Universidad de las Ciencias Informáticas

Facultad 2

Tesis: Menú Framework J2ME

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

Autores: Camilo Roberto Legón Sarmiento.
Yuniel Bermudez Diaz.

Tutores: Ing. Arian Zulueta Casal.
Ing. Yunisel Viera Vargas**.**

Co-tutores: Lic. Mijaíl del Toro Céspedes.
Ing. York Figueroa Valdés.

Ciudad de la Habana, junio de 2008.

DECLARACIÓN DE AUTORÍA

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

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

_____	_____
Camilo Roberto Legón Sarmiento **(Autor)**	Arian Zulueta Casal **(Tutor)**
_____	_____
Yuniel Bermudez Diaz **(Autor)**	Yunisel Viera Vargas **(Tutor)**

El hombre debe ser eternamente útil, no eternamente joven. En la utilidad está la juventud.

Goette

AGRADECIMIENTOS

A nuestros tutores, a todo el colectivo de Procyon y a la UCI.
A York por su ayuda, interés y entusiasmo.

Camilo y Yuniel

A mis padres por su apoyo, dedicación, amor y sobre todo por guiarme en este largo camino.
A mis hermanos por su cariño y comprensión.
A mi novia Lisandra (bebé) por su pasión y ternura.
A mi familia por obligarme a ser mejor persona.
A todos mis amigos y compañeros de estudio por su compañía y los momentos vividos.
A mi compañero de tesis por su amistad y su ayuda para lograr este sueño.
A todas esas personas que de una forma u otra me ayudaron a lograr esto.

Camilo

A mi madre la mejor del mundo, que me lo ha dado todo y que es y será siempre la luz al final del camino, la guía sin la que hoy no estuviera aquí.
A mi abuelita del alma, fuente inagotable de experiencia y amor y que ha hecho de todo por mi educación.
A mi hermanita que es la que me hace sentir más maduro y responsable.
Al loco Taño que lo arregla y lo resuelve todo
A mi padre y Milda, por su entusiasmo y comprensión.
A toda mi familia, en la que todos ponemos algo nuestro en los demás y que me han brindado muchísimo apoyo y todo el cariño del mundo.
A mis compañeros de clase que nos hicimos hombres y mujeres juntos en las buenas y las malas, por el apoyo y las experiencias que me dieron.
A mis amigos, los de siempre y los de ahora, los de aquí y los de allá, de todos he aprendido algo.
A mis profes, los forjadores de este sueño.
A mi compañero de tesis por su amistad y ayuda.

Yuniel

DEDICATORIA

A mi madre (Mamichuli) por ser el motivo y sentido de este sueño, por su amor incondicional.

A mi padre (Puro) por ser ejemplo a seguir y sacarme siempre ileso de tormentas.

A mi hermanita Celi por ser tan especial.

A mi hermana Katy por apoyarme siempre y darme una de las alegrías más grande de mi vida, mis sobrinos (Rauli y Papo).

A mi hermano Ernesto por creer en mí y darme a mi sobrino (Samurai).

A Lisandra (bebé) por estar siempre a mi lado y confiar en mí.

A mis amigos del alma.

Camilo

A mami, mama y tata que se merecen más que todo lo que les pueda dar.

Yuniel

A la UCI que vimos crecer…

RESUMEN

Los dispositivos móviles se han convertido en un artículo personal indispensable en la actualidad por la alta infraestructura de comunicación y cobertura que posee la telefonía celular. Los contenidos desarrollados para los mismos por ende gozan de la misma popularidad y demanda.

La entidad cubana PROCYON dedica parte de su producción al desarrollo de aplicaciones J2ME para dispositivos móviles, un elemento fundamental en estos productos es la creación de sistemas de menú. Éstos son implementados por cada desarrollador lo que provoca un aumento en el tiempo de desarrollo y en la complejidad de las aplicaciones que se producen. La carencia de un estándar de implementación para la empresa incide en la diferencia de calidad entre los productos. Existen soluciones de software en el mercado actual que se ajustan a dicha situación pero están sujetas a licencias, que en ocasiones son excesivamente costosas.

En este trabajo se presenta la implementación de un *API GUI* en forma de un paquete de clases para interfaz visual con el objetivo de facilitar la creación de componentes *GUI* para los desarrolladores de aplicaciones *J2ME*. Este framework brinda soporte a toda la gama de dispositivos móviles que soportan *Java* y facilita la disminución aproximadamente de un 80% del tiempo de desarrollo de sistemas de menú así como la complejidad de las aplicaciones que se desarrollan. También proporciona la utilización de nuevos componentes visuales para interfaces gráficas con respecto a otras plataformas existentes en la actualidad y proponen aspectos novedosos con respecto a otros lenguajes de programación. Plantea un estándar de implementación en entidades y empresas que se dedican a la producción de contenidos para la telefonía celular.

Índice

Índice de Tablas

Introducción:

La telefonía celular actualmente está a la avanzada tanto en el campo tecnológico como financiero debido a que es un producto muy popular entre todos los sectores de la población por sus múltiples utilidades, accesibilidad y disponibilidad en cuanto a comunicación. Los dispositivos móviles se han convertido en un artículo personal indispensable en esta era digital y por completo tecnológica debido a que ofrecen una alta gama de funcionalidades que tienen impacto, que va desde las relaciones sociales hasta el ámbito empresarial. En los mismos se puede encontrar desde una simple agenda de direcciones y teléfonos hasta una sala chat que permita la conexión desde cualquier parte del mundo. El acceso a internet no podría faltar en estos dispositivos por las características del mundo actual, lo que posibilita que un ama de casa pueda conseguir una receta de cocina, o un funcionario cerrar un importante negocio en línea. Esto está relacionado con la alta infraestructura de comunicación y cobertura que posee dicha tecnología en el mundo actual que lo hace un método factible y seguro de comunicación.

Al cierre del año 2007 se registraron unos 3,31 millones de suscripciones celulares con un crecimiento de 40 millones de nuevas conexiones cada mes distribuidos por todo el planeta (REYES, 08), considerando estos datos y unido a lo anteriormente planteado se puede decir que los móviles o teléfonos celulares constituyen el dispositivo tecnológico más popular e importante en este naciente siglo.

Los contenidos desarrollados para los mismos por ende gozan de la misma popularidad, estos tienen un amplio espectro que abarcan desde las aplicaciones indispensables hasta el ocio. La producción de juegos es una de las más importantes áreas de este tipo de aplicaciones. Lo que es deducible por las ganancias de grandes compañías que se dedican a esta rama de la informática como Sony Ericsson, Motorola, Nokia, Siemens entre muchas otras, que han destinado un considerable número de esfuerzo y recursos al desarrollo de los mismos. PROCYON, entidad pionera en este ámbito en nuestro país, la cual se encuentra en la Universidad de las Ciencias Informáticas (UCI) en la infraestructura productiva, fomenta el desarrollo de contenidos para dispositivos móviles. Los juegos de todo tipo constituyen una parte significativa en esta producción en cuanto a demanda y resultados por parte de clientes, lo que significa que el desarrollo de herramientas que faciliten su creación seria de máxima importancia, máxime cuando el mercado para este tipo de aplicación es enorme, fundamentalmente en el continente americano y el área latinoamericana (Telecom, 2005).

El desarrollo de un API (del inglés Application Program Interface) GUI (del inglés Graphical User Interface) para dispositivos móviles brindaría una infinidad de utilidades dado que el API incorporado por J2ME (inglés Java 2 Micro Edition) es básico y en ocasiones primitivo. Para el desarrollo de este API se ha planteado la siguiente situación problémica.

Situación problémica:

Parte de la producción de la empresa PROCYON consiste en la implementación de juegos para teléfonos móviles, todos estos juegos tienen algunos elementos comunes como son los sistemas de menú, hasta ahora cada desarrollador implementa su propio paquete de clases para desarrollar dicho sistema, esto hace que aumente el tiempo de desarrollo y la complejidad de las aplicaciones que se desarrollan. Esta situación también implica la carencia de un estándar de implementación para la empresa, lo que provoca la diferencia de calidad entre productos. La empresa actualmente no tiene cómo resolver este problema y las soluciones de software existentes en el mercado actual que se ajustan a dicha situación tienen licencia, que en ocasiones son excesivamente costosas además de estar sujetas a las leyes del bloqueo económico aplicado a Cuba por los EE.UU.

Problema:
¿Cómo establecer un estándar de implementación para los sistemas de menú usados en las diferentes aplicaciones desarrolladas en la entidad PROCYON?

Objeto de estudio:
Sistemas de menú en aplicaciones J2ME para dispositivos móviles.

Campo de acción:
Sistemas de menú en aplicaciones J2ME para dispositivos móviles en la entidad PROCYON.

Objetivo General:

1. Analizar, diseñar e implementar un API base que facilite a los programadores la creación de menú para aplicaciones en J2ME.

Objetivos Específicos:

1. Crear contenedores de componentes ajustables a distintos tipos de dispositivos.
2. Crear componentes visuales ajustables a distintos tipos de dispositivos.

Para el desarrollo del proyecto, se tuvieron en cuenta las siguientes **preguntas de investigación**:

1. ¿Cómo funcionan los contenedores de componentes?
2. ¿Qué características y funciones tienen los diferentes componentes visuales existentes en otros lenguajes de programación?

Con el propósito de darle cumplimiento a los objetivos trazados se realizaron las siguientes **tareas de la investigación**:

1. Estudio del funcionamiento de J2ME-polish.
2. Estudio de los componentes disponibles en otras plataformas actualmente.
3. Análisis de los componentes visuales que serán adaptados a la plataforma J2ME.

El presente documento está estructurado en capítulos para lograr una mayor organización de contenido y mejor entendimiento, a continuación se describen los mismos.

- ✓ **Capítulo 1: Fundamentación teórica:** Se describen los principales conceptos involucrados en la realización de la investigación, el estado del arte del tema tratado a nivel internacional, nacional y de la Universidad, así como las principales tendencias, técnicas, tecnologías, metodologías y software usados en la solución del problema.

- ✓ **Capítulo 2: Características del Sistema:** Se refiere a la propuesta del sistema. Basado fundamentalmente en, los objetivos estratégicos de la organización, el objeto de automatización, y un análisis mayormente comparativo de las soluciones existentes encontradas durante la investigación.

- ✓ **Capítulo 3: Exploración y Planificación del sistema:** Se detallan los artefactos relacionados con exploración y planificación de la aplicación.

- ✓ **Capítulo 4: Implementación y prueba:** Se describen los artefactos relacionados con la implementación y prueba de la aplicación.

- ✓ **Capítulo 5: Estudio de Factibilidad:** Detalla todo el estudio realizado para justificar la factibilidad del sistema.

Capítulo1: Fundamentación teórica

1.1. Introducción:

A continuación se expone de forma explicativa los principales conceptos y aspectos que fueron necesarios analizar para la realización de la presente investigación. Se propone además un acercamiento a la situación actual en la que se encuentran los APIs, plataformas y herramientas que facilitan el desarrollo de juegos en dispositivos móviles así como las principales tendencias, técnicas, metodologías y software que de una forma u otra han ayudado a dar solución a la problemática que dio origen a esta investigación.

1.2. Los juegos como contenido para móviles

El mercado de los móviles alcanza cifras sorprendentes, por lo que no es de extrañar que paralelamente a este aumento de terminales, aparezca un mercado de software y programas de entretenimiento cada vez más potente, a pesar de que algunos lo clasifiquen como de segunda división o regional preferentemente. La distribución y descarga de juegos para celulares se ha convertido en un lucrativo negocio, quizás cinco centavos por descarga puede parecer algo extremadamente barato, pero cuando se piensa en cientos de descargas por minutos las cifras recaudadas por una aplicación son considerables.

Tal es así, que en el año 2001, las principales compañías móviles anunciaron el Foro de Interoperabilidad de Juegos Móviles (MGI) que trabaja para definir una especificación de interoperabilidad de juegos móviles para los servidores basados en redes(Telecom,2001). Esto permitiría que los desarrolladores de juegos produzcan juegos móviles que puedan distribuirse a través de diferentes servidores de juegos y redes inalámbricas y puedan ser ejecutados en diferentes dispositivos móviles. Este es un ejemplo claro del término universal que ha adquirido este tipo de aplicación. Los juegos para móvil tienen que tener un tamaño muy pequeño (un juego de 1 Mb para un teléfono es algo demasiado grande) y a menudo se basan en ofrecer una maniobrabilidad sencilla y un uso racional de memoria. Esto se debe a las limitaciones de memoria física, de espacio en los teclados y la falta de potencia del procesador de los dispositivos, aunque en muchas ocasiones es la propia tecnología sobre la que se programa la que realmente limita la aplicación.

A finales de los años 1990 los teléfonos móviles todavía eran dispositivos que solo podían ser utilizados para realizar llamadas. Compañías como Nokia (pionera en el ámbito de juegos en la telefonía celular junto con Philips) decidieron ofrecer algún tipo de entretenimiento en esos pequeños dispositivos que tenían botones y una pantalla LCD (del inglés Liquid Cristal Display) en blanco y negro. Así fue como aparecieron los primeros juegos basándose en las primeras consolas de principios de la década del 80 pero jamás pensaron que esto revolucionaría el mundo de los videojuegos portátiles (Rodríguez, 2002).

Este tipo de juegos fueron evolucionando con un nivel de crecimiento totalmente insólito. Japón fue uno de los primeros países que lanzó móviles programables con tecnología Java. Hasta ese momento los juegos en los móviles estaban integrados en el teléfono y programados directamente en código máquina y grabados en la memoria ROM (del inglés Read Only Memory) del móvil, pero con los móviles programables, había una zona de memoria donde se podían grabar datos, y se podía utilizar un lenguaje de desarrollo como Java para crear aplicaciones y un cable USB o una conexión a Internet para introducir el programa en el móvil (García, 2006).

El objetivo de este tipo de teléfono era el desarrollo de pequeñas aplicaciones tipo calculadora, nota, y no videojuegos. Pero aún así algunas empresas como la francesa Gameloft[1] desarrollaron videojuegos en blanco y negro para esas pequeñas pantallas y resultó un éxito (Jenui, 2004). Los móviles fueron evolucionando y con ellos la memoria, potencia y los lenguajes de programación de los mismos, tales como, C, C++ o J2ME 2.0 entre otros. Actualmente el mercado de los videojuegos para móvil es más grande que cualquier otro mercado de videojuegos portátiles, teniendo cifras de ventas elevadas.

1.3. La utilización de APIs de desarrollo

Las APIs de desarrollo siempre han sido desde su creación una herramienta fundamental para la creación de aplicaciones en el ámbito informático de la producción de software. El mundo actual, adicto a la tecnología provoca que el desarrollo de software sea vertiginoso y para lograr este nivel de crecimiento se necesitan soluciones que faciliten la creación de los mismos. Los videojuegos o aplicaciones de entretenimiento constituyen un eslabón muy importante en esta demanda, para su realización existen numerosos APIs que facilitan el desarrollo arrojando como resultado un producto de

[1] **Gameloft:** Empresa internacional dedicada a desarrollar y editar videojuegos para dispositivos móviles. Tiene su sede central en Francia, con oficinas alrededor de todo el mundo.

alta calidad, ejemplos de éstos es el llamado DirectX[2] que es uno de los más utilizados a nivel mundial junto al OpenGL[3](del inglés Open Graphics Library). Se puede decir también que aunque DirectX fue creado por Microsoft muchos de los software libres están desarrollados sobre él (Bargen, 1998).

Para la creación de aplicaciones y juegos en dispositivos móviles basados en la plataforma J2ME, existe parte del API que posee la misma, que es aplicable a todos los dispositivos inalámbricos y una parte que es específica para celulares.

La parte de J2ME destinada a desarrollar aplicaciones para dispositivos de telefonía celular contiene un API de multimedia a partir de la cual, conociendo sus funcionalidades, posibilita trabajar con sonidos y demás elementos de este tipo. De reciente aparición se encuentra también el API 3D que permite desarrollar juegos de alta calidad para aquellos dispositivos que soportan J2ME como lenguaje de programación.

1.4. API GUI

1.4.1. Definición de API

API es un conjunto de convenciones internacionales que definen como debe invocarse una determinada función de un programa desde una aplicación (HispaNetwork, 2006). Lo que está muy relacionado con los aspectos función/funcionalidades que en un determinado módulo de software (pieza o componente) que provee a otros módulos en implementación de estándares de desarrollo. Cuando se intenta estandarizar una plataforma, se estipulan unos APIs comunes a los que deben ajustarse todos los desarrolladores de aplicaciones (Isaac, 2006). Con esto se logra una transparencia en la implementación interna de funcionalidades en módulos de software que se establezcan como herramienta APIs de desarrollo.

[2] **DirectX:** Es una colección de programas que aceleran el sistema en las tareas gráficas.

[3] **OpenGL:** Conjunto de especificaciones estándar que definen una API multilenguaje y multiplataforma para escribir aplicaciones o juegos que producen gráficos en 3D. Fue desarrollada originalmente por Silicon Graphics Incorporated (SGI).

1.4.2. Definición de GUI

La definición genérica de GUI plantea, que es un sistema de interacción entre un equipo y el usuario, caracterizado por la utilización de iconos y elementos gráficos en su concepción. Habitualmente las acciones se realizan mediante manipulación directa (estilo de interacción hombre-máquina) para facilitar la interacción del usuario (Isaac, 2003). Esto junto a la retroalimentación rápida y creciente permiten a los desarrolladores cometer menos errores y completar las tareas en menos tiempo.

1.5. APIs de desarrollo para aplicaciones en dispositivos móviles. Soluciones existentes.

Actualmente existen APIs para la creación de aplicaciones en dispositivos móviles que ya vienen integradas en plataformas de desarrollo. Ejemplo de lo anteriormente planteado es la plataforma J2ME que posee un API utilizada en la producción de aplicaciones y en particular juegos. A continuación se describe parte de dicha API que a su vez constituye un conjunto de APIs en específico, las cuales son muy utilizadas y conocidas en el ámbito de los desarrolladores.

1.5.1. MMAPI.

MMAPI (del inglés Mobile Media API), es un API que permite reproducir y grabar tanto datos de audio como de video, los paquetes generados ofrecen un número reducido de acciones relacionados con la reproducción de audio (Terminales, 2005).

1.5.2. API de multimedia.

El API multimedia que posee la plataforma J2ME define paquetes que son un subconjunto del MMAPI enteramente compatible con el API completo. El mismo constituye una interfaz flexible, simple y potente para el manejo de capacidades multimedia (Terminales, 2005).

1.5.3. API 3D

Para la realización de juegos que exigen una mayor disponibilidad gráfica, éste API proporciona una colección de constructores de alto-nivel para crear y manipular geometrías 3D así como estructuras que lo faciliten (Knudsen, 2003).

1.6. XP como metodología de desarrollo

No existe una metodología universal para hacer frente con éxito a cualquier proyecto de desarrollo de software. Toda metodología debe ser adaptada al contexto del proyecto. Recursos técnicos y humanos, tiempo de desarrollo y tipo de sistema, son algunas de las cuestiones a valorar a la hora de escoger una metodología. Históricamente, las metodologías tradicionales han intentado abordar la mayor cantidad de situaciones de contexto del proyecto, exigiendo un esfuerzo considerable para ser adaptadas, sobre todo en proyectos pequeños y con requisitos muy cambiantes (Acebal, 2005). Las metodologías ágiles ofrecen una solución casi a la medida para una gran cantidad de proyectos que tienen estas características. Una de las cualidades más destacables en una metodología ágil es su sencillez, tanto en su aprendizaje como en su aplicación, reduciéndose así los costos de implantación en un equipo de desarrollo.

XP (del inglés: Extreme Programming) es una metodología basada en valores de simplicidad, comunicación, retroalimentación. Funciona mediante la unión de todo el equipo y la aplicación de prácticas simples, con suficiente retroalimentación para permitirle al equipo ver donde están y adaptar sus prácticas a cada situación única (Beck, 2000).

1.6.1. ¿Por qué XP?

La metodología XP encaja perfectamente con el tipo de proyecto las condiciones así como la idea de desarrollo que se tiene del sistema. A continuación, las razones fundamentales que se tuvieron en cuenta al escoger esta metodología.

- ✓ **El proyecto es pequeño.** XP está concebida para ser utilizada dentro de proyectos pequeños y de desarrollo rápido se adapta perfectamente a este caso.

- ✓ **Empieza en pequeño y añade funcionalidad con retroalimentación continua.** El desarrollo del sistema comienza a partir de los requerimientos básicos y a partir de ahí se van añadiendo funcionalidades que tanto el desarrollador como el cliente entiendan necesarias.

- ✓ **Pocos roles.** Esta metodología está dirigida a grupos de desarrollo pequeños y con pocos roles como este caso.

- ✓ **El manejo del cambio se convierte en parte sustantiva del proceso.** A medida que el proyecto avanza pueden surgir nuevas expectativas o ideas que pueden ser incorporadas fácilmente permitiéndole mayor adaptabilidad al producto, con la metodología XP esto es completamente factible pues esta se adapta perfectamente a los proyectos cuyos requerimientos cambian a menudo.

- ✓ **El cliente o el usuario se convierte en miembro del equipo.** Con el uso de esta metodología y la importancia que esta le concede a la retroalimentación, el cliente es parte del equipo de desarrollo y en este caso que se desarrolla un proyecto para desarrolladores la relación es aún más fuerte.

- ✓ **Propiedad colectiva del código.** XP plantea que todos los programadores pueden realizar cambios en cualquier parte del código en cualquier momento. En el proceso de desarrollo con que cuenta la empresa esta es una práctica común.

- ✓ **Comunicación de los programadores a través del código.** XP enfatiza el uso de líneas directivas para la codificación que están bien establecidas. Desde sus comienzos la empresa cuenta con una línea directiva para la codificación.

1.7. Qué plataforma de desarrollo y lenguaje de programación fueron utilizados en el desarrollo de la aplicación.

Los dispositivos móviles están en constante desarrollo en cuanto a capacidades o capavility[4] se refiere, no obstante la programación en los lenguajes convencionales se hace imposible. De aquí la necesidad de desarrollar en aquellos lenguajes que brinden bibliotecas o paquetes de clases de tamaños reducidos. Para la creación de la solución que se propone en la presente investigación se utilizará Java como lenguaje de programación y la plataforma J2ME como API de desarrollo.

1.8. ¿Qué es J2ME y por qué utilizarlo?

J2ME es la plataforma basada en el lenguaje Java que Sun Microsystems ha creado para la programación de dispositivos inalámbricos pequeños como teléfonos celulares, y PDAs (del inglés Personal Digital Assistant) (S. Gálvez, 2003). La edición micro de Java se compone, además del lenguaje, de una máquina virtual, configuraciones, perfiles y paquetes adicionales como se muestra en la figura 2. Las configuraciones son especificaciones que detallan una máquina virtual y un conjunto base de APIs que pueden ser usadas en cierta clase de dispositivos. La maquina virtual puede ser completa, como la describe la especificación o algún derivado de ella. También utiliza derivados de componentes J2SE (del inglés Java 2 Standard Edition), como son su máquina virtual que es más pequeña y el conjunto de APIs menos potente (S. Gálvez, 2003). Por otra parte se puede plantear que es una colección de tecnologías y especificaciones diseñadas para diferentes partes del mercado de los dispositivos pequeños.

[4] **Capavility (en español prestaciones):** Los desarrolladores de aplicaciones para móviles se le llama así a las prestaciones que poseen dichos dispositivos, ya sea resolución y tamaño de pantalla, contenidos multimedia que soporta entre muchas otras capacidades o características en cuanto a tecnología se refiere.

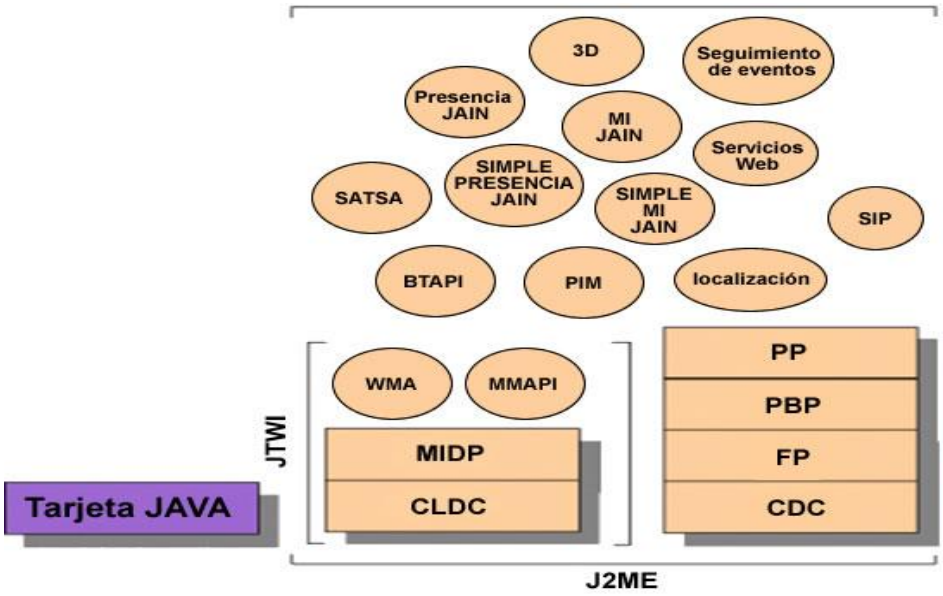


Figura #1 Componentes de J2ME

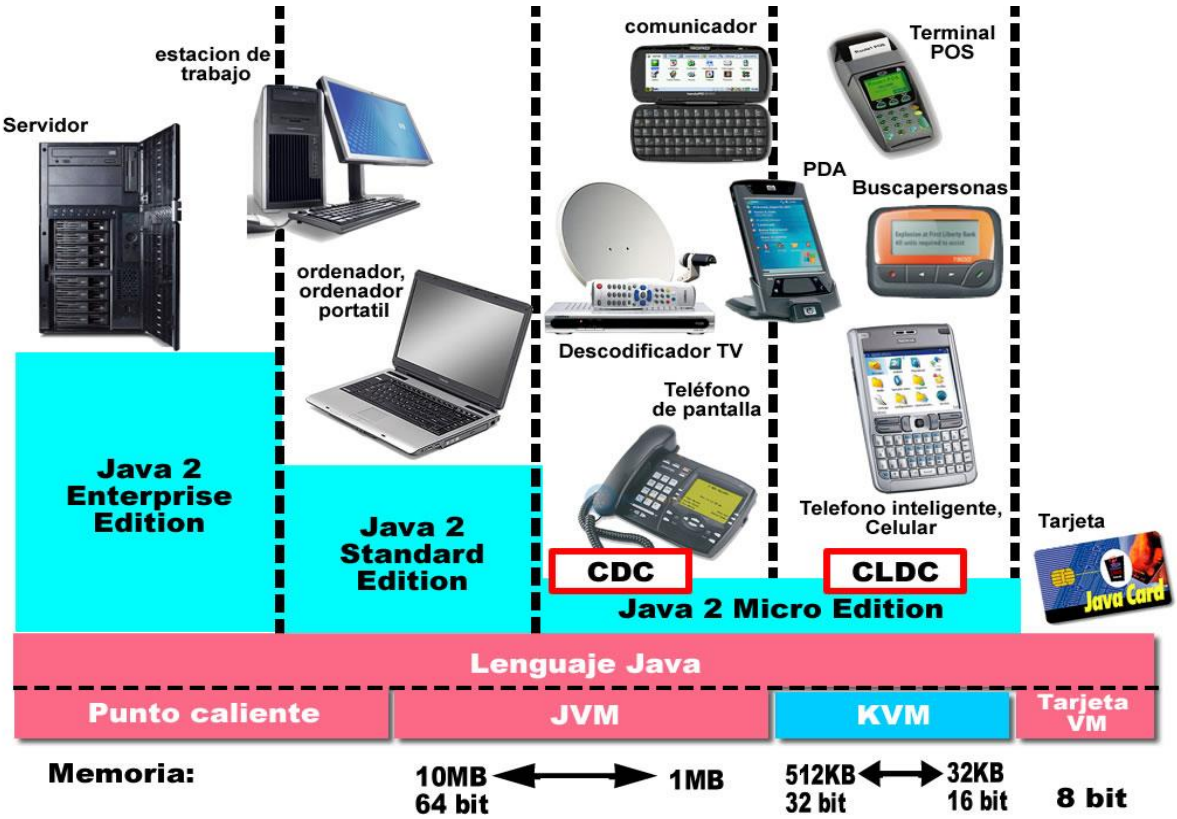


Figura #2 Plataforma Java

El perfil MIDP (del inglés Mobile Information Device Profile) que incluye el J2ME es el de mayor desarrollo en la plataforma Java, aunque se está investigando actualmente el PDA Profile. MIDP incluye APIs de interfaz de usuario, de ciclo de vida de aplicaciones y algunas de almacenamiento persistente (S. Gálvez, 2003). Hasta este momento es el único perfil aplicado a los dispositivos en el mercado. Este se combina con la configuración CLDC (del inglés Connected Limited Device Configuration) que no es más que la configuración para dispositivos con conexión limitada, para proporcionar un entorno de ejecución para dispositivos móviles, que viene integrada en el hardware de celulares relativamente modernos mediante el uso de applets[5] en los mismos, tales como juegos y aplicaciones.

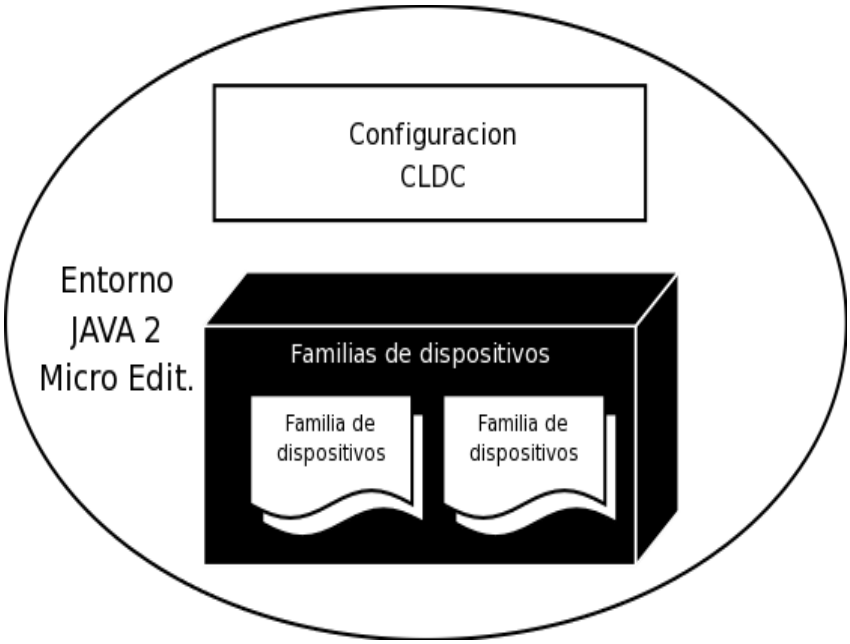


Figura #3 Relación entre categorías en J2ME

Las aplicaciones MIDP permiten ser de tipo intuitivas y gráficas. La interfaz gráfica del usuario se ha optimizado para las pequeñas pantallas, mecanismos de introducción de datos y otras características de los dispositivos móviles. Éstas se pueden instalar y ejecutar localmente (en el dispositivo), trabajar en red o de forma desconectada y pueden almacenar o gestionar de forma segura datos en el móvil.

[5] **Applets:** Es un componente de una aplicación que se ejecuta en el contexto de otro programa, por ejemplo un navegador web.

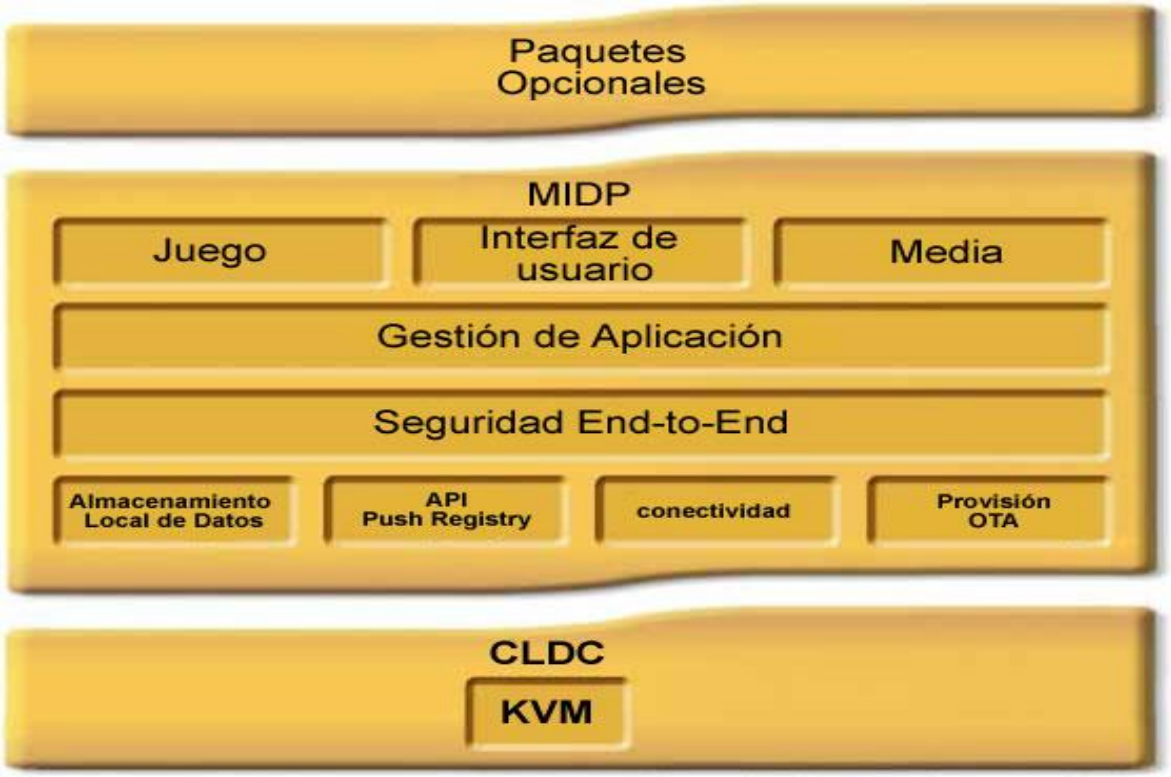


Figura #4 Estructura MIDP

Las pantallas y elementos son relativos al dispositivo, con soporte incorporado en los mismos para el tamaño de la pantalla, mecanismos de navegación e introducción de datos. Las aplicaciones creadas, gracias a la portabilidad proporcionada por J2ME, se adaptarán a la disposición y mecanismos de navegación propios de cada dispositivo proporcionando un modelo de seguridad robusto que protege aplicaciones y dispositivos móviles

1.9. Eclipse como herramienta de desarrollo

Eclipse definido por sus creadores como un IDE (del inglés Integrated Development Environment) para todo y para nada en particular es en el fondo, únicamente un armazón sobre el que se pueden montar herramientas de desarrollo para cualquier lenguaje, mediante la implementación o instalación de los plugins[6] adecuados.

[6] **Plugin (o plug-in en español enchufar):** Es una aplicación informática que interactúa con otra aplicación para aportarle una función o utilidad específica.

La plataforma Eclipse posee código abierto independiente de plataforma alguna para desarrollar aplicaciones de cliente enriquecido con entornos integrados de desarrollo (Daum, 2005). Principalmente se utiliza para el IDE de Java llamado Java Development Toolkit (JDT) y el compilador (conocido como ECJ) que se entrega como parte del mismo (en ocasiones son usados para desarrollar el mismo Eclipse). Las aplicaciones desarrolladas en el mismo tienen un amplio rango de sistemas operativos en los que pueden ejecutarse. Esto, junto a un alto nivel de extensibilidad hace del mismo una herramienta ideal para el desarrollo de aplicaciones.

El plugin específico para el desarrollo de aplicaciones J2ME es el J2ME Wireless Toolkit que proporciona herramientas para desarrollar aplicaciones para dispositivos móviles compatibles con la especificación MIDP (Daum, 2005).

Figura #5 Herramientas asociadas al desarrollo de aplicaciones para J2ME

1.10. Filosofía del J2ME-polish

El J2ME-polish constituye fundamentalmente una suite o colección de herramientas de código abierto para desarrollar aplicaciones J2ME. Incluye herramientas de compilación con una base de datos integrada de dispositivos móviles, una potente interfaz gráfica de usuario, un motor para juegos, una estructura de soporte para registrar datos o información y una colección de diversas utilidades relacionadas (Serantes, 2008).

La filosofía que plantea está relacionada con la construcción de aplicaciones empaquetadas a partir de un único proyecto fuente con múltiples móviles y regionalizaciones. Posee además una base de datos

de dispositivos donde se listan más de 300 dispositivos J2ME y sus respectivas capacidades J2ME tales como tamaño del Canvas[7], APIs, formatos soportados, etc. (Polish, 2007). Estas características permiten utilizarlo para ajustar la construcción de aplicaciones a diferentes dispositivos mediante un pre procesamiento (etapa donde se cambia el código antes de ser compilado.). También puede reprocesar, compilar, ofuscar y empaquetar aplicaciones sin dejar de mencionar que es muy configurable y se integra perfectamente en todos los IDEs de Java.

La inclusión es una parte fundamental con la que cuenta esta herramienta dado que incluye automáticamente los recursos correspondientes a la aplicación que se desarrolle. Esto permite que sea fácil afinar la inclusión de recursos, puedes por ejemplo incluir ficheros MIDI (del inglés Musical Instrument Digital Interface) solo cuando el móvil soporta este tipo de sonido, y no MP3.Si la aplicación va a ser comercializada en diferentes regiones cuenta también con algo llamado regionalización ya que el J2ME-polish puede ajustar los recursos, ya sean textos, imágenes o sonidos, a las diferentes regiones. Toda esta acción no genera sobrecarga (en comparación con las aplicaciones no-regionalizadas) porque las traducciones se incluyen directamente dentro de la aplicación. Las interfaces gráficas de usuarios también se pueden diseñar en esta parte con un alto nivel simplemente utilizando archivos de texto de tipo CSS (del inglés Cascading Style Sheets) que es lo mismo que hojas de estilo en cascada.

Con el motor de juego que posee el J2ME-polish puedes utilizar el API de juegos de MIDP/2.0 en teléfonos MIDP/1.0 (Polish, 2007).El mismo está altamente optimizado en velocidad para lograr que tu juego rinda al máximo utilizando un único código fuente para todos los dispositivos donde se quiera que ejecutar la aplicación.

1.11 Herramientas usadas para el modelado del proyecto (UML)

El equipo de trabajo decidió utilizar la herramienta Rational Rose Enterprise Edition 2003 para el análisis y diseño de artefactos, diagramas y objetos de los distintos flujos de trabajo a lo largo del ciclo de vida de la aplicación. El uso de esta solución de software provee al proyecto de estándares a nivel mundial tanto en la arquitectura como en el diseño mediante el desarrollo dirigido por modelo, prueba de componentes y actividades de análisis de tiempo de ejecución (Fuentes, 2005).

[7] **Canvas:** Es un Componente básico que captura eventos de exposición, del mouse y demás eventos relacionados. Un applet que trabaja con imágenes directamente, ya sea un applet gráfico o de dibujo, los lienzos o zonas de dibujo.

1.11.1. Fundamentos del Rational Rose Enterprise Edition 2003

Rational Rose Enterprise es uno de los productos más completos de la familia Rational Rose los cuales dan soporte al UML(del inglés Unified Modeling Language), lenguaje de modelado de sistemas de software más conocido en la actualidad, el mismo constituye el estándar internacional aprobado por la OMG(del inglés Object Management Group) que se dedica al desarrollo y revisión de especificaciones para la industria del software (Coltell, 2003). Mediante esta herramienta se pueden definir las iteraciones que se producirán entre los usuarios finales y los sistemas informáticos mediante diversas técnicas de diagramación, las clases u objetos que se deben implementar en la fase de desarrollo y generar código Java (Jost, 2004).

El desarrollo de software con una calidad profesional y competitiva exige herramientas que complementen los recursos ofrecidos por compiladores y ambientes de desarrollo integrados convencionales. Rational Rose Enterprise Edition 2003 brinda esta posibilidad maximizando la productividad del grupo de desarrollo a la hora de construir aplicativos de negocios, productos de software y sistemas. A través del mismo también se pueden administrar artefactos del proyecto integrando sistemas de control de versiones.

1.12. Sistemas de Control de Versiones

Los sistemas de control de versiones se definen en una serie de métodos y herramientas disponibles para controlar todo lo referente a los cambios en el tiempo de un archivo (Serrano J., 2007). Estos son extremadamente útiles debido a que es difícil que un archivo de código fuente o un documento de texto estén terminados con la primera escritura; se necesitan cambios o reescrituras para corregir errores o modificar su contenido. Sistemas como éstos brindan dos posibilidades fundamentales:

- ✓ A medida que el archivo se modifica mantener su historial de cambios.
- ✓ Dejar que evolucione sin memoria.

El control de versiones es un método estandarizado para mantener este tipo de memoria o historial haciendo que además sea útil para el desarrollo futuro. Esta práctica es muy utilizada en el mundo de la industria informática pues facilita la administración de las distintas versiones de cada producto desarrollado, como posibles especializaciones realizadas (Serrano J., 2007).Este concepto es

aplicable en otros ámbitos o a cualquier tipo de archivo como documentos, imágenes, sitios web, así tantos como se pueda mencionar.

Con la implementación de sistemas de este tipo, el presente proyecto se nutre de un mecanismo de almacenamiento de los elementos que deba gestionar, la posibilidad de realizar cambios sobre los elementos almacenados y un registro histórico de las acciones realizadas con cada elemento o conjunto de elementos. Los informes con los cambios introducidos entre dos versiones y/o informes de estado juegan un papel protagónico en la organización del trabajo y disponibilidad de la aplicación en su desarrollo.

1.12.1. El Subversion (SVN) como sistema de control de versiones.

El Subversion (SVN) surgió para remplazar al CVS (del inglés Concurrent Versions System) sistema del mismo tipo pero con unas algunas deficiencias en cuanto a implementación. El objetivo fundamental de este software es el control de versiones de sistemas en general siendo de tipo libre bajo una licencia de tipo Apache/BSD (Tigris.org., 2000).

La característica que hace este software una solución factible para la administración de versiones en el flujo de trabajo de desarrollo de este proyecto es que la historia de los archivos y directorios es seguida en un registro, mientras que las modificaciones (incluyendo cambios a varios archivos) son automáticas. La creación de ramas y etiquetas es una operación eficiente y el bloqueo de archivos selectivamente constituye una herramienta en la organización en la actualización del código fuente (Tigris.org., 2000). Existen interfaces o programas individuales que integran a este sistema en entornos de desarrollo. En este proyecto en específico se utilizaran dos para la manipulación y actualización de versiones.

- ✓ **TortoiseSVN:** Provee integración con el explorador de Windows. Es la interfaz más popular en este sistema operativo.
- ✓ **Subclipse:** Plugin que integra Subversion al entorno Eclipse.

1.13. Conclusiones

En el desarrollo del capítulo se presentó de forma detallada y se argumentaron los principales conceptos y aspectos que se han tenido en cuenta a la hora de darle solución a la investigación propuesta. Se abordaron temas que tratan de una forma u otra la tecnología y herramientas con las que se desarrollan los juegos para dispositivos móviles y las principales tendencias en la actualidad a nivel mundial. También se ha planteado y analizado en cuanto a características se refiere, las soluciones actuales que brindan las empresas existentes a la problemática que es motivo del presente proyecto.

Capítulo 2: Características del sistema

2.1. Introducción

En este capítulo se tratan los temas referentes al objeto de estudio de la aplicación, principalmente atendiendo a los procesos involucrados en el campo de acción. Se plantea también la situación problémica que refleja las principales necesidades de los usuarios a los que va encaminado el resultado de esta investigación.

Otro aspecto en el que se profundiza es el concepto de framework[8], se enuncian sus características, se fundamenta qué tipo de framework se ha de aplicar y cómo se desarrollará el mismo según la captura de sus principales requerimientos y el análisis del dominio en el que está ubicado.

2.2. Objetivos estratégicos de la organización

La entidad PROCYON provee soluciones y servicios en el mundo de la telefonía móvil, prestando especial atención en el empleo de nuevas tecnologías de acceso a contenidos, nuevos modelos de dispositivos móviles así como nuevos formatos multimedia. Constantemente se actualiza la infraestructura tecnológica garantizando capacidad inmediata de respuesta a los distintitos modelos de negocios que puedan presentar operadores, clientes y proveedores de contenido. El principal objetivo de la entidad es adentrarse en al mercado de la aplicaciones para móviles.

2.3. Flujo de los procesos involucrados en el campo de acción

En la actualidad las APIs de desarrollo para el diseño de sistemas de menús en dispositivos móviles o interfaces gráficas en general vienen integradas en las plataformas de desarrollo como es el caso de J2ME.El objetivo principal es desarrollar un paquete de clases siguiendo la filosofía de J2ME-polish, que permita a los desarrolladores crear sistemas de menú y/o aplicaciones de forma libre y fácil.

[8] **Framework:** Traducido al español sería armazón. Según Johnson & Foote, un Framework es: "Un conjunto de clases que incorporan un diseño abstracto para soluciones a una familia de problemas relacionados, y soporta re-uso a un nivel de granularidad mayor que el de las clases"

2.4. Análisis crítico de soluciones existentes a nivel mundial

Empresas de software han desarrollado internacionalmente APIs para el desarrollo de interfaces gráficas en aplicaciones dirigidas a dispositivos móviles. Todas en general presentan un problema que es la dependencia de estas APIs a las plataformas de desarrollo, por ejemplo el API de multimedia que trae incluida el J2ME no es compatible en la plataforma Windows Mobile. Por lo tanto las interfaces gráficas de las aplicaciones están sujetas a las plataformas en las que fueron desarrolladas.

Una de las APIs existentes mundialmente de mayor utilización para la creación de este tipo de aplicación es la que trae consigo la plataforma J2ME, que entre otras herramientas llevan las soluciones Java a los mercados de consumo y dispositivos integrados. La parte del API que posee dicha plataforma que se dedica a la manipulación de elementos gráficos, es la llamada MMAPI la cual posee un subconjunto de APIs que conforman una interfaz flexible y simple para el manejo de capacidades multimedia. Éstas unidas a otras como el API 3D posibilitan una mayor disponibilidad gráfica.

Existen otras soluciones que proponen plataformas como Windows Mobile, Linux, Palm, Brew, Symbian, Blackberry e iPhone (este último es el peor parado dado que no tiene un SDK[9] público). Todas proporcionan o destinan de forma similar parte del API que traen incluidas para el desarrollo de soluciones gráficas de esta gama de aplicaciones y fundamentalmente juegos. Todas tienen sus ventajas y desventajas principalmente basadas en el soporte por parte del terminal como son sus capacidades de almacenamiento, procesamiento y tecnología que utiliza.

Cabe aclarar y destacar que la solución que propone esta investigación le proporcionaría a la entidad PROCYON una herramienta factible para ganar en calidad de productos y disminuir el tiempo de desarrollo en las aplicaciones haciéndolas más competitivas en el mercado mundial.

2.5. Objeto de automatización

Desarrollar un conjunto de clases agrupadas en un paquete atendiendo a la filosofía de J2ME-polish que se incluyan al proyecto en el que se este trabajando para la creación de interfaces gráficas. Crear

[9] **SDK (del inglés Software Development Kit):** Es un conjunto de herramientas de desarrollo que le permite a un programador crear aplicaciones.

un estándar de implementación en el desarrollo de aplicaciones atendiendo fundamentalmente a las soluciones gráficas.

Permitir a los desarrolladores abstraerse del manejo de gráficos que propone la plataforma en la que trabaja en cuanto al desarrollo de sistemas de menú y se puedan enfocar solamente en la lógica del juego o aplicación.

2.6. Propuesta del Sistema

El desarrollo de un framework se fundamenta en una serie de proyectos de aplicaciones actuales o en proyecto que comparten un dominio específico y una lógica sobre este (Markiewicz, 2001). Éstos se dividen según la forma en que permiten su extensión, los cuales son:

- ✓ **Caja Blanca:** Requiere del completamiento o redefinición de algunos métodos mediante herencia, por lo cual es necesario el conocimiento de mecanismos internos del sistema.
- ✓ **Caja Negra:** Es un sistema completo, requiere solo seleccionar y configurar componentes ya existentes, para lo cual basta con conocer la interfaz y comportamiento general del sistema.
- ✓ **Caja Gris:** Define una forma intermedia de reutilización, en donde sólo parte del interior de un framework (o componente) se ofrece de forma pública, mientras que el resto se oculta. Es una solución intermedia entre las cajas blancas y negras, y de ahí su nombre. Aunque el nombre de caja gris es menos conocido que los otros, es el que describe la forma de extensión más utilizada.

El framework que dará como resultado la investigación será un sistema completo en forma de un paquete de clases y solo se requerirá seleccionar y configurar los componentes que se propongan mediante el conocimiento e implementación de interfaces y su comportamiento en general. Por lo cual es clasificado de Aplicación por Caja Negra y para su construcción se mantendrá el enfoque desde una aplicación. En este tipo de enfoque se agrupa los componentes comunes y sus relaciones en aplicaciones pertenecientes a un dominio, desarrollando desde éstas el framework (Markiewicz, 2001).

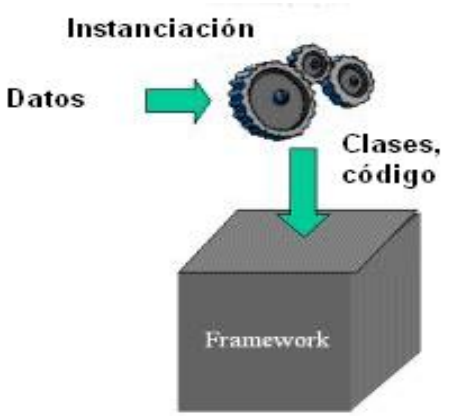


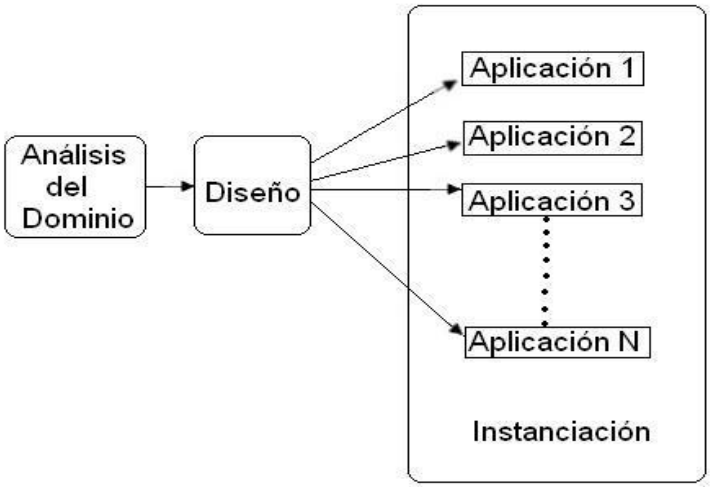
Figura #6 Framework de Caja Negra.

El análisis del dominio se selecciona y abstraen las funciones y objetos específicos, se identifican las características comunes y las relaciones fundamentales. Durante esta etapa, los puntos calientes o de entrada hot-spots (puntos calientes)[10] y los puntos congelados o de salida frozen-spots (puntos congelados)[11] se destapan parcialmente. Se modelan los puntos calientes y los puntos congelados, la extensión y la flexibilidad propuesta en el análisis del dominio se esboza en líneas generales en la fase del diseño (Markiewicz, 2001).

Por último, en la fase de instanciación, los puntos calientes son implementados, generando un software del sistema. Cada una de estas aplicaciones tendrá los puntos congelados del framework en común.

[10] **Los puntos calientes o hot-spots:** Son las clases o los métodos abstractos que deben ser implementados o puestos en ejecución.

[11] **Los puntos congelados o frozen-spots:** Son los puntos inmutables constituyen el núcleo o kernel de un framework ya que algunas de las características del framework no son mutables ni tampoco pueden ser alteradas fácilmente.



Figura#7 Proceso de desarrollo del Framework.

El modelo de desarrollo estará orientado a la reutilización dado que reduce el tiempo de entrega y disminuye el esfuerzo, el costo y los riesgos durante el desarrollo (García F. J., 2002). Este proceso consta de cuatro fases las cuales son:

- ✓ **Análisis de componentes:** Determina qué componentes pueden ser utilizados para el sistema en cuestión. Casi siempre hay que hacer ajustes para adecuarlos.
- ✓ **Modificación de requisitos:** Adaptan los requisitos para concordar con los componentes de la etapa anterior. Si no se puede realizar modificaciones en los requisitos, hay que seguir buscando componentes más adecuados.
    - Las dos fases anteriores entrarían dentro de la etapa de Análisis del Dominio del desarrollo del framework, que según la metodología de desarrollo escogida para la realización de este proyecto encajarían en la fase de exploración y planificación del sistema. Las siguientes se relacionan con las otras dos etapas respectivamente.
- ✓ **Diseño del sistema con reutilización:** Diseña o reutiliza el marco de trabajo para el sistema. Se debe tener en cuenta los componentes localizados en la fase 2 para diseñar o determinar este marco.
    - Proceso encavado dentro de la fase de diseño según XP.

- ✓ **Desarrollo e integración:** Se integran los componentes y subsistemas. La integración es parte del desarrollo en lugar de una actividad separada.
    - Proceso perteneciente a la fase de codificación.

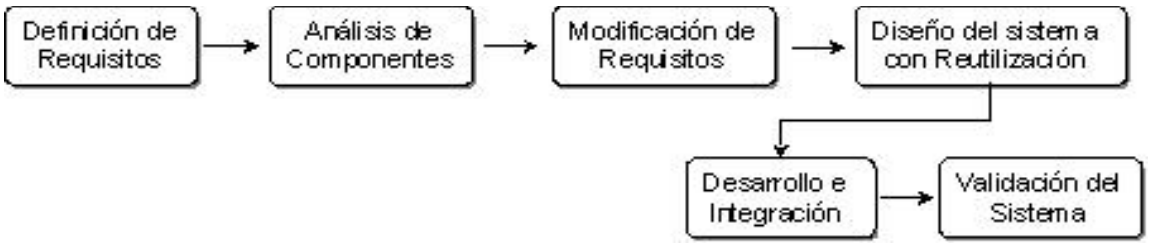


Figura #8 Desarrollo basado en reutilización de componentes.

Dicha implementación es realizada mediante la clasificación de sus clases concretas, utilizando el concepto de colaboración. De esta forma, los componentes de software pueden ser reutilizados a través de composición.

Cualquier aplicación construida a partir de un framework reutiliza tanto la estructura de composición, como la estructura de control. Los framework permiten la modularidad porque encapsulan los detalles de implementación detrás de una interfaz estable para crear nuevas aplicaciones (Markiewicz, 2001).

El framework que se propone como resultado, está compuesto por una serie de clases permitiéndole a los desarrolladores configurar el diseño de las pantallas en las aplicaciones para móviles, agregar y eliminar elementos al sistema de menú, así como crear e incorporar en caso necesario, interfaces de usuario. Esto permite centrar el esfuerzo sólo en el desarrollo de la lógica de la aplicación.

2.7. Conclusiones

En este capítulo se realizó un análisis de la situación problémica que da origen al presente trabajo así como un estudio de los diferentes tipos de framework más utilizados a nivel mundial, con el fin de identificar técnicas utilizadas por éstos y que fueran de utilidad en el presente trabajo.

Capítulo 3: Exploración y Planificación del sistema

3.1. Introducción

En el presente capítulo se hace alusión a las fases de exploración y planificación propias de la metodología de desarrollo utilizada para la implementación del sistema que se propone. Se exponen además los artefactos generados durante el transcurso de las mismas.

3.2. Fase de Exploración. Definición

El ciclo de vida de XP enfatiza en el carácter interactivo e incremental del desarrollo. Una iteración de desarrollo es un período de tiempo en el que se realiza un conjunto de funcionalidades determinadas, que en el caso de XP corresponden a un conjunto de historias de usuarios (Beck, 2000).

La metodología de desarrollo XP comienza con la fase de exploración, en fase los clientes plantean a grandes rasgos las historias de usuario mediante un proceso de identificación que son de interés para la primera entrega del producto. Al mismo tiempo el equipo de desarrollo se familiariza con las herramientas, tecnologías y prácticas que se utilizarán en el proyecto.

Se prueba la tecnología y se exploran las posibilidades de la arquitectura del sistema construyendo un prototipo. La fase de exploración toma de pocas semanas a pocos meses, dependiendo del tamaño y familiaridad que tengan los programadores con la tecnología (XP, 2006).

3.2.1. Historias de Usuario

Las historias de usuarios (HU) representan una breve descripción del comportamiento del sistema empleando una terminología del cliente sin lenguaje técnico. Éstas se realizan una por cada característica principal del sistema y se emplean para hacer estimaciones de tiempo y para el plan de lanzamientos. Una de las ventajas es, que reemplazan un gran documento de requisitos y preceden la creación de las pruebas de aceptación. Durante el análisis en la fase de exploración se identificaron doce HU que a continuación se describen:

Tabla 1. Historia de usuario creación del contenedor de componentes.

Historia de Usuario	
Número: 1	**Nombre:** Creación del contenedor de componentes
Usuario: Desarrollador	
Prioridad en negocio: Alta	**Riesgo de desarrollo:** Alto
Puntos Estimados: 2	**Iteración asignada:** 1
Descripción: En este contenedor de componentes estarán agrupados los componentes visuales que se desarrollen	
Observaciones:	

Tabla 2. Historia de usuario creación del componente Button.

Historia de Usuario	
Número: 2	**Nombre:** Creación del componente Button
Usuario: Desarrollador	
Prioridad en negocio: Alta	**Riesgo de desarrollo:** Medio
Puntos Estimados: 1	**Iteración asignada:** 1
Descripción: Este componente permitirá el desarrollador definir acciones y la navegación dentro del sistema de menú.	
Observaciones: Su uso está ligado a una interface que permitirá al desarrollador programarla para definir la acción dado las opciones del menú.	

Tabla 3. Historia de usuario creación del componente Menu.

Historia de Usuario	
Número: 3	**Nombre:** Creación del componente Menu
Usuario: Desarrollador	
Prioridad en negocio: Alta	**Riesgo de desarrollo:** Alto
Puntos Estimados: 2	**Iteración asignada:** 1

Descripción: Este componente brindara la posibilidad al desarrollador de crear un menú de opciones.
Observaciones: Su uso está ligado a una interface que permitirá al desarrollador programar las opciones que incorpore al menú.

Tabla 4. Historia de usuario creación del componente TextField.

Historia de Usuario	
Número: 4	**Nombre:** Creación del componente TextField
Usuario: Desarrollador	
Prioridad en negocio: Media	**Riesgo de desarrollo:** Medio
Puntos Estimados: 1	**Iteración asignada:** 2
Descripción: Este componente permitirá el desarrollador capturar entradas de textos, contraseñas, campos numéricos y establecer fechas según el tipo que se defina. También será configurable para mostrar mensajes en salas de chateo.	
Observaciones:	

Tabla 5. Historia de usuario creación del componente CheckBox.

Historia de Usuario	
Número: 5	**Nombre:** Creación del componente CheckBox
Usuario: Desarrollador	
Prioridad en negocio: Media	**Riesgo de desarrollo:** Bajo
Puntos Estimados: 1	**Iteración asignada:** 1
Descripción: Este componente permitirá el desarrollador determinar la entrada de datos o parámetros de forma sencilla.	
Observaciones:	

Tabla 6. Historia de usuario creación del componente RadioButton.

Historia de Usuario	
Número: 6	**Nombre:** Creación del componente RadioButton
Usuario: Desarrollador	
Prioridad en negocio: Baja	**Riesgo de desarrollo:** Bajo
Puntos Estimados: 1	**Iteración asignada:** 2
Descripción: Creación de un componente generalmente usado para selección de un único elemento en una lista de varios.	
Observaciones:	

Tabla 7. Historia de usuario creación del componente RadioGroup.

Historia de Usuario	
Número: 7	**Nombre:** Creación del componente RadioGroup
Usuario: Desarrollador	
Prioridad en negocio: Media	**Riesgo de desarrollo:** Medio
Puntos Estimados: 1	**Iteración asignada:** 2
Descripción: Este componente agrupa en los otros componentes de tipo RadioGroup dándole al desarrollador mayor facilidad a la hora de trabajar estos componentes.	
Observaciones:	

Tabla 8. Historia de usuario creación del componente MessageScreener.

Historia de Usuario	
Número: 8	**Nombre:** Creación del componente MessageScreener
Usuario: Desarrollador	
Prioridad en negocio: Media	**Riesgo de desarrollo:** Medio
Puntos Estimados: 1	**Iteración asignada:** 2

Descripción: Este componente brinda una solución visual atractiva a los posibles mensajes generados por algún evento o acción en el sistema de menú.
Observaciones:

Tabla 9. Historia de usuario creación del componente TextScreen.

Historia de Usuario	
Número: 9	**Nombre:** Creación del componente TextScreen
Usuario: Desarrollador	
Prioridad en negocio: Media	**Riesgo de desarrollo:** Alto
Puntos Estimados: 1	**Iteración asignada:** 2
Descripción: Este componente permite al desarrollador mostrar pantallas de texto generalmente a manera de información, adaptando el texto a las dimensiones de pantalla.	
Observaciones:	

Tabla 10. Historia de usuario creación del componente LabelString.

Historia de Usuario	
Número: 10	**Nombre:** Creación del componente LabelString
Usuario: Desarrollador	
Prioridad en negocio: Baja	**Riesgo de desarrollo:** Bajo
Puntos Estimados: 1	**Iteración asignada:**3
Descripción: Este componente le permite al usuario mostrar mensajes de texto (generalmente cortos) con gran facilidad.	
Observaciones:	

Tabla 11. Historia de usuario creación del componente ComboBox.

Historia de Usuario	
Número: 11	**Nombre:** Creación del componente ComboBox
Usuario: Desarrollador	
Prioridad en negocio: Media	**Riesgo de desarrollo:** Alto
Puntos Estimados: 2	**Iteración asignada:** 3
Descripción: Este componente le permite al desarrollador mostrarle al usuario final una lista de opciones donde el usuario final debe seleccionar una.	
Observaciones: Su uso está ligado a una interface que permitirá al desarrollador programar las acciones sobre las opciones que incorpore al menú.	

Tabla 12. Historia de usuario creación del componente ProgressInformer.

Historia de Usuario	
Número: 12	**Nombre:** Creación del componente ProgressInformer
Usuario: Desarrollador	
Prioridad en negocio: Baja	**Riesgo de desarrollo:** Medio
Puntos Estimados: 1	**Iteración asignada:** 3
Descripción: Este componente brinda una solución visual atractiva a los momentos de espera de la aplicación que se realice, ya sea mientras se cargan archivos o configuraciones o mientras se espera a que termine otro evento, además el desarrollador puede utilizarlo para darle información al usuario final sobre el tiempo de espera restante.	
Observaciones:	

3.3. Planificación. Definición

En esta fase se priorizan las historias de usuario y se acuerda el alcance de la planificación de entrega (*release*). Los programadores estiman cuánto esfuerzo requiere cada historia y a partir de allí se define el cronograma. Esta estimación se expresa utilizando como medida el punto. Un punto se considera

como una semana ideal de trabajo donde los miembros de los equipos de desarrollo trabajan el tiempo planeado sin ningún tipo de interrupción (Beck, 2000).

La planificación no debe ser estricta puesto que hay muchas variables en juego, debe ser flexible para poder adaptarse a los cambios que puedan surgir. Una buena estrategia es hacer planificaciones detalladas para unas pocas semanas y planificaciones mucho más abiertas para unos pocos meses (XP, 2006).

3.3.1. Estimación de esfuerzo

La estimación de esfuerzo que se decidió para el desarrollo de cada una de las historias de usuario de la investigación se representa en la siguiente tabla.

Tabla 13. Plan de estimación de esfuerzo por historias de usuarios.

Historias de Usuarios	Puntos Estimados
Creación del contenedor de componentes	2
Creación del componente Button	1
Creación del componente TextField	1
Creación del componente CheckBox	1
Creación del componente Menu	2
Creación del componente RadioButton	1
Creación del componente RadioGroup	1
Creación del componente MessageScreener	1
Creación del componente TextScreen	1
Creación del componente ComboBox	2
Creación del componente LabelString	1
Creación del componente ProgressInformer	1

3.3.2. Plan de Iteraciones

La metodología XP contiene la filosofía de las metodologías ágiles, las cuales dan mayor valor al individuo, a la colaboración con el cliente y al desarrollo incremental del software con iteraciones muy cortas (Letelier, 2004). La etapa de implementación se ha planificado realizarla en tres iteraciones atendiendo a la cantidad de HU junto a sus prioridades en negocio y los puntos estimados de cada una. Además de las entregas habituales al final de cada iteración, todas las semanas se hace un resumen con la presencia del cliente y se muestran los adelantos del sistema, y con el mismo objetivo se realizan reuniones diarias entre los desarrolladores al final de cada sesión de trabajo. Las iteraciones se detallan a continuación.

3.3.2.1. Iteración 1

Esta iteración tiene como objetivo la implementación de las HU de una prioridad en negocio alta. Al finalizar la misma estarán listos los elementos fundamentales del framework como son el contenedor de componentes, los botones y los items[12] de una lista de menú que constituyen una versión prueba para obtener una retroalimentación del equipo de trabajo.

3.3.2.2. Iteración 2

En esta iteración se implementaran las HU de una prioridad en negocio media y se corregirán errores o disconformidades del usuario con los componentes implementados en la iteración anterior. Como resultado se obtendrá una versión mas completa sobre las funcionalidades del sistema de menú. Esta segunda versión será mostrada a posibles clientes con el único objetivo de realizar cambios en base a la aceptación del mismo.

3.3.2.3. Iteración 3

El objetivo fundamental de esta iteración es la implementación de las HU de baja prioridad en el negocio y corregir errores de iteraciones anteriores. Al final de esta iteración se obtendrá la versión 1.0

[12] **Ítems**: elementos

final del producto. Esta versión se pondrá en funcionamiento utilizándola en la creación de productos para la evaluación de su comportamiento y rendimiento.

3.3.3. Duración de las Iteraciones

Para una mayor organización del trabajo como lo plantea el ciclo de vida de XP se crea un plan de duración de las iteraciones, en este caso se realizaría un solo plan ya que existe un único equipo de desarrolladores. A continuación se muestra cómo estarán organizadas las HU según el orden que serán abordadas en cada iteración según su prioridad y el tiempo de duración de las mismas.

Tabla 14. Plan de duración de iteraciones para equipo de trabajo.

Iteración	Orden de la HU a implementar	Duración total de la iteración
Iteración #1	1- Creación del contenedor de componentes 2- Creación del componente Button 3- Creación del componente Menu	5 semanas
Iteración #2	1- Creación del componente TextField 2- Creación del componente CheckBox 3- Creación del componente RadioButton 4- Creación del componente RadioGroup 5- Creación del componente MessageScreener	5 semanas
Iteración #3	1- Creación del componente TextScreen 2- Creación del componente LabelString 3- Creación del componente ComboBox 4- Creación del componente ProgressInformer	5 semanas

3.3.4. Plan de entregas

En el plan de entrega que se plantea a continuación se hace una propuesta de la fecha aproximada en que se harán versiones (releases) al sistema al finalizar cada iteración en la fase de implementación.

Tabla 15. Plan de entregas.

Módulo	Final 1 iteración 2da semana Marzo	Final 2 iteración 2da semana Abril	Final 3 iteración 3ra semana Mayo
com.procyon.pmgs.api	0.1	0.2	1.0

3.4. Conclusiones

Como parte del presente capítulo se abordó todo lo referente a las fases de exploración y planificación del proyecto, haciendo una descripción de cada uno de los artefactos generados durante el transcurso de las mismas.

Capítulo 4: Diseño, desarrollo y pruebas

4.1. Introducción

En el presente capítulo se describen las fases de diseño, desarrollo y prueba propias de la metodología de desarrollo XP. Se detallan las tres iteraciones realizadas durante la etapa de codificación del proyecto, así como las tareas de programación y pruebas de aceptación desarrolladas en cada una de éstas.

4.2. Diseño y Desarrollo

El desarrollo es la parte más importante en el proceso de la programación extrema. Todos los trabajos tienen como objetivo que se programen lo más rápidamente posible, sin interrupciones y en dirección correcta. También es muy importante el diseño, y se establecen los mecanismos, para que éste sea revisado y mejorado de manera continuada a lo largo del proyecto, según se van añadiendo funcionalidades al mismo (Beck, 2000).

Para una producción efectiva la metodología XP plantea una serie de prácticas básicas que deben seguirse al pie de la letra para el desarrollo exitoso de un proyecto .A continuación se mencionan las que fueron pilares en el desarrollo de la presente investigación y las que se abordan fundamentalmente en este capítulo.

- ✓ **Test del cliente:** El cliente, con la ayuda de los desarrolladores, propone sus propias pruebas para validar las mini-versiones.
- ✓ **Versiones pequeñas:** Las mini-versiones deben ser lo suficientemente pequeñas como para poder hacer una cada pocas semanas. Deben ser versiones que ofrezcan algo útil al usuario final y no trozos de código que no pueda ver funcionando.
- ✓ **Desarrollo guiado por las pruebas automáticas:** Se deben realizar programas de prueba automática y deben ejecutarse con mucha frecuencia.

La forma iterativa para la implementación de un software que plantea XP junto a estas prácticas da como resultado que al culminar cada iteración se obtenga una versión del producto funcional, éste debe ser probado y mostrado al cliente sirviendo de retroalimentación para el equipo de trabajo. En el

presente capítulo se exponen detalladamente las tres iteraciones generadas por la planificación descrita anteriormente así como las tareas que se plantearon para la realización de cada una de las historias de usuarios y las pruebas que se efectuaron al sistema.

4.2.1. Iteración 1

En esta primera iteración se desarrollan las historias de usuarios de mayor prioridad en el sistema definiéndose la arquitectura del mismo con el objetivo de obtener una primera versión del producto con las principales características o funcionalidades para ser mostrado al cliente.

Tabla 16. Historias abordadas en la primera iteración.

Historia de Usuario	Estimación	Real
Creación del contenedor de componentes	2	2
Creación del componente Button	1	1
Creación del componente Menu	2	2
Total	**5**	**5**

4.2.1.1. Tareas generadas por cada historia de usuario

Creación del contenedor de componentes

Tabla 17. Tarea #1 HU Creación del contenedor de componentes.

Tarea	
Número de tarea: 1	**Número de historia:** 1
Nombre de la tarea: Creación y diseño del ContainerScreen	
Tipo de tarea: Desarrollo	**Puntos estimados:** 2
Fecha inicio: 2008-02-03	**Fecha fin:** 2008-02-06
Programador responsable: Camilo R. Legón Sarmiento	
Descripción: Se creará el contendor de componentes ContainerScreen que debe ser configurable en cuanto a diseño para ajustarse con las necesidades del cliente.	

Tabla 18. Tarea #2 HU Creación del contenedor de componentes.

Tarea	
Número de tarea: 2	**Número de historia:** 1
Nombre de la tarea: Diseño utilizando imágenes	
Tipo de tarea: Desarrollo	**Puntos estimados:** 2
Fecha inicio: 2008-02-07	**Fecha fin:** 2008-02-10
Programador responsable: Camilo R. Legón Sarmiento	
Descripción: El contenedor podrá ser diseñado mediante imágenes para ajustarse a las necesidades del cliente.	

Tabla 19. Tarea #3 HU Creación del contenedor de componentes.

Tarea	
Número de tarea:3	**Número de historia:** 1
Nombre de la tarea: Manejo de eventos en el contenedor	
Tipo de tarea: Desarrollo	**Puntos estimados:** 2
Fecha inicio: 2008-02-11	**Fecha fin:** 2008-02-17
Programador responsable: Camilo R. Legón Sarmiento	
Descripción: El contenedor agrupará componentes visuales y será capaz de manejar los eventos que generen.	

Creación del componente Button

Tabla 20. Tarea #1 HU Creación del componente Button.

Tarea	
Número de tarea:1	**Número de historia:** 2
Nombre de la tarea: Creación y diseño del componente Button	
Tipo de tarea: Desarrollo	**Puntos estimados:** 1
Fecha inicio: 2008-02-18	**Fecha fin:** 2008-02-19
Programador responsable: Camilo R. Legón Sarmiento	

Descripción: Se creará el componente Button, éste debe brindar la opción de ser configurable en cuanto a diseño para ajustarse con las necesidades del cliente.

Tabla 21. Tarea #2 HU Creación del componente Button.

Tarea	
Número de tarea:2	**Número de historia:** 2
Nombre de la tarea: Diseño utilizando imágenes	
Tipo de tarea: Desarrollo	**Puntos estimados:** 1
Fecha inicio: 2008-02-20	**Fecha fin:** 2008-02-21
Programador responsable: Camilo R. Legón Sarmiento	
Descripción: El componente Button podrá ser diseñado mediante imágenes para ajustarse a las necesidades del cliente.	

Tabla 22. Tarea #3 HU Creación del componente Button.

Tarea	
Número de tarea: 3	**Número de historia:** 2
Nombre de la tarea: Creación de la interfaz del componente Button	
Tipo de tarea: Desarrollo	**Puntos estimados:** 1
Fecha inicio: 2008-02-22	**Fecha fin:** 2008-02-24
Programador responsable: Camilo R. Legón Sarmiento	
Descripción: Se creará una interfaz para poder manejar y configurar los eventos del componente Button.	

Creación del componente Menu

Tabla 23. Tarea #1 HU Creación del componente Menu.

Tarea	
Número de tarea:1	**Número de historia:** 3
Nombre de la tarea: Creación y configuración del Menu	

Tipo de tarea: Desarrollo	**Puntos estimados:** 2
Fecha inicio: 2008-02-25	**Fecha fin:** 2008-03-03
Programador responsable: Yuniel Bermudez Diaz	
Descripción: Se creará el componente Menu que debe brindar la opción de ser configurable en cuanto a diseño para ajustarse con las necesidades del cliente.	

Tabla 24. Tarea #2 HU Creación del componente Menu.

Tarea	
Número de tarea: 2	**Número de historia:** 3
Nombre de la tarea: Creación de la interfaz del componente Menu	
Tipo de tarea: Desarrollo	**Puntos estimados:** 2
Fecha inicio: 2008-03-04	**Fecha fin:** 2008-03-10
Programador responsable: Yuniel Bermudez Diaz	
Descripción: Se creará una interfaz para poder manejar y configurar los eventos que generen cada uno de los elementos que posea el menú.	

4.2.2. Iteración 2

En el transcurso de esta iteración se implementaron las historias de usuarios con un nivel de prioridad medio para el sistema .Esto permitió agregar nuevas funcionalidades al sistema por lo que se obtuvo una versión mas completa donde el cliente podrá observar algunas de las facilidades en cuanto a configuración y diseño que brindará el framework en su culminación.

Tabla 25. Historias abordadas en la segunda iteración.

Historia de Usuario	Estimación	Real
Creación del componente TextField	1	1
Creación del componente CheckBox	1	1
Creación del componente RadioButton	1	1
Creación del componente RadioGroup	1	1
Creación del componente MessageScreener	1	1
Total	**5**	**5**

4.2.2.1. Tareas generadas por cada historia de usuario

Creación del componente TextField

Tabla 26. Tarea#1 HU Creación del componente TextField.

Tarea	
Número de tarea: 1	**Número de historia:** 4
Nombre de la tarea: Creación y diseño del componente TextField	
Tipo de tarea: Desarrollo	**Puntos estimados:** 1
Fecha inicio: 2008-03-12	**Fecha fin:** 2008-03-14
Programador responsable: Camilo R. Legón Sarmiento	
Descripción: Se creará el componente TextField que tendrá la caracteristica de ser configurable en cuanto a diseño para ajustarse a las necesidades del cliente.	

Tabla 27. Tarea#2 HU Creación del componente TextField.

Tarea	
Número de tarea: 2	**Número de historia:** 4
Nombre de la tarea: Configuración de los tipos de TextField	
Tipo de tarea: Desarrollo	**Puntos estimados:** 1
Fecha inicio: 2008-03-15	**Fecha fin:** 2008-03-19
Programador responsable: Camilo R. Legón Sarmiento	
Descripción: El componente TextField brindará la posibilidad de configurar el tipo de datos que se van a tratar (*texto, fechas, contraseñas, números, mensajes de chat).*Así como el tratamiento gráfico y de eventos que lleva cada uno en particular.	

Creación del componente CheckBox

Tabla 28. Tarea #1 HU Creación del componente CheckBox.

Tarea	
Número de tarea: 1	**Número de historia:** 5
Nombre de la tarea: Diseño y configuración del componente CheckBox	

Tipo de tarea: Desarrollo	**Puntos estimados:** 1
Fecha inicio: 2008-03-20	**Fecha fin:** 2008-03-25
Programador responsable: Camilo R. Legón Sarmiento	
Descripción: Se creará el componente CheckBox que brindara la posibilidad de ser configurable en cuanto a diseño y tratamiento de eventos que el mismo genere.	

Creación del componente RadioButton

Tabla 29. Tarea #1 HU Creación del componente RadioButton.

Tarea	
Número de tarea: 1	**Número de historia:** 6
Nombre de la tarea: Creación y diseño del componente RadioButton	
Tipo de tarea: Desarrollo	**Puntos estimados:** 1
Fecha inicio: 2008-03-26	**Fecha fin:** 2008-04-02
Programador responsable: Yuniel Bermudez Diaz	
Descripción: Creación del componente RadioButton que tendrá la opción de configurar el diseño para ajustarse con las necesidades del cliente.	

Creación del componente RadioGroup

Tabla 30. Tarea #1 HU Creación del componente RadioGroup.

Tarea	
Número de tarea: 1	**Número de historia:** 7
Nombre de la tarea: Creación y diseño del componente RadioGroup	
Tipo de tarea: Desarrollo	**Puntos estimados:** 1
Fecha inicio: 2008-04-03	**Fecha fin:** 2008-04-05
Programador responsable: Yuniel Bermudez Diaz	
Descripción: Se creará el componente RadioGroup que brindará la posibilidad de ser configurable en cuanto a diseño para ajustarse a las necesidades del cliente.	

Tabla 31. Tarea #2 HU Creación del componente RadioGroup.

Tarea	
Número de tarea: 2	**Número de historia:** 7
Nombre de la tarea: Integración del componente RadioButton	
Tipo de tarea: Desarrollo	**Puntos estimados:** 1
Fecha inicio: 2008-04-06	**Fecha fin:** 2008-04-07
Programador responsable: Yuniel Bermudez Diaz	
Descripción: El componente RadioGroup será capaz de integrar componentes RadioButton como ítems.	

Tabla 32. Tarea #3 HU Creación del componente RadioGroup.

Tarea	
Número de tarea: 3	**Número de historia:** 7
Nombre de la tarea: Manejo de eventos generados por ítems	
Tipo de tarea: Desarrollo	**Puntos estimados:** 1
Fecha inicio: 2008-04-08	**Fecha fin:** 2008-04-09
Programador responsable: Yuniel Bermudez Diaz	
Descripción: El componente RadioGroup será capaz de manejar los eventos generados por cada ítem agregado.	

Tabla 33. Tarea #1 HU Creación del componente MessageScreener.

Tarea	
Número de tarea: 1	**Número de historia:** 8
Nombre de la tarea: Creación y diseño del componente MessageScreener	
Tipo de tarea: Desarrollo	**Puntos estimados:** 1
Fecha inicio: 2008-04-10	**Fecha fin:** 2008-04-16
Programador responsable: Camilo R. Legón Sarmiento	
Descripción: Creación del componente MessageScreener que tendrá la opción de configurar el diseño para ajustarse con las necesidades del cliente.	

4.2.3. Iteración 3

Al finalizar esta iteración se habrán implementado todos los elementos y funcionalidades del sistema brindando una versión final del producto listo para ponerlo en funcionamiento.

Tabla 34. Historias abordadas en la tercera iteración.

Historia de Usuario	Estimación	Real
Creación del componente TextScreen	1	1
Creación del componente LabelString	1	1
Creación del componente ComboBox	2	2
Creación del componente ProgressInformer	1	1
Total	**5**	**5**

4.2.3.1. Tareas generadas por cada historia de usuario

Creación del componente TextScreen

Tabla 35. Tarea #1 HU Creación del componente TextScreen.

Tarea	
Número de tarea: 1	**Número de historia:** 9
Nombre de la tarea: Creación y diseño del componente TextScreen	
Tipo de tarea: Desarrollo	**Puntos estimados:** 1
Fecha inicio: 2008-04-17	**Fecha fin:** 2008-04-20
Programador responsable: Yuniel Bermudez Diaz	
Descripción: Se creará el componente TextScreen que brindará la posibilidad de ser configurable en cuanto a diseño para ajustarse con las necesidades del cliente.	

Tabla 36. Tarea #2 HU Creación del componente TextScreen.

Tarea	
Número de tarea: 2	**Número de historia:** 9
Nombre de la tarea: Configuración del componente TextScreen	
Tipo de tarea: Desarrollo	**Puntos estimados:** 1
Fecha inicio: 2008-04-20	**Fecha fin:** 2008-04-26
Programador responsable: Yuniel Bermudez Diaz	
Descripción: El componente TextScreen brindará la opción de configurar el origen o fuente del texto a mostrar visualmente ya sea mediante un fichero o por entrada directa de datos.	

Creación del componente LabelString

Tabla 37. Tarea #1 HU Creación del componente LabelString.

Tarea	
Número de tarea: 1	**Número de historia:** 10
Nombre de la tarea: Creación y diseño del componente LabelString	
Tipo de tarea: Desarrollo	**Puntos estimados:** 1
Fecha inicio: 2008-04-27	**Fecha fin:** 2008-05-04
Programador responsable: Camilo R. Legón Sarmiento	
Descripción: Creación del componente LabelString que tendrá la opción de ser configurable en cuanto a diseño para ajustarse con las necesidades del cliente.	

Creación del componente ComboBox

Tabla 38. Tarea #1 HU Creación del componente ComboBox.

Tarea	
Número de tarea: 1	**Número de historia:** 11
Nombre de la tarea: Creación y diseño del componente ComboBox	
Tipo de tarea: Desarrollo	**Puntos estimados:** 2
Fecha inicio: 2008-05-05	**Fecha fin:** 2008-05-08

Programador responsable: Camilo R. Legón Sarmiento
Descripción: Creación del componente ComboBox que tendrá la opción de ser configurable en cuanto a diseño para ajustarse con las necesidades del cliente.

Tabla 39. Tarea #2 HU Creación del componente ComboBox.

Tarea	
Número de tarea: 2	**Número de historia:** 11
Nombre de la tarea: Diseño utilizando imágenes	
Tipo de tarea: Desarrollo	**Puntos estimados:** 1
Fecha inicio: 2008-05-09	**Fecha fin:** 2008-05-14
Programador responsable: Camilo R. Legón Sarmiento	
Descripción: El componente ComboBox podrá ser diseñado mediante imágenes para ajustarse a las necesidades del cliente.	

Tabla 40. Tarea #3 HU Creación del componente ComboBox.

Tarea	
Número de tarea: 3	**Número de historia:** 11
Nombre de la tarea: Creación de la interfaz del componente ComboBox	
Tipo de tarea: Desarrollo	**Puntos estimados:** 1
Fecha inicio: 2008-05-14	**Fecha fin:** 2008-05-16
Programador responsable: Camilo R. Legón Sarmiento	
Descripción: Se creará una interfaz para poder configurar y manejar el evento generado en la selección de un ítem del componente ComboBox.	

Tabla 41. Tarea #1 HU Creación del componente ProgressInformer.

Tarea	
Número de tarea: 1	**Número de historia:** 12
Nombre de la tarea: Creación y diseño del componente ProgressInformer	
Tipo de tarea: Desarrollo	**Puntos estimados:** 2
Fecha inicio: 2008-05-17	**Fecha fin:** 2008-05-24
Programador responsable: Yuniel Bermudez Diaz	
Descripción: Creación del componente ProgressInformer que tendrá la opción de ser configurable en cuanto a diseño para ajustarse con las necesidades del cliente.	

4.2.4. Diagramas de clases

La metodología XP plantea un nuevo concepto que son las llamadas metáforas, éstas son desarrolladas por los programadores al inicio del proyecto, define una historia de cómo funciona el sistema completo. XP estimula historias de usuario, que son breves descripciones de un trabajo de un sistema, en lugar de los tradicionales diagramas y modelos UML. La metáfora expresa la visión evolutiva del proyecto que define el alcance y propósito del sistema (Beck, 2000).

Las tarjetas CRC (Clase, Responsabilidad y Colaboración) también ayudarán al equipo a definir actividades durante el diseño del sistema. Cada tarjeta representa una clase en la programación orientada a objetos y define sus responsabilidades (lo que ha de hacer) y las colaboraciones con las otras clases (Beck, 2000). Aunque la representación de un sistema mediante diagramas de clases utilizando notación UML no es necesaria ,se puede aplicar esta práctica siempre y cuando la comunicación mejore y no sea un peso su mantenimiento, no sean extensos y se enfoquen en la información importante (Fowler, 2004).

Los diagramas de clases generados para el desarrollo del sistema propuesto se pueden ver en el Anexo 1.

4.2.5. Decisiones de diseño

"*Una arquitectura orientada a objetos bien estructurada está llena de patrones. La calidad de un sistema orientado a objetos se mide por la atención que los diseñadores han prestado a las colaboraciones entre sus objetos. Los patrones conducen a arquitecturas más pequeñas, más simples y más comprensibles.*" (Grady Booch[13]).

El diseño es un modelo del sistema, realizado con una serie de principios y técnicas, que permite describir el sistema con el suficiente detalle como para ser implementado. Actualmente, los patrones de diseño son sin duda alguna la herramienta más importante de la que disponen los ingenieros, arquitectos, analistas y desarrolladores para la creación de sistemas robustos, escalables, fácilmente adaptables y con grandes cotas de reutilización (Lago, 2007).

En el diseño y desarrollo del framework que se propone en este trabajo se utilizan algunos de los patrones GRASP (del inglés General Responsibility Assignment Software Patterns) para la asignación de responsabilidades en el diseño de objetos, a continuación se mencionan los de mayor peso.

- ✓ **Bajo Acoplamiento:** El sistema debe contener pocas dependencias entre clases (principio básico para la creación un buen diseño).
- ✓ **Experto:** La responsabilidad de realizar una labor es de la clase que tiene o puede tener los datos involucrados (atributos).
- ✓ **Alta Cohesión:** Cada elemento del diseño debe realizar una labor única dentro del sistema. El empleo de este patrón proporciona refactorización[14] en el sistema.
- ✓ **Creador:** El sistema debe definir un principio general para la asignación de las responsabilidades de creación entre clases. El empleo de este patrón provee al diseño la capacidad de soportar un bajo acoplamiento, mayor claridad, encapsulación y reutilización.

[13] **Grady Booch :(**nació el 27 de febrero de 1955) es un diseñador de software, investigador de la ingeniería de software y entusiasta de diseño de patrones. Es director científico de Rational Software (ahora parte de IBM) y editor de una serie de Benjamín/Cummings.

[14] **Refactorización:** Cambio realizado a la estructura interna del software para hacerlo mas fácil de comprender y mas fácil de modificar sin cambiar su comportamiento observable.

- ✓ **Controlador:** La responsabilidad de controlar el flujo de eventos del sistema se debe asignar a clases específicas. Esto facilita la centralización de actividades en el sistema.

Entre los patrones de diseño GOF (del inglés Gang of Four[15]) se utiliza el patrón de comportamiento Template Method (Método Plantilla) que plantea la definición de una operación con el esqueleto de un algoritmo, delegando en las subclases algunos de sus pasos. Permite que las subclases redefinan ciertos pasos del algoritmo sin cambiar su estructura (Gracia, 2005).

La principal ventaja que presenta este patrón es que facilita la reutilización de código, por eso es fundamental la aplicación de éste en muchos frameworks. Otro uso particular se da en la creación de sistemas de plugins. Por definición este patrón garantiza la existencia de distintos métodos en una clase que pueden ser sobrescritos por un descendiente (Merseguer, 2006).

Otro de los patrones GOF utilizados es el patrón Facade (Fachada), este es un patrón de tipo estructural y sus creadores dan la siguiente definición: "*Provide a unified interface to a set of interfaces. Facade defines a higher-level interface that makes the subsystem easier to use*". [Proporciona una interfaz unificada para un conjunto de interfaces de un subsistema. Define una interfaz de alto nivel que hace que el subsistema sea más fácil de usar] (Gamma, y otros, 1994).

Este patrón es utilizado para garantizar mayor comodidad y para facilitarles la implementación a los desarrolladores de aplicaciones que utilicen el framework propuesto. Para que el uso del API sea lo más sencillo posible, la mayor parte del manejo de gráficos al nivel más bajo es invisible para el desarrollador, los programadores acceden a dichos gráficos a través de funciones y interfaces que se ponen a su disposición, y éstas se encargan de interpretar las configuraciones y traducirlas a elementos visuales.

Con la aplicación de los patrones anteriormente descritos se logra la combinación de tres factores importantes para el logro de un producto de software con calidad, los cuales son: metodologías de desarrollo, patrones de diseño y refactorización.

[15] **Gang of Four** (en español Pandilla de Cuatro): Sé le denomina así a la unión de Erich Gamma, Richard Helm, Ralph Johnson y John Vlisides cuatro profesionales del desarrollo de la ingeniería de software que dieron autoría al libro Design Patterns de gran éxito en el mundo de la informática, en el cual se describen 23 patrones de diseño comunes.

4.3. Pruebas

Mientras todos los procesos o metodologías de desarrollo mencionan la comprobación, la mayoría lo hace con muy poco énfasis. Sin embargo la metodología XP pone la comprobación como el fundamento del desarrollo, con cada programador escribiendo pruebas cuando escriben su código de producción. Las pruebas se integran en el proceso de integración continua y construcción lo que rinde una plataforma altamente estable para el desarrollo futuro. La programación extrema define dos tipos de pruebas las cuales son:

- ✓ **Pruebas unitarias.**
- ✓ **Pruebas de aceptación**.

4.3.1. Pruebas Unitarias

La producción de código está dirigida por las pruebas unitarias. Las pruebas unitarias son establecidas antes de escribir el código y son ejecutadas constantemente ante cada modificación del sistema. Los clientes escriben las pruebas funcionales para cada historia de usuario que deba validarse. En este contexto de desarrollo evolutivo y de énfasis en pruebas constantes, la automatización para apoyar esta actividad es crucial (Beck, 2000).

4.3.2. Pruebas de Aceptación

Las pruebas de aceptación permiten confirmar que la HU ha sido implementada correctamente al final de cada iteración. Este período de prueba se conoce también como período de caja negra donde se definirán las entradas al sistema y los resultados esperados de estas entradas. Una HU puede tener todas las pruebas de aceptación que necesite para asegurar su correcto funcionamiento. El objetivo final de éstas es garantizar que los requerimientos han sido cumplidos y que el sistema es aceptable (XP, 2006). Las pruebas de aceptación realizadas para cada HU del sistema se encuentran recogidas en el Anexo 2.

4.4. Conclusiones

En el presente capítulo se hizo referencia a las etapas de implementación y pruebas del software en desarrollo exponiendo todos los artefactos generados con este objetivo junto a sus descripciones.

Capítulo 5: Estudio de la Factibilidad

5.1. Introducción

El objetivo principal de la planificación de proyectos es la estimación de tiempo, coste y riesgo teniendo en cuenta principalmente los valores de esfuerzo, personas, tiempo y los recursos de software y hardware involucrados en el desarrollo. La planificación es una técnica para minimizar la incertidumbre y dar más consistencia al desempeño de la empresa (Idalberto, Chiavenato).

En el desarrollo de este capítulo se expone el estudio de la factibilidad realizado por el equipo de trabajo para llevar a cabo el proyecto propuesto.

5.2. Características del proyecto

Las características de un proyecto de software está dado por varios factores como son el flujo de datos que maneja y los recursos informáticos que necesita en su ejecución. Existen modelos de trabajo que definen estas características para su manejo y a su vez realizan la estimación de costos, este es el caso de COCOMO (del inglés: Constructive Cost Model).

Tabla 42. Entradas externas (EI).

Nombre de la entrada externa	Cantidad de Ficheros	Cantidad de elementos de datos	Clasificación (simple ,media y compleja)
Configurar componente visual	1	5	simple
Configurar contenedor de componentes	1	6	simple
Total		**11**	

Tabla 43. Salidas externas (EO).

Nombre de la salida externa	Cantidad de Ficheros	Cantidad de elementos de datos	Clasificación (simple ,media y compleja)
Componente visual	1	7	simple
Contenedor de componentes	1	9	simple
Total		**16**	

Tabla 44. Ficheros internos (LIF).

Nombre del fichero interno	Cantidad de Ficheros	Cantidad de elementos de datos	Clasificación (simple, media y compleja)
Recursos	1	2	simple
Total		**2**	

Tabla 45. Interfaces externas (EIF).

Nombre de la interfaz externa	Cantidad de Ficheros	Cantidad de elementos de datos	Clasificación (simple, media y compleja)
Device manager	1	5	simple
Total		**5**	

El modelo COCOMO II usa Puntos Función y/o Líneas de Código Fuente (SLOC) como base para medir tamaño en los modelos de estimación de Diseño Temprano y Post-Arquitectura. Los Puntos de Función procuran cuantificar la funcionalidad de un sistema de software. Éstos son útiles estimadores ya que están basados en información que está disponible en las etapas tempranas del ciclo de vida del

desarrollo de software. COCOMO II considera solamente UFP (del inglés: Puntos Función Desajustados) (COCOMO, 2004).

Tabla 46. Puntos de función desajustados.

Elementos	Simple		Medio		Complejo		Subtotal
	No.	Peso	No.	Peso	No.	Peso	
Entradas externas (EI)	**11**	**3**	**0**	**4**	**0**	**6**	**33**
Salidas externas (EO)	**16**	**4**	**0**	**5**	**0**	**7**	**64**
Consultas externas (EQ)	**0**	**3**	**0**	**4**	**0**	**6**	**0**
Ficheros internos (LIF)	**2**	**7**	**0**	**10**	**0**	**15**	**14**
Interfaces externas (EIF)	**5**	**5**	**0**	**7**	**0**	**10**	**25**
Total							**136**

COCOMO II abandona definitivamente la idea de medir el tamaño del código en líneas físicas y se utilizan instrucciones o líneas lógicas de código fuente. Para medir a posteriori el número de líneas lógicas, Boehm[16] pone a la disposición el programa CodeCount[17] y las reglas sobre las cuales se basa tal programa. Para efectos de Java, cada punto de función corresponde a 53 líneas lógicas de código fuente (Moreno, 2005).

[16] **Boehm:** Dr. Barry W. Boehm creador del método COCOMO a finales de los 70 y comienzos de los 80, exponiéndolo detalladamente en su libro "Software Engineering Economics" (Prentice-Hall, 1981).

[17] **CodeCount:** Programa para calcular a posteriori el número de líneas lógicas en un programa, creado por Dr. Barry W. Boehm

Tabla 47. Características del proyecto.

Características	Valor
Puntos de función desajustados	136
Lenguaje(Java)	53
Instrucciones fuentes por puntos de función	7208
Instrucciones fuentes	7.21 KSLOC

5.3. Estimación

En el proceso de estimación se realiza una aproximación de los costos de los recursos necesarios para completar las actividades del proyecto. COCOMO II pasó a ser parte de una familia de modelos de productividad, estimación y toma de decisiones, por lo que se ajustó el sistema propuesto a su modelo base de estimación para determinar el cálculo de esfuerzo, tiempo de desarrollo, cantidad de hombres y costo (COCOMO, 2004).

Tabla 48. Factores de escala.

Nombre	Valor	Justificación
PREC	3.72	Presenta aspectos novedosos con respecto a las soluciones existentes a nivel mundial.
FLEX	1.01	Cuenta con alta flexibilidad en el desarrollo.
TEAM	1.10	Existe alta cohesión en el equipo de trabajo.
RESL	4.24	Se identificaron solo tres riesgos críticos.
PMAT	1.56	El equipo de trabajo tiene experiencia previa en proyectos de este tipo.
Total(SF)	**11.63**	

Tabla 49. Multiplicadores de esfuerzos.

Nombre	Valor	Justificación
RCPX	**0.98**	La complejidad del producto es media y la documentación necesaria es moderada.
RUSE	**1.07**	El nivel de reutilización es alto
PDIF	**0.87**	Uso de memoria y almacenamiento mínimo, producto estable.
PREX	**0.90**	Alto grado de experiencia en el desarrollo sobre la plataforma y el lenguaje.
PERS	**0.88**	Alta capacidad del personal
FCIL	**0.82**	Se utilizan entornos de desarrollo integrados y herramientas de modelación que facilitan el trabajo.
SCED	**1.00**	Se empleó el tiempo planificado para el desarrollo del sistema.
Total(EM)	**6.52**	

Cálculos:

A (constante de calibración) = **2.94**

B (factor de escala) = **0.91** * **0.1** * ∑ **SF**

B (factor de escala) = **0.91** * **0.1** * **11.63** ≈ **1.06**

C = **3.67**

D = **0.24**

Esfuerzo nominal (PM nominal**):**

PM nominal = **A** * (Size) ^**B**

PM nominal = **2.94** * (**7.21**) ^**1.06** = **2.94** * **8.12** = **23.87**

Esfuerzo (PM):

∏EM = **RCPX * RUSE * PDIF * PREX * PERS * FCIL * SCED**

∏EM = **0.98** * **1.07** * **0.87** * **0.90** * **0.88** * **0.82** * **1.00 ≈ 0.59**

PM = **PM** nominal ***** **∏EM** = **23.87*** **0.59 ≈ 14.08**

Tiempo calendario del desarrollo (TDEV):

TDEV = **C * PM** (esfuerzo) **^ F**

F = **D** + **0.2** * **0.01** * ∑ **SF** = **0.24** + **0.2** * **0.01** * **11.63** = **0.24**+ **0.02**

F = **0.26**

TDEV = **3.67** * **14.08** ^ **0.26** = **3.67** * **1.99**

TDEV ≈ 7.30

Costo humano (CH):

CH = **PM / TDEV** = **14.08** / **7.30** = **1.93 ≈ 2**

C = **CH * Salario * PM** = **2** * **100** * **14.08**

C = **$2 816**

Tabla 50. Resultados.

Cálculo de:	Valor
Esfuerzo	14.08 hombres/mes
Tiempo de desarrollo	7.3 meses
Cantidad de hombres	2 hombres
Salario medio	$ 100
Costo	$ 2 816

5.4. Beneficios tangibles e intangibles

La utilización de todo tipo de framework y APIs GUI en la actualidad se ha hecho habitual y necesaria dado la facilidad de trabajo, calidad de productos desarrollados y ahorro de tiempo de producción que se logran con su empleo. En la programación para dispositivos móviles, debido a la tecnología cambiante, la rapidez y calidad con que se desarrollen las aplicaciones son dos elementos vitales.

El desarrollo del producto propuesto en esta investigación brinda beneficios tangibles a la empresa PROCYON Soluciones como es el caso del aumento de la calidad de las aplicaciones desarrolladas en cuanto a interfaces gráficas con respecto a las limitantes de la solución existente .Propone además un estándar de implementación distintivo para la entidad así como la reducción considerable de tiempo empleado en la producción. La implementación y configuración de un sistema de menú para un producto en desarrollo consume un tiempo promedio de de 3 a 4 semanas, con la utilización del framework propuesto el mismo se reduciría a 1 o 2 días para un desarrollador. El proyecto propuesto no está concebido para ser un producto comercial, aunque en un futuro pueda reportar beneficios monetarios a la entidad adquiriendo este carácter.

5.5. Análisis de costo

Desarrollar cualquier sistema o producto tiene un costo de producción asociado, la justificación de este costo debe estar orientada en base a los beneficios que reporta o reportará el sistema una vez puesto en funcionamiento. El sistema que se propone en este documento no conlleva a grandes gastos, puesto que sobre su costo solo influye el salario de los desarrolladores que seria recuperable una vez puesto en funcionamiento el *API* reduciendo a un 70-80% el tiempo de desarrollo en las aplicaciones. Mediante este estudio de factibilidad se arriba a la conclusión que la implementación de este sistema es factible. Uno de los principales elementos que presenta este sistema a favor de su factibilidad es la utilización de plataformas y herramientas libres que no requieren el pago de alguna licencia.

5.6. Conclusiones

En este capítulo se realizó un análisis de factibilidad de la solución que se propone, mediante la comparación del precio de costo estimado obtenido con los precios de soluciones similares en el mercado y tomando en cuenta los beneficios de su puesta en funcionamiento, se puede concluir que el desarrollo del sistema es viable.

Conclusiones

- ✓ El desarrollo del *API GUI* resuelve las limitantes presentadas por plataformas utilizadas anteriormente para el desarrollo de aplicaciones *J2ME.*
- ✓ El framework menú desarrollado presenta nuevos componentes visuales y otros que proponen aspectos novedosos respecto a otros *API* de desarrollo de su tipo.
- ✓ Los sistemas de menú producidos con el framework desarrollado son soportables por toda la gama de dispositivos móviles dotados con tecnología Java.
- ✓ El sistema propuesto disminuye aproximadamente a un 20% el tiempo de desarrollo que anteriormente se empleaba para la creación de sistemas de menú.
- ✓ El sistema propone un estándar de implementación en entidades y empresas que se dedican al desarrollo de aplicaciones para dispositivos móviles.

Recomendaciones

- ✓ Establecer el framework desarrollado como estándar de implementación en la entidad Procyon.
- ✓ Continuar con el desarrollo de nuevos componentes GUI para adicionarlos al framework.

Bibliografía

Acebal, Cesar F. 2005. *Extreme Programming (XP): Un nuevo método de desarrollo de software.* España : Universidad de Oviedo, 2005.

Bargen, Bernard. 1998. *A fondo DirectX.* España : McGraw Hill Interamericana de España SA, 1998.

Beck, Kent. 2000. *Extreme Programming Explained.* s.l. : Addison Wesley, 2000.

Cabot, Jordi. 2006. *La relación de materialización en UML.* s.l. : Dept. Llenguatges i Sistemes Informàtics, 2006.

COCOMO, II. 2004. www.ghostamd.googlepages.com. [Online] 2004. [Cited: febrero 6, 20008.] http://ghostamd.googlepages.com/EstimacionesfuerzoII.pdf.

Coltell, Oscar. 2003. *INTRODUCCIÓN AL PROCESO UNIFICADO DE DESARROLLO DE SOFTWARE.* s.l. : Universidad Jaume , 2003.

Daum, Berthold. 2005. *Profesional eclipse 3 para desarrolladores JAVA.* s.l. : Anaya Multimedia. España., 2005.

Dr Sergio Gálvez Rojas, Lucas Ortega Díaz. 2003. *Java a tepe.* s.l. : Sun Microsystem, 2003.

Fowler, M. 2004. www.martinfowler.com. [Online] 2004. [Cited: febrero 14, 2008.] http://www.martinfowler.com/articles/designDead.html.

Fowler. 2005. www.martinfowler.com. [Online] 2005. [Cited: marzo 1, 2008.] http://martinfowler.com/articles/newMethodology.html.

Fowler, M. 2006. www.martinfowler.com. [Online] 2006. [Cited: febrero 14, 2008.] http://martinfowler.com/articles/continuousIntegration.html.

Fuentes, Lidia. 2005. *Una Introducción a los Perfiles UML.* s.l. : Depto. de Lenguajes y Ciencias de la Computación, Universidad de Málaga, 2005.

García, Francisco Javier Pérez. 2002. *Framework para la integración de herramientas de desarrollo basado en reutilización.* s.l. : Departamento de Informática, Universidad de Valladolid, 2002.

García, Juan A. Recio. 2006. *Aprendizaje de técnicas avanzadas de Programación Orientada.* España : Universidad Complutense de Madrid, 2006.

Gamma, Erich, y otros. 1994. *Design Patterns: Elements of Reusable Object-Oriented Software.* s.l. : Addyson-Wesley, 1994.

Gracia, Joaquin. 2005. www.ingenierosoftware.com. *www.ingenierosoftware.com.* [Online] mayo 27, 2005. [Cited: marzo 16, 2008.] http://www.ingenierosoftware.com/analisisydiseno/patrones-diseno.php.

Gutierrez, Jorge A. Saavedra. 2006. www.jorgesaavedra.wordpress.com. *www.jorgesaavedra.wordpress.com.* [Online] octubre 17, 2006. [Cited: marzo 15, 2008.] http://jorgesaavedra.wordpress.com/2006/08/17/patrones-grasp-craig-larman/.

HispaNetwork. 2006. tecnologia.glosario.net. *tecnologia.glosario.net.* [Online] octubre 27, 2006. [Cited: marzo 2, 2008.] http://tecnologia.glosario.net/terminos-tecnicos-internet/api-100.html.

2002. iie.fing.edu.uy. *iie.fing.edu.uy.* [Online] 2002. [Cited: febrero 20, 2008.] http://iie.fing.edu.uy/investigacion/grupos/bicoti/bicoti1/SoftEngineering/softeng03.htm.

Interficies de las Comunidades Virtuales. **Monguet, Dr. Josep M. 2005.** 2005. Interficies de las Comunidades Virtuales.

Isaac, Germán. 2006. www.mastermagazine.info. *www.mastermagazine.info.* [Online] julio 26, 2006. [Cited: marzo 6, 2008.] http://www.mastermagazine.info/articulo/10112.php.

Lago, Ramiro. 2007. www.proactiva-calidad.com. *www.proactiva-calidad.com.* [Online] abril 2007. [Cited: marzo 12, 2008.] http://www.proactiva-calidad.com/java/patrones/index.html.

Master. 2003. www.mastermagazine.info. *www.mastermagazine.info.* [Online] marzo 13, 2003. [Cited: marzo 8, 2008.] http://www.mastermagazine.info/articulo/9254.php.

Merseguer, José. 2006. phpsenior.blogspot.com. *phpsenior.blogspot.com.* [Online] abril 2006. [Cited: marzo 12, 2008.] http://phpsenior.blogspot.com/2006/04/php5-patrn-de-diseo-template-method.html.

Jenui. 2004. *Las Telecomunicaciones y la Movilidad.* s.l. : AHCIET, 2004.

Jost, Sosa. 2004. *Enseñanza de la Técnica de Modelado y Diseño Orientado a Objetos.* s.l. : U N I V E R S I D A D N A C I O N AL D E L N O R D E S T E, 2004.

Kent Beck, M. Fowler. 2000. *Planning Extreme Programming.* s.l. : Addison Wesley, 2000.

Knudsen, Jonathan. 2003. *J2ME para dispositivos móviles.* s.l. : Ed.Apress, 2003. pp. 46-50.

Letelier, P. 2004. *Metodologías ágiles para el desarrollo de software.* 2004.

Markiewicz, Marcus Eduardo. 2001. *El Desarrollo del Framework Orientado al Objeto.* Brasil : PUC-Rio, 2001.

Moreno, Ana Mª. 2005. www.trevinca.ei.uvigo.es. [Online] 2005. trevinca.ei.uvigo.es/~cfajardo/Nueva_carpeta/presentaciones/cocomo2k.pdf.

Polish, Primeros pasos con J2ME. 2007. *Aitor Almeida Escondrillas.* s.l. : Dialnet, 2007.

REYES, ADAY DEL SOL. 08. Cubahora. [Online] 05 09, 08. [Cited: 05 15, 08.] http://cubahora.co.cu/index.php?tpl=principal/ver-noticias/ver-not_soc.tpl.html&newsid_obj_id=1025444.

Rodrigues, Jorge Nascimento. 2002. www.mujeresdeempresa.com. *www.mujeresdeempresa.com.* [Online] agosto 15, 2002. [Cited: febrero 16, 2008.] http://www.mujeresdeempresa.com/management/management020803.shtml.

Senso, J. 2003. www.scielo.br. *www.scielo.br.* [Online] 2003. [Cited: marzo 1, 2008.] http://www.scielo.br/pdf/ci/v32n2/17038.pdf.

Serantes, Luis Miguel Cubillo. 2008. blog-java.informaticos.com/. *blog-java.informaticos.com/.* [Online] enero 28, 2008. [Cited: marzo 20, 2008.] http://blog-java.informaticos.com/2008/1/30/un-breve-recorrido-j2me-polish.

Serrano, Alberto García. 2003. *PROGRAMACIÓN DE JUEGOS PARA MÓVILES.* 2003.

Serrano, J. 2007. elprofesionaldelainformacion.metapress.com. *elprofesionaldelainformacion.metapress.com.* [Online] 2007. [Cited: febrero 25, 2008.] http://elprofesionaldelainformacion.metapress.com/app/home/contribution.asp?referrer=parent&backto=issue,5,14;journal,7,59;linkingpublicationresults,1:105302,1.

Telecom. 2005. www.pc-news.com. *www.pc-news.com.* [Online] septiembre 19, 2005. [Cited: mayo 29, 2008.] http://www.pc-news.com/detalle.asp?sid=&id=4&Ida=2144.

PC-News. 2001. www.pc-news.com. *www.pc-news.com.* [Online] julio 17, 2001. [Cited: febrero 12, 2008.] http://www.pc-news.com/detalle.asp?sid=4&id=39&Ida=388.

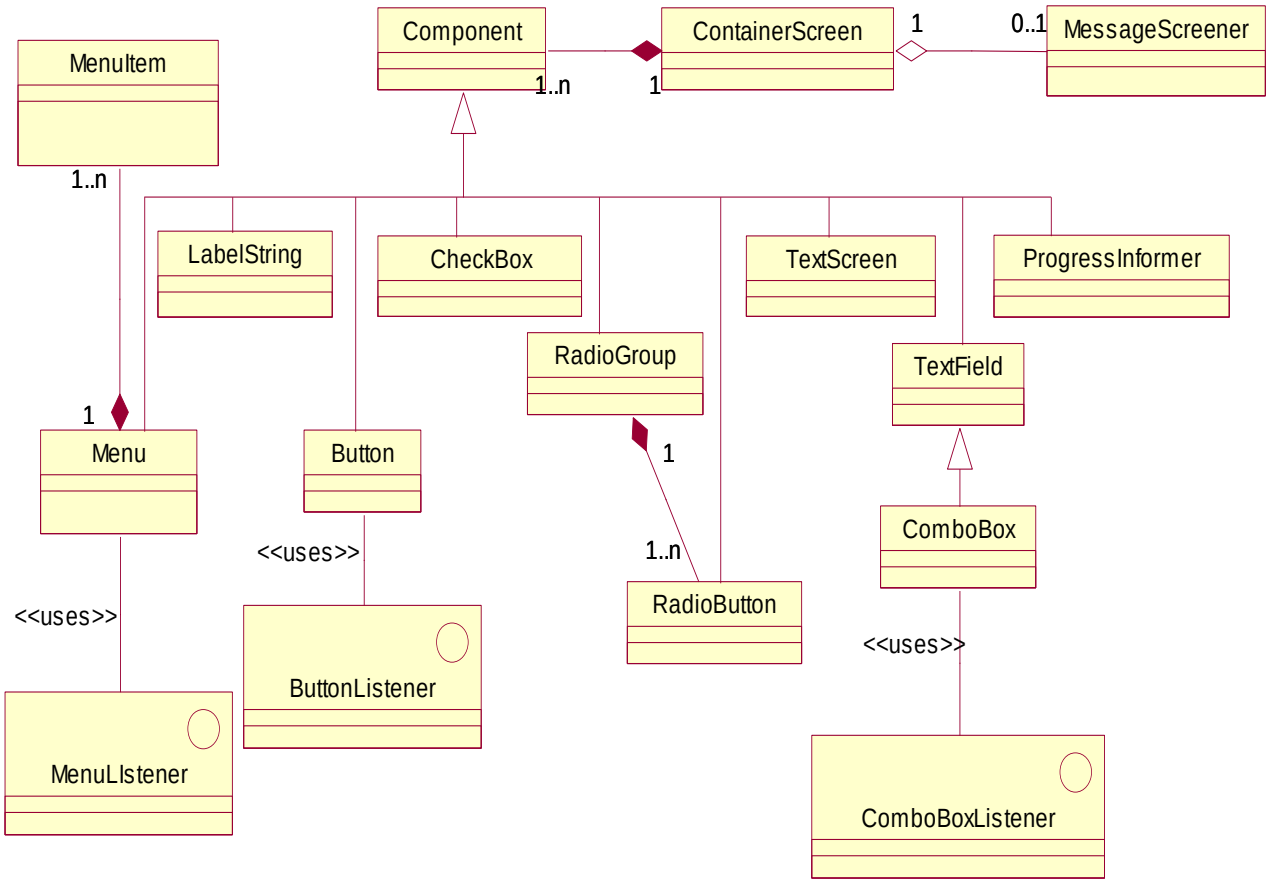
terminales, Aplicaciones de tratamiento de imagen en. 2005. *Celeste Campo Vázquez.* España : javahispano, 2005.

Tigris.org. 2000. www.subversion.tigris.org. *www.subversion.tigris.org.* [Online] 2000. [Cited: febrero 10, 2008.] http://subversion.tigris.org.

XP. www.extremeprogramming.org. *www.extremeprogramming.org.* [Online] [Cited: marzo 2008, 10.] http://www.extremeprogramming.org/.

Anexos

Anexo I Diagrama de Clases



Anexo II Pruebas de Aceptación

Caso de Prueba de Aceptación	
Código: HU1_P1	**Historia de Usuario:** Creación del contenedor de componentes.
Nombre: Configuración del contenedor de componentes.	
Descripción: Prueba la funcionalidad de configuración del contenedor en cuanto a diseño para ajustarse a las necesidades del cliente.	
Condiciones de Ejecución: El contenedor de componentes debe estar creado e inicializado.	
Entrada / Pasos de Ejecución: Se configura los colores de fondo, encabezado y la fuente que se utilizara dentro del contenedor.	
Resultado Esperado: El contenedor de componentes es configurado sin errores.	
Evaluación de la Prueba: Prueba satisfactoria.	

Caso de Prueba de Aceptación	
Código: HU1_P2	**Historia de Usuario:** Creación del contenedor de componentes.
Nombre: Diseño mediante imágenes del contenedor de componentes.	
Descripción: Prueba la funcionalidad de configuración del contenedor en cuanto a diseño mediante imágenes para ajustarse a las necesidades del cliente.	
Condiciones de Ejecución: El contenedor de componentes debe estar creado e inicializado.	
Entrada / Pasos de Ejecución: Se configura el fondo y el encabezado mediante imágenes.	
Resultado Esperado: El contenedor de componentes es configurado sin errores.	
Evaluación de la Prueba: Prueba satisfactoria.	

Caso de Prueba de Aceptación	
Código: HU1_P3	**Historia de Usuario:** Creación del contenedor de componentes.
Nombre: Adicionar componentes visuales al contenedor.	
Descripción: Prueba la funcionalidad de agrupar a uno o más componentes visuales.	
Condiciones de Ejecución: El contenedor de componentes debe estar creado e inicializado.	
Entrada / Pasos de Ejecución: Se adiciona uno o más componentes visuales.	
Resultado Esperado: La navegación entre los componentes visuales insertados se realizo sin errores.	
Evaluación de la Prueba: Prueba satisfactoria.	

Caso de Prueba de Aceptación	
Código: HU1_P4	**Historia de Usuario:** Creación del contenedor de componentes.
Nombre: Seleccionar un componente visual agrupado en el contenedor.	
Descripción: Prueba la funcionalidad del contenedor de manejar los eventos generados por los componentes visuales insertados por individual.	
Condiciones de Ejecución: El contenedor de componentes debe estar creado e inicializado.	
Entrada / Pasos de Ejecución: Se adiciona uno o más componentes visuales. Se selecciona un componente visual adicionado en el contenedor y se verifica el tratamiento de los eventos generados por el mismo.	
Resultado Esperado: El contenedor es capaz de manejar los eventos generados por cada componente visual sin errores.	
Evaluación de la Prueba: Prueba satisfactoria.	

Caso de Prueba de Aceptación	
Código: HU2_P1	**Historia de Usuario:** Creación del componente Button.
Nombre: Configuración del componente Button.	
Descripción: Prueba la funcionalidad de configuración del componente Button en cuanto a diseño para ajustarse a las necesidades del cliente.	
Condiciones de Ejecución: El contenedor de componentes debe estar creado e inicializado.	
Entrada / Pasos de Ejecución: Se configura los colores y la fuente que se utilizará en el componente Button.	
Resultado Esperado: El componente Button es configurado sin errores.	
Evaluación de la Prueba: Prueba satisfactoria.	

Caso de Prueba de Aceptación	
Código: HU2_P2	**Historia de Usuario:** Creación del componente Button.
Nombre: Diseño mediante imágenes del componente Button.	
Descripción: Prueba la funcionalidad de configuración del componente Button en cuanto a diseño mediante imágenes para ajustarse a las necesidades del cliente.	
Condiciones de Ejecución: El contenedor de componentes debe estar creado e inicializado.	
Entrada / Pasos de Ejecución: Se configura mediante imágenes el diseño visual del componente Button.	
Resultado Esperado: El componente Button es configurado sin errores.	
Evaluación de la Prueba: Prueba satisfactoria.	

Caso de Prueba de Aceptación	
Código: HU2_P3	**Historia de Usuario:** Creación del componente Button.
Nombre: Manejo de los eventos generados por el componente Button.	
Descripción: Prueba la funcionalidad de configuración de la interfaz del componente Button para el manejo específico de un evento generado.	
Condiciones de Ejecución: El contenedor de componentes debe estar creado e inicializado. El componente Button debe estar creado y adicionado al contenedor de componentes.	
Entrada / Pasos de Ejecución: Se configura la interfaz del componente Button para el manejo específico de un evento generado. Se selecciona el componente Button para generar el evento para el cual se configuró la interfaz.	
Resultado Esperado: El manejo de eventos en el componente Button es configurado sin errores.	
Evaluación de la Prueba: Prueba satisfactoria.	

Caso de Prueba de Aceptación	
Código: HU3_P1	**Historia de Usuario:** Creación del componente Menu
Nombre: Creación y configuración del componente Menu tipo texto	
Descripción: Prueba la capacidad de configuración del componente al crear un Menu de texto y que se cree correctamente cambiando los parámetros de entradas.	
Condiciones de Ejecución: El contenedor de componentes debe estar creado e inicializado.	
Entrada / Pasos de Ejecución: Se crea una instancia de menú del tipo texto y se incorpora dentro de un contenedor, se cambian y combinan los parámetros fundamentales (nombre, alineación, color de elemento seleccionado, color de elemento no seleccionado).	
Resultado Esperado: El componente funciona sin errores.	

Caso de Prueba de Aceptación	
Código: HU3_P2	**Historia de Usuario:** Creación del componente Menu
Nombre: Creación y configuración del componente Menu tipo imagen	
Descripción: Prueba la capacidad de configuración del componente de imágenes y que se cree correctamente cambiando los parámetros de entradas.	
Condiciones de Ejecución: El contenedor de componentes debe estar creado e inicializado.	
Entrada / Pasos de Ejecución: Se crea una instancia de Menu del tipo imagen y se incorpora dentro de un contenedor, se cambian y combinan los parámetros fundamentales (nombre, cantidad de filas, cantidad de columnas, color de elemento seleccionado).	
Resultado Esperado: El componente funciona sin errores.	
Evaluación de la Prueba: Prueba satisfactoria.	

Caso de Prueba de Aceptación	
Código: HU3_P3	**Historia de Usuario:** Creación del componente Menu
Nombre: Agregar ítems al Menu	
Descripción: Prueba la capacidad del componente de incorporar nuevos elementos.	
Condiciones de Ejecución: El contenedor de componentes debe estar creado e inicializado.	
Entrada / Pasos de Ejecución: Se crea una instancia de Menu y se le insertan elementos con diferentes propiedades	
Resultado Esperado: El componente muestra una lista de opciones a seleccionar.	
Evaluación de la Prueba: Prueba satisfactoria.	

Caso de Prueba de Aceptación	
Código: HU3_P4	**Historia de Usuario:** Creación del componente Menu
Nombre: Recibir eventos generados por la acción sobre el Menu	
Descripción: Prueba la capacidad del componente de manejar los eventos relacionados a él (selección de opciones).	
Condiciones de Ejecución: El contenedor de componentes debe estar creado e inicializado. El menú debe estar creado inicializado y debe contener al menos un elemento.	
Entrada / Pasos de Ejecución: Se acciona sobre un elemento del Menu.	
Resultado Esperado: El componente genera correctamente el evento correspondiente a la opción seleccionada.	
Evaluación de la Prueba: Prueba satisfactoria.	

Caso de Prueba de Aceptación	
Código: HU4_P1	**Historia de Usuario:** Creación del componente TextField.
Nombre: Configuración del componente TextField.	
Descripción: Prueba la funcionalidad de configuración del componente TextField en cuanto a diseño para ajustarse a las necesidades del cliente.	
Condiciones de Ejecución: El contenedor de componentes debe estar creado e inicializado.	
Entrada / Pasos de Ejecución: Se configura los colores y la fuente que se utilizarán en el componente TextField.	
Resultado Esperado: El componente TextField es configurado sin errores.	
Evaluación de la Prueba: Prueba satisfactoria.	

Caso de Prueba de Aceptación	
Código: HU4_P2	**Historia de Usuario:** Creación del componente TextField.
Nombre: Configuración del componente TextField para el manejo de datos tipo ***TEXT*** *(texto)*.	
Descripción: Prueba la funcionalidad de configuración del componente TextField para el manejo del tipo de dato ***TEXT*** *(texto)* en cuanto a diseño.	
Condiciones de Ejecución: El contenedor de componentes debe estar creado e inicializado.	
Entrada / Pasos de Ejecución: Se configura el componente TextField para el manejo del tipo de dato ***TEXT*** *(texto).*	
Resultado Esperado: El componente TextField es configurado sin errores.	
Evaluación de la Prueba: Prueba satisfactoria.	

Caso de Prueba de Aceptación	
Código: HU4_P3	**Historia de Usuario:** Creación del componente TextField.
Nombre: Manejo de eventos del componente TextField para el tipo ***TEXT*** *(texto)*.	
Descripción: Prueba la funcionalidad del manejo de eventos del componente TextField para el tipo de dato ***TEXT*** *(texto)* en específico.	
Condiciones de Ejecución: El contenedor de componentes debe estar creado e inicializado.	
Entrada / Pasos de Ejecución: Se configura el componente TextField para el manejo del tipo de dato ***TEXT*** *(texto).* Se selecciona el componente TextField introduciéndole datos de tipo ***TEXT*** *(texto)* para el manejo de eventos generados.	
Resultado Esperado: El manejo de eventos en el componente TextField para el tipo de dato ***TEXT*** *(texto)* transcurre sin errores.	
Evaluación de la Prueba: Prueba satisfactoria.	

Caso de Prueba de Aceptación	
Código: HU4_P4	**Historia de Usuario:** Creación del componente TextField.
Nombre: Configuración del componente TextField para el manejo de datos tipo ***DATE*** *(fechas).*	
Descripción: Prueba la funcionalidad de configuración del componente TextField para el manejo del tipo de dato ***DATE*** *(fechas)* en cuanto a diseño.	
Condiciones de Ejecución: El contenedor de componentes debe estar creado e inicializado.	
Entrada / Pasos de Ejecución: Se configura el componente TextField para el manejo del tipo de dato ***DATE*** *(fechas).*	
Resultado Esperado: El componente TextField es configurado sin errores.	
Evaluación de la Prueba: Prueba satisfactoria.	

Caso de Prueba de Aceptación	
Código: HU4_P5	**Historia de Usuario:** Creación del componente TextField.
Nombre: Manejo de eventos del componente TextField para el tipo ***DATE*** *(fechas).*	
Descripción: Prueba la funcionalidad del manejo de eventos del componente TextField para el tipo de dato ***DATE*** *(fechas)* en específico.	
Condiciones de Ejecución: El contenedor de componentes debe estar creado e inicializado.	
Entrada / Pasos de Ejecución: Se configura el componente TextField para el manejo del tipo de dato ***DATE*** *(fechas).* Se selecciona el componente TextField introduciéndole datos de tipo ***DATE*** *(fechas)* para el manejo de eventos generados.	
Resultado Esperado: El manejo de eventos en el componente TextField para el tipo de dato ***DATE*** *(fechas)* transcurre sin errores.	

Caso de Prueba de Aceptación	
Código: HU4_P6	**Historia de Usuario:** Creación del componente TextField.
Nombre: Configuración del componente TextField para el manejo de datos tipo ***PASS*** *(contraseñas).*	
Descripción: Prueba la funcionalidad de configuración del componente TextField para el manejo del tipo de dato ***PASS*** *(contraseñas)* en cuanto a diseño.	
Condiciones de Ejecución: El contenedor de componentes debe estar creado e inicializado.	
Entrada / Pasos de Ejecución: Se configura el componente TextField para el manejo del tipo de dato ***PASS*** *(contraseñas).*	
Resultado Esperado: El componente TextField es configurado sin errores.	
Evaluación de la Prueba: Prueba satisfactoria.	

Caso de Prueba de Aceptación	
Código: HU4_P7	**Historia de Usuario:** Creación del componente TextField.
Nombre: Manejo de eventos del componente TextField para el tipo ***PASS*** *(contraseñas).*	
Descripción: Prueba la funcionalidad del manejo de eventos del componente TextField para el tipo de dato ***PASS*** *(contraseñas)* en específico.	
Condiciones de Ejecución: El contenedor de componentes debe estar creado e inicializado.	
Entrada / Pasos de Ejecución: Se configura el componente TextField para el manejo del tipo de dato ***PASS*** *(contraseñas).* Se selecciona el componente TextField introduciéndole datos de tipo ***PASS*** *(contraseñas)* para el manejo de eventos generados.	
Resultado Esperado: El manejo de eventos en el componente TextField para el tipo de dato ***PASS*** *(contraseñas)* transcurre sin errores.	
Evaluación de la Prueba: Prueba satisfactoria.	

Caso de Prueba de Aceptación	
Código: HU4_P8	**Historia de Usuario:** Creación del componente TextField.
Nombre: Configuración del componente TextField para el manejo de datos tipo ***CHAT*** *(mensajes de chat).*	
Descripción: Prueba la funcionalidad de configuración del componente TextField para el manejo del tipo de dato ***CHAT*** *(mensajes de chat)* en cuanto a diseño.	
Condiciones de Ejecución: El contenedor de componentes debe estar creado e inicializado.	
Entrada / Pasos de Ejecución: Se configura el componente TextField para el manejo del tipo de dato ***CHAT*** *(mensajes de chat).*	
Resultado Esperado: El componente TextField es configurado sin errores	
Evaluación de la Prueba: Prueba satisfactoria.	

Caso de Prueba de Aceptación	
Código: HU4_P9	**Historia de Usuario:** Creación del componente TextField.
Nombre: Manejo de eventos del componente TextField para el tipo ***CHAT*** *(mensajes de chat).*	
Descripción: Prueba la funcionalidad del manejo de eventos del componente TextField para el tipo de dato ***CHAT*** *(mensajes de chat)* en específico.	
Condiciones de Ejecución: El contenedor de componentes debe estar creado e inicializado.	
Entrada / Pasos de Ejecución: Se configura el componente TextField para el manejo del tipo de dato ***CHAT*** *(mensajes de chat).* Se selecciona el componente TextField introduciéndole datos de tipo ***CHAT*** *(mensajes de chat)* para el manejo de eventos generados.	
Resultado Esperado: El manejo de eventos en el componente TextField para el tipo de dato ***PASS*** *(contraseñas)* transcurre sin errores.	
Evaluación de la Prueba: Prueba satisfactoria.	

Caso de Prueba de Aceptación	
Código: HU5_P1	**Código:** Creación del componente CheckBox.
Nombre: Configuración del componente CheckBox.	
Descripción: Prueba la funcionalidad de configuración del componente CheckBox en cuanto a diseño para ajustarse a las necesidades del cliente.	
Condiciones de Ejecución: El contenedor de componentes debe estar creado e inicializado.	
Entrada / Pasos de Ejecución: Se configura los colores y la fuente que se utilizará en el componente CheckBox.	
Resultado Esperado: El componente CheckBox es configurado sin errores.	
Evaluación de la Prueba: Prueba satisfactoria.	

Caso de Prueba de Aceptación	
Código: HU5_P2	**Historia de Usuario:** Creación del componente CheckBox.
Nombre: Manejo visual de la propiedad check del componente CheckBox.	
Descripción: Prueba la funcionalidad de ser seleccionado o no del componente CheckBox en cuanto a diseño visual.	
Condiciones de Ejecución: El contenedor de componentes debe estar creado e inicializado. El componente CheckBox debe estar adicionado en el contenedor de componentes.	
Entrada Pasos de Ejecución: Se selecciona como marcado y no marcado al componente CheckBox.	
Resultado Esperado: El componente CheckBox es marcado sin errores.	
Evaluación de la Prueba: Prueba satisfactoria.	

Caso de Prueba de Aceptación	
Código: HU6_P1	**Historia de Usuario:** Creación del componente RadioButton
Nombre: Configuración del componente RadioButton.	
Descripción: Prueba la funcionalidad de configuración del componente RadioButton en cuanto a diseño para ajustarse a las necesidades del cliente	
Condiciones de Ejecución: El contenedor de componentes debe estar creado e inicializado.	
Entrada / Pasos de Ejecución: Se configura los colores y la fuente que se utilizará en el componente.	
Resultado Esperado: El componente debe mostrarse de acuerdo a la configuración programada.	
Evaluación de la Prueba: Prueba satisfactoria.	

Caso de Prueba de Aceptación	
Código: HU6_P2	**Historia de Usuario:** Creación del componente RadioButton
Nombre: Manejo visual de la propiedad “check” del componente RadioButton.	
Descripción: Prueba la funcionalidad de ser seleccionado o no del componente RadioButton en cuanto a diseño visual.	
Condiciones de Ejecución: El contenedor de componentes debe estar creado e inicializado. El componente RadioButton debe estar adicionado al contenedor de componentes.	
Entrada / Pasos de Ejecución: Se selecciona como marcado y no marcado al componente.	
Resultado Esperado: El componente debe seleccionarse o deseleccionarse correctamente según la acción.	
Evaluación de la Prueba: Prueba satisfactoria.	

Caso de Prueba de Aceptación	
Código: HU7_P1	**Historia de Usuario:** Creación del componente RadioGroup
Nombre: Configuración del componente RadioGroup.	
Descripción: Prueba la funcionalidad de configuración del componente RadioButton en cuanto a diseño para ajustarse a las necesidades del cliente.	
Condiciones de Ejecución: El contenedor de componentes debe estar creado e inicializado. Se deben crear componentes RadioButton para adicionarlos al RadioGroup.	
Entrada / Pasos de Ejecución: Se crea el componente RadioGroup. Se configura el diseño del componente RadioGroup. Se adicionan al menos dos componentes RadioButton al RadioGroup.	
Resultado Esperado: El componente se muestra en correspondencia a la configuración que se le ha programado.	
Evaluación de la Prueba: Prueba satisfactoria.	

Caso de Prueba de Aceptación	
Código: HU7_P2	**Historia de Usuario:** Creación del componente RadioGroup
Nombre: Agregar RadioButton al grupo.	
Descripción: Prueba la funcionalidad de agregar nuevos elementos al componente.	
Condiciones de Ejecución: El contenedor de componentes debe estar creado e inicializado. Se deben crear componentes RadioButton para adicionarlos al RadioGroup.	
Entrada / Pasos de Ejecución: Se adicionan al menos dos componentes RadioButton al RadioGroup.	
Resultado Esperado: El componente muestra un conjunto de elementos de los cuales solo se puede seleccionar uno.	
Evaluación de la Prueba: Prueba satisfactoria.	

Caso de Prueba de Aceptación	
Código: HU8_P1	**Historia de Usuario:** Creación del componente MessageScreener.
Nombre: Configuración del componente MessageScreener.	
Descripción: Prueba la funcionalidad de configuración del componente MessageScreener en cuanto a diseño para ajustarse a las necesidades del cliente.	
Condiciones de Ejecución: El contenedor de componentes debe estar creado e inicializado.	
Entrada / Pasos de Ejecución: Se configura los colores y la fuente que se utilizará en el componente MessageScreener.	
Resultado Esperado: El componente MessageScreener es configurado sin errores.	
Evaluación de la Prueba: Prueba satisfactoria.	

Caso de Prueba de Aceptación	
Código: HU8_P2	**Historia de Usuario:** Creación del componente MessageScreener.
Nombre: Diseño mediante imágenes del componente MessageScreener.	
Descripción: Prueba la funcionalidad de configuración del componente MessageScreener en cuanto a diseño mediante imágenes para ajustarse a las necesidades del cliente.	
Condiciones de Ejecución: El contenedor de componentes debe estar creado e inicializado.	
Entrada / Pasos de Ejecución: Se configura mediante imágenes el diseño visual del componente MessageScreener.	
Resultado Esperado: El componente MessageScreener es configurado sin errores.	
Evaluación de la Prueba: Prueba satisfactoria.	

Caso de Prueba de Aceptación	
Código: HU9_P1	**Historia de Usuario:** Creación del componente TextScreen
Nombre: Configuración del componente TextScreen.	
Descripción: Prueba la funcionalidad de configuración del componente TextScreen en cuanto a diseño para ajustarse a las necesidades del cliente.	
Condiciones de Ejecución: El contenedor de componentes debe estar creado e inicializado.	
Entrada / Pasos de Ejecución: Se crea el componente TextScreen. Se configura el diseño del componente TextScreen. Se adiciona el componente al contenedor.	
Resultado Esperado: El componente se muestra en correspondencia a la configuración que se le ha programado.	
Evaluación de la Prueba: Prueba satisfactoria.	

Caso de Prueba de Aceptación	
Código: HU9_P2	**Historia de Usuario:** Creación del componente TextScreen
Nombre: Prueba de funcionamiento del scroll.	
Descripción: Prueba la funcionalidad del componente TextScreen con textos largos.	
Condiciones de Ejecución: El contenedor de componentes debe estar creado e inicializado. Se adiciona el componente al contenedor, se muestra y se trata de leer todo el texto.	
Resultado Esperado: El componente debe mostrar una señal de que hay más texto del que se muestra en la pantalla, y al presionar las teclas del joystick hacia arriba y abajo el texto debe desplazarse.	
Evaluación de la Prueba: Prueba satisfactoria.	

Caso de Prueba de Aceptación	
Código: HU9_P3	**Historia de Usuario:** Creación del componente TextScreen
Nombre: Prueba de casos limites.	
Descripción: Prueba la adaptabilidad del componente TextScreen ante casos críticos (letra más grande, letra más pequeña).	
Condiciones de Ejecución: El contenedor de componentes debe estar creado e inicializado.	
Entrada / Pasos de Ejecución: Se crea el componente TextScreen. Se configura el componente con los casos críticos. Se adiciona el componente al contenedor y se muestra.	
Resultado Esperado: El componente debe adaptarse a la pantalla independientemente del tamaño de la letra.	
Evaluación de la Prueba: Prueba satisfactoria.	

Caso de Prueba de Aceptación	
Código: HU10_P1	**Historia de Usuario:** Creación del componente LabelString.
Nombre: Configuración del componente LabelString.	
Descripción: Prueba la funcionalidad de configuración del componente LabelString en cuanto a diseño para ajustarse a las necesidades del cliente.	
Condiciones de Ejecución: El contenedor de componentes debe estar creado e inicializado.	
Entrada / Pasos de Ejecución: Se configura los colores y la fuente que se utilizará en el componente LabelString.	
Resultado Esperado: El componente LabelString es configurado sin errores	
Evaluación de la Prueba: Prueba satisfactoria.	

Caso de Prueba de Aceptación	
Código: HU11_P1	**Historia de Usuario:** Creación del componente ComboBox.
Nombre: Configuración del componente ComboBox.	
Descripción: Prueba la funcionalidad de configuración del componente ComboBox en cuanto a diseño para ajustarse a las necesidades del cliente.	
Condiciones de Ejecución: Se configura los colores y la fuente que se utilizará en el componente ComboBox.	
Entrada / Pasos de Ejecución: Se configura los colores y la fuente que se utilizará en el componente ComboBox.	
Resultado Esperado: El componente ComboBox es configurado sin errores.	
Evaluación de la Prueba: Prueba satisfactoria.	

Caso de Prueba de Aceptación	
Código: HU11_P2	**Historia de Usuario:** Creación del componente ComboBox.
Nombre: Diseño mediante imágenes del componente ComboBox.	
Descripción: Prueba la funcionalidad de configuración del componente ComboBox en cuanto a diseño mediante imágenes para ajustarse a las necesidades del cliente.	
Condiciones de Ejecución: El contenedor de componentes debe estar creado e inicializado.	
Entrada / Pasos de Ejecución: Se configura mediante imágenes el diseño visual del componente ComboBox.	
Resultado Esperado: El componente ComboBox es configurado sin errores.	
Evaluación de la Prueba: Prueba satisfactoria.	

Caso de Prueba de Aceptación	
Código: HU11_P3	**Historia de Usuario:** Creación del componente ComboBox.
Nombre: Manejo de los eventos generados por el componente ComboBox.	
Descripción: Prueba la funcionalidad de configuración de la interfaz del componente ComboBox para el manejo específico de un evento generado.	
Condiciones de Ejecución: El contenedor de componentes debe estar creado e inicializado. El componente ComboBox debe estar adicionado al contenedor de componentes.	
Entrada / Pasos de Ejecución: Se configura la interfaz del componente ComboBox para el manejo específico de un evento generado. Se selecciona un elemento del ComboBox para generar el evento para el cual se configuró la interfaz.	
Resultado Esperado: El manejo de eventos en el componente ComboBox es configurado sin errores.	
Evaluación de la Prueba: Prueba satisfactoria.	

Caso de Prueba de Aceptación	
Código: HU12_P1	**Historia de Usuario:** Creación del componente ProgressInformer.
Nombre: Configuración del componente ProgressInformer.	
Descripción: Prueba la funcionalidad de configuración del componente ProgressInformer en cuanto a diseño para ajustarse a las necesidades del cliente.	
Condiciones de Ejecución: El contenedor de componentes debe estar creado e inicializado.	
Entrada / Pasos de Ejecución: Se crea el componente ProgressInformer. Se configura el diseño del componente ProgressInformer. Se adiciona el componente al contenedor.	
Resultado Esperado: El componente debe mostrarse en correspondencia a la configuración que se le ha programado.	
Evaluación de la Prueba: Prueba satisfactoria.	

Glosario de Términos

CSS (del inglés Cascading Style Sheets): Es un formato usado en las páginas web para separar el estilo (la forma en la que se ve una página web) de la estructura (o código), es una característica del HTML que da más control a los programadores, diseñadores y una característica del HTML que da más control a los programadores, diseñadores y usuarios de la web sobre cómo se muestran las páginas web.

C++: Versión de C orientada a objetos creada por Bjarne Stroustrup. C++ se ha popularizado porque combina la programación tradicional en C con programación orientada a objetos.

C: Es un lenguaje de programación de propósito general que ofrece economía sintáctica, control de flujo y estructuras sencillas y un buen conjunto de operadores. No es un lenguaje de muy alto nivel y más bien un lenguaje pequeño, sencillo y no está especializado en ningún tipo de aplicación.

CVS (del inglés Concurrent Versions System): Herramienta que permite que varios programadores trabajen de forma colaborativa en un mismo proyecto llevando un control de las versiones de los ficheros. De esta forma se permiten cambios concurrentes en un mismo fichero sin perder lo cambios realizados.

IDE (del inglés Integrated Development Environment): Programa compuesto por un conjunto de herramientas para un programador. Entorno de programación que ha sido empaquetado como un programa de aplicación, es decir, consiste en un editor de código, un compilador, un depurador y un constructor de interfaz gráfica GUI.

Java: A principios de los años '90 la firma Sun Microsystems desarrolló el lenguaje de programación Java. Este lenguaje orientado a objetos es compilado generalmente en un bytecode, pero también es posible crear código de máquina nativo.

J2SE (del inglés Java 2 Standard Edition): Es una colección de APIs del lenguaje de programación Java útiles para muchos programas de la Plataforma Java.

Kernel: Núcleo. Parte fundamental de un programa, por lo general de un sistema operativo, que reside en memoria todo el tiempo y que provee los servicios básicos.

LCD (del inglés Liquid Cristal Display): Las pantallas de cristal líquido se utilizan principalmente en la construcción de ordenadores portátiles, en los que el tamaño y el peso son dos premisas esenciales. La tecnología LCD permite fabricar pantallas muy finas, pero que no tienen una excesiva calidad de imagen y que, además, presentan problemas de visualización al salirse de ciertos ángulos de visión.

MIDI (del inglés Musical Instrument Digital Interface): Conjunto de estándares de 128 sonidos para tarjetas y dispositivos de sonido MIDI (sintetizadores, módulos de sonido, etc.).

MP3: Llamado exactamente MPEG-1 Audio Layer 3, es una forma de codificar audio usando un algoritmo de compresión que genera pérdida de datos reduciendo la cantidad de información requerida para representar una grabación de audio, pero de modo en que a su vez sea muy similar a la calidad del archivo original para la mayoría de los oyentes.

OMG (del inglés Object Management Group): Es un consorcio a nivel internacional que integra a los principales representantes de la industria de la tecnología de información orientada a objeto.

PDA (del inglés Personal Digital Assistant): Es un dispositivo de pequeño tamaño que combina un ordenador, teléfono/fax, Internet y conexiones de red. Organizador personal con memoria entre 2 y 16 MB que funciona como un miniordenador.

ROM (del inglés Read Only Memory): Al español Memoria de Solo Lectura. Se le denomina así, debido a que los datos almacenados en ella no pueden ser modificados, y generalmente está ubicada en la tarjeta del sistema.

SVN (Subversion): Aplicación para el control de versiones que permite gestionar los cambios y versiones que se realizan durante el desarrollo, de una forma sencilla.

UML (del inglés Unified Modeling Language): Lenguaje unificado de modelado para sistemas de software más conocido y utilizado en la actualidad.