# *Universidad de las Ciencias Informáticas*
# *Facultad 3*

***Análisis y Diseño del Módulo Contratación del Proyecto Convenio Integral de Cooperación Cuba-Venezuela (CCV).***

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

**Autor:** Hussein Gil Gomez.

**Tutores:** Ing. Yaniet Piñeiro Pérez
Ing. Yudier Cervantes Puga

## DECLARACIÓN DE AUTORÍA

Declaro ser el único autor de este trabajo y autorizo a la Universidad de las Ciencias Informáticas hacer uso del mismo en su beneficio. Para que así conste firmo la presente a los ____ días del mes de _____ del año _____.

| **Hussein Gil Gomez** | **Yaniet Piñeiro Pérez** | **Yudier Cervantes Puga** |
|---|---|---|
| _____ | _____ | _____ |
| Firma del Autor | Firma del Tutor | Firma del Tutor |

# Agradecimientos

*Agradezco en primer lugar, a mis tutores, en especial a la Ing. Yaniet, por todo su apoyo y comunicación durante el proceso de desarrollo de este trabajo, por su preocupación y confianza en mí…*

*A la Revolución, por darme la oportunidad de convertirme en un profesional…*

*A mi familia, por su contribución en la formación de mi personalidad, por su solidaridad y apoyo…*

*A la gente que me ha soportado durante estos 5 cursos aquí en la Universidad de la Excelencia, como la llamo nuestro comandante Fidel, a toda esa gente bonita que ha colaborado de una forma u otra en la culminación del presente proyecto…*

*A todos ustedes gracias…*

*Muchas Gracias…*

*Hussein…*

# Dedicatoria

*Ante todo a mis padres que me dieron la posibilidad de existir…*

*A mi Padre,* ***Marcelino Gil Jaime,*** *que se que este era uno de sus sueños, donde quiera que se encuentre, este triunfo es de él también, que le estaré en deuda eternamente y que siempre está conmigo…*

*A mi madre,* ***Isabel Gomez Caballero****, que lo ha sido todo para mí, que soy todo lo que soy gracias a ella, que no tengo palabras, ni hechos, ni pensamientos para expresar mi gratitud hacia ella. A ti, que ha sido lo mejor que me ha pasado en la vida, que me formo como hombre y como persona, y que sin sus consejos y su apoyo, este momento no existiría…*

*A mi padrastro, que ha sido como un padre, que hoy veo con claridad sus consejos y le debo mucho de lo que he logrado en la vida…*

*A mi novia, que está ahí en cada momento, que me regalo el amor que todos esperamos en la vida, que se ha convertido en mi reina, te amo BB…*

*A la gente de mi zona, a ese campo querido de Cuba, a todas esas personas que me criaron como a un hijo, donde he aprendido a valorar la amistad y la solidaridad…*

*A toda esa gente maravillosa que ha entrado y salido de mi vida en todas sus etapas, a mis incomparables compañeros del IPVCE, que no sé como agradecerles lo que hicieron por mi cuando más lo necesite, a todos ellos donde quiera que estén…*

*A mis grandes amigos, que no tengo necesidad de señalar nombres, pero que están ahí y que siempre estarán. A todos ellos…*

*Y por último, a* ***Elena Caballero Pino****, la abuela más comprensible y hermosa del mundo, vieja tu eres lo más grande que me ha dado la vida, de una forma muy especial te dedico a ti este momento, te quiero como nunca he sabido demostrarlo…*

*No vayas por donde te lleve el camino,*

*Ve por donde no haya camino y deja tu sendero.*

*Ralph Waldo Emerson*

*Resumen*

El incremento continúo de los proyectos entre las naciones de Cuba y Venezuela hace que surja la necesidad de desarrollar un sistema que permita gestionar los trámites y actividades que se llevan a cabo durante la ejecución de los procesos internos que intervienen. Lo cual se realiza desde que surge la idea de crear un nuevo proyecto, hasta su aprobación y contratación, garantizando el control estricto de cada uno de ellos. Es por lo antes mencionado que surge el proyecto CCV, el cual cuenta con 7 módulos, y dentro de ellos se encuentra el de “Contratación”, en el cual se centra el presente trabajo. El cual tiene como principal objetivo elaborar el contrato de aquellos proyectos que han sido financiados. Para el desarrollo del mismo se ha hecho necesario el estudio y análisis de los procesos que se ejecutan, además de las metodologías, lenguajes, herramientas y patrones a utilizar, así como un estudio del estado del arte de los sistemas de gestión. Finalmente se evaluaron los resultados obtenidos del análisis y diseño con la utilización de métricas dirigidas a garantizar la calidad de los artefactos.

***Palabras Claves***

[**Análisis, Diseño, Artefactos**]

## Índice

## *Introducción*

En la actualidad, las Nuevas Tecnologías Informáticas son muy usadas a nivel mundial para el desarrollo de software con el fin de resolver una problemática dada. Estas tecnologías, hoy en día se han convertido en un poderoso elemento influyente en la transformación de la sociedad. Criticada por unos e idealizada por otros, la tecnología por su naturaleza innovadora y dinámica, a menudo resulta incomprendida o subvalorada en su potencial. Es necesario comprender que la tecnología es el resultado de la efímera intelectualidad del hombre, que bien utilizada, encaminará y transformará positivamente nuestra sociedad.

Durante el proceso de desarrollo de un software se cometen gran cantidad de errores a nivel mundial, ya sea no contar con un personal adecuado para el desarrollo del producto, presentar una gestión de riesgo insuficiente, así como la poca utilización de patrones, en el momento de confeccionar el análisis y diseño de un sistema pueden ocasionar pérdidas de tiempo y dinero.

Se sabe que entre los factores claves en el fallo de la ejecución de proyectos está:

- Requisitos incompletos (13%).
- Falta de implicación del usuario (12.4%).
- Falta de recursos (10.6%).
- Expectativas no realistas (9.9%).
- Falta de soporte (9.3%).
- Cambios en requisitos (8.7%).
- Falta de planificación (8.1%).
- El sistema ya no era necesario (7.5%).

El 31% de los proyectos de software a nivel mundial, se cancelan antes de ser terminados. Solo en grandes compañías, el 9% de los proyectos, se entregan en tiempo y forma.

Existen otras muchas causas que reflejan la mala calidad en el proceso de construcción en la industria del software, según el libro:”Software Development’s Classic Mistakes 2008”, existen disímiles errores que los analistas cometen en el momento de confeccionar la estructura básica de un software.

La República de Cuba y la República Bolivariana de Venezuela con el deseo de mejorar sus respectivas economías y adquirir las ventajas que resulten de una cooperación, que tenga resultados efectivos en el avance económico y social de ambos países, y la integración de América Latina y el Caribe acordaron el cumplimiento del Convenio Integral de Cooperación entre ambas naciones.Los presidentes de Venezuela, Hugo Chávez Frías, y

de Cuba, Fidel Castro Ruz, firmaron el acuerdo de cooperación integral, en el Salón Ayacucho del Palacio de Miraflores, el 30 de octubre de 2000.

El artículo V del Convenio plantea la presentación de una Comisión Mixta integrada por representantes de ambos gobiernos, con el objetivo de poseer un mecanismo para el cumplimiento y seguimiento de las acciones de cooperación previstas dentro del convenio.Ambos Presidentes acordaron elaborar programas y proyectos de cooperación, para la ejecución de los mismos se contará con la participación de organismos públicos y privados de ambos países, universidades y organizaciones no gubernamentales.

Desde entonces se lleva a cabo un amplio margen de cooperación económica y social enmarcado dentro del Convenio Integral de Cooperación, la cual está centralizada en la presentación y aprobación por ambos países de proyectos y contratos.

Actualmente no se posee un control eficiente de este inmenso espectro de proyectos y contratos, ya que hoy día este proceso se desarrolla manualmente, debido a la falta de una herramienta capaz de llevar a cabo esta tarea. Es por esto que se identificaron una serie de dificultades en el seguimiento y control de los mismos.

Las posibles pérdidas económicas que pueden desatarse a partir de la pérdida de información de estos documentos, nos llevan a la proposición de un sistema automatizado capaz de desempeñar esta tediosa, pero necesaria tarea de mantener al día toda la documentación de estos contratos, generados dentro de la cooperación entre Cuba y Venezuela.

La presentación de estas propuestas de proyectos a realizarse dentro del marco de las comisiones mixtas entre ambos países debe pasar por un flujo de trabajo con el objetivo de ser aprobados y financiados, para posteriormente realizar su contratación.

Teniendo en cuenta que nuestro principal propósito no es el de producir mecánicamente grandes cantidades de software, sino la de producir software con calidad, y basándonos en un el estudio realizado sobre el estado de la producción de software hoy día, se determinó el siguiente problema científico de la investigación.

**Problema Científico de Investigación:** La no existencia del análisis y diseño del Módulo Contratación en el proyecto Informatización del Convenio Cuba-Venezuela afecta el desarrollo de un software que permita el control de los proyectos existentes entre ambas naciones.

**Objeto de Estudio:** Proceso de desarrollo de software.

**Campo de Acción:** Flujo de Análisis y Diseño.

**Objetivo:** Desarrollar el análisis y diseño del Módulo Contratación del Proyecto Informatización del Convenio Cuba-Venezuela, para la posterior implementación de un software que cumpla con las necesidades del cliente.

Para darle cumplimiento a este objetivo se plantean los siguientes **objetivos específicos**:

- Realizar un estudio sobre el estado del arte de las metodologías, patrones y de las herramientas que pueden ser utilizadas en el proceso de análisis y diseño.

- Describir los artefactos generados a partir del desarrollo del análisis y diseño del sistema.

- Validar los resultados obtenidos.

**Hipótesis:** Si se realiza un correcto análisis y diseño del Módulo Contratación del proyecto Informatización del Convenio Cuba Venezuela, entonces se logrará la implementación de un software que cumpla con las necesidades del cliente.

Se establecen entonces las siguientes **tareas de la investigación**:

- Estudio de sistemas de gestión de proyectos, metodología y herramientas de modelado a utilizar en la solución.
- Documentación de los procesos del negocio.
- Elaboración del diagrama de casos de uso del sistema.
- Descripción de los casos de uso del sistema.
- Elaboración del diagrama de clases del diseño.
- Elaboración del diagrama de clases persistentes.
- Elaboración del modelo de datos.
- Validación de los artefactos obtenidos.

La estrategia de investigación a seguir es la exploratoria, ya que se realizará un estudio acerca de las diferentes metodologías, herramientas para el desarrollo y modelado de software, así como los patrones que pueden ser utilizados para el desarrollo del análisis y el

diseño. Esto con el objetivo de adquirir los conocimientos necesarios para darle la mejor solución posible al problema planteado anteriormente.

Los métodos científicos de investigación a utilizar son los teóricos y los empíricos.

***Dentro de los métodos teóricos se utilizarán los siguientes:***

El **histórico-lógico**, para desarrollar un estudio del estado del arte sobre todo lo relacionado con el análisis y diseño de sistemas a nivel mundial, y así no cometer los errores que comúnmente se desatan.

El **Analítico-Sintético** es el método que me ayudará a determinar cual estrategia se tomara en el desarrollo del producto y cual metodología será la más adecuada para la concepción del sistema.

Para el análisis y modelación del negocio y sistema, así como para el diseño, es necesaria la elaboración de diagramas, figuras y otros artefactos importantes, por lo que se hará uso del método de modelación, pues mediante este se pueden crear abstracciones con el propósito de representar y explicar la realidad.

***Dentro de los métodos empíricos se tendrán en cuenta los siguientes:***

**Observación**, este método nos permite analizar las estrategias y herramientas que se utilizan actualmente en el mundo en el proceso de desarrollo de software y después de un estudio exhaustivo determinar, que ningún flujo de análisis y diseño desarrollado en otros sistemas similares que existen a nivel mundial, poseen las especificidades de nuestro sistema.

**Entrevista**, ya que es una de las técnicas fundamental para la obtención de la información referente a las necesidades de los clientes y así realizar la captura de requisitos del software a implementar. Sin restar importancia a otras técnicas.

A partir del análisis y diseño del sistema se esperan obtener los modelos correspondientes a esta actividad, de forma que permitan un entendimiento tanto para los desarrolladores como para los clientes, en cuanto a las funcionalidades que debe tener el software. Los artefactos generados durante estos procesos constituirán dentro del equipo de desarrollo, la entrada para el desarrollo de un software que cumpla con las necesidades del cliente.

La importancia y la actualidad que posee este trabajo, es que permitirá mejor la relación entre los países de Cuba y Venezuela y además permitirá la implementación del producto en tiempo y además servirá de material de estudio para la confección y desarrollo de otros sistemas similares.

### ***Estructura de la tesis***.

*El trabajo de diploma está compuesto por 3 capítulos.*

**Capítulo 1**. *Fundamentación Teórica:* En este capítulo se desarrolla un estudio de estado del arte de sistemas para la gestión de proyectos así como de las técnicas más usadas para la captura de requisitos. Además se realiza un estudio sobre las herramientas y patrones más utilizados en los flujos de análisis y diseño.

**Capítulo 2**. *Solución Propuesta:* En este capítulo se lleva a cabo la modelación de los procesos del negocio, teniendo en cuenta las reglas del negocio plantadas. Se realiza la especificación de los requisitos funcionales y no funcionales, así como de los actores del sistema. Además se obtiene el diagrama de casos de uso, con la respectiva descripción de cada uno, así como los diagramas de clases del diseño de cada caso de uso, orientados a procesos.

**Capítulo 3**. *Validación de los Resultados:* Se desarrolla una validación de los artefactos obtenidos en capítulos anteriores utilizando para ello, métricas de validación, ya sea para el diagrama de casos de uso, para la descripción de los mismos o para los diagramas de diseño.

# *Capítulo 1. Fundamentación Teórica.*

## *1.1. Sistemas de gestión de proyectos.*

En la actualidad existen varias herramientas que permiten la planificación y seguimiento de proyectos tanto entre empresas como instituciones gubernamentales. Aunque casi todas poseen un alto nivel tecnológico y son reconocidas en toda la comunidad informática, hay que decir que no poseen todas las características necesarias para afrontar esta empresa con la envergadura que posee. Con el objetivo de conocer y estudiar características individuales de cada una de estas herramientas, que permitan una mejor comprensión del sistema a implementar, atribuyendo ideas a las analistas y diseñadores, se desarrolla un estudio de las mismas. Algunas de las herramientas estudiadas, son:

### *1.1.1. Collabtive*

Collabtive es un proyecto alemán de tipo “open source”, para la Administración y Configuración de proyectos y grupos de trabajo a través de la web. Con esta excelente herramienta se puede administrar fácilmente todo tipo de trabajos teniendo varias herramientas para ello como carga de archivos, mensajería, listas de tarea, tareas compartidas en entornos colaborativos, gestión y configuración de proyectos y un completo sistema de gestión de grupos de trabajo. La herramienta es multiplataforma.

Esta aplicación web está realizada íntegramente con PHP, utiliza base de datos MySQL, es gratuita, de código abierto [1].

Pero no posee todas las características necesarias para cumplir con las necesidades de nuestro cliente.

### *1.1.2. DotProject*

Es una herramienta creada en el año 2000 con el fin de construir una herramienta para la Gestión de Proyectos, es una Aplicación Web, desarrollada sobre software libre, soporta varios lenguajes y es un sistema multiusuario [2].

Fue creado inicialmente para Linux aunque en la actualidad se pueden encontrar ejecutables para Microsoft Windows y Macintosh esencialmente [3].

DotProject esta creado fundamentalmente para:

- Ser de código abierto, libre acceso y utilización.
- Proveer a los usuarios de funcionalidades orientadas a la Gestión de Proyectos.

- Construir una herramienta con una interfaz de usuario simple, claro y consistente.

Esta herramienta está orientada principalmente a la administración de recursos para desarrollar un proyecto y producto cuya culminación dependan de un conjunto de actividades que se desarrollen de formas paralelas o independientes [3].

### *1.1.3. Microsoft Project*

Creada en el año 1984 y perteneciente a Microsoft Office, esta herramienta es desarrollada y vendida por la corporación de Microsoft. Microsoft Project (o MSP) es un software de administración de proyectos diseñado, desarrollado y comercializado para asistir a administradores de proyectos en el desarrollo de planes, asignación de recursos a tareas, dar seguimiento al progreso, administrar presupuesto y analizar cargas de trabajo.

Aunque es básicamente una herramienta para Windows, se puede decir que posee versiones para Macintosh. Como parte de la producción de la gran transnacional Microsoft, es una herramienta privativa. Aunque este software ha sido etiquetado como miembro de la familia Microsoft Office hasta el momento no ha sido incluido en ninguna de las ediciones de Office y está disponible en dos versiones, Standard y Professional [5].

La aplicación crea calendarización de rutas críticas, además de cadenas críticas. Las gráficas visualizadas en una Gráfica de Gantt. Adicionalmente, Project puede reconocer diferentes clases de usuarios, los cuales pueden contar con distintos niveles de acceso a proyectos, vistas y otros datos [4].

*Existen otras herramientas para la Gestión de Proyectos menos conocidas como son:*

### *1.1.4. Gantt PV*

Gantt PV es un programa gratuito, de apariencia sencilla y sin grandes complicaciones, para planificación de proyectos, descomposición, representación y seguimiento de tareas sobre diagrama de Gantt [6][7].

Existen versiones para Windows, Macintosh y Linux.

### *1.1.5. Milestone Planner*

**Milestone Planner** es una sencilla herramienta online que nos permite la planificación y seguimiento de la evolución de nuestros propios proyectos. Es una aplicación de escritorio para la gestión y seguimiento de proyectos, con descomposición en tareas y sub-tareas, dependencias, identificación de la ruta crítica y diagramas de Gantt [8].

Inicialmente desarrollada para Linux, dispone de versión (beta) para Windows.

Aunque la mayoría de las herramientas cumplen aquí con alguna de las exigencias que se necesitan cumplir, al culminar este proyecto, es necesario resaltar, que ninguna de las herramientas aquí mencionadas, proporciona un flujo de trabajo con la envergadura necesaria, ni poseen las características suficientes para presentar, planificar y dar seguimiento a los proyectos con la seriedad y la eficiencia requerida.

## *1.2. Metodologías de desarrollo*

Las metodologías de desarrollo de software son un conjunto de procedimientos, técnicas y ayudas a la documentación para el desarrollo de productos software. Dentro de las clasificaciones existentes para las mismas se destacan dos: las metodologías tradicionales y las ágiles [9].

Las metodologías tradicionales están basadas en normas provenientes de estándares seguidos por el entorno de desarrollo, son aplicadas generalmente a grandes proyectos e indican paso a paso todas las actividades a realizar para lograr el producto deseado, indicando además qué personas deben participar en el desarrollo de las actividades y qué papel deben de tener [10].

Por otra parte se encuentran las metodologías ágiles que se centran más en la obtención del sistema sin tener en cuenta cuán documentado esté el mismo, generalmente propone que los stakeholder o interesados participen en el proceso de desarrollo para ir mejorando las funcionalidades del sistema en producción, así como proponen un trabajo de desarrollo en parejas.

Dentro de las metodologías tradicionales y que puede ser visto como una metodología ágil además, pues es tan configurable como deseemos, se encuentra el RUP.

### *1.2.1. Rup*

El Proceso Racional Unificado (Rational Unified Process), es un proceso de desarrollo de software y junto con el Lenguaje Unificado de Modelado UML, constituye la metodología estándar más utilizada para el análisis, diseño, implementación y documentación de sistemas orientados a objetos.

*Sus principales características son:*

- Forma disciplinada de asignar tareas y responsabilidades (quién hace qué, cuándo y cómo)
- Pretende implementar las mejores prácticas en Ingeniería de Software

- Desarrollo iterativo
- Administración de requisitos
- Uso de arquitectura basada en componentes
- Control de cambios
- Modelado visual del software
- Verificación de la calidad del software

RUP se caracteriza por ser iterativo e incremental, estar centrado en la arquitectura y guiado por los casos de uso. Incluye artefactos (que son los productos tangibles del proceso como por ejemplo, el modelo de casos de uso, el código fuente, etc.) y roles (papel que desempeña una persona en un determinado momento, una persona puede desempeñar distintos roles a lo largo del proceso de desarrollo de un producto) [11].

RUP divide el proceso de desarrollo en ciclos, teniendo un producto al culminar de cada ciclo, estos se dividen en fases que finalizan con un hito donde se debe tomar una decisión importante:

- **Inicio:** Se describe el negocio y se delimita el proyecto describiendo sus alcances con la identificación de los casos de uso del sistema

- **Elaboración:** Se define la arquitectura del sistema y se obtiene una aplicación ejecutable que responde a los casos de uso que la comprometen

- **Construcción:** Se concentra en la elaboración de un producto totalmente operativo y eficiente y el manual de usuario

- **Transición:** Se implementa el producto en el cliente y se entrena a los usuarios. Como consecuencia de esto suelen surgir nuevos requisitos a ser analizados.

Hace algunos años se reunieron un grupo de impulsores de lo que hoy son las metodologías ágiles para debatir los puntos que tenía en común y como resultado se obtuvo lo que hoy se conoce como Manifiesto Ágil.

Hay diversos métodos ágiles que recogen estas ideas como: *eXtreme Programming (XP), Cristal Methods, SCRUM,* etc. [12]. Pero casi todos plantean algo así:

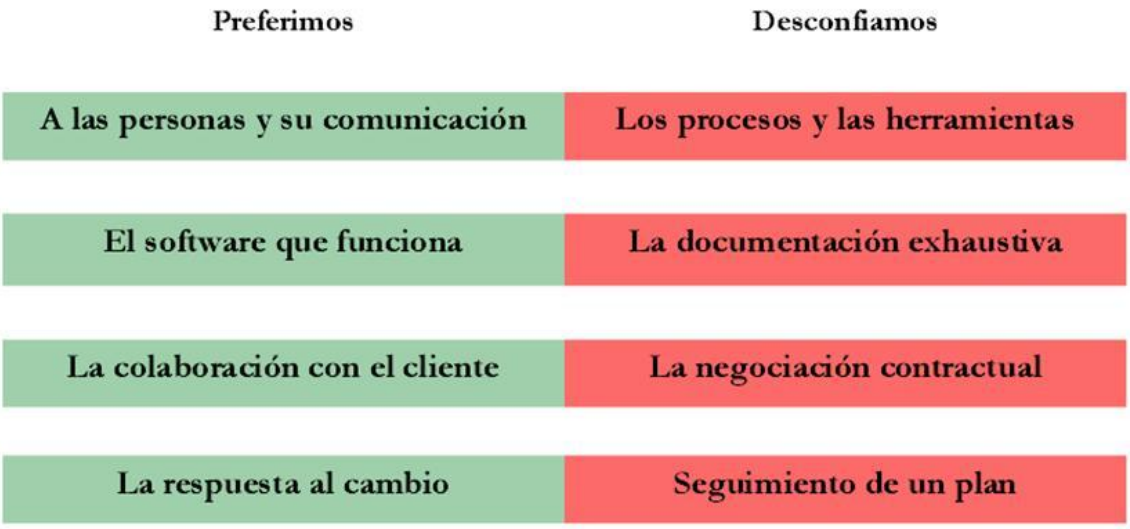



***Figura 1.*** Características principales de las metodologias ágiles.

### *1.2.2. Programación Extrema (XP)*

XP es una metodología ágil centrada en potenciar las relaciones interpersonales como clave para el éxito en desarrollo de software, promoviendo el trabajo en equipo, preocupándose por el aprendizaje de los desarrolladores, y propiciando un buen clima de trabajo. XP se basa en realimentación continua entre el cliente y el equipo de desarrollo, comunicación fluida entre todos los participantes, simplicidad en las soluciones implementadas y coraje para enfrentar los cambios. XP se define como especialmente adecuado para proyectos con requisitos imprecisos y muy cambiantes, y donde existe un alto riesgo técnico [13].

Es la más destacada de los procesos ágiles de desarrollo de software. Al igual que éstos, la programación extrema se diferencia de las metodologías tradicionales principalmente en que pone más énfasis en la adaptabilidad que en la previsibilidad. Los defensores de XP consideran que los cambios de requisitos sobre la marcha son un aspecto natural, inevitable e incluso deseable del desarrollo de proyectos. Creen que ser capaz de adaptarse a los cambios de requisitos en cualquier punto de la vida del proyecto es una aproximación mejor y más realista que intentar definir todos los requisitos al comienzo del proyecto e invertir esfuerzos después en controlar los cambios en los requisitos [13].

***Las características fundamentales de XP son:***

- **Desarrollo iterativo e incremental**: pequeñas mejoras, unas tras otras.
- **Pruebas unitarias** continuas, frecuentemente repetidas y automatizadas, incluyendo pruebas de regresión.

- **Programación por parejas**: se recomienda que las tareas de desarrollo se lleven a cabo por dos personas en un mismo puesto.
- **Frecuente interacción del equipo de programación con el cliente o usuario**. Se recomienda que un representante del cliente trabaje junto al equipo de desarrollo.
- **Corrección de todos los errores antes de añadir nueva funcionalidad**. Hacer entregas frecuentes.
- **Refactorización del código**, es decir, reescribir ciertas partes del código para aumentar su legibilidad pero sin modificar su comportamiento.
- **Propiedad del código compartida**: en vez de dividir la responsabilidad en el desarrollo de cada módulo en grupos de trabajo distintos, este método promueve el que todo el personal pueda corregir y extender cualquier parte del proyecto.
- **Simplicidad en el código**: Cuando todo funcione se podrá añadir funcionalidad si es necesario. La programación extrema apuesta que es más sencillo hacer algo simple y tener un poco de trabajo extra para cambiarlo si se requiere, que realizar algo complicado y quizás nunca utilizarlo.

La simplicidad y la comunicación son extraordinariamente complementarias. Con más comunicación resulta más fácil identificar qué se debe y qué no se debe hacer. Mientras más simple es el sistema, menos tendrá que comunicar sobre este, lo que lleva a una comunicación más completa, especialmente si se puede reducir el equipo de programadores.

### *1.2.3. SCRUM*

Otra de las metodologías muy difundidas dentro de las ligeras o ágiles es Scrum, expuesta por Hirotaka Takeuchi e Ikujiro Nonaka y que define un marco para la gestión de proyectos, que se ha utilizado con éxito durante los últimos 10 años. Scrum es una metodología ágil de administración de proyectos, entre ellos, proyectos de desarrollo de software, si bien se puede aplicar a distintas áreas. Con Scrum los proyectos progresan a través de una serie de iteraciones que duran entre una o dos semanas y un mes, llamadas "Sprints". No es una metodología de análisis, ni de diseño, como podría ser RUP, es una metodología de gestión del trabajo. Esta es, después de XP, la metodología ágil mejor conocida y la que otros métodos ágiles recomiendan como complemento. Además no está concebido como método independiente, sino que se promueve como complemento de otras

metodologías, incluyendo XP o RUP. Como método, enfatiza valores y prácticas de gestión, sin pronunciarse sobre requerimientos, implementación y demás cuestiones técnicas; de allí su deliberada insuficiencia y su complementariedad. Esta metodología se define como un proceso de gestión y control que implementa técnicas de control de procesos. Se le puede considerar como un conjunto de patrones organizacionales. Scrum está especialmente indicada para proyectos con un rápido cambio de requisitos. Sus principales características se pueden resumir en dos. El desarrollo de software se realiza mediante iteraciones, denominadas *sprints*, con una duración de 30 días. El resultado de cada *sprint* es un incremento ejecutable que se muestra al cliente. La segunda característica importante son las reuniones a lo largo proyecto, entre ellas destaca la reunión diaria de 15 minutos del equipo de desarrollo para coordinación e integración [14].

### *Principales Artefactos* [15]

**1) Especificación del Producto (Product Backlog):** Es una lista de requerimientos que se encuentra bajo responsabilidad del dueño del producto. Este mismo es quien debe ir ingresando los requerimientos deseados priorizándolos según el valor de retorno de la inversión que le den a sus representados. Estos requerimientos deben cumplir con el siguiente formato: Como un <Usuario> Yo quisiera <Función> Con lo cual <Valor de Negocio> [14]. La especificación del producto nunca está completa. Es dinámica y existe mientras haya un producto. Si un requerimiento no se termina en una iteración, pasa a la siguiente (o a otra) con mayor o menor prioridad dependiendo de la decisión del dueño del producto.

**2) Especificación del Trabajo (Sprint Backlog ó Commited Backlog):** Acá es donde se definen los requerimientos que el equipo seleccionó para convertir en un incremento de funcionalidad. Las tareas deben demandar entre 4 y 16 horas cada una. Si demandan más de eso se debe analizar dividirlas en sub-tareas. La especificación del trabajo es una foto de lo que el equipo planea tener hecho en esa iteración. Y si bien es creada en la reunión de planeamiento, es actualizada constantemente durante la iteración. En cada iteración el equipo debe construir un incremento de la funcionalidad del producto. Dicho incremento debe estar listo para ser implementado si así lo desea el dueño del producto, al final del Sprint.

**3) Cuadro de Velocidad (Velocity Chart):** Es una herramienta que muestra con que velocidad se fue realizando cada una de las iteraciones. Esto le sirve al equipo como referencia a la hora de determinar el promedio de requerimientos con los que puede comprometerse en una determinada iteración [15].

## *1.3. Fundamentación de la metodología a utilizar.*

Se decidió utilizar RUP, porque es una metodología tradicional con la característica de ser tan ágil como se desee, por su característica de ser configurable. Es la metodología más usada a nivel mundial en materia de software. Posee una gran cantidad de características que la hacen fácil de entender y de usar. Por la condiciones de lejanía del cliente del producto, es necesario llevar una documentación detallada del estado del proyecto, y esta metodología nos permite llevar todo documentado y actualizado. Posee un aspecto muy importante a tener en cuenta, que es su capacidad para el control de cambios. Es, además, la herramienta que se imparte dentro del plan de clases de la asignatura de Ingeniería del Software, dentro de la Universidad.

## *1.4. Lenguajes de Modelado*

Un lenguaje de modelado es un conjunto de estandarizado de símbolos que se utilizan de diferentes modos y formas para modelar el diseño de un software. Generalmente se utiliza en combinación con alguna herramienta de modelado existente actualmente a nivel mundial. A continuación se presentan algunos de los lenguajes más usados a nivel mundial en la actualidad.

### *1.4.1. UML*

UML es un lenguaje usado para especificar, visualizar y documentar los componentes de un sistema en desarrollo *orientado a objetos*. Es el lenguaje de modelado de sistemas de software más conocido y utilizado en la actualidad; aún cuando todavía no es un estándar oficial, está apoyado en gran manera por el *OMG* (Object Management Group). UML ofrece un estándar para describir un plano del sistema (modelo), incluyendo aspectos conceptuales tales como procesos de negocios y funciones del sistema, y como expresiones de lenguajes de programación, esquemas de bases de datos y componentes de software reutilizables [17].

Es importante resaltar que UML es un "lenguaje de modelado" para especificar o para describir métodos o procesos. Se utiliza para definir un sistema, para detallar los artefactos en el sistema y para documentar y construir. En otras palabras, es el lenguaje en el que está descrito el modelo. En UML 2.0 hay 13 tipos diferentes de diagramas. Para comprenderlos de manera concreta, a veces es útil categorizarlos jerárquicamente [11].

- Los ***Diagramas de Estructura*** enfatizan en los elementos que deben existir en el sistema modelado.

- Los ***Diagramas de Comportamiento*** enfatizan en lo que debe suceder en el sistema modelado.

- Los ***Diagramas de Interacción*** son un subtipo de diagramas de comportamiento, que enfatiza sobre el flujo de control y de datos entre los elementos del sistema modelado.

*A continuación se muestra algunas de las principales características de UML.*

- Permite modelar sistemas utilizando técnicas orientadas a objetos (OO).

- Permite especificar todas las decisiones de análisis, diseño e implementación, construyéndose así modelos precisos, no ambiguos y completos.

- Permite documentar todos los artefactos de un proceso de desarrollo (requisitos, arquitectura, pruebas, versiones, etc.).

UML es independiente del proceso, aunque para utilizarlo óptimamente se debería usar en un proceso que fuese dirigido por los casos de uso, centrado en la arquitectura, iterativo e incremental.

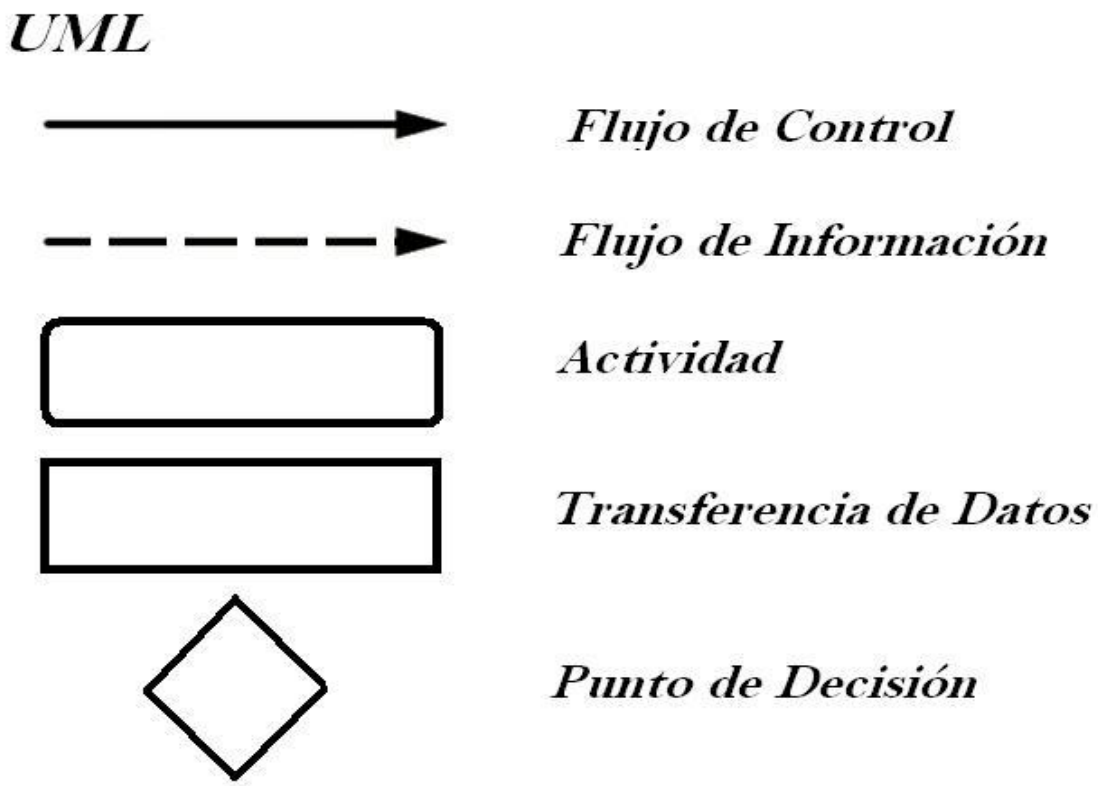



***Figura 2.*** Unidades básicas de UML.

### *1.4.2. Método para la modelación Funcional de Procesos (IDEF)*

IDEF (Métodos Integrados de Definición) es un estándar para el desarrollo estructurado de gráficos representativos de procesos. Su propósito es proveer técnicas de modelado simples y formales que permitan describir, analizar y evaluar distintos puntos de vista de una organización.

IDEF0 permite la modelación funcional de procesos de manera estructurada y jerarquizada, teniendo en cuenta las decisiones, acciones y objetos o datos que soportan la interacción de esas actividades de la organización. Representa lo que se hace en la empresa de un modo *no temporal*. Su principal preocupación es *qué* actividades se llevan a cabo [16].

La representación de un proceso consta de la combinación de cinco unidades básicas que interactúan: Entrada (s), Salida (s), Control (es), Mecanismo (s) (ICOMs por sus siglas en inglés: Input, Output, Control, Mechanism) y la actividad. Las cuatro primeras representadas por flechas y la última mediante una forma rectangular.

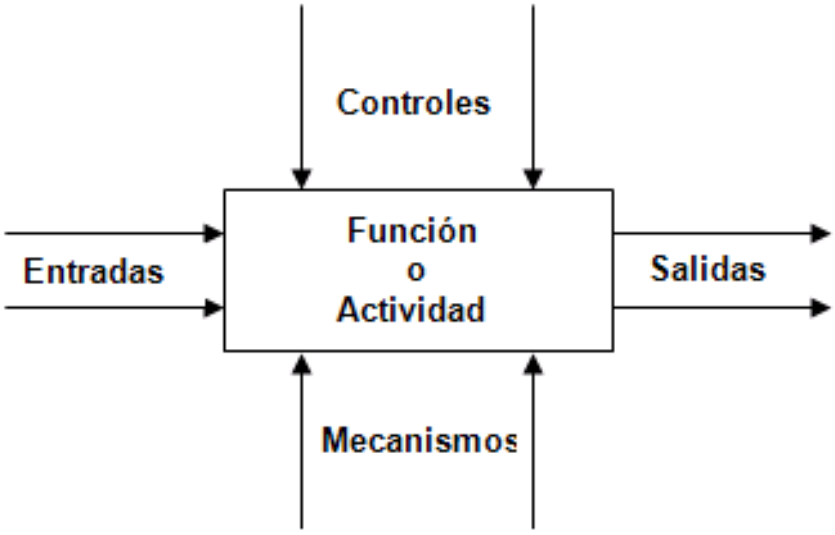



***Figura 3.*** Unidades básicas de IDEF0

Las entradas expresan los datos u objetos que serán transformados por la función en salidas.

Los controles expresan condiciones requeridas para producir las salidas correctas. Los datos u objetos tratados como controles pueden ser transformados por la función creando una salida.

- Las salidas expresan los datos u objetos producidos por la función.
- Los mecanismos expresan los medios utilizados para ejecutar la función.

Existen algunas reglas para estructurar un modelo IDEF0: El primer diagrama del modelo es el diagrama de contexto (diagrama A-0): conformado por una sola actividad, número 0 que representa el objetivo del modelo. Los diagramas de descomposición (A0, A1, A2,…) deben tener de 3 a 8 actividades. Todas las ICOM de la actividad "padre" deben aparecer en las actividades "hijo". Las flechas al igual que las actividades, se pueden dividir en 2 ó más en los diagramas "hijo" [18].

***De manera general IDEF0 tiene las siguientes características:***

- Es una técnica genérica que permite modelar gráficamente procesos de sistemas de diferentes propósitos y a cualquier nivel de detalle.
- Es comprensivo y expresivo, capaz de representar gráficamente reglas y una amplia variedad de negocio.
- Es un lenguaje simple y coherente.
- Realza la comunicación entre los analistas de sistemas, los desarrolladores y los usuarios por la facilidad de aprender sus distintas representaciones y por su énfasis en la exposición jerárquica de los detalles.
- Está bien probado y comprobado, con muchos años de uso por la Fuerza Aérea de Estados Unidos (EEUU), otros proyectos gubernamentales e industria privada.
- Puede ser generado por una variedad de herramientas de modelado gráfico de computadora.

### *1.4.3. Notación de modelado de procesos de negocio (BPMN)*

La Notación de Modelado de Procesos de Negocio (Business Process Modeling Notation, BPMN) es un nuevo estándar de modelado de procesos de negocio. Esta notación está diseñada específicamente para coordinar la secuencia de procesos y los mensajes que fluyen entre los diferentes procesos participantes. BPMN define un diagrama de procesos de negocio (BPD, Business Process Diagrams), basado en una técnica de diagramas de flujo. BPD está conformado por un conjunto de elementos gráficos. Las cuatro categorías básicas de estos elementos son: objetos de flujo, objetos de conexión, calles y artefactos.

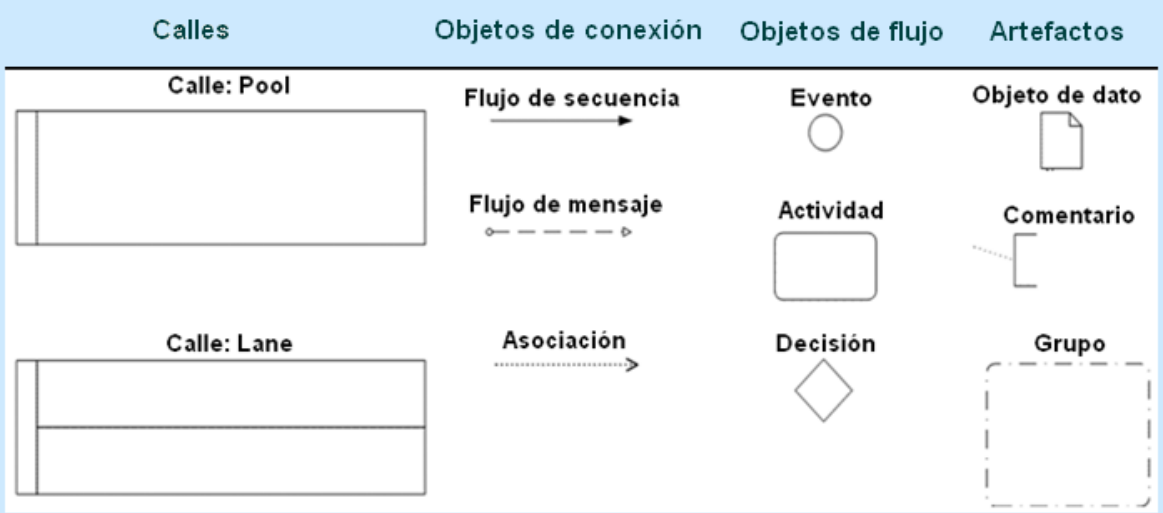



***Figura 4.*** Unidades básicas de BPMN.

Los objetos de flujo tienen tres elementos centrales: Evento, Actividad y Decisión. Los mismos se conectan en un diagrama para crear el esqueleto básico de la estructura de un proceso de negocio. Existen tres objetos de conexión: Flujo de secuencia (Sequence flow), Flujo de Mensaje (Message flow) y Asociación (Association).

Las calles son construidas teniendo en cuenta dos categorías diferentes, pool y lane. Pool representa un participante en un proceso y también actúa como un contenedor gráfico para separar un conjunto de actividades de otro pool. Lane es una sub-partición dentro de un pool y es utilizado para organizar y categorizar actividades.

La versión actual de BPMN predefine sólo tres tipos de artefactos: Objeto de datos (Data object), Grupo y Comentario [19].

### *1.5. Fundamentación del lenguaje de modelado a utilizar*

El uso de un lenguaje de modelado es más sencillo que la auténtica programación, pues existen menos medios para verificar efectivamente el funcionamiento adecuado del modelo. Esto puede suponer también que las interacciones entre partes del programa den lugar a sorpresas cuando el modelo ha sido convertido en un software.

Se decidió desarrollar el modelado de procesos mediante la utilización del lenguaje IDEF0, por las características amigables y la fácil comprensión que posee el lenguaje. Además se puede decir que el equipo de desarrollo estaba familiarizado con el lenguaje, debido a desarrollo de sistemas anteriores. Destacar además, que IDEF0 es un lenguaje multipropósito, cuyo uso se está estandarizando y aumentando entre los analistas hoy en día. Su característica de basarse en los procesos del sistema, nos permite analizar junto al

cliente los aspectos críticos del proyecto, dándole más participación y protagonismo a la comunicación cliente-desarrollador.

## *1.6. Herramientas CASE para el desarrollo de software*

Las **herramientas CASE** (Ingeniería de Software Asistida por Ordenador, por su siglas en ingles) 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. Las herramientas CASE nos pueden ayudar en todos los aspectos del ciclo de vida de desarrollo del software en tareas como el proceso de realizar un diseño del proyecto, calculo de costes, implementación de parte del código automáticamente con el diseño dado, compilación automática, documentación o detección de errores entre otras.

De acuerdo con Kendall y Kendall la ingeniería de sistemas asistida por ordenador es la aplicación de tecnología informática a las actividades, las técnicas y las metodologías propias de desarrollo, como objetivo tiene, acelerar el proceso para el que han sido diseñadas, en el caso de CASE para automatizar o apoyar una o más fases del ciclo de vida en el desarrollo de sistemas.

Existen algunas herramientas con características similares y que hoy en día son muy difundidas entre la comunidad informática mundial.

### *1.6.1. Rational Rose*

Este paquete de Rational permite documentar un producto software brindando una serie de facilidades para la elaboración de los artefactos que propone el Proceso Unificado Racional. Está compuesto a su vez por un grupo de herramientas que se especializan en un aspecto en específico:

- **Rational Rose Enterprise Edition:** Esta herramienta permite modelar visualmente un grupo de artefactos que se generan como parte de la metodología. Dividido en cuatro vistas fundamentales: Vista de Casos de Uso, Vista Lógica, Vista de Componentes y Vista de Despliegue; propone un grupo de artefactos predefinidos que facilitan el desarrollo de software con calidad.
- **Rational Requisite Pro:** Esta herramienta gestiona los requerimientos funcionales y no funcionales del sistema a elaborar.

- **Rational Clear Case:** Rational Clear Case proporciona una gestión del ciclo de vida y control de los activos de desarrollo de software.

- **Rational ClearQuest:** proporciona un seguimiento flexible de defectos y cambios en toda la empresa. Soporte completo para consultas con generación de múltiples informes y gráficos.

- **Rational Soda:** Automatiza la documentación del proyecto de software a lo largo de todo el ciclo de vida. Genera documentos mediante la extracción de datos solicitados directamente de los repositorios de datos de herramientas.

Existen dentro del paquete otras herramientas con otras funcionalidades que gestionan y satisfacen todo el proceso de desarrollo de software.

***Otras características son:***

- Software propietario.

- Ingeniería inversa para Java 1.5.

- Generación de código a partir de modelos en Ada, ANSI C ++, C++, CORBA, Java y Visual Basic.

- Capacidad de análisis de calidad de código.

- El Add-In para modelado Web provee visualización, modelado y las herramientas para desarrollar aplicaciones de Web.

- Modelado UML para trabajar en diseños de base de datos, con capacidad de representar la integración de los datos y los requerimientos de aplicación a través de diseños lógicos y físicos.

- Capacidad de crear definiciones de tipo documento XML para el uso en la aplicación.

- Integración con otras herramientas de desarrollo de Rational.

- Publicación web y generación de informes para optimizar la comunicación dentro del equipo.

- Soporte RUP para Ingeniería de Sistema.

- Sistemas Operativos apropiados: Windows 2000, Windows NT, Windows XP.

### *1.6.2. Visual Paradigm*

Visual Paradigm es una de las herramientas CASE del mercado, considerada muy completa y fácil de usar, con soporte multiplataforma y que proporciona excelentes facilidades de interoperabilidad con otras aplicaciones. Fue creada para el ciclo vital completo del desarrollo de software que lo automatiza y acelera, permitiendo la captura de requisitos, análisis, diseño e implementación. Visual Paradigm también proporciona características tales como generación del código, ingeniería reversa y generación de informes.

Tiene la capacidad de crear el esquema de clases a partir de una base de datos y crear la definición de base de datos a partir del esquema de clases. Permite invertir código fuente de programas, archivos ejecutables y binarios en modelos UML al instante, creando de manera simple toda la documentación. Está diseñada para usuarios interesados en sistemas de software de gran escala con el uso del acercamiento orientado a objeto, además apoya los estándares más recientes de las notaciones de Java y de UML. Incorpora el soporte para trabajo en equipo, que permite que varios desarrolladores trabajen a la vez en el mismo diagrama y vean en tiempo real los cambios hechos por sus compañeros.

**Ventajas de Visual Paradigm:**

- Soporta UML versión 2.1.
- Interoperabilidad con modelos UML2 (metamodelos UML 2.x para plataforma Eclipse) a través de XMI.
- Presenta un Editor de Detalles de Casos de Uso.
- Permite realizar el Diagramas de flujo de datos.
- Soporta la Generación de objetos Java desde la base de datos.
- Ofrece un Generador de informes para generación de documentación.
- Importación y exportación de ficheros XMI.
- Software propietario.
- 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.

### *1.6.3. BPwin*

BPwin es una herramienta de modelamiento de procesos de negocios, que te permite capturar las actividades (entiéndase procesos o funciones) del negocio para entenderlas mejor, modelarlas y cambiarlas en el caso necesario. Podemos definir el software como una potente herramienta para analizar, documentar y mejorar los procesos de negocio [26]. BPwin nos va a permitir documentar de manera clara los elementos más importantes de nuestro sistema, como que actividades son necesarias, cómo se realizan y qué recursos consumen, lo cual nos proporciona una visión exacta, no solo de qué es lo que necesita hacer el software, sino si lo hace de forma eficiente.

BPwin proporciona un marco de trabajo para poder representar y entender los procesos de negocio, determinando el impacto de los diferentes sucesos y definiendo cómo los procesos interactúan unos con otros mediante flujos de información permitiéndonos identificar actividades poco eficientes o redundantes.

***Como principales características posee [27]:***

- **Interface Intuitiva:** Se puede navegar fácilmente en todo el entorno pulsando un click a través de la técnica drag and drop.
- **Diseño Automatizado de Procesos:** Use BPwin para corregir o consistenciar los resultados en el diseño del proceso.
- **Propiedades Definidas por el Usuario:** Puede personalizar BPwin para capturar información relevante a sus procesos y utilizarlas cuando lo requiera.
- **Técnicas de Integración:** BPwin provee las técnicas más comerciales para documentar los procesos empresariales y flujos de proceso a través del IDEF0, IDEF3 y DFD.
- **Métrica y Análisis de Costos:** Actividades basadas en costos para la maximización de beneficios en los procesos pueden ser creadas de manera

simple y sencilla en BPwin. Todas estas podrán ser visualizadas fácilmente por la pantalla de los diagramas o a través de los reportes.

- **Pre-evaluación:** BPwin ofrece una interface para la simulación del software con la finalidad de explorar los efectos producidos en el negocio debido a los cambios realizados en el sistema.
- **Explorador de Modelos:** BPwin incluye un explorador que permite visualizar de manera alterna los objetos creados o la ramificación de un diagrama elegido con la finalidad de acceder más rápido a cada uno de ellos.
- **Diccionarios:** Todos los diccionarios poseen una interface gráfica formada por celdas y columnas parecida a una hoja de cálculo. Dichos diccionarios pueden ser personalizados con la finalidad de especificar sus necesidades de impresión, exportación o importación.
- **Diálogo de Propiedades:** Tanto los diagramas como cada uno de los elementos que participan en él, poseen una caja de dialogo con la finalidad de establecer propiedades relacionadas a color, tamaño, tipo de gráfico, cabeceras y pies del diagrama, configuración de pagina, etc.
- **Asociación de Entidades y Datos:** Usted puede asociar las entidades creadas en BPwin con el ERwin al momento de exportarlas.

## *1.7. Fundamentación de la herramienta CASE a utilizar.*

Se decide escoger Visual Paradigm como herramientas CASE para los diagramas de casos de uso y los diagramas de diseño, por sus amplias cualidades. Esta herramienta soporta hasta la fecha UML 2.1 completo y BPMN, permite realizar ingeniería tanto directa como inversa, ya que a partir de un modelo relacional en SQL Server, MySql, etc., es capaz de desplegar todas las clases asociadas a las tablas (siguiendo el patrón de diseño Una Clase-Una Tabla). Para gestionar la persistencia y el mapeo de estas clases con la base de datos utiliza Hibernate para Java y NHibernate en el caso de un proyecto .Net. Es una herramienta que no es libre y que solo tiene una versión freeware, la Community Edition. Se puede agregar además que la escuela en estos momentos posee una licencia para el uso de esta herramienta. Lo que posibilita su utilización en el desarrollo dentro del centro.

Se escoge además la herramienta BP Win como software de asistencia para la modelación de procesos de negocio. Esta herramienta está dirigida por funcionalidades o procesos, y nos permite modelar de forma eficiente como se desarrollan las actividades dentro del sistema. Utiliza dos metodologías: **DFD** e **IDEF0**. Podemos a agregar a sus

características fundamentales, el conocimiento de la herramienta que posee el equipo de desarrollo, su utilización en el desarrollo de otros productos con esta herramienta

## *1.8. Ingeniería de Requisitos*

La Ingeniería de Requisitos (IR), también conocida como *Etapa de Requerimientos* o *Administración de Requerimientos* es una actividad de suma importancia dentro de la Ingeniería del Software. La Etapa de Requerimientos se realiza como una actividad dentro de las fases iniciales en el desarrollo de un software.

Según Pressman el proceso de de administración de requerimientos, bien desarrollado, es la vía más factible de establecer una concordancia adecuada entre las necesidades del cliente y las ideas desarrolladas dentro del grupo de desarrollo [10].

Tradicionalmente entendida como una parte borrosa del ciclo de vida del software, en la que se obtiene una especificación formal, a partir de ideas informales [21].

Es necesario aclarar que uno de los grande dilemas que existen dentro de la comunidad informática es la de definir qué es lo que el cliente necesita y qué es lo que se necesita desarrollar para satisfacer al mismo. Es una de las fases más importantes en el desarrollo de un software, pues un concepto mal interpretado en esta fase, ocasionaría errores en el producto final, pues son difíciles de reparar.

La ingeniería puede ser derivada en varias actividades. Pressman las define de la siguiente forma:

- Elicitación de Requisitos.
- Análisis y negociación de Requisitos.
- Especificación de Requisitos.
- Validación de Requisitos.

### *1.8.1. Elicitación de requisitos*

***Elicitación:*** Es el proceso de adquirir (“eliciting”) o sonsacar todo el conocimiento relevante necesario para producir un modelo de los requerimientos de un dominio de problema, tiene como principal objetivo entender el dominio del problema en particular.

El proceso de elicitación de requisitos puede resultar complejo y costoso debido a numerosos problemas. Muchas veces los clientes/usuarios no están seguros de lo que

realmente necesitan, les es muy difícil comunicarle al analista sus necesidades, omiten información suponiendo que es obvia y la mala definición del límite del sistema [10].

Por la complejidad de este proceso, la Ingeniería de Requisitos propone técnicas o métodos que permiten hacerlo de una forma más eficiente y precisa. Las técnicas más utilizadas son las Entrevistas, el Desarrollo Conjunto de Aplicación (Joint Application Development, JAD), la Tormenta de ideas (Brainstorming) y otras como la utilización de Escenarios y Mapas conceptuales (Concept Mapping), entre otras.

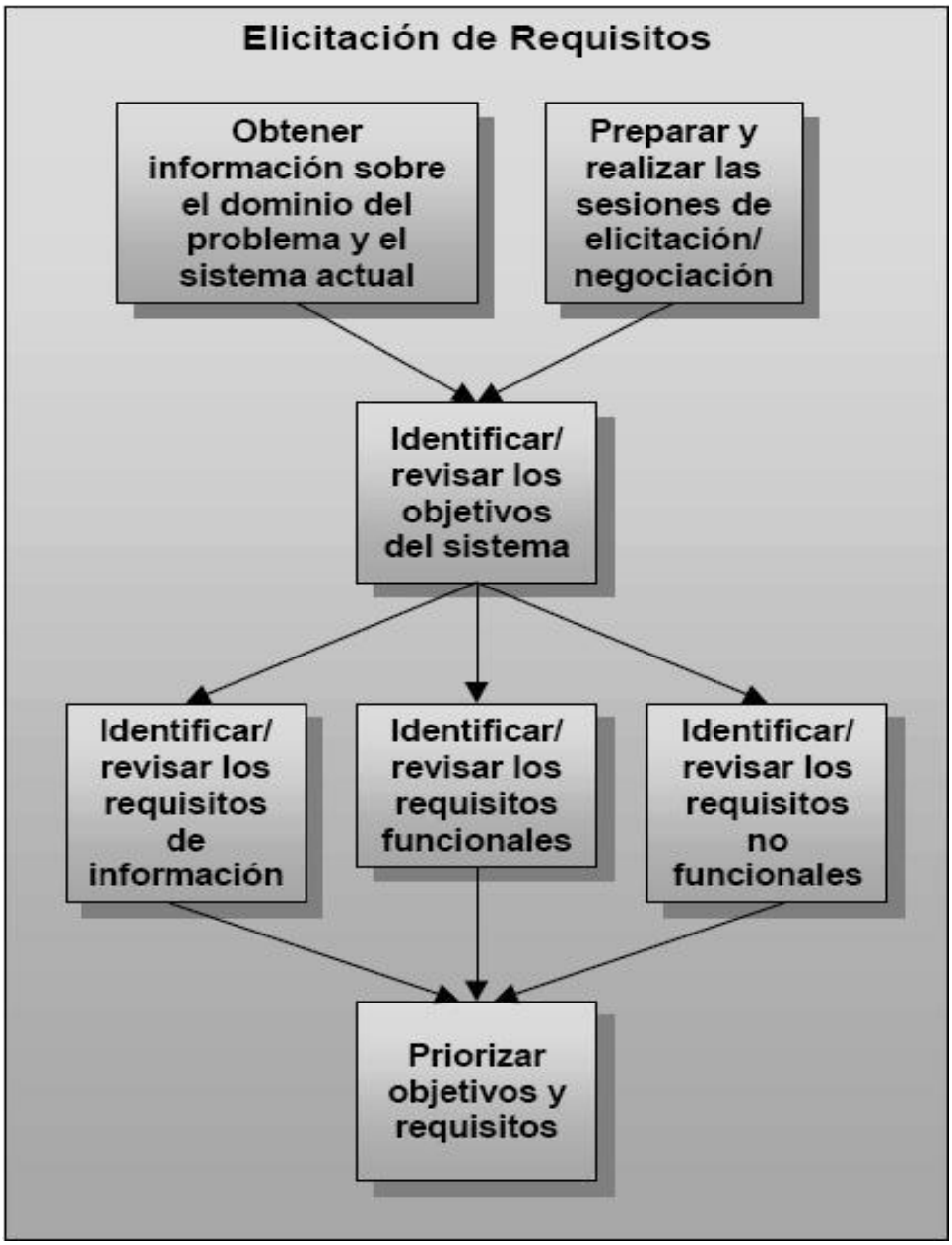



***Figura 5.*** Pasos de la Elicitación de Requisitos.

### *1.8.2. Análisis y negociación de Requisitos*

El análisis de requisitos es la clasificación de los mismos por categorías, organizándolos en subconjuntos, se examina su consistencia, completitud y ambigüedad y se clasifican en base a las necesidades de los clientes o usuarios. En caso de existir algún tipo de conflicto con los requisitos identificados se debe solucionar mediante el proceso de negociación. Los riesgos asociados con cada requisito serán identificados y analizados. Se valora el impacto de cada requisito con el coste y plazo de entrega del proyecto [10].

La negociación de requisitos es el proceso de discutir los conflictos que surgen entre los requisitos y llegar a algún compromiso que satisfaga a todos los usuarios. Dentro de la negociación se realizan las siguientes tareas:

- Discutir los requisitos conflictivos.
- Priorizar los requisitos.
- Compromiso final sobre el conjunto de requisitos a implementar.

### *1.8.3. Especificación de requisitos*

Para lograr que los requisitos del software se presenten de manera consistente y comprensible se deben especificar en una plantilla estándar. La alternativa más viable para sistemas grandes, es utilizar un documento escrito, con descripciones en lenguaje natural y modelados gráficos. La especificación de los requisitos de un sistema de software, describe su función, características, y las restricciones que gobiernan su desarrollo. Describe la información que entra y sale del sistema y su estado en cada momento. La flexibilidad es un punto a tener en cuenta cuando se va a realiza una especificación, los requisitos cambian, incluso en la etapa de la captura [10].

### *1.8.4. Validación de requisitos*

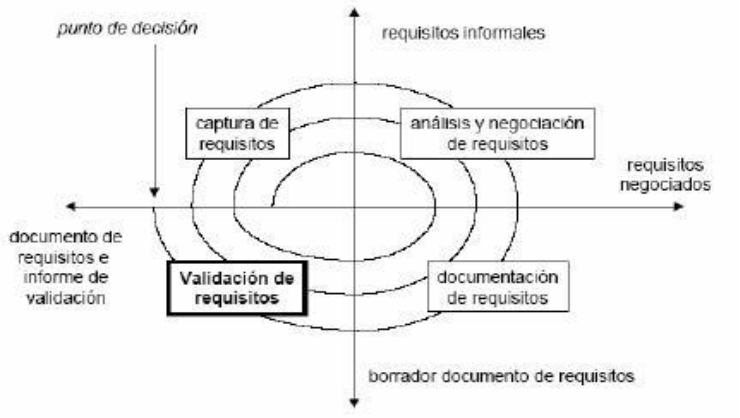



***Figura 6.*** Esboce grafico de la Ingeniería de Requisitos.

Para medir la calidad de los requisitos obtenidos y de los artefactos generados dentro de ingeniería de requisitos, es necesario realizar un proceso de validación.

Una revisión técnica formal (RTF) es una actividad que se desarrolla con el fin de garantizar la calidad del software, llevada a cabo por un grupo de personas (ingenieros de software, clientes, usuarios). Es el primer mecanismo para la validación de los requisitos. Con ella se buscan errores en el contenido, en la interpretación; se generan sugerencias al documento: dónde se necesita aclaración, dónde hay inconsistencias, dónde hay requisitos contradictorios o imposibles de cumplir. Se pretende garantizar el cumplimiento de los estándares previamente fijados y lograr un desarrollo uniforme.

Existen otras vías de validación como son:

- Prototipos no funcionales.
- Test requeridos al cliente e involucrados.

### *1.8.5. Técnicas para la captura de requisitos*

Una técnica es un procedimiento o conjunto de reglas, normas o protocolos, que tienen como objetivo obtener un resultado determinado.

*Captura de requisitos*: Actividad mediante la que se extraen las necesidades del sistema.

*Técnicas:*

Para la captura o recogida requisitos la disciplina IR propone una serie de métodos o técnicas dependiendo de la operación o la fase en la que se encuentra el analista como son:

- Entrevistas.
- Desarrollo Conjunto de Aplicación (JAD).
- Tormentas de Ideas.
- Mapas Conceptuales.
- Bocetos y Diagramas.
- Cuestionarios y Listas de Chequeo.
- Casos de Uso.
- Comparación de Terminología.

Podemos agregar que a nivel mundial hoy día existe una tendencia hacia la representación grafica y hacia la representación textual de los requisitos. Además existe

poco soporte en cuanto a una herramienta CASE se refiere, y que entre la comunidad internacional se desarrolla la disciplina de IR utilizando terminologías completamente variadas dependiendo de la experiencia y certidumbre del analista en cuestión.

### *1.9. Patrones*

Según Martin Fowler en su libro "*Analysis Patterns: Reusable Object Models",* expone que "…*Un patrón es una idea que ha sido útil en un contexto práctico y probablemente será útil en otros..."* [22]*.*

El arquitecto Christopher Alexander, fue uno de los primeros hombres en hablar acerca de patrones, cuando en 1979, publica el libro: **The Timeless Way of Building,** en este caso serian patrones arquitectónicos. Donde proponía una seria de patrones para la construcción de edificios con una mayor calidad.

Según Christopher, "Cada patrón describe un problema que ocurre infinidad de veces en nuestro entorno, así como la solución al mismo, de tal modo que podemos utilizar esta solución un millón de veces más adelante sin tener que volver a pensarla otra vez."

El libro *A Pattern Language*, referencia los patrones del arquitecto y sus colegas con el objetivo de formalizar y plasmar una forma práctica de este conocimiento arquitectónico. Los patrones no son principios abstractos que requieran su redescubrimiento para obtener una aplicación satisfactoria, ni son específicos a una situación particular o cultural, son algo intermedio. Un patrón define una posible solución correcta para un problema de diseño dentro de un contexto dado, describiendo las cualidades invariantes de todas las soluciones expuestas.

**Using Pattern Languages for OO Programs**, es la primera publicación que trato sobre patrones para el desarrollo de software con programación orientada a objetos. Ward Cunningham y Kent Beck son los autores de este articulo, quienes tomando ideas de Christopher, desarrollaron en 1987, 5 patrones de interacción hombre – ordenador.

Los Patrones poseen el siguiente formato:

- **Nombre del patrón**: nombre estándar del patrón por el cual será reconocido en la comunidad (normalmente se expresan en inglés).
- **Clasificación del patrón**: dentro de que clasificación entra.
- **Intención**: ¿Qué problema pretende resolver el patrón?
- **También conocido como**: Otros nombres de uso común para el patrón.
- **Motivación**: Escenario de ejemplo para la aplicación del patrón.

- **Aplicabilidad**: Usos comunes y criterios de aplicabilidad del patrón.
- **Estructura**: Diagramas de clases oportunos para describir las clases que intervienen en el patrón.
- **Participantes**: Enumeración y descripción de las entidades abstractas (y sus roles) que participan en el patrón.
- **Colaboraciones**: Explicación de las interrelaciones que se dan entre los participantes.
- **Consecuencias**: Consecuencias positivas y negativas en el diseño derivadas de la aplicación del patrón.
- **Implementación**: Técnicas o comentarios oportunos de cara a la implementación del patrón.
- **Código de ejemplo**: Código fuente ejemplo de implementación del patrón.
- **Usos conocidos**: Ejemplos de sistemas reales que usan el patrón.
- **Patrones relacionados**: Referencias cruzadas con otros patrones.

*Los patrones pretenden:*

- Proporcionar catálogos de elementos reusables en el análisis y diseño de sistemas software.
- Evitar la reiteración en la búsqueda de soluciones a problemas ya conocidos y solucionados anteriormente.
- Formalizar un vocabulario común entre los roles.
- Estandarizar el modo en que se realiza el análisis y el diseño.
- Facilitar el aprendizaje de las nuevas generaciones de diseñadores condensando conocimiento ya existente.

*Los patrones no pretenden:*

- Imponer ciertas alternativas de diseño frente a otras.
- Eliminar la creatividad inherente al proceso de análisis y diseño.

No es obligatorio utilizar los patrones en la elaboración de un sistema, solo es aconsejable en el caso de tener el mismo problema o similar que soluciona el patrón, siempre teniendo en cuenta que en un caso particular puede no ser aplicable. *Abusar o forzar el uso de los patrones puede ser un error* [23].

Puede decirse que hoy existen diferentes tipos de patrones, como son los patrones de análisis, patrones de casos de uso, los patrones de diseño, entre otros.

### *1.9.1. Patrones de Casos de Uso*

La experiencia en la utilización de casos de uso ha evolucionado en un conjunto de patrones que permiten con mayor precisión representar los requisitos reales, haciendo el trabajo con los sistemas más fácil, y mucho más simple su mantenimiento. Dado un contexto y un problema a resolver, estas técnicas han mostrado ser la solución adoptada en la comunidad de desarrollo de software. Se presentan como herramientas que permiten resolver los problemas que se les planteen a los desarrolladores de una forma rápida y concisa [17]. Estos patrones se enfocan hacia el diseño y las técnicas utilizadas en modelos de alta calidad, y no en cómo modelar casos de usos específicos. Utilizando estos patrones, ya seas arquitecto, analista, ingeniero, y gerente, puedes lograr mejores resultados de forma más rápida.

*Los patrones más conocidos de casos de uso, son los siguientes:*

- Reglas de negocio.
- Concordancia (Commonality).
- Componente jerárquico (Component hierarchy)
- Extensión o Inclusión Concreta.
- CRUD (Creating, Reading, Updating, Deleting).
- Caso de uso grande (Large Use case).
- Sistema de Capas.
- Múltiples actores (Roles Comunes).
- Servicio opcional.
- Vistas ortogonales.
- Secuencia de casos de uso.

### *1.9.2. Patrones de diseño*

Los primeros en hablar de patrones de diseño fueron Ward Cunningham y Kent Beck quienes adaptaron por primera vez las ideas de Alexander a la Ingeniería de Software.

Crearon cinco patrones para el diseño de las interfaces: Window per Task, Few Panes, Standard Panes, Nouns and Verbs, y Short Menu.

La influencia de Christopher Alexander se evidencia en muchas ramas de la sociedad, incluyendo a la informática, que no se quedo atrás. Así comienzan a surgir los patrones de diseño, que se agrupan mayormente en 2 grupos fundamentales.

**Patrones GRASP**

Los patrones generales de software para asignar responsabilidades, GRASP por sus siglas en ingles (General Responsability Asignment Software Paterns), describen en su totalidad los principios fundamentales sobre la asignación de responsabilidades a objetos, todo en forma de patrones.

Los patrones GRASP, son aplicables a los problemas surgidos en el diseño de cada sistema, y que la responsabilidad asignada no equivale a los métodos de la clase de la cual se instancia el objeto, los métodos son los encargados de cubrir las responsabilidades.

Los patrones GRASP constituyen el fundamento de cómo se diseña un sistema orientado a objetos, no introducen ideas nuevas, son una mera codificación de los principios básicos más usados. Cada uno de estos, soluciona problemas que contribuyen a que los sistemas sean más fuertes, robustos, flexibles y fáciles de mantener y extender. Los patrones GRASP son guías, ayudan a encontrar los patrones de diseños adecuados para aplicar al sistema que se esté diseñando. Puede decirse que dentro del mundo informático, se plantea, que más que patrones, propiamente dichos, son una serie de "buenas prácticas" de aplicación recomendable en el diseño de software.

Existen un total de 9 patrones GRASP:

| Nombre | Solución | Problema que resuelve |
|---|---|---|
| Experto | Asignar una responsabilidad al experto en información: la clase que cuenta con la información necesaria para cumplir la responsabilidad. | ¿Cuál es el principio fundamental en virtud del cual se asignan las responsabilidades en el diseño orientado a objetos? |
| Creador | Asignarle a una clase la responsabilidad de crear una instancia de otra clase en casos diferentes. | ¿Quién debería ser responsable de crear una nueva instancia de alguna clase? |

| | | |
|---|---|---|
| Bajo acoplamiento | Asignar una responsabilidad para mantener bajo acoplamiento. | ¿Cómo dar soporte a una dependencia escasa y a un aumento de la reutilización? |
| Alta cohesión | Asignar una responsabilidad de modo que la cohesión siga siendo alta. | ¿Cómo mantener la complejidad dentro de límites manejables? |
| Controlador | Asignar la responsabilidad del manejo de un mensaje de los eventos de un sistema a una de las clases. | ¿Quién debería encargarse de atender un evento del sistema? |
| polimorfismo | Cuando por el tipo varían las alternativas o comportamientos afines, las responsabilidades del comportamiento se asignarán mediante operaciones polimórficas a los tipos en que el comportamiento presenta variantes. | ¿Cómo manejar las alternativas basadas en el tipo? ¿De qué manera crear componentes de software conectables? |
| Fabricación Pura | Asignar un conjunto altamente cohesivo de responsabilidades a una clase artificial que no representa nada en el dominio del problema. | ¿A quién asignar la responsabilidad cuando se está desesperado y no se quiere violar los patrones Alta Cohesión y Bajo Acoplamiento? |
| Induración | Se asigna la responsabilidad a un objeto intermedio para que medie entre otros componentes o servicios, y estos no terminen directamente acoplados. | ¿A quién se asignarán las responsabilidades a fin de evitar el acoplamiento directo? ¿De qué manera se desacoplarán los objetos de modo que se obtengan un Bajo Acoplamiento ¿A quién se asignarán las responsabilidades a fin de evitar el acoplamiento directo? ¿De qué manera se desacoplarán los objetos de modo que se obtengan un Bajo Acoplamiento |
| No hables con Extraños | Se asigna la responsabilidad a un objeto directo del cliente para que colabore con un objeto indirecto, de modo que el cliente no necesite saber nada del objeto indirecto | ¿A quién asignar las responsabilidades para evitar conocer las estructuras de los objetos indirectos? |

## Patrones Gof

Fue a principios de los años 90, cuando surge el concepto de patrones de diseño, a partir del libro “Patrones de Diseño: Elementos de Software Reusable Orientado a Objetos” (Design Patterns: Elements of reusable Object Oriented Software), escrito por el grupo Gang of Four (“La pandilla de los 4”), compuesto por Erich Gamma, Jhon Vlisides, Richard Helm y Ralph Johnson, y donde exponían 23 patrones comunes para problemas de diseño, conocidos actualmente como los patrones Gof.

Estos patrones se dividen esencialmente en 4 grandes clasificaciones: Patrones creacionales o de creación, Patrones estructurales, Patrones de comportamiento y Patrones de sistema.

***Patrones creacionales***

- **Abstract Factory** (Fábrica abstracta): Permite trabajar con objetos de distintas familias de manera que las familias no se mezclen entre sí y haciendo transparente el tipo de familia concreta que se esté usando.
- **Builder** (Constructor virtual): Abstrae el proceso de creación de un objeto complejo, centralizando dicho proceso en un único punto.
- **Factory Method** (Método de fabricación): Centraliza en una clase constructora la creación de objetos de un subtipo de un tipo determinado, ocultando al usuario la casuística para elegir el subtipo que crear.
- **Prototype** (Prototipo): Crea nuevos objetos clonándolos de una instancia ya existente.
- **Singleton** (Instancia única): Garantiza la existencia de una única instancia para una clase y la creación de un mecanismo de acceso global a dicha instancia.

***Patrones Estructurales***

- **Adapter** (Adaptador): Adapta una interfaz para que pueda ser utilizada por una clase que de otro modo no podría utilizarla.
- **Bridge** (Puente): Desacopla una abstracción de su implementación.
- **Composite** (Objeto compuesto): Permite tratar objetos compuestos como si de un simple se tratase.
- **Decorator** (Envoltorio): Añade funcionalidad a una clase dinámicamente.

- **Facade** (Fachada): Provee de una interfaz unificada simple para acceder a una interfaz o grupo de interfaces de un subsistema.
- **Flyweight** (Peso ligero): Reduce la redundancia cuando gran cantidad de objetos poseen idéntica información.
- **Proxy**: Mantiene un representante de un objeto.

***Patrones de Comportamiento***

- **Chain of Responsibility** (Cadena de responsabilidad): Permite establecer la línea que deben llevar los mensajes para que los objetos realicen la tarea indicada.
- **Command** (Orden): Encapsula una operación en un objeto, permitiendo ejecutar dicha operación sin necesidad de conocer el contenido de la misma.
- **Interpreter** (Intérprete): Dado un lenguaje, define una gramática para dicho lenguaje, así como las herramientas necesarias para interpretarlo.
- **Iterator** (Iterador): Permite realizar recorridos sobre objetos compuestos independientemente de la implementación de estos.
- **Mediator** (Mediador): Define un objeto que coordine la comunicación entre objetos de distintas clases, pero que funcionan como un conjunto.
- **Memento** (Recuerdo): Permite volver a estados anteriores del sistema.
- **Observer** (Observador): Define una dependencia de uno-a-muchos entre objetos, de forma que cuando un objeto cambie de estado se notifique y actualicen automáticamente todos los objetos que dependen de él.
- **State** (Estado): Permite que un objeto modifique su comportamiento cada vez que cambie su estado interno.
- **Strategy** (Estrategia): Permite disponer de varios métodos para resolver un problema y elegir cuál utilizar en tiempo de ejecución.
- **Template Method** (Método plantilla): Define en una operación el esqueleto de un algoritmo, delegando en las subclases algunos de sus pasos, esto permite que las subclases redefinan ciertos pasos de un algoritmo sin cambiar su estructura.
- **Visitor** (Visitante): Permite definir nuevas operaciones sobre una jerarquía de clases sin modificar las clases sobre las que opera.

***Patrones de Sistema***

- **MVC** (Modelo Vista Controlador): Divide un componente o un subsistema en tres partes lógicas: modelo, vista y controlador, facilitando la modificación o personalización de cada parte.
- **Session** (Sesión): Ofrece una forma de que los servidores de sistemas distribuidos sean capaces de distinguir los clientes, permitiendo que las aplicaciones asocien el estado con la comunicación entre el cliente y el servidor.
- **Worker Thread** (Thread trabajador): Mejora la productividad y minimiza la latencia media.
- **Callback** (Retro llamada): Permite que un cliente se registre en un servidor para ciertas operaciones. De esta forma, el servidor puede notificar al cliente cuando la operación ha finalizado.
- **Succesive Update** (Actualización Sucesiva): Ofrece a los clientes una forma de recibir actualizaciones continuas.
- **Router** (Encaminador): Desacopla múltiples fuentes de información de los objetos de esa información.
- **Transaction** (Transacción): Agrupa una colección de métodos de forma que todos ellos finalicen correctamente o fallen de forma colectiva.

## *1.10. Lenguajes de Programación*

Un lenguaje de programación es un lenguaje que puede ser utilizado para controlar el comportamiento de una máquina, particularmente una computadora. Consiste en un conjunto de símbolos y reglas sintácticas y semánticas que definen su estructura y el significado de sus elementos y expresiones. Un lenguaje de programación permite a uno o más programadores especificar de manera precisa: sobre qué datos una computadora debe operar, cómo deben ser estos almacenados y transmitidos y qué acciones debe tomar bajo una variada gama de circunstancias.

En la actualidad existen numerosos lenguajes de programación para desarrollar la web, tanto lenguajes de lado cliente como de lado servidor. Cuando se habla de lenguajes de lado cliente, son aquellos capaces de ser leídos directamente por el navegador sin necesidad de un pre-tratamiento.

Entre los lenguajes de lado cliente se encuentran:

- ***HTML*** (HyperText Markup Language por sus siglas en ingles, en español Lenguaje de Marcas Hipertextuales), es un lenguaje estático para el desarrollo de sitios web, desarrollado por la W3C (World Wide Web Consortium). Los archivos pueden tener las extensiones htm y html. Entre sus ventajas se encuentra que es admitido por todos los exploradores, sencillo y de archivos pequeños. Como desventaja resalta que la interpretación de cada navegador puede ser diferente.

- ***Java script*** es un lenguaje interpretado creado por Brendan. El código Java script puede ser integrado dentro de las páginas web. El W3C para evitar incompatibilidades diseñó un estándar para la modelación de objetos del documento, más conocido como DOM (Document Object Model, por sus siglas en ingles). Entre sus ventajas está su scripting seguro y fiable y contradictoriamente su debilidad es su visibilidad al alcance de cualquier usuario.

- ***AJAX*** acrónimo para **Asynchronous Java Script + XML,** en realidad no es una tecnología, sino la unión de varias tecnologías que juntas se encargan de cargar y renderizar (mostrar) una página, luego mantenerse en esa página, mientras scripts y rutinas van al *servidor* buscando los datos que son usados para actualizar la pagina, sólo re-renderizando (actualizando) la misma y mostrando u ocultando porciones de ella. AJAX incorpora:
  - Presentación basada en estándares usando XHTML y CSS
  - Exhibición e interacción dinámicas usando el Document Object Model
  - Intercambio y manipulación de datos usando XML and XSLT
  - Recuperación de datos asincrónica usando XMLHttpRequest

Por otra parte se encuentran los lenguajes de lado servidor que son aquellos lenguajes reconocidos, ejecutados e interpretados por el propio servidor y que se envían al cliente en un formato comprensible para él.

*Entre los lenguajes de lado servidor se encuentran:*

- **ASP.NET** es un lenguaje comercializado por Microsoft. Fue desarrollado para resolver las limitantes de su antecesor ASP. Para el desarrollo de ASP.NET se puede utilizar C# o J#, entre otros. Creado para desarrollar, tanto webs sencillas como grandes aplicaciones para Internet. Para su funcionamiento se necesita tener instalado IIS (Internet Information Server) con el Framework .Net. Entre sus ventajas se encuentra:

- ❖ Es un lenguaje completamente orientado a objetos.
- ❖ Su velocidad de respuesta del servidor.
- ❖ Su seguridad.

El consumo de recursos es su desventaja.

- **JSP** (Java Server Pages) desarrollado por Sun Microsystems, está orientado a desarrollar páginas web en Java. Comparte ventajas similares a la de ASP.NET y fue desarrollado principalmente para la creación de aplicaciones web potentes. Posee un motor de páginas basado en los servlets de Java. Se necesita tener instalado un servidor Tomcat en alguna de sus versiones para su funcionamiento. Comparado mucho por su potencialidad con ASP.NET; donde se destaca por ser multiplataforma. Es un tanto complejo en su aprendizaje.

Por decisión del equipo de arquitectura, se desarrollara el sistema, basándose en la filosofía 3 capas (MVC), implementando con el lenguaje Java sobre el IDE Eclipse. Se escogió además utilizar la tecnología AJAX basada en Java Script.

## *1.11. Conclusiones parciales*

Podemos concluir esta parte alegando que durante este capítulo se ha desarrollado un exhaustivo análisis sobre el desarrollo del software a nivel mundial. Se ha demostrado que existen herramientas para la gestión de proyectos pero que ninguno cumple con las exigencias descritas por lo que se decide confeccionar un software que asuma las características necesarias para satisfacer al cliente.

En el mundo informático de hoy las metodologías y los lenguajes de modelado en el desarrollo de un software determinan la calidad y el tiempo de entrega del mismo, se determinó utilizar en nuestro caso RUP e IDEF0 por razones anteriormente expuestas y como herramienta CASE para el desarrollo de los mismo Visual Paradigm y BPwin.

Un estudio sobre la disciplina de Ingeniería de Requisitos nos permitió estar al día en cuanto a técnicas y métodos para la captura de requisitos más utilizados hoy por la comunidad informática.

## *Capitulo 2. Solución Propuesta*

### *2.1. Introducción*

La contratación de los proyectos financiados, se realiza en el nivel más bajo del flujo de trabajo que está integrado esencialmente en 3 niveles. Los Entes Ejecutores (EE), son aquellas pequeñas sucursales o delegaciones de una empresa determinada, que existen en todo el país. Estos Entes Ejecutores (EE) son los encargados de elaborar la propuesta de contrato a los proyectos financiados anteriormente. Una vez aprobado por los mismos, los contratos suben al nivel inmediato superior, es decir a los Ministerios (M). Estos son los encargados de revisar los contratos con el objetivo de encontrar alguna irregularidad o dato no valido dentro del contrato. Si el contrato es rechazado por alguno de los Ministerios (M), el mismo regresa al nivel anterior, con el objetivo de que los Entes Ejecutores (EE) realicen los cambios necesarios. Una vez que los contratos son aprobados por los Ministerios (M), estos pasan a nivel de las Secretarías Técnicas (ST), que son las entidades a nivel de estado que revisan haciendo énfasis en aspectos económicos del contrato o aspectos de RRHH, en caso de que algún contrato pase a estado rechazado por la Secretarías Técnicas (ST), el mismo desciende un nivel, es decir pasa al nivel inmediato inferior, los Ministerios (M). Los contratos después de ser aprobados por las mismas, pasan a ser firmados por los Ministerios (M), con la participación de los Entes Ejecutores (EE).

### *2.2. Modelado de Procesos*

Se procede a la modelación de los procesos organizacionales que tienen lugar en la contratación de los proyectos con financiamiento. Para la modelación se utilizó la metodología IDEF0, la cual es una metodología para el modelado funcional de procesos que permite representar de manera estructurada y jerarquizada las actividades que conforman un sistema o empresa y los objetos o datos que soportan la interacción de esas actividades a cualquier nivel de detalle que se desee.

El modelado de procesos permite comunicar reglas y procesos de negocios, se utiliza para la obtención de la visión estratégica de cualquier proceso de negocios logrando así la facilitación del análisis para la identificación de áreas de mejora.

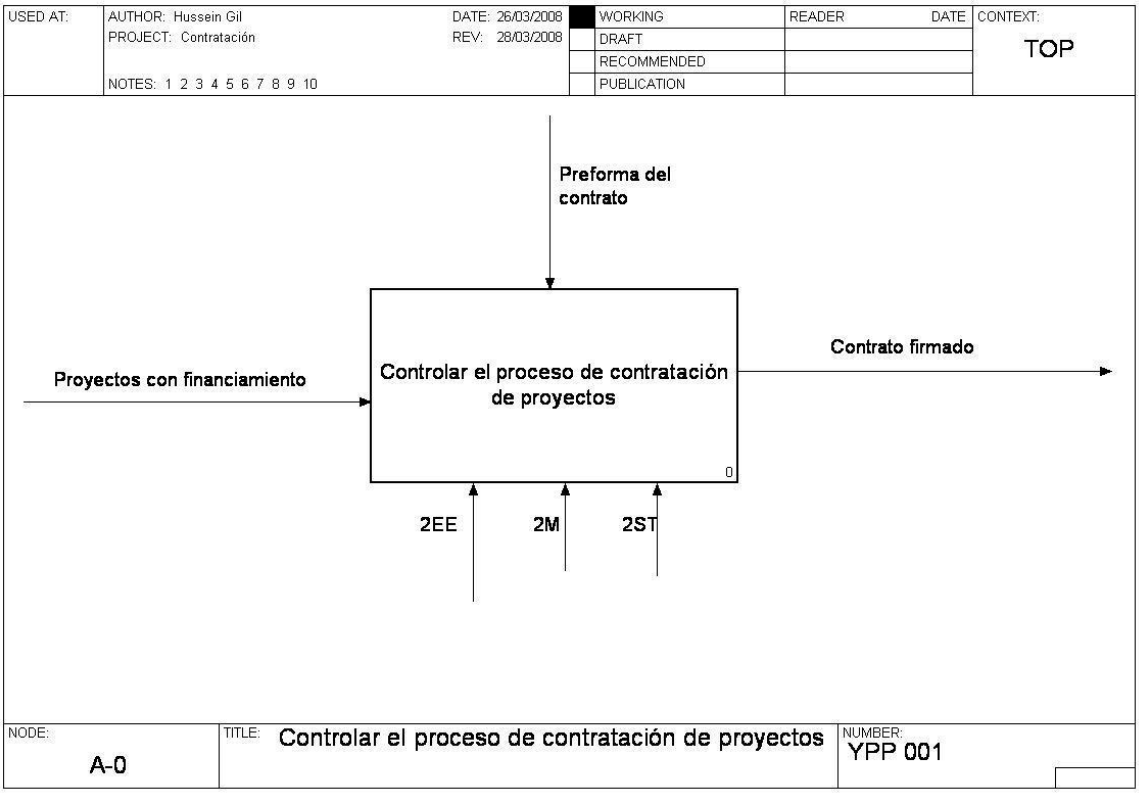



***Figura 7.*** Objetivo del Módulo Contratación de Proyectos

Dentro del proceso de contratación se enmarcan cinco actividades fundamentales enunciadas a continuación:

- La elaboración del contrato es donde se elabora y aprueba el mismo por los Entes Ejecutores.
- La aprobación del contrato por los Ministerios, donde es revisado el contrato por los Ministerios para ser aceptado o rechazado.
- La aprobación del contrato por la Secretaria Técnica, donde es revisado el contrato por la misma después de ser aprobado por los Ministerios.
- La firma del contrato, donde los Ministerios pasan a firmar el contrato después de ser aprobado por Secretaria Técnica.
- La modificación del cronograma de Ejecución Financiera, donde los Entes Ejecutores realizan alguna modificación al cronograma de E. Financiera de un proyecto, que luego pasara ha ser aprobado o rechazado por los Ministerios y Secretaria Técnica.

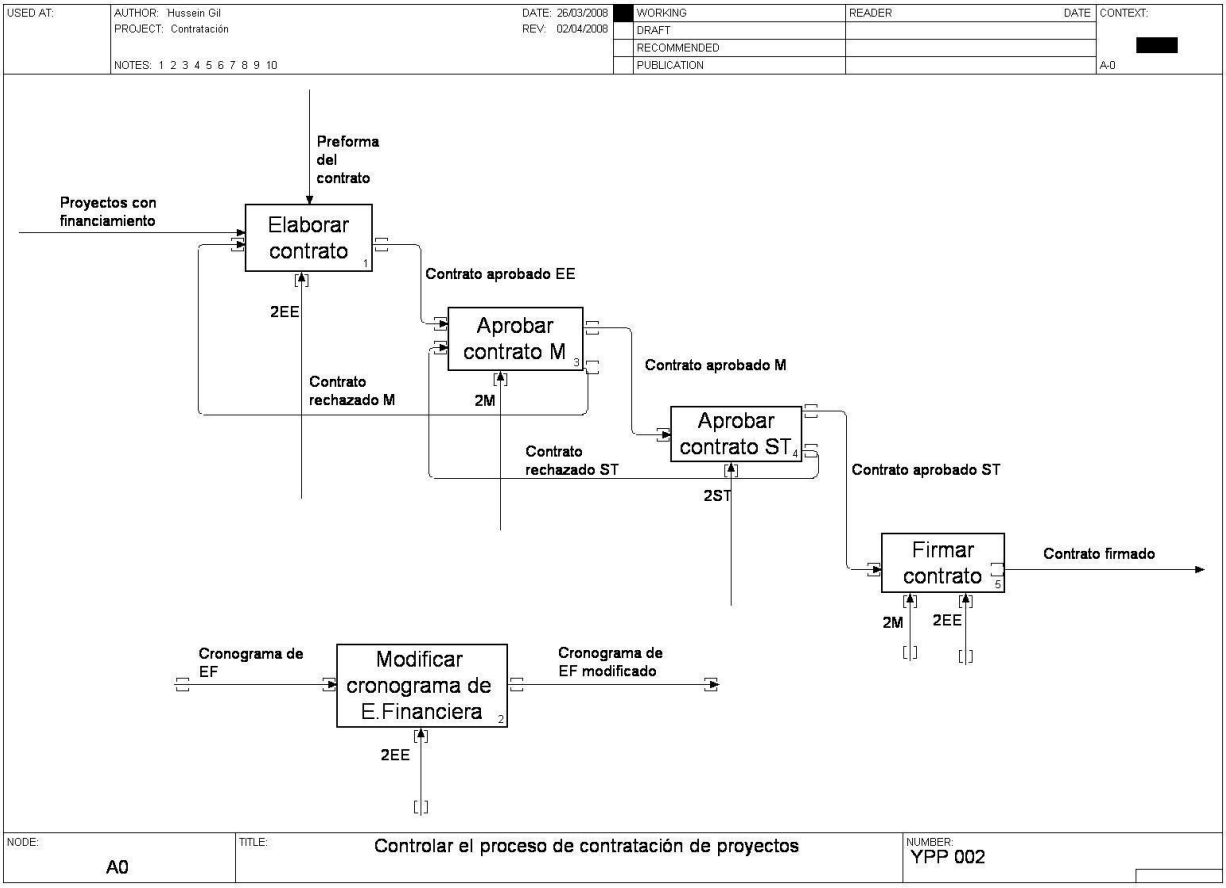



***Figura 8.*** Actividades del Módulo Contratación.

Para lograr un mejor entendimiento por parte de los clientes y desarrolladores de los procesos del negocio, a continuación se detallaran cada uno de los procesos anteriormente especificados.

### *2.2.1. Elaborar propuesta de contrato*

Un Ente Ejecutor (EE) elabora una propuesta de contrato, la cual envía a su EE contraparte para que este la evalúe, es decir la acepte o la rechace. La propuesta se elabora haciendo una búsqueda del(los) proyecto(s) a incluir en el posible contrato, luego se buscara la preforma de contrato, previamente llenada con los datos del contrato. Una vez creada la propuesta, esta puede ser modificada por el EE que la concibió. Si es rechazada por el EE contraparte, envía una notificación de rechazo, además de la propuesta rechazada con una nota incluida donde especifica el motivo por el cual la propuesta sufrió el rechazo. En caso de ser modificada por un EE vuelve al flujo de aprobación por el EE contraparte. Si es aceptada la propuesta de contrato, envía una notificación de aceptación, además de la propuesta de contrato aceptada. Luego de ser elaborada la propuesta de contrato, un EE puede proponer el envío de la propuesta al nivel superior, en este caso los Ministerios.

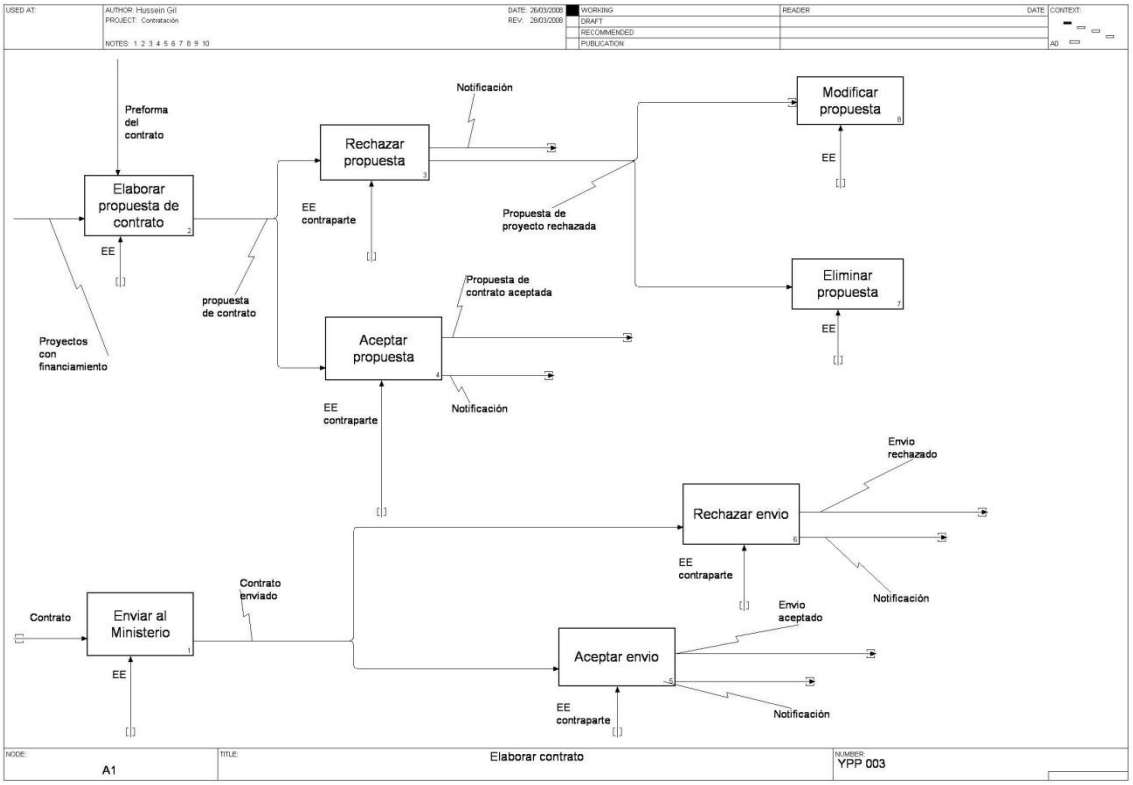



***Figura 9***. Elaborar propuesta de contrato.

Una vez aprobada la propuesta cualquiera de los Entes Ejecutores involucrados en la propuesta, pueden proponer una modificación en la propuesta de contrato.

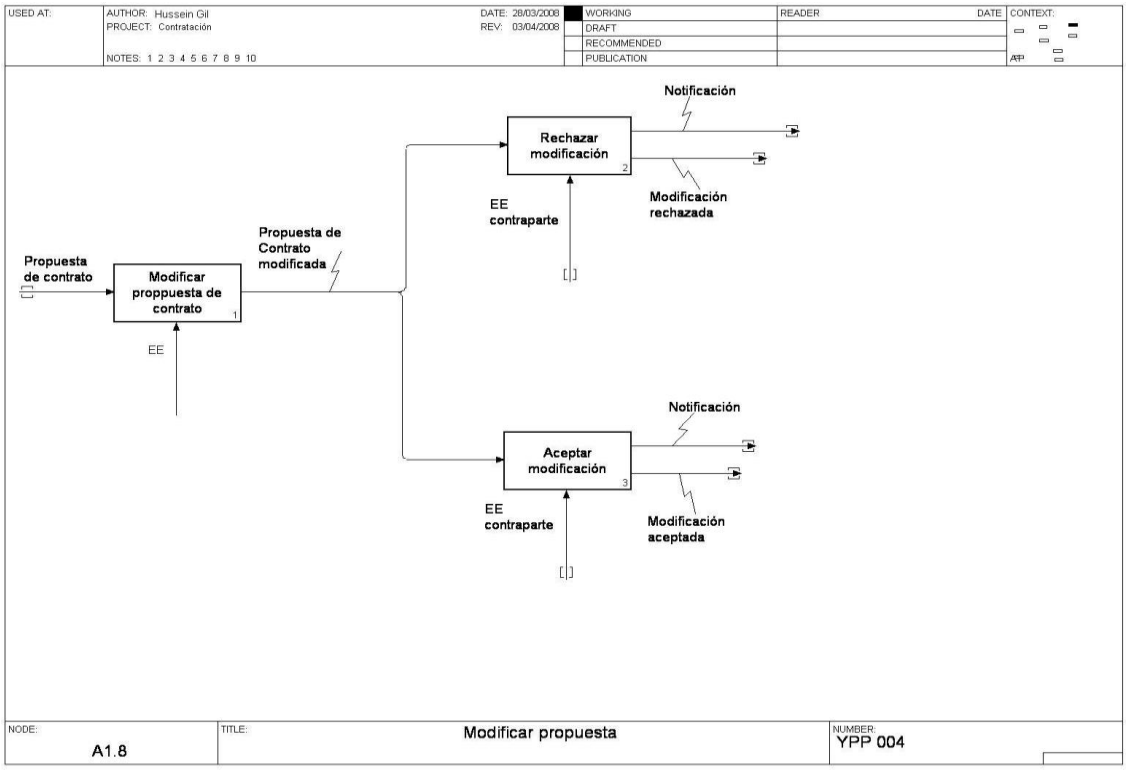



***Figura 10.*** Modificar propuesta de contrato

La propuesta de contrato puede estar sujeta a la proposición de eliminación por parte de cualquier EE involucrado, y esta propuesta entra nuevamente en el flujo aceptación y rechazo por ambos Entes Ejecutores.

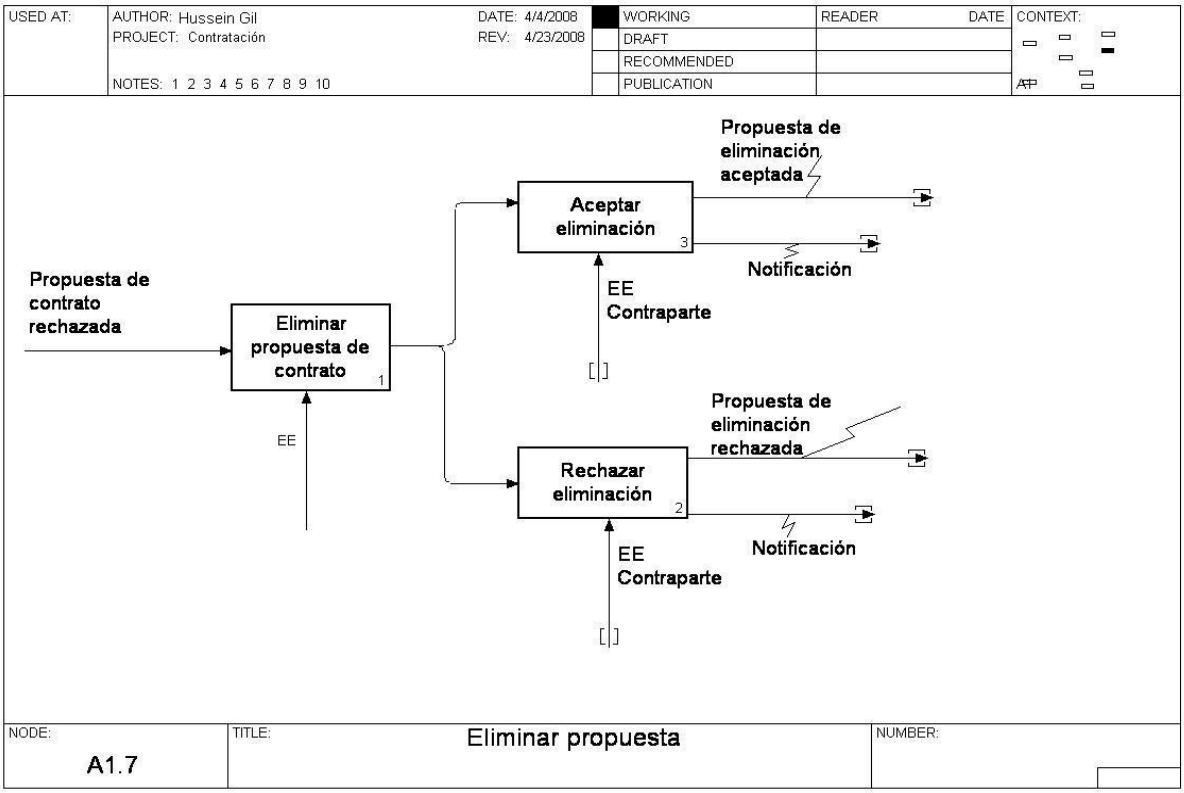



***Figura 11.*** Eliminar propuesta de contrato.

### *2.2.2. Aprobación de la propuesta de contrato por Ministerios*

Los Ministerios verifican la validez del contrato. Si uno de los Ministerios rechaza el contrato, envía una notificación a los EE, además del contrato rechazado con una nota incluida donde se especifica el motivo, por lo que regresa al proceso de elaboración. En caso de ser aprobado por uno de ellos pasa a ser revisada por el Ministerio contraparte para que acepte o rechace. Si finalmente es aceptado por los dos Ministerios envían una notificación, además del contrato aceptado. Después de aceptado el contrato los Ministerios pueden modificarlo, enviando una notificación a los EE o enviarlo a ser evaluado por la ST.

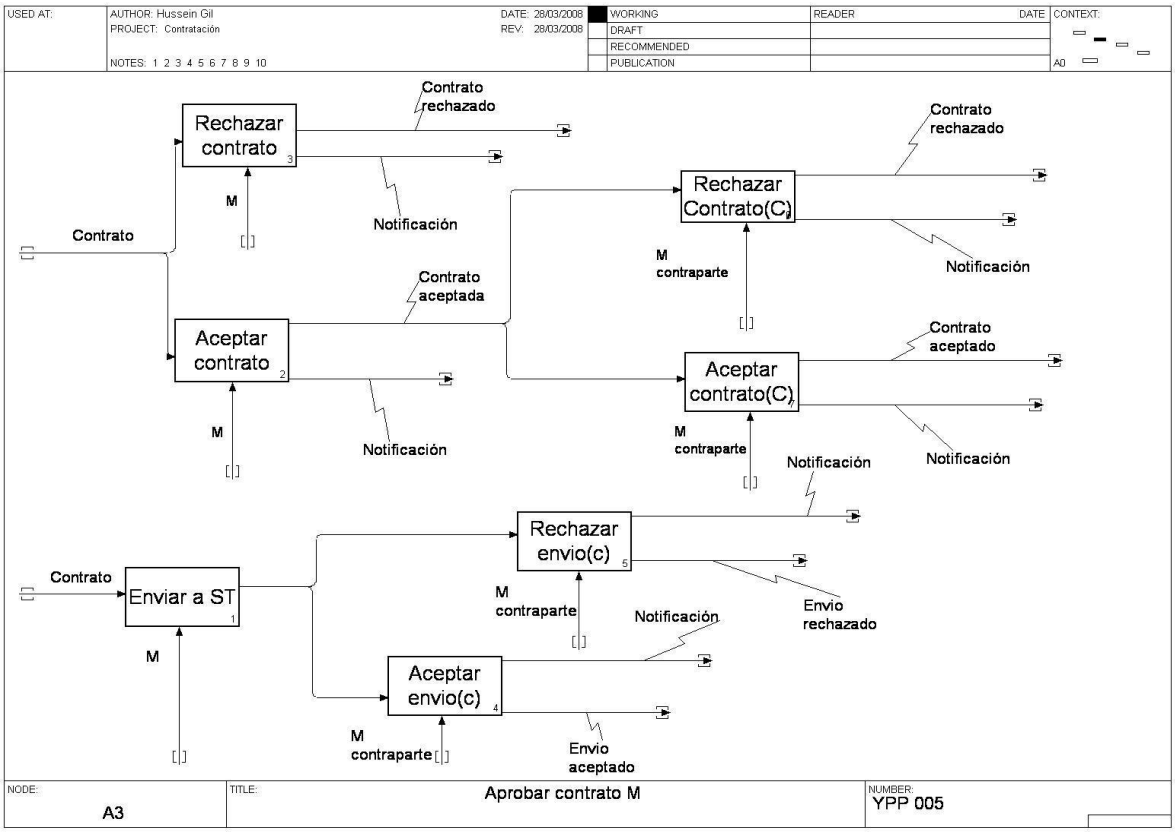



***Figura 12.*** Aprobaciónde propuesta de contrato por Ministerios.

En esta parte se puede definir como objetivo: La Aprobación de la propuesta de Contrato y que los Ministerios son los responsables.

### *2.2.3. Aprobación de la propuesta de contrato por las Secretarias Técnicas*

Las Secretarías Técnicas verifican la validez del contrato. Si una de las ST rechaza el contrato, envía una notificación a los EE y Ministerios implicados, además del contrato rechazado por lo que regresa a ser evaluado por los Ministerios, los cuales pueden modificarlo o enviarlo nuevamente a los EE para que lo modifiquen. En caso de ser aprobada por una de ellas, pasa a ser revisada por la ST contraparte para que acepte o rechace. Si finalmente es aceptada por las dos ST envían una notificación, además del contrato aceptado, pasando así al proceso de firmar el contrato por los Ministerios y los Entes Ejecutores.

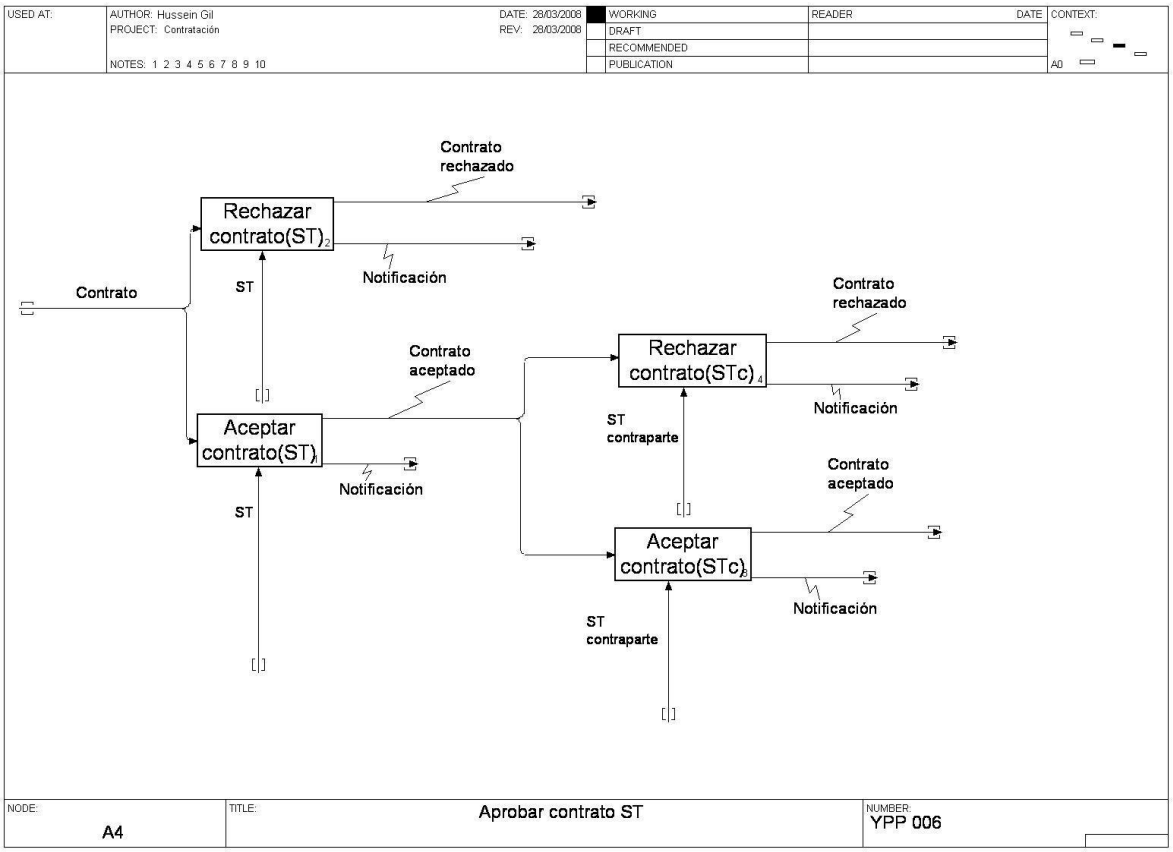



***Figura 13.*** Aprobación de propuesta de contrato por Secretarias Técnicas

### *2.2.4. Firmar propuesta de contrato*

Después que el contrato es aprobado por las ST pasa al proceso de firma. El mismo estará regido por los Ministerios involucrados en el proceso del contrato en cuestión. La Firma del contrato, se desarrollara con la presencia de los Entes Ejecutores responsables de la elaboración de la propuesta de contrato y por los Ministerios rectores de los mismos.

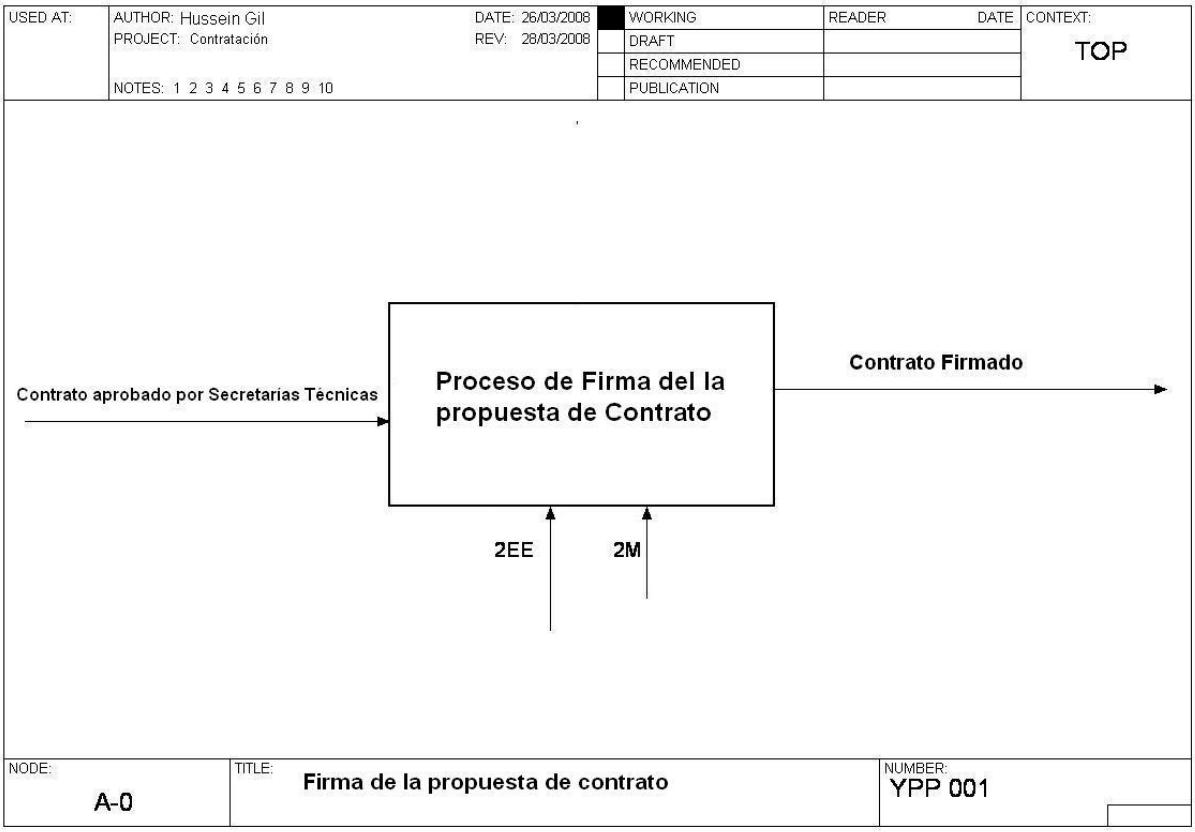



***Figura 14.*** Firma de la propuesta de contrato.

### *2.3. Reglas del negocio*

| | |
|---|---|
| Nombre | Los contratos son elaborados por los EE. |
| Identificador | RN 01 |
| Tipo | |
| Descripción | Los Entes Ejecutores son los únicos que pueden elaborar contratos, para esto redactan un documento teniendo en cuenta los datos que se piden en la preforma de contrato y además incluyen los proyectos que forman parte del mismo. |
| Fuente | Documento Modelo de Procesos |
| Reglas relacionadas | |

| | |
|---|---|
| Nombre | La aprobación ó rechazo de cambios en los contratos y cronograma de Ejecución Financiera son aprobados por |

| | |
|---|---|
| | ambas partes (cubana y venezolana). |
| Identificador | RN 02 |
| Tipo | |
| Descripción | Todos los cambios que se efectúen, de modificación, eliminación de contratos y modificaciones en el cronograma de Ejecución Financiera debe ser aprobado ó rechazados por las dos partes implicadas, por la venezolana y la cubana, ya sea a nivel de EE, Min, ST ó siguiendo esa misma jerarquía dependiendo de todos los niveles de aprobación. |
| Fuente | Documento Modelo de Procesos |
| Reglas relacionadas | |

| | |
|---|---|
| Nombre | La modificación y eliminación de contratos se notifica. |
| Identificador | RN 03 |
| Tipo | |
| Descripción | Todos los cambios que se efectúen, de modificación y eliminación de contratos deben ser notificados a todos los implicados en el proceso. |
| Fuente | Documento Modelo de Procesos |
| Reglas relacionadas | |

| | |
|---|---|
| Nombre | Colores de los vínculos del nombre de los Contratos. |
| Identificador | RN 04 |
| Tipo | |
| Descripción | El color azul de los vínculos significan que se está en un estado normal, donde ambas partes, cubana y venezolana, ya sea a nivel de EE, Min o ST están de acuerdo en el estado en que esta el proyecto o la mixta.<br><br>El color rojo significa que una de las partes ha hecho un cambio o ha adicionado algo nuevo que está en espera que la contraparte lo apruebe o lo rechace.<br><br>El color verde significa que se ha producido un cambio o modificación en el proyecto o la mixta y que se requiere |

| | |
|---|---|
| | de la aprobación de la parte que lo observa verde. |
| Fuente | Documento Modelo de Procesos |
| Reglas relacionadas | |

| | |
|---|---|
| Nombre | Los contratos son modificados por los EE y Ministerios. |
| Identificador | RN 05 |
| Tipo | |
| Descripción | Un contrato puede ser modificado por el EE que lo elabora y los Ministerios, por este último después de ser aceptada la propuesta de los EE. |
| Fuente | Documento Modelo de Procesos |
| Reglas relacionadas | |

| | |
|---|---|
| Nombre | Los contratos son eliminados por los EE. |
| Identificador | RN 06 |
| Tipo | |
| Descripción | Los contratos solos pueden ser eliminados por el EE que los elabora. |
| Fuente | Documento Modelo de Procesos |
| Reglas relacionadas | |

| | |
|---|---|
| Nombre | Los contratos pueden ser rechazados por la ST. |
| Identificador | RN 07 |
| Tipo | |
| Descripción | Si un contrato es rechazado por al menos una ST, este queda rechazado y vuelve a nivel de los Ministerios para ser preconcebido por este o para que lo envíe a los EE y sean ellos lo que se encarguen de modificarlo. |
| Fuente | Documento Modelo de Procesos |
| Reglas relacionadas | |

| | |
|---|---|
| Nombre | El monto total del contrato es la suma de los montos de los proyectos incluidos en el mismo. |
| Identificador | RN 08 |
| Tipo | |
| Descripción | El monto total se determina después de incluir los proyectos que formaran parte del contrato, por lo que está dado por la suma del monto de cada uno de estos proyectos. |
| Fuente | Documento Modelo de Procesos |
| Reglas relacionadas | |

| | |
|---|---|
| Nombre | Los contratos son propuestos por los EE. |
| Identificador | RN 09 |
| Tipo | |
| Descripción | Para proponer un contrato es necesario elaborar el documento con los datos especificados en la preforma, además de agregar los proyectos que estarán incluidos en el contrato. |
| Fuente | Documento Modelo de Procesos |
| Reglas relacionadas | |

| | |
|---|---|
| Nombre | Los contratos son firmados por los Ministerios y EE. |
| Identificador | RN 10 |
| Tipo | |
| Descripción | Los contratos que tienen como estado aprobados por la ST, pasan a ser firmados por los Ministerios y EE implicados. |
| Fuente | Documento Modelo de Procesos |
| Reglas relacionadas | |

| | |
|---|---|
| Nombre | Los contratos aprobados por los Ministerios pasan a ST. |
| Identificador | RN 11 |
| Tipo | |
| Descripción | Todo contrato enviado a las ST debe ser aprobado por los dos ministerios implicados en el contrato y después es que se pasará a ser revisada por ambas ST. |
| Fuente | Documento Modelo de Procesos |
| Reglas relacionadas | |

| | |
|---|---|
| Nombre | Los EE pueden modificar el cronograma de Ejecución Financiera de un proyecto. |
| Identificador | RN 12 |
| Tipo | |
| Descripción | Si es necesario realizar algún cambio en el cronograma de Ejecución Financiera de un proyecto, los únicos que pueden efectuar la modificar son los EE, sin alterar el monto total, el monto a invertir en Venezuela y el monto a transferir a Cuba. |
| Fuente | Documento Modelo de Procesos |
| Reglas relacionadas | |

| | |
|---|---|
| Nombre | Los contratos aprobados por EE pasan a los Ministerios. |
| Identificador | RN 13 |
| Tipo | |
| Descripción | Todo contrato enviado a los ministerios debe ser aprobado por los dos entes ejecutores implicados en el contrato y después es que se pasará a ser revisada por ambos ministerios. |
| Fuente | Documento Modelo de Procesos |
| Reglas relacionadas | |

| | |
|---|---|
| Nombre | Los contratos pueden ser rechazados por los Ministerios. |

| | |
|---|---|
| Identificador | RN 14 |
| Tipo | |
| Descripción | Si un contrato es rechazado por al menos un ministerio ya el proyecto queda rechazado y vuelve a nivel de los entes para ser preconcebido o eliminado. |
| Fuente | Documento Modelo de Procesos |
| Reglas relacionadas | |

## *2.4. Especificación de requisitos*

Los requisitos generales del sistema se obtienen del cliente. Estos requisitos recogen las funcionalidades que debe cumplir el sistema, según las necesidades del cliente. Además de su comportamiento y rendimiento.

Los requisitos funcionales definen el comportamiento interno del software, ya sean cálculos, detalles técnicos, manipulación de datos y otras funcionalidades específicas que muestran cómo los casos de uso serán llevados a la práctica.

Los requisitos no funcionales especifican criterios que pueden usarse para juzgar la operación de un sistema en lugar de sus comportamientos específicos.

### *2.4.1. Requisitos funcionales*

**RF02.001 Elaborar contrato**

El sistema permitirá que uno de los Entes Ejecutores elabore una propuesta de contrato y se la envíe a su EE contraparte para que este acepte o rechace la propuesta.

**RF02.002 Aceptar contrato**

El sistema permitirá la aceptación de contrato.

**RF02.003 Rechazar contrato**

El sistema permitirá rechazar contrato.

**RF02.004 Modificar contrato**

El sistema permitirá la modificación de contrato, la modificación está enmarcada en incluir o excluir proyectos del contrato.

**RF02.005 Eliminar contrato**

El sistema permitirá que los Entes Ejecutores eliminen una propuesta de contrato.

**RF02.006 Enviar contrato**

El sistema permitirá el envío de contrato.

**RF02.007 Realizar búsqueda de contratos.**

El sistema permitirá realizar la búsqueda de los contratos por los nombres de los proyectos, EE contraparte y/o Ministerio contraparte.

**RF02.008 Listar contratos.**

El sistema permitirá listar todos los contratos como resultado de la búsqueda realizada, incluyendo por contrato: su Nombre, EE contraparte, Ministerio contraparte y Estado del contrato.

**RF02.009 Listar proyectos.**

El sistema permitirá listar todos los proyectos como resultado de la búsqueda realizada, incluyendo por proyecto su Nombre, EE contraparte, Monto, Invertir en Venezuela, transferir a Cuba y modalidad.

**RF02.010 Realizar búsqueda de documento.**

El sistema permitirá realizar una búsqueda de documentos en cualquier directorio de la máquina.

**RF02.011 Modificar cronograma Ejecución Financiera.**

El sistema permitirá que uno de los Entes Ejecutores modifique el cronograma de Ejecución financiera de un proyecto y se la envíe a su EE contraparte para que este acepte o rechace la modificación.

**RF02.012 Aceptar modificación de cronograma Ejecución Financiera.**

El sistema permitirá la aceptación de modificación de cronograma de Ejecución Financiera.

**RF02.013 Rechazar modificación de cronograma Ejecución Financiera.**

El sistema permitirá rechazar la modificación del cronograma de Ejecución Financiera.

**RF02.014 Descargar contrato de la aplicación.**

El sistema permitirá descargar el contrato de la aplicación y almacenarlo en el directorio especificado por el usuario.

**RF02.015 Listar contratos para firmar.**

El sistema permitirá listar todos los contratos que han sido aceptados por las Secretarías Técnicas como resultado de la búsqueda realizada, incluyendo por contrato: su Nombre, Ministerio Contraparte, Ente Contraparte, Monto del contrato y Estado.

**RF02.016 Mostrar datos de contrato.**

El sistema permitirá mostrar los datos generales del contrato seleccionado, los mismos son:

- Nombre del contrato
- Ministerio(s) por Venezuela
- Ministerio (s) por Cuba
- Ente Ejecutor(s) por Venezuela
- Ente Ejecutor(s) por Cuba

Y de los proyectos incluidos se mostrará:

- Nombre del proyecto
- Monto de cada proyecto
- Cantidad a invertir en Venezuela
- Cantidad a transferir a Cuba
- Modalidad del proyecto

**RF02.017 Mostrar datos de proyecto.**

El sistema permitirá mostrar datos del proyecto seleccionado.

**RF02.018 Adjuntar documento.**

El sistema permitirá adjuntar un documento.

**RF02.019 Listar cronogramas de Ejecución Financiera.**

El sistema permitirá listar todos los cronogramas como resultado de la búsqueda realizada, incluyendo por cronograma su Nombre, Ministerio Contraparte, Ente Contraparte, Monto y Estado del cronograma de Ejecución Financiera.

**RF02.020 Aceptar modificación de contrato.**

El sistema permitirá aceptar la modificación de contrato.

**RF02.021 Rechazar modificación de contrato.**

El sistema permitirá rechazar la modificación de contrato.

**RF02.022 Aceptar eliminación de contrato.**

El sistema permitirá aceptar la eliminación de contrato.

**RF02.023 Rechazar eliminación de contrato.**

El sistema permitirá rechazar la eliminación de contrato.

**RF02.024 Aceptar envío de contrato.**

El sistema permitirá aceptar el envío de contrato.

**RF02.025 Rechazar envío de contrato.**

El sistema permitirá rechazar el envío de contrato.

**RF02.026 Realizar búsqueda de cronograma de Ejecución Financiera.**

El sistema permitirá realizar una búsqueda de cronogramas de Ejecución Financiera.

**RF02.027 Mostrar datos de cronograma de Ejecución Financiera.**

El sistema permitirá mostrar los datos de cronograma de Ejecución Financiera.

**RF02.028 Eliminar actividad datos de una actividad del cronograma de Ejecución Financiera.**

El sistema permitirá eliminar la distribución del monto de una actividad.

**RF02.029 Agregar nuevos datos de una actividad.**

El sistema permitirá agregar una nueva distribución del monto asignado a una actividad.

**RF02.030 Generar reporte de proyectos contratados.**

El sistema permitirá generar un reporte que muestre los proyectos contratados.

**RF02.031 Generar reportes de proyectos no contratados.**

El sistema permitirá generar un reporte que muestre los proyectos no contratados.

**RF02.032 Realizar búsqueda de proyectos contratados.**

El sistema permitirá realizar una búsqueda de proyectos que ya están contratados.

**RF02.033 Realizar búsqueda de proyectos no contratados.**

El sistema permitirá realizar una búsqueda de proyectos que no están contratados.

### *2.4.2. Requisitos no funcionales*

Para brindar una organización de los requisitos no funcionales que se especifican en este documento se ha empleado la clasificación por requisitos de calidad. A continuación mostramos las categorías en que agrupamos los requisitos no funcionales.

- **Usabilidad:** Requisititos que determinan las características generales de la capa de presentación del sistema en cuanto a las características de diseño gráfico de la misma, además de las facilidades para que el uso del sistema por parte del usuario final.

- **Fiabilidad:** Estos requerimientos están relacionados con la capacidad del usuario para confiar en las respuestas del sistema, en un sentido técnico, es decir, que la funcionalidad del sistema no se vea afectada por factores ajenos al sistema como los son los factores técnicos.

- **Eficiencia:** Los requerimientos clasificados en esta categoría están relacionados con tiempos de respuesta estimados, requeridos y esperados para la ejecución en línea de procesos del sistema, teniendo como base la plataforma tecnológica y escenarios específicos a los que en teoría el sistema estará expuesto y frente a los que deberá responder.

- **Seguridad:** Requerimientos relacionados con la confidencialidad de los datos en la transmisión y en el almacenamiento, junto con las necesidades del sistema para evitar intrusiones no autorizadas al mismo y la capacidad para seguir eventos que comprometan esta seguridad a través del tiempo.

- **Portabilidad:** Estos requerimientos describen la capacidad del sistema para migrar de una plataforma hardware a otra sin que esto represente mayores traumatismos para el cliente del mismo, teniendo en cuenta los requisitos técnicos presentados y las generalidades naturales de configuración del sistema.

- **Reusabilidad:** Estos requerimientos consideran la capacidad de los componentes del sistema de prestar servicios a otros sistemas, de tal manera que el componente como valor adicional al cumplimiento de sus funciones pueda prestar sus servicios a otros sistemas sin que esto implique modificaciones o redefiniciones en dicho componente.

- **Capacidad:** Consideran los requerimientos de tecnología que permitirán al sistema ejecutar sus funciones de manera que responda a las expectativas del usuario en términos de eficiencia y eficacia en la respuesta de las operaciones. Se tienen en cuenta también las características, que harán que sea posible que el sistema funcione de manera estable durante un tiempo determinado, planeando el crecimiento del mismo.

### *2.5. Actores del sistema*

La siguiente tabla muestra los actores que están representados en el Módulo contratación del proyecto CCV y sus características.

| Actor | Descripción |
|---|---|
| **Gestor de Contrato** | El actor es el que accede al sistema, se autentica y se le da un determinado nivel de acceso según su rol. Este actor puede, modificar un contrato. También puede aceptar o rechazar una modificación que se haya realizado sobre un contrato. |
| **Coordinador de EE** | El actor es el encargado de crear los contratos, gestionar cronograma de EF y se comporta como Gestor de Contrato y Coordinador**,** heredando todos sus permisos para ejecutar las funcionalidades a la cual puede acceder el actor, puede enviar los contratos concebidos a los Ministerios para que estos sean revisados, además del cronograma de EF modificado. |
| **Coordinador de M** | El actor puede comportarse como Gestor de Contrato y Coordinador heredando todos los permisos para ejecutar las funcionalidades a la cual puede acceder el actor, además puede Aceptar o Rechazar los contratos que son enviados a los Ministerios, enviar contratos a ST y cronograma de EF. |

| | |
|---|---|
| **Coordinador de ST** | El actor puede comportarse como Coordinador heredando todos los permisos para ejecutar las funcionalidades a la cual puede acceder el actor, además puede Aceptar o Rechazar el cronograma de EF enviado por los Ministerios. |
| **Coordinador** | Es el actor que accede al sistema, se autentica y se la da un determinado nivel de acceso según su rol. Este actor puede, Aceptar o Rechazar un contrato. |

### *2.6. Diagrama de casos de uso*

Un diagrama de casos de uso (*Use Case Diagrama*) es una representación gráfica de parte o el total de los actores y casos de uso del sistema, incluyendo sus interacciones. Todo sistema tiene como mínimo un Diagram *Main Use Case*, que es una representación gráfica del entorno del sistema (actores) y su funcionalidad principal (casos de uso).

Un diagrama de casos de uso muestra, por tanto, los distintos requisitos funcionales que se esperan de una aplicación o sistema y cómo se relaciona con su entorno (usuarios u otras aplicaciones). Un caso de uso, denotando un requisito funcional exigido al sistema, se representa en el diagrama por una elipse y un nombre significativo. Entre los elementos de un diagrama de casos de uso se pueden presentar tres tipos de relaciones, representadas por líneas dirigidas o no entre ellos:

- ``comunica'' (<<communicates>>): Relación (asociación) entre un actor y un caso de uso que denota la participación del actor en dicho caso de uso.
- ``usa'' (<<uses>>) (o <<include>> en la nueva versión de UML): Relación de dependencia entre dos casos de uso que denota la inclusión del comportamiento de un escenario en otro.
- ``extiende'' (<<extends>>): Relación de dependencia entre dos casos de uso que denota que un caso de uso es una especialización de otro.

A continuaciónse muestra el diagrama de Casos de Uso del sistema del Módulo Contrataciónde l proyecto CCV. Para le implementación del mismo, se tuvieron en cuenta los patrones CRUD y Multiples Roles(Multiples Actores).

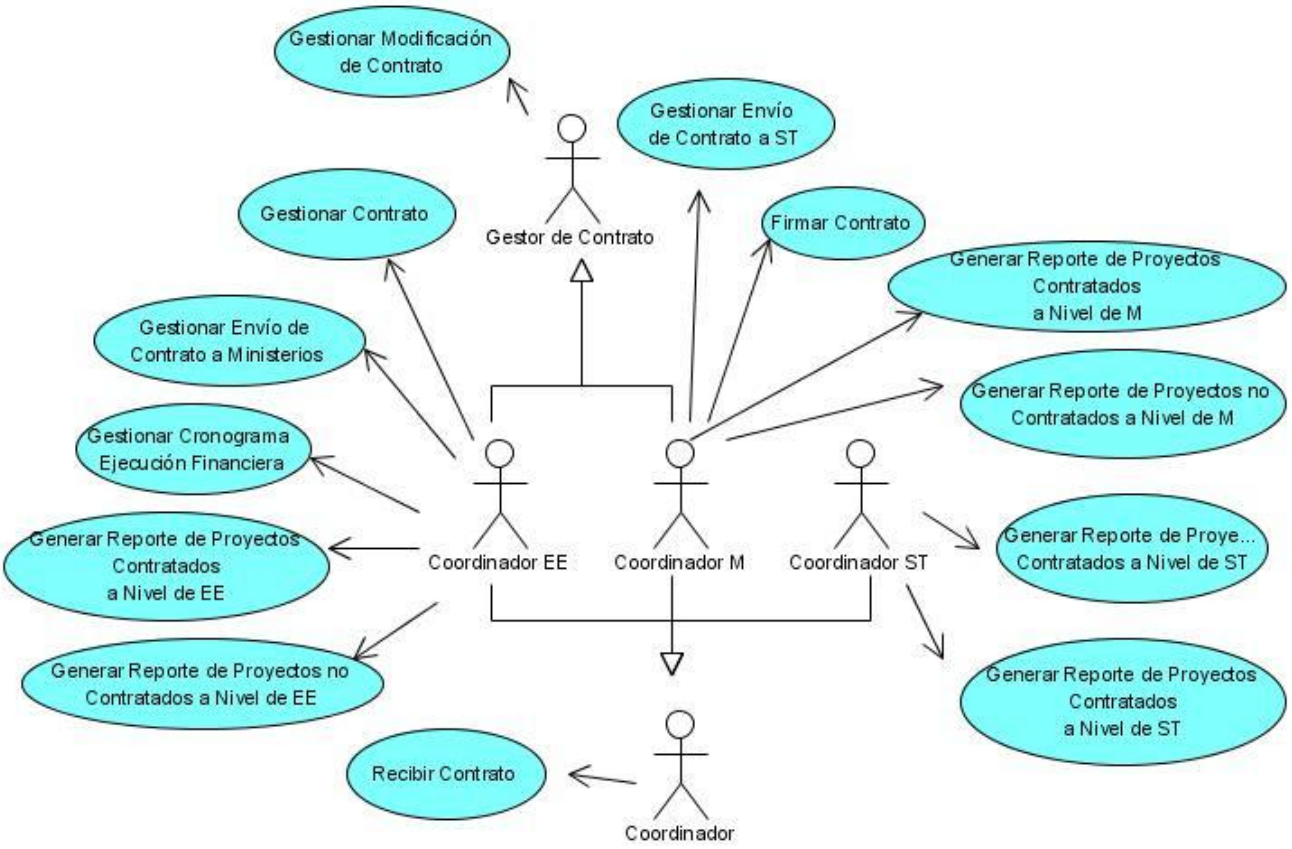



***Figura 13.*** Diagrama de Casos de Uso, Módulo Contratación.

### *2.7. Descripción de los casos de uso*

Una vez definido los casos de uso del sistema, se procede a realizar la descripción textual de los mismos con la objetivo de especificar en detalles cada una de las funcionalidades que deben ser implementadas. La descripción de los casos de uso constituye una guía para los desarrolladores y un documento de obligatorio cumplimiento en cuanto al desarrollo de las funcionalidades del sistema. Se muestra a continuación, la descripción de uno de los principales casos de uso del módulo Contratación. Para acceder a la descripción íntegra de todos los Casos de Uso, ver el documento Modelo de casos de uso del sistema CCV, Módulo Contratación [24].

| **Caso de Uso:** | Gestionar Contrato. |
|---|---|
| **Actores:** | Coordinador EE. |
| **Resumen:** | El caso de uso inicia cuando el Coordinador EE crea una propuesta de contrato que será presentada al ente contraparte, esta propuesta podrá ser eliminada si está pendiente de aprobación, si es rechazada la propuesta. En caso de que este aprobado por los EE, la eliminación será propuesta por uno de ellos y deberá ser |

| | |
|---|---|
| | aceptada o rechazada por el ente contraparte. |
| **Precondiciones:** | El sistema debe estar instalado y ejecutándose correctamente. El actor debe estar autenticado con los permisos necesarios. |
| **Referencias** | RF 02.001, RF 02.005, RF 02.023, RF 02.024, RF 02.008, RF 02.009, RF 02.007, RF 02.016, RF 02.018 |
| **Prioridad** | Crítico |
| **Nivel** | Usuario |

**Flujo Normal de Eventos**

**Sección "Elaborar Propuesta de Contrato"**

| Acción del Actor | Respuesta del Sistema |
|---|---|
| 1. El actor solicita crear una propuesta de contrato. | 2. El sistema muestra una interfaz que permite la elaboración de un nuevo contrato. |
| 3. El actor accede a la opción de crear un nuevo contrato. | 4. El sistema muestra una interfaz que permite incluir proyectos en un contrato, realizando una búsqueda por:<br>• Nombre de proyecto<br>• Ministerio Contraparte<br>• EE contraparte |
| 5. El actor solicita la búsqueda de proyectos especificando un criterio de búsqueda. | 6.El sistema realiza la búsqueda por el criterio especificado y muestra una lista de proyectos existentes como resultado de la búsqueda realizada, donde se visualizan los siguientes datos:<br>• Nombre del Proyecto.<br>• Monto.<br>• Invertir en Venezuela.<br>• Transferir a Cuba.<br>• Modalidad. |
| 7. El actor selecciona los proyectos a incluir | 8. El sistema muestra los proyectos incluidos en una |

| | |
|---|---|
| en el contrato. | nueva tabla, donde se visualizan los siguientes datos:<br>• Nombre del Proyecto.<br>• Monto.<br>• Invertir en Venezuela.<br>• Transferir a Cuba.<br>• Modalidad.<br>• Monto total. |
| 9. El actor solicita la búsqueda del documento de contrato firmado. | 10. El sistema realiza una búsqueda del documento, en los diferentes directorios de la pc. |
| 11. El actor selecciona el documento. | 12. El sistema adjunta el documento del contrato firmado. |
| 13. El usuario acepta la creación del contrato. | 14. Envía la propuesta al ente contraparte, terminando así el caso de uso. |
| **Prototipo de Interfaz** | |

Sistema Convenio Cuba - Venezuela

# Banner

**Procesos**

Concebir Contrato

Modificar Cronograma de Ejecuciòn Financiera

**Documentos**

Contrato

**Reportes**

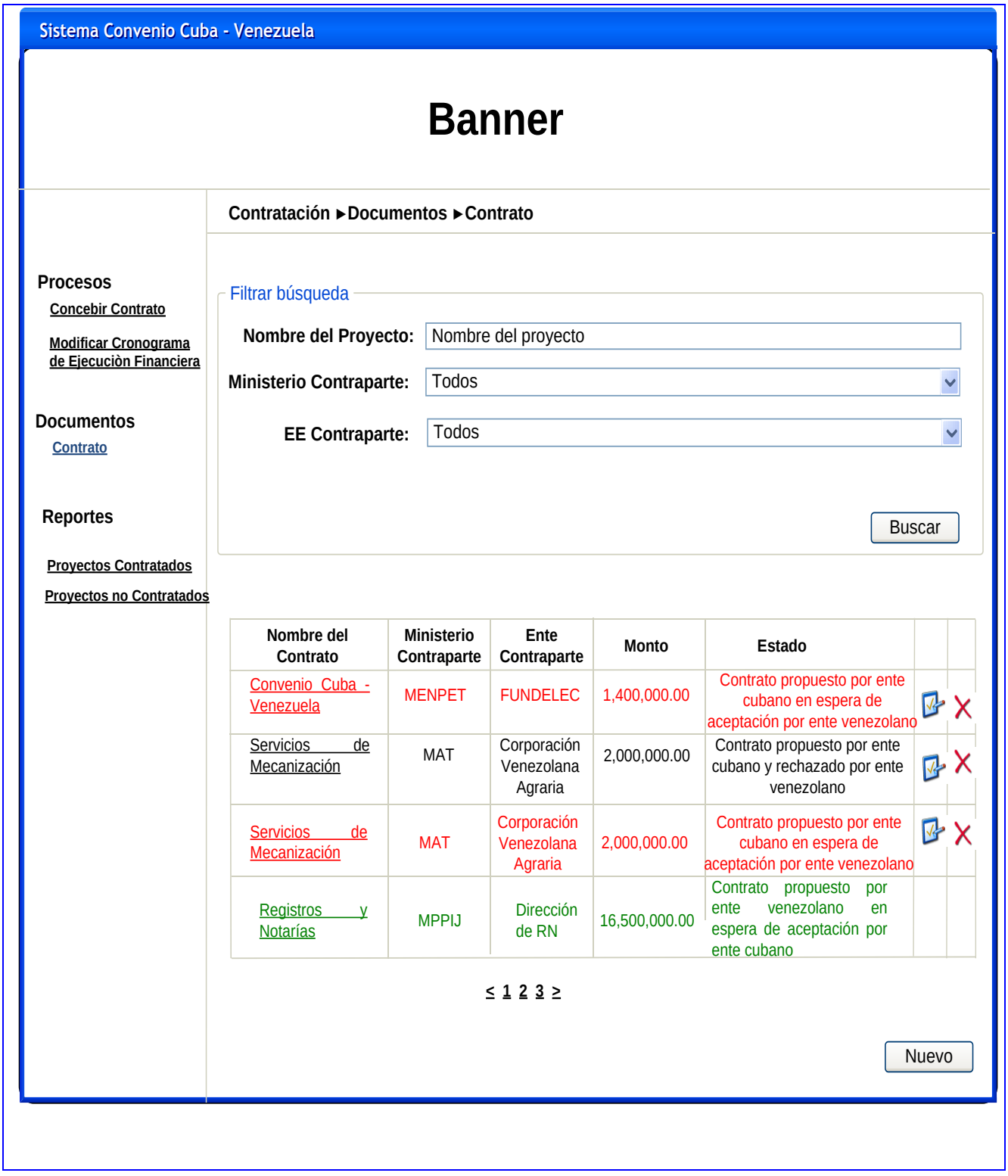
Proyectos Contratados

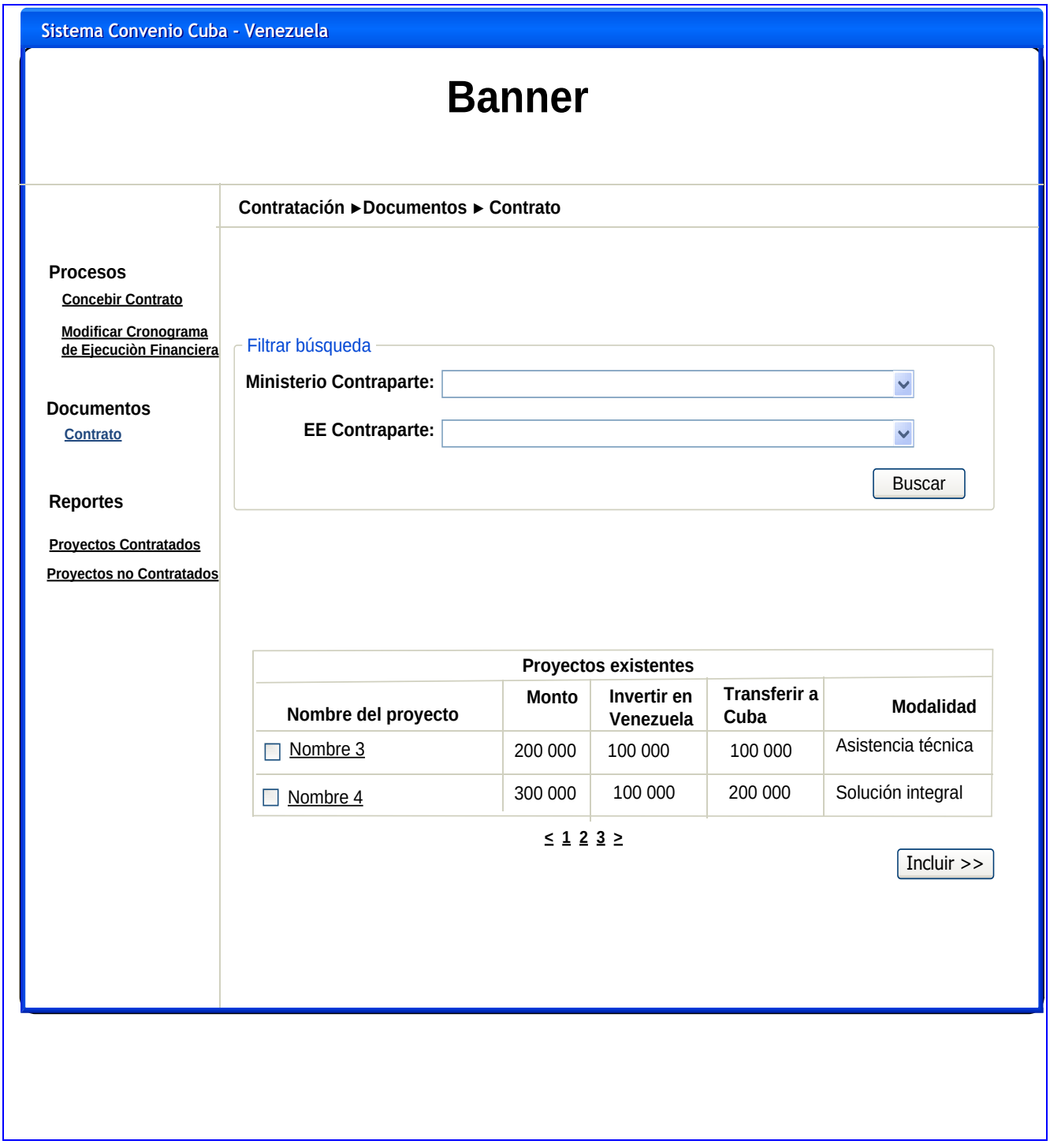
Proyectos no Contratados

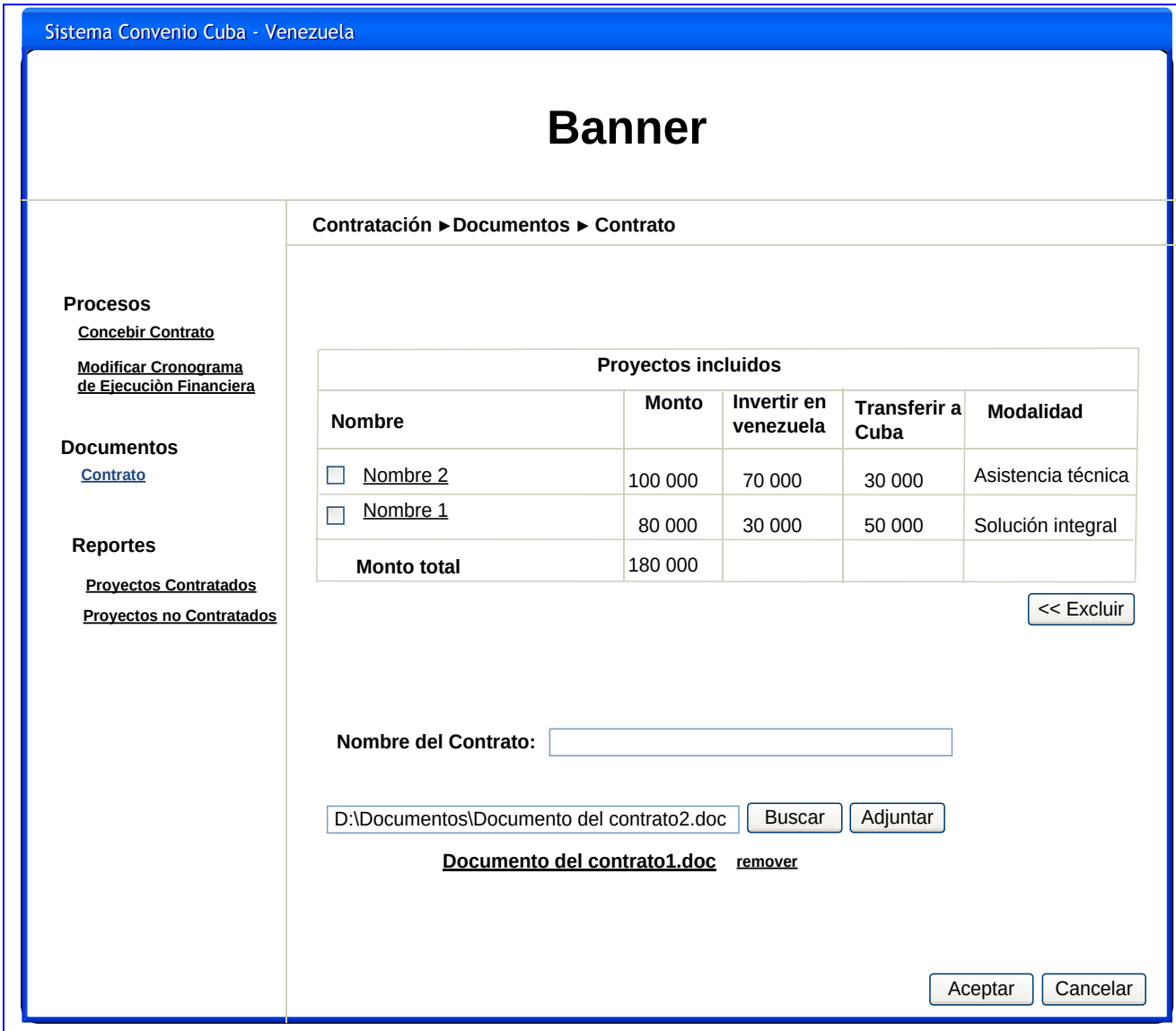
**Contratación**

**Aquí va una descripción del módulo, además de la especificación de las clausulas que no pueden ser modificadas en el contrato.**

**Preforma del contrato.doc** Descargar

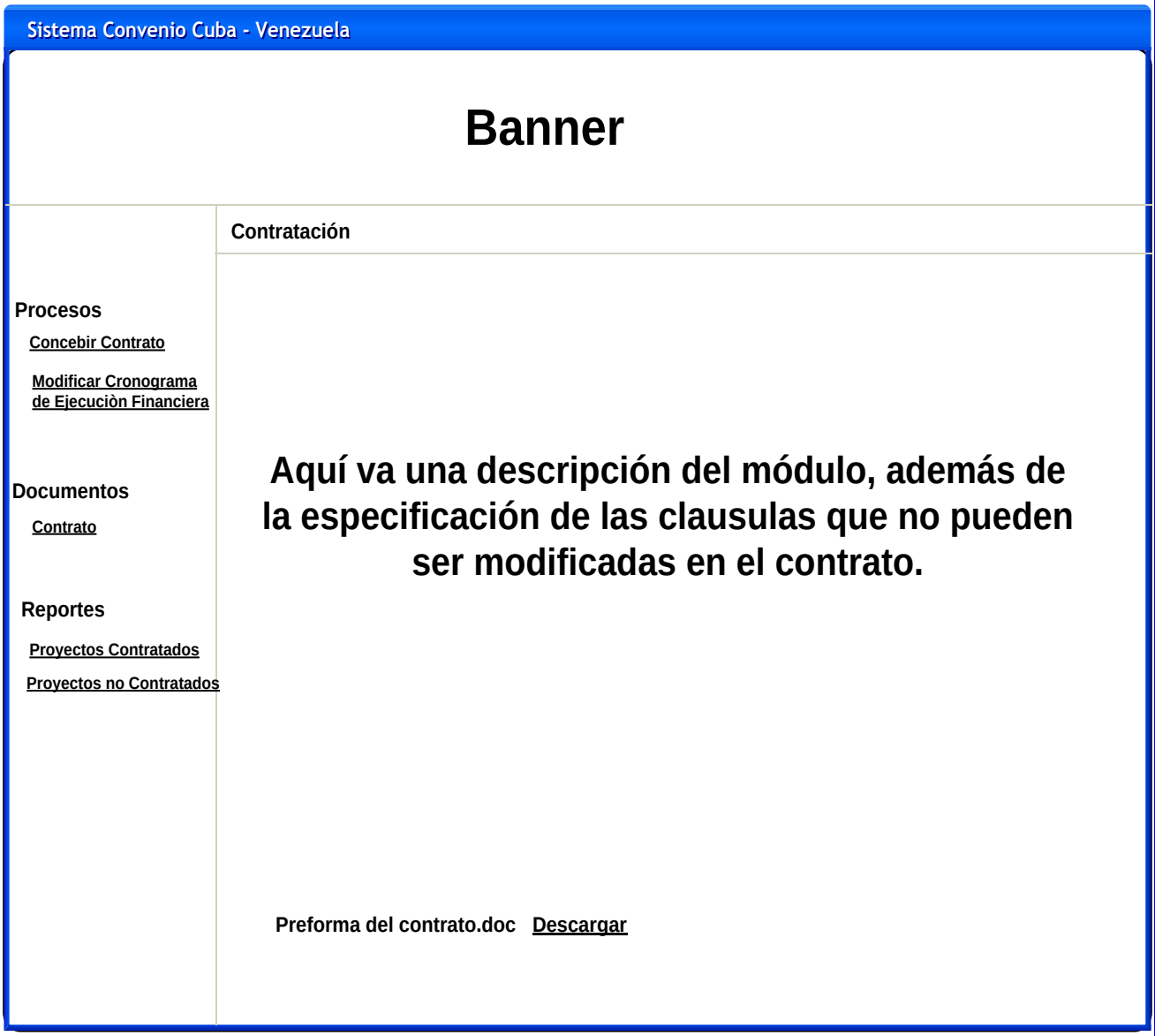

Sistema Convenio Cuba - Venezuela
Banner
Contratación ▸Documentos ▸Contrato
Procesos
Concebir Contrato
Modificar Cronograma de Ejecuciòn Financiera
Documentos
Contrato
Reportes
Proyectos Contratados
Proyectos no Contratados
Filtrar búsqueda
Nombre del Proyecto:
Nombre del proyecto
Ministerio Contraparte:
Todos
EE Contraparte:
Todos
Buscar
Nombre del Contrato
Ministerio Contraparte
Ente Contraparte
Monto
Estado
Convenio Cuba - Venezuela
MENPET
FUNDELEC
1,400,000.00
Contrato propuesto por ente cubano en espera de aceptación por ente venezolano
Servicios de Mecanización
MAT
Corporación Venezolana Agraria
2,000,000.00
Contrato propuesto por ente cubano y rechazado por ente venezolano
Servicios de Mecanización
MAT
Corporación Venezolana Agraria
2,000,000.00
Contrato propuesto por ente cubano en espera de aceptación por ente venezolano
Registros y Notarías
MPPIJ
Dirección de RN
16,500,000.00
Contrato propuesto por ente venezolano en espera de aceptación por ente cubano
≤ 1 2 3 ≥
Nuevo

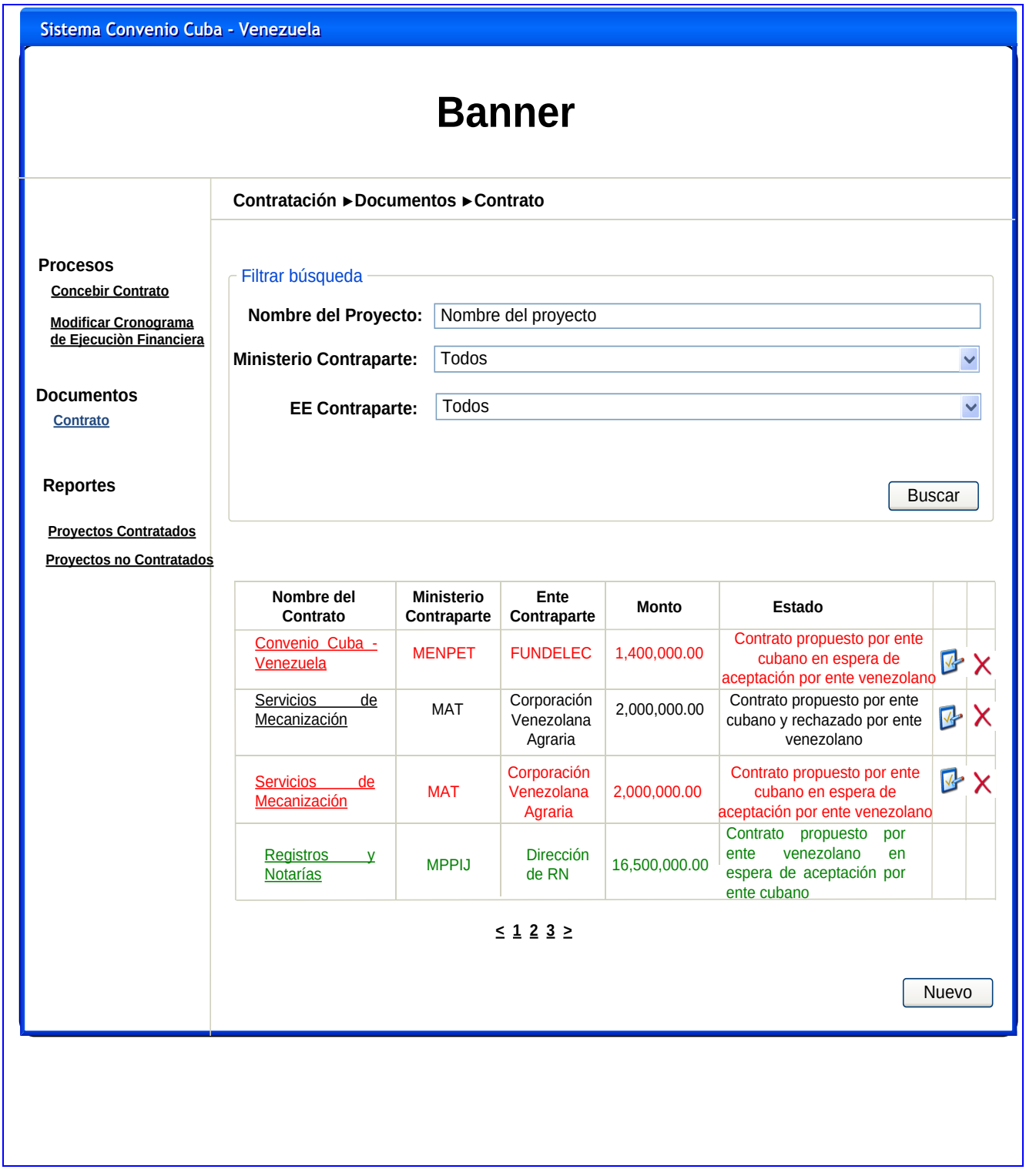

Sistema Convenio Cuba - Venezuela
Banner
Contratación ►Documentos ► Contrato
Procesos
Concebir Contrato
Modificar Cronograma de Ejecuciòn Financiera
Documentos
Contrato
Reportes
Proyectos Contratados
Proyectos no Contratados
Filtrar búsqueda
Ministerio Contraparte:
EE Contraparte:
Buscar
Proyectos existentes
Nombre del proyecto
Monto
Invertir en Venezuela
Transferir a Cuba
Modalidad
Nombre 3
200 000
100 000
100 000
Asistencia técnica
Nombre 4
300 000
100 000
200 000
Solución integral
≤ 1 2 3 ≥
Incluir >>

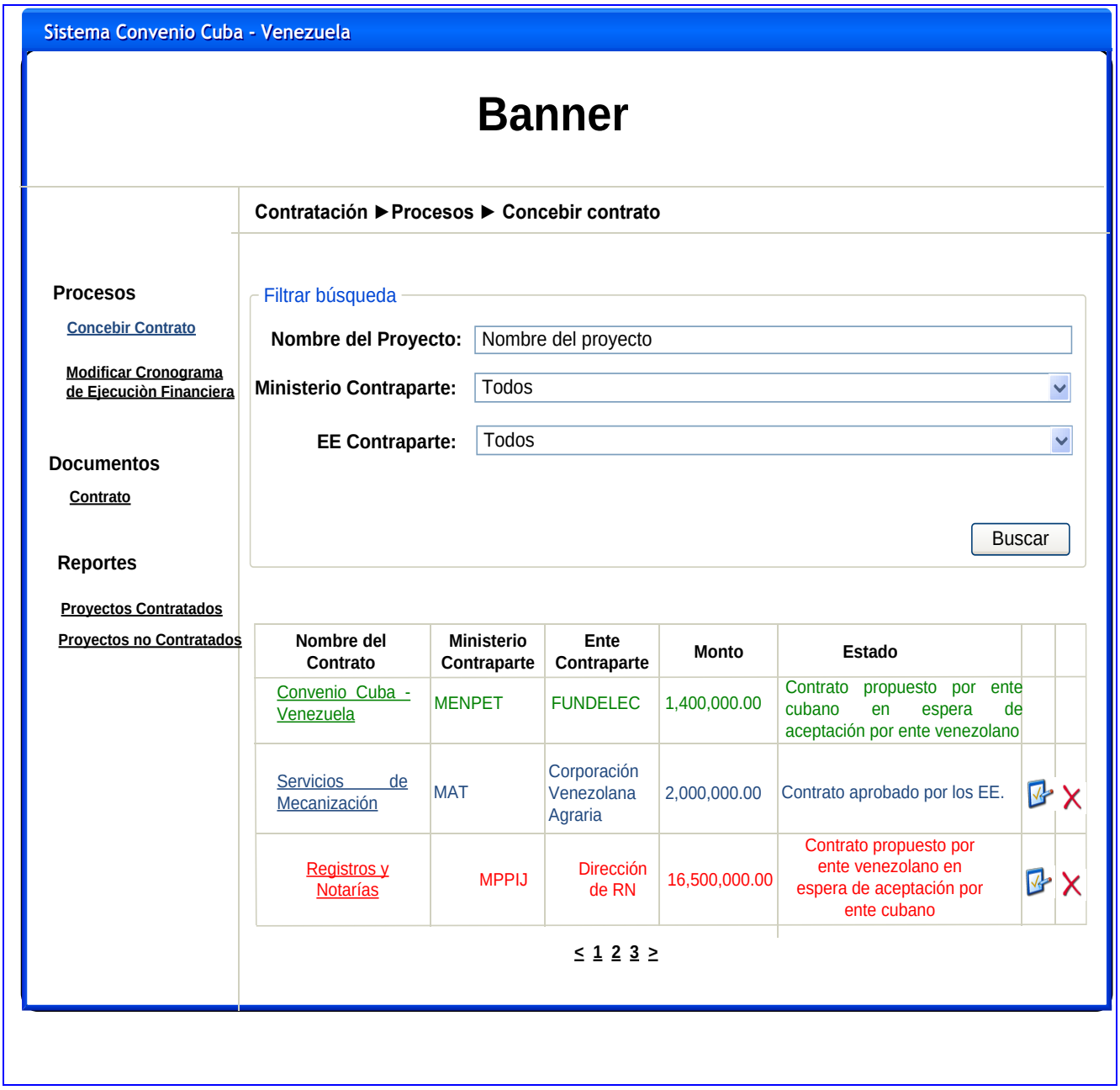


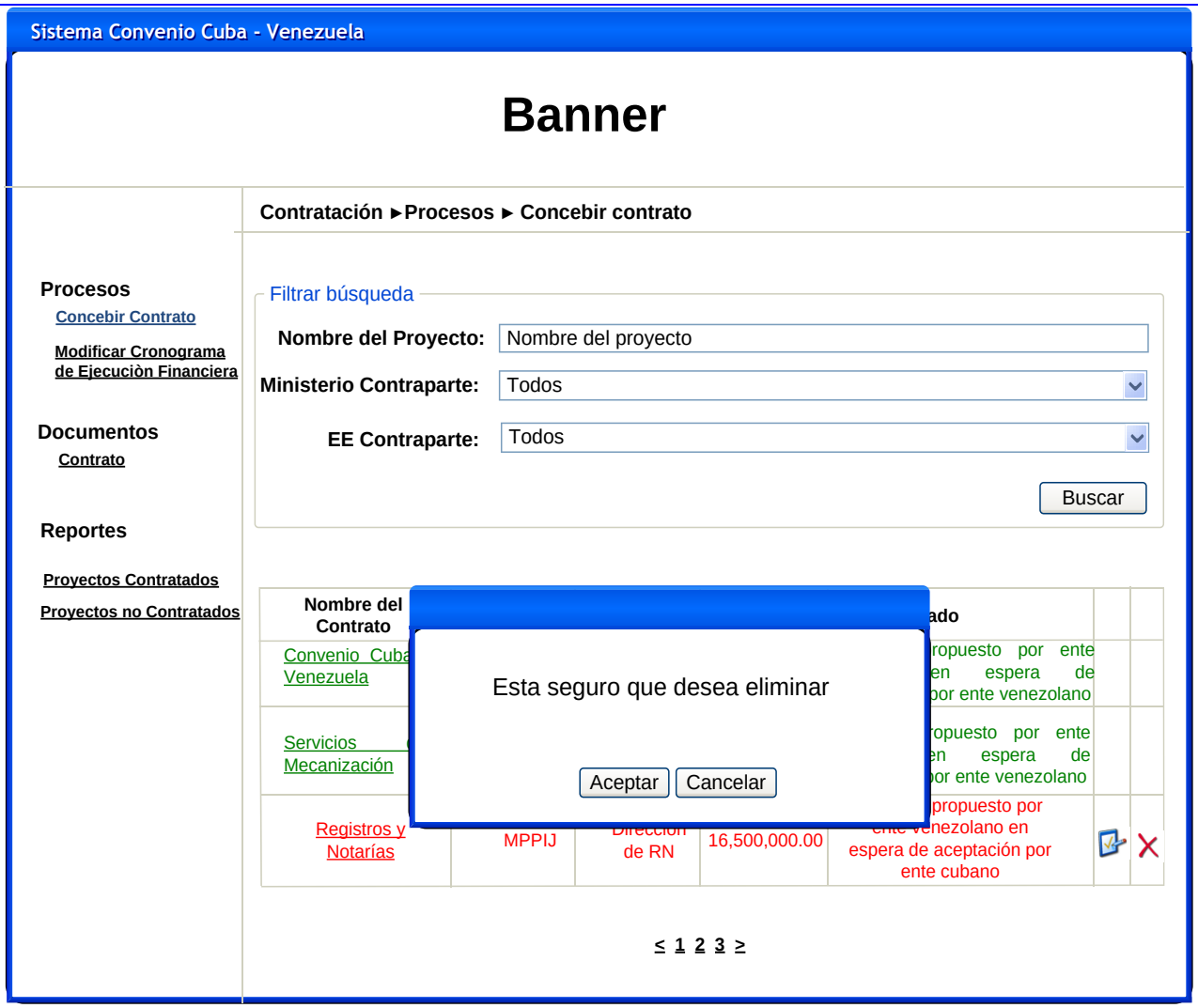

| Flujos Alternos | |
|---|---|
| **Acción del Actor** | **Respuesta del Sistema** |
| 13.1 El actor cancela la creación del contrato. | 13.2. El sistema vuelve a la interfaz que permite crear el contrato . |
| **Flujos alternos para Datos Incompletos** | |
| **Acción del Actor** | **Respuesta del Sistema** |
| 9.1 El actor no solicita adjuntar el documento del contrato firmado. | 9.2 El sistema muestra un mensaje especificando que debe adjuntar el documento. |
| **Sección “Eliminar Propuesta de Contrato”** | |

| Acción del Actor | Respuesta del Sistema |
|---|---|
| 1. El actor solicita eliminar un contrato. | 2. El sistema muestra una interfaz que permite realizar una búsqueda de contratos por:<br>• Nombre del proyecto<br>• Ministerio Contraparte<br>• EE Contraparte |
| 3. El actor solicita realizar una búsqueda especificando un criterio. | 4. El sistema realiza la búsqueda por el criterio especificado y muestra una lista de contratos como resultado de la búsqueda realizada, donde se visualizan los siguientes datos:<br>• Nombre del Contrato<br>• Ministerio Contraparte<br>• Ente Contraparte<br>• Estado (El contrato se muestra en el color correspondiente a los estados definidos. Ver Documento de Estados de Contratos) |
| 5. El actor selecciona el contrato a eliminar. | 6. El sistema muestra un mensaje para confirmar la eliminación. |
| 7. El actor acepta la opción de eliminar el contrato. | 8. El sistema elimina el contrato. |
| **Prototipo de Interfaz** | |


Sistema Convenio Cuba - Venezuela
Banner
Contratación
Procesos
Concebir Contrato
Modificar Cronograma de Ejecuciòn Financiera
Documentos
Contrato
Reportes
Proyectos Contratados
Proyectos no Contratados
Aquí va una descripción del módulo, además de la especificación de las clausulas que no pueden ser modificadas en el contrato.
Preforma del contrato.doc Descargar


Sistema Convenio Cuba - Venezuela
Banner
Contratación ▸Documentos ▸Contrato
Procesos
Concebir Contrato
Modificar Cronograma de Ejecuciòn Financiera
Documentos
Contrato
Reportes
Proyectos Contratados
Proyectos no Contratados
Filtrar búsqueda
Nombre del Proyecto:
Nombre del proyecto
Ministerio Contraparte:
Todos
EE Contraparte:
Todos
Buscar
Nombre del Contrato
Ministerio Contraparte
Ente Contraparte
Monto
Estado
Convenio Cuba - Venezuela
MENPET
FUNDELEC
1,400,000.00
Contrato propuesto por ente cubano en espera de aceptación por ente venezolano
Servicios de Mecanización
MAT
Corporación Venezolana Agraria
2,000,000.00
Contrato propuesto por ente cubano y rechazado por ente venezolano
Servicios de Mecanización
MAT
Corporación Venezolana Agraria
2,000,000.00
Contrato propuesto por ente cubano en espera de aceptación por ente venezolano
Registros y Notarías
MPPIJ
Dirección de RN
16,500,000.00
Contrato propuesto por ente venezolano en espera de aceptación por ente cubano
≤ 1 2 3 ≥
Nuevo

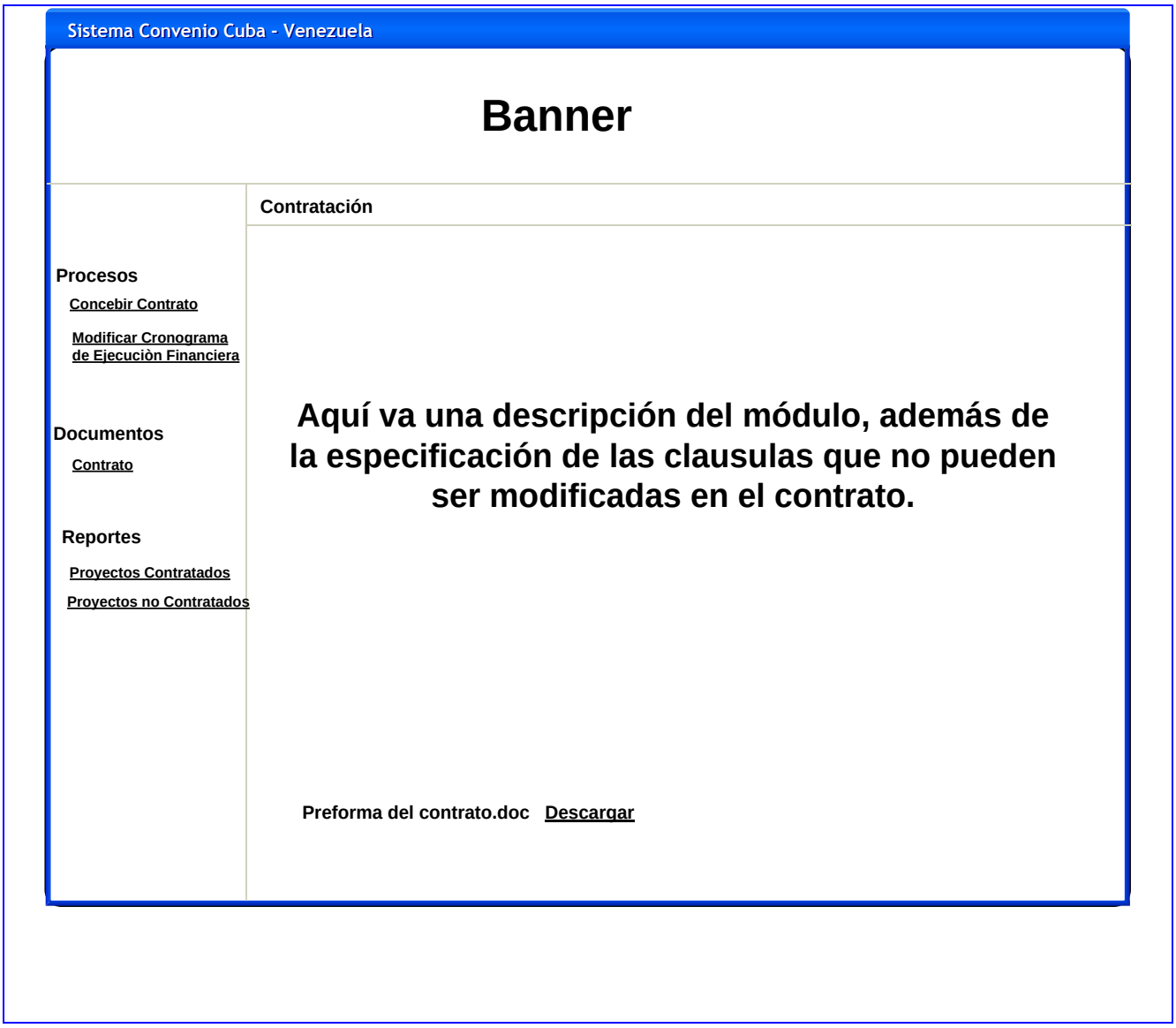

Sistema Convenio Cuba - Venezuela
Banner
Contratación ▶Procesos ▶ Concebir contrato
Procesos
Concebir Contrato
Modificar Cronograma de Ejecuciòn Financiera
Documentos
Contrato
Reportes
Proyectos Contratados
Proyectos no Contratados
Filtrar búsqueda
Nombre del Proyecto:
Nombre del proyecto
Ministerio Contraparte:
Todos
EE Contraparte:
Todos
Buscar
Nombre del Contrato
Ministerio Contraparte
Ente Contraparte
Monto
Estado
Convenio Cuba - Venezuela
MENPET
FUNDELEC
1,400,000.00
Contrato propuesto por ente cubano en espera de aceptación por ente venezolano
Servicios de Mecanización
MAT
Corporación Venezolana Agraria
2,000,000.00
Contrato aprobado por los EE.
Registros y Notarías
MPPIJ
Dirección de RN
16,500,000.00
Contrato propuesto por ente venezolano en espera de aceptación por ente cubano
≤ 1 2 3 ≥

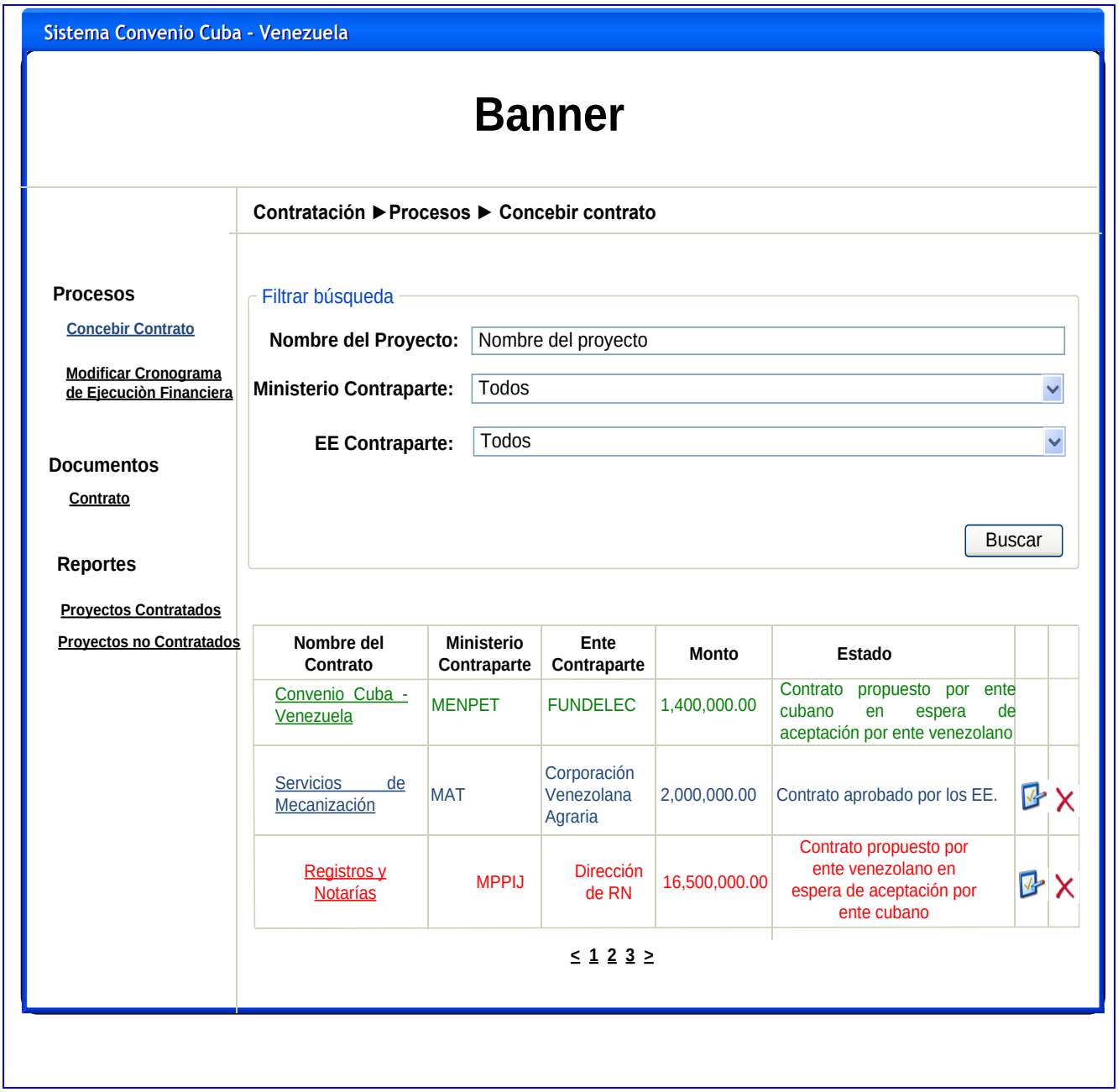



| Flujos Alternos | |
|---|---|
| **Acción del Actor** | **Respuesta del Sistema** |
| 7.1. El actor cancela la acción de eliminar contrato. | 7.2 El sistema regresa a la interfaz donde se muestran los contratos. |

| Sección “Aceptar Eliminación de Contrato” | |
|---|---|
| **Acción del Actor** | **Respuesta del Sistema** |
| 1. El actor solicita revisar un contrato. | 2. El sistema muestra una interfaz que permite realizar una búsqueda de contratos por:<br>• Nombre del proyecto<br>• Miinisterio Contraparte<br>• EE Contraparte |

| | |
|---|---|
| 3. El actor solicita realizar la búsqueda especificando un criterio. | 4. El sistema realiza la búsqueda por el criterio especificado y muestra una lista de contratos como resultado de la búsqueda realizada, donde se visualizan los siguientes datos:<br>• Nombre del Contrato<br>• Ministerio Contraparte<br>• Ente Contraparte<br>• Estado (El contrato se muestra en el color correspondiente a los estados definidos. Ver Documento de Estados de Contratos) |
| 5. El actor selecciona el contrato a revisar. | 6. El sistema muestra los datos del contrato. |
| 7. El actor acepta la eliminación del contrato. | 8. El sistema elimina el contrato y envía una notificación al ente contraparte. |
| **Prototipo de Interfaz** | |

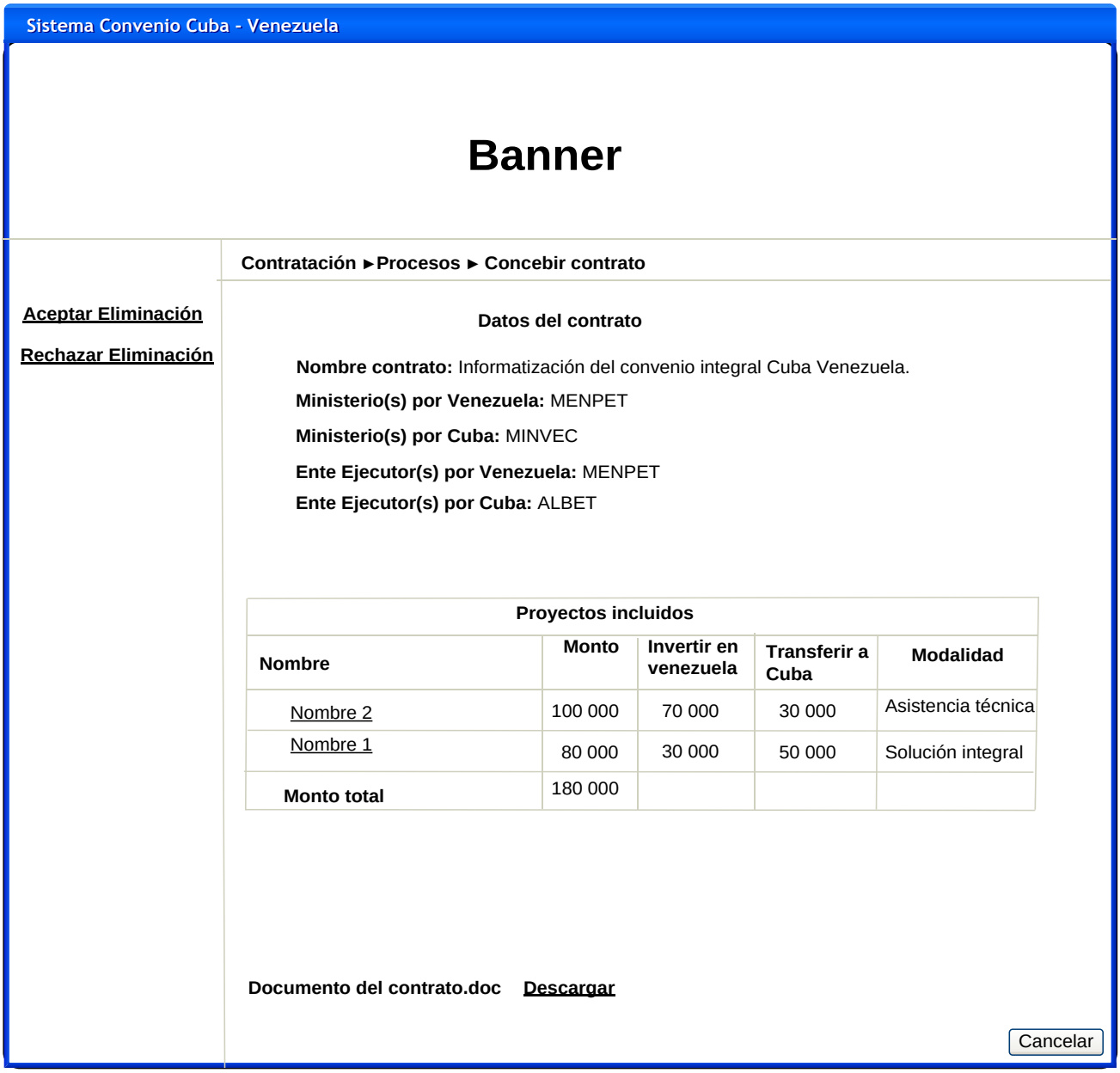

Sistema Convenio Cuba - Venezuela
Banner
Contratación
Procesos
Concebir Contrato
Modificar Cronograma de Ejecuciòn Financiera
Documentos
Contrato
Reportes
Proyectos Contratados
Proyectos no Contratados
Aquí va una descripción del módulo, además de la especificación de las clausulas que no pueden ser modificadas en el contrato.
Preforma del contrato.doc Descargar


Sistema Convenio Cuba - Venezuela
Banner
Contratación ▶Procesos ▶ Concebir contrato
Procesos
Concebir Contrato
Modificar Cronograma de Ejecuciòn Financiera
Documentos
Contrato
Reportes
Proyectos Contratados
Proyectos no Contratados
Filtrar búsqueda
Nombre del Proyecto:
Nombre del proyecto
Ministerio Contraparte:
Todos
EE Contraparte:
Todos
Buscar
Nombre del Contrato
Ministerio Contraparte
Ente Contraparte
Monto
Estado
Convenio Cuba - Venezuela
MENPET
FUNDELEC
1,400,000.00
Contrato propuesto por ente cubano en espera de aceptación por ente venezolano
Servicios de Mecanización
MAT
Corporación Venezolana Agraria
2,000,000.00
Contrato aprobado por los EE.
Registros y Notarías
MPPIJ
Dirección de RN
16,500,000.00
Contrato propuesto por ente venezolano en espera de aceptación por ente cubano
≤ 1 2 3 ≥


| Flujos Alternos | |
|---|---|
| **Acción del Actor** | **Respuesta del Sistema** |
| 7.1 El actor rechaza la eliminación del contrato. | 7.2 El sistema muestra una interfaz que permite especificar el motivo del rechazo. |

| | |
|---|---|
| 7.3 El actor redacta la nota de rechazo y acepta. | 7.4 El sistema envía una notificación a todos los implicados, además de la propuesta de eliminación rechazado con una nota incluida especificando el motivo. |

**Prototipo de interfaz**




| Flujos Alternos | |
|---|---|
| **Flujo alterno a los pasos 7 y 7.1** | |
| **Acción del Actor** | **Respuesta del Sistema** |
| 7.1.1 El actor cancela la opción de aceptar o rechazar contrato. | 7.1.2 El sistema vuelve a la interfaz donde se muestra el listado de contratos. |
| **Poscondiciones** | 1. El contrato debe quedar elaborado.<br>2. El proyecto debe quedar eliminado.<br>3. El proyecto debe quedar aceptado o rechazado. |

## *2.8. Diseño*

La utilización del diseño es uno de los pilares fundamentales en la ingeniería del software. Es una de las actividades técnicas necesarias para la elaboración del software. Entre las tareas fundamentales del diseño están: producir el diseño de datos, diseño arquitectónico, diseño de interfaz y diseño de componentes.Cuando se diseñan Sistemas Informáticos es indispensable hacerlo de forma correcta. El diseño propuesto tiene que cumplir a cabalidad con los requerimientos del sistema. De esta forma nos aseguraremos de convertir exactamente, los requisitos de un cliente en un producto o sistema de software al concluir la construcción del mismo; debe ser capaz de facilitar las mejoras del software, debe especificarse, de forma tal que sea entendible por otros diseñadores y no diseñadores, servir como guía para los demás flujos de la ingeniería de software, permitir la comprobación del sistema fácilmente [20].Roger Pressman en su libro:"Ingeniería del Software. Un enfoque práctico" plantea una serie de principios básicos que se tienen que tener en cuenta en el momento de diseñar, ellos garantizan el correcto funcionamiento del sistema y un buen uso de las técnicas de diseño [20].

- El diseño deberá poderse rastrear hasta el modelo de análisis.
- El diseño no deberá crear nada que ya fue creado.
- El diseño deberá presentar uniformidad e integración.
- El diseño deberá estructurarse para admitir cambios.
- El diseño deberá estructurarse para degradarse poco a poco.
- El diseño no es escribir código y viceversa.
- El diseño deberá ser evaluado continuamente en función de su calidad durante su realización, no después de terminado.
- El diseño deberá ser revisado para minimizar errores conceptuales.

Cuando estos principios son aplicados correctamente, el diseño creado muestra la calidad requerida. Es necesario verificar esta calidad por factores externos e internos. Los factores externos son aquellas propiedades del software que se observan fácilmente: velocidad, fiabilidad, grado de corrección, usabilidad. Los internos son importantes para los ingenieros y personal encargado de realizar el proyecto.

Entre los patrones de diseño utilizados, se encuentran los patrones Bridge y el patrón DAO para la modelación del acceso a datos.

### *2.8.1. Diagrama de Clases del Diseño*

CCV, por sus fines de uso, se definió que debía ser una aplicación web, por tal razón, en los diagramas de clases de diseño elaborados se hicieron uso de los estereotipos web, con el objetivo de ilustrar la colaboración real y la representación de todos los elementos que interactúan en la ejecución de las funcionalidades que debe brindar el sistema.

Se muestra a continuación, 3 de los diagramas de clases de diseño de los principales casos de uso, pertenecientes al Módulo Contratación del proyecto CCV, los demás diagramas de clases de diseño pueden ser consultados en el documento Modelo de Diseño del Módulo Contratación del proyecto CCV [25].

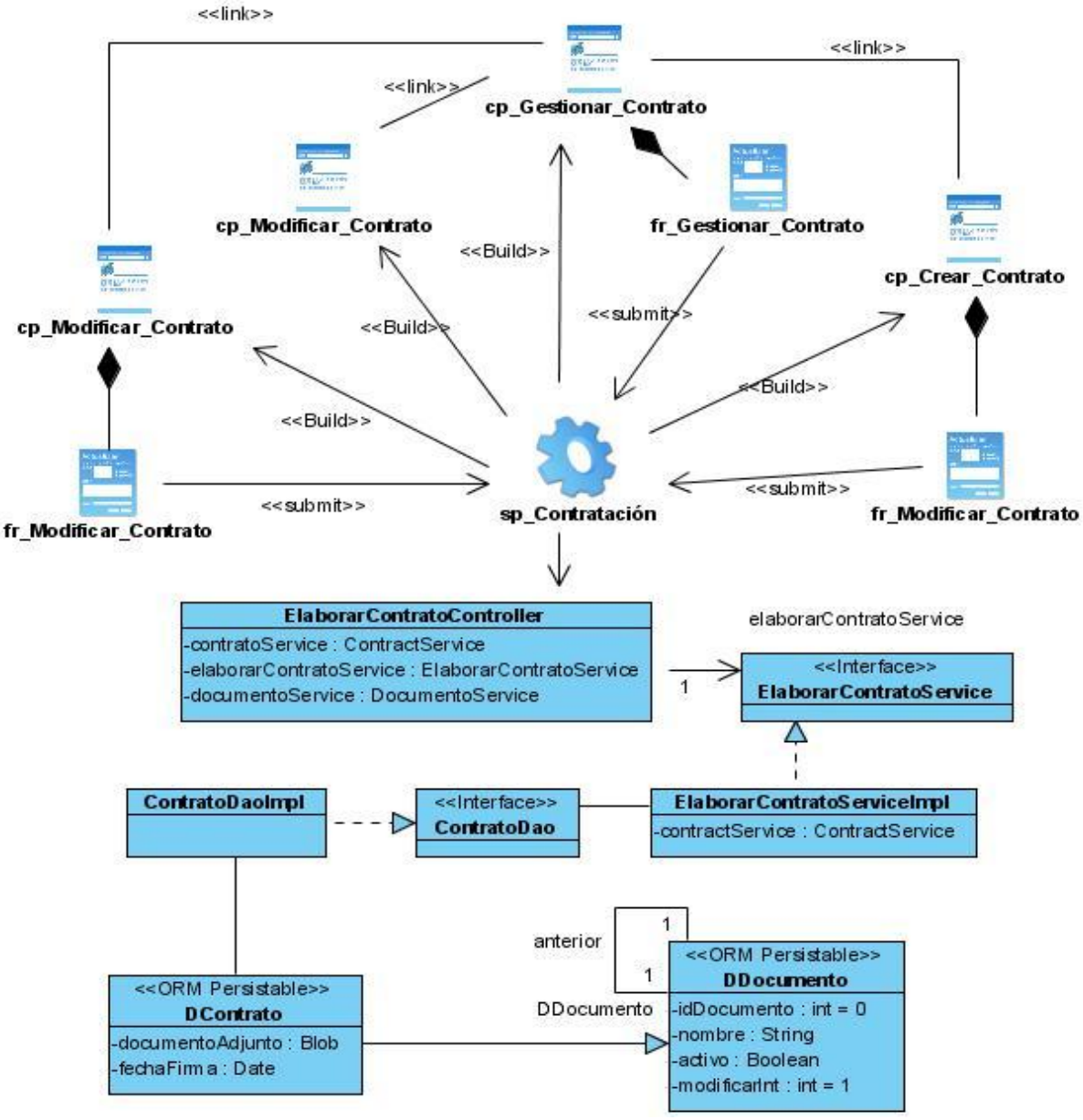



**Figura 13.** Diagrama de clases de diseño, CU Gestionar Propuesta de Contrato.

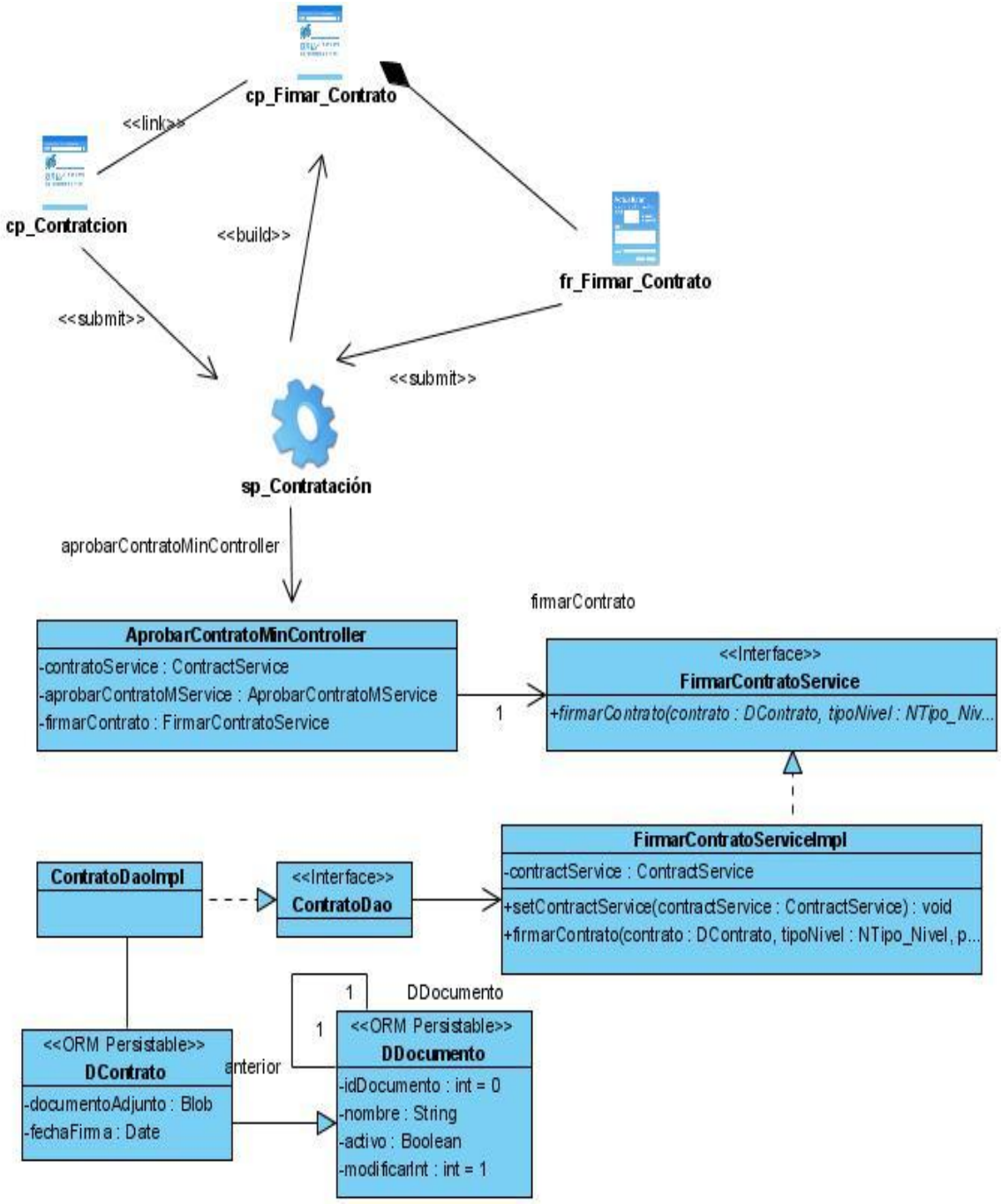



**Figura 14.** Diagrama de clases de diseño, CU Firmar Propuesta de Contrato.

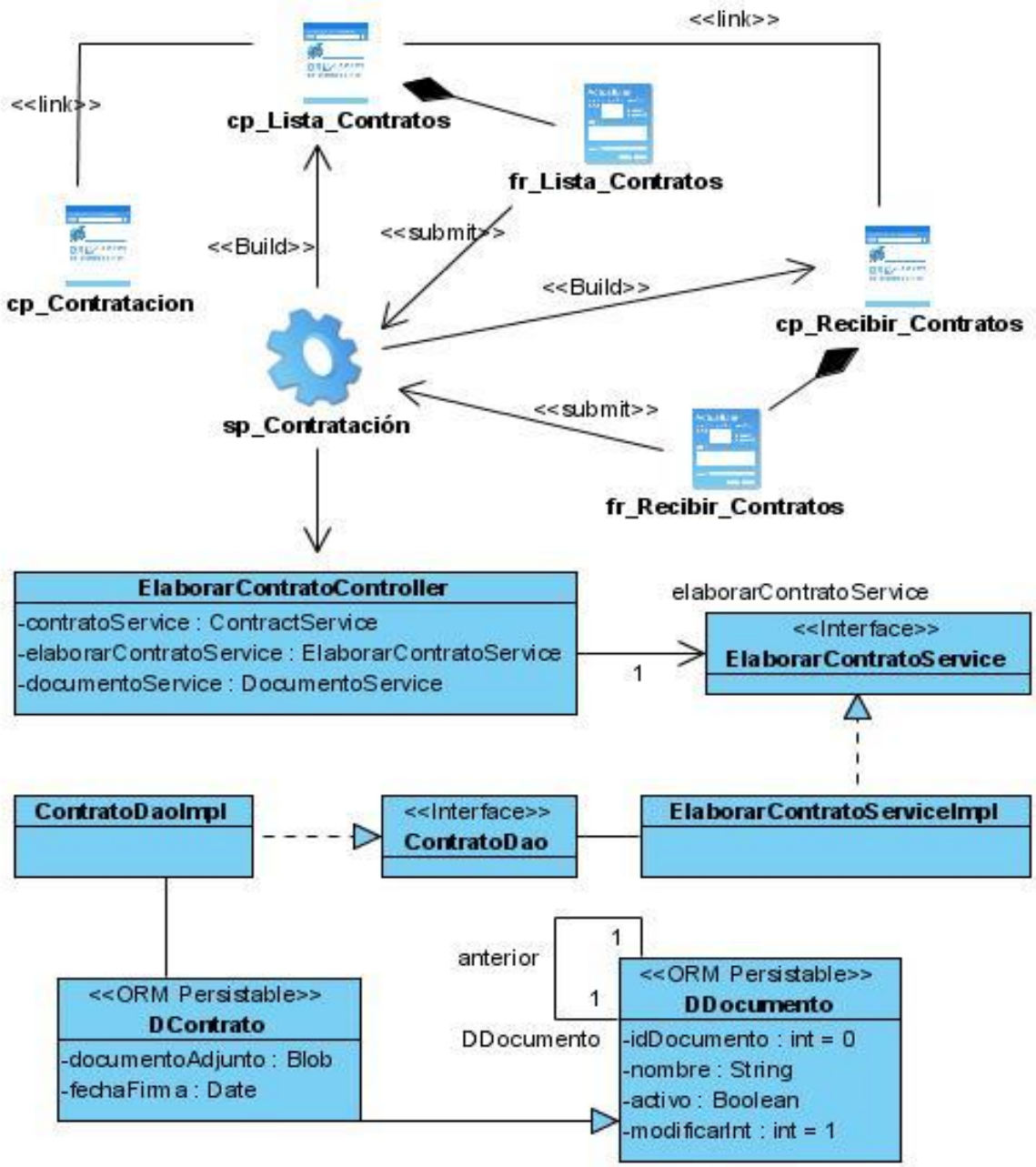



**Figura 15.** Diagrama de clases de diseño, CU Recibir Propuesta de Contrato.

### *2.8.3. Diagrama de Clases Persistentes*

Para facilitar la persistencia de los datos y para poder generar el modelo de datos, que utilizara el sistema de base de datos, y las tablas de la base de datos se generó el diagrama de clases persistentes. Este diagrama representa una relación entre los objetos persistentes y sus respectivas relaciones, dependencias y cardinalidades. Se muestra a continuación como quedo el diagrama en cuestión.

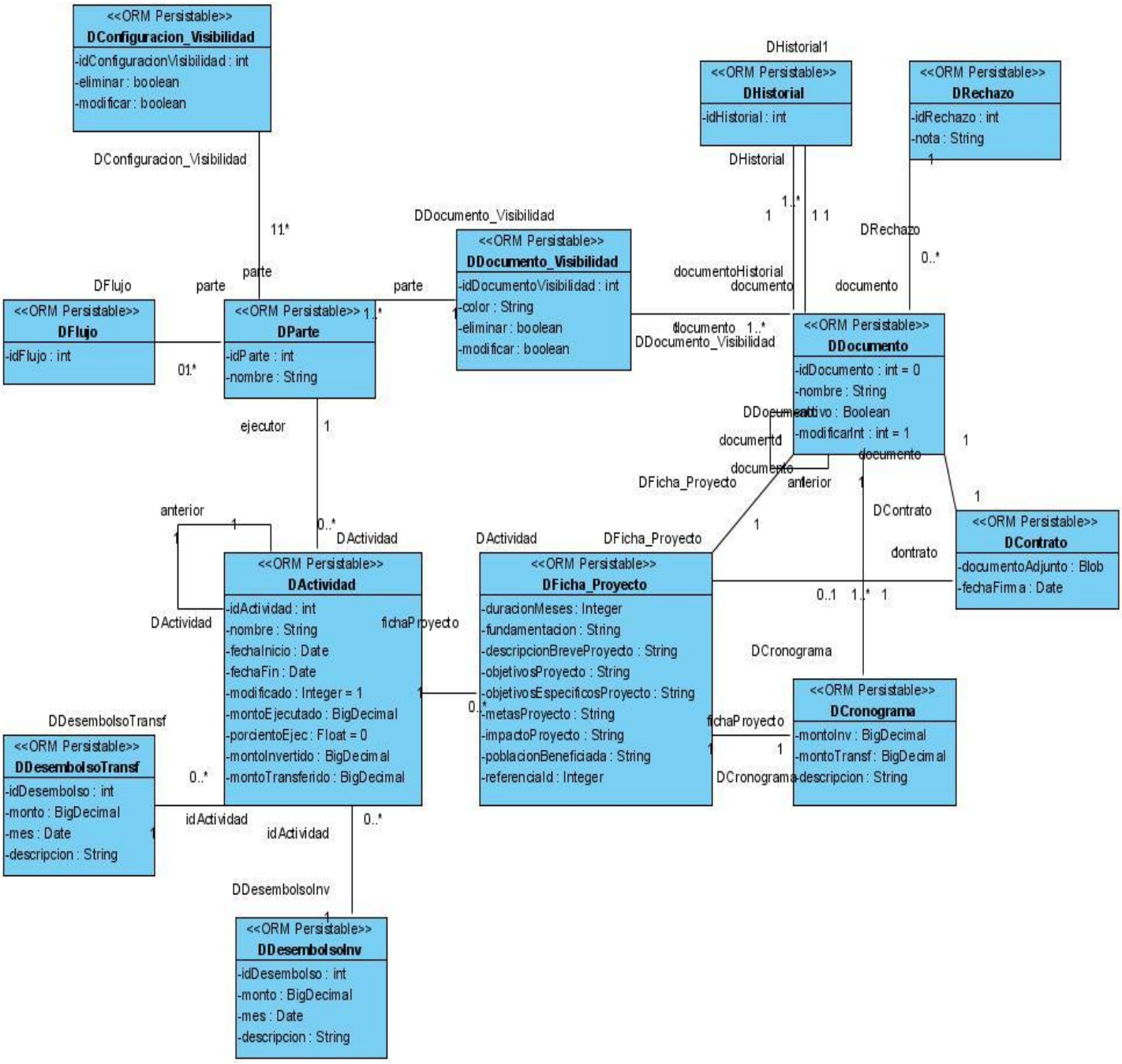



**Figura 16.** Diagrama de clases de Persistentes, Módulo Contratación.

### *2.8.4. Modelo de Datos*

El modelo de datos describe el comportamiento lógico y físico de los elementos persistentes de utilidad para dar soporte a toda la información que se maneja en el sistema. Con el fin de garantizar la seguridad y persistencia de los datos se modeló y normalizó el modelo de entidad relación del módulo Contratación del sistema CCV.

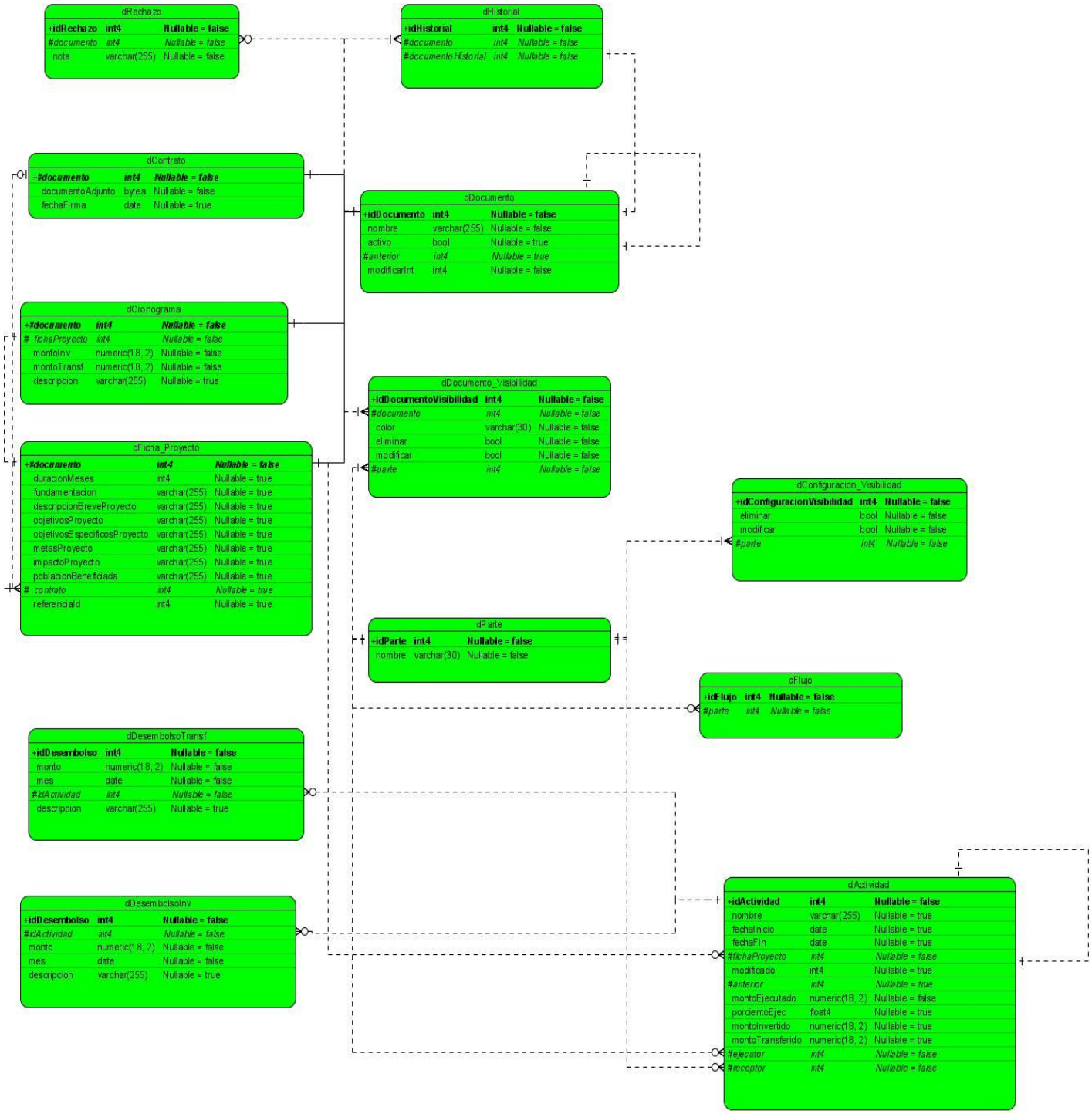



**Figura 17.** Modelo de datos, Módulo Contratación.

## *2.9. Conclusiones parciales*

Podemos decir que en presenta capitulo se llega a las siguientes conclusiones:

- La utilización de las herramientas adecuadas para la modelación de negocio y, el análisis y diseño, del Módulo Contratación, permitió que los clientes y el equipo de desarrollo llegaran a un entendimiento.

- La identificación y especificación de los Requerimientos del sistema facilitó una mayor comprensión y acuerdo común entre los clientes y los desarrolladores, en cuanto a las funcionalidades que el sistema debe cumplir.

- La aplicación de patrones, ya sean de análisis, casos de uso o de diseño, permitió optimizar aun más, los artefactos generados en esta etapa.

- La modelación de los diagramas de clases de diseño con estereotipos web permitió dar al cliente y a los desarrolladores, una idea general del sistema y así verificar las funcionalidades requeridas.

## *Capitulo 3. Validación de los Resultados*

Actualmente, dentro de la comunidad informática se utilizan diferentes técnicas y métricas para el análisis y la validación de los resultados obtenidos en las diferentes etapas de la construcción de un software.

Los flujos de Análisis y Diseño no están exentos de estos procesos de corrección y verificación de los artefactos generados en estas etapas con el fin de especificar la veracidad y la correctitud con el cual fueron obtenidos dichos artefactos.

Existen métricas para la medición de los artefactos obtenidos en estas etapas de la construcción de un software. En el presente trabajo se desarrollan 3 técnicas para la validación de los resultados obtenidos durante el desarrollo del análisis y diseño del Módulo Contratación del proyecto CCV.

### *3.1. Modelo de Negocio*

El desarrollo del modelo de negocio permitió que se llegara a un entendimiento general, entre los diferentes involucrados en el desarrollo del Módulo Contratación del proyecto CCV. El lenguaje utilizado optimizó la identificación de diferentes artefactos, como son las reglas del negocio. Permitió además que el grupo de desarrolladores comprendieran correctamente la estructura y dinámica de la organización involucrada.

La comunicación cliente-desarrollador, permitió además, eliminar posibles atrasos por divagación o ambigüedad de los requisitos obtenidos, los cuales fueron representados correctamente el Diagrama de Casos de Uso del Negocio.

### *3.2. Métrica para la calidad de la especificación de los requisitos de software*

Los requisitos del módulo Contratación fueron comprobados a través de la métrica de la calidad de especificación. Para determinar la especificidad, *ausencia de ambigüedad,* de los requisitos se aplica la métrica basada en la consistencia de la interpretación de los revisores para cada requisito:

$$Q_1 = n_{ui} / n_r$$

Donde:

$n_{ui}$ es el número de requisitos para los que todos los revisores tuvieron interpretaciones idénticas.

$n_r$ es el número total de requisitos, funcionales y no funcionales.

$Q_1$ representa un valor que cuanto más cerca esté de 1, menor será la ambigüedad de la especificación.

Teniendo en cuenta que:

$n_r = n_f + n_{nf}$

Donde:

$n_f$ es la cantidad total de requisitos funcionales.

$n_{nf}$ es la cantidad total de requisitos no funcionales.

Seguidamente se muestran los resultados arrojados tras la aplicación de la métrica.

Para poder evaluar la métrica de la especificidad de los requisitos se realizaron 2 revisiones, con el objetivo de obtener el menor nivel de ambigüedad y la mayor claridad posible en los mismos y así reflejar de forma más exacta las necesidades del cliente. En la siguiente figura podemos observar los resultados recogidos en cada una de las revisiones.

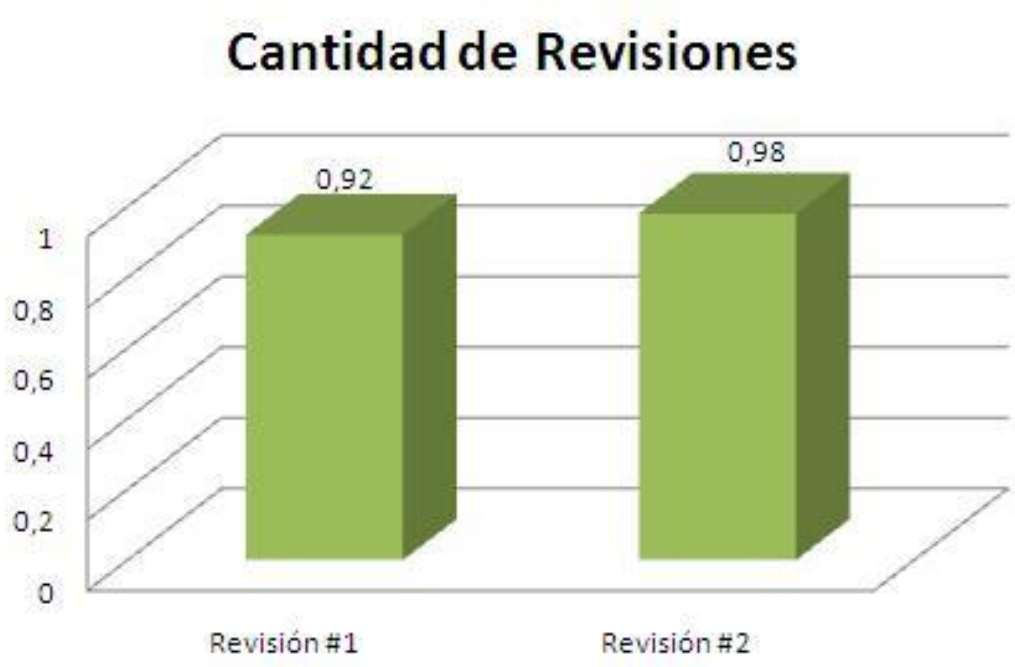



**Figura 18.** Gráfica de Control de la Calidad de la Especificación de Requisitos.

Revisión #1:

| Atributo de Calidad | Tipo de Requisito | Interpretaciones de los revisores | |
|---|---|---|---|
| | | Iguales | Desiguales |
| **Especificidad** | Funcionales | 30 | 3 |
| | No Funcionales | 29 | 2 |
| **Total** | | 59 | 5 |

$n_f = 33$

$n_{nf} = 31$

$n_r = n_f + n_{nf} = 33+31=64$

$Q_1 = nui / n_f = 59 / 64 = 0.9218$

Se observaron irregularidades en los requisitos funcionales y no funcionales obtenidos. Requisitos no funcionales de eficiencia y de seguridad y 3 requisitos funcionales presentaron problemas de redacción. Para una optimización de los requisitos del sistema, se decidió realizar una segunda iteración de revisión de los requisitos.

Revisión #2:

| Atributo de Calidad | Tipo de Requisito | Interpretaciones de los revisores | |
|---|---|---|---|
| | | Iguales | Desiguales |
| **Especificidad** | Funcionales | 32 | 1 |
| | No Funcionales | 31 | 0 |
| **Total** | | 63 | 1 |

$n_f = 33$

$n_{nf} = 31$

$n_r = n_f + n_{nf} = 33+31=64$

$Q_1 = n_{ui} / n_f = 63 / 64 = 0.9843$

Podemos observar que el valor de $Q_1$ (0.9843), en la segunda revisión se aproxima al valor esperado 1. Este resultado demuestra que el grado de ambigüedad en la especificación de los requisitos de software del módulo Contratación del proyecto CCV fue muy bajo, y por consiguiente se demuestra que se desarrolló la especificación de los requisitos, con la calidad necesaria y requerida.

### *3.3. Validación por prototipos no funcionales*

En el presente trabajo se aplicó uno de los nuevos principios operativos del análisis. Este modelo de software es el llamado prototipo no funcional de  de sistemas, y actúa como una forma de mostrarle al cliente cómo quedarán reflejados los requisitos en la aplicación.

*Ventajas de la utilización de Prototipos no funcionales en el Módulo Contratación.*

Permite mostrarle al cliente como se va modelando el sistema, según sus criterios y necesidades, de forma tal de comprender y mejorar los requerimientos obtenidos en las entrevistas y las tormentas de ideas generadas en los flujos de Análisis y Diseño. Permitiendo optimizar la elicitación de requisitos, pues permite realizar cambios significativos y de menor coste que si se realizaran en etapas posteriores. Además su interacción en el momento de concebir los prototipos no funcionales, permite al cliente modificar en tiempo, las posibles interfaces del sistema, permite además agilizar la capacidad de respuesta del equipo de desarrollo. Los prototipos de interfaz realizados en el presente trabajo pueden ser vistos en el Capítulo 2, en cada una de las descripciones de los casos de uso del sistema.

### *3.4. Comprobación De La Calidad Del Modelo De Casos De Uso Del Sistema*

El Modelo de Métricas para Casos de Uso, que se presenta a continuación, tiene por objetivo principal, medir la calidad de la funcionalidad del Sistema a partir del diagrama de casos de uso generado en este proyecto de software. Dicho modelo define cuatro atributos genéricos, que tienen un significado concreto de acuerdo al tipo de artefacto software y al nivel de abstracción que éste describe. Un atributo se analiza en términos de un conjunto de factores cada uno de los cuales tendrá asociada una métrica.

Para evaluar la calidad del diagrama de casos de uso se centró el estudio en un modelo de métricas con el objetivo de medir la calidad de los productos intermedios generados en el proyecto. Los cuatro atributos genéricos de propiedades de calidad, los cuales son:

- **Consistencia:** permite definir el grado en que los elementos del artefacto representan en forma única y no contradictoria un aspecto del problema.

- **Correctitud:** permite establecer el grado de adecuación del artefacto para satisfacer los requisitos establecidos.

- **Completitud:** permite determinar el grado en que se ha incluido de forma clara y concisa todos los elementos necesarios para la descripción del aspecto.

- **Complejidad:** permite medir el grado de claridad y re-uso del artefacto.

Del Modelo de Métricas propuesto se tomaron solamente aquellos factores que se consideraron fundamentales a la hora de evaluar el Diagrama de Casos de Uso del Sistema, de acuerdo a las particularidades del mismo.

Se muestra a continuación los factores que fueron aplicados y sus respectivos valores:

| No. | Atributo | Métricas | Valor (%) |
|---|---|---|---|
| **1** | Completitud | **Número de casos de uso que no tiene descripción resumida.** | 7.69% |
| **2** | | **Número de casos de uso que tienen requisitos omitidos.** | 0% |
| **3** | | **Número de casos de uso que no poseen una descripción extendida.** | 0% |
| **4** | | **Número de casos de uso que tienen acciones del flujo de eventos no redactados en función del responsable.** | 0% |
| **5** | | **Número de casos de uso que no describen condiciones de excepción relevantes.** | 0% |
| **6** | | **Número de casos de uso que no han sido clasificados.** | 0% |
| **7** | Consistencia | **Número de casos de uso que tienen un nombre incorrecto.** | 0% |
| **8** | | **Número de casos de uso que no representan una interacción observable** | 0% |

| | | | |
|---|---|---|---|
| | | **por un actor.** | |
| **9** | | **Número de casos de uso que tienen acciones del flujo de eventos asignados a un responsable que no le corresponde.** | 7.69% |
| **10** | | **Número de casos de uso no aceptados.** | 0% |
| **11** | | **Número de casos de uso complejos que no tienen separación del flujo básico y de flujos alternos.** | 0% |
| **12** | Correctitud | **Número de casos de uso que no tienen un usuario responsable.** | 0% |
| **13** | | **Número de casos de uso en que los requisitos representados no son comprensibles por el usuario.** | 0% |
| **14** | | **Número de casos de uso que deben ser modificados para adecuarlos a la funcionalidad del sistema.** | 0% |
| **15** | | **Número de casos de uso que deben ser modificados para mejorar el proceso actual.** | 0% |
| **16** | Complejidad | **Número de elementos del diagrama que requieren reubicación.** | 0% |

Observaciones:

- En el caso de la métrica #1: De un total de 13 casos de uso en el Módulo Contratación, se detecto que uno de los casos de uso no posee una descripción resumida. Por lo que se tiene un error de 7.69%, valor que no sobrepasa el umbral para esta métrica, que posee un valor de un 10%.
- En caso de la métrica #9: De un total de 13 casos de uso en el Módulo Contratación, se detecto que uno de los casos de uso, existían acciones del flujo de eventos en las cuales el responsable no era el que le correspondía, o sea, especificaba quién realizaba la acción en cuestión, pero el actor era incorrecto. Debido a este error se presento un error de 7.69%, valor que no sobre pasa el 10%, valor que posee como umbral esta métrica.

- Las demás métricas arrojaron un 0% de error en sus respectivos métodos de análisis.

Con el objetivo de optimizar los resultados obtenidos tras la aplicación de esta métrica, se decide por parte del grupo de expertos, someter el diagrama de Casos de Uso del Módulo Contratación a una nueva revisión.

| No. | Atributo | Métricas | Valor (%) |
|---|---|---|---|
| **1** | Completitud | **Número de casos de uso que no tiene descripción resumida.** | 0% |
| **2** | | **Número de casos de uso que tienen requisitos omitidos.** | 0% |
| **3** | | **Número de casos de uso que no poseen una descripción extendida.** | 0% |
| **4** | | **Número de casos de uso que tienen acciones del flujo de eventos no redactados en función del responsable.** | 0% |
| **5** | | **Número de casos de uso que no describen condiciones de excepción relevantes.** | 0% |
| **6** | | **Número de casos de uso que no han sido clasificados.** | 0% |
| **7** | Consistencia | **Número de casos de uso que tienen un nombre incorrecto.** | 0% |
| **8** | | **Número de casos de uso que no representan una interacción observable por un actor.** | 0% |
| **9** | | **Número de casos de uso que tienen acciones del flujo de eventos asignados a un responsable que no le corresponde.** | 0% |
| **10** | | **Número de casos de uso no aceptados.** | 0% |
| **11** | | **Número de casos de uso complejos que no tienen separación del flujo básico y de flujos alternos.** | 0% |

| | | | |
|---|---|---|---|
| **12** | Correctitud | **Número de casos de uso que no tienen un usuario responsable.** | 0% |
| **13** | | **Número de casos de uso en que los requisitos representados no son comprensibles por el usuario.** | 0% |
| **14** | | **Número de casos de uso que deben ser modificados para adecuarlos a la funcionalidad del sistema.** | 0% |
| **15** | | **Número de casos de uso que deben ser modificados para mejorar el proceso actual.** | 0% |
| **16** | Complejidad | **Número de elementos del diagrama que requieren reubicación.** | 0% |

Observaciones:

- La aplicación de las métricas durante la segunda iteración, obtuvo resultados satisfactorios, arrojando un 0% de error en todos aspectos verificados.

Los datos obtenidos durante las revisiones quedaron graficados de la siguiente manera:

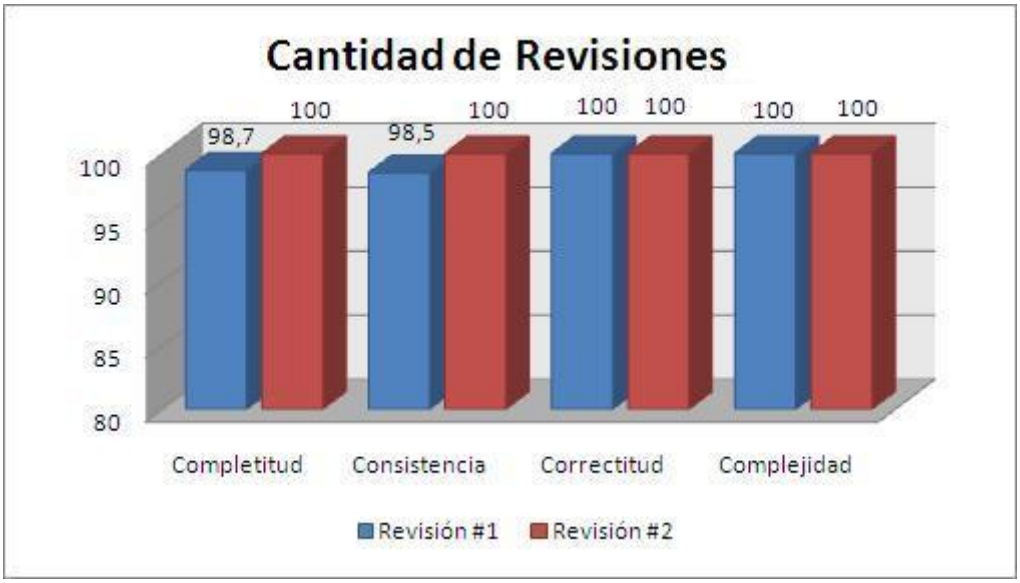



**Figura 19.** Gráfica de Resultados de aplicación de Métricas al diagrama de casos de uso del Módulo Contratación.

### *3.5. Validación por Métricas orientadas a clases. Tamaño de clases*

Para medir el tamaño de las clases (TC), se tienen en cuenta los aspectos siguientes:

- Total de operaciones, ya sean las de ella o las heredadas de las clases padres o interfaces que implementen.
- Cantidad de atributos, al igual que el anterior, tanto los de ella, como los de los padres.
- Promedio general de los dos anteriores para el sistema completo.

Para evaluar las métricas son necesarios los valores de los umbrales, aspecto que es muy discutido por los especialistas en el tema, y dependen mucho del sistema que se esté diseñando. Las clases se clasifican en tres grupos, según su tamaño, los que se presentan en la siguiente tabla junto con los umbrales seleccionados para su clasificación.

| *Clasificación* | *Valores de los Umbrales* |
|---|---|
| ***Pequeño*** | ***<=20*** |
| ***Medio*** | ***>20 y<=30*** |
| ***Grande*** | ***>30*** |

Esta métrica se aplico a las clases de mayor peso y prioridad dentro del Módulo Contratación del proyecto CCV.

| Clases | No. Atributos | No. Operaciones |
|---|---|---|
| **1. ContratoDaoImpl** | 0 | 22 |
| **2. CronogramaDaoImpl** | 0 | 7 |
| **3. BusquedaServiceImpl** | 5 | 27 |
| **4. ContractServiceImpl** | 1 | 8 |
| **5. CronogramServiceImpl** | 1 | 5 |
| **6. ElaborarContratoServiceImpl** | 0 | 14 |
| **7. FirmarContratoServiceImpl** | 1 | 2 |
| **8. ContratacionController** | 0 | 1 |
| **9. ElaborarContratoController** | 3 | 24 |
| **10. BusquedaGenericController** | 2 | 8 |
| **11. BusquedaDetallesController** | 2 | 21 |

En las 11 clases presentadas se obtuvo un promedio de 1.34 atributos y 12.6 operaciones o métodos por clase. Atendiendo a los resultados arrojados por la métrica se tienes que:

| Umbral | Tamaño | Cantidad de Clases |
|---|---|---|
| **<=20** | Pequeño | 7 |
| **>20 y <=30** | Mediano | 4 |
| **>30** | Grande | 0 |

De esta forma la gráfica quedaría de la siguiente manera:

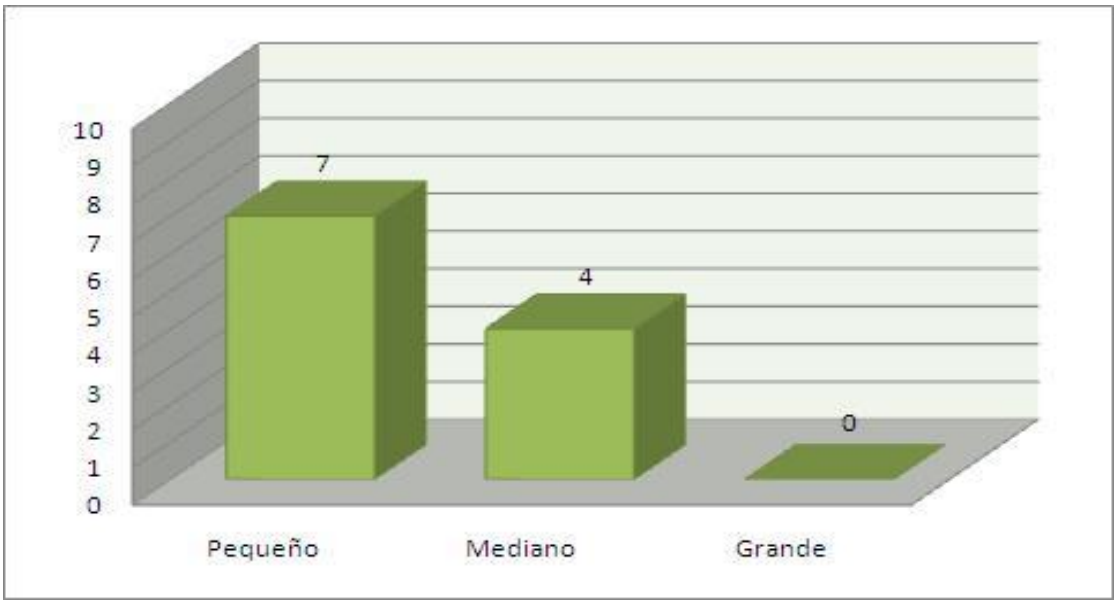



**Figura 20.** Gráfica de Resultados de aplicación de la Métrica de tamaño de clases.

## *3.6. Conclusiones Parciales*

Los resultados arrojados por las métricas utilizadas en este capítulo, para el análisis y validación de los artefactos obtenidos en las diferentes etapas de la Ingeniería del Software por las que transitó el presente trabajo.

La métrica utilizada en la medición de la calidad de la especificación de los requisitos de software, permitió tras una primera revisión que los algunos requisitos presentaron problemas de redacción y ambigüedad. Una segunda revisión al documento de Requisitos de software tras simples modificaciones arrojo un porciento de error nulo para esta métrica.

Los prototipos no funcionales ayudaron a equipo de analistas a tomar decisiones adecuadas en el momento de captura de requisitos, permitió además modelar un posible sistema adecuándose a las necesidades del cliente. Mostro además la posibilidad de agilizar los procesos en esta primera etapa de la construcción de un software.

La métrica aplicada al modelo de casos de uso del sistema, una vez analizada, demostrar que existían errores en el documento de casos de uso. Permitió analizar y verificar los errores cometidos, obteniendo 0% de error tras una segunda iteración de la misma sobre el documento modificado.

# *Conclusiones*

Podemos decir que durante el desarrollo del presente trabajo se cumplieron los objetivos propuestos ya que:

- El estudio del arte de las diferentes metodologías y herramientas permitieron conocer cuáles eran las adecuadas a implementar dentro del desarrollo del sistema.
- Gracias al estudio de los procesos de negocio, se logró una comprensión de la estructura y la dinámica del Convenio Cuba-Venezuela, facilitando un mayor entendimiento entre los clientes y el grupo de desarrollo.
- La construcción de los artefactos generados durante los flujos de trabajo de análisis y diseño del proyecto CCV, cumplieron las metas propuestas y resolvieron la problemática existente. Permitió además obtener los elementos que deberán ser implementados para que el sistema cumpla requerimientos especificados.
- El proceso de análisis implementado sobre los artefactos obtenidos a partir de la utilización de métricas para la validación, permitió conocer el grado de factibilidad que poseían los objetos generados, arrojando resultados satisfactorios.

# Recomendaciones

Con vistas al desarrollo del sistema CCV y la implementación del Módulo Contratación, se recomienda:

- Continuar con la elaboración de los restantes artefactos que genera el flujo de análisis y diseño, que son importantes para una posterior implementación del Módulo Contratación.

- Agregar las funcionalidades de aceptar y rechazar el Cronograma de Ejecución Financiera a nivel de Ministerio y Secretaría Técnica.

- Agregar un nuevo reporte que permita la búsqueda de Contratos por estados.

# Bibliografía Referenciada


[1]. **Denis Szalkowski, 2008** http://dszalkowski.free.fr/Telechargement/support-gestion-de-projet.pdf

[2] **DotProject. 2008**. http://www.abartiateam.com/dotproject

[3] **DotProject. 2009**. http://www.dotproject.net/

[4] **Microsoft Project. 2009**. http://office.microsoft.com/es-es/project/FX100487773082.aspx

[5] **Microsoft Project. 2009**http://office.microsoft.com/es-es/project/HA101656383082.aspx

[6] **Christensen, Brian. 2009.** http://www.pureviolet.net/ganttpv/

[7] **Christensen, Brian. 2008.** http://alpha.simpleprojectmanagement.com/log/category/ganttpv-book/

[8]http://www.genbeta.com/web/milestone-planner-sencillo-planificador-de-proyectos

[9] **Barzanallana, Rafael-** Metodologías de desarrollo de software. 2008. http://www.um.es/docencia/barzana

[10] **Pressman, Roger S.** Ingeniería del software. Un enfoque práctico. Quinta edición. S.l.: McGraw-Hill/Interamericana de España. S.A., 2002.

[11] **Jacobson, I, Booch, G y Rumbaugh, J. 2000.** *El Proceso Unificado de Desarrollo de Software.* Madrid: PEARSON EDUCACIÓN, 2000. 84-7829-036-2.

[12] **JOSÉ H. CANÓS, PATRICIO LETELIER Y Mª CARMEN PENADÉS** (enero 2007): *Todo Ágil*. Disponible en http://www.willydev.net/descargas/prev/TodoAgil.Pdf

[13] *Metodologias de Desarrollo*. 2007. Disponible en:

http://www.um.es/docencia/barzana/IAGP/Iagp2.html#BM1

[14]**Oscar E. Guzman Matus, 2007**: Presentación - Metodología Scrum.ppt

[15]http://www.epidataconsulting.com/tikiwiki/tiki-index.php?page=Scrum&highlight=Scrum%20es%20una%20metodolog%C3%ADa%20%C3%A1gil%20de%20administraci%C3%B3n

[16]**Knowledge Based Systems, Inc.(KBSI). 2006.** Integrated Definition Methods. http://www.idef.com.

[17] **Larman, Craig. 1999.** *UML y Patrones Introducción al análisis y diseño orientado a objetos.* Mexico: Prentice-Hall, 1999. 970-17-0261-1.

[18] Modelado de Procesos. *Departamento de Computación.* 2005. www.decom-uv.cl/~INF203/docs/11%20-%20TGS%20-%20ModProcesos.ppt.

[19] **White, Stephen A. 2004.** Object Management Group/ Business ProcessManagement Initiative. *IBM Corporation* Mayo de 2004: Disponible en: www.bpmn.org/Documents/Introduction%20to%20BPMN.pdf.

[20]**Pressman, Roger**. Ingeniería del Software. Un enfoque práctico. La Habana, Félix Varela, 2005.

[21] **Brooks, Frederick P. 1987.** *No Silver Bullet: Essence and Accidents of Software Engineering.* 1987.

[22]**Fowler, Martin**. 1996. *AnalysisPatterns: Reusable ObjectModels*.

[23]http://martinfowler.com/articles.html#ap

[24] **Gil Gomez. Hussein. 2008.** *Modelo de casos de uso del sistema, Módulo Contratación, Sistema CCV.* La Habana: s.n., 2008.

[25] **Gil Gomez. Hussein. 2008.** *Modelo de Diseño, Módulo Contratación, Sistema CCV.* La Habana: s.n., 2008.

[26] http://www.taringa.net/posts/downloads/943984/ALLFusion-BPWin-4_0-y-4_1-SP2,-Administrando-Procesos-Empres.html

[27] http://dorol.wordpress.com/2007/12/16/case-dorol/

# Bibliografía Consultada


**IBM.** IBM. *IBM.* http://www-01.ibm.com/software/rational/

**Rational Software, Corporation. 2003.** *RUP Help.* s.l. : Rational Software Corporation, 2003.

**Sommerville, Ian y Sawyer, Pete.** *Requirements Engineering: A Good Practice Guide.* Chichester, UK: John Wiley & Sons, 1997.

**Rational Rose Enterprice for UML**. http://www.Rational-rose.com/index-htm.

**Grady Brooch, J. R., Ivar Jacobson.** —UML.

**BPMN**. http://www.emb.cl/gerencia/articulo.mv?sec=7&num=224&mag=1&wmag=64

**BPMN.** http://kuainasi.ciens.ucv.ve/ideas07/documentos/articulos_ideas/Articulo44.pdfh

**Visual Paradigm for UML Model-Code-Deploy platform**. *Visual Paradigm.* http://www.visual-paradigm.com/index-htm.

**Visual Paradigm International. 2005.** Visual Paradigm. *Visual Paradigm.* 2005. http://www.visual-paradigm.com/product/vpuml/.

**IEEE Standards Association. 2004.** *IEEE Standards Association.* 2004. http://standards.ieee.org/reading/ieee/std_public/description/se/610.12-1990_desc.html.

**MOLPECERES, Alberto**. **2002.** *Procesos de desarrollo: RUP, XP y FDD*, 2002.: http://www.javaHispano.org

**SAWYER, Pete & Sommerville, Ian. 1997.** *Requirements Engineering,* 1997. ISBN-13: 978-0-471-97444-4 –

**Denis Szalkowski, 2008** http://dszalkowski.free.fr/Telechargement/support-gestion-de-projet.pdf

**DotProject. 2008**. http://www.abartiateam.com/dotproject

**DotProject. 2009**. http://www.dotproject.net/

**Microsoft Project. 2009**. http://office.microsoft.com/es-es/project/FX100487773082.aspx

**Microsoft Project. 2009**http://office.microsoft.com/es-es/project/HA101656383082.aspx

**Christensen, Brian. 2009.** http://www.pureviolet.net/ganttpv/

**Christensen, Brian. 2008.** http://alpha.simpleprojectmanagement.com/log/category/ganttpv-book/

**Cyberfrancis, rss2. 2009.** http://www.genbeta.com/web/milestone-planner-sencillo-planificador-de-proyectos

**Barzanallana, Rafael-** Metodologías de desarrollo de software. 2008. http://www.um.es/docencia/barzana

**Pressman, Roger S.** Ingeniería del software. Un enfoque práctico. Quinta edición. S.l.: McGraw-Hill/Interamericana de España. S.A., 2002.

**Jacobson, I, Booch, G y Rumbaugh, J. 2000.** *El Proceso Unificado de Desarrollo de Software.* Madrid: PEARSON EDUCACIÓN, 2000. 84-7829-036-2.

**JOSÉ H. CANÓS, PATRICIO LETELIER Y Mª CARMEN PENADÉS** (enero 2007): *Todo Ágil*. Disponible en http://www.willydev.net/descargas/prev/TodoAgil.Pdf

***Metodologias de Desarrollo*. 2007**. Disponible en:

http://www.um.es/docencia/barzana/IAGP/Iagp2.html#BM1

**Oscar E. Guzman Matus, 2007**: Presentación - Metodología Scrum.ppt. http://www.epidataconsulting.com/tikiwiki/tiki-index.php?page=Scrum&highlight=Scrum%20es%20una%20metodolog%C3%ADa%20%C3%A1gil%20de%20administraci%C3%B3n

**Knowledge Based Systems, Inc. (KBSI). 2006.** Integrated Definition Methods. http://www.idef.com.

**Larman, Craig. 1999.** *UML y Patrones Introducción al análisis y diseño orientado a objetos.* Mexico: Prentice-Hall, 1999. 970-17-0261-1.

**Modelado de Procesos, 2005.** *Departamento de Computación.* 2005. www.decom-uv.cl/~INF203/docs/11%20-%20TGS%20-%20ModProcesos.ppt.

**White, Stephen A. 2004.** Object Management Group/ Business ProcessManagement Initiative. *IBM Corporation* Mayo de 2004: www.bpmn.org/Documents/Introduction%20to%20BPMN.pdf.

**Pressman, Roger**. Ingeniería del Software. Un enfoque práctico. La Habana, Félix Varela, 2005.

**Brooks, Frederick P. 1987.** *No Silver Bullet: Essence and Accidents of Software Engineering.* 1987.

**Fowler, Martin**. **1996**. *AnalysisPatterns: Reusable ObjectModels*.

**Fowler, Martin**. **2008**. http://martinfowler.com/articles.html#ap

**Gil Gomez. Hussein. 2008.** *Modelo de casos de uso del sistema, Módulo Contratación, Sistema CCV.* La Habana: s.n., 2008.

**Gil Gomez. Hussein. 2008.** *Modelo de Diseño, Módulo Contratación, Sistema CCV.* La Habana: s.n., 2008.

# *Anexos*

***Anexo #1***: Acta de Aceptación del Módulo Contratación.

albet INGENIERÍA Y SISTEMAS | PROYECTOS

ACTA DE ACEPTACIÓN

**Producto:** Sistema de Gestión para Seguimiento de los Proyectos del Convenio Integral de Cooperación Cuba-Venezuela.

**Categoría:** Aceptación CU Módulo de Contratación.

**Fecha de la conciliación: 3/07/2008.**

**Involucrados en el proceso:**

**Por la parte del Cliente (MENPET):** Sandra Cortés
**Por la parte del Suministrador (ALBET):** Ing. Nahuel Massón

**Observaciones del proceso:**

Por acuerdo entre las partes involucradas en el proceso se Acepta el Documento CU del Módulo de Contratación de la versión v.2.0 del Sistema de Gestión para Seguimiento de los Proyectos del Convenio Integral de Cooperación Cuba-Venezuela, con fecha 3 de julio de 2008.

Para que conste la aceptación de la descripción de los CU y requerimientos del Módulo de Contratación, dando fé al acuerdo, firman la presente los principales representantes de las Partes.

Sandra Cortés
Representante MENPET

Ing. Nahuel Massón
Representante ALBET

Referencia: CV-SW-CC-011

**ALBET, S.A.**
Centro de Negocios Miramar. Edificio Barcelona, Oficina 322. Avenida 5ta e/ 76 y 78. Miramar. Playa, Ciudad Habana, Cuba
**Tel/Fax:** +53 (7) 837 2407
**E-mail:** albet@albet.cu

***Anexo #2***: Actas de Finiquito del Proyecto CCV.

# ACTA DE FINIQUITO

Entre la República Bolivariana de Venezuela, actuando por órgano del Ministerio del Poder Popular para la Energía y Petróleo de la República Bolivariana de Venezuela, representado en este acto por el ciudadano **Ammar Jabour,** venezolano, mayor de edad, de este domicilio, titular de la Cédula de Identidad No. 9962729, quien actúa en su condición de Coordinador General del Convenio Integral de Cooperación Cuba-Venezuela, suficientemente facultado para este acto y debidamente designado para ejercer tal condición conforme a lo establecido en **CONTRATO E08-001-000 SOLUCIÓN TECNOLÓGICA INTEGRAL PARA EL DESARROLLO DEL SISTEMA DE GESTIÓN PARA SEGUIMIENTO DE LOS PROYECTOS DEL CONVENIO INTEGRAL DE COOPERACIÓN CUBA VENEZUELA,** que en lo sucesivo se denominará la "**Parte Venezolana**", por una parte; y por la otra, la sociedad mercantil **ALBET, Ingeniería y Sistemas, S.A.**, sociedad mercantil cubana constituida mediante Escritura 271 de fecha 7 de Noviembre de 2005, autorizada por la Notario Lic. Isabel Cristina Martínez Alfonso con sede en Notaría Especial del Ministerio de Justicia de Ciudad de la Habana, inscrita en el Registro Mercantil de esta ciudad con fecha 14 de Noviembre del año 2005, al Tomo XVIII, Folio 120, Hoja 11, Sección SM, con Nº de inscripción 1 con domicilio social en Centro de Negocios de Miramar, Edificio Barcelona, Oficina 322, Avenida 5ta entre 76 y 78, Miramar, Municipio Playa, Ciudad de La Habana, República de Cuba, representada en este acto por el ciudadano cubano **Ibrahim Nápoles Albanés**, mayor de edad, portador de carné de identidad Nº 62032504808 en su condición de Coordinador General, suficientemente facultado para este acto según lo dispuesto en **CONTRATO E08-001-000 SOLUCIÓN TECNOLÓGICA INTEGRAL PARA EL DESARROLLO DEL SISTEMA DE GESTIÓN PARA SEGUIMIENTO DE LOS PROYECTOS DEL CONVENIO INTEGRAL DE COOPERACIÓN CUBA VENEZUELA**, que en lo sucesivo se denominará **"Parte Cubana",** al tenor de las siguientes declaraciones y cláusulas:

## CONSIDERANDO

**Primero:** Consta de documento privado suscrito en fecha 18 de marzo de 2008, que ambas partes celebraron un **CONTRATO DE SOLUCIÓN TECNOLÓGICA INTEGRAL PARA EL DESARROLLO DEL SISTEMA DE GESTIÓN PARA SEGUIMIENTO DE LOS PROYECTOS DEL CONVENIO INTEGRAL DE COOPERACIÓN CUBA VENEZUELA,** el cual fue celebrado al Amparo del Convenio Integral de Cooperación, suscrito el 30 de octubre del 2000 y su Addendum; de la Declaración Conjunta y el Acuerdo suscrito entre ambas Repúblicas, para la aplicación de la Alternativa Bolivariana para las Américas, firmados en diciembre del 2004 y de los Acuerdos y las Condiciones Generales firmados en el Acta de la Reunión de la Comisión Mixta Cuba-Venezuela, celebrada en La Habana, Cuba, cuyos contenidos se dan aquí enteramente por reproducidos.

**Segundo**: Consta que la **Parte Cubana** ha cumplido a entera satisfacción de la **Parte Venezolana** con el objeto, alcance y actividades previstas en el **Contrato** ya mencionado, así como con sus Anexos y Suplementos, cumpliendo por tanto con todos sus deberes y obligaciones por lo que se considera que la Solución ha sido implementada en las condiciones previstas y bajos los requisitos y especificaciones técnicas pactadas entres **Las Partes**.

**Tercero:** Consta que la **Parte Cubana** ha hecho entrega del **Informe Técnico Final**, de conjunto a toda la documentación que avala y soporta lo descrito en el Segundo de los **CONSIDERANDOS**, debidamente firmada y aceptada por los especialistas de la **Parte Venezolana**.

**Cuarto**: Consta que la **Parte Venezolana** ha pagado la totalidad de **UN MILLÓN CUATROCIENTOS DIECISIETE MIL OCHOCIENTOS VEINTIOCHO DOLARES DE LOS ESTADOS UNIDOS DE AMERICA CON VEINTIUN CENTAVOS** mediante la forma de pago prevista en el Contrato ya mencionado, a entera satisfacción de la Parte Cubana, quedando a la firma de la presente **Acta**.

## LAS PARTES CONVIENEN:

**Primero:** Dar por terminada la relación contractual derivada del **CONTRATO DE SOLUCIÓN TECNOLÓGICA INTEGRAL PARA EL DESARROLLO DEL SISTEMA DE GESTIÓN PARA SEGUIMIENTO DE LOS PROYECTOS DEL CONVENIO INTEGRAL DE COOPERACIÓN CUBA VENEZUELA** de fecha 16 de marzo de 2008.

**Segundo: Las Partes** se otorgan el finiquito más amplio que en derecho proceda, no reservándose acción o derecho que ejercitar con posterioridad.

Leída que les fue la presenta Acta, las partes se ratificaron en su contenido, para constancia firman en cuatro (4) ejemplares de igual tenor en la Ciudad de Caracas el día 17 del mes de diciembre del 2008.

| Por la PARTE VENEZOLANA | Por la PARTE CUBANA |
|---|---|
| **Ammar Jabour**<br>Coordinador General | **Ibrahim Nápoles Albanés**<br>Coordinador General |

## *Glosario de Términos*

***Patrón de diseño***: Es una solución a un problema de diseño. Para que una solución sea considerada un patrón debe poseer ciertas características. Una de ellas es que debe haber comprobado su efectividad resolviendo problemas similares en ocasiones anteriores. Otra es que debe ser reusable, lo que significa que es aplicable a diferentes problemas de diseño en distintas circunstancias.

***Metodologías:*** es una palabra compuesta por tres vocablos griegos: *metà* (**"más allá"**), *odòs* (**"camino"**) y *logos* (**"estudio"**).Son los **métodos de** investigación que permiten lograr ciertos objetivos en una ciencia. En otras palabras, la metodología es una etapa específica que procede de una posición teórica y epistemológica, para la **selección de técnicas** concretas de investigación.

***Herramientas de modelado:*** Las herramientas de modelado de sistemas informáticos, son herramientas que se emplean para la creación de modelos de sistemas que ya existen o que se desarrollarán. Un sistema informático puede requerir diferentes herramientas de modelado, que resultarán en diferentes tipos de modelos. Las herramientas de modelado utilizadas dependen del analista, del tipo de sistema, de los requerimientos, etc.

***Requisitos o requerimientos de un software:*** La Especificación de Requisitos de un Software (ERS) es una descripción completa del comportamiento del sistema que se va a desarrollar. Incluye un conjunto de casos de uso que describe todas las interacciones que tendrán los usuarios con el software. Los casos de uso también son conocidos como requisitos funcionales. Además de los casos de uso, la ERS también contiene requisitos no funcionales (o complementarios). Los requisitos no funcionales son requisitos que imponen restricciones en el diseño o la implementación (Como por ejemplo restricciones en el diseño o estándares de calidad).

***Casos de Uso:*** Un caso de uso es una técnica para la captura de requisitos potenciales de un nuevo sistema o una actualización de software. Cada caso de uso proporciona uno o más escenarios que indican cómo debería interactuar el sistema con el usuario o con otro sistema para conseguir un objetivo específico. Normalmente, en los casos de usos se evita el empleo de jergas técnicas, prefiriendo en su lugar un lenguaje más cercano al usuario final.

***Módulo Contratación:*** Es la parte de sistema que se encarga de automatización de los contratos, que se proponen a partir de los proyectos que se presentan dentro de las mixtas de cooperación entre Cuba y Venezuela, y todo el flujo que los afecta.

***Métrica:*** Medida cuantitativa del grado en que un sistema, componente o proceso posee un atributo dado.

***CASE:*** (Computer Aided Software Engineering), Ingeniería de Software Asistida por Ordenador).

***EE:*** Ente Ejecutor. Organismo u Empresa que funciona como coordinador y responsable de un proyecto.

***M:*** Ministerio, entidad rectora de una rama especifica, actúa como rector de los Entes Ejecutores.

***ST:*** Secretarias Técnicas. Oficinas rectoras de las relaciones Cuba-Venezuela.