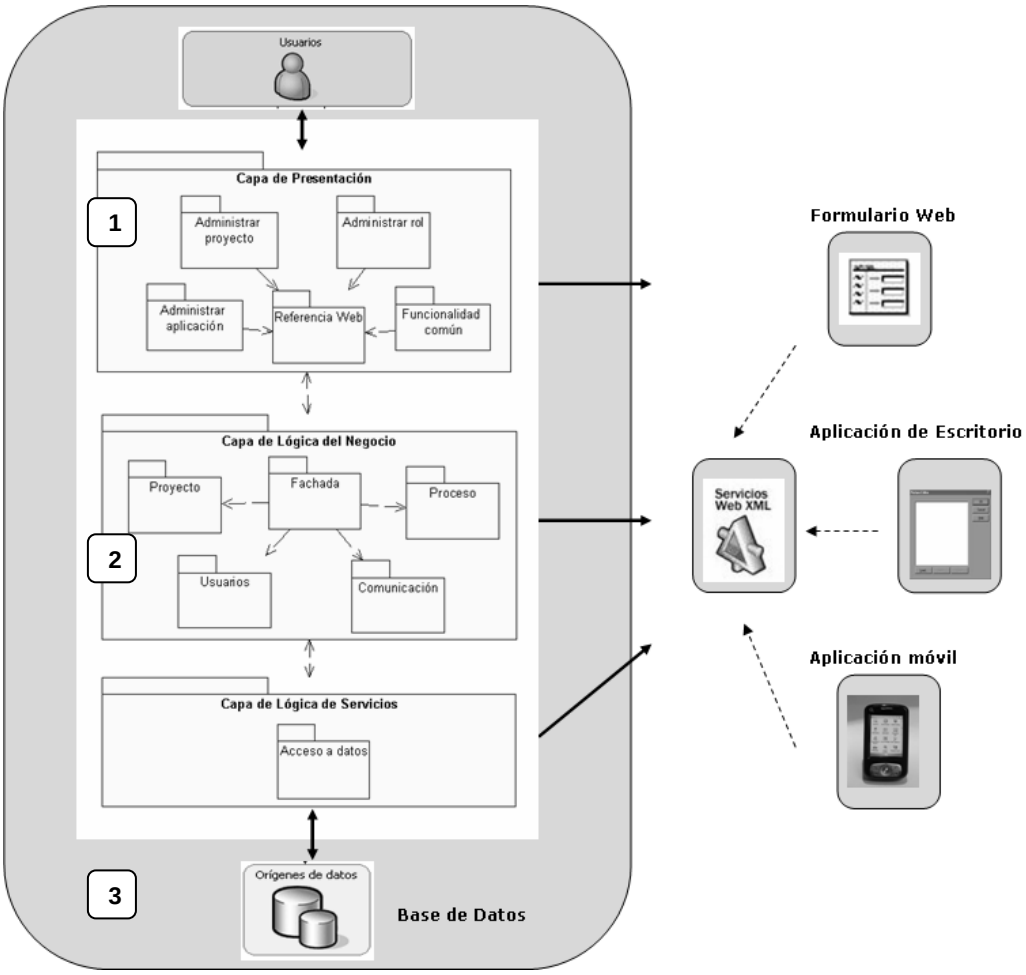


Figura 14: Diagrama de Paquetes de la arquitectura

Para el desarrollo del proyecto, se establecieron las siguientes capas, atendiendo al modelo propuesto por Microsoft (ver Figura 4: Arquitectura de tres capas basada en servicios):

1. Capa de Presentación

La aplicación debe permitir al usuario gestionar un proyecto, consultar información de las actividades, roles y artefactos, y asignarlos dependiendo del rol que tenga el usuario, escribir mensajes en la pizarra, entre otras funciones. En esta capa, se muestra información al usuario y se capturan los datos de los usuarios. La aplicación implementa esta capa mediante formularios Web. Se utilizan controles de validación que optimizan la tarea de garantizar que los datos escritos por el usuario se ajusten a la funcionalidad de la herramienta. Esta capa se comunica solamente con la capa de lógica de negocios.

Esta capa contiene las páginas que verá el usuario en el navegador y mediante las cuales podrá utilizar la funcionalidad que provee la aplicación. Además, dado que se debe establecer comunicación con la

capa de lógica de negocios, se tiene un paquete que contiene la referencia al servicio Web de Fachada perteneciente a la capa de lógica de negocios.

2. Capa de Lógica del Negocio

Esta capa se encarga de implementar las reglas del negocio que deben cumplirse. Esta capa se comunica con la capa de presentación, para recibir las solicitudes y presentar los resultados, y con la capa de lógica de servicios, para solicitar al gestor de base de datos almacenar o recuperar datos de él. La aplicación implementa esta capa mediante servicios Web. Se tiene un servicio Web por cada una de las clases de la lógica del negocio y se cuenta con un servicio Web que actúa de fachada para acceder al resto de servicios Web. La capa de presentación se comunica con esta capa, haciendo uso del servicio Web de Fachada. Cada servicio Web, de esta capa, se comunica con las clases de la capa de lógica de servicios.

3. Capa de Lógica de Servicios

Esta capa se encarga de interactuar con la base de datos para almacenar o recuperar datos. La aplicación implementa esta capa con un servicio Web que contiene los métodos necesarios para ejecutar instrucciones del Lenguaje de Manipulación de Datos (DML) y consultas en la base de datos. La capa de lógica de servicios, accede a la base de datos relacional a través de la tecnología ADO.NET de Microsoft. La base de datos se implementa con el motor de base de datos SQL Server 2000.

Ventajas de la arquitectura propuesta

a) Uso de Servicios Web

- Al usar servicios Web se permite la comunicación estructurada entre diferentes plataformas puesto que XML es un estándar definido por la W3E. Gracias a esto la arquitectura planteada permite que los servicios Web sean consumidos por cualquier tipo de aplicación sin importar la plataforma, por lo que se pueden consumir desde aplicaciones de escritorio, paginas Web (Asp, Php, Jsp, etc.), aplicaciones móviles, etc.
- La aplicación se hace escalable ya que al aparecer nuevos requerimientos, se crean nuevos servicios y la aplicación no se ve afectada.

b) Sentencias SQL Estándar

- Gracias a que en la capa de lógica de negocios se construyen sentencias SQL Estándar y en la capa de lógica de servicios se ejecutan esas sentencias es posible que el almacenamiento de datos se haga en cualquier motor de base de datos (SQLSERVER, POSTGRES, MYSQL, etc.).

- La migración a otra tecnología no es un problema puesto que como se usa SQL Estándar gran parte del código se puede reutilizar.

3.1.5 Implementación

En las siguientes figuras, se muestra la organización de la solución. La lógica de presentación se implementó mediante el proyecto ***PrototipoSedise***, donde se encuentran las páginas aspx (ver Figura 15: Organización del proyecto PrototipoSedise). La lógica de negocio se encuentra en el proyecto ***WSSedise*** y contiene los servicios Web necesarios para implementar la lógica de la aplicación (ver Figura 16: Organización del proyecto WSSedise). La lógica de servicios pertenece al mismo proyecto y principalmente contiene un servicio Web llamado WSBd.asmx, encargado de las operaciones relacionadas con la base de datos. El proyecto ***WSSedise*** también contiene un servicio Web que actúa como fachada para acceder a todos los servicios Web de la lógica de negocio y de la lógica de servicios.

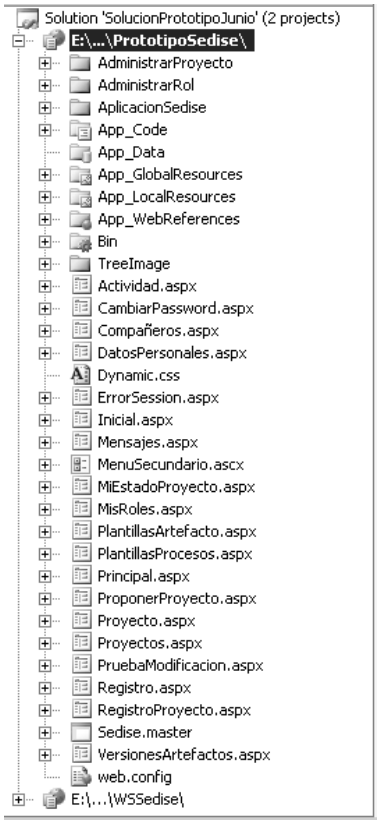


Figura 15: Organización del proyecto PrototipoSedise

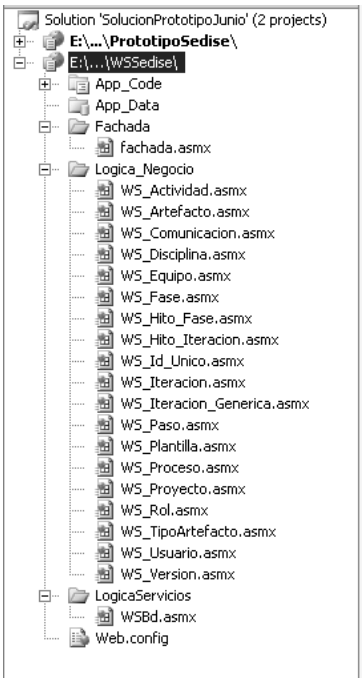


Figura 16: Organización del proyecto WSSedise

En los siguientes tres capítulos, del presente documento, se describe y explica la funcionalidad de cada uno de los tres módulos identificados en la arquitectura lógica de la herramienta (ver numeral 3.1.3). Se adicionan algunos artefactos obtenidos durante la construcción de la funcionalidad de los módulos.

CAPITULO 4: MÓDULO DE GESTIÓN DEL PROCESO DISTRIBUIDO

Tomando como referente la descripción de la herramienta realizada en el capitulo anterior, en la cual se describe la arquitectura lógica, se procede a explicar uno de los módulos de la arquitectura llamado Módulo de Gestión del Proceso Distribuido.

Este capitulo pretende explicar la solución obtenida para el primer objetivo especifico propuesto para la construcción de la herramienta "*Apoyar las responsabilidades del proceso, relacionadas con la gestión del ciclo de vida (fases, disciplinas e iteraciones), trabajadores (roles, responsabilidades y conformación de equipos de desarrollo) y documentación (modelos y artefactos), mediante una plataforma software*", (ver numeral 1.1.3).

Esta solución es el resultado de implementar la funcionalidad del Módulo de Gestión del Proceso Distribuido, el cual tiene como propósito apoyar las responsabilidades del proceso de software utilizado por el proyecto en comunidad.

El conjunto de artefactos software obtenidos durante la construcción de este módulo se citan a continuación.

Nombre artefacto	**Ubicación en documento de anexos (No. De Página)**
Especificación de requerimientos funcionales	24, 38, 54,
Lista de actores	25, 62
Lista de casos de uso	26, 40, 59,
Diagramas de casos de uso	27, 42, 43, 64, 65
Especificación de alto nivel de los casos de uso	30, 31, 44, 45, 46, 68
Especificación expandida	32, 33, 34, 47, 48
Diagrama conceptual	35, 50, 72
Diagramas de secuencia	37, 51, 104, 105, 106, 108, 109, 111, 112
Arquitectura lógica	52, 53
Diagramas de colaboración	77, 78, 95

Diagramas de clases	79, 80, 81, 107, 110, 113
Modelo de datos	82, 83, 84, 85, 86, 96, 97, 98
Casos de uso reales	74, 89, 90
Arquitectura de implementación	99, 100, 101, 102, 103
Descripción de implementación	116, 117
Vista física del sistema	114
Diagrama de despliegue	118
Pruebas	118, 119, 120, 121

Tabla 6: Conjunto de artefactos resultantes – Módulo de gestión del proceso distribuido

La descripción detallada y especificada de los artefactos software, organizados en iteraciones, se puede encontrar en el ANEXO B: Desarrollo de la herramienta en iteraciones.

A continuación se muestran los principales artefactos resultantes del proceso de construcción del módulo.

4.1 REQUISITOS DEL PROCESO[15] SOPORTADOS POR LA HERRAMIENTA

Los requisitos que debe satisfacer el proceso distribuido de desarrollo [7], y tomados como referentes para el desarrollo del Módulo de Gestión del Proceso Distribuido de la herramienta se citan a continuación, estos requisitos se pueden leer en detalle en el ANEXO A: Proceso Distribuido de Desarrollo.

Requisitos Funcionales:

- Permitir la planificación de proyectos.
- Permitir la asignación de tiempo a fases, iteraciones y actividades.
- Permitir la asignación de personal a roles, actividades y artefactos.
- Organización de los miembros del equipo en perfiles con diversidad de derechos de acceso.
- Organización, publicación y descarga de archivos.
- Control de versiones en archivos.
- Debe garantizar eficazmente la asignación y revisión del estado de actividades y sus artefactos (resultados de la actividad).

[15] Proceso distribuido de desarrollo concebido en la tesis de maestría [7]

Requisitos No Funcionales:

- Ser de fácil uso.
- Estar disponible para todos los miembros del equipo geográficamente disperso. Se recomienda Internet.

4.2 ARQUITECTURA DE LA HERRAMIENTA

En la Figura 17 se muestra la arquitectura lógica de la herramienta detallando el módulo de Gestión del Proceso Distribuido.

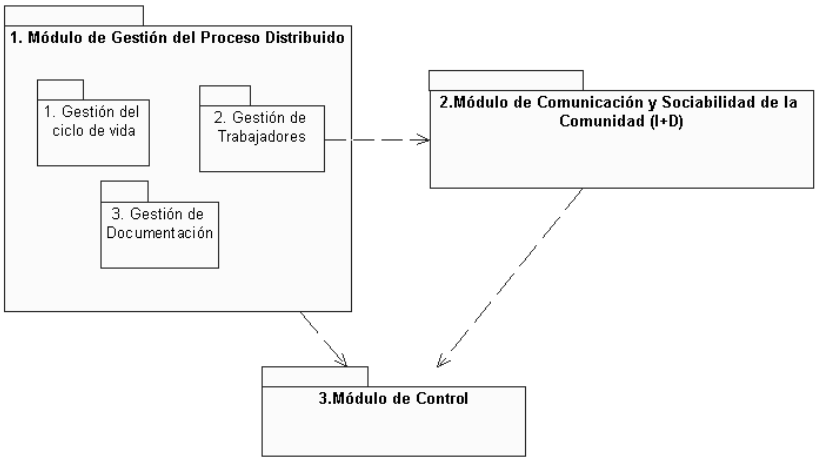


Figura 17: Módulo de Gestión del Proceso Distribuido

Este módulo se encuentra conformado a la vez por tres módulos, explicados así:

- **Módulo 1: Gestión del ciclo de vida.** Módulo encargado de la gestión del ciclo de vida del proceso instanciado por un proyecto. El ciclo de vida incluye fases, disciplinas e iteraciones. Este módulo intenta dar soporte a la siguiente sección del objetivo específico "*Apoyar las responsabilidades del proceso, relacionadas con la gestión del ciclo de vida (fases, disciplinas e iteraciones)"* (ver numeral 1.1.3) e intenta satisfacer el siguiente requisito del proceso distribuido de desarrollo:

    - Permitir la asignación de tiempo a fases, iteraciones y actividades.

- **Módulo 2: Gestión de trabajadores**. Módulo encargado de gestionar la asignación de roles, gestionar la asignación y estado de responsabilidades (actividades y artefactos) y gestionar la conformación de equipos de trabajo, de un proyecto. Incluye la modificación del estado de actividades y artefactos. Este módulo intenta dar soporte a la siguiente sección del objetivo específico "*Apoyar las responsabilidades del proceso, relacionadas con la gestión de trabajadores (roles, responsabilidades y conformación de equipos de desarrollo)"* (ver numeral 1.1.3) e intenta satisfacer los siguientes requisitos del proceso distribuido de desarrollo:

    - Permitir la asignación de personal a roles, actividades y artefactos.
    - Organización de los miembros del equipo en perfiles con diversidad de derechos de acceso.
    - Debe garantizar eficazmente la asignación y revisión del estado de actividades y sus artefactos (resultados de la actividad).

- **Módulo 3: Gestión de Documentación (artefactos).** Módulo encargado de gestionar la organización, publicación y control de versiones de los artefactos generados y manipulados en cada actividad, dentro de un proyecto. Este módulo intenta dar soporte a la siguiente sección del objetivo específico "*Apoyar las responsabilidades del proceso, relacionadas con la gestión de la documentación"* (ver numeral 1.1.3) e intenta satisfacer los siguientes requisitos del proceso distribuido de desarrollo:

    - Organización, publicación y descarga de archivos.
    - Control de versiones en archivos.

4.3 REQUERIMIENTOS FUNCIONALES DEFINIDOS PARA LA HERRAMIENTA

Para la construcción del módulo de Gestión del Proceso Distribuido se tomaron como requerimientos funcionales los siguientes:

Módulo de Gestión del ciclo de vida:

- **Gestionar el ciclo de vida del proceso.**
  Poder seleccionar el proceso de desarrollo y personalizar su ciclo de vida de acuerdo a las necesidades y características del proyecto. La personalización incluye seleccionar fases, crear iteraciones, seleccionar actividades y definir hitos.

- **Gestionar la temporalidad del proyecto.**
  Poder establecer aspectos de temporalidad (fecha de inicio y fecha de finalización) de los elementos básicos del proyecto como son las fases, las iteraciones y las actividades.

Módulo de Trabajadores:

- **Inscribirse en un nuevo proyecto.**
  Poder vincularse a un proyecto, e indicar el equipo dentro del proyecto al cual se incorpora, pero antes debe recibir la aprobación como nuevo miembro del proyecto, para ser parte activa en él.
- **Conocer los roles asignados dentro del proyecto.**
  Poder conocer los roles que tiene asignados en las iteraciones de cada fase del proyecto. En este caso puede aceptar o rechazar un rol asignado. En una iteración puede tener varios roles asignados.
- **Gestionar trabajadores del equipo.**
  Poder asignar ó eliminar la asignación de roles a los trabajadores de un equipo de trabajo. También poder cambiar el estado de un trabajador, del equipo, dentro del proyecto (posibles estados: activo, inactivo), siempre y cuando no tengan asignados ningún rol.
- **Gestionar equipos de trabajo del proyecto.**
  Poder realizar acciones de gestión con los equipos de trabajo del proyecto, como crear, modificar, eliminar y consultar equipos, así como asignar al jefe del equipo y asignar roles al equipo.
- **Conocer las actividades y artefactos asignados.**
  Poder conocer las actividades y artefactos, dentro de un rol asignado en una iteración, perteneciente a una fase del proyecto. En este caso puede aceptar o rechazar una actividad. Al aceptar una actividad, acepta por defecto los artefactos de esa actividad que le sean asignados. En una iteración, el usuario puede tener varias actividades asignadas. En una actividad, el usuario puede tener varios artefactos asignados.
- **Gestionar actividades.**
  Poder asignar o eliminar la asignación de una actividad a un trabajador con el mismo rol en la misma iteración. También poder modificar el estado de una actividad para darla por terminada teniendo en cuenta que todos los artefactos de la actividad hayan sido aprobados.
- **Gestionar artefactos de una actividad.**
  Poder definir los artefactos a generar en una actividad, así como modificar, consultar, eliminar y asignar ó eliminar la asignación de estos artefactos a los trabajadores que tienen asignada la actividad.

Módulo de Gestión de Documentación:

- **Gestionar versiones (archivos) de los artefactos asignados.**
  Poder adjuntar archivos ó versiones de los artefactos asignados para que sean publicados, indicando el nivel de avance de la versión.
- **Descargar versiones de artefactos.**
  Poder descargar los archivos de versiones de los artefactos generados en el proyecto, ya que son documentación libre para los miembros del proyecto.

4.4 LISTA DE CASOS DE USO

En la siguiente tabla se observa la relación de cada requerimiento con los casos de uso que se deducen:

Requerimiento	Caso de Uso
Inscribirse en un nuevo proyecto	1. Inscribirse en proyecto
Conocer los roles asignados dentro del proyecto	2. Consultar roles asignados
	2.1 Atender asignación rol
	2.1.1 Aceptar rol
	2.1.2 Rechazar rol
Gestionar el ciclo de vida del proceso	3. Gestionar plan de proyecto
	3.1 Seleccionar plantilla
	3.2 Personalizar plan
	3.3 Modificar plan
	3.4 Consultar plan
	3.5 Gestionar hito fase
Gestionar la temporalidad del proyecto	4. Gestionar temporalidad de proyecto
	4.1 Temporalizar fases
	4.2 Temporalizar iteraciones
	4.3 Temporalizar actividades
Gestionar trabajadores del equipo	5. Gestionar trabajadores
	5.1 Cambiar estado trabajadores del proyecto
	5.2 Gestionar asignación de roles a trabajador
Gestionar equipos de trabajo del proyecto	6. Gestionar equipos
	6.1 Crear equipo
	6.2 Modificar equipo
	6.3 Eliminar equipo
	6.4 Gestionar asignación Jefe de Equipo

	6.5 Gestionar asignación roles al equipo
Conocer las actividades y artefactos asignados	7. Consultar actividades asignadas
	7.1 Atender asignación de actividad
	7.1.1 Aceptar actividad
	7.1.2 Rechazar actividad
	8. Consultar artefactos asignados
Gestionar versiones (archivos) de los artefactos asignados	9. Gestionar versiones de artefactos
Descargar versiones de artefactos	10. Descargar versiones
Gestionar actividades	11. Gestionar actividad
	11.1 Gestionar asignación actividad
	11.2 Modificar estado de actividad
Gestionar artefactos de una actividad	12. Gestionar artefactos de actividad
	12.1 Definir artefactos
	12.2 Modificar artefactos propios
	12.3 Consultar artefactos
	12.4 Eliminar artefactos
	12.5 Gestionar asignación artefacto

Tabla 7: Requerimientos y casos de uso módulo de gestión del proceso

4.5 DIAGRAMAS DE CASOS DE USO

Los diagramas de casos de uso se realizan teniendo en cuenta la lista de casos de uso obtenida (ver Tabla 4: Requerimientos y casos de uso módulo de gestión del proceso) y la clasificación de los actores del sistema.

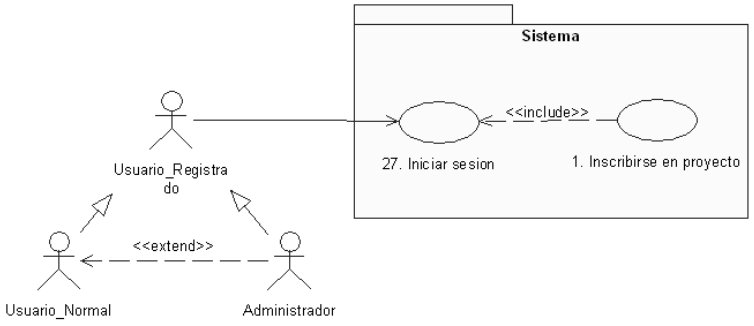


Figura 18: Casos de uso de Usuario Registrado-Parte 1

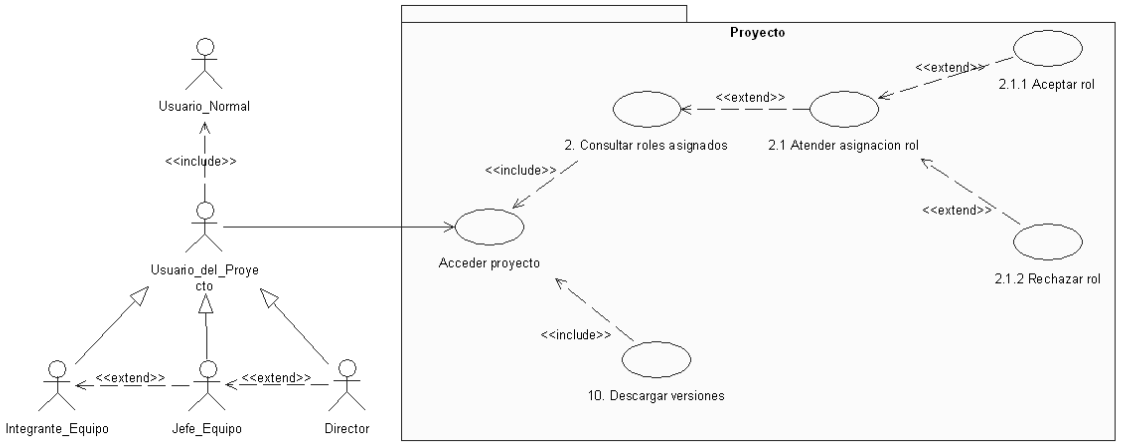


Figura 19: Casos de uso de Usuario del Proyecto-Parte 1

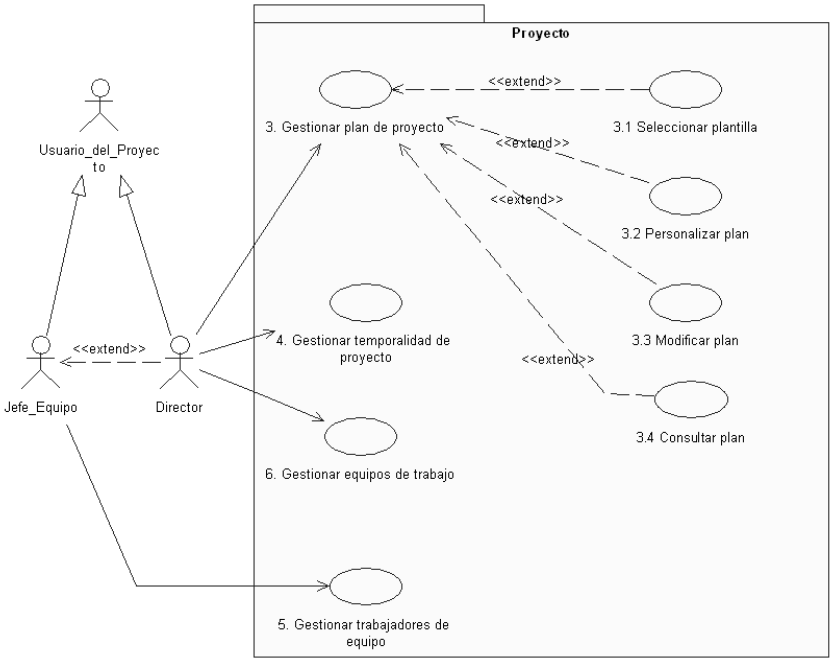


Figura 20: Casos de uso del Director

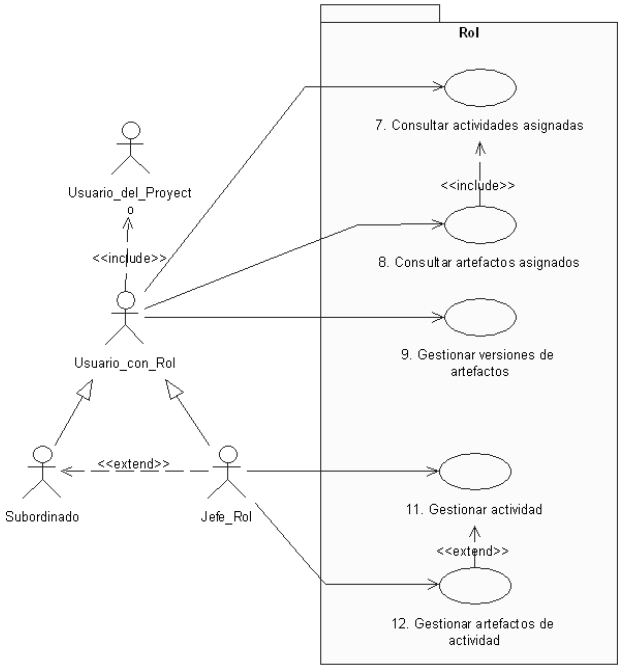


Figura 21: Casos de uso del Usuario con Rol-Parte 1

4.6 MODELO DE BASE DE DATOS

A continuación se presenta el modelo relacional de la base de datos obtenido en el proyecto, relacionado con el módulo en cuestión, dividido en vistas para mejor comprensión.

Modelo Relacional Vista 1
Los conceptos involucrados dentro de este modelo son: • **Proyecto:** representa un proyecto de desarrollo de software. • **Proceso:** representa un proceso de desarrollo de software. • **Fase:** representa una etapa del proceso. El proceso de desarrollo de software se divide en fases. • **Disciplina:** representa una colección de actividades relacionadas. • **Rol:** representa el papel que un individuo puede desempeñar en el desarrollo de software. • **Iteración:** representa un conjunto de actividades llevadas a cabo de acuerdo a un plan que lleva a producir una versión de un producto. Cada fase del proyecto está constituido por una o más iteraciones. • **Actividad:** representa la ejecución de una operación por un rol. • **TipoArtefacto:** representa un tipo de artefacto software, es decir, ó tipo de producto obtenido en el desarrollo de software, del cual pueden resultar muchas versiones. Estos conceptos se transforman en entidades, a las cuales se les asigna un identificador único o clave primaria, identificadas con el atributo <PI>, el tipo de dato y el tamaño, y su obligatoriedad o ausencia de ella. Las relaciones entre las entidades se establecen así: • Un proyecto instancia un proceso. Un proceso puede ser instanciado por cero o muchos proyectos. • Un proceso está conformado por una o muchas fases. Una fase pertenece a un solo proceso. • Una fase contiene cero o muchas iteraciones. Una iteración pertenece a una fase. • Un proceso tiene una o muchas disciplinas. Una disciplina pertenece a un proceso.

La asociación Fases_del_Proyecto permite asociar una fase a un proyecto con la información de fecha inicial, fecha final y estado de la fase dentro del proyecto.

La asociación Roles_del_Proceso permite conocer los roles relacionados con un proceso. Un rol se puede asociar con uno o muchos procesos. Un proceso puede tener uno o muchos roles.

La asociación Actividades_de_Disciplina permite conocer las actividades que hacen parte de una disciplina.

La asociación Rol_de_Actividad permite conocer el rol encargado de ejecutar una actividad. Un rol puede ejecutar cero o muchas actividades. Una actividad es ejecutada por un rol.

La asociación Actividades_de_Iteración permite conocer la información de una actividad que se ejecuta dentro de una iteración. Muestra la fecha inicial, la fecha final y el estado de la actividad. Una actividad se puede ejecutar en cero o muchas iteraciones. Una iteración ejecuta cero o muchas actividades.

La asociación TipoArtefacto_de_Actividad permite conocer los tipos de artefactos a ser generados en la actividad. Una actividad puede generar cero o muchos tipos de artefactos. Un tipo de artefacto puede ser generado por una o muchas actividades.

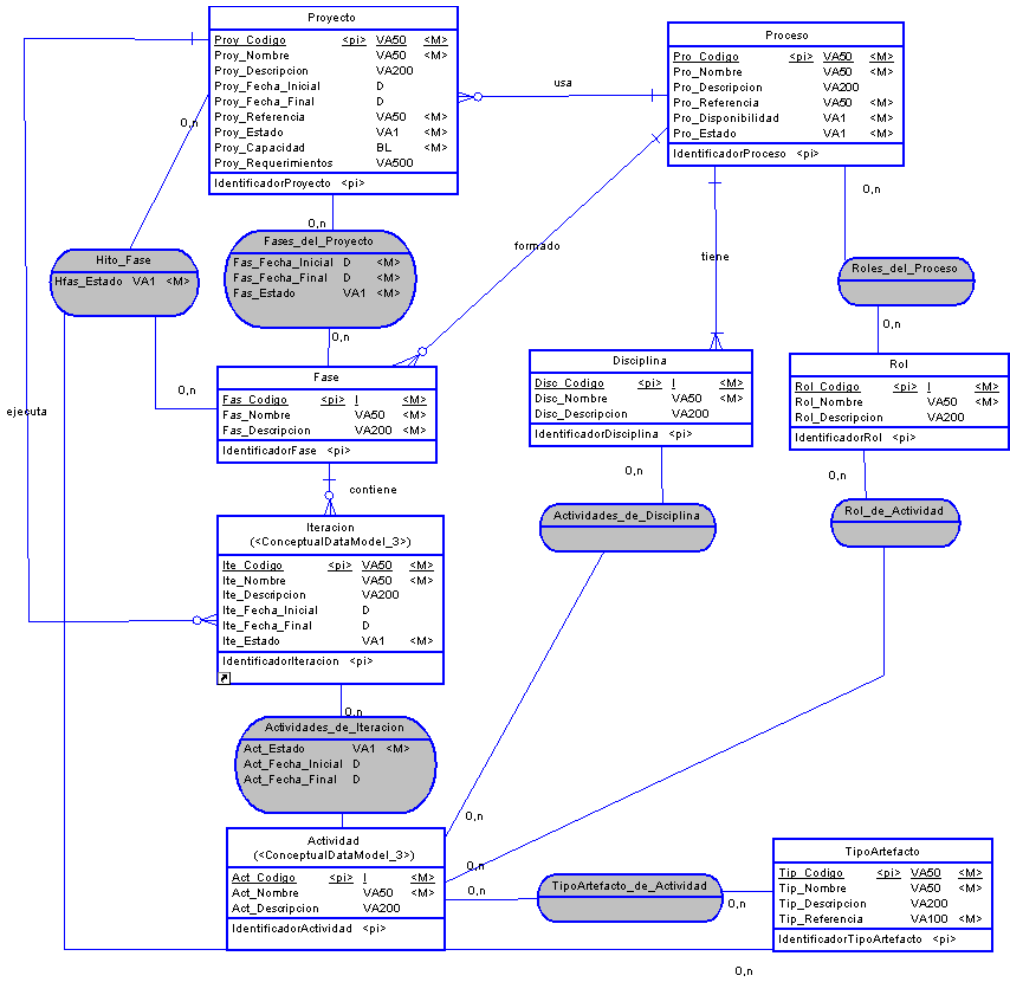


Tabla 8: Modelo relacional vista 1

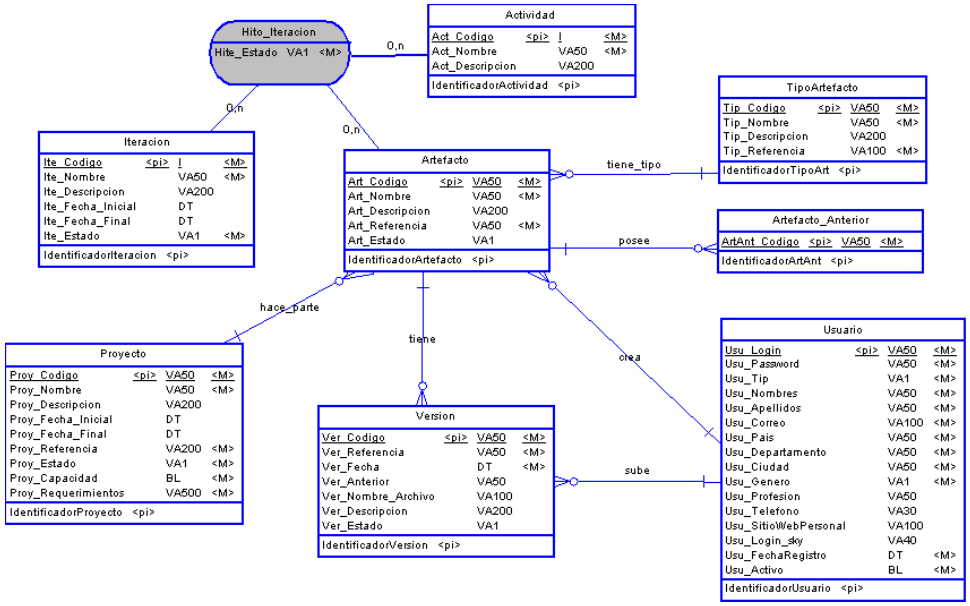
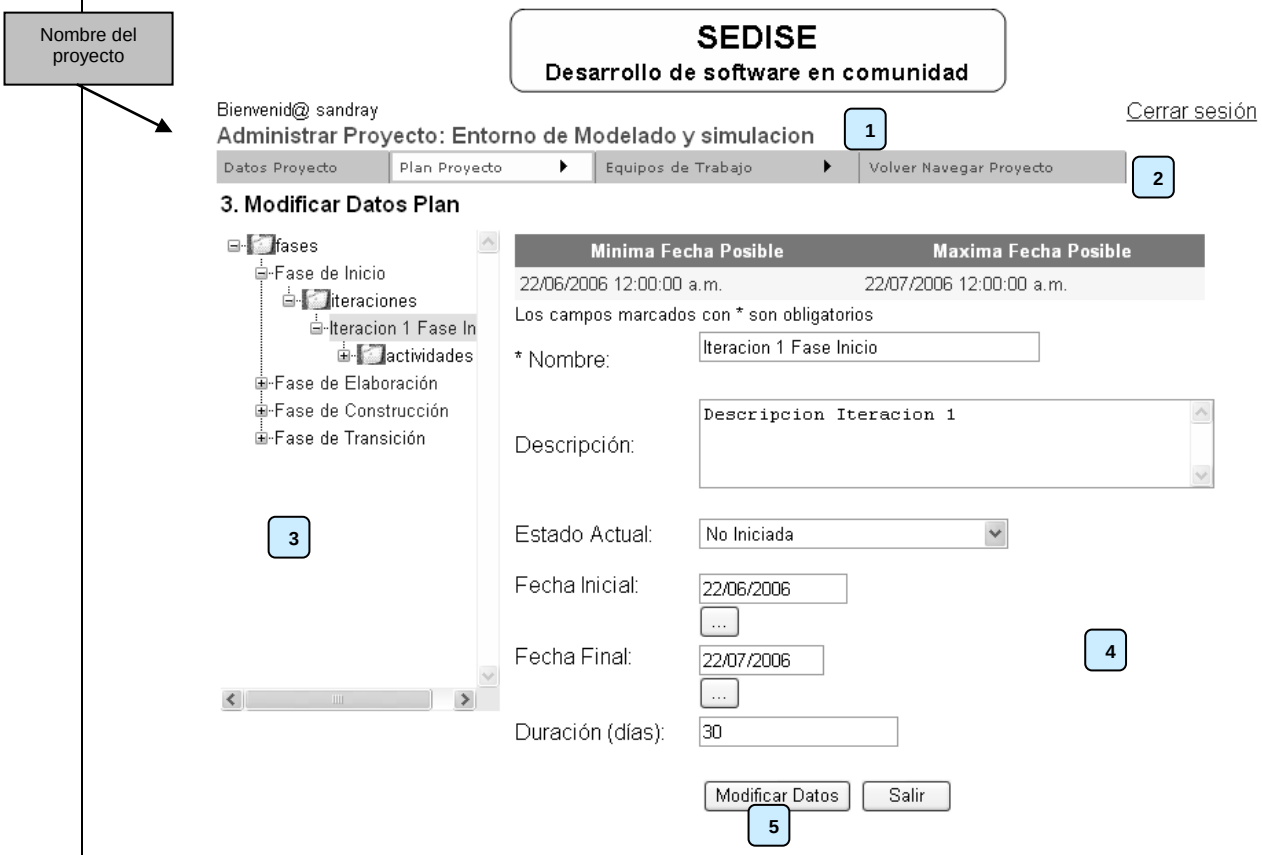
Modelo Relacional Vista 2
Los nuevos conceptos involucrados dentro de este modelo son: • **Artefacto:** representa la instancia de un tipo de artefacto, que es asignado a un usuario para generar versiones del mismo. • **Artefacto_Anterior:** representa el artefacto de entrada para generar otro artefacto. • **Usuario:** representa a la persona registrada en el sistema. • **Version:** representa la actualización de un artefacto. Estos conceptos se transforman en entidades, a las cuales se les asigna un identificador único o clave primaria, identificadas con el atributo <PI>, el tipo de dato y el tamaño, y su obligatoriedad o ausencia de ella. Las relaciones entre las entidades se establecen así: • Un artefacto está basado en un tipo de artefacto. De un tipo de artefacto se pueden generar cero o muchos artefactos. • Un artefacto puede poseer cero o muchos artefactos anteriores. • Un artefacto es creado por un usuario. Un usuario puede crear cero o muchos artefactos. • Un artefacto tiene cero o muchas versiones. Una versión pertenece a un solo artefacto. • Un usuario sube cero o muchas versiones. Una versión es subida por un usuario. • Un artefacto hace parte de un proyecto. Un proyecto puede tener cero o muchos artefactos. La asociación Hito_Iteracion permite determinar que artefactos se están generando en una actividad dentro de una iteración y el estado del artefacto.
	

Tabla 9: Modelo relacional vista 2

4.7 CASOS DE USO REALES

Los casos de uso reales muestran la interacción del usuario con la herramienta construida, involucrando elementos de interfaz. A continuación se muestra un ejemplo de un caso de uso real obtenido para el módulo en cuestión.

CASO DE USO REAL	Modificar plan de proyecto
ACTOR	Director del proyecto
DESCRIPCION	Este caso de uso comienza cuando el usuario decide gestionar el plan del proyecto, modificando la temporalidad de fases o iteraciones o actividades. Este caso de uso finaliza cuando el usuario logra realizar la función de gestión.



CURSO NORMAL DE LOS EVENTOS	
Acción del Actor	**Respuesta del Sistema**
1. El usuario da clic en el control [1] "Administrar proyecto".	**2.** El sistema carga el menú de funciones para administrar un proyecto [2].
3. El usuario selecciona la opción "Modificar datos plan" del menú "Gestionar plan de proyecto" [2].	**4.** El sistema carga la información del proyecto organizada en fases, iteraciones y actividades, en el árbol de navegación [3].
5. El usuario selecciona un nodo del árbol de navegación para ver el detalle [3].	**6.** El sistema carga la información del nodo [4].
7. El usuario procede a realizar las modificaciones deseadas en los campos modificables [4].	
8. El usuario da clic en el botón "Modificar Datos" para confirmar la modificación [5]	**9.** El sistema almacena los cambios realizados por el usuario.

Tabla 10: Caso de uso real Modificar datos del plan de proyecto

En el siguiente capítulo, del presente documento, se describe y explica la funcionalidad de uno de los módulos lógicos de la herramienta construida, llamado “Módulo de Comunicación y Sociabilidad de la Comunidad (I+D)”. Se adicionan algunos artefactos obtenidos durante la construcción de la funcionalidad del módulo.

CAPITULO 5: MÓDULO DE COMUNICACIÓN Y SOCIABILIDAD DE LA COMUNIDAD (I+D)

Tomando como referente la descripción de la herramienta realizada en el capitulo 3, en la cual se describe la arquitectura lógica, se procede a explicar uno de los módulos de la arquitectura llamado Módulo de Comunicación y Sociabilidad de la Comunidad (I+D).

Este capitulo pretende explicar la solución obtenida para el segundo objetivo especifico propuesto para la construcción de la herramienta "*Apoyar las actividades del proceso relacionadas con la comunicación y sociabilidad*[16] *de los equipos de desarrollo distribuido (programación de reuniones, asignación, seguimiento y entrega de tareas, publicación de calendarios e informes del avance y estado del proyecto entre otros) mediante una plataforma software*", (ver numeral 1.1.3).

Esta solución es el resultado de implementar la funcionalidad del Módulo de Comunicación y Sociabilidad de la Comunidad (I+D), el cual tiene como propósito soportar las necesidades de comunicación, coordinación y gestión de la disponibilidad para personas y equipos (I+D), bajo un ambiente de dispersión geográfica, que pertenecen a la comunidad de desarrollo, y que se encuentran ejecutando un proyecto de software.

El conjunto de artefactos software obtenidos durante la construcción de este módulo se citan a continuación.

Nombre artefacto	**Ubicación en documento de anexos (No. De Página)**
Especificación de requerimientos funcionales	25, 38,56
Lista de actores	25, 62,
Lista de casos de uso	26,41,60
Diagramas de casos de uso	27,66,67

[16] Modelo de normas o protocolos de comportamiento en una comunidad.

Especificación de alto nivel de los casos de uso	30,45,69,70,71
Diagrama conceptual	35,50,72
Diagramas de secuencia	73
Arquitectura lógica	52, 55
Diagramas de clases	79,80,81
Modelo de datos	85,86,87,88,89,98
Casos de uso reales	90,91,92,93
Arquitectura de implementación	99,100,101,102,103
Descripción de implementación	116,117
Vista física del sistema	114
Diagrama de despliegue	118
Pruebas	118,119,120,121

Tabla 11: Conjunto de artefactos resultantes – Módulo de comunicación y sociabilidad de la comunidad (i+d)

La descripción detallada y especificada de los artefactos software, organizados en iteraciones, se puede encontrar en el ANEXO B: Desarrollo de la herramienta en iteraciones.

A continuación se muestran los principales artefactos resultantes del proceso de construcción del módulo.

5.1 REQUISITOS DEL PROCESO[17] SOPORTADOS POR LA HERRAMIENTA

Los requisitos que debe satisfacer el proceso distribuido de desarrollo, resultado de la tesis de maestría [7], y tomados como referentes para el desarrollo del Módulo de Comunicación y Sociabilidad de la Comunidad (I+D) de la herramienta se citan a continuación, estos requisitos se pueden leer en detalle en el ANEXO A: Proceso Distribuido de Desarrollo. .

Requisitos Funcionales:

- Concebir reuniones presenciales y virtuales.
- Concebir mecanismos para la publicación de calendarios, con el propósito de coordinar todos los equipos distribuidos geográficamente.
- Soportar mensajería a través de e-mail.

[17] Proceso distribuido de desarrollo concebido en la tesis de maestría [7].

- Sugerir mecanismos eficaces para controlar el avance del proyecto.
- Sugerir estrategias para establecer un clima de pertenencia, confianza y solidaridad dentro de la comunidad de desarrollo.

5.2 ARQUITECTURA DE LA HERRAMIENTA

En la Figura 22 se muestra la arquitectura lógica de la herramienta detallando el módulo de Comunicación y Sociabilidad de la comunidad (I+D).

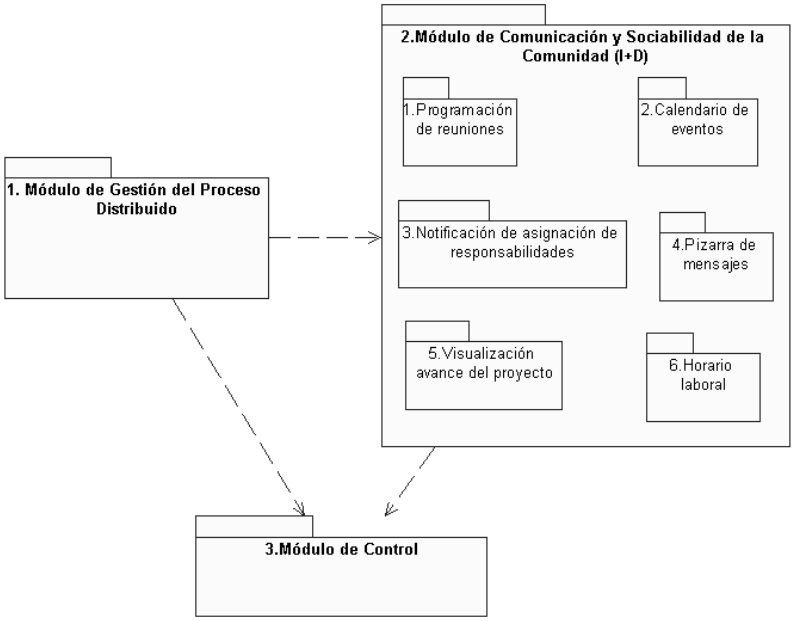


Figura 22: Módulo de Comunicación y Sociabilidad de la Comunidad (I+D)

Este módulo se encuentra conformado a la vez por cinco módulos, explicados así:

- **Módulo 1: Programación de reuniones.** Módulo encargado de permitir la programación de reuniones (presenciales o virtuales) que harán parte de la agenda personal de reuniones de un usuario de un proyecto. Estas reuniones han sido planeadas previamente usando otros medios de comunicación como la pizarra de mensajes o el correo electrónico. Este módulo intenta dar soporte a la siguiente sección del objetivo específico "*Apoyar las actividades del proceso relacionadas con la*

comunicación y sociabilidad[18] *de los equipos de desarrollo distribuido (programación de reuniones)"* (ver numeral 1.1.3) e intenta satisfacer el siguiente requisito del proceso distribuido de desarrollo:

- Concebir reuniones presenciales y virtuales.

- **Módulo 2: Publicación de calendarios.** Módulo encargado de permitir el registro de eventos planeados por un equipo de trabajo, en un calendario, para ser publicado y conocido por los miembros del equipo, dentro de un proyecto. Este módulo intenta dar soporte a la siguiente sección del objetivo específico "*Apoyar las actividades del proceso relacionadas con la comunicación y sociabilidad de los equipos de desarrollo distribuido (publicación de calendarios)"* (ver numeral 1.1.3) e intenta satisfacer el siguiente requisito del proceso distribuido de desarrollo:

  - Concebir mecanismos para la publicación de calendarios, con el propósito de coordinar todos los equipos distribuidos geográficamente.

- **Módulo 3: Mensajes de notificación de asignación de responsabilidades.** Módulo encargado de enviar al usuario un mensaje, vía correo electrónico, cada vez que se le asigne un rol, una actividad ó un artefacto, como mecanismo de notificación de asignación, dentro de un proyecto. Este módulo intenta dar soporte a la siguiente sección del objetivo específico "*Apoyar las actividades del proceso relacionadas con la comunicación y sociabilidad de los equipos de desarrollo distribuido (asignación, seguimiento y entrega de tareas)"* (ver numeral 1.1.3) e intenta satisfacer el siguiente requisito del proceso distribuido de desarrollo:

  - Soportar mensajería a través de e-mail.

- **Módulo 4: Pizarra de mensajes.** Módulo encargado de poner a disposición de los miembros del proyecto un mecanismo por el cual cada usuario puede enviar mensajes públicos a los compañeros de proyecto, compañeros de actividad y compañeros de rol ó enviar un mensaje privado a un compañero. Este módulo intenta satisfacer el siguiente requisito del proceso distribuido de desarrollo:

  - Sugerir estrategias para establecer un clima de pertenencia, confianza y solidaridad dentro de la comunidad de desarrollo.

[18] Modelo de normas o protocolos de comportamiento en una comunidad.

- **Módulo 5: Visualización avance del proyecto.** Módulo encargado de permitir a un usuario del proyecto, visualizar el estado y avance del proyecto. Este módulo intenta dar soporte a la siguiente sección del objetivo específico "*Apoyar las actividades del proceso relacionadas con la comunicación y sociabilidad de los equipos de desarrollo distribuido (informes del avance y estado del proyecto)"* (ver numeral 1.1.3) e intenta satisfacer el siguiente requisito del proceso distribuido de desarrollo:

    - Sugerir mecanismos eficaces para controlar el avance del proyecto.

- **Módulo 6: Horario laboral.** Módulo encargado de solicitar el horario de disponibilidad laboral de un miembro del proyecto y publicarlo para que sea conocido por los demás miembros del proyecto en la comunidad. Este módulo intenta dar soporte e intenta satisfacer el siguiente requisito del proceso distribuido de desarrollo:

    - Concebir mecanismos con el propósito de facilitar la coordinación entre miembros.

5.3 REQUERIMIENTOS FUNCIONALES DEFINIDOS PARA LA HERRAMIENTA

Para la construcción del módulo de Comunicación y Sociabilidad de la comunidad (I+D) se tomaron como requerimientos funcionales los siguientes:

Módulo de programación de reuniones:

- **Programar reuniones.**
  Poder programar reuniones en la agenda personal de reuniones del usuario, dentro de un proyecto.

Módulo de publicación de calendarios:

- **Programar eventos en calendario.**
  Poder realizar la programación de eventos entre miembros del equipo de trabajo dentro de un proyecto y publicar esta información para el resto de miembros del equipo.

- **Gestionar calendario de eventos**
  Poder eliminar eventos del calendario, que maneja el equipo de trabajo, para quitar eventos que ya se hayan realizado o que se desean borrar.

Módulo de mensajes de notificación de asignación de responsabilidades:

- **Recibir correo de notificación de la asignación de un rol, de una actividad o de un artefacto.**
  Poder recibir un correo electrónico anunciando la asignación de un rol, una actividad o de un artefacto, dentro de una iteración, como mecanismo de notificación de mensajes.

Módulo de la pizarra de mensajes:

- **Utilizar la pizarra de mensajes con los compañeros del proyecto.**
  Poder hacer uso de un mecanismo de comunicación y colaboración entre compañeros del proyecto, que permite publicar un mensaje disponible para todos los compañeros del proyecto.
- **Enviar un mensaje privado a un compañero del proyecto.**
  Poder hacer uso de un mecanismo de comunicación y colaboración con un compañero específico del proyecto, que permite enviarle un mensaje para ser observado por el receptor en el correo electrónico ó en la plataforma (sección de “Mis mensajes”).
- **Utilizar la pizarra de mensajes con los compañeros del rol.**
  Poder hacer uso de un mecanismo de comunicación y colaboración entre compañeros del mismo rol, que permite publicar un mensaje disponible para todos los compañeros del rol, dentro de una iteración.
- **Utilizar la pizarra de mensajes con los compañeros de la actividad.**
  Poder hacer uso de un mecanismo de comunicación y colaboración entre compañeros de la actividad, que permite publicar un mensaje disponible para todos los compañeros de la actividad, dentro de una iteración.
- **Enviar un mensaje privado a un compañero del rol.**
  Poder hacer uso de un mecanismo de comunicación y colaboración con un compañero específico del mismo rol, que permite enviarle un mensaje para ser observado por el receptor en el correo electrónico ó en la plataforma (sección de “Mis mensajes”). ”
- **Enviar un mensaje privado a un compañero de la actividad.**
  Poder hacer uso de un mecanismo de comunicación y colaboración con un compañero específico de la actividad, que permite enviarle un mensaje para ser observado por el receptor en el correo electrónico ó en la plataforma (sección de “Mis mensajes”).

Módulo de visualización avance del proyecto:

- **Obtener avance del proyecto.**
  Poder visualizar un informe del estado y avance del proyecto.

Módulo de horario laboral:

- **Definir el horario de disponibilidad laboral.**
  Poder registrar en un horario los días y las horas disponibles para la dedicación al proyecto de tal manera que pueda ser consultado por los demás integrantes del proyecto.

Otro requerimiento:

- **Consultar la información personal de los compañeros del proyecto.**
  Poder consultar la información personal de los compañeros del equipo de trabajo así como del resto de compañeros del proyecto, como mecanismo de socialización con las personas que trabaja.

5.4 LISTA DE CASOS DE USO

En la siguiente tabla se observa la relación de cada requerimiento con los casos de uso que se deducen:

Requerimiento	Caso de Uso
Recibir correo de notificación de la asignación de un rol, de una actividad o de un artefacto	13. Recibir mensajes de notificación
Recibir correo de notificación de asignación como Director de proyecto	
Definir el horario de disponibilidad laboral	14. Definir horario laboral
Consultar la información personal de los compañeros del proyecto	15. Consultar información de compañeros
Obtener avance del proyecto	16. Obtener avance del proyecto
Utilizar la pizarra de mensajes con los compañeros del proyecto	17. Usar pizarra de mensajes del proyecto
Enviar un mensaje privado a un compañero del proyecto	18. Enviar mensaje privado de proyecto
Utilizar la pizarra de mensajes con los compañeros del rol o de la actividad	19. Usar pizarra de mensajes de rol o de actividad
Enviar un mensaje privado a un compañero del rol o de la actividad	20. Enviar mensaje privado de rol o actividad
Programar reuniones	21. Programar reuniones
Gestionar calendario de eventos	22. Gestionar calendario eventos
Programar eventos en calendario	23. Programar eventos calendario equipo

Tabla 12: Requerimientos y casos de uso módulo de comunicación y sociabilidad de la comunidad (i+d)

5.5 DIAGRAMAS DE CASOS DE USO

Los diagramas de casos de uso se realizan teniendo en cuenta la lista de casos de uso obtenida (ver Tabla 9: Requerimientos y casos de uso módulo de comunicación y sociabilidad de la comunidad (i+d)) y la clasificación de los actores del sistema.

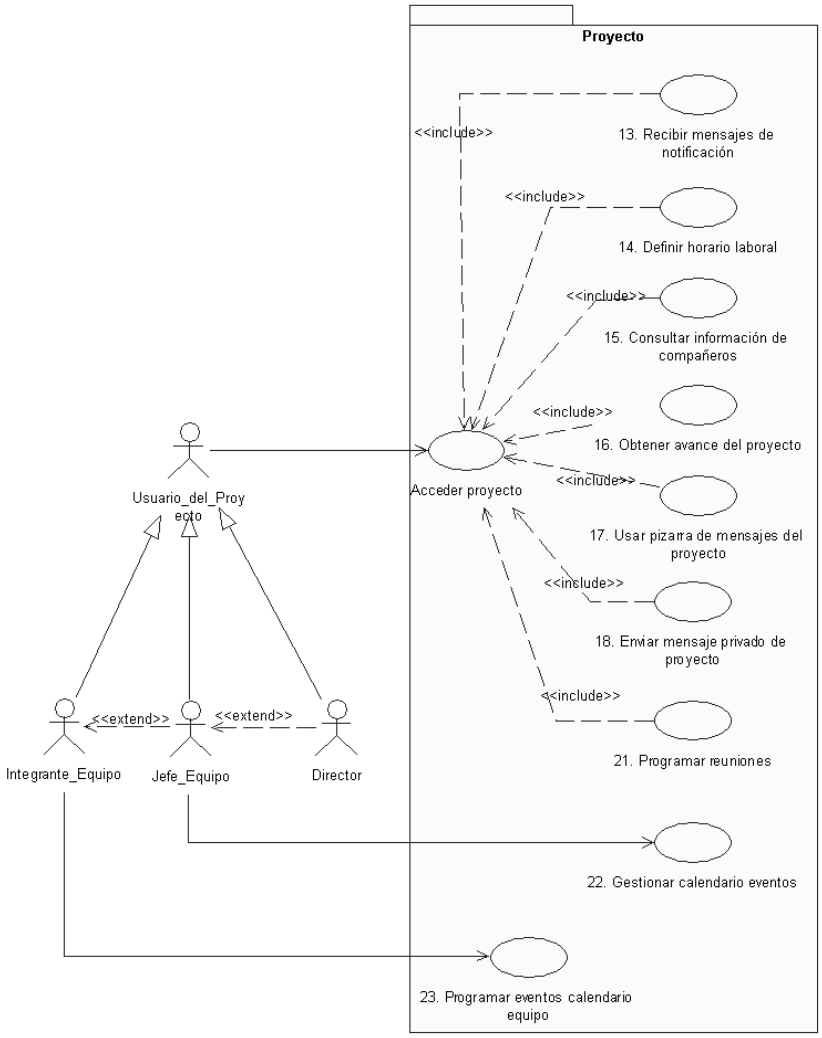


Figura 23: Casos de uso del Usuario del Proyecto-Parte 2

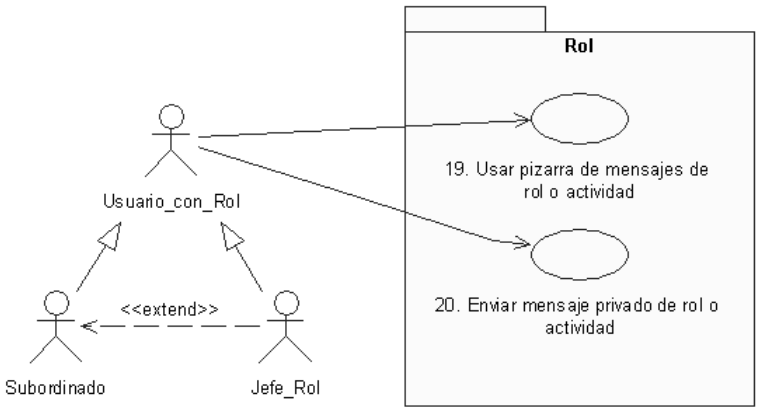


Figura 24: Casos de uso del Usuario con Rol-Parte 2

5.6 MODELO DE BASE DE DATOS

A continuación se presenta el modelo relacional de la base de datos obtenido en el proyecto, relacionado con el módulo en cuestión, dividido en vistas para mejor comprensión.

Modelo Relacional Vista 3
Los conceptos involucrados dentro de este modelo son: • **Proyecto:** representa un proyecto de desarrollo de software. • **Usuario:** representa a la persona registrada en el sistema. • **Equipo:** representa un grupo de usuarios. • **CalendarioEventos:** representa los eventos que puede programar un equipo de trabajo. Estos conceptos se transforman en entidades, a las cuales se les asigna un identificador único o clave primaria, identificadas con el atributo <PI>, el tipo de dato y el tamaño, y su obligatoriedad o ausencia de ella. Las relaciones entre las entidades se establecen así: • Un proyecto contiene cero o muchos equipos de trabajo. • Un equipo puede programar cero o muchos eventos.

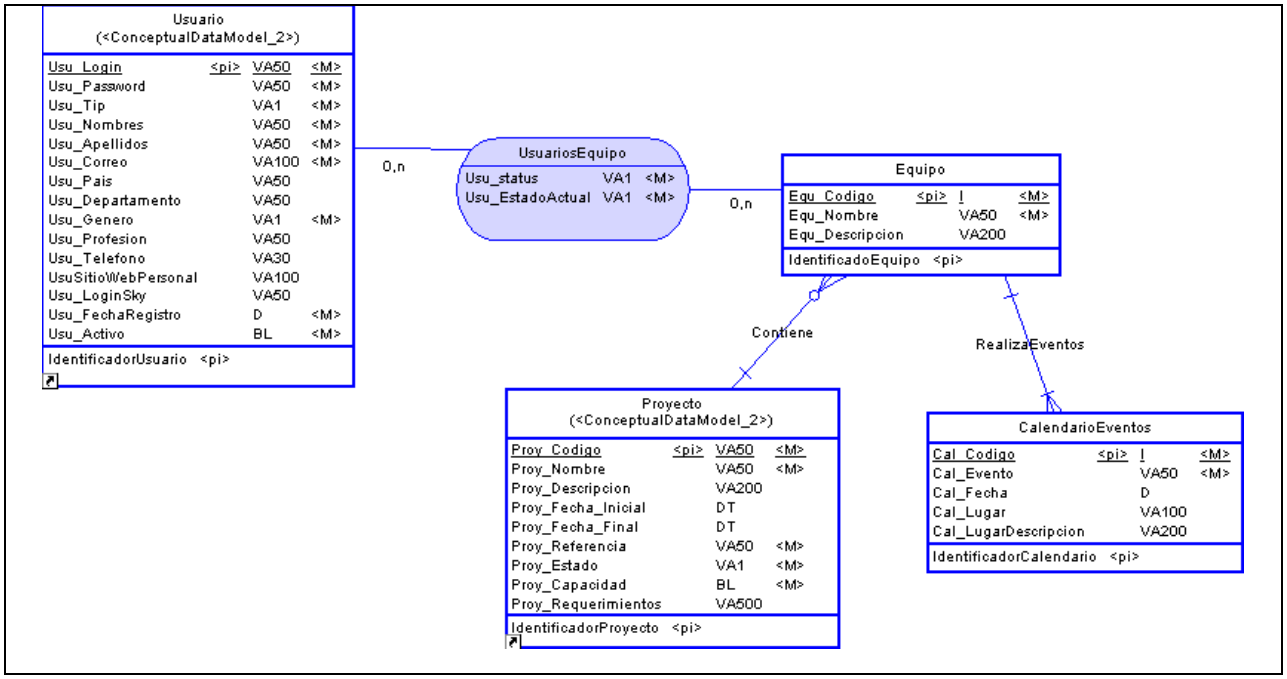


Tabla 13: Modelo relacional vista 3

Modelo Relacional Vista 4
Los nuevos conceptos involucrados dentro de este modelo son: • **Mensaje:** representa los mensajes públicos y privados entre compañeros de un proyecto, compañeros de rol y compañeros de actividad, en la pizarra o de forma privada. Estos conceptos se transforman en entidades, a las cuales se les asigna un identificador único o clave primaria, identificadas con el atributo <PI>, el tipo de dato y el tamaño, y su obligatoriedad o ausencia de ella. Las relaciones entre las entidades se establecen así: • Un proyecto contiene cero o muchos mensajes privados o públicos. Un mensaje pertenece a un único proyecto. • Un usuario escribe cero o muchos mensajes, pero un mensaje es escrito por un solo usuario.

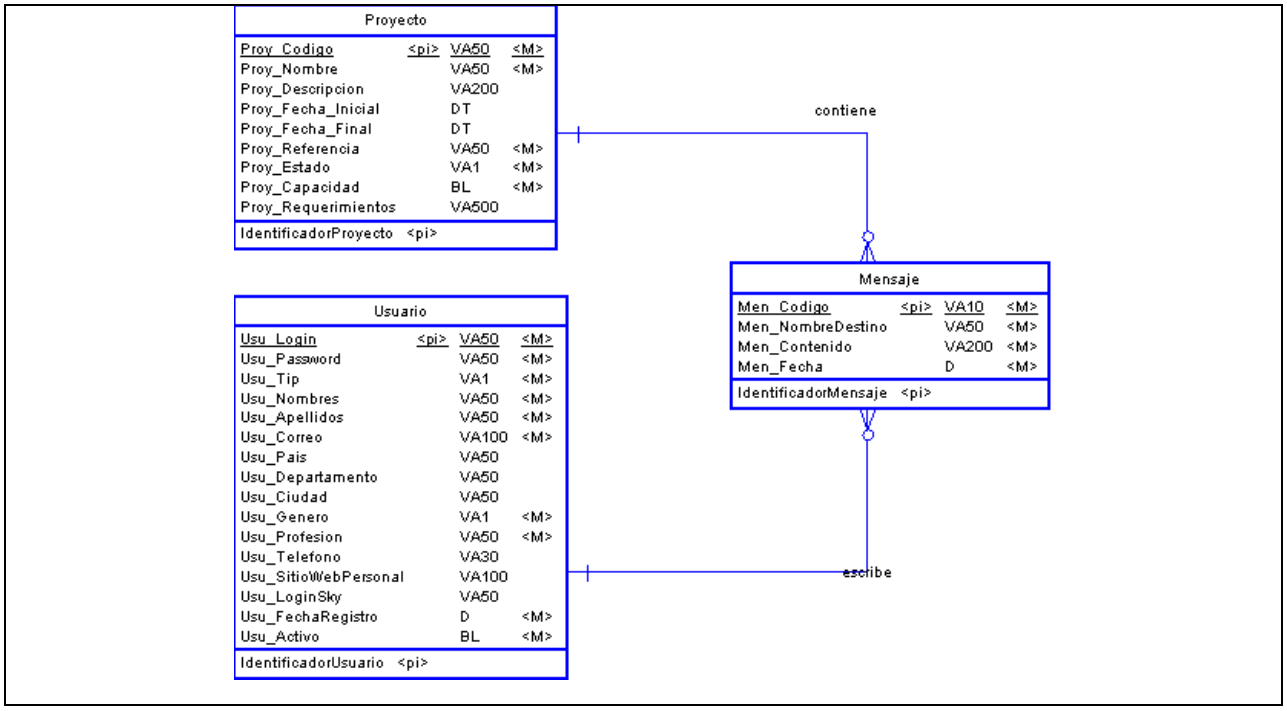


Tabla 14: Modelo relacional vista 4

Modelo Relacional Vista 5
Los nuevos conceptos involucrados dentro de este modelo son: • **Iteración:** representa un conjunto de actividades llevadas a cabo de acuerdo a un plan que lleva a producir una versión de un producto. Cada fase del proyecto está constituido por una o más iteraciones. • **Rol:** representa el papel que un individuo puede desempeñar en el desarrollo de software. Estos conceptos se transforman en entidades, a las cuales se les asigna un identificador único o clave primaria, identificadas con el atributo <PI>, el tipo de dato y el tamaño, y su obligatoriedad o ausencia de ella. La asociación Rol_Usuario_Iteracion permite consultar los roles asignados a un usuario dentro de una iteración. Además describe la jerarquía de rol que tiene asignado el usuario (Jefe, Subordinado), Indica si el usuario ha aceptada la asignación de dicho rol, y la fecha de asignación del rol y la fecha de aceptación del rol por parte del usuario. El campo Rol_Jerarquia en la tabla Rol_Usuario_Iteracion indica si el usuario actúa como Jefe de Rol ó Subordinado de rol. El campo Rol_Aceptacion indica si el usuario ha aceptado o rechazado el rol asignado.

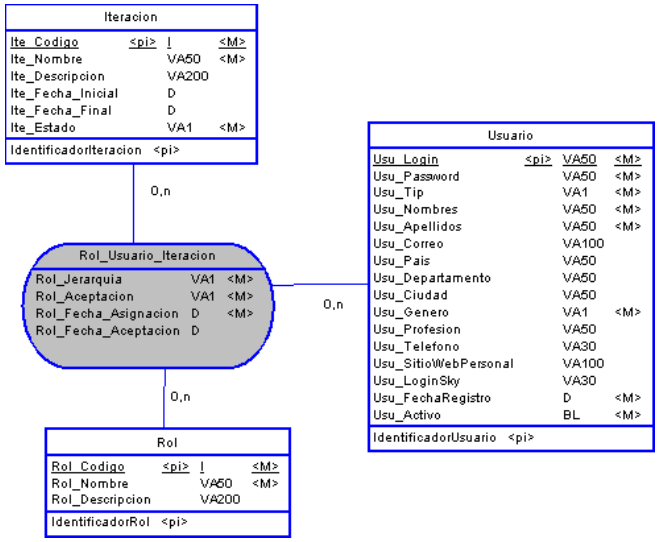


Tabla 15: Modelo relacional vista 5

Modelo Relacional Vista 6
Los nuevos conceptos involucrados dentro de este modelo son: • **Reuniones:** representa la reunión que puede ser programada por un usuario dentro de un proyecto. • **Horario:** representa las horas de dedicación al proyecto del usuario. Estos conceptos se transforman en entidades, a las cuales se les asigna un identificador único o clave primaria, identificadas con el atributo <PI>, el tipo de dato y el tamaño, y su obligatoriedad o ausencia de ella. Las relaciones entre las entidades se establecen así: • Un usuario puede programar cero o muchas reuniones. • Un usuario dedica cero o muchas horas al proyecto.

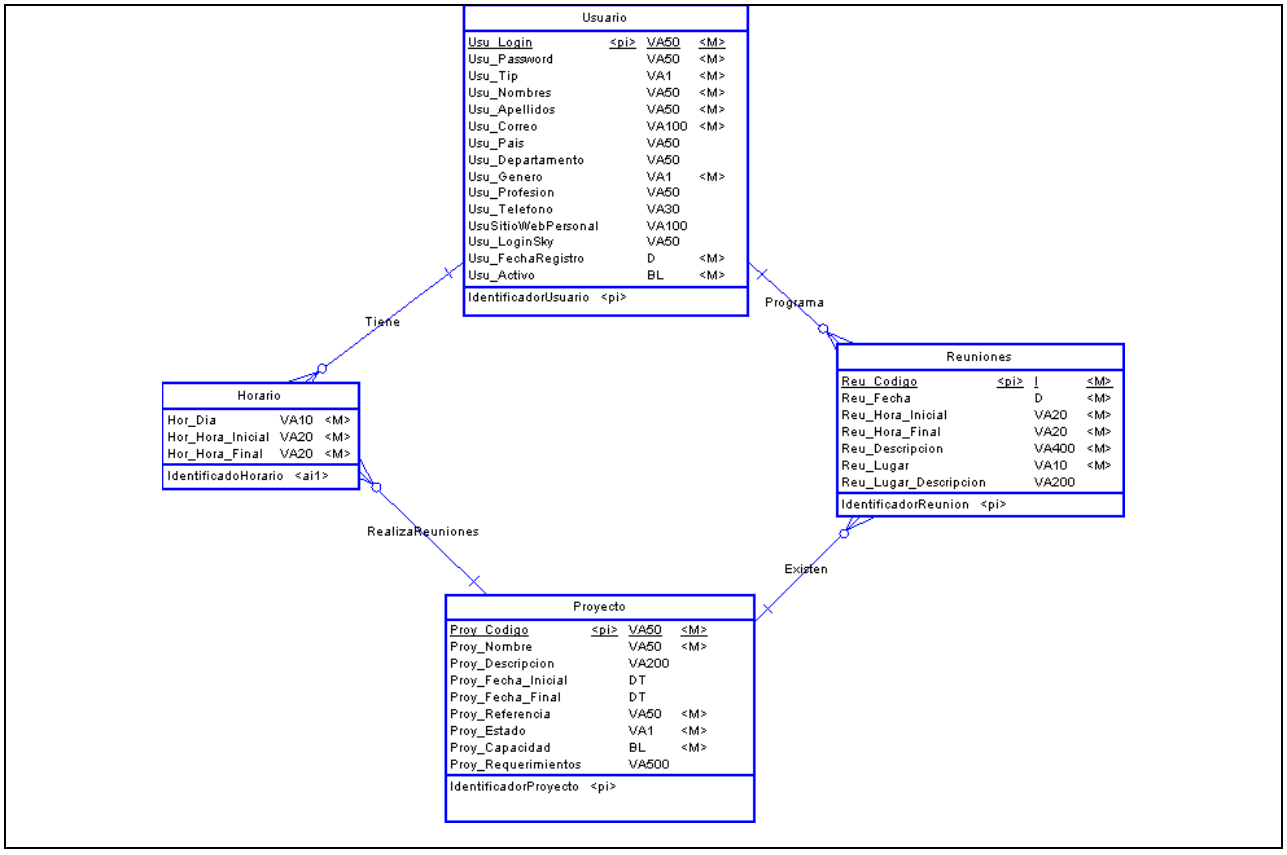
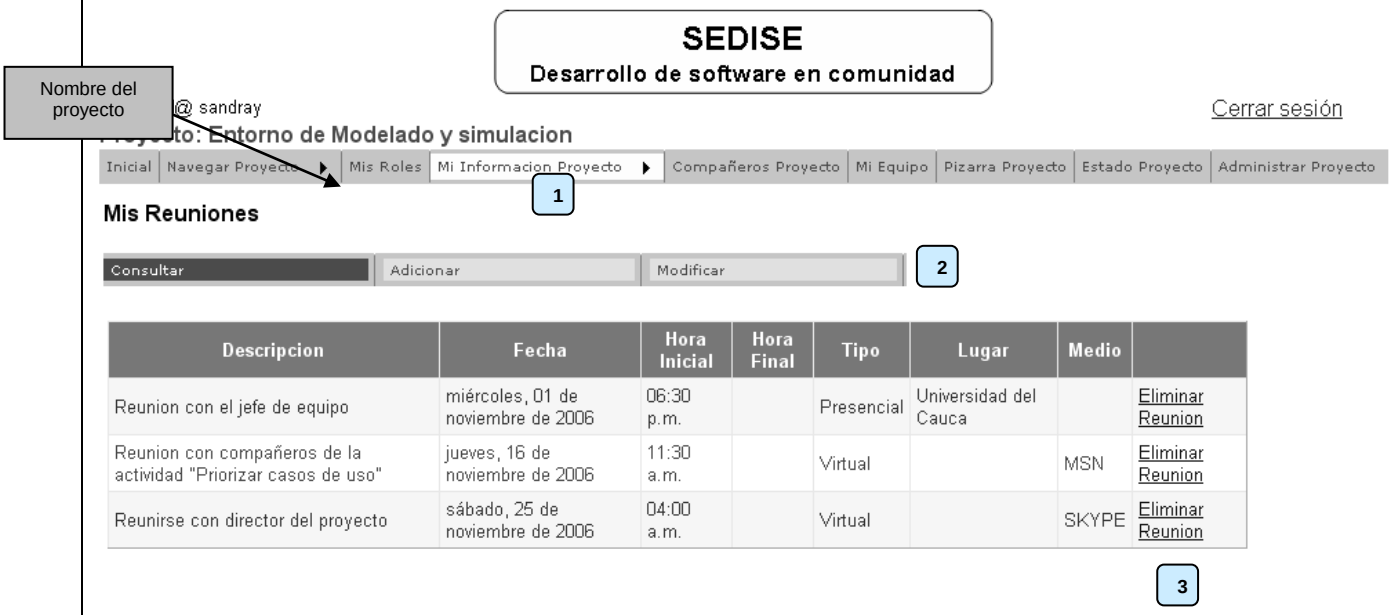


Tabla 16: Modelo relacional vista 6

5.7 CASOS DE USO REALES

Los casos de uso reales muestran la interacción del usuario con la herramienta construida, involucrando elementos de interfaz. A continuación se muestra un ejemplo de un caso de uso real obtenido para el módulo en cuestión.

CASO DE USO REAL	Programar reuniones
ACTOR	Usuario del proyecto
DESCRIPCION	Este caso de uso comienza cuando el usuario decide programar reuniones como parte de su agenda personal, dentro de un proyecto.



CURSO NORMAL DE LOS EVENTOS	
Acción del Actor	**Respuesta del Sistema**
1. El usuario da clic en el enlace de "Mis Reuniones" [1].	**2.** El sistema carga la pagina de reuniones con el menú de funcionalidad [2].
3. El usuario selecciona la opción de Consultar y da clic [2].	**4.** El sistema carga las reuniones programadas por el usuario dentro del proyecto.
5. El usuario puede eliminar alguna de las reuniones dando clic en el enlace [3].	

Tabla 17: Caso de uso real Programar reuniones

En el siguiente capítulo, del presente documento, se describe y explica la funcionalidad de uno de los módulos lógicos de la herramienta construida, llamado "Módulo de Control". Se adicionan algunos artefactos obtenidos durante la construcción de la funcionalidad del módulo.

CAPITULO 6: MÓDULO DE CONTROL

Tomando como referente la descripción de la herramienta realizada en el capitulo 3, en la cual se describe la arquitectura lógica, se procede a explicar uno de los módulos de la arquitectura llamado Módulo de Control.

Este capitulo pretende explicar la solución obtenida para soportar el módulo de control, cuyo propósito consiste en soportar la gestión de usuarios de la herramienta, la adecuación, puesta en marcha y mantenimiento de los proyectos de software en comunidad.

El conjunto de artefactos software obtenidos durante la construcción de este módulo se citan a continuación.

Nombre artefacto	Ubicación en documento de anexos (No. De Página)
Funciones del sistema	39
Requerimientos no funcionales	40
Especificación de requerimientos funcionales	38, 58
Lista de actores	25, 62
Lista de casos de uso	26, 41, 61
Diagramas de casos de uso	27, 44, 68
Especificación de alto nivel de los casos de uso	28, 29, 30, 46, 47
Especificación expandida de los casos de uso	31, 32
Diagrama conceptual	35, 50, 72
Diagramas de secuencia	36, 37
Arquitectura lógica	52, 58
Diagramas de colaboración	76, 77
Diagramas de clases	79, 80, 81
Casos de uso reales	74, 93, 94
Arquitectura de implementación	99, 100, 101, 102, 103
Descripción de implementación	116, 117
Vista física del sistema	114
Diagrama de despliegue	118
Pruebas	118, 119, 120, 121

Tabla 18: Conjunto de artefactos resultantes – Módulo de control

La descripción detallada y especificada de los artefactos software, organizados en iteraciones, se puede encontrar en el ANEXO B: Desarrollo de la herramienta en iteraciones.

A continuación se muestran los principales artefactos resultantes del proceso de construcción del módulo.

6.1 FUNCIONES DEL SISTEMA

Según Larman (2003), las funciones del sistema son las acciones que el sistema debe hacer y pueden priorizarse de acuerdo a su grado de importancia o visibilidad en el sistema, es decir, se puede establecer cuales de ellas son esenciales para el funcionamiento adecuado del sistema y cuales por el contrario pasarían inadvertidas pero que consumen tiempo y recurso.

Las funciones se pueden categorizar en:

- Evidente: la función debe realizarse y el usuario debería saber que se ha realizado.
- Oculta: la función debe realizarse, aunque no es visible para los usuarios.
- Superflua: son funciones opcionales; su inclusión no repercute significativamente en el costo del proyecto ni en otras funciones.

De acuerdo a esta categorización, las funciones del sistema se muestran a continuación en la tabla 19.

Referencia	Función	Categoría
FS-01	Procesar la información del usuario para registrarlo en el sistema	Evidente
FS-02	Enviar correo electrónico de notificación de registro al nuevo usuario, con la contraseña personal para poder iniciar sesión	Oculta
FS-03	Solicitar y procesar la información del usuario para iniciar sesión en el sistema (analizar el tipo de usuario)	Evidente
FS-04	Conectarse al sistema gestor de base de datos	Oculta
FS-05	Consultar la información de los proyectos registrados en la base de datos	Evidente
FS-06	Visualizar la información del proyecto	Evidente
FS-07	Enviar correo electrónico de notificación de la inscripción del usuario a un proyecto al nuevo usuario	Oculta

FS-08	Enviar correo electrónico de notificación del cambio de estado de un usuario a inactivo, a todos los integrantes del proyecto	Oculta
FS-09	Solicitar y procesar la información de un proyecto para registrarlo en la base de datos	Evidente
FS-10	Enviar correo electrónico de notificación, al usuario, de la asignación de un rol dentro del proyecto	Oculta
FS-11	Crear la estructura de organización de los archivos (versiones de artefactos)	Oculta
FS-12	Almacenar los archivos (versiones de artefactos) de acuerdo con la estructura de organización definida	Oculta
FS-13	Enviar correo electrónico de notificación, al usuario, de la asignación de una actividad dentro del proyecto	Oculta
FS-14	Enviar correo electrónico de notificación, al usuario, de la asignación de un artefacto dentro del proyecto	Oculta
FS-15	Enviar correo electrónico con la propuesta de creación de un proyecto al Administrador	Oculta

Tabla 19: Funciones del Sistema

6.2 REQUERIMIENTOS NO FUNCIONALES

Los requerimientos no funcionales tienen que ver con características que de una u otra forma puedan limitar el sistema, como por ejemplo, el rendimiento (en tiempo y espacio), interfaces de usuario, fiabilidad (robustez del sistema, disponibilidad de equipo), mantenimiento, seguridad, portabilidad, estándares, etc.

Atributo	Detalles y restricciones de frontera
Facilidad de uso	El usuario de la herramienta SEDISE podrá instanciar y manejar un proceso distribuido de desarrollo de software sin requerir de un amplio conocimiento técnico de la herramienta.
Interfaz del usuario	La interfaz con el usuario debe ser realizada en un entorno Web para ser ejecutada por los navegadores más comunes del mercado. Las pantallas deben ser sencillas e intuitivas para una fácil interacción del usuario y la herramienta y deben ser mostradas en castellano. Se debe mantener la misma distribución física en las pantallas, es decir si en más de una pantalla existe el mismo icono, en todas debe ubicarse en el mismo lugar y orden.
Tiempo de respuesta	Las actividades que se lleven a cabo a través de la herramienta deben requerir de intervalos de

	tiempo pequeños para agilidad de los procesos del usuario.
Plataformas	El sistema en el entorno del usuario debe ser soportado por cualquier equipo que pueda ejecutar un navegador de Internet, preferiblemente Internet Explorer. En el servidor se debe contar con Windows Server, .Net Framework 2.0, SQL Server 2000.
Seguridad	El acceso al sistema debe ser seguro; por lo tanto se requiere la identificación del usuario y el ingreso de una contraseña.
Mantenibilidad	El sistema debe ser modular para facilitar el mantenimiento y las futuras ampliaciones de acuerdo a las necesidades cambiantes.
Fiabilidad	El sistema debe comportarse consistentemente, sin perder información y respondiendo de la misma forma ante pedidos iguales.

Tabla 20: Requerimientos no funcionales

6.3 ARQUITECTURA DE LA HERRAMIENTA

En la Figura 25 se muestra la arquitectura lógica de la herramienta detallando el módulo de Control de la herramienta.

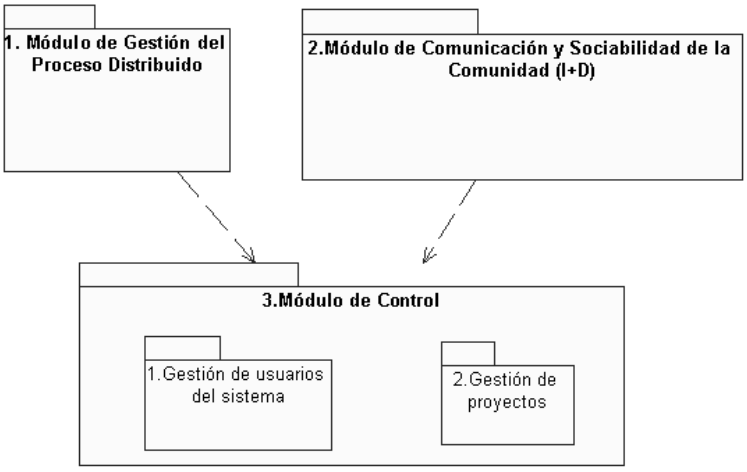


Figura 25: Módulo de Control

Este módulo se encuentra conformado a la vez por dos módulos, explicados así:

- **Módulo 1: Gestión de usuarios del sistema.** Módulo encargado de controlar el registro e inicio de sesión de usuarios, permitir la modificación de datos personales de un usuario y la realización de funciones de gestión de usuarios (consultar, eliminar, cambiar tipo de usuario).

- **Módulo 2: Gestión de proyectos.** Módulo encargado de permitir las funciones de gestión de proyectos (crear, asignar director, modificar y eliminar) y visualizar la lista de proyectos activos en el sistema y a los que pertenece un usuario.

6.4 REQUERIMIENTOS FUNCIONALES DEFINIDOS PARA LA HERRAMIENTA

Para la construcción del módulo de Control de la herramienta se tomaron como requerimientos funcionales los siguientes:

Módulo de Gestión de usuarios del sistema:

- **Registrarse en el sistema.**
  Poder obtener una cuenta en el sistema ingresando un login y una contraseña.
- **Iniciar sesión.**
  Poder iniciar sesión en el sistema ingresando el login y la contraseña obtenidos en el proceso de registro como nuevo usuario.
- **Gestionar usuarios del sistema.**
  Poder consultar, eliminar ó modificar el tipo de usuario de los usuarios registrados en el sistema.
- **Consultar y modificar la información referente a sus datos personales.**
  Poder modificar sus datos personales en cualquier momento.

Módulo de Gestión de proyectos:

- **Ver la información de los proyectos actualmente registrados.**
  Poder observar la información general de los proyectos que se encuentran registrados en el sistema.
- **Consultar el listado de los proyectos a los cuales se ha vinculado.**
  Poder conocer la lista de proyectos en los cuales se encuentra registrado el usuario como trabajador activo y dar la posibilidad de asociarse a un proyecto como nuevo miembro.
- **Gestionar proyectos.**
  Poder ejecutar acciones de gestión sobre proyectos como crear proyecto, asignar director al proyecto, y eliminar proyecto.

- **Proponer la creación de un nuevo proyecto.**
  Poder proponer al administrador la creación de un nuevo proyecto indicando las características como nombre, descripción, fechas de inicio y fin y el propósito del proyecto propuesto.

6.5 LISTA DE CASOS DE USO

En la siguiente tabla se observa la relación de cada requerimiento con los casos de uso que se deducen:

Requerimiento	Caso de Uso
Registrarse en el sistema.	24. Registrarse en el sistema
Ver la información de los proyectos actualmente registrados.	25. Ver listado de todos los proyectos
Iniciar sesión.	26. Iniciar sesión
Consultar el listado de los proyectos a los cuales se ha vinculado.	27. Consultar su listado de proyectos
Gestionar usuarios del sistema	28. Gestionar usuarios
	28.1 Consultar usuario
	28.2 Eliminar usuario
	28.3 Modificar tipo de usuario
Gestionar proyectos	29. Gestionar proyecto
	29.1 Crear proyecto
	29.2 Asignar director
	29.3 Eliminar proyecto
Consultar y modificar la información referente a sus datos personales	30. Modificar información personal
Proponer la creación de un nuevo proyecto	31. Proponer nuevo proyecto

Tabla 21: Requerimientos y casos de uso módulo de control

6.6 DIAGRAMAS DE CASOS DE USO

El diagrama de casos de uso se realiza teniendo en cuenta la lista de casos de uso obtenida (ver Tabla 14: Requerimientos y casos de uso módulo de control) y la clasificación de los actores del sistema.

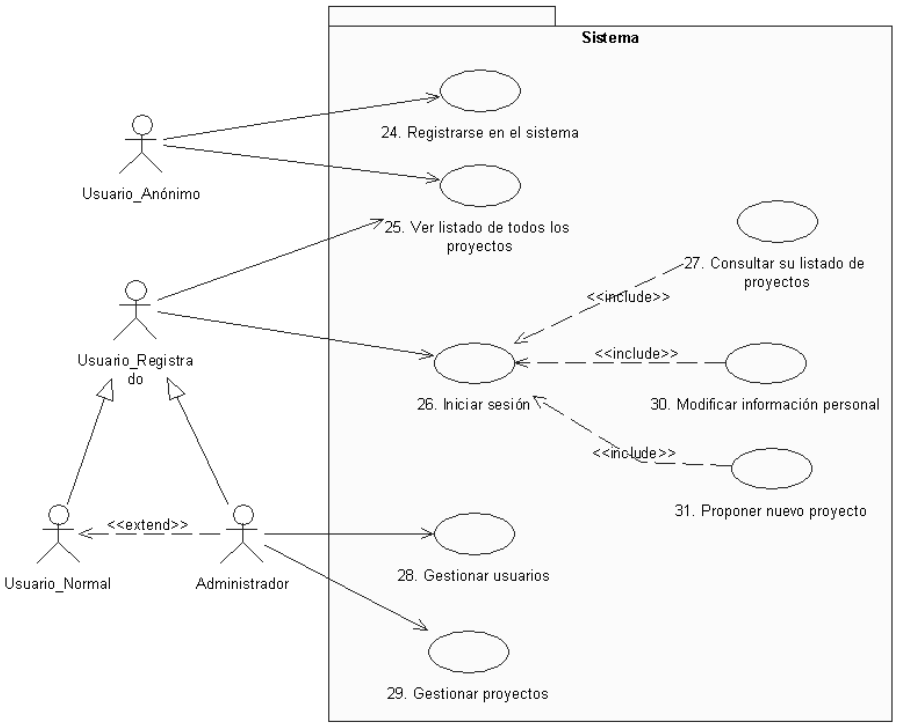


Figura 26: Casos de Uso del Usuario Registrado-Parte 2

6.7 MODELO DE BASE DE DATOS

A continuación se presenta el modelo relacional de la base de datos obtenido en el proyecto relacionado con el módulo en cuestión.

Modelo Relacional Vista 6
Los conceptos involucrados dentro de este modelo son: • **Proyecto:** representa un proyecto de desarrollo de software. • **Usuario:** representa a la persona registrada en el sistema. Este modelo tiene entidades, a las cuales se les asigna un identificador único o clave primaria, identificadas con el atributo <PI>, el tipo de dato y tamaño, y su obligatoriedad o ausencia de ella. La asociación Usuarios_del_Proyecto, permite registrar usuarios en los proyectos. Además tiene la información de qué tipo de usuario es dentro del proyecto: Director ó Trabajador. El campo Estado permite determinar si el usuario ha sido aceptado dentro del proyecto. El campo Usu_Tip en la tabla Usuario indica si el usuario es Usuario Normal ó Administrador. El campo Usu_Activo en la tabla Usuario indica si el usuario ha realizado la autenticación basada en correo electrónico. El campo Proy_Capacidad en la tabla Proyecto indica si el cupo de usuarios en el proyecto está copado. El campo Proy_Requerimientos en la tabla Proyecto permite registrar los requerimientos necesarios para ser parte del proyecto y los cuales se tendrán en cuenta para aceptar o no a un usuario como integrante del proyecto.

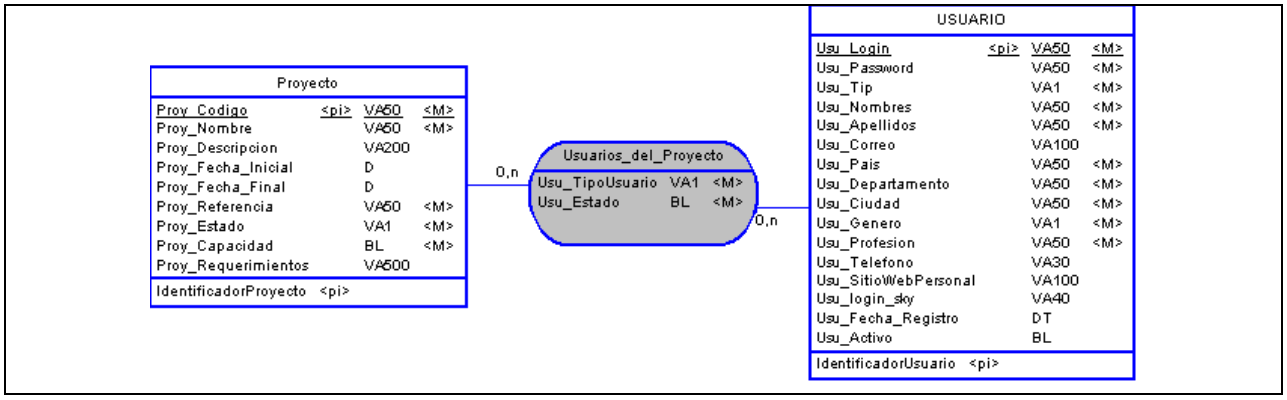
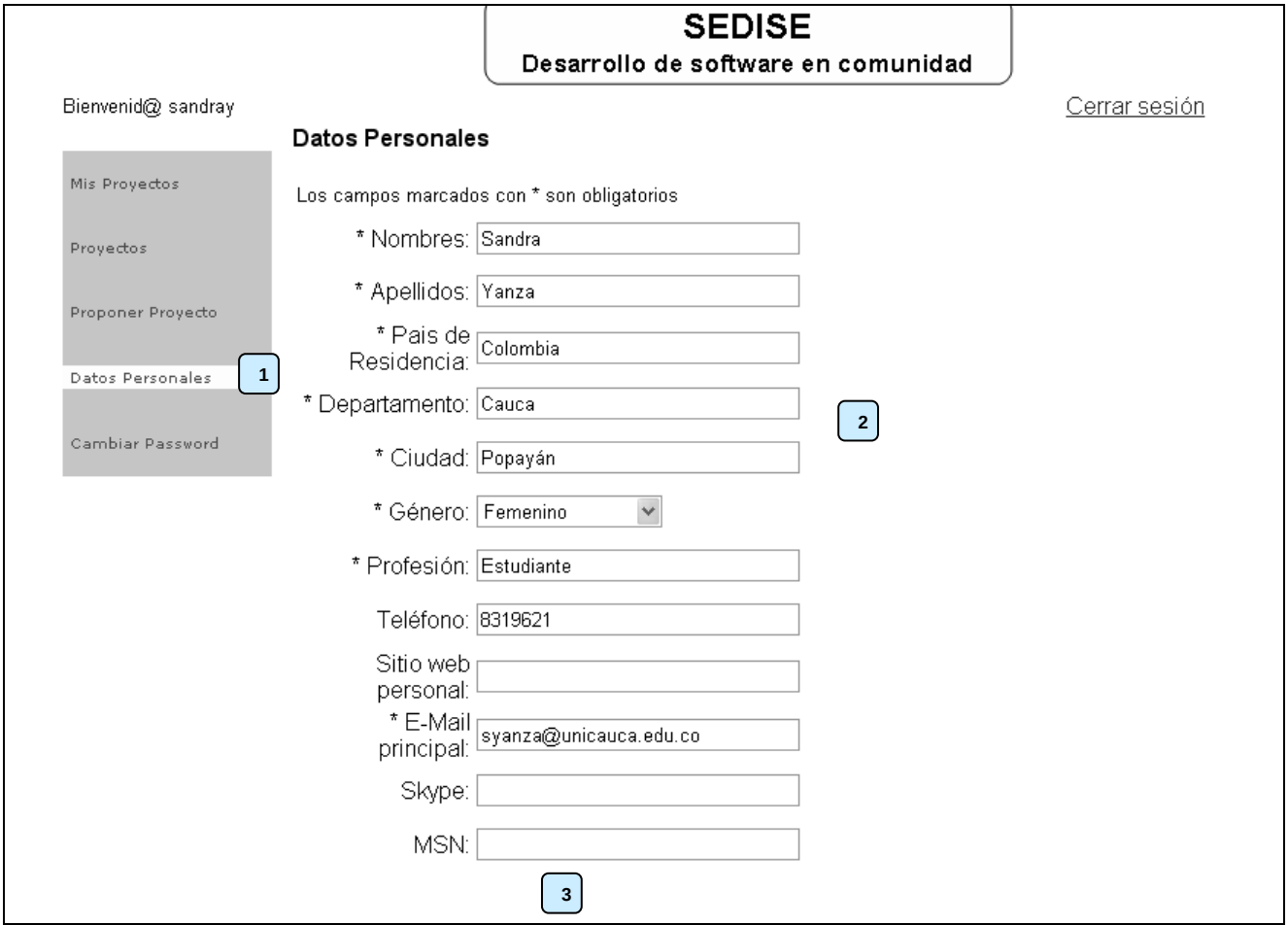


Tabla 22: Modelo relacional vista 6

6.8 CASOS DE USO REALES

Los casos de uso reales muestran la interacción del usuario con la herramienta construida, involucrando elementos de interfaz. A continuación se muestra un ejemplo de un caso de uso real obtenido para el módulo en cuestión.

CASO DE USO REAL	Modificar información personal
ACTOR	Usuario Registrado
DESCRIPCION	Este caso de uso comienza cuando el usuario decide modificar, actualizar o consultar los datos personales con los cuales se ha registrado en el sistema. Este caso de uso finaliza cuando el usuario puede con éxito, consultar o actualizar sus datos personales.



CURSO NORMAL DE LOS EVENTOS	
Acción del Actor	**Respuesta del Sistema**
1. El usuario, después de iniciar sesión, da clic en la opción de "Datos Personales" [1].	**2.** El sistema carga los datos personales del usuario en la sección [2].
3. El usuario procede a modificar los datos personales [2].	
4. El usuario da clic en el botón "Modificar" para confirmar la modificación realizada [3].	**5.** El sistema valida los datos y almacena la nueva información suministrada.

En el siguiente capitulo, se presentan unas recomendaciones o sugerencias para maximizar la funcionalidad de la herramienta y promover un ambiente adecuado entre los miembros de la comunidad.

CAPITULO 7: BUENAS PRÁCTICAS SUGERIDAS

En este capítulo se han recopilado recomendaciones orientadas a maximizar la funcionalidad de la herramienta, intentando que el trabajo del usuario con la misma sea más efectivo y se aproveche la sinergia de la comunidad, todo orientado al éxito del proyecto en el marco de un desarrollo en comunidad concreto[19].

7.1 PRINCIPIOS Y BUENAS PRÁCTICAS DEL PROCESO DISTRIBUIDO DE DESARROLLO APLICABLES UTILIZANDO LA HERRAMIENTA

En esta sección se explica cómo la funcionalidad de la herramienta se puede encapsular en las prácticas sugeridas por un proceso distribuido de desarrollo [7], recomendando de qué manera los usuarios o miembros de la comunidad pueden actuar para obtener el resultado y la efectividad esperada, dentro de cada proyecto de software y mostrando que estas prácticas son realizables a través del uso de la herramienta.

La definición de las prácticas sugeridas por el proceso distribuido de desarrollo se pueden observar en detalle en el ANEXO A: Proceso Distribuido de Desarrollo.

- **Promueva el desarrollo ágil, iterativo, incremental y comunitario**

La herramienta desarrollada permite al usuario que actúa como Director de un proyecto, realizar la planificación del proyecto estableciendo múltiples iteraciones en cada fase del ciclo de vida. En cada iteración se ejecutan actividades que manipulan y generan múltiples versiones de artefactos. Se plantea la figura de una persona encargada de crear y asignar los diferentes artefactos, para que de cada uno se generen múltiples versiones o incrementos de software para ser publicados y compartidos en la comunidad que desarrolla el proyecto. Al entregar estos resultados a la comunidad se intenta promover

[19] En el contexto de la comunidad de dinámica de sistemas y pensamiento sistémico que agrega a varias universidades públicas del territorio nacional.

la retroalimentación entre los mismos trabajadores, los cuales pueden enviarse mensajes de apoyo o critica que permitan evaluar su trabajo, aprovechando la sinergia de la crítica comunitaria.

- **Promueva una gestión de requerimientos en comunidad**

La herramienta ofrece la posibilidad de asignar una actividad, dentro de una iteración, a varios trabajadores, los cuales tienen a disposición el espacio de pizarra de mensajes para, en conjunto, debatir, organizar y planear una manera de realizar y nombrar las versiones de artefactos a generar en la actividad, y de esta manera promover un ambiente de discusión, comunicación y colaboración alrededor de los requerimientos del sistema, sin subestimar los aportes de ningún miembro de la comunidad, en su lugar, reconocerlos públicamente y de manera positiva lo que promoverá la motivación individual y grupal. El reconocimiento público se puede realizar haciendo uso de la pizarra de mensajes que contiene mensajes públicos dentro del proyecto.

- **Verifique calidad constantemente aprovechando la comunidad**

La herramienta fomenta la verificación y evaluación de la calidad de los resultados de elaboración (versiones) de los artefactos en comunidad, para cumplir con el hito definido al final de cada iteración. Para cada fase, se tiene la posibilidad de definir un hito para cumplir con ciertos artefactos, cuya calidad ha sido analizada en comunidad. El estado exitoso de una actividad depende del logro obtenido en los artefactos generados. El estado exitoso de una iteración y de una fase, esta supeditado al logro del hito de la iteración y al hito de la fase, respectivamente. Cuando una nueva versión de un artefacto es adicionada, la herramienta automáticamente publica un mensaje en la pizarra de actividad o de rol informando este suceso, para que los usuarios interesados la descarguen, la analicen y envíen sugerencias, criticas y comentarios al autor de la versión, todo encaminado a verificar y mejorar la calidad de los artefactos resultantes.

- **Promueva la gestión del cambio en comunidad**

La herramienta ha implementado un mecanismo de control para soportar la trazabilidad entre los artefactos y versiones de artefactos en cualquier iteración. De esta manera se tiene la posibilidad de conocer los artefactos utilizados como referencia para crear un nuevo artefacto. Al publicar la versión de un artefacto -un archivo- se hace necesario indicar la versión anterior o versión base usada para generarla, sin importar que la versión base tenga como autor a otro trabajador del proyecto, ya que los resultados se propagan hacia toda la comunidad.

- **Promueva el uso de una plataforma de integración en comunidad**

La herramienta desarrollada esta planteada como una plataforma de integración en comunidad, ya que aprovecha las TIC[20] como mecanismo de integración para la comunidad al ofrecer servicios para el logro efectivo de la gestión, comunicación, colaboración y coordinación de los equipos de trabajo geográficamente dispersos. Hoy en día, las TIC, permiten a la sociedad de la información, superar las barreras geográficas, políticas y culturales con el propósito de allanar el camino hacia el emprendimiento de esfuerzos conjuntos. De esta manera, la plataforma permite que cada miembro de la comunidad pueda sentirse parte de ella desde cualquier lugar y a cualquier hora, al estar disponible para todos los miembros de los equipos geográficamente dispersos, a través de Internet.

- **Trate a los usuarios como colaboradores**

La herramienta intenta fomentar la colaboración entre miembros de la comunidad, como la forma mas apropiada de mejorar los artefactos resultantes. De esta manera, se promueve la filosofía de estimulo y reconocimiento constante para aquellos usuarios colaboradores que toman parte en la acción, usando el enfoque de desarrollar software abierto y en comunidad, para que las fallas sean detectadas fácilmente al poder ser revisadas por múltiples miembros a la vez. El estimulo y reconocimiento a los usuarios se puede realizar a través de mensajes públicos en la pizarra ó mensajes privados a un usuario en específico.

- **Controle su documentación**

La herramienta dispone de un mecanismo para controlar las versiones de los artefactos generados durante la ejecución de un proyecto. El mecanismo consiste en permitir la agregación de nuevos archivos de versiones, publicar las versiones (archivos) para que estén disponibles para la comunidad, permitir la descarga de estos archivos como documentación libre para todos los miembros del proyecto y gestionar la organización y almacenamiento físico de estos archivos, teniendo en cuenta que son resultados tangibles de una actividad que se ejecuta en una iteración, la cual hace parte de una fase del proyecto.

7.2 BUENAS PRÁCTICAS PARA APLICAR EN COMUNIDAD USANDO LA HERRAMIENTA

A continuación se sugieren varias recomendaciones orientadas a aplicar en comunidad y a enriquecer el uso, que realiza el miembro de la comunidad, de la herramienta construida, con el fin de maximizar su funcionalidad. Estas prácticas fueron deducidas y sintetizadas por los autores de este trabajo de grado, al realizar las pruebas a la funcionalidad de la herramienta, a medida que se implementaba, haciendo

[20] Tecnologías de la Información y las Comunicaciones

uso de información ficticia y solicitando la colaboración de algunos compañeros de estudio, quienes proporcionaron sugerencias y recomendaciones, que permitieron detectar los principales aspectos que deben tener en cuenta los usuarios para hacer un mejor uso de la herramienta y facilitar que el proyecto de software que se ejecute pueda culminar con éxito.

Las pruebas que dieron origen a las buenas prácticas, que siguen a continuación, se basaron en la utilización de algunos equipos de tesistas del programa de Ingeniería de Sistemas, tomando como criterios de evaluación los mecanismos de comunicación, coordinación y colaboración, entre los equipos.

- **Manejar herramientas tecnológicas seleccionadas de común acuerdo**

Antes de poner en ejecución el proyecto en comunidad, es conveniente y necesario que se realice en consenso el establecimiento de las herramientas tecnológicas para ser usadas en la elaboración de los artefactos. Se aconseja que la selección de estas herramientas se realice en las reuniones presenciales, en las que todos los miembros de la comunidad puedan sugerir opciones que permitan plantear una discusión en torno al nivel de conocimiento, experiencia, ventajas e inconvenientes de cada herramienta. El hacer uso de herramientas de común acuerdo permite establecer, por ejemplo, una comunicación sin ambigüedad con los artefactos de modelado y soportar la integración entre artefactos de implementación.

- **Utilizar un modelo de documentación estándar en el proyecto**

Antes de poner en ejecución el proyecto en comunidad, es conveniente establecer un modelo de documentación estándar en el proyecto, que permita fomentar una comunicación sin ambigüedad. Se puede sugerir un formato para documentar el código realizado, de tal manera que no sólo los creadores lo entiendan, sino que además los otros programadores sepan qué está haciendo y porqué. Para los artefactos de requerimientos, análisis y diseño, se puede dotar a los miembros de la comunidad de un conjunto de plantillas que puedan adaptar a las necesidades del proyecto, que promuevan el uso de un lenguaje en común.

- **Entender las responsabilidades asignadas**

Cuando se le haya asignado un rol de gestión como Director del proyecto, Jefe de Equipo ó Jefe de Rol, se le notificará mediante un mensaje de correo electrónico dicha asignación explicando las funciones que conlleva dicho rol. En caso de necesitar mayor información sobre cómo realizar su trabajo no dude en consultar y solicitar una información mas detallada al respecto, a la persona que le haya asignado dicho rol, que le permita ejecutar su trabajo de manera eficiente y sin contratiempos. El conocimiento de todos los roles y su funcionalidad dentro de la herramienta para todos los usuarios permite entender la

necesidad de los compañeros y la importancia del rol desempeñado, valorando más su trabajo para actuar en el momento de trabajar en equipo de manera colaborativa.

- **Evaluar en comunidad la calidad de todos los artefactos**

El propósito del trabajo en comunidad se orienta a la filosofía de trabajo y comportamiento de los participantes en el desarrollo de software libre y las dinámicas de grupo, roles y normas que surgen para lograr con éxito la consecución descentralizada de proyectos de programación con código abierto en Internet. Tal situación busca promover que los resultados de la elaboración de sus artefactos sean propagados a la comunidad que ejecuta el proyecto, haciendo uso de la herramienta presentada en este documento, para que los demás puedan valorar su trabajo y utilizarlos como base para el desarrollo de otros artefactos, y que usted pueda realizar las mismas acciones con el trabajo propagado por los demás miembros de la comunidad.

- **Mantener siempre en mente el alcance de los logros propuestos**

La planeación que se realiza al inicio de cada fase e iteración esta orientado al logro de ciertos objetivos. Por esto es necesario que cada miembro se forme una mentalidad de que el trabajo en equipo o trabajo colaborativo es la capacidad de un grupo humano para integrarse en una dirección para lograr objetivos, y que este tipo de trabajo produce un resultado mayor que la suma de los esfuerzos individuales. Para el logro de estos objetivos se definen los hitos de iteración y los hitos de fase, que establecen los artefactos con los cuales se debe cumplir para satisfacer los requerimientos planteados. De esta manera se pueden seleccionar personas que sean encargadas de comunicar de manera constante el hito en el cual se está trabajando para que los miembros lo tengan siempre presente en su trabajo. Estas personas pueden usar la funcionalidad de publicar mensajes, que posee la herramienta construida. Los miembros también tienen la posibilidad de consultar estos hitos en la herramienta, obtenida como resultado en este trabajo.

- **Planear el proyecto que será soportado por la herramienta software**

Antes de empezar a utilizar la herramienta software presentada en este trabajo, se hace necesario tener formalizada la idea del proyecto software que se piensa poner en ejecución. Tal proyecto puede estar caracterizado por la lejanía geográfica que presentan los integrantes. En general, la idea de la herramienta es dar soporte a este tipo de proyectos facilitando la comunicación, colaboración y coordinación de los miembros, pero la responsabilidad de la ejecución adecuada del proyecto recae en las personas, quienes deben encargarse de atraer gente, interesarla en lo que se esta haciendo y mantenerla contenta con el trabajo que se esta desarrollando, todo esto encaminado a la constitución de una verdadera comunidad de desarrollo en torno a un proyecto software.

- **Evaluar las propuestas de nuevos proyectos**

El administrador de la herramienta software, presentada en este trabajo, tiene entre sus funciones, la creación de proyectos. Cualquier usuario de la herramienta tiene la posibilidad de generar una propuesta para la creación de un nuevo proyecto, la cual será enviada al administrador. Esta propuesta consiste en completar un formulario con la siguiente información del proyecto: nombre, descripción, requerimientos, descriptores ó palabras clave, el tipo de software a desarrollar, el planteamiento del problema, la duración del proyecto, el valor total del proyecto, los potenciales beneficiarios, los resultados esperados, el impacto esperado, etc. En este caso se recomienda que la persona que cumpla el rol de administrador de la herramienta, posea la capacidad y habilidad para evaluar las propuestas de nuevos proyectos que reciba, teniendo en cuenta que sean propuestas de proyectos viables para ser desarrollados en comunidad.

- **Seleccionar adecuadamente al Director del proyecto**

Se recomienda al administrador, poner especial interés en la selección de la persona a la cual le va a asignar el rol de Director del proyecto, tomando como puntos a favor, ser una persona a la cual haya conocido previamente, poseer una referencia de sus capacidades de liderazgo, gestión y motivación para poner en ejecución un proyecto de software en comunidad y ser la persona que ha propuesto el nuevo proyecto siendo el más interesado en su ejecución.

- **Planificar la organización en equipos dispersos geográficamente**

Teniendo en cuenta que las personas involucradas en la comunidad se encuentran en ubicaciones dispersas geográficamente, se recomienda analizar y planear la posible organización de los equipos de trabajo procurando que se integren las personas que se encuentran en una misma ubicación física ó ubicaciones físicas cercanas, con la finalidad de que sea posible realizar reuniones presenciales sin mayores inconvenientes. De esta manera, se promueve la organización en equipos dispersos geográficamente, todos trabajando juntos en múltiples iteraciones y versiones del producto software, sobre la herramienta software que permite su integración como comunidad.

- **Iniciar la creación del equipo de trabajo en una ubicación física única**

Es recomendable que el jefe de cada equipo de trabajo, sea el promotor de la idea de integrar y familiarizar a los integrantes del equipo, incluso antes de iniciar las labores dentro del proyecto en comunidad. Para esto se recomienda llevar a cabo una reunión de integración, o las necesarias, en una ubicación física única, que promueva el conocimiento grupal, el contacto cara a cara, la colaboración, la importancia del trabajo en equipo, y que brinde el espacio para que se expliquen, se aclaren y se discutan las estrategias de trabajo y los objetivos que se esperan lograr en el proyecto.

- **Determinar la estructura y configuración de los equipos de trabajo**

Una comunidad de desarrollo esta compuesta en su esencia más profunda e importante por las personas. En consecuencia, es preciso determinar la estructura y configuración que mas se adapte para cada equipo. La estructura se basa en el objetivo del equipo y la configuración depende de la organización interna y distribución de funciones. Para conocer en detalle los diferentes tipos de estructuras y las configuraciones de equipos, refiérase a la tesis de maestría [7] usada de referente en este trabajo de grado.

- **Prepararse para cumplir adecuadamente el papel de Director del proyecto**

La persona nombrada como Director de proyecto debe poseer habilidades, competencias y conocimiento que le permitan realizar su rol de la mejor manera. Si usted es nombrado Director y no posee el entrenamiento necesario, y desea cumplir tal rol, busque y obtenga el entrenamiento necesario pero por ningún motivo se enfrente a lo desconocido. Como todo líder, un buen gerente o director de proyecto requiere capacidad para analizar y resolver problemas, poseer un sólido conocimiento, en este caso, en ingeniería de software y gerencia de proyectos, desarrollar una adecuada interacción humana, tener carácter, principios, valores, empatía, comunicación, contribución a la comunidad, innovación, creatividad, delegación, realizar crítica constructiva y valorar los aportes de cada miembro del proyecto, entre otros. Por otro lado, las condiciones en las cuales se realiza el tipo de proyecto que se plantea en este trabajo son muy diferentes a las convencionales. En este caso, en el proyecto se forman equipos de trabajo dispersos geográficamente, los cuales deben sentir la gestión de un líder, que les proporcione sentido de pertenencia al proyecto aunque se encuentren en ubicaciones remotas y que valore su trabajo estando siempre pendiente de sus logros.

- **Asumir la responsabilidad adquirida**

El usuario inscrito a un proyecto debe tener conciencia de la responsabilidad adquirida como miembro del proyecto. De esta manera, al tener asignado un rol, dentro de una iteración, el siguiente paso es ejecutar las actividades y elaborar los artefactos que se le hayan asignado, teniendo en cuenta las restricciones de tiempo. El entorno le ofrece servicios de comunicación, para tener información de los compañeros del rol y de actividad e interactuar con ellos para planear el trabajo en conjunto.

- **Poseer conocimientos técnicos básicos**

Se recomienda que el miembro del proyecto, posea un conocimiento básico en Ingeniería de software, que le permita comprender la metodología de desarrollo de software implementada para el proyecto, realizar de mejor manera su trabajo si se le asignan roles, actividades por ejecutar y artefactos por elaborar y de esta manera conocer bien el terreno en el cual se desenvuelve.

- **Saber que se cuenta con libertad de aceptar asignaciones**

Cuando al usuario se le asignan roles y actividades en un proyecto, se le envía un mensaje vía correo electrónico, que le notifica la asignación. De esta manera, el usuario al acceder al entorno software, debe proceder a aceptar o rechazar la asignación, según su disponibilidad y disposición, intentando que sea lo más pronto posible, ya que en el caso de aceptar la asignación, la persona encargada debe proceder a asignarle responsabilidades, y en el caso de rechazar la asignación, la persona encargada debe realizar la asignación a otro usuario.

- **Asignar un rol a un solo equipo de trabajo por iteración**

Se aconseja que exista un único Jefe de Rol por iteración, que sea la persona encargada de asignar responsabilidades (actividades y artefactos) a las demás personas que posean el mismo rol en una iteración. Para lograr esto, se recomienda que un rol sea asignado a un solo equipo de trabajo, para que el Jefe de Equipo asigne a un solo Jefe de Rol. Al existir varios equipos de trabajo que tengan el mismo rol asignado, en una iteración, se puede presentar la situación de que cada equipo cuente, entre sus miembros, con un Jefe de Rol, ocasionando un ambiente poco cordial y problemas de coordinación y dirección entre los miembros que realicen el papel de Jefe de Rol.

- **Modificar el estado de una actividad del proyecto**

La persona que cumpla el papel de Jefe de Rol, encargado de modificar el estado de una actividad, debería tomar como regla general que para modificar el estado de una actividad, todos los artefactos planeados para generar en la actividad, dentro de una iteración, posean el estado de “aprobados”, de esta manera se busca que los procesos de gestión de las actividades se realicen con calidad, porque de esta función depende la calidad de las iteraciones y por consiguiente de las fases del proyecto. También es necesario procurar que las actividades se logren cumplir en el tiempo planeado, ya que cualquier atraso en la ejecución de alguna retrasara la fecha fin del proyecto.

- **Definir en consenso los artefactos a generar**

Se recomienda que la definición de artefactos a generar en el proyecto se realice en una reunión presencial en la cual los miembros del equipo puedan debatir y generar sus propias opiniones al respecto, ya que serán ellos los encargados de elaborar los artefactos. Luego la persona que cumple el papel de Jefe de Rol, teniendo el listado de artefactos concebidos en consenso, se encargue de crear los artefactos en la herramienta software y gestionar la asignación de cada uno entre los miembros de la comunidad. De esta manera, cuando el usuario se entere de la asignación del artefacto, tenga un conocimiento previo acerca de lo que se espera como resultado de elaboración del artefacto.

- **Valorar los aportes de los miembros de la comunidad**

El usuario al publicar las versiones de los artefactos asignados, para ser conocidos por el resto de miembros de la comunidad, puede aprovechar la sinergia generada y con mente abierta, valorar los aportes propios y de sus compañeros y directivos. Las críticas (preferiblemente enviadas a través de mensajes privados) que realicen a su trabajo pueden resultar beneficiosas para mejorar los productos que está realizando y reflejan la actitud de colaboración de los miembros del equipo. Además la ingeniería es una actividad de carácter colectivo que involucra dinámica de trabajo en grupo, en la cual se sugiere mantenerse receptivo y aunque el comentario esté mal planteado es conveniente centrarse en el contenido e ignorar los ataques personales, aprovechando las sugerencias, sin temor a reconocer sus propios defectos.

- **Aportar crítica constructiva**

El valor de la crítica constructiva se fundamenta en el propósito de lograr un cambio favorable que beneficie a todas y cada una de las personas involucradas en el equipo de desarrollo, con actitud de respeto y sentido de colaboración. La idea es aprovechar la sinergia de la comunidad para fomentar un ambiente de discusión y promover la critica entre compañeros de actividad que se encuentren elaborando artefactos relacionados, compañeros de rol y compañeros de equipo de desarrollo, que permita mejorar el trabajo realizado y fomente las relaciones y la comunicación entre compañeros.

- **Utilizar un recurso alterno de comunicación**

Se recomienda hacer uso de la herramienta denominada SKYPE (ver ANEXO C: Herramienta sugerida: SKYPE), como medio de comunicación sincrónica remota basada en Chat, y una manera de apoyarse en el uso de otras herramientas que usadas en conjunto con la herramienta presentada en este trabajo, complementan los servicios de comunicación y colaboración. Para motivar el uso de esta herramienta se solicita a cada usuario que se registre en el sistema proporcionar un login de SKYPE que facilite y motive la utilización de esta herramienta de comunicación.

- **Interactuar con los compañeros**

Se aconseja programar reuniones y eventos periódicos para incrementar las relaciones interpersonales de todos los miembros del equipo de trabajo, de tal manera que la cooperación remota se facilite gracias a los lazos de confianza creados. Además se puede llegar a conocer personas que posean altos conocimientos en un tema determinado, que permitan reforzar el conocimiento propio y adquirir nuevo conocimiento para su utilización en el trabajo a realizar. Es recomendable que la programación de estos eventos se realice haciendo uso de la herramienta con el servicio de calendario de eventos, el cual permite registrar eventos para ser publicados y conocidos por los miembros del equipo de trabajo, para que participen de manera eficaz en los mismos.

- **Publicar el horario de trabajo**

Hacer uso del servicio de horario laboral que ofrece la herramienta software, el cual permite registrar los días y las horas destinadas para la dedicación al proyecto y ser publicadas a la comunidad del proyecto, de tal manera que pueda ser consultado por los demás integrantes para obtener apropiada coordinación entre los miembros del equipo, que les permita planear franjas de dedicación conjunta al proyecto.

- **Planear reuniones**

La herramienta puede ser utilizada para planear y organizar reuniones entre los miembros de la comunidad del proyecto, haciendo uso de la pizarra compartida de proyecto, de actividad y de rol que permiten publicar mensajes públicos, ó utilizar el envío de mensajes privados entre compañeros. En el mensaje es conveniente incluir, por lo menos, la información de fecha, hora, tipo de reunión (presencial o virtual), lugar (si es presencial) ó medio de comunicación a utilizar (si es virtual) y descripción de la reunión, con el fin de transmitir un mensaje completo. Al concretar estas reuniones cada miembro puede utilizar la agenda personal para registrarlas.

- **Programar reuniones en la agenda personal**

Al planear reuniones en comunidad y concretarlas, es conveniente que cada usuario interesado en la reunión, registre este evento en la agenda personal propia, como mecanismo de organización y consulta de los eventos que tenga planeados.

- **Prestar ayuda cuando alguien la necesite**

Es recomendable no negar ayuda a nadie que la este solicitando especialmente si es desde una ubicación remota, pues probablemente la persona que solicita la ayuda enfrenta una dificultad en su trabajo que le es de imperiosa necesidad poder resolver y usted puede ayudarle a resolver el conflicto. El colaborar hace parte del espíritu de cooperación que debe rondar entre los miembros de la comunidad, con el fin de conseguir el éxito del proyecto en estas condiciones.

- **Administrar equipos de trabajo**

Las personas que cumplen el papel de Jefe de Equipo necesitan utilizar estrategias para manejar a los miembros que hacen parte del equipo. Se recomienda solicitar diaria o semanalmente reportes de trabajo a todos los miembros del equipo ya sean locales o remotos y proporcionarles realimentación frecuente para crearles un sentimiento de conexión y pertenencia.

- **Brindar de manera completa los datos personales**

Procure brindar de manera completa, y mantener actualizada, la información referente a sus datos personales para que los compañeros del proyecto puedan conocer detalladamente su perfil y los datos de contacto para ubicarlo de manera eficaz.

- **Utilizar la pizarra de mensajes**

Se recomienda hacer uso de la pizarra de mensajes, integrada en la herramienta, como mecanismo de comunicación y socialización entre compañeros de proyecto, de rol y de actividad. Su aporte es muy importante. En la pizarra de mensajes se publican mensajes públicos, ideas y comentarios, para ser leídos por todos los miembros interesados.

- **Usar el envío de mensajes privados a un compañero**

El usuario tiene la posibilidad de enviar mensajes privados a un compañero, como alternativa a los mensajes públicos de la pizarra de mensajes. Cuenta con dos opciones de envío: enviar un mensaje privado que el destinatario podrá leer en la herramienta software (Sección de “Mis mensajes”) ó enviar un mensaje que será redactado en la herramienta pero enviado al correo electrónico del compañero. También se pueden utilizar las dos opciones de envío a la vez, esperando que el usuario remitente tenga acceso rápido al mensaje enviado. Se debe tener en cuenta que para la redacción de mensajes existen reglas que muestran la etiqueta en la red, conocidas con el término de Netiquette (la etiqueta en la red). Algunas reglas para aplicar a los mensajes son [14]:

  - No use mayúsculas. Esto equivale a gritar. Los mensajes con letras mayúsculas son difíciles de leer y son muy irritantes.
  - Trate de no escribir líneas que contengan más de 80 caracteres. Algunas personas que usan otros programas pueden tener problemas para leerlos.
  - En lo posible no use símbolos raros o poco convencionales. Pueden ser mal “traducidos” por otra computadora.
  - No se exprese en forma denigrante, sarcástica o burlona de ninguna persona u organización.

- **Controlar el estado de las actividades**

Se recomienda, al usuario que cumple el papel de Jefe de Rol, dentro de una iteración que se ejecuta en una fase del proyecto, usar y valorar la información que la herramienta le presenta al consultar las actividades que debe administrar, la cual le indica el estado en que se encuentra cada actividad y la cantidad de usuarios responsables de la actividad. De esta manera puede poner énfasis en las actividades que tienen estado de “No Terminada” para enviar un mensaje de alerta a los responsables, tratando de controlar el tiempo planeado y la calidad de los resultados.

- **Utilizar los informes de avance del proyecto**

Se aconseja que el Director del proyecto se encargue de revisar de manera periódica el informe global del proyecto que le brinda la herramienta, donde se le calcula de forma automática el grado de avance del proyecto. Este informe puede ser utilizado para realizar funciones de seguimiento y control y si es el caso, realizar modificaciones en el plan del proyecto.

En el siguiente capitulo se describen las conclusiones, recomendaciones y trabajo futuro deducidos por los autores del presente documento, durante y al final del desarrollo del proyecto.

CAPITULO 8: CONCLUSIONES, RECOMENDACIONES Y TRABAJO FUTURO

En este capitulo se detallan las conclusiones, trabajo futuro y recomendaciones, obtenidas con la realización de la herramienta y en general con el trabajo de grado.

8.1 CONCLUSIONES

- En este trabajo de grado se desarrolló un soporte informático para el modelo de proceso distribuido diseñado en la tesis de maestría e ideas propias. Este soporte informático está orientado a facilitar la operacionalización de la logística de la comunidad (I+D) que la tesis de maestría pretende integrar a partir de grupos de investigación (I+D) dispersos geográficamente, pero interesados en una temática común que es el modelado y simulación.

- Como resultado del trabajo de grado se logró construir una herramienta software que soporta un proceso distribuido de desarrollo utilizado por una comunidad (I+D), de la siguiente manera:
    - Se implementó la funcionalidad que apoya las responsabilidades del proceso de software, relacionadas con la gestión del ciclo de vida, trabajadores y documentación, encapsulada, de manera lógica, en el módulo de Gestión del Proceso.
    - Se implementó la funcionalidad que apoya las actividades del proceso relacionadas con la comunicación y la sociabilidad de los equipos de desarrollo distribuido (programación de reuniones, asignación, seguimiento y entrega de tareas, publicación de calendarios de eventos e informes del avance del proyecto entre otros), encapsulada, de manera lógica, en el módulo de Comunicación y Sociabilidad de la comunidad (I+D).

- La herramienta construida propicia las condiciones para favorecer una cultura de cooperación y colaboración entre investigadores geográficamente dispersos, a nivel nacional, que deseen llevar a cabo un proyecto software.

- El desafío real comienza ahora, formar la comunidad (I+D) a partir de equipos de investigación distantes geográficamente, a nivel nacional, y mantenerla unida, usando la herramienta construida y fomentando el trabajo en equipo.

- El proceso distribuido de desarrollo plantea unos principios y buenas prácticas, adecuados para el desarrollo en comunidad, apropiadamente armonizados y pensados para facilitar esa situación, si las personas los aplican. La herramienta facilita que los principios y buenas prácticas sugeridas por el proceso sean realizables a través de su uso y además se recomiendan otras prácticas para poner en ejecución en comunidad y usando la herramienta. Estas prácticas se aconsejan con el fin de que sean valoradas y promovidas a través de la comunidad con enseñanza, capacitación, motivación, políticas y estrategias, que permitan que los principios encuentren su curso y finalmente florezcan en la comunidad.

- El comportamiento colaborativo de las personas no es innato, es un comportamiento que se aprende y valora cuando se dan trabajos en equipo siendo un proceso lento en donde se concientiza a las personas a trabajar con los demás con la ayuda que estos les puedan brindar. Las herramientas que involucren colaboración tienen que manejar características importantes en donde se debe buscar la participación de los integrantes, el compartir ideas y que el conocimiento de un equipo de personas sea construido colectivamente facilitando así el desarrollo de habilidades personales y sociales permitiendo el trabajo en equipo con mecanismos que provean y faciliten la comunicación y colaboración entre los miembros del grupo para poder interactuar.

- Los resultados del trabajo de investigación y desarrollo realizado en este proyecto, resultan de gran utilidad para dos grupos de investigación y una tesis de maestría:

    - **Grupo SIMON.** Para el grupo SIMON se pone a disposición una herramienta que soporta informáticamente un proceso que lidia con los aspectos distribuidos de la comunidad, para que puedan liderar un proyecto de desarrollo de un Entorno Software de Modelamiento y Simulación (ESMS), el cuál agrega a una red de investigadores[21] pertenecientes a varias universidades colombianas[22].
    - **Grupo GTI.** El resultado obtenido en este proyecto hace parte de la presentación de resultados del proyecto SIMEP-SW del grupo GTI. Con el fin de afrontar la nueva tendencia relativa al desarrollo de software, el paradigma de desarrollo distribuido, este trabajo de grado intenta constituirse en un avance hacia el futuro en la forma como las organizaciones

[21] En dinámica de sistemas y otras áreas del modelamiento – simulación.
[22] Universidad del Magdalena, Universidad del Cauca, Universidad Nacional y EAFIT.

de desarrollo de software en Colombia, enfrentarán el desarrollo distribuido basado en comunidades virtuales de (I+D).

- **Trabajo de maestría.** Los resultados obtenidos en este trabajo de grado, contribuyen con el desarrollo del soporte informático para el modelo de proceso distribuido diseñado en la tesis de maestría. Este soporte informático permitirá la operacionalización de la logística de la comunidad (I+D) que la tesis de maestría pretende integrar a partir de grupos de investigación (I+D) dispersos geográficamente, pero interesados en una temática común que es el modelado y simulación.

- La experiencia obtenida al desarrollar el trabajo de grado, ha mostrado que cuando se está iniciando el desarrollo de este tipo de proyectos es importante, empezar a modelar los temas del negocio de los que se tiene mayor información y comprensión. Además, una vez identificadas las primeras funcionalidades, es aconsejable iniciar con un modelado básico, sin pretender de una vez vislumbrar e incluir todos los casos posibles de funcionalidad que soporta la herramienta. Estos se pueden incluir en una próxima versión que rápidamente provoca un incremento y mejora en el modelado, lo que llega a constituir un desarrollo iterativo e incremental.

- En la elaboración de este proyecto se ha realizado una exploración de una nueva estrategia de desarrollo de software, como lo es el paradigma de desarrollo distribuido, buscando constituirse en un soporte que permita poner en ejecución iniciativas de desarrollo de proyectos de software entre redes de investigadores geográficamente dispersas, a nivel nacional. De igual manera, se abre paso para futuros esfuerzos en este nuevo campo. Los futuros investigadores cuentan desde ya con una base teórica y tecnológica para seguir madurando y aportando a la evolución de la industria del software colombiana.

- El Proceso Unificado (UP) fue tomado como inspiración, para la construcción de la herramienta, debido a que es un proceso en el cual tanto sus prácticas, principios y estructura, no desconocen la naturaleza evolutiva del software cuando se desarrolla, además es un proceso que puede personalizarse para diferentes sistemas de software, áreas de aplicación, niveles de aptitud y diferentes tamaños de proyectos. Para realizar el modelado de UP se realizó un estudio de los elementos de su estructura (características y función) y las relaciones entre ellos, para obtener un modelo base, el cual fue ampliado con los elementos que soportan la gestión de proyectos y la comunicación entre miembros de la comunidad de desarrollo. No se hace uso del metamodelo SPEM, por la complejidad que implica este metamodelo y además porque la herramienta no se orienta a la definición de procesos de software si no a la ejecución de proyectos de software.

- La herramienta, en su versión actual, pone a disposición dos especializaciones del Proceso Unificado de Rational (RUP) [13], como proceso a utilizar por un proyecto software que sea definido en la herramienta. El usuario encargado, al seleccionar este proceso, lo personaliza, obteniendo una plantilla o plan que guiará el desarrollo del proyecto.

- El diseño arquitectónico de implementación utilizado en la herramienta, se caracteriza por la definición de tres capas, con funcionalidad claramente definida, garantizando escalabilidad, y facilitando el mantenimiento y futuras reestructuraciones del software. Una capa se utiliza para aspectos de interfaz, la siguiente para implementar la lógica del negocio (dominio del problema) y la última, para implementar la lógica de servicio (acceso a datos).

- Normalmente Internet ha sido considerada como una gran base de datos de cobertura global más que como un lugar en donde poder llevar a cabo un trabajo concreto. De esta manera, herramientas clásicas como el correo electrónico y los foros no son capaces de ofrecer un mayor soporte para la realización de proyectos de software en comunidad a través de la red. Por este motivo la herramienta obtenida en este trabajo de grado se presenta como una aplicación orientada a facilitar el desarrollo y gestión de proyectos de software entre usuarios geográficamente distantes ofreciendo estrategias de comunicación y socialización que promuevan el auge de una comunidad I+D.

- La herramienta generada busca constituirse en un precedente para futuros esfuerzos distribuidos de I+D, no solo en el área del desarrollo de software orientado hacia el Modelamiento y Simulación, sino también en aquellas áreas, donde un proceso distribuido de desarrollo se considere como una opción seria para equipos geográficamente distantes o comunidades que emprenden un proyecto en común.

- Aunque el desarrollo de aplicaciones con .NET tiene un modelo común de programación gracias a la librería de clases base y el entorno de programación es el mismo para cualquier tipo de aplicación, pensar que es fácil construir una aplicación Web, teniendo experiencia en otro tipo de aplicación es una concepción errada, puesto que el desarrollo de aplicaciones Web implica mayor complejidad y el conocimiento de conceptos más avanzados. Sin embargo, en Internet se pueden encontrar muchos sitios Web con información útil para el desarrollo de este tipo de aplicaciones en la tecnología de .NET, que permiten realizar autoaprendizaje.

- La construcción de la herramienta ha permitido satisfacer el interés personal por aprender la tecnología .NET, orientada al desarrollo de aplicaciones Web. Sin embargo, el conocimiento adquirido es muy básico y se

continúa con el interés por seguir estudiando los conceptos involucrados en el desarrollo de este tipo de aplicación.

8.2 TRABAJO FUTURO

El trabajo de acuerdo al alcance planeado está terminado. Sin embargo, durante sus fases de desarrollo surgen nuevos interrogantes los cuales se han decidido abordar como trabajo futuro, aspectos que servirán como base a otras personas que tuvieran la intención de llevarlo a cabo. A continuación se describen los aspectos propuestos como trabajo a futuro:

- Se plantea la realización de un caso de estudio de la aplicación de la herramienta con una comunidad de desarrolladores, que permita observar la funcionalidad requerida por la herramienta, ver resultados de la misma en un entorno en el cual los integrantes se encuentran dispersos geográficamente y recopilar información de los usuarios que usan la herramienta. Existen proyectos llevados a cabo por grupos de investigación de la Universidad del Cauca, cuya dinámica involucra el trabajo en equipos que se encuentran dispersos geográficamente, los cuales serian usuarios potenciales para probar la herramienta y sugerir mejoras, por ejemplo el proyecto Link-All [15] y el proyecto Ehas [16].

- Se sugiere adaptar o ampliar la funcionalidad de la herramienta, de tal manera que sea posible gestionar equipos de trabajo fuera de un proyecto específico y que un equipo pueda ser asociado a varios proyectos, ya que actualmente la herramienta permite crear equipos de trabajo en un proyecto en particular, sin la opción de que un mismo equipo trabaje en varios proyectos.

Para el módulo de Gestión del Proceso Distribuido se plantea:

- Incrementar la funcionalidad con un servicio que permita la utilización de Diagramas de Gantt para el control de las actividades.

- Proponer como trabajo para futuros investigadores, la funcionalidad para definir procesos de desarrollo de software, para que el usuario cuente con diferentes opciones de procesos, seleccione el deseado, lo personalice y lo use como plantilla para un proyecto de software definido en la herramienta.

- Incluir la funcionalidad para controlar los costos del proyecto, que permita dar seguimiento al proyecto en el sentido de planeación y ruta crítica.

Para el módulo de Comunicación y Sociabilidad de la Comunidad (I+D) se plantea:

- Incluir la funcionalidad para interacciones de audio y video entre usuarios.

- Integrar el servicio de Chat que soporte la comunicación sincrónica de los miembros de la comunidad.

- Incrementar la funcionalidad de los servicios de Calendario y Programación de Reuniones incluyendo el soporte para coordinar la diferencia horaria, en el caso en que los usuarios se encuentren en ubicaciones geográficas con grandes diferencias horarias.

- Incrementar la funcionalidad del servicio de Programación de Reuniones para que permita definir actividades periódicas y el envío de avisos de una próxima reunión a los interesados.

8.3 RECOMENDACIONES

- Para una versión futura de la herramienta se recomienda hacer uso de SPEM [12], Metamodelo para la Ingeniería de Procesos de Software (Software Process Engineering Metamodel - SPEM), para modelar los procesos de software que vayan a ser utilizados por los usuario para personalizar y utilizar en proyectos definidos en la herramienta, utilizando la metodología estándar base ofrecida por SPEM.

CAPITULO 9: GLOSARIO Y BIBLIOGRAFIA

En este capítulo se describe el glosario del presente trabajo y la bibliografía utilizada.

9.1 GLOSARIO

- **Actividad:** Representa la ejecución de una operación por un rol. Toda actividad está constituida por pasos.

- **Artefacto:** Es la instancia de un tipo de artefacto, que es asignado a un usuario para generar versiones del mismo. Ejemplos de artefactos son: caso de uso comprar, caso de uso vender, etc. De un tipo de artefacto pueden resultar muchos artefactos, pero un artefacto es de un solo tipo de artefacto.

- **Disciplina:** Es una colección de actividades relacionadas.

- **Equipo (I+D):** Es un grupo de trabajadores dentro de un proyecto. Las personas se pueden agrupar por su cercanía geográfica. El equipo (I+D) se define como un conjunto pequeño de personas con habilidades complementarias, que están comprometidas en un propósito, objetivos de rendimiento y enfoque comunes en el que todos y cada uno de los miembros del equipo es responsable ante los demás. La motivación fundamental de estas personas consiste en la investigación y desarrollo del software.

- **Fase:** Es una etapa del proceso o un periodo de tiempo entre dos hitos principales de un proceso de desarrollo. El proceso de desarrollo de software se divide en fases.

- **Hito de Fase:** Representa un punto de sincronización en los que coinciden una serie de objetivos bien definidos. Cada fase acaba en un hito principal.

- **Iteración:** Es un conjunto de actividades llevadas a cabo de acuerdo a un plan que lleva a producir una versión de un producto. Cada fase del proyecto está constituido por una o más iteraciones.

- **Mensaje:** Es un escrito, nota o comentario que se puede publicar en la herramienta o enviar entre compañeros. Los mensajes pueden ser públicos y privados.

- **Proceso:** Es el conjunto de actividades necesarias para transformar los requisitos de un usuario en un sistema software. Un proceso es una plantilla para crear proyectos.

- **Proyecto:** Un proyecto se define como el esfuerzo temporal que se realiza para crear un producto o servicio único. El término temporal de la definición, se refiere a que cada proyecto tiene una fecha de inicio y una fecha de término, y, el concepto único, se refiere a que el producto o servicio contienen elementos o rasgos que los distinguen de los productos o servicios ya existentes. Un proyecto es ejecutado por usuarios, planeado, ejecutado y controlado.

- **Requerimiento:** consiste en una condición o capacidad que debe satisfacer un sistema.

- **Rol:** Es el papel que un individuo puede desempeñar en el desarrollo de software, como puede ser un especificador de casos de uso, un arquitecto, o un ingeniero de componentes. Cada rol es responsable de un conjunto de actividades. Cada proceso define uno o varios roles.

- **Tipo de Artefacto:** Es un tipo de artefacto software, ó tipo de producto obtenido en el desarrollo de software, del cual pueden resultar muchas versiones. Ejemplos de tipos de artefactos son los casos de uso, el modelo conceptual, diagramas de interacción, etc. En una actividad se utilizan y se generan diversos tipos de artefactos de software.

- **Usuario:** Es la persona registrada en el sistema, que puede ser parte de un equipo de trabajo dentro de un proyecto.

- **Versión:** Es un archivo que representa la actualización de un artefacto.

9.2 BIBLIOGRAFIA

Artículos y Libros

AMBLER, Scott. “Bridging the distance”. SoftwareDevelopment. http://www.sdmagazine.com/documents/s=826/sdm0209i/ . Septiembre 2002.

ANTILLANCA, Héctor. FULLER, David. “Sistemas colaborativos sincrónicos cara-a-cara: requisitos, problemas y resultados”. Universidad de Santiago de Chile. www.diinf.usach.cl/webdiinf/ArchivosSubidos%5C17620041627143erECHC%2095%20HAE.pdf. Junio de 2004.

CRUZ, David. “¿Cómo administrar proyectos de Ingeniería de Software?”. Octubre de 2004. http://www.informatizate.net/articulos/como_administrar_proyectos_de_ingenieria_de_software_05102004.html.

DESANCTIS, Gerardine. MONGE, Peter. “Introduction to the special issue: Communication Processes for Virtual Organizations”. Junio 1998. http://www.ascusc.org/jcmc/vol3/issue4/desanctis.html.

ELLIS, Clarence. GIBBS, Simon. REIN, Gail. "Groupware. Some issues and experiences". Comm. of the ACM, Vol. 34, No. 1 (1991), 38-58.

GALBRAITH, J. R. Designing Organizations. Jossey-Bass, San Francisco, CA. ISBN: 0-7879-5745-3. Año 1995.

JACOBSON, Ivar. BOOCH, Grady. RUMBAUGH, James. “El Proceso Unificado de Desarrollo de Software”. Addison Wesley. Madrid. 2000. ISBN: 8478290362.

KIRCHER, Michael. JAIN, Prashant. CORSARO, Angelo. LEVINE, David. “Distributed Extreme Programming”, DXP. 2001. Italia. Mayo 21-23, 2001.

LARMAN, Craig. “UML y Patrones: Una introducción al Análisis y Diseño Orientado a Objetos y al Proceso Unificado” (2da Edición). Editorial Prentice Hall. ISBN: 84-205-3438-2. Año 2003.

MARQUES, Helena. RAMOS, Rodrigo. SILVA Ismenia. “Adaptacão de um Processo de Desenvolvimento para Fábricas de Software Distribuídas”. Universidade Federal de Pernambuco. Brasil. http://www.spc.org.pe/ideas2004/.

MCCONNELL, Steve. “Desarrollo y gestión de proyectos informáticos”. Editorial McGraw-Hill, 1997. Madrid, España. ISBN: 84-481-1229-6.

ORTEGA, Manuel. “Trabajo cooperativo con ordenador”. Universidad de Castilla, La Mancha. Diciembre de 2001. http://griho.udl.es/ipo/doc/13Cooper.doc.

OSOY, Omar. “Organizaciones Virtuales”. Año 2005. http://www.monografias.com/trabajos14/organiz-virtual/organiz-virtual.shtml.

PREECE, Jenifer. “Online Communities: Design Usability and Suporting Sociability”. Editorial WILEY. 2003. ISBN: 0-471-80599-8.

PRESSMAN Roger. Ingeniería del Software “Un Enfoque Práctico” 5 Ed. Editorial Mc Graw Hill. ISBN: 0-07-052182-4.

RING, P. S., VAN de Ven. “Developmental Processes of cooperative interorganizational relationships". Academy of Management Review. http://www.aom.pace.edu/amr/. 1994.

SOSA, Mabel. ZARCO, Raquel. POSTIGLIONI, Analia. “Modelando aspectos de grupo en entornos colaborativos para proyectos de investigación”. Departamento de Informática. Facultad de Ciencias Exactas y Tecnologías. Universidad Nacional de Santiago del Estero, Argentina. Junio de 2003. http://www.fi.uba.ar/laboratorios/lie/Revista/Articulos/030307/A3Jun2006.pdf.

ZANONI, Roberto. AUDY, Jorge. “Project Management Model: Proposal for Performance in a Physically Distributed Software Development Environment”. Engineering Management Journal. Vol. 16 No. 2. Junio de 2004.

Instituciones, Proyectos, Grupos y Personas

[1] Grupo de Investigación en Tecnologías de la Información – GTI perteneciente al departamento de sistemas de la Facultad de Electrónica de la Universidad del Cauca. http://gti.unicauca.edu.co.

[2] Grupo SIMON de modelos y simulación de la Universidad Industrial de Santander. http://www.uis.edu.co/site/investigacion/grupos/simon/index.html.

[3] Sistema Integral de Mejoramiento de Procesos de Software - SIMEP-SW, Proyecto del grupo GTI. http://groups.yahoo.com/group/simep-sw/

[4] THE YANKEE GROUP. “Creating and Delivering Value with Collaborative Software Development Tools”. http://www.yankeegroup.com .Septiembre 2003.

[5] SYSTEMS AND SOFTWARE CONSORTIUM. “Software productivity Consortium”. http://www.software.org/pub.

[6] IBM. “Optimize Geographically Distributed Development with IBM Rational”. http://www.thesoftwareinstitute.org/flashProgramPage/?registered=1_cacronym=IGDC&sectionName=Rational.

[7] MORENO CHAUSTRE, Jorge Jair. Proyecto de maestría “Diseño de una arquitectura para un entorno de modelamiento-simulación y creación de un proceso para su desarrollo por una comunidad (I+D)”. Docente del departamento de Sistemas de la FIET.

[8] PMBOK Guide. “A Guide to the Project Management Body of Knowledge“. Project Management Institute. 2000 Edition.

[9] IAAP, Instituto Argentino de Administración de Proyectos. http://www.iaap.com.ar/.

[10] Programa desarrollador cinco estrellas. “Introducción al entorno de desarrollo .NET”. http://www.microsoft.com/spanish/msdn/comunidad/dce.

[11] GENESIS. “Generalized Environment for Process Management In Cooperative Software Engineering”. http://www.genesis-ist.org/spanish/description.htm. Septiembre 2001.

[12] SPEM. Software Process Engineering Metamodel. Metamodelo usado para describir procesos de desarrollo de software. http://www.omg.org/technology/documents/formal/spem.htm

[13] RUP. Rational Unified Process. Version 2003.06.00.65. Copyright 1987-2003. Rational Software Corporation. All rights reserved.

[14] GAIASUR, Infoteca. Año 2002. http://planeta.gaiasur.com.ar/infoteca/varios/netiquette.html.

[15] LINK-ALL. Red de Inserción de Comunidades Remotas de América Latina. Grupo de Ingeniería Telemática de la Universidad del Cauca. http://git.ucauca.edu.co/link-all.

[16] EHAS. Enlace Hispanoamericano de Salud. Grupo de Ingeniería Telemática de la Universidad del Cauca. http://git.ucauca.edu.co/ehas.