Universidad de las Ciencias Informáticas

Facultad 3

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

Título: Ingeniería de Requisitos del subsistema Administración y Gobierno del Proyecto Tribunales Populares Cubanos.

Autora: Ligia María Caballero Piñeiro

Tutora: Ing. Daylen Benítez Matos

Asesora: Ing. María del Carmen Lerma Tejada

Ciudad de la Habana

Curso 2010-2011

DECLARACIÓN DE AUTORÍA

Declaro ser la autora de la presente tesis y reconozco a la Universidad de las Ciencias Informáticas los derechos patrimoniales de la misma, con carácter exclusivo.

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

_____

Firma de la Autora

_____ _____

Firma de la Tutora Firma de la Asesora

DEDICATORIA

A Tetera,
ojalá estuvieras a mi lado en este momento.

A mami y a papi,
a ustedes mi amor infinito.

A Carlitos,
mi bebé bonito, sin ti no sé qué sería de mí.

AGRADECIMIENTOS

A Teterita por ser mi segunda madre, por complacerme en todos mis caprichos y aguantarme mis malcriadeces, por alegrar mis días. Te quiero y te querré por siempre.

A mi Papi: por hacerme querer lo imposible, eres mi ejemplo a seguir. Te adoro con la vida.

A mi mamita: por querer hacer de mí siempre una mejor persona, por su amor profundo, su cariño y ternura. ¿Sabías q eres la mejor mamá del mundo?

A mi hermanito Carlitos: por hacerme reír y llorar, eres lo mejor que me ha pasado en la vida. Te quiero un mundo y pase lo que pase siempre estaré a tu lado.

A mis abuelitas Mininga y Nereida: por apoyarme y animarme cuando el camino se hacía difícil. Por sus consejos y su amor.

A mi tía Nereidita: por ser como otra mamá en mi vida, en muchos sentidos este sueño es hoy realidad gracias a ti. Te quiero mucho.

A Lalito, Daysi y Liliet, a Marta y a tío Richi: por estar ahí cuando los necesité, por su cariño y apoyo.

A mis tutoras Daylen y María: por su apoyo incondicional y su paciencia infinita. Aprendí mucho de ustedes.

A Lestercito: lo mejor que me ha pasado en estos 5 años. Por comprenderme y hacerme reír como una niña chiquita. Te amo mucho, como nunca podré amar a nadie más.

A Yailencita: te quiero mucho bobiña, gracias por estar cuando más lo necesitaba.

A todas las nuevas amistades que hice en la UCI, principalemente las que me han acompañado desde primer año: María, Aliesky, Leanet, Mailín, Yule, Nani, Danae Anabel, Roxana en fin todas. Nunca los olvidaré.

A Diosito por su protección y ayuda.

Resumen

En el presente trabajo se realizan las etapas de la Ingeniería de Requisitos (IR) propuestas por Roger S. Pressman, para lograr un entendimiento entre los clientes y desarrolladores del subsistema Administración y Gobierno del proyecto Tribunales Populares Cubanos, el cual se encargará de la creación de la estructura y composición de los tribunales así como todo lo referente a los usuarios, roles y permisos que serán asignados para interactuar con el sistema. Donde se generarán los artefactos necesarios que contribuyan al diseño de dicho subsistema.

Para llevar a cabo dicho propósito se utiliza a RUP[1] como metodología de desarrollo de software, UML[2] como lenguaje de modelado, Visual Paradigm como herramienta CASE[3] y Axure RP Pro 5.5 como herramienta de modelado de prototipos no funcionales. Además se aplican técnicas y métricas de software con el objetivo de verificar la calidad de los principales artefactos de software obtenidos y el método de Kano para medir la satisfacción del cliente.

A partir de esta propuesta y con su seguimiento, se espera contribuir al desarrollo de una solución informática capaz de lograr la integración y control de la información generada en los Tribunales Populares Cubanos, ayudando a impartir justicia de manera ética, eficaz y humanista.

Palabras claves: Artefactos de Software, Requisitos de Software, Ingeniería de Requisitos, Métricas de Software, Tribunales, Herramientas, Kano, Metodología.

[1] RUP (Rational Unified Procces) Proceso Unificado de Desarrollo.

[2] UML (Language Unified Model) Lenguaje Unificado de Modelado.

3 CASE (Computer Aided Software Engineering) Ingeniería de Software Asistida por Computadora.

Índice de Contenido

Índice de Figuras.

Índice de Tablas.

INTRODUCCIÓN

La introducción masiva de las nuevas Tecnologías de la Información y las Comunicaciones (TIC's) está influenciando fuertemente la estructura y dinámica de los procesos económicos y sociales, redefiniendo aceleradamente las formas de producir, vender y competir, en prácticamente todos los sectores de bienes y servicios; así como en las nuevas formas de interacción y comunicación entre las personas y organismos de la sociedad.

Desde hace algunos años Cuba ha emprendido el reto de la informatización de la sociedad, con el objetivo de elevar la calidad de vida del pueblo y lograr mejoras en la economía. La Universidad de Ciencias Informáticas (UCI) es un ejemplo de todo este proceso de informatización que se está desarrollando, fue creada como un proyecto de la Revolución con el fin de formar jóvenes con el mayor conocimiento en la rama de la informática y crear grupos de proyectos para la creación de servicios informáticos tanto nacionales como para exportación y darle solución de esta manera a diferentes problemas existentes en el país.

Uno de los proyectos que está llevando a cabo la UCI es el encargado de informatizar todos los procedimientos que se realizan en los Tribunales Populares Cubanos (TPC). Este proyecto desarrolla la solución informática: Sistema de Informatización de los Tribunales (SIT), conformada por siete subsistemas, siendo uno de ellos el subsistema Administración y Gobierno, encargado de gestionar la estructura y composición de los Tribunales Populares Cubanos así como definir todo lo relacionado a los usuarios, roles y permisos que serán asignados para interactuar con el sistema.

En los Tribunales Populares Cubanos actualmente se realiza un gran número de trámites y se atiende un volumen excesivo de expedientes, realizándose todo este trabajo de forma manual. La situación anterior se ve agudizada al guardar los expedientes en estantes, donde la búsqueda de los mismos resulta agotadora, además del deterioro o pérdida al que se encuentran expuestos. Trayendo esto consigo la no existencia de un control efectivo de la información y por ende un atraso en el trabajo. Para dar solución a estos problemas surge el proyecto Tribunales Populares Cubanos, encargado de desarrollar la solución informática SIT, dentro de la cual es de vital importancia la creación del subsistema Administración y Gobierno que actúa como rector informático, garantizando que todos los subsistemas de SIT se

desarrollen de manera correcta, permitiendo la centralización de algunos procedimientos y configuraciones globales que garantizarán el buen funcionamiento del mismo.

Para lograr un correcto desarrollo del subsistema Administración y Gobierno es necesario hacer un análisis previo donde se identifiquen las necesidades del cliente y se traduzcan al lenguaje de los desarrolladores, pues estos últimos no tienen claridad acerca de lo que el cliente quiere y no se delimita qué debe y qué no debe hacer el sistema. Es decir, es necesario llegar a un acuerdo entre los involucrados sobre lo que el software debe hacer, para así proporcionar a los desarrolladores un mejor entendimiento de los requisitos del software.

Las razones expuestas anteriormente dan lugar a que el ***problema de la investigación*** a solucionar sea: ¿Cómo lograr un entendimiento entre clientes y desarrolladores que contribuya al diseño del subsistema Administración y Gobierno?

Donde se tiene como ***objeto de estudio***: La Ingeniería de Software.

Para dar respuesta al problema planteado se define como ***objetivo general*****:** Realizar la Ingeniería de Requisitos, para lograr un entendimiento entre clientes y desarrolladores que contribuya al diseño del subsistema Administración y Gobierno.

Enmarcándose el ***campo de acción*** en la Ingeniería de Requisitos del subsistema Administración y Gobierno.

Inicialmente se parte de la siguiente ***idea a defender***: Con la realización de la Ingeniería de Requisitos del subsistema Administración y Gobierno, se logrará un entendimiento entre clientes y desarrolladores para así contribuir a su posterior diseño.

Los ***objetivos específicos*** trazados para dar cumplimiento al objetivo general son:

- ✓ Elaborar el marco teórico de la investigación.
- ✓ Desarrollar los artefactos de software correspondientes a las etapas de la IR.
- ✓ Validar los resultados obtenidos.

Las ***tareas*** de la investigación trazadas para dar cumplimiento a estos objetivos son:

- ✓ Realización de búsquedas bibliográficas encaminadas a investigar las metodologías de desarrollo de software, herramientas y lenguajes de modelado más utilizados actualmente.
- ✓ Estudio e identificación de los aspectos fundamentales de los proceso que se llevan a cabo en los Tribunales Populares Cubanos y que se relacionen con el subsistema Administración y Gobierno.
- ✓ Confección del Modelo de Dominio, Especificación de Requisitos de Software, Modelo del Sistema, Prototipo no Funcional del Sistema y Matriz de Trazabilidad de los Requisitos.
- ✓ Validación de los principales resultados obtenidos a través de métricas para el desarrollo de un software.
- ✓ Aplicación del método de Kano para medir la satisfacción del cliente.

Para dar cumplimiento a las tareas propuestas anteriormente se emplean métodos científicos de la investigación Teóricos y Empíricos.

De los métodos Teóricos se emplearon:

- ✓ EL **Analítico-Sintético**: Posibilitó la realización del estudio teórico de la investigación y en el análisis previo sobre el funcionamiento del proceso Administración y Gobierno que se lleva a cabo en los Tribunales Populares Cubanos, permitiendo extraer sus elementos más importantes y establecer relaciones entre ellos.
- ✓ EL **Histórico-Lógico**: Permitió analizar la trayectoria completa del proceso Administración y Gobierno desde su origen hasta la actualidad y revelar sus etapas principales.
- ✓ La **Modelación**: Para la creación de modelos que representan abstracciones, con el objetivo de explicar cómo funciona el proceso Administración y Gobierno que se lleva a cabo en los Tribunales Populares Cubanos, en función de ayudar a comprender sus leyes y teorías.

De los métodos Empíricos se utilizó:

- ✓ La **Entrevista**: Con el fin de obtener información valiosa del proceso Administración y Gobierno que se lleva a cabo en los Tribunales Populares Cubanos.
- ✓ La **Observación**: Para la mejor comprensión del proceso Administración y Gobierno mediante la percepción planificada y prolongada de sus aspectos.

Con este Trabajo de Diploma se pretende obtener como **resultados**:

- ✓ Modelo del Dominio.
- ✓ Especificación de Requisitos de Software.
- ✓ Modelo del Sistema.
- ✓ Prototipo no Funcional del Sistema.

El presente trabajo ha sido organizado de la siguiente manera:

Capítulo 1 Fundamentación Teórica: En este capítulo se realiza un estudio de las distintas etapas de la Ingeniería de Requisitos desde el punto de vista de varios autores, seleccionándose las que serán utilizadas en el desarrollo del subsistema. Se analizan las principales técnicas usadas durante estas etapas, además de investigar sobre las metodologías de desarrollo, lenguajes de modelado, herramientas para el desarrollo del software, herramientas de prototipado y métodos para medir la satisfacción del cliente, justificándose en cada caso la variante seleccionada.

Capítulo 2 Propuesta de Solución: En este capítulo se realiza el Modelo de Dominio para el subsistema Administración y Gobierno. Además se especifican los Requisitos Funcionales y los Requisitos no Funcionales y se exponen las técnicas de obtención de requisitos empleadas para la captura de los mismos, así como los patrones de Casos de Uso que se utilizan. También se presentan los artefactos obtenidos: Actores del Sistema, Diagrama de Casos de Uso del Sistema (DCUS) y la descripción textual de cada Caso de Uso con su prototipo no funcional de interfaz.

Capítulo 3 Validación de los Resultados: En este capítulo se aplican métricas para garantizar la calidad de la especificación de los requisitos y del Diagrama de Caso de Uso del Sistema. Se aplica además el método de Kano para medir la satisfacción del cliente.

CAPÍTULO 1. FUNDAMENTACIÓN TEÓRICA.

1.1 Introducción.

En este capítulo se hace un estudio de las etapas de la Ingeniería de Requisitos desde el punto de vista de varios autores, así como las técnicas más usadas para la realización de las mismas. Se realiza además un análisis de las herramientas, lenguajes de modelado y metodologías de desarrollo de software actuales y métodos para medir la satisfacción del cliente, justificando las empleadas en el subsistema Administración y Gobierno del proyecto Tribunales Populares Cubanos.

1.2 Subsistema Administración y Gobierno.

Los sistemas informáticos actuales son cada vez más flexibles y adaptables en dependencia de las necesidades de los clientes, por lo general están compuestos por varios módulos independientes que realizan determinadas funciones y todos ellos unidos por una estructura organizativa que permitirá que el sistema funcione de forma ordenada, segura y consistente.

Por esta razón generalmente la mayoría de los sistemas cuentan con un rector informático, el cual se encargaría de permitir el control de la estructura organizativa del sistema, garantizando eficiencia en todos sus módulos y procesos, permitiendo la centralización de algunos procedimientos y configuraciones globales que garantizarían el buen funcionamiento del mismo. Manejando diferentes conceptos en dependencia de las necesidades finales del cliente, por ejemplo: Usuarios, roles, permisos, seguridad, entre otros.

En el caso específico el subsistema de Administración y Gobierno soportará la gestión de tribunales en las diferentes instancias. Los tribunales a su vez están formados por salas o por secciones, que contendrán usuarios, cada uno con roles predefinidos que podrán ser modificados en los respectivos tribunales para los que fueron creados. Debe permitir la gestión de las plantillas de calendario para el establecimiento de los días hábiles por la que se regirán todos los tribunales del país, así como el calendario de señalamientos donde se va a permitir el señalamiento de los actos judiciales públicos. Debe dar la posibilidad de configurar el período de tiempo que un juez lego trabajará en un tribunal, así como de visualizar las trazas del sistema donde se mostrarán las acciones de los usuarios.

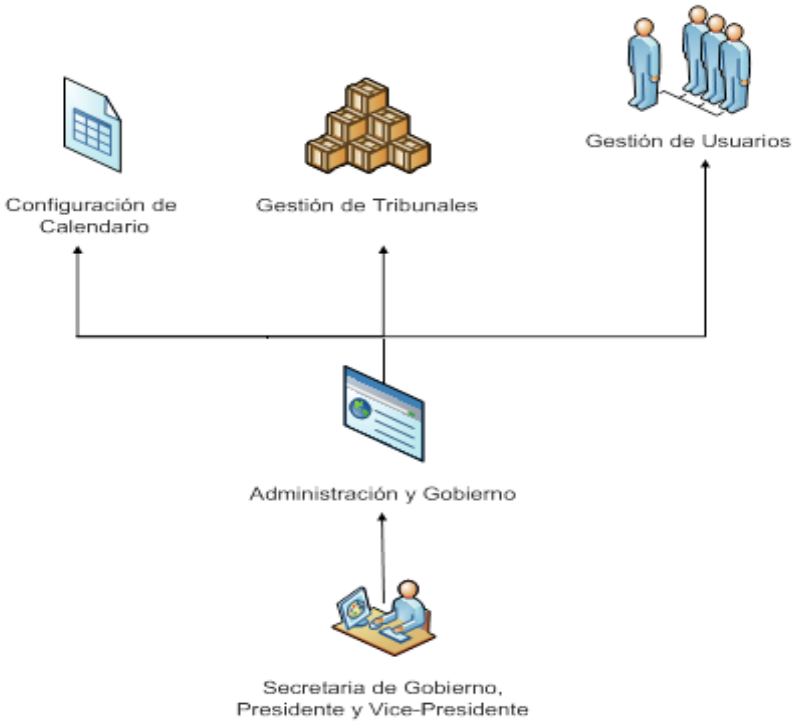


Fig.1 Subsistema Administración y Gobierno.

Para el subsistema Administración y Gobierno, la seguridad de la solución informática SIT constituye uno de los requisitos más importantes. Donde el uso de los conceptos de autenticación y autorización, basados en requerir una cuenta de usuario válida y activa, además de un conjunto de roles les permitirán al usuario realizar solo aquellas operaciones que se le han asignado, brindado así un alto nivel de seguridad, fiabilidad y el control de las operaciones a realizar.

1.3 Ingeniería de Requisitos.

Cuando se va a crear una solución informática, es primordial reconocer y establecer las funcionalidades que debe brindar, así como las restricciones sobre las que debe operar, tarea que resulta difícil para los desarrolladores, a pesar de contar con avanzadas herramientas y tecnologías para el desarrollo del software. Es por esto que surge la Ingeniería de Requisitos, que tiene como meta comprender las necesidades exactas de los usuarios y transformarlas al

lenguaje de los desarrolladores, cubriendo las actividades relacionadas con descubrir, documentar, validar y mantener un conjunto de requisitos para un sistema informático.

La misma tiene lugar durante todo el ciclo de vida del software, principalmente en las primeras etapas cuando es necesario descubrir y comunicar las necesidades de clientes y usuarios, y más adelante para la gestión de los cambios en dichas necesidades. Dentro de los principales beneficios que se obtienen con un buen desarrollo de la Ingeniería de Requisitos se encuentran los siguientes:

- ✓ Proporciona un punto de partida para la estimación de costos, tiempo y recursos necesarios.
- ✓ Disminuye los costos y retrasos del proyecto.
- ✓ Mejora la calidad del software en cuanto a funcionalidad, facilidad de uso, confiabilidad, desempeño, etc.
- ✓ Mejora la comunicación entre equipos: la Especificación de Requisitos representa una forma de consenso entre clientes y desarrolladores.

1.3.1 Etapas de la Ingeniería de Requisitos.

El proceso de la Ingeniería de Requisitos comprende un conjunto de etapas encaminadas a la obtención, refinamiento y validación de las funcionalidades que debe cumplir el software a desarrollar, a continuación se mostrará la relación de estas etapas definidas por varios autores:

Ian Sommerville[4]

Define tres procesos fundamentales, donde cada una de ellos engloba una serie de actividades que dan cumplimiento al objetivo fundamental de cada uno de estos: (Sommerville, 2005)

Estudio de viabilidad: La entrada de este proceso es un conjunto de requisitos de negocio preliminares, una descripción resumida del sistema y de cómo este pretende contribuir a los procesos del negocio. Comprende la evaluación y recopilación de la información.

Obtención y análisis de requisitos: En esta actividad, los ingenieros de software trabajan con los clientes para determinar el dominio de la aplicación, qué servicios debe proporcionar el

[4] Profesor titular, autor de uno de los principales libros de referencia de la Ingeniería de Requisitos.

sistema, las restricciones de hardware, etc. Es un proceso iterativo con retroalimentación continua. Para el cumplimiento de este proceso Sommerville define las siguientes actividades:

- ✓ Descubrimiento de los requisitos: Es el proceso de interactuar con los usuarios del sistema para recopilar sus funcionalidades. Las funcionalidades del dominio de los clientes y la documentación también se va creando durante esta actividad. Define diferentes técnicas para la obtención de los requisitos como: Entrevistas, escenarios y la etnografía.
- ✓ Clasificación y organización de los requisitos: Partiendo de un grupo de requisitos, se da a la tarea de agruparlos siguiendo un orden coherente.
- ✓ Ordenación por prioridades y negociación de los requisitos: Se refiere a ordenar según las prioridades los requisitos, y a encontrar y resolver los requisitos en conflictos a través de la negociación.
- ✓ Documentación de los requisitos: Se documentan los requisitos y se entra en la siguiente vuelta de la espiral.

Validación de requisitos: Trata de mostrar que estos realmente definen el sistema que el cliente desea. Se deben llevar a cabo verificaciones sobre los requisitos, estas pueden ser: verificaciones de validez, de consistencia, de completitud y de realismo. En esta etapa se plantean algunas técnicas de validación de requisitos entre las que se encuentran: las revisiones de los requisitos, la construcción de prototipos y la generación de casos de prueba.

Sommerville es abarcador en las etapas que define, sin embargo no tiene en cuenta la gestión de los requisitos, aspecto fundamental para el control de cambios y trazabilidad en los mismos.

Griselda Báez[5]

Define tres etapas para el desarrollo de la Ingeniería de Requisitos: (Báez, 2003)

Elicitación: Es la etapa de mayor interacción con el usuario. Es el momento en el que se recurre a las entrevistas y en el que equipos multidisciplinarios trabajan conjuntamente con el cliente/usuario, para obtener de la mejor manera, los requisitos reales.

[5] Profesora de la Universidad Nacional del Litoral.

Análisis: La etapa de análisis de requisitos permite al analista representar el dominio de la información de la aplicación a desarrollar, a través del uso de un lenguaje más técnico, procurando reducir ambigüedades. Brinda al analista, la representación de la información y las funciones que facilitarán la definición del futuro diseño.

Especificación: El objetivo de esta etapa es obtener una especificación de los requisitos que ya han sido analizados y negociados con los clientes, la cual puede desarrollarse en un documento escrito, de manera que sean presentados de una forma más consistente y comprensible.

Al igual que Sommerville no define la gestión de los requisitos ni siquiera la validación, tan importante para evaluar la calidad en la especificación de estos.

Roger S. Pressman[6]

Define siete etapas para el desarrollo de la Ingeniería de Requisitos, destacando que algunas de estas ocurren en paralelo y que todas deben adaptarse a las necesidades del proyecto: (Pressman, 2005)

Inicio: El objetivo de esta etapa es establecer una comprensión del problema, las personas que quieren una solución, la naturaleza de la solución que se desea, y la efectividad de la comunicación preliminar entre el cliente y el desarrollador. Esto se logra al inicio del proyecto, donde los ingenieros de software hacen una serie de preguntas libres de texto a los clientes. Pressman aconseja que las preguntas libres de contexto se usen solo para el primer encuentro, y después reemplazar por un formato de obtención de requisitos que combine elementos de resolución de problemas, negociación y especificación.

Obtención: Parece muy simple preguntarle al cliente, a los usuarios finales y a otros interesados cuáles son los objetivos del sistema, qué es lo que se debe lograr, de qué forma el producto satisface las necesidades del negocio y por último como se utilizará el sistema. Todo ello se logra en esta fase de obtención de los requisitos, de ahí que esta etapa sea la fundamental para la recopilación de información, y a las vez una de las más complejas.

[6] Presidente de la firma Consultora Especialista en Métodos y Entrenamientos de la Ingeniería de Software. Autor de uno de los principales libros de la Ingeniería de Software.

Elaboración: La información conseguida con el cliente durante el inicio y la obtención se expande y se refina durante la elaboración. Esta se conduce mediante la creación y refinamiento de escenarios del usuario que describen la forma en que interactuará con el sistema. El resultado final de la elaboración es un modelo de análisis que define el dominio de la información, las funciones y el comportamiento del problema.

Negociación: Durante el desarrollo del proyecto es común que diferentes clientes o usuarios propongan requisitos, entrando en conflicto al argumentar que su versión es esencial, en estos casos el ingeniero de requisitos debe conciliar estos conflictos por medio de un proceso de negociación. Se pide a los clientes, usuarios y otros interesados que ordenen sus requisitos y después discutan los conflictos relacionados con la prioridad. Se identifican y analizan los riesgos asociados con cada requisito. Mediante un enfoque iterativo, los requisitos se eliminan, combinan o modifican de forma que cada parte alcance cierto grado de satisfacción.

Especificación: La especificación es el producto del trabajo final que genera la ingeniería de requisitos, donde se describe la función y el desempeño del sistema y las restricciones que regirán su desarrollo.

Validación: La validación examina la especificación para asegurar que todos los requisitos se han establecido de manera precisa; que se han detectado las inconsistencias, omisiones y errores y qué estos han sido corregidos. Define como principal mecanismo de validación la revisión técnica.

Gestión: La gestión de requisitos es un conjunto de actividades que ayudan al equipo de proyecto a identificar, controlar y rastrear los requisitos y los cambios de estos en cualquier momento mientras se desarrolla el proyecto. La gestión comienza desde la identificación. Para el cumplimiento de esta actividad, se define que se lleven a cabo las siguientes Tablas de rastreabilidad: de las características, de la fuente, de dependencia, del subsistema y de la interfaz.

Justificación de las etapas seleccionadas.

Luego de un estudio de varios autores se decide utilizar las etapas definidas por Roger S. Pressman, teniendo en cuenta las características y amplitud del proyecto, estas se adaptarían de la siguiente manera: las etapas de Inicio y Obtención de los requisitos se desarrollarán

paralelamente, logrando la recopilación de la información y una determinación previa de las características y restricciones que deberá satisfacer el sistema a desarrollar. Así mismo las fases de Elaboración y Negociación se trabajarán simultáneamente, de manera que se logre una definición del dominio de la información, las funciones, comportamientos del problema, la conciliación de los conflictos que surjan entre los clientes, identificando y analizando los riesgos asociados a los requisitos; logrando de manera general que cada parte alcance cierto grado de satisfacción. Posteriormente se realizará la etapa de Especificación de Requisitos, luego la de Validación y durante todo el proceso de desarrollo de la Ingeniería de Requisitos, la etapa de Gestión.

1.3.2 Técnicas utilizadas en la Ingeniería de Requisitos.

Con el objetivo de minimizar las dificultades que se presentan en la IR, se aplican un conjunto de técnicas en cada una de sus etapas, a continuación se detallarán algunas de estas:

Entrevista: Las entrevistas resultan una técnica muy aceptada dentro de la IR para la obtención de los requisitos y su uso está ampliamente extendido. A través de esta técnica el equipo de trabajo se acerca al problema de una forma natural. Básicamente, la estructura de la entrevista abarca cuatro pasos: identificación de los entrevistados, preparación de la entrevista, realización de la entrevista y documentación de los resultados. (Escalona & Koch, 2002).

Tormenta de Ideas: Es una técnica de desarrollo en grupo muy utilizada en la captura de requisitos. Su propósito es que los participantes muestren sus ideas referentes a un problema, a través de una intervención participativa y en un ambiente libre de críticas y formalidades. La participación en las sesiones es más importante que la creatividad individual (Bartle, 2003). Ayudan a generar diferentes vistas del problema. Son muy fáciles de aprender y requieren poca organización.

Introspección: Método para la captura de requisitos que se utiliza para entender las necesidades del cliente. Recomienda que el entrevistador se ponga en el lugar del cliente y trate de imaginar cómo desearía el sistema, si fuera para él y en base a estas suposiciones comenzar a recomendarle sobre la funcionalidad que debería presentar el sistema. (Goguen, 1993).

Observación y Análisis de Tareas: En esta técnica un observador estudia a los futuros usuarios en su entorno de trabajo. Anota todo aquello que es susceptible de mejora, para posteriormente obtener una serie de requisitos tentativos. En ocasiones se utiliza el video.

Casos de Uso: Los casos de uso son una técnica para especificar el comportamiento de un sistema. Un caso de uso es la descripción de una secuencia de interacciones entre el sistema y uno o más actores en la que se considera al sistema como una caja negra y en la que los actores obtienen resultados observables. (Durán, 2000).

Prototipos: Útiles cuando la incertidumbre es total acerca del futuro sistema. Los prototipos se utilizan para validar los requisitos hallados. Son simulaciones del posible producto, que luego son utilizados por el usuario final. Permiten verificar si el sistema está diseñado en base a los requisitos recolectados. (Torres, 2008).

Revisiones Técnicas Formales: constituyen el mecanismo primario para la validación de los requisitos, desarrollada a partir de las revisiones que realice un equipo destinado a ello. Los diferentes encuentros y entregas de los artefactos para su revisión estarán encaminados a la examinación de la especificación y a la búsqueda de errores en el contenido, a la interpretación, a la información faltante, inconsistencias, conflictos entre los requisitos y la existencia de requisitos irreales (inalcanzables). (Freeman, et al., 1990).

Matriz de Trazabilidad: una de las técnicas utilizadas para la gestión de requisitos, indica cómo se relacionan los requisitos entre sí, permitiendo determinar cómo afecta el impacto de un cambio en los mismos. Determina si todos los casos de uso del sistema satisfacen cada uno de los requisitos identificados. (Pressman, 2005).

1.4 Metodologías de desarrollo de Software.

Las metodologías de desarrollo de software surgen para guiar a las personas implicadas en el desarrollo de software, brindando un conjunto de procedimientos, técnicas y herramientas, de forma que sepan qué hacer en cada momento, y cómo alcanzar un producto de alta calidad. Actualmente han proliferado dos corrientes opuestas para el desarrollo de software, las metodologías tradicionales y las metodologías ágiles.

1.4.1 Metodologías Pesadas.

Las metodologías tradicionales o pesadas son aquellas que están guiadas por una fuerte planificación durante todo el proceso de desarrollo, se centran en la definición detallada de los procesos y tareas a realizar así como de las herramientas a utilizar, y requieren una extensa documentación, ya que pretenden prever todo de antemano. Este tipo de metodologías son más eficaces y necesarias cuanto mayor es el proyecto que se pretende realizar respecto a tiempo y recursos que son necesarios emplear, donde una gran organización es requerida.

Proceso Unificado de Desarrollo (RUP).

RUP es un proceso de desarrollo de software, que define “quién” está haciendo “qué”, “cuándo” y “cómo” para alcanzar un determinado objetivo. Es una metodología robusta que representa uno de los procesos más generales, pues está pensado para adaptarse a una gran variedad de sistemas de software, de diferentes tamaños de proyectos y diferentes áreas de aplicación. (Jacobson, Booch, & Rumbaugh, 2000).

Su propósito es asegurar la producción de software de alta calidad que se ajuste a las necesidades de sus usuarios finales con unos costos y calendario predecibles.

Tiene la particularidad de estar dirigido por Casos de Uso: pues estos no son sólo una herramienta para especificar los requisitos del sistema, también guían su diseño, implementación y prueba; está centrado en la arquitectura: muestra la visión común del sistema completo por lo que describe los elementos del modelo que son más importantes para su construcción, es el cimiento del sistema necesario para comprenderlo, desarrollarlo y producirlo económicamente y es iterativo e incremental: divide al proyecto para que los casos de uso y la arquitectura cumplan sus objetivos de manera más depurada, además de utilizar el UML como lenguaje de representación visual. (Jacobson, Booch, & Rumbaugh, 2000).

RUP divide el proceso en cuatro fases, dentro de las cuales se realizan varias iteraciones en número variable según el proyecto y en las que se hace un mayor o menor hincapié en las distintas actividades. Las fases son: Inicio, Elaboración, Construcción, Transición.

Inicio: tiene como objetivo definir los límites y la visión del proyecto. En esta fase se planifica el proyecto. Poniendo mayor énfasis en las actividades de modelado del negocio y de requisitos.

Elaboración: se define la arquitectura del sistema, es donde se realiza la especificación detallada de cada caso de uso. Además se hace un plan de proyecto y se eliminan los riesgos.

Construcción: fundamentándose en la línea base de la arquitectura establecida, se fabrica el producto y se define si está preparado para el despliegue.

Transición: se garantiza la disponibilidad del software para los usuarios, además se prueba el producto para detectar y corregir errores. Como consecuencia de esto suelen surgir nuevos requisitos a ser analizados. En la misma también se le brinda asistencia y ayuda al cliente para su formación.

En RUP se han agrupado las actividades en grupos lógicos definiéndose nueve flujos de trabajo principales a realizar en las fases del proyecto: Modelado del Negocio, Requisitos, Análisis y Diseño, Implementación, Prueba, Despliegue, Gestión de Configuración de Cambios, Gestión del Proyecto, Gestión del Entorno.

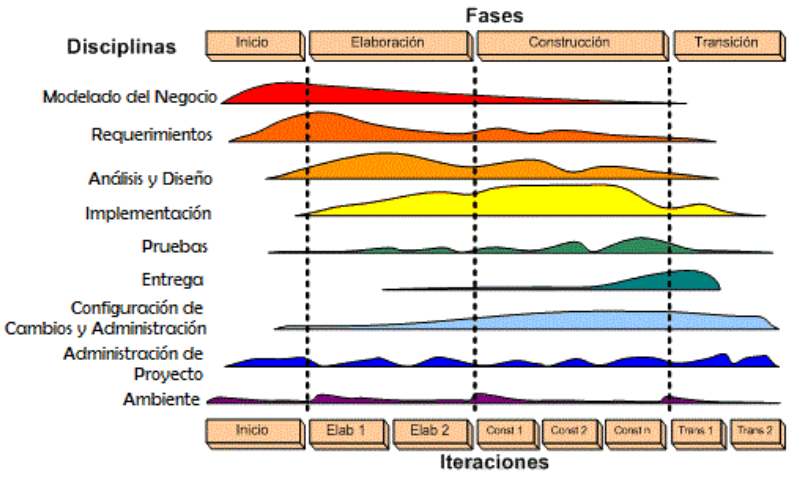


Fig.2 Fases e iteraciones de RUP.

Marco de solución de Microsoft (MSF).

MSF es una metodología flexible e interrelacionada con una serie de conceptos, modelos y prácticas de uso, que controlan la planificación, el desarrollo y la gestión de proyectos tecnológicos. Se centra en los modelos de procesos y de equipo dejando en un segundo plano las elecciones tecnológicas. (Mendoza, 2004). Las principales características que presenta son:

- ✓ Adaptable: usada en cualquier parte como un mapa, del cual su uso es limitado a un lugar específico.
- ✓ Escalable: puede organizar equipos tan pequeños entre tres o cuatro personas, así como también, proyectos que requieren cincuenta personas o más.
- ✓ Flexible: es utilizada en el ambiente de desarrollo de cualquier cliente.
- ✓ Tecnología Agnóstica: porque puede ser usada para desarrollar soluciones basadas sobre cualquier tecnología.

Además se compone de varios modelos encargados de planificar las diferentes partes implicadas en el desarrollo de un proyecto: Modelo de Arquitectura del Proyecto, Modelo de Equipo, Modelo de Proceso, Modelo de Gestión del Riesgo, Modelo de Diseño de Proceso y finalmente el Modelo de Aplicación.

El proceso de desarrollo en MSF consta de cinco fases como muestra la figura 3 (Pyme Actual, 2006), las cuales junto a los modelos mencionados anteriormente, complementan el ciclo de desarrollo de software en esta metodología.

Fig.3 Ciclo de vida de la metodología MSF.

MSF provee un marco de trabajo adaptable para liberar exitosamente soluciones rápidas, con pocas personas y producir resultado de alta calidad, sin embrago su principal desventaja es que su desarrollo se basa en tecnología Microsoft la cual es cara y restringe mucho las herramientas de desarrollo.

1.4.2 Metodologías Ágiles.

Un proceso es ágil cuando el desarrollo de software es incremental, cooperativo, sencillo y adaptable**.** (Pressman, 2005)**.** Siendo más efectivo su uso cuando se tiene un pequeño equipo de desarrollo enfrentándose a requisitos inestables y se exige minimizar el tiempo de desarrollo.

Programación Extrema (XP).

Esta metodología consiste en una programación rápida (extrema), cuya particularidad es tener como parte del equipo al usuario final, pues es uno de los requisitos para llegar al éxito del proyecto (Sánchez, y otros, 2004). El cliente recibe después de cada iteración una parte funcional del programa, por lo que estará informado continuamente sobre el proyecto, teniendo la posibilidad de intervenir si el desarrollo se desvía de sus necesidades (Molpeceres, 2002). Para proyectos de corto plazo y reducido equipo, XP es una de las metodologías de desarrollo de software más utilizada y exitosa en la actualidad. Entre sus características principales se tienen las siguientes:

- ✓ Desarrollo iterativo e incremental: pequeñas mejoras, unas tras otras.
- ✓ Pruebas unitarias continuas, frecuentemente repetidas y automatizadas, incluyendo pruebas de regresión.
- ✓ Frecuente integración del equipo de programación con el cliente o usuario.
- ✓ Refactorización del código: reescribir ciertas partes del código para aumentar su legibilidad y mantenibilidad pero sin modificar su comportamiento.

XP es una metodología ligera, eficiente, con bajo riesgo, flexible, predecible y divertida para desarrollar software. (Beck, 2000).

Justificación de la metodología seleccionada.

Después de realizar el estudio de las principales metodologías de desarrollo de software, se decide utilizar RUP por definir en cada momento del ciclo de vida del software, cuáles artefactos, con qué nivel de detalle y qué roles deben ser creados. Permite identificar problemas y fallos de modo que puedan ser prevenidos o corregidos a tiempo. Además es una metodología robusta que se adapta muy bien a proyectos de larga duración, complejos y con

un gran equipo de desarrollo como lo es el proyecto TPC. Ha sido probada a nivel mundial y es utilizada frecuentemente en la UCI por lo que se cuenta con el conocimiento apropiado. Permite un mantenimiento adecuado en una segunda etapa del proyecto debido a la suficiente documentación que genera, además seleccionada por contar con clientes que no tendrán una relación directa con el equipo del proyecto.

1.5 Lenguajes de modelado.

El constante avance en el nivel de complejidad de las soluciones informáticas ha hecho de la utilización de los modelos un mecanismo para facilitar la comprensión de estos. “Los modelos proporcionan un mayor nivel de abstracción, permitiendo trabajar con sistemas mayores y más complejos, y facilitando el proceso de codificación e implementación del sistema de forma distribuida y en distintas plataformas.” (Fuentes & Vallecillo, 2003).

“Un modelo es una descripción de (parte de) un sistema, descrito en un lenguaje bien definido.” (Fuentes & Vallecillo, 2003). Los modelos de procesos de negocio se usan para mejorar la comunicación tanto entre el analista y el desarrollador como entre el analista y el cliente. En cuanto a los lenguajes y estándares que van a permitir realizar modelos de procesos de negocio se destacan:

Lenguaje Unificado de Modelado (UML).

UML “es un lenguaje de modelado visual que se usa para especificar, visualizar, construir y documentar artefactos de un sistema de software “(Jacobson, Booch, & Rumbaught, 2000). No define un proceso de desarrollo específico, tan solo se trata de una notación. “Permite modelar sistemas de información, y su objetivo es lograr modelos que, además de describir con cierto grado de formalismo tales sistemas, puedan ser entendidos por los clientes o usuarios de aquello que se modela” (Jacobson, Booch, & Rumbaught, 2000).

Posibilita el intercambio de modelos entre las distintas herramientas de modelado orientadas a objetos del mercado, para ello era necesario definir una notación y semántica común, es decir, un estándar. UML es además un método formal de modelado, lo que aporta las siguientes ventajas:

- ✓ Mayor rigor en la especificación.

- ✓ Permite realizar una verificación y validación del modelo realizado.
- ✓ Concurrencia, es un lenguaje distribuido y adecuado a las necesidades de conectividad actuales y futuras.
- ✓ Reemplaza a decenas de notaciones empleadas con otros lenguajes.
- ✓ Modela estructuras complejas.
- ✓ Las estructuras más importantes que soportan tienen su fundamento en las tecnologías orientadas a objetos, tales como objetos, clase, componentes y nodos.
- ✓ Emplea operaciones abstractas como guía para variaciones futuras, añadiendo variables si es necesario.
- ✓ Comportamiento del sistema: casos de uso, diagramas de secuencia y de colaboraciones, que sirven para evaluar el estado de las máquinas.

UML ofrece una amplia variedad de diagramas para visualizar el sistema desde varias perspectivas. Incluye los siguientes diagramas: Diagrama de casos de uso, Diagrama de clases, Diagrama de objetos, Diagrama de secuencia, Diagrama de colaboración, Diagrama de estados, Diagrama de actividades, Diagrama de componentes, Diagrama de despliegue. (Jacobson, Booch, & Rumbaught, 2000).

En relación con los Diagramas de Actividad de UML, a pesar de que su uso para la modelación de los procesos del negocio ha sido criticado debido a la limitada expresividad de sus versiones anteriores, razón por la cual “en el paso de la versión 1.5 a la versión 2.0 la parte más modificada es precisamente la referente a los Diagramas de Actividad” (Pérez J. D., 2007), lográndose que los mismos fueran una notación con la expresividad adecuada para modelar todo tipo de procesos de negocio, se debe tener presente que:

- ✓ Existen muchas herramientas para trabajar con ellos como por ejemplo: IBM Rational Software Architect, Enterprise Architect, Visual Paradigm.
- ✓ Existe gran cantidad de procesos de negocio que están modelados usando esta notación.
- ✓ Los desarrolladores tienen gran experiencia utilizando tanto UML como sus Diagramas de Actividad. (Pérez J. D., 2007)

Definición de la Integración para el Modelado de Funciones (IDEF).

IDEF0[7] es una técnica de modelación capaz de representar de manera estructurada y jerárquica las actividades que conforman un sistema o empresa a cualquier nivel de detalle, y los objetos o datos que soportan la interacción de esas actividades. En las vistas superiores del modelo la interacción entre las actividades representadas permite visualizar los procesos fundamentales que sustentan la organización.

La representación de un proceso consta de la interconexión de cinco elementos básicos que trabajan juntos para la realización de una función útil: Entrada, Salida, Control, Sujeto y la actividad (Loyola, 2006). Algunas de sus características generales son:

- ✓ Es un medio comprensivo que posibilita comunicar reglas y procesos de negocios.
- ✓ Se puede utilizar para modelar una variedad amplia de sistemas automatizados y no automatizados. Para los nuevos sistemas, se puede utilizar primero para definir los requisitos y para especificar las funciones, y después para diseñar una puesta en práctica que resuelva los requisitos y realice las funciones.
- ✓ Puede ser generado por una gran variedad de herramientas gráficas en computadores. Muchos productos comerciales apoyan específicamente el desarrollo y el análisis de los diagramas IDEF0 y de sus modelos. (Knowledge Based System, 2006).

Notación para el Modelado de Procesos del Negocio (BPMN).

BPMN[8] es un nuevo estándar de modelado de procesos de negocio, en donde se presentan gráficamente las diferentes etapas del proceso del mismo. La notación ha sido diseñada específicamente para coordinar la secuencia de procesos y los mensajes que fluyen entre los diferentes procesos participantes, incluyendo la unión con el diseño y la implementación.

Con BPMN, un diagrama puede ser transformado en un código ejecutable automáticamente, sin la necesidad de programación. De esta forma, el analista de negocios puede definir, diseñar y generar una solución a sus procesos.

[7] IDEF0: Definición de la integración para la modelación de las Funciones o Integration Definition for Function Modeling.

[8] BPMN: Notación de Modelado de Procesos de Negocio o Business Process Modeling Notation.

Ofrece a los analistas de negocios una forma consistente con su manera de trabajar. Igualmente, sus componentes mapean las dimensiones Qué, Cómo, Cuándo, Dónde y Por Qué. Algunas de sus características principales son (Durocher, 2007):

- ✓ Visibilidad de los procesos de las empresas.
- ✓ Mayor flexibilidad y agilidad para adaptación al cambio.
- ✓ Brinda la posibilidad de integrar la información del negocio dispersa en diferentes sistemas y permite adquirir una ruta de mejoramiento y eficiencia continua al convertir actividades ineficientes en menores costos a través de uso de tecnología enfocada en procesos.
- ✓ Brinda la posibilidad de adquirir la habilidad para diseñar, simular y monitorear procesos de manera automática y sin la participación de usuarios técnicos.

Presenta una gran expresividad a la hora de especificar procesos, mucho más expresivo que los diagramas de actividad de UML, es gráficamente más rico, con menos símbolos, pero con más variaciones de estos, lo que facilita su comprensión por parte de personas que no tienen dominio del lenguaje.

Justificación del lenguaje seleccionado.

Después de realizado un estudio de los lenguajes de modelado se decide utilizar UML, debido a que es uno de los más usados a nivel mundial para modelar los artefactos creados durante todo el proceso de desarrollo de software. Es un lenguaje estándar, fácil de aprender y permite una comunicación fluida entre los desarrolladores de software. Por otra parte ofrece una amplia variedad de diagramas para visualizar el sistema desde varias perspectivas. Se tiene en cuenta además que fue desarrollado junto con la metodología RUP, por lo que responde a todas sus necesidades.

1.6 Herramientas de modelado.

Ingeniería de Software Asistida por Computadora (Computer Aided Software Engineering, CASE) es un tipo de ingeniería de software en la que se intenta aumentar la eficacia de sus procesos, al soportar la realización de las tareas con el uso de tecnologías (Pérez, 1999).

Las herramientas CASE son diversas aplicaciones informáticas destinadas a aumentar la productividad en el desarrollo de software reduciendo el coste de las mismas en términos de tiempo y de dinero. Estas herramientas permiten modelar y documentar los artefactos, cubriendo el ciclo de vida del proceso de desarrollo de software.

Rational Rose.

Es una herramienta de diseño de software destinado a modelado visual y construcción de componentes de aplicaciones empresariales de software a nivel. Dos características populares que presenta es su capacidad para proporcionar el desarrollo iterativo y la ingeniería de viaje redondo. Permite a los diseñadores tomar ventaja del desarrollo iterativo porque con la aplicación se pueden crear nuevas etapas y la salida de una iteración puede convertirse en la entrada de la siguiente. Algunas características que ofrece son (Rational, 2009):

- ✓ Soporte para análisis de patrones ANSI C++, Rose J y Visual C++.
- ✓ Característica de control por separado de componentes de modelo que permite una administración más granular y el uso de modelos.
- ✓ Soporte de ingeniería directa y/o inversa para algunos de los conceptos más comunes de Java 1.5.
- ✓ La generación de código Ada, ANSI C ++, C++, CORBA, Java y Visual Basic, con capacidad de sincronización modelo- código configurables.
- ✓ Soporte Enterprise Java Beans™ 2.0.
- ✓ Capacidad de análisis de calidad de código.
- ✓ El Add-In para modelado Web provee visualización, modelado y las herramientas para desarrollar aplicaciones Web.

Enterprise Architect.

Enterprise Architect es una herramienta de modelado multiusuario[9] diseñada para el desarrollo de sistemas robustos y duraderos. Cubre el ciclo de vida completo para el desarrollo de un

[9] Multiusuario: Propiedad que permite proveer servicio y procesamiento a múltiples usuarios simultáneamente.

proyecto, para lo que brinda capacidades de gestión de requisitos y permite la realización de especificaciones de alto nivel a modelos de análisis, diseño, implementación, pruebas y mantenimiento, usando UML y BPMN.

Entre sus características se encuentra la rapidez pues carga modelos grandes en pocos segundos, proporciona una trazabilidad completa desde requisitos, análisis y diseño hasta la implementación y despliegue. Posibilita un modelado de procesos de negocios con extensiones personalizadas, de base de datos, así como modelar diagramas estructurales y de comportamiento.

Esta herramienta genera código fuente en C++, Java, C#, VB.Net, Visual Basic, Delphi, PHP, Python y ActionScript. Cuenta con Plug-ins[10] para vincularse a Visual Studio, NET o Eclipse. La ingeniería inversa para muchos de los sistemas populares DBMS, como Oracle, SQL Server, My SQL, Access y Postgre SQL es otra de las habilidades que oferta. Algunas de sus características principales son:

- ✓ Software propietario.
- ✓ Disponibilidad en múltiples plataformas (Windows, Linux, etc.)
- ✓ Documentación de alta calidad compatible con Microsoft Word.
- ✓ Intuitivo y simple de usar.
- ✓ Bajo costo de licencias.
- ✓ Corrector Ortográfico (2006)

Visual Paradigm.

Visual Paradigm es una herramienta de diseño que soporta todos los diagramas UML y de entidad-relación. Ofrece amplias características de modelado de caso de uso incluyendo la función completa de UML, Diagrama de Casos de Uso, flujo de eventos y la generación de un diagrama de actividad. Produce la documentación completa del sistema en formato PDF, HTML y MS Word. Los desarrolladores pueden diseñar la documentación del sistema y los

[10] Plug-ins: Módulo de hardware o software que añade una característica o un servicio específico a un sistema más grande.

analistas estimar las consecuencias de los cambios a través de los diagramas. Es una herramienta CASE profesional que soporta el ciclo de vida completo del desarrollo de software: análisis y diseño orientados a objetos, construcción, pruebas y despliegue. Permite dibujar todos los tipos de diagramas de clases, código inverso, generar código desde diagramas. Soporta un conjunto de lenguajes, tanto en generación de código e ingeniería inversa como Java, C + +, PHP y XML. (Visual Paradigm, 2008). Algunas de sus caracteríticas principales son:

- ✓ Disponibilidad en múltiples plataformas (Windows, Linux, etc.)
- ✓ Diseño centrado en casos de uso y enfocado al negocio que generan un software de mayor calidad.
- ✓ Uso de un lenguaje estándar común a todo el equipo de desarrollo que facilita la comunicación.
- ✓ Capacidades de ingeniería directa e inversa.
- ✓ Modelo y código que permanece sincronizado en todo el ciclo de desarrollo.
- ✓ Disponibilidad de múltiples versiones, para cada necesidad.

Justificación de la herramienta seleccionada.

Teniendo en cuenta las características de las herramientas analizadas, se decide utilizar Visual Paradigm debido a que es una herramienta multiplataforma, robusta, de fácil uso y que da la posibilidad de exportación de documentos. Posee alta capacidad de integración con el lenguaje PHP, el cual será utilizado para el desarrollo del sistema. Esta elección se ve sustentada en el hecho de que la UCI cuenta con la licencia para el uso de dicha herramienta y propone su uso para el desarrollo de los proyectos productivos.

1.7 Herramientas para Modelado de Prototipos no Funcionales.

Existen varias herramientas de prototipos no funcionales que representan el modelo de un sistema y nos facilitan verificar sus requisitos antes de comenzar la implementación del mismo. Además brindan la posibilidad de presentarlo ante clientes y el equipo de desarrollo para su aprobación.

Microsoft Office Visio 2007.

Microsoft Office Visio 2007 es una herramienta sofisticada que brinda plantillas y formas que ayudan a crear una amplia variedad de diagramas. Estos últimos representan información que es más comprensible que los textos y tablas complejas, pueden representar información sobre sistemas, recursos y procesos organizativos de una empresa. De manera general facilitan la visualización, el análisis y la comunicación de información compleja.

Dentro de la gran rama de facilidades que ofrece para distintas funciones, es una herramienta que brinda un fácil y eficiente manejo de prototipado, agilizando el trabajo y brindando todo tipo de forma que se necesite a solo un click.

Además cuenta con distintas vistas como Drawing Explorer que en forma de árbol se accede a las carpetas para ver todos los trabajos y seleccionar directamente el deseado, Pan & Zoom permite a la vez aumentar o disminuir la imagen y saber en qué área específica del documento se está trabajando, Size & Position permite ver todo lo referente a tamaño y posición de un componente facilitando su alineación con otros. Brinda además la posibilidad de:

- ✓ Aumentar la productividad integrando diagramas con información de varias fuentes.
- ✓ Visualizar y actuar sobre información compleja mostrando datos en diagramas.
- ✓ Analizar datos y realizar de forma sencilla el seguimiento de tendencias, la identificación de problemas y la señalización de excepciones con diagramas dinámicos.
- ✓ Comunicar información compleja con nuevas plantillas y formas.

BPWin 4.1.

BPwin 4.1 es una herramienta específica para el modelado de procesos. Permite documentar de manera clara los elementos más importantes de una organización como las actividades necesarias, forma en que se realizan y recursos que consumen. Proporciona un marco de trabajo para poder representar y entender los procesos de negocio, dando una visión exacta de lo que se hace, determinando el impacto de los diferentes sucesos y definiendo cómo los procesos interactúan unos con otros mediante flujos de información permitiendo identificar actividades poco eficientes o redundantes. (DoRol, 2007). Puede adaptarse a las exigencias

del cambio acelerado y realiza la gestión de procesos de negocio complejos. Otras de las características que presenta son:

- ✓ Diseño automatizado de procesos.
- ✓ Propiedades definidas por el usuario.
- ✓ Técnicas de integración (para documentar flujos de proceso a través de IDEF0).
- ✓ Métrica y análisis de costos.
- ✓ Interface Intuitiva.
- ✓ Preevaluación.
- ✓ Asociación de Entidades y Datos.

Axure RP Pro 5.5.

Axure RP Pro es una herramienta para crear prototipos de sitios web y aplicaciones web. Se basa en conceptos conocidos desde Visio y herramientas de diseño web, y combina lo mejor de ambos mundos. Cuenta con todo lo que se pueda necesitar para crear los prototipos de forma eficiente y permite componer la página web visualmente, añadiendo, quitando y modificando los elementos con suma facilidad. (Axure, 2007) Demuestra su grado de especialización en las anotaciones. En este punto, permite especificar el estado de cada elemento (Propuesto, Aceptado, Incorporado), el beneficio esperado (Crítico, Importante, Útil), el riesgo, la estabilidad, a quién va dirigido y a quién se le asignará la tarea. Algunas de las ventajas que presenta son:

- ✓ Flexibilidad y sencillez de uso.
- ✓ Crea un documento con especificaciones.
- ✓ Instantánea creación de prototipos funcionales.
- ✓ Permite trabajar en un mismo proyecto a varias personas a la vez.
- ✓ Generar prototipos en un formato que se comporta como páginas web reales (se puede interactuar con los formularios y las páginas pueden desplazarse).

Justificación de la herramienta para prototipado seleccionada.

Teniendo en cuenta las características de las herramientas analizadas, se decide utilizar la herramienta Axure RP Pro 5.5 debido a que es una herramienta flexible, fácil de usar, que

genera código HTML, y que posibilita crear prototipos de sitios y aplicaciones web, que es lo que el proyecto TPC espera como resultado final. Mostrando los prototipos de una forma dinámica y permitiendo al usuario interactuar con formularios, para de esta manera brindar una idea más clara de cómo será la aplicación en un futuro.

1.8 Métodos para la Medición de la Satisfacción del Cliente.

La satisfacción del cliente se logra cuando las expectativas que se generan antes de recibir un servicio son superadas por el valor que percibe una vez que lo ha recibido. El nivel general de satisfacción de los clientes puede determinarse mediante: la recolección de información acerca de las necesidades de los clientes, la evaluación que hacen respecto a diferentes aspectos del servicio brindado y la intención de volver a contratar el mismo servicio.

Las medidas proporcionan un modelo básico de desempeño y un posible modelo de excelencia que se debe tratar de alcanzar. Esto llevará a mejorar la calidad e incrementar la satisfacción de los clientes. (Adriano, 2006).

Hoja de Verificación.

La hoja de verificación conocida como hoja de comprobación o de chequeo es un método basado en la observación del comportamiento de un proceso con el fin de detectar tendencias. Facilita la recolección de datos en una forma ordenada y de acuerdo al estándar requerido en el análisis que se esté realizando.

Se diseña una tabla con columnas que identifican los eventos que se están investigando y el período de la investigación. Después se recolectan los datos dentro de cada área de eventos y se coloca una marca dentro de la columna del período.

Esta técnica permite determinar de dónde vienen los costos por baja calidad, así como las fuentes importantes de insatisfacción de los clientes.

SERVQUAL.

SERVQUAL es un método para la medición de la satisfacción del cliente, en forma de cuestionario, su propósito es evaluar la calidad de servicio ofrecida por una organización a lo largo de cinco dimensiones básicas que caracterizan a un servicio: (AITECO, 2008)

- ✓ Elementos tangibles: Representan las características físicas y apariencia del proveedor.

- ✓ Fiabilidad: Implica la habilidad que tiene la organización para ejecutar el servicio prometido de forma adecuada y constante.
- ✓ Capacidad de respuesta: Representa la disposición de ayudar a los clientes y proveerlos de un servicio rápido.
- ✓ Seguridad (Garantía): Son los conocimientos y atención mostrados por los empleados respecto al servicio que están brindando, además de la habilidad de los mismos para inspirar confianza y credibilidad.
- ✓ Empatía: Es el grado de atención personalizada que ofrecen las empresas a sus clientes.

El método, propone además cinco brechas que determinan el grado de ineficiencia en la gestión de los servicios de satisfacción del cliente. Si el valor de la brecha es positivo se puede concluir que las expectativas de los clientes fueron superadas. De ocurrir lo contrario, en que la brecha tome un valor negativo, entonces el cliente esperaba más acerca de su experiencia que lo que en realidad percibió. (Parusaraman, 1988).

Expectativas del cliente**:** Las expectativas del cliente son una medida anticipada de la calidad que el cliente espera recibir por los productos y servicios que la organización ofrece.

Calidad percibida: Tomando como entrada las expectativas del cliente, la Calidad percibida se considera asociada principalmente a dos factores: la personalización y la fiabilidad.

Valor percibido: Este parámetro expresa la relación entre la calidad obtenida y el precio pagado.

Quejas del cliente: Las quejas son la expresión más palpable de la insatisfacción. Cuanto más satisfecho está un cliente, menos ganas tiene de expresar una queja.

Fidelidad del cliente: La fidelidad del cliente es el componente crítico del modelo. Se observa que, si bien la satisfacción del cliente ocupa un lugar central en el diagrama, las flechas relacionales desembocan en este parámetro.

Método de Kano.

El método de Kano, evalúa la relación entre la funcionalidad de los productos y la satisfacción que esta funcionalidad les brinda a los clientes. Establece para cada requisito del cliente, la relación entre satisfacción y funcionalidad, permitiendo discriminar y clasificar los mismos. Kano distingue tres tipos de requisitos del producto (Figura 4) que influyen de diferente manera en la satisfacción del cliente. (Leon, 2005).

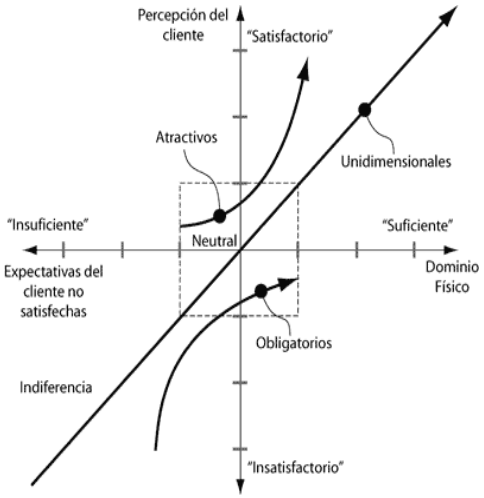


Fig. 4 Representación Gráfica del Método de Kano.

Requisitos Obligatorios: Son aquellos que aumentan la satisfacción en relación directa con la funcionalidad, pero superado cierto umbral dejan de producir un incremento importante en la satisfacción. Si estos requisitos no se cumplen, entonces el cliente estará sumamente disconforme.

Requisitos Unidimensionales: Se caracterizan porque la satisfacción que producen aumenta de modo aproximadamente proporcional al nivel de funcionalidad y satisfacción del cliente. Los requisitos unidimensionales están explícitamente demandados por el cliente.

Requisitos Atractivos: Son aquéllos que por debajo de cierto umbral de funcionalidad, mantienen un nivel de satisfacción relativamente bajo y constante, pero una vez superado ese umbral, producen un aumento significativo de la satisfacción.

La determinación de los requisitos consta de cuatro pasos: (a) identificación de los requisitos del producto; (b) construcción del cuestionario Kano; (c) administración de la entrevista al cliente; y (d) evaluación e interpretación.

El método se basa en la aplicación de un cuestionario a los clientes, donde el análisis e interpretación de los datos recolectados permite disponer del mapa de respuestas de clientes. Este mapa refleja la clasificación de los requisitos funcionales necesarios para lograr la conceptualización del producto y su clasificación.

En el cuestionario se formulan dos preguntas por cada requisito analizado:

- ✓ ¿Cómo se siente si la característica X está presente en el producto?

  (Requisitos funcionales)

- ✓ ¿Cómo se siente si la característica X NO está presente en el producto?

  (Requisitos disfuncionales)

Para cada pregunta el cliente responde entre cinco posibles opciones:

- ✓ Me agrada
- ✓ Es de esperarse
- ✓ Neutral
- ✓ Lo acepto
- ✓ Me desagrada

Una vez respondidas las dos preguntas se busca su combinación en la Tabla 1, y es de esta forma que los requisitos de un producto se pueden clasificar en una de las seis categorías que se muestran a continuación: Atractivo (A), Obligatorio (O), Unidimensional (U), Indiferencia (I), Respuesta Inversa (Inv), Respuesta Dudosa (D).

	Requisitos Disfuncionales					
		1	2	3	4	5
	1	D	A	A	A	U
	2	Inv	I	I	I	O
	3	Inv	I	I	I	O

Requisitos Funcionales	4	Inv	I	I	I	O
	5	Inv	Inv	Inv	Inv	D

Tabla 1. Clasificación de los Requisitos.

Un cliente se comporta *indiferente* cuando la funcionalidad respecto al requisito no se refleja en un aumento o disminución de la satisfacción del cliente y se representa como una recta paralela al eje horizontal de la figura 4.

Una respuesta *inversa* indica que la interpretación de criterios funcionales y disfuncionales del diseñador es la inversa a la percepción del cliente.

Se clasifica como una respuesta *dudosa* cuando existe una contradicción en las respuestas a las preguntas.

Método Seleccionado

Se selecciona el método Kano para la medición de la satisfacción del cliente pues le permite al diseñador la clarificación de los requisitos funcionales o necesidades del cliente.

Conclusiones.

El estudio de las diferentes metodologías de desarrollo de software, lenguajes de modelado, herramientas CASE y herramientas para el modelado de prototipos, permitió conocer cuáles son las que más se ajustan para el desarrollo del subsistema, en este caso las seleccionadas fueron: RUP, UML, Visual Paradigm y Axure. A través del análisis de las diferentes etapas de la IR propuestas por varios autores, se decide utilizar las enunciadas por Pressman, teniendo en cuenta las características y condiciones del proyecto y de su equipo de desarrollo. Se decide utilizar el método de Kano para medir la satisfacción del cliente, sentando las bases para comenzar la realización del modelado del subsistema Administración y Gobierno del proyecto Tribunales Populares Cubanos.

CAPÍTULO 2. PROPUESTA DE SOLUCIÓN

2.1 Introducción.

A partir de lo investigado en el capítulo anterior, en el presente se realiza el Modelo de Dominio correspondiente al subsistema de Administración y Gobierno. A partir de las actividades que RUP propone, se realiza el modelado del sistema con los artefactos pertinentes: se realiza la especificación de requisitos funcionales y requisitos no funcionales, se definen los actores, casos de uso y su descripción detallada junto con el diagrama de casos de uso del sistema.

2.2 Modelo de Dominio.

El Modelo de Dominio (o Modelo Conceptual) es una representación visual de los conceptos u objetos del mundo real significativos para un problema o área de interés. Define un modelo de clases común para todos los implicados en el desarrollo, representadas en objetos del dominio, sirve como interlocutor entre clientes y desarrolladores. El propósito fundamental de este modelo es generar una terminología común y sentar las bases del entendimiento del desarrollo y no para definir el sistema completo. En el caso específico del subsistema Administración y Gobierno, se decide realizar un Modelo de Dominio debido a que no se cuentan con procesos del negocio bien definidos.

2.2.1 Realización del Modelo de Dominio.

Los tribunales populares cubanos están organizados de forma jerárquica, contándose con un tribunal Supremo que radica en Ciudad de La Habana, un Tribunal Provincial en cada provincia y un Tribunal Municipal en cada municipio. Estos están constituidos por salas o secciones en el caso de los municipales, que atienden materias: Laboral, Penal, Civil, Económico, Administrativo o combinaciones de éstas. Donde en dependencia de las materias serán los asuntos que se atenderán en la sala o sección.

Los tribunales tendrán asociados usuarios, cada uno con roles específicos entre los que se encuentran: Presidente de tribunal, Vice-Presidente, Secretaria de gobierno, Secretaria judicial, Secretaria auxiliar, Juez, Juez de ejecución, Secretaria asistente, Abogado, Directivos, Administrador, Supervisor y Técnicos de estadísticas, en el caso de los Jueces lego se le configurará el periodo de tiempo que trabajarán en un determinado tribunal.

Cada tribunal deberá regirse por una plantilla de calendario para el establecimiento de los días no hábiles, la cual se creará a partir de un conjunto de reglas; estas pueden ser fijas para todos los años (1ro de enero, 1ro de mayo, 26 de julio, 10 de octubre, 25 de diciembre, etc.) o agregarse durante el transcurso del mismo debido a factores específicos de cada tribunal (situaciones climatológicas, etc.) Se deberá establecer además la duración del día hábil, siendo de gran importancia para el seguimiento del vencimiento de los términos.

Cada tribunal deberá regirse además por un calendario de señalamientos, el cual va a permitir el señalamiento de los actos judiciales públicos, estas que pueden ser vistas o juicios.

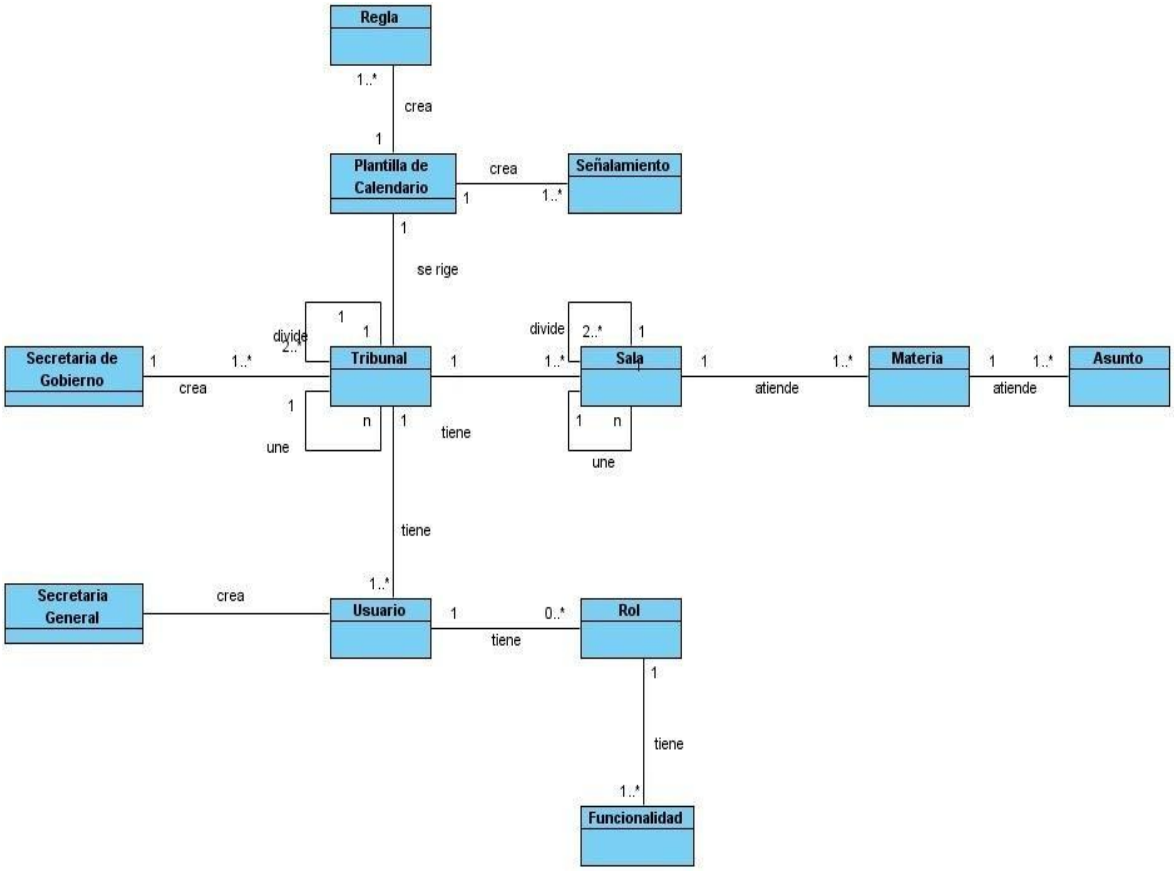


Fig.5 Modelo de Dominio del Subsistema Administración y Gobierno.

A partir de la necesidad de comprender las relaciones que se establecen entre las diferentes clases conceptuales se realiza la descripción de los principales conceptos representados en el Modelo del Dominio:

Tribunales: sistema de órganos estatales encargado de ejercer la máxima autoridad judicial en el país y cuyas decisiones, en este orden, son definitivas.

Sala: estructura organizativa por la que se compone el tribunal Supremo, los tribunales provinciales y que en el caso de los municipales se le llamarán secciones. En dependencia de la densidad de población, el volumen o la naturaleza de los asuntos, así será el número de salas existentes y el número de materias que atenderá.

Materia: clasificación que se le da a cada uno de los conflictos a resolver en el tribunal. Las materias pueden ser de lo: Penal, Laboral, Civil, Administrativo y Económico.

Asunto: pueden ser causas o expedientes en dependencia de la materia, el cual contiene toda la documentación del proceso.

2.3 Técnicas utilizadas en las etapas de la IR.

Para la obtención de los requisitos del subsistema se utilizó la técnica de Tormenta de Ideas durante las primeras reuniones con los clientes, para obtener una vista general del subsistema Administración y Gobierno que se lleva a cabo en los Tribunales Populares Cubanos e identificar las necesidades fundamentales que el sistema debía satisfacer; estas primeras ideas se fueron concretando mediante las Entrevistas en las cuales se detallaron cada una de las funcionalidades que el sistema debía cumplir, se utilizaron como técnicas de apoyo la Observación y Análisis de Tareas, e Introspección. Para la especificación de requisitos se utilizó los Casos de Uso, los cuales describieron las interacciones entre el sistema y los actores. La validación se realizó a través de los Prototipos, estos permitieron crear un modelo del software que se desea construir, así como mejorar la comunicación entre clientes y desarrolladores. Para la gestión se empleó la matriz de trazabilidad, muy usada durante las primeras etapas del proceso de desarrollo para verificar si los casos de uso del sistema satisfacen cada uno de los requisitos identificados.

2.4 Requisitos de Software.

Los requisitos de software deben ser especificados por escrito, como todo contrato o acuerdo entre dos partes, deben ser posibles de probar o verificar debido a que si un requisito no se puede comprobar no hay forma de saber si se cumplió con él o no, deben estar descritos como una característica del sistema a entregar, esto es, lo que el sistema debe hacer y no cómo

debe hacerlo. Es importante destacar que deben estar redactados de la manera más abstracta y concisa posible para evitar malas interpretaciones. (UCI, curso 2007-2008)

Los Requisitos Funcionales son capacidades o condiciones que el sistema debe cumplir. Es una tarea simple enunciada con un solo verbo y se corresponde con futuras opciones, acciones ocultas y condiciones extremas a determinar por el software. Los Requisitos No Funcionales son propiedades o cualidades que el producto debe tener. Debe pensarse en estas propiedades como las características que hacen al producto atractivo, usable, rápido o confiable. (UCI, curso 2007-2008).

2.4.1 Requisitos Funcionales.

A continuación se muestran los 45 requisitos funcionales identificados en el subsistema Administración y Gobierno, para ver los detalles, ver el documento entregable Especificación de Requisitos de Software.

Requisitos Funcionales		
RF.01 Crear tribunal.	**RF.16** Listar Salas o Secciones.	**RF.31** Visualizar acciones de los usuarios
RF.02 Modificar tribunal.	**RF.17**Crear Usuario.	**RF.32** Ver detalles de acciones de los usuarios
RF.03 Activar Tribunal.	**RF.18** Modificar usuario.	**RF.33** Registrar acción.
RF.04 Desactivar Tribunal.	**RF.19** Asignar roles a usuario.	**RF.34** Adicionar regla.
RF.05 Dividir tribunal.	**RF.20** Activar Usuario.	**RF.35** Modificar regla.
RF.06 Unir Tribunal.	**RF.21** Desactivar Usuario.	**RF.36** Eliminar regla.
RF.07 Buscar Tribunal.	**RF.22** Autenticar usuarios.	**RF.37** Ver calendario.
RF.08 Listar tribunales.	**RF.23** Cambiar contraseña de un usuario.	**RF.38** Configurar horas hábiles.
RF.09 Crear sala o sección.	**RF.24** Restaurar contraseña de un usuario.	**RF.39** Adicionar señalamiento.
RF.10 Modificar sala o sección.	**RF.25** Buscar usuarios.	**RF.40** Eliminar señalamiento.
RF.11 Activar Sala o	**RF.26** Listar usuarios.	**RF.41** Adicionar

Sección.		señalamientos automáticamente.
RF.12 Desactivar Sala o Sección.	**RF.27** Adicionar Rol.	**RF.42** Listar Jueces Legos.
RF.13 Dividir Sala o Sección.	**RF.28** Modificar Rol.	**RF.43** Buscar Jueces Legos.
RF.14 Unir Sala o Sección.	**RF.29** Eliminar Rol.	**RF.44** Asociar fecha a Juez Lego
RF.15 Buscar Sala o Sección.	**RF.30** Listar Roles	**RF.45** Transferir asuntos.

Tabla 2. Requisitos Funcionales de Software.

2.4.2 Requisitos No Funcionales.

A continuación se muestran algunos de los requisitos no funcionales de los 34 identificados por el equipo de arquitectura del proyecto TPC, para verlos todos ver el documento entregable Especificación de Requisitos no Funcionales.

Requisitos no Funcionales	
Requisitos de Usabilidad	**RNF.01.** El software tendrá siempre la posibilidad de ayuda disponible para cualquier tipo de usuario, lo que le permitirá un avance considerable en la explotación de la aplicación en todas sus funcionalidades. **RNF.06.** Debe poseer una interfaz agradable para el cliente.
Requisitos de Fiabilidad	**RNF.07.** El sistema estará disponible 24 horas al día, 7 días a la semana. **RNF.08.** Disponibilidad de los casos asignados desde cualquier parte del país. **RNF.11.** La precisión y exactitud requerida en las salidas del sistema o sea el máximo de errores, es de 5 errores/MLC.

	RNF.12. La herramienta de implementación a utilizar tiene soporte para recuperación ante fallos y errores.
Requisitos de Eficiencia	**RNF.13.** Tiempo de respuesta promedio de las peticiones que se realizan al servidor no deberá ser mayor de 3 segundos. **RNF.14.** El número de clientes o transacciones que el sistema puede alojar es de 2000.
Requisitos de Seguridad	**RNF.17.** El sistema debe garantizar la confidencialidad, integridad y disponibilidad de la información que se procese en el sistema. **RNF.18.** La seguridad se establecerá por roles que se le asignarán a los usuarios que interactúen con el sistema.
Requisitos de Soporte	**RNF.21.** Soporte para grandes volúmenes de datos y velocidad de procesamiento. **RNF.22.** Tiempo de respuesta rápido en accesos concurrentes.
Requisitos de Restricciones de Diseño	**RNF.23.** El lenguaje de programación es PHP.V. 5.5.9. **RNF.24.** El framework de desarrollo es Sauxe 1.0. **RNF.25.** La herramienta IDE de desarrollo utilizada será NetBeans. **RNF.26.** La herramienta case utilizada es el Visual Paradigm para UML .V.6.1. **RNF.27.** La herramienta case utilizada para el modelado de la Base de Datos ER/Studio.V.7.1. **RNF.28.** Se utilizará el patrón de arquitectura Modelo – Vista – Controlador. **RNF.29.** La herramienta gestor de base de datos es el PostgreSQL 8.4.

Requisitos de Interfaz	**RNF.32.** Diseño sencillo, con pocas entradas, permitiendo que no sea necesario mucho entrenamiento para que los usuarios puedan utilizar el sistema.

Tabla 3. Requisitos no Funcionales de Software.

2.5 Definición de actores del Sistema.

A continuación se muestran los futuros usuarios del sistema identificados y las respectivas funcionalidades que podrán llevar a cabo:

Actor	Descripción
Usuario general.	Actor encargado de autenticarse en el sistema y cambiar su contraseña. Este rol lo juegan todos los usuarios que interactuarán con el sistema: Presidente de tribunal, Vice-Presidente, Secretaria de gobierno, Secretaria judicial, Secretaria auxiliar, Juez, Juez lego, Juez de ejecución, Secretaria asistente, Abogado, Supervisor y Técnicos de estadísticas.
Secretaria de gobierno.	Actor encargado de la gestión de instancias en el Tribunal Supremo Popular y de las plantillas de Calendario.
Secretaria general.	Actor encargado de la gestión de usuarios de una instancia, este rol lo juegan las Secretarias de Gobierno y las Secretarias Judiciales.
Directivos.	Actor encargado de regular el total funcionamiento de sus respectivas instancias. Tendrá acceso a todas las funcionalidades con propósitos de organización y dirección, así como visualizar las trazas del sistema. Este rol lo juegan tanto Presidentes de Tribunal, Vice

	presidentes y Secretaria de gobierno, Supervisor y Técnicos de Estadísticas.

Tabla 4. Descripción de los actores del sistema.

2.6 Patrones de CU utilizados.

A los requisitos identificados se le aplicaron una serie de Patrones de Casos de Uso para agruparlos en Casos de Uso y estructurar los Diagramas de Caso de Uso, lo que permitió ganar en tiempo, en organización y reflejar con mayor precisión las necesidades reales del cliente. Los patrones utilizados son los que se muestran a continuación, descritos por (Övergaard & Palmkvist, 2004):

Inclusión Concreta: En este patrón existe una relación de inclusión entre el caso de uso base y el caso de uso incluido. Este último puede ser instanciado por sí solo. El caso de uso base puede ser concreto o abstracto. Se utiliza este patrón cuando un flujo de datos se puede incluir en el flujo de datos de otro caso de uso y también realizarse por sí solo.

Extensión Concreta: Este patrón consiste en dos casos de uso y una relación de extensión entre ellos. El caso de uso extendido es concreto, es decir, este puede ser instanciado por sí solo, así como, ser una extensión del caso de uso base. El caso de uso base puede ser concreto o abstracto. Este patrón es aplicable cuando un flujo de datos puede ser extendido del flujo de datos de otro caso de uso, así como ser ejecutado por sí solo.

Concordancia – Adición: este patrón extrae una subsecuencia de acciones que aparecen en diferentes lugares del flujo de casos de uso y es expresado por separado. La subsecuencia común de casos de uso, extiende los casos de uso compartiendo la subsecuencia de acciones. Los otros casos de uso modelan el flujo que será expandido con la subsecuencia. Este patrón es preferible usarlo cuando otros casos de uso se encuentran propiamente completos, o sea, que no requieren de una subsecuencia común de acciones para modelar los usos completos del sistema.

Actores Múltiples (Rol común): Es un patrón de estructura que plantea que cuando dos actores juegan el mismo papel hacia un caso de uso se representa otro actor, del que heredan los actores que comparten este rol. Este patrón es aplicable cuando, desde el punto de vista de

un caso de uso, hay solo una entidad externa interactuando con cada instancia del caso de uso.

CRUD (Crear, Modificar, Eliminar, Mostrar) – Completo: este patrón se basa en la fusión de casos de uso simples para formar una unidad conceptual. Consta de un caso de uso, llamado Información CRUD o Gestionar información que modela todas las operaciones que pueden ser realizadas sobre una parte de la información de un tipo específico, tales como creación, lectura, actualización y eliminación. Suele ser utilizado cuando todos los flujos contribuyen al mismo valor del negocio, y estos a su vez son cortos y simples.

CRUD (Crear, Modificar, Mostrar) – Parcial: patrón alternativo que modela una de las vías de los casos de uso como un caso de uso separado. Es preferiblemente utilizado cuando una de las alternativas de los casos de uso es más significativa, larga o más compleja que las otras.

2.7 Diagrama de Paquetes.

Los requisitos funcionales obtenidos fueron agrupados en casos de uso. Estos últimos se estructuraron en cuatro paquetes, lo que permitió mantener un mayor grado de organización a la hora de consultar la documentación generada para esta etapa.

Paquete Gestión de Instancias: Este paquete agrupa los casos de usos referentes a la gestión de instancias.

- ✓ Gestionar Tribunal.
- ✓ Gestionar Sala.
- ✓ Transferir asuntos.
- ✓ Reestructurar Sala o Sección.
- ✓ Reestructurar Tribunal.

Paquete Seguridad: Este paquete se encarga de la seguridad del sistema, permitiendo restringir funcionalidades por roles.

- ✓ Cambiar Contraseña.
- ✓ Restaurar Contraseña.

- ✓ Gestionar Rol.
- ✓ Autenticar Usuario.

Paquete Gestión de Usuarios: Este paquete contiene los casos de usos referentes a la gestión de usuarios, desde su creación hasta las trazas que se generan.

- ✓ Gestionar Usuario.
- ✓ Asignar Roles a Usuario.
- ✓ Visualizar Trazas de Usuario.
- ✓ Buscar Usuario.

Paquete Gestión de Calendario: Este paquete contiene los casos de usos referentes a la configuración de las plantillas de calendarios, tanto el calendario de los días no hábiles como el calendario de señalamientos.

- ✓ Configurar Calendario de Días no hábiles.
- ✓ Gestionar Regla.
- ✓ Configurar Calendario de señalamientos.
- ✓ Gestionar Señalamiento.

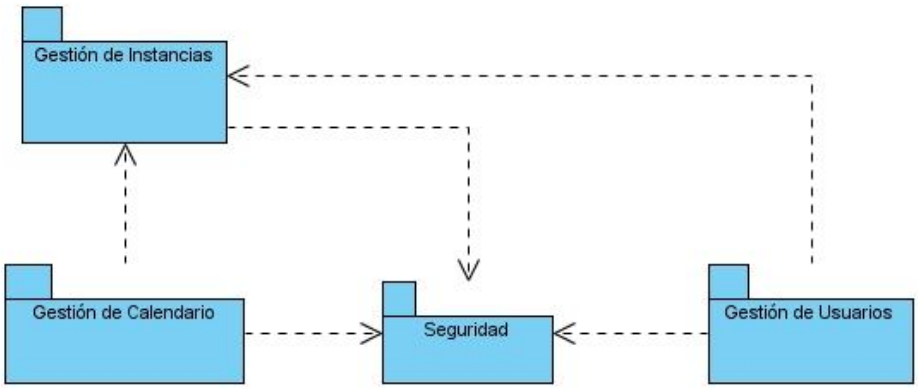


Fig.6 Diagrama de Paquetes del Subsistema Administración y Gobierno.

2.8 Diagramas de Casos de Uso del Sistema (DCUS).

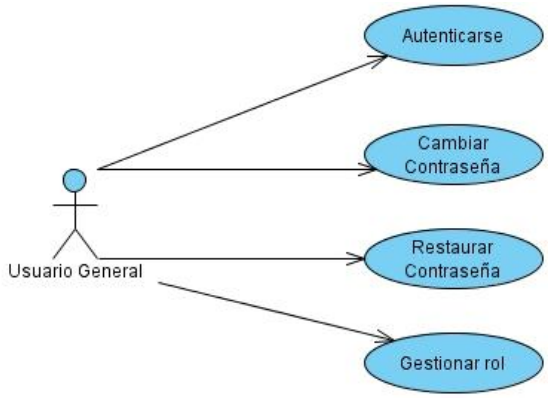


Fig.7 Diagrama de Caso de Uso del Paquete Seguridad.

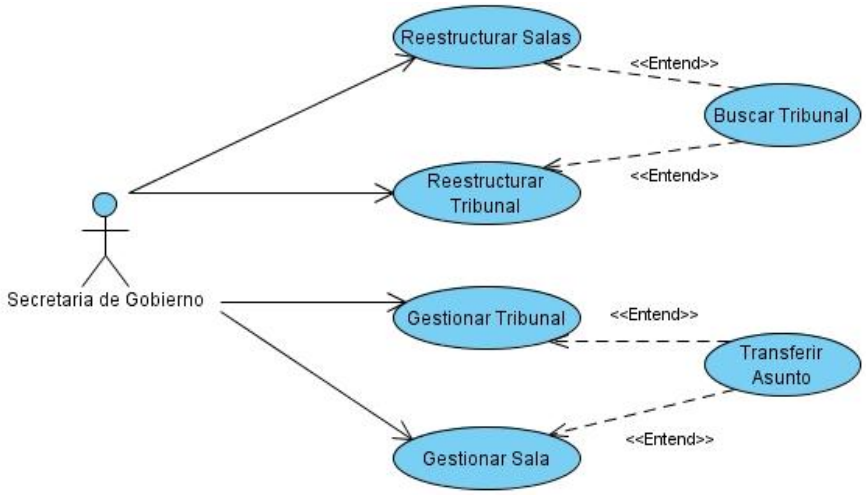


Fig. 8 Diagrama de Caso de Uso del Paquete Gestión de Instancias.

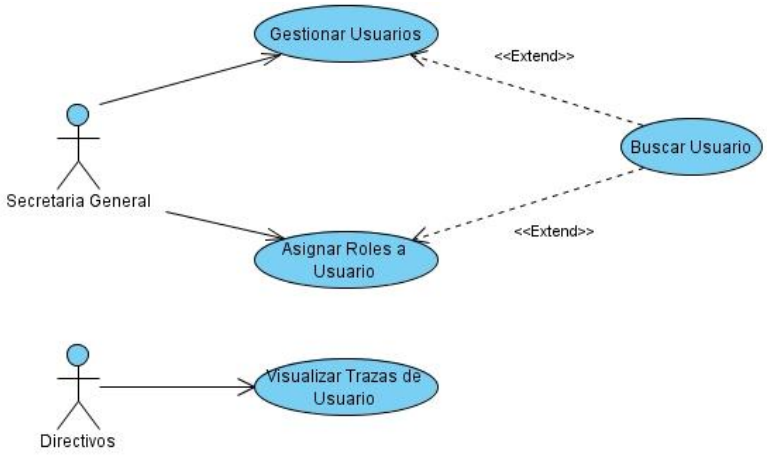


Fig.9 Diagrama de Caso de Uso del Paquete Gestión de Usuarios.

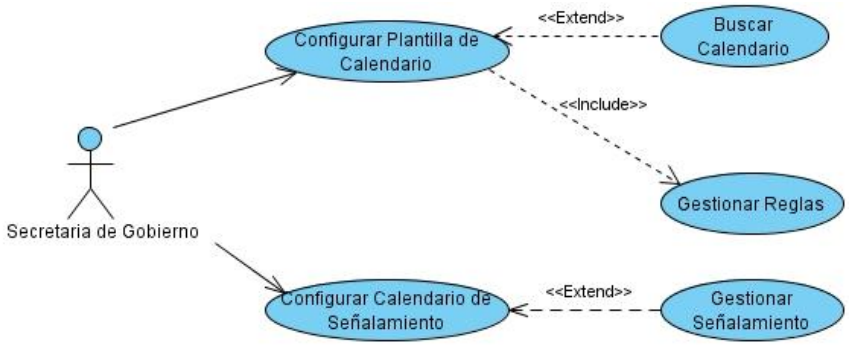
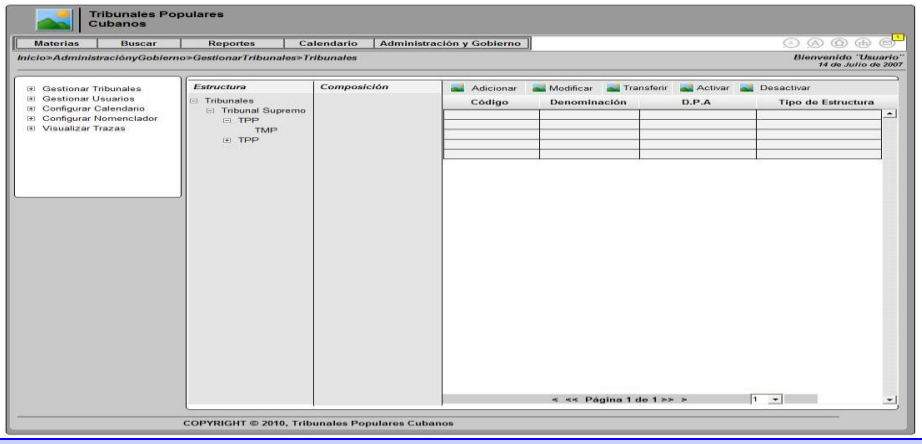


Fig.10 Diagrama de Caso de Uso del Paquete Gestión de Calendarios.

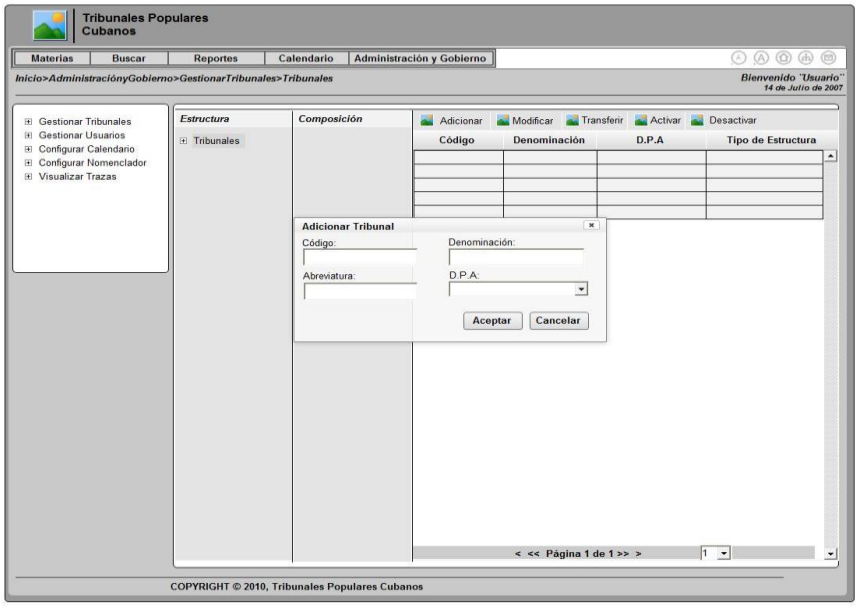
2.9 Descripción de los Casos de Uso del Sistema.

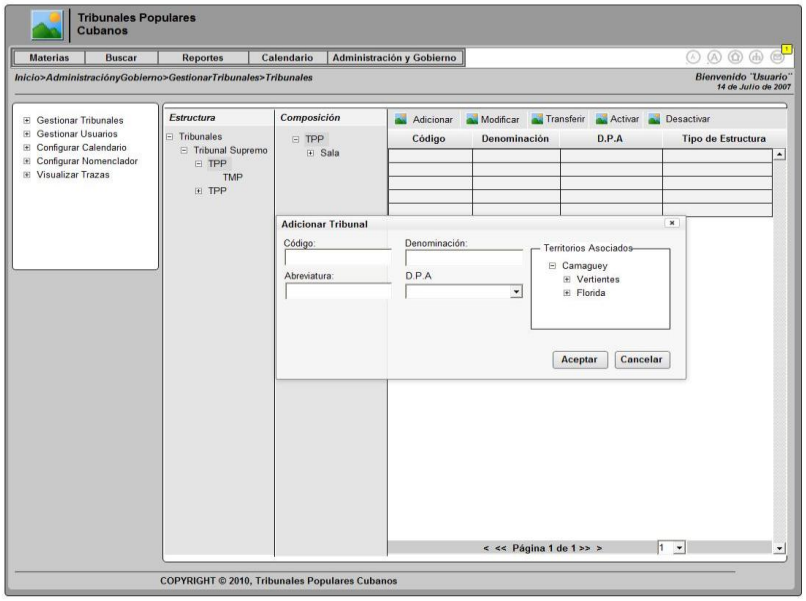
A continuación se describen los Casos de Uso, solo se mostrará la descripción del Caso de Uso Gestionar Tribunal, del paquete Gestión de Instancias que ha sido evaluado de crítico por su significación, los restantes se encuentran en el documento entregable Modelo del Sistema.

Caso de Uso:	**Gestionar Tribunal.**
Actor:	Secretaria de gobierno.
Resumen:	El caso de uso se inicia cuando la Secretaria de gobierno necesita gestionar un tribunal de cualquier instancia. Permite introducir en el sistema los tribunales, así como seleccionar un tribunal existente ya sea para modificarlo, activarlo o desactivarlo, así como transferir asuntos de un tribunal a otro. El caso de uso termina con la creación, actualización, activación o desactivación de un tribunal, así como transferir asuntos de un tribunal a otro.
Precondiciones:	• El sistema debe estar instalado y ejecutándose correctamente. • Debe ser inicializado por la Secretaria de gobierno. • El usuario y la contraseña deben ser correctos.
Referencias:	RF.01, RF.02, RF.03, RF.04. CU extendido “Transferir asuntos”.
Prioridad:	Crítico.

<table>
<tr><th colspan="2">Flujo Normal de Eventos</th></tr>
<tr><th>Acción del Actor</th><th>Respuesta del Sistema</th></tr>
<tr><td>1-El caso de uso comienza cuando el actor Secretaria de gobierno solicita la opción “Gestionar Tribunales”.</td><td>2-El sistema muestra una interfaz con los tribunales existentes en el sistema.</td></tr>
<tr><td>3-Selecciona un tribunal y escoge una de las siguientes opciones:
• Adicionar (ver Sección 1: ”Adicionar Tribunal”)
• Modificar (ver Sección 2: ”Modificar Tribunal”)
• Activar (ver Sección 3: ”Activar Tribunal”)
• Desactivar (ver Sección 4: ”Desactivar Tribunal”)
• Transferir asuntos de un tribunal (ver CU extendido ”Transferir Asuntos de un tribunal”)</td><td></td></tr>
<tr><td colspan="2">Prototipo de Interfaz</td></tr>
<tr><td colspan="2"> 
</td></tr>
<tr><td colspan="2">Sección1: “Adicionar Tribunal”</td></tr>
</table>

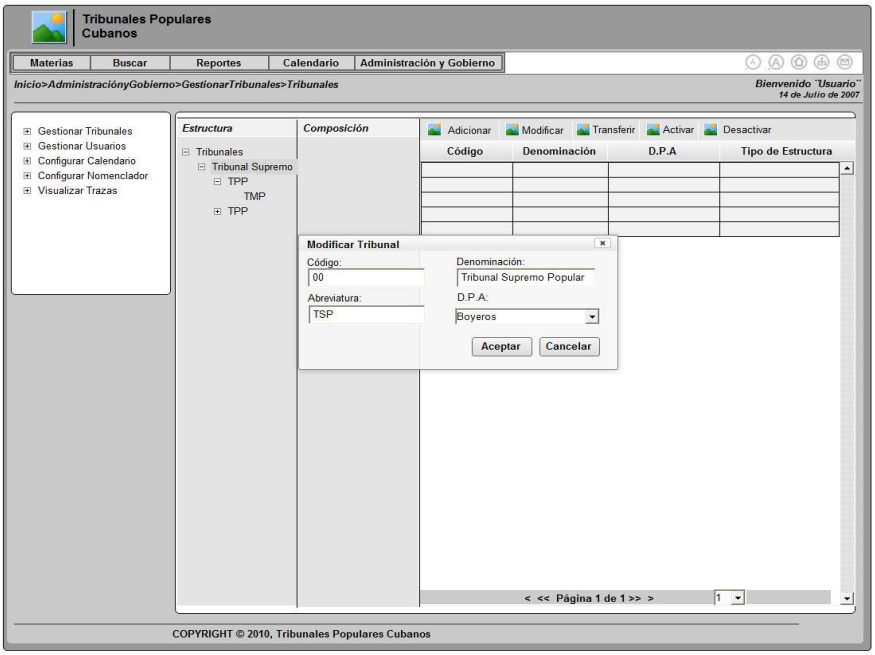
Acción del Actor	Respuesta del Sistema
	1- El sistema muestra una interfaz con los campos requeridos para la creación del tribunal: • Código (Obligatorio). • Denominación (Obligatorio). • Abreviatura (Obligatorio). • D.P.A (División política administrativa) (Obligatorio). Para el caso de los tribunales municipales se mostrará el siguiente campo: • Territorios asociados (Obligatorio).
2- Inserta los datos.	
3- Solicita aceptar la operación.	4- El sistema valida que los datos introducidos son correctos y que no hay campos obligatorios vacíos.
	5- El sistema crea el nuevo tribunal, terminando así el caso de uso.
Prototipo de Interfaz	





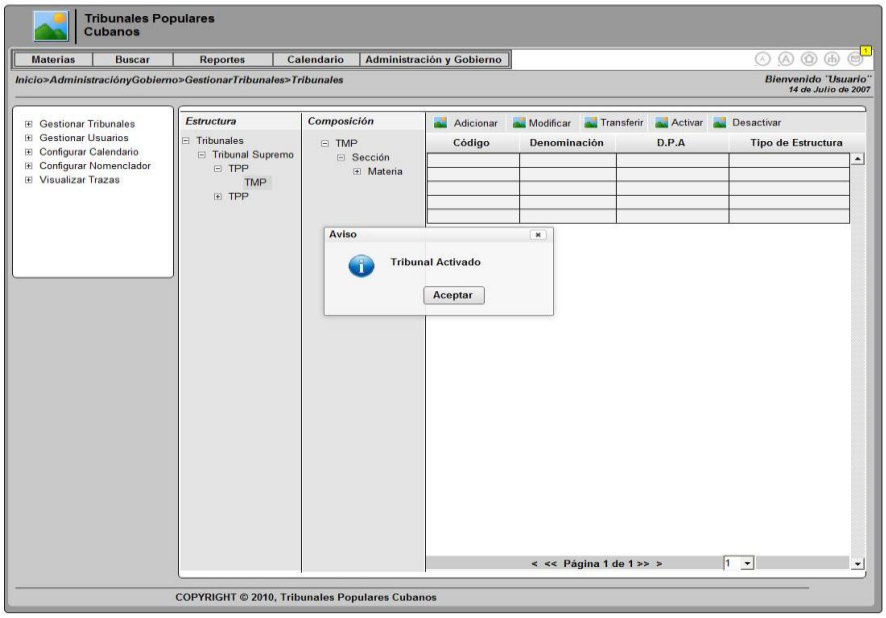
Flujos Alternos	
Flujo alterno al paso 3 “Operación cancelada”	
Acción del Actor	**Respuesta del Sistema**
3-Solicita cancelar la operación	3.1- Cancela la Operación y vuelve a la interfaz previa.

Flujo alterno al paso 4 “Datos incorrectos y/o campos vacíos”	
Acción del Actor	**Respuesta del Sistema**
	4.1- El sistema muestra símbolos de error resaltando los campos obligatorios vacíos y/o donde se introdujeron datos incorrectos. Terminando así el caso de uso.
Sección 2: “Modificar Tribunal”	
Acción del Actor	**Respuesta del Sistema**
	1-El sistema muestra una interfaz con los campos requeridos para la modificación del tribunal: • Código (Obligatorio). • Denominación (Obligatorio). • Abreviatura (Obligatorio). • D.P.A (División política administrativa) (Obligatorio). Para el caso de los tribunales municipales se podrá modificar el siguiente campo: • Territorios asociados (Obligatorio).
2-Inserta los datos.	
3-Solicita aceptar la operación.	4- El sistema valida que los datos introducidos son correctos y que no hay campos obligatorios vacíos.
	5- El sistema modifica el tribunal, terminando así el caso de uso.
Prototipo de Interfaz	

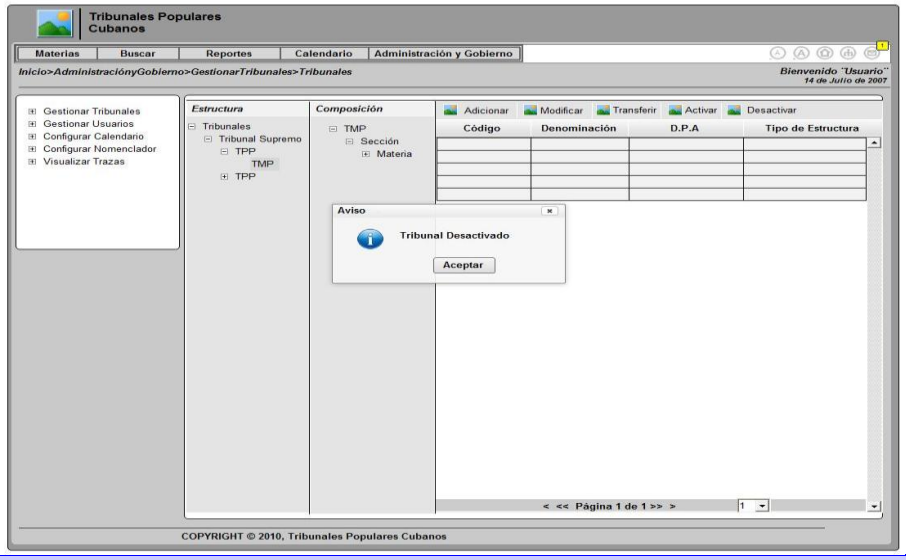


Flujos Alternos	
Flujo alterno al paso 3 "Operación cancelada"	
Acción del Actor	**Respuesta del Sistema**
3-Solicita cancelar la operación.	3.1- Cancela la Operación y vuelve a la interfaz previa.
Flujo alterno al paso 4 "Datos incorrectos y/o campos vacíos"	
Acción del Actor	**Respuesta del Sistema**
	4.1- El sistema muestra símbolos de error resaltando los campos obligatorios vacíos y/o donde se introdujeron datos incorrectos. Terminando así el caso de uso.
Sección 3: "Activar Tribunal"	
Acción del Actor	**Respuesta del Sistema**
	1-El sistema muestra un mensaje confirmando la activación del tribunal.
2-Acepta la operación, terminando así el	

caso de uso.	
Prototipo de Interfaz	



Flujo alterno al paso 2 “Operación denegada”	
Acción del Actor	**Respuesta del Sistema**
2-Solicita denegar el mensaje de confirmación.	2.1- Mantiene la interfaz previa al mensaje de confirmación pero sin persistir los datos.
Sección 4: “Desactivar Tribunal”	
Acción del Actor	**Respuesta del Sistema**
	1-El sistema muestra un mensaje confirmando la desactivación del tribunal.
2-Acepta la operación, terminando así el caso de uso.	
Prototipo de Interfaz	



Flujo alterno al paso 2 "Operación denegada"	
Acción del Actor	**Respuesta del Sistema**
2-Solicita denegar el mensaje de confirmación.	2.1- Mantiene la interfaz previa al mensaje de confirmación pero sin persistir los datos.
Poscondiciones:	Quedaron creados los tribunales en el sistema. Quedaron modificados los tribunales. Quedaron activados los tribunales. Quedaron desactivados los tribunales.

2.10 Gestión de Requisitos del subsistema Administración y Gobierno.

Como parte de las actividades de la gestión de los requisitos, para seguir y controlar el cambio en los mismos, se puso en práctica la técnica de trazabilidad de requisitos. Esta técnica trata de relacionar cada requisito del sistema (compilados, a partir de las necesidades expresadas por el cliente, en el documento Especificación de Requisitos Funcionales) con su caso de uso.

De las cinco matrices de seguimiento definidas por Pressman, se utilizó la matriz de seguimiento de dependencias, la cual indica cómo se relacionan los requisitos entre sí, permitiendo determinar cómo afecta el impacto de un cambio en los mismos. La matriz de trazabilidad de requisitos, permitió determinar si todos los casos de uso del sistema satisfacen

cada uno de los requisitos identificados. La figura 11 muestra una porción de la matriz de trazabilidad, construida a partir de los casos de uso del sistema y los requisitos funcionales especificados con anterioridad. Para ver la matriz completa ver Documento entregable Matriz de Trazabilidad.

	Requisitos														
Casos de Uso	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
Gestionar Usuario															
Asignar roles a usuario															
Buscar Usuario.															
Visualizar Trazas															
Autenticar Usuario															
Cambiar Contraseña															
Restaurar Contraseña															
Gestionar Rol.															
Configurar Plantilla de Calendario.															
Gestionar Regla															
Gestionar Tribunal	x	x	x	x											
Gestionar Sala o Sección									x	x	x	x			
Reestructurar Tribunal.					x	x									
Reestructurar sala o sección													x	x	x
Transferir asuntos.															
Buscar Tribunal							x	x							
Configurar juez lego															
Configurar Calendario de señalamientos															
Gestionar señalamiento															

Fig.11 Matriz de Trazabilidad.

Conclusiones.

El Modelo del Dominio realizado a partir de los aspectos identificados permitió entender la estructura y la dinámica de la organización Tribunales Populares Cubanos, en la cual se va a implantar la solución informática a crear. Además, mediante la realización de la disciplina Ingeniería de Requisitos y con la estrategia de obtención de requisitos puesta en práctica, se consiguió identificar las funcionalidades que el subsistema Administración y Gobierno debe brindar y las restricciones sobre las que va a operar. La identificación y especificación de los artefactos del Modelo del Sistema facilitó un mayor entendimiento y un acuerdo común entre los clientes y los desarrolladores, en cuanto a la concepción de las funcionalidades que el sistema debe cumplir.

CAPÍTULO 3. VALIDACIÓN DE LOS RESULTADOS OBTENIDOS.

3.1 Introducción.

En este capítulo se aplican métricas para medir la calidad de la especificación de los requisitos identificados y de la funcionalidad del diagrama de casos de uso del sistema. Además se aplica el método Kano para medir la satisfacción del cliente del subsistema Administración y Gobierno de proyecto Tribunales Populares Cubanos

3.2 Métricas de Software.

La métrica es una medida cuantitativa del grado en que un sistema, componente o proceso posee un atributo dado. La medida es un factor clave en el desarrollo del software ya que ayuda a entender qué está pasando durante el desarrollo y el mantenimiento, permite controlar el desarrollo del proyecto, y estimula la mejora de procesos y productos. (Adriano, 2006). Una aplicación correcta de las métricas de software permite realizar mejores estimaciones de tiempo y costo, cuantificar la productividad de los desarrolladores y del uso de las herramientas de desarrollo.

3.2.1 Métricas de la Calidad de la Especificación de Requisitos.

Existe un conjunto de características que se utilizan para valorar la calidad del análisis y la especificación de requisitos, los mismos son: especificidad (ausencia de ambigüedad), compleción, corrección, comprensión, capacidad de verificación, consistencia interna y externa, capacidad de logro, concisión, trazabilidad, capacidad de modificación, exactitud y capacidad de reutilización. (Pressman, 2005).

NR: Requisitos que hay en una especificación.

NR = NF + NNF

Donde NF es el número de Requisitos Funcionales y NNF es el número de Requisitos No Funcionales.

Luego se procede a medir la especificidad de los requisitos con la siguiente fórmula:

Q1 = Nu1 / NR

Donde Nu1 es el número de requisitos para los que todos los revisores tuvieron interpretaciones idénticas y Q1: Consistencia de la interpretación de los revisores. El valor óptimo de Q1 es 1, que significa la ausencia de ambigüedad en los requisitos lo que significa que mientras más cercano sea el valor de Q1 a 1 mayor será la consistencia de la especificación de los requisitos.

Los miembros del equipo de inspección se presentan a continuación:

Nombre y Apellidos	Rol Desempeñado
Marily Fuentes Águila	Especialista
Ortelio Ruiz Prieto	Especialista
Daylen Benítez Matos	Analista
Chaveliz Téllez Larramendi	Analista principal
Yinet Hernández	Analista

Tabla 5. Miembros del equipo de inspección.

Un resumen de los resultados obtenidos se muestra a continuación:

Atributo de calidad	Tipo de requisito	Interpretaciones	
		Iguales	Desiguales
Especificidad	Funcionales	45	2
	No Funcionales	34	2
	TOTAL	79	4

Tabla 6. Resumen de los resultados obtenido al aplicar la Métrica de la Calidad de Especificación de Requisitos.

NF = 45 NNF = 34 NR = NF + NNF = 45+ 34 = 79

Q1 = Nu1 / NR = 75/ 79 = 0.949

El valor óptimo de Q1 es 1, que significa la ausencia de ambigüedad en los requisitos, en este caso se obtuvo Q1 con un valor de 0.949, éste resultado demuestra que el grado de ambigüedad en la especificación de los requisitos de software del subsistema Administración y Gobierno fue muy bajo y por consiguiente hubo calidad en la especificación. Los requisitos ambiguos fueron sometidos a un nuevo proceso de análisis, donde se determinó que fueran renombrados logrando de esta manera una mayor claridad en su interpretación, sin embrago no se realizó una segunda iteración de la métrica, pues se consideró haber alcanzado un valor de especificidad suficientemente alto.

3.2.2 Modelo de Métricas Orientadas a Objeto aplicada al DCUS.

Para medir la calidad del Diagrama de Casos de uso del Sistema (DCUS) se le aplica al mismo un modelo de Métricas, donde se tienen en cuenta cuatro atributos: Completitud, Consistencia, Correctitud, Complejidad, los cuales cuentan con un conjunto de Factores. Cada uno de estos Factores tiene asociada una o más Métricas, que establecen una medida cuantitativa del grado en que los Factores indiquen una mala calidad. (EAFIT, 2007).

- ✓ Completitud: Grado en que se ha logrado detallar todos los casos de uso relevantes.
- ✓ Consistencia: Grado en que los casos de uso del sistema describen las interacciones adecuadas entre el usuario y el sistema.
- ✓ Correctitud**:** Grado en que las interacciones actor / sistema soportan adecuadamente el proceso del negocio.
- ✓ Complejidad**:** Grado de claridad en la presentación de los elementos que describen el contexto y la claridad del sistema.

Los resultados obtenidos luego de dos iteraciones, se muestran en la siguiente tabla:

		Evaluación para el	Evaluación para el

Factores de Completitud	Métricas Asociadas	Subsistema A&G (Primera iteración)	Subsistema A&G (Segunda iteración)
Factor 1. ¿Han sido definidos todos los roles relevantes de usuario encargados de generar/ modificar o consultar información?	Métrica 1: Número de roles relevantes omitidos. Umbral: < 10% Acciones sugerida: Revisar el alcance del sistema e Involucrar tipos de usuarios representativos de cada una de las áreas funcionales.	Total de roles relevantes: 4 Número de roles relevantes omitidos: 0 Representa: 0%	Total de roles relevantes: 4 Número de roles relevantes omitidos: 0 Representa: 0%
Factor 2. ¿Están definidos todos los requisitos que justifican la funcionalidad del caso de uso?	Métrica 2: Número de requisitos omitidos por caso de uso. Umbral < 10%	Total de requisitos: 45 Número de requisitos omitidos por caso de uso: 0 Representa: 0%	Total de requisitos: 45 Número de requisitos omitidos por caso de uso: 0 Representa: 0%
	Métrica 3: Número de casos de uso que tienen requisitos omitidos. Umbral < 10% Acción sugerida: Revisar la lista de	Total de casos de uso: 19 Número de casos de uso que tienen requisitos omitidos : 0 Representa: 0%	Total de casos de uso: 19 Número de casos de uso que tienen requisitos omitidos : 0 Representa: 0%

	requisitos para determinar cuáles serán apoyados por cada caso de uso.		
Factor 3. ¿Se describen las condiciones de excepción relevantes que debe contemplar cada flujo de eventos?	Métrica 4: Número de casos de uso que no describen condiciones de excepción relevante. Umbral < 20% Acción sugerida: Revisar las excepciones presentadas en el flujo de eventos que producen un mensaje de error al usuario	Total de casos de uso: 19 Número de casos de uso que no describen condiciones de excepción relevantes: 2 Representa: 10.52	Total de casos de uso: 19 Número de casos de uso que no describen condiciones de excepción relevantes: 0 Representa: 0%
Factores de Consistencia	Métricas Asociadas	Evaluación para el Subsistema A&G (Primera iteración)	Evaluación para el Subsistema A&G (Segunda iteración)
Factor 4. ¿El nombre dado a los	Métrica 5: Número de casos de uso que	Total de casos de uso: 19	Total de casos de uso: 19

casos de uso es una expresión verbal que describe alguna funcionalidad relevante en el contexto del usuario?	tienen un nombre incorrecto Umbral < 20% Acción sugerida: Modifique el nombre del caso de uso de tal manera que signifique una acción desde el punto de vista del usuario	Número de casos de uso que tienen un nombre incorrecto: 1 Representa: 5.26%	Número de casos de uso que tienen un nombre incorrecto: 0 Representa: 0%
Factor 5. ¿Representa el caso de uso una interacción observable por un actor?	Métrica 6: Número de casos de uso que no representan una interacción observable por un actor Umbral< 5% Acción sugerida: Elimine el caso de uso e incorpore su funcionalidad como una responsabilidad del sistema dentro de otro caso de uso	Total de casos de uso: 19 Número de casos de uso que no representan una interacción observable por un actor: 0 Representa: 0%	Total de casos de uso: 19 Número de casos de uso que no representan una interacción observable por un actor: 0 Representa: 0%
Factor 6. ¿La descripción del flujo de eventos se inicia	Métrica 7: Número de casos de uso cuya descripción	Total de casos de uso: 19 Número de casos de	Total de casos de uso: 19 Número de casos de

con la descripción de una acción externa originada por un actor o por una condición interna del sistema claramente identificable?	extendida no inicia con una acción externa o con una condición monitoreada por el sistema Umbral: < 10% Acción sugerida: Complete la definición del caso de uso incluyendo la acción fuera del sistema que da inicio al caso de uso o la condición interna que el sistema tiene controlar para dar inicio al caso de uso	uso cuya descripción extendida no inicia con una acción externa o con una condición monitoreada por el sistema: 0 Representa: 0%	uso cuya descripción extendida no inicia con una acción externa o con una condición monitoreada por el sistema: 0 Representa: 0%
Factor 7. Si en el caso de uso interviene más de un actor, ¿existe claridad en cuál de ellos es el actor iniciador?	Métrica 8: Número de casos de uso con más de un actor, que no describe cuál es el actor iniciador Umbral: < 20% Acción sugerida: Revise los puntos de inicio del caso de uso y asigne el actor	Total de casos de uso: 19 Número de casos de uso con más de un actor, que no describe cuál es el actor iniciador: 0 Representa: 0%	Total de casos de uso: 19 Número de casos de uso con más de un actor, que no describe cuál es el actor iniciador: 0 Representa: 0%

	que inicia la acción		
Factores de Correctitud	Métricas Asociadas	Evaluación para el Subsistema A&G (Primera iteración)	Evaluación para el Subsistema A&G (Segunda iteración)
Factor 8. ¿Representa el caso de uso requisitos comprensibles por el usuario?	Métrica 9: Grado en que los requisitos representados por el caso de uso son comprensibles por el usuario.	Total de requisitos: 45 Cantidad de requisitos que no son comprensibles por el usuario: 0 Representa: 0%	Total de requisitos: 45 Cantidad de requisitos que no son comprensibles por el usuario: 0 Representa: 0%
	Métrica 10: Número de casos de uso en que los requisitos representados no son comprensibles por el usuario Umbral: < 5% Acción sugerida: Discuta con el usuario la interacción que describe el caso de uso y ajuste dicha descripción de manera que sea comprensible por el usuario.	Total de casos de uso: 19 Número de casos de uso en que los requisitos representados no son comprensibles por el usuario: 0 Representa: 0%	Total de casos de uso: 19 Número de casos de uso en que los requisitos representados no son comprensibles por el usuario: 0 Representa: 0%

Factores de Complejidad	Métricas Asociadas	Evaluación para el Subsistema A&G (Primera iteración)	Evaluación para el Subsistema A&G (Segunda iteración)
Factor 9. ¿Los elementos dentro del diagrama están adecuadamente ubicados de manera que facilitan su interpretación?	Métrica 11: Número de elementos del diagrama que requieren reubicación. Umbral: < 30% Acción sugerida: Modifique la ubicación de los elementos del diagrama de manera que los elementos relacionados se encuentren lo más cercano posible.	Total de elementos: 23 Número de elementos del diagrama que requieren reubicación: 0 Representa: 0%	Total de casos de uso: 23 Número de elementos del diagrama que requieren reubicación: 0 Representa: 0%

Tabla 7. Factores por Atributos.

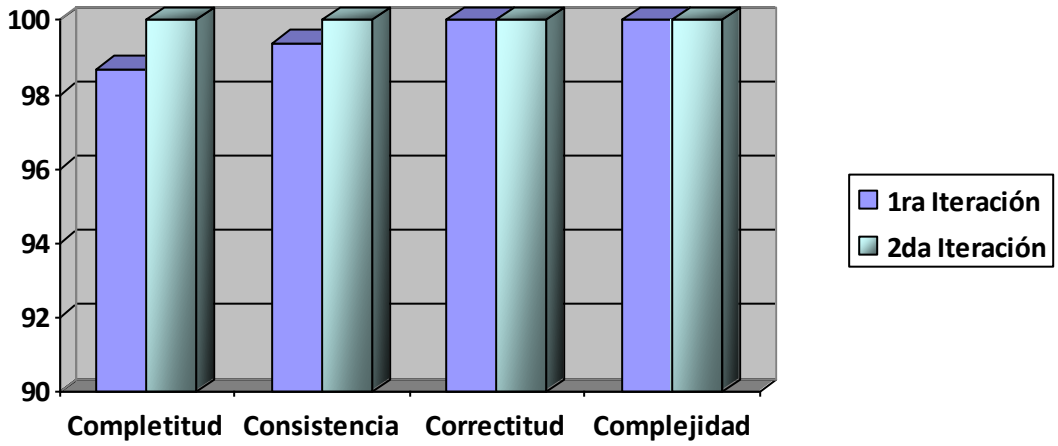


Fig.12 Grado de Funcionalidad del DCUS.

En la primera evaluación realizada, el DCUS alcanzó una calificación de 99.36% de funcionalidad, la contribución de cada atributo a la calidad total fue: Completitud 98.7%, Consistencia 99.4%, Correctitud 100%, Complejidad 100%. Mientras que en la segunda, el DCUS alcanzó una calificación de 100% de funcionalidad, la contribución de cada atributo a la calidad total fue: Completitud 100%, Consistencia 100%, Correctitud 100%, Complejidad 100%.

Luego de la aplicación del modelo de Métricas Orientada a Objeto al DCUS en una segunda iteración, se demostró que el mismo posee la calidad requerida para el futuro diseño del sistema debido a que: cumple con todos los requisitos identificados a través de los casos de uso, los cuales presentan una descripción detallada con todas las acciones del flujo de eventos. En las descripciones de los casos de uso complejos están bien definidas las acciones que corresponden al flujo básico de eventos, a los flujos alternos y a los flujos subordinados. Para lograr una mejor comprensión del DCUS, todos los casos de uso se nombraron con una expresión verbal que describe una funcionalidad relevante para el usuario, representando así una interacción observable para un actor del sistema. Los elementos dentro del diagrama están ubicados lo que facilita su interpretación.

3.3 Medición del grado de satisfacción del Cliente.

En este epígrafe se presentan los resultados obtenidos una vez aplicado el método de Kano. Para ello se muestra primeramente los requisitos analizados. Luego se muestra el mapa de respuestas obtenidas en cada uno de los quince requisitos evaluados. Por último, se lleva a cabo la clasificación de los requisitos según los resultados obtenidos.

Selección de los Requisitos y Encuestas.

Los requisitos seleccionados fueron obtenidos a partir de los requisitos que aparecen en el documento de Especificación de Requisitos. Para la selección de la muestra se tuvo en cuenta que fueran requisitos que permitieran poder realizar acciones importantes en el tribunal, además se tuvo presente que no fueran demasiados para no agobiar al cliente en el llenado de la encuesta. En la siguiente tabla se muestra el listado de requisitos.

No. Requisitos
1. Crear Tribunal
2. Transferir asuntos
3. Crear sala o sección
4. Dividir sala o sección
5. Restaurar contraseña de un usuario.
6. Crear usuario
7. Asignar roles a usuario
8. Autenticar usuario
9. Adicionar rol
10. Visualizar acciones de los usuarios
11. Asignar horas hábiles
12. Adicionar regla

13. Ver calendario
14. Adicionar señalamiento
15. Asociar fecha a Juez Lego

Tabla 8. Requisitos a evaluar utilizando el método de Kano.

La encuesta fue realizada vía correo electrónico. Para cada requisito se mostraban dos preguntas. Una medía la funcionalidad del requisito y la otra la disfuncionalidad. Para cada pregunta el encuestado tenía la posibilidad de seleccionar una de 5 respuestas. Luego de estas preguntas debía seleccionar el grado de importancia que le atribuía al requisito mostrado, los valores de importancia fueron:

- ✓ Para nada importante
- ✓ Algo importante
- ✓ Importante
- ✓ Muy importante
- ✓ Extremadamente importante.

Representación de las Respuestas.

En esta sección se muestra un análisis detallado del mapa de las respuestas, las cuales son agrupadas en una tabla de concentración, que corresponde a cada una de las preguntas del cuestionario. El objetivo fundamental de esta tabla es observar la dispersión de las respuestas. En el anexo se muestran los resultados obtenidos en las encuestas para cada uno de los 15 requisitos evaluados.

Para la realización de la tabla que agrupan los datos de clasificación de cada uno de los requisitos según las respuestas de los encuestados, se utilizó la propuesta hecha por Leon Duarte en (Leon, 2005). La interpretación de la clasificación de los requisitos se basa en el incremento de la satisfacción (columna “Mejor”), o bien, en el decremento de ella (columna “Peor”), debido a una inclusión o no de una característica o requisito del producto.

Otro de los valores reflejados en la tabla 9 es la importancia promedio que le atribuyen los encuestados a cada uno de los requisitos (columna “Imp”). Las columnas C1 y C2 se obtienen

de la multiplicación de los valores Mejor y Peor por la importancia promedio respectivamente. Estos valores se grafican en una escala bidimensional de Mejor y Peor donde se puede identificar la clasificación del tipo de requisito (ver Figura 13)

Req.	A	O	U	I	Inv	D	Imp.	Mejor	Peor	C1	C2
1	2	4	1	1	1	1	0,90	0,37	0,62	0,33	0,56
2	3	1	3	0	2	1	0,80	0,85	0,57	0,68	0,45
3	1	3	2	1	2	1	0,85	0,43	0,71	0,37	0,60
4	3	1	3	0	2	1	0,80	0,85	0,57	0,68	0,45
5	3	3	1	3	0	0	0,63	0,40	0,40	0,25	0,25
6	1	5	3	0	1	0	0,85	0,44	0,88	0,37	0,75
7	0	3	5	1	1	0	0,88	0,55	0,88	0,48	0,77
8	1	1	4	1	2	1	0,95	0,71	0,71	0,68	0,68
9	3	3	1	3	0	0	0,87	0,40	0,40	0,35	0,35
10	4	0	2	1	3	0	0,89	0,86	0,29	0,76	0,25
11	3	2	2	0	2	1	0,85	0,71	0,57	0,61	0,49
12	2	1	3	1	1	2	0,98	0,71	0,57	0,70	0,56
13	4	1	2	1	1	1	0,99	0,75	0,38	0,74	0,37
14	1	0	4	0	2	3	0,81	1,00	0,8	0,81	0,65
15	1	1	4	1	2	1	0,90	0,71	0,71	0,64	0,64

Tabla 9. Clasificación de los requisitos.

Clasificación de los Requisitos.

Una vez analizados los datos recogidos en las encuestas se clasificaron los requisitos, donde resultaron ser: atractivos cinco, unidimensionales cuatro, obligatorios cuatro y dos indiferentes. La distribución de los requisitos se puede ver en la Figura 13, al igual que la Tabla 10 donde se listan los requisitos con su respectiva clasificación.

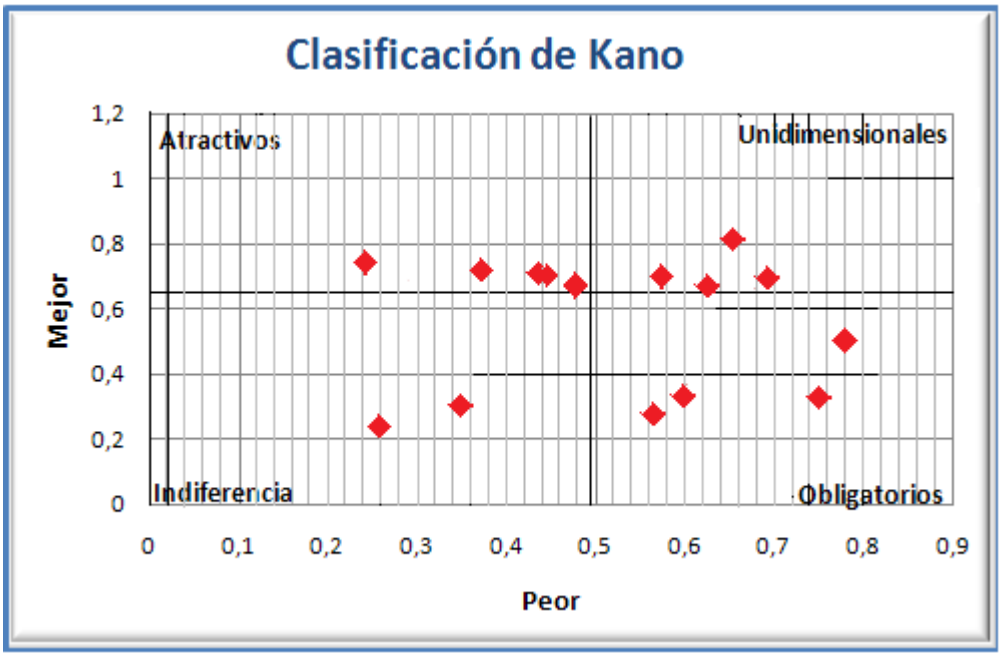


Fig.13 Clasificación de los requisitos en función de los resultados obtenidos.

No. Requisito	Clasificación
1. Crear Tribunal	Obligatorio
2. Transferir asuntos	Atractivo
3. Crear sala o sección	Obligatorio
4. Dividir sala o sección	Atractivo
5. Restaurar contraseña de un usuario.	Indiferencia
6. Crear usuario	Obligatorio
7. Asignar roles a usuario	Obligatorio

8. Autenticar usuario	Unidimensional
9. Adicionar rol	Indiferencia
10. Visualizar acciones de los usuarios	Atractivo
11. Asignar horas hábiles	Atractivo
12. Adicionar regla	Unidimensional
13. Ver calendario	Atractivo
14. Adicionar señalamiento	Unidimensional
15. Asociar fecha a Juez Lego	Unidimensional

Tabla 10. Clasificación de los requisitos en función de los resultados obtenidos.

Los resultados obtenidos evidencian que la satisfacción del cliente es elevada. Siendo de gran importancia, que la mayor cantidad de los requisitos clasificados estén en el rango de los atractivos, debido a que estos son los que más satisfacción causan en los clientes, además de los unidimensionales y obligatorios.

Conclusiones.

La aplicación de las métricas para la calidad de la especificación de los requisitos de software demostró que el grado de ambigüedad de los mismos fue muy bajo. Mientras que la aplicación de las métricas para la calidad de la funcionalidad del DCUS reflejó que se construyó un diagrama con calidad. Por su parte la aplicación del método de Kano también arrojó resultados satisfactorios en la mayoría de los requisitos, asegurándose de esta manera la satisfacción del cliente. A partir de todo lo anterior se puede plantear que los artefactos obtenidos tienen la calidad necesaria, lo que contribuirá al posterior diseño del Subsistema Administración y Gobierno.

CONCLUSIONES GENERALES.

Una vez finalizado el desarrollo del presente trabajo se pudo arribar a las siguientes conclusiones:

- ✓ El estudio realizado sobre metodologías de desarrollo de software, lenguajes de modelado y herramientas CASE, posibilitó justificar la selección de las utilizadas en este trabajo, para modelar el Subsistema Administración y Gobierno de la solución informática SIT, teniendo presente las necesidades de los clientes y usuarios finales.
- ✓ El Modelado de Dominio permitió conocer a fondo los conceptos que se manejan en el Subsistema Administración y Gobierno que forma parte del proyecto Tribunales Populares Cubanos.
- ✓ Los Requisitos identificados tuvieron un alto nivel de aceptación, quedando documentados y asegurando una comprensión común por parte de los clientes y desarrolladores.
- ✓ La validación de la Especificación de Requisitos y del DCUS, así como los resultados obtenidos una vez aplicado el Método de Kano, demostró que estos artefactos presentan la calidad requerida para dar continuidad al Subsistema Administración y Gobierno de la solución informática SIT.

RECOMENDACIONES.

A lo largo de este trabajo, se pudo apreciar cómo se cumplió con cada uno de los objetivos trazados en el mismo, no obstante, se realizan varias recomendaciones a aquellos que le darán continuación, entre las que se encuentran las siguientes:

- ✓ Dar continuidad a los flujos de trabajo propuestos por RUP logrando de esta forma una solución informática que sea capaz de resolver los problemas reales para los cuales fue creada.
- ✓ Dar continuidad a la gestión de requisitos para tener control sobre los riesgos futuros que puedan atentar contra el buen desempeño del software.

BIBLIOGRAFÍA.

1. **Amoroso Fernandez, Yarina. 2002.** *Sociedad de la Información: Contribución de la Informática Jurídica, Revista de Derecho Informático.***.** s.l. : Editorial Alfa-Redi, 2002.
2. **Adriano. 2006.** 2006.
3. **Adriano, Natalia. 2006.** *Comparación del Proceso de Elicitación de Requerimientos en el desarrollo de Software a Medida y Empaquetado. Propuesta de métricas para la elicitación.* 2006.
4. **AITECO. 2008.** [En línea] 2008. http://www.aiteco.com/web/.
5. **Axure. 2007.** [En línea] 2007. http://axure-rp.softonic.com.
6. **Campos, Saúl González y Martínez, Luis Felipe Fernández. 2006.** *Programación Extrema: Prácticas, Aceptación y Controversia.* México : s.n., 2006.
7. **Canós, José H., Letelier, Patricio y Penadés, Mª Carmen. 2003.** *Métodologías Ágiles en el Desarrollo de Software.* Valencia : s.n., 2003.
8. **Davila, Nicolas. 2001.** *Ingeniería de Requerimientos una guía para extraer, analizar, especificar y validar los requerimientos de un proyecto.* 2001.
9. **2007.** DoRol. [En línea] 2007. http://dorol.wordpress.com.
10. **Duarte, Jaime Alfonso Leon. 2005.** 2005.
11. **Durán Toro, Amador y Bernárdez Jiménez, Beatriz. 2000.** *Metodología para la Elicitación de Requisitos de Sistemas Software.* Sevilla : s.n., 2000.
12. **Durán, Amador. 2000.** *Un entorno metodológico de Ingeniería de Requisitos para Sistemas de Información.* 2000.
13. **Durocher, Eric. 2007.** Business process Managements Notation. 2007.
14. **2007.** EAFIT. [En línea] 2007. [Citado el: 13 de marzo de 2011.] http://dis.eafit.edu.co/~ranaya/marcoref/metricas/Artefacto_CasoUsoEscenario.htm.
15. **2007.** EAFIT. *Modelo de Métricas Orientado a Objetos.* [En línea] 2007. [Citado el: 20 de febrero de 2011.] http://dis.eafit.edu.co/~ranaya/marcoref/metricas/Artefacto_CasoUsoEscenario.htm..
16. **Edith, Maylen y Ortiz. 2008.** 2008.

17. **Escalona, María José y Koch, Nora. 2002.** *Ingeniería de Requisitos en Aplicaciones Web.* Sevilla : s.n., 2002.

18. **Fuentes, Lidia y Antonio, Valecillo. 2003.** Una Introducción a los Perfiles UML. España : s.n., 2003.

19. **FUNDIBEQ. 2008.** [En línea] 2008. www.fundibeq.org.

20. **Garzón, Darwin Jiménez.** *Ingeniería de Requerimientos.*

21. **1996.** Gedex. *Gedex.* [En línea] 1996. www.gedex.net.

22. **Goguen, Joseph A. y Linde, Chalotte. 1993.** *Techniques for Requirements Elicitation.* s.l. : IEEE Computer Society, 1993.

23. **Hilera, Jose R. 2003.** [En línea] 2003. http://www.cc.uah.es/hilera/docs/1999/c_jisbd/c_jisbd.htm.

24. **IEEE Standard Association. 2004.** IEEE Standard Glossary of Software Engineering Terminology12-1990. *IEEE Standard Association.* [En línea] 2004. http://standards.ieee.org/reading/ieee/std_public/description/se/610.12-1990_desc.html.

25. **IEEE Swebok. 2005.** *Guide to the Software Engineering Body of Knowledge.* 2005.

26. **1998.** Infolex, Gestión Jurídica. *Infolex, Gestión Jurídica.* [En línea] 1998. www.infolex.com.mx.

27. **ISO9126. 2007.** [En línea] 2007.

28. **Jacobson, Ivar, Booch, Grady y Rumbaugh, James. 2000.** *El proceso Unificado de desarollo de software.* Madrid : s.n., 2000.

29. **Kendall, Kenneth E. 1997.** *Análisis y diseño de sistemas. .* México: s.n., 1997.

30. **2006.** Knowledge Based System. [En línea] 2006. http://www.idef.com/IDEF0.html..

31. **Leon. 2005.** [En línea] 2005.

32. **Mendoza Sánchez, María A. Junio 2004.** *Metodologías De Desarrollo De Software. .* Junio 2004.

33. **Molpeceres. 2002.** [En línea] 2002.

34. **Övergaard, Gunnar y Palmkvist, Karin. 2004.** *Use Cases Patterns and Blueprints.* s.l. : Pearson Education, 2004.

35. **Parusaraman. 1988.** 1988.

36. **Pérez, J.D., Durán, A. y Ruiz, A. 2007.** *¿Por qué OMG ha elegido BPMN para modelar procesos de negocio si ya existe UML? .* España: s.n., 2007.

37. **Pérez, M. 1999.** Arquitectura para Ambientes CASE Integrados. Tesis Doctoral. UCV. 1999.

38. **Pressman, R. 2005.** *Ingeniería del Software. Un enfoque práctico.* La Habana : Félix Varela, 2005. pág. 601.

39. **Pressman, Roger S. 2005.** *Ingeniería de Software, un enfoque práctico. .* 2005.

40. **Prieto, Ortelio Juiz. 2011.** Globedia. *Cuba comenzará en 2011 la digitalización del sistema judicial. .* [En línea] 2011. http://cu.globedia.com/cuba-comenzara-2011-digitalizacion-sistema-judicial.

41. **Pyme Actual. 2006.** Calidad del desarrollo Web. *Pyme Actual.* [En línea] 2006. http://www.pymeactual.com/desarrollo-web/calidad-desarrollo-web.php.

42. **Quintero, Juan Bernardo, y otros. 2005.** *Un estudio comparativo de herramientas para el modelado con UML.* Colombia : s.n., 2005.

43. **Raghavan, S, Zelesnik y Ford, G. 1994.** *Lectures Notes of Requirements Elicitacitation. .* s.l. : Educational Materials, 1994.

44. **Sanchez, y otros. 2004.** [En línea] 2004.

45. **Sommerville. 2005.** *Ingeniería de Software.* s.l. : Editorial Pearson, 2005.

46. **Sommerville, I. 2005.** *Ingeniería del Software.* México DF. Editora Pearson, 4(7), 114-156 : s.n., 2005.

47. **Curso 2007-2008.** teleformación. *Conferencia 3: Flujo de trabajo de requerimientos.* [En línea] Curso 2007-2008. [Citado el: 13 de abril de 2011.] http://teleformacion.uci.cu/mod/resource/view.php?id=8865..

48. **Torres, José Luis. 2008.** *Especificación de Requisitos en Ingeniería de Software.* 2008.

49. Tribunal Supremo Popular. [En línea] www.tcp.cu.

50. *V Encuentro Internacional de Justicial y Derecho.* **Tribunal Supremo Popular y Equipo de desarrollo TPC UCI. . 2010.** 2010.

51. *Visual Paradigm.* **2008.** 2008.