UNIVERSIDAD DE LAS CIENCIAS INFORMÁTICAS

FACULTAD 3

Título: "Componente Gestor de Intercambio de Información JIT para el Sistema para la Planificación de Actividades (SIPAC)".

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

Autores: Liu Pérez Ballester

Jiorqui Vázquez Fitó

Tutores: Ing. Néstor Bernal Vidal

Ing. Yinet Pérez-Terán Pérez

La Habana 18 de Junio 2012

"Los planes son solamente buenas intenciones a menos que degeneren inmediatamente en trabajo duro"

Peter Drucker

Declaración de autoría

Declaramos ser los únicos autores de este trabajo y autorizamos a la Universidad de las Ciencias Informáticas a hacer uso del mismo en su beneficio.

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

Liu Pérez Ballester

_____

Firma del autor

Jiorqui Vázquez Fitó

_____

Firma del autor

Ing. Néstor Bernal Vidal

_____

Firma del tutor

Ing. Yinet Pérez-Terán Pérez

_____

Firma del tutor

Agradecimientos

Primero que todo quiero agradecer a mi madre, mi novia fiel, mi amor eterno, que se ha sacrificado tanto por mí, se que nada en el mundo sería suficiente para pagarle su esfuerzo, desvelo y lágrimas por mí. A mi padre por demostrarme que siempre se puede ser un poco mejor y que de los errores también se aprende y pueden nacer cosas bellas. A mi abuela Caridad por ser la viejita más linda del mundo y ocuparse tanto de mí, a Morell que aunque ya no está con nosotros siempre lo llevo en mi corazón y sé que estaría muy orgulloso de mí por este logro. A mis hermanos pero en especial a Richi que siempre comprendió cuando no pude jugar con el por quedarme en la UCI a estudiar. A mis tías, tíos, primos y primas por ser su mango y a mi padrino Santy. A la UCI por darme la oportunidad de conocer tanta gente maravillosa que nunca olvidaré, en especial a Liu por enseñarme lo que es el amor, por ser tan especial en mi vida, por su incondicionalidad y por ayudarme tanto en esta tesis, a Angel a Irina por ser tan buenos amigos. Al piquete de siempre: al bode, al wilfre, al danis, al vladi, a efra, a yani, a jardo, en fin a todos. Al equipo del projecto por ayudarnos tanto. Al tribunal por ser tan pacientes y exigentes, al oponente por sus señalamientos y su exigencia, al tutor por luchar tanto con nosotros y por siempre pedirnos el máximo, a wendy por ser tan buena amiga y ayudarnos tanto. En fin a todos.

Jiorqui

Quiero agradecer primeramente a mis padres porque gracias a ellos hoy soy Ingeniera, ellos son mi mayor impulso y todo lo que hago es pensando en mí pero también en ellos para que estén siempre orgullosos de mí y de mi hermano ya que ellos lo han dado todo para hacernos muy feliz a los dos. También agradezco a mi lindo hermano que tiene un corazón tan grande que no le cabe en el pecho, se que siempre estará a mi lado para cuidarme, apoyarme, celarme y quererme mucho como yo lo quiero a él. A toda mi familia sin excepción de nadie, porque es una familia muy unida y estoy muy orgullosa de todos, a mis abuelas Martha y Carmita, mis tíos y tías que sé que soy como una hija más para ellos y los quiero como mis padres, a mis primos, a todos los quiero mucho por confiar en mí y por apoyarme cuando los he necesitado. Quiero agradecer a todas las personas especiales que he conocido en el transcurso de mis años, que han sido parte de mi vida y de los cuales siempre he tenido algo que aprender, a todos esos amigos gracias y aunque unos estén lejos y otros más cerca siempre los llevaré en mi corazón. A la UCI una universidad tan importante, donde me forme como una verdadera profesional y además conocí a muchas personas buenas como Dayana, Yailen, Rosalia, Walva mi fiel compañero de mesa y su novia Lisandra, a La Flaca, a mi amigo Oslando y a sus padres, a Irina y Angel(los Tutos) que siempre los llevare conmigo y recordare todos los momentos que pasamos juntos en estos últimos tiempos, a todos gracias. Quiero agradecer la linda oportunidad que me dio la vida de conocer a una persona tan especial, sencilla, educada e inteligente, Miji, alguien a quien amo y le agradezco su amor, su confianza, su comprensión, por estar a mi lado y soportar mis malacrianzas y mi carácter. A mis suegros y mi cuñado Richard que en este poco tiempo me han acogido en su casa como una hija donde me he sentido como en familia. Agradezco a los profesores que tuve en estos 5 años de universidad, a los compañeros de aula, los de proyecto Maire, Rodo, Yinet, Ari y los

demás por hacer tan buen equipo, al tutor Néstor por buscar tiempo donde no había para dedicarlo a nosotros, también a su novia Wendy que es muy buena amiga y nos ha acogido en su apto como hijos, al tribunal por sus señalamientos, a la oponente.

Gracias
Liu

Dedicatoria

Dedico este trabajo a mi madre por sus días sin dormir y por todos los sacrificios, para que se sienta aún más orgullosa de mi como yo lo estoy de ella, a mi padre por enseñarme lo que es la honestidad y el trabajo duro y por obligarme a poner todo mi empeño, a mi hermano Richard para que vea en mi un ejemplo y siempre de lo mejor de sí, a Morell por ser un segundo padre y quererme tanto, a mi abuela por ser tan buena y tener tan buena opinión de mi, a Liu por amarme tanto como yo a ti.

Jiorqui

Dedico este trabajo de diploma especialmente a mis padres, Pilar y Rubén y a mi hermano Ray, a ellos por su apoyo toda mi vida y por hacerme crecer en un hogar tan lindo como el nuestro, a ellos por estar tristes, felices, nerviosos, preocupados cuando yo lo estoy; a mi madre que es mi mejor amiga por tener siempre un consejo, una frase linda para mi, una caricia, un beso; a mi Pa por sus consejos, su confianza, su esfuerzo, por cumplir su parte del trato y aquí está la mía cumplida, es para ti. A toda mi familia, a mis abuelas, mis tíos, mis primos y a los más pequeños para que sigan mi ejemplo y para que estudien y se hagan profesionales. A mis amigos y todas las personas que me quieren bien. A Miji quiero dedicar este trabajo también y toda mi vida...

Liu

Resumen

En Cuba es sumamente importante lograr una planificación de actividades eficiente en concordancia con los recursos que se disponen. Esta planificación se elabora siguiendo el modelo de Planificación por Objetivos que surge de la unión de la Dirección por Objetivos y la Planeación Estratégica. Un proceso vital de este modelo es el Intercambio y Actualización de la información que tiene lugar en varios momentos, desde que se elabora el documento que va a regir la planificación y se necesita hacerlo llegar a los diferentes niveles involucrados para su estudio, aprobación y ejecución del plan donde se necesita que cada responsable o involucrado en la planificación sepa lo que debe hacer. A pesar de su importancia, este intercambio se hacía a través de mecanismos y herramientas que no abarcaban todos los escenarios y no era posible decidir qué información, qué cantidad y en qué momento se iba a efectuar el intercambio. A raíz de esta situación se propone el análisis, diseño e implementación del Componente Gestor de Intercambio de Información JIT para el Sistema para la Planificación de Actividades.

Palabras claves

Actividades, Actualización, Elementos de planificación, Intercambio, Objetivos, Plan, Planificación, Planificación de Actividades, Planificación por Objetivos.

Tabla de contenido

Índice de Figuras

Índice de Tablas

Introducción

En el mundo actual uno de los métodos para organizar, planificar, dirigir y controlar más utilizado y difundido en cuanto a buenas prácticas de planificación se refiere es la Planificación por Objetivos (PPO) .

A mediados de los años 90 se incorporaron en el país numerosos y complejos procedimientos relacionados con la Planeación Estratégica y la Dirección por Objetivos, luego en octubre del año 1997 en el V Congreso del Partido Comunista de Cuba (PCC), se pasó a un plano superior la planificación tomando como base los resultados obtenidos en el perfeccionamiento empresarial llevado a cabo en las Fuerzas Armadas Revolucionarias (FAR) y su extensión a las entidades civiles, se trazaron estrategias para adecuar y mejorar las bases teóricas, organizativas y metodológicas de la planificación a las realidades de Cuba. La dirección del país, como parte del fortalecimiento de la empresa estatal socialista y la informatización de la sociedad cubana, ha planteado la necesidad de informatizar los procesos que comprende la Planificación por Objetivos en las entidades cubanas. (1)

A partir de ese momento se plantea la necesidad de un sistema que informatice el proceso de planificación de actividades bajo el modelo PPO en las entidades cubanas, teniendo en cuenta las características particulares y generales de cada una de ellas y lo establecido en la Instrucción No.1 del Presidente de los Consejos de Estado y de Ministros para la planificación de los objetivos y actividades en los órganos, organismos de la administración central del estado, entidades nacionales y las administraciones locales del Poder Popular.

Hasta el día de hoy la informatización de este proceso se venía llevando a cabo en herramientas que no respondían a todas las exigencias planteadas en la instrucción mencionada, entre ellas Microsoft Outlook y Microsoft Excel mediante formatos definidos por el Comité Ejecutivo del Consejo de Ministros (CECM) para la planificación en el país. Otra herramienta utilizada mayormente en las entidades de las FAR es P-TRAB, catalogada como la de mayor uso y eficiencia entre las mencionadas anteriormente pero que en la actualidad no responde a las concepciones de la PPO en el país, debido a la evolución que ha tomado la misma en aras del perfeccionamiento; además de la inconveniente arquitectura de dicho sistema (desarrollado sobre MSDOS[1]).

[1] **S.O. MSDOS**: Microsoft Disk Operating System, sistema operativo de disco de Microsoft.

Ante la necesidad de contar con una herramienta que informatice este proceso al nivel requerido y teniendo en cuenta las características de las entidades cubanas surge el proyecto Sistema para la Planificación de Actividades (SPA), producto de la unión y cooperación entre la Unidad de Compatibilización, Integración y Desarrollo de software para las FAR (UCID) y el Centro de Informatización de Gestión de Entidades (CEIGE), cuya misión radica en desarrollar una herramienta con las funcionalidades necesarias para poder ejercer la planificación en dichas entidades y que permita adaptarse lo más posible a cualquier forma de planificación en base a actividades para así aumentar la flexibilidad del mismo, por si en un futuro se cambia estratégicamente la forma o método de planificación en las entidades cubanas.

En diciembre de 2010 se libera una primera versión del producto conocido como SIPAC (Sistema para la Planificación de Actividades), con la cual se procede a efectuar pruebas en varios ministerios del país, dígase Ministerio de Educación (MINED), Ministerio de las Fuerzas Armadas Revolucionarias (MINFAR), entre otros, con el objetivo de explorar las funcionalidades del sistema en un ambiente lo más real posible y utilizarla como plataforma para la posible identificación de funcionalidades y términos de usabilidad no previstos durante la modelación de procesos y definición de requisitos de SIPAC. Una de estas necesidades que surge a raíz de que inicialmente se pensaba poder tener lista una infraestructura que garantizara la comunicación en línea todo el tiempo entre las partes involucradas en la planificación del país para el momento de la liberación e implantación de la primera versión del producto y que por diversas razones no se materializó, es la de contar con un mecanismo de intercambio y actualización de la información planificada entre las entidades y que sea de fácil acceso hasta por el usuario más simple dentro del sistema, sin depender de terceros o usuarios avanzados, en el momento deseado y con el tipo y cantidad de información deseada.

A raíz de este análisis surge el siguiente **problema a resolver**:

El intercambio y actualización de la información que se lleva a cabo durante el proceso de planificación en las entidades cubanas con el Sistema para la Planificación de Actividades (SIPAC) afecta la gestión del tiempo, el volumen de información y la capacidad de ejecución de los planificadores.

El **objeto de estudio** en el cual se enmarca el problema anteriormente planteado es: el proceso de planificación de actividades en las entidades cubanas, trazándose como **objetivo general:** Desarrollar el componente Gestor de Intercambio de Información JIT para el Sistema para la Planificación de

Actividades (SIPAC), de modo que se contribuya a que el proceso de intercambio y actualización de la información se realice en menos tiempo, con el volumen de información deseado y una mayor capacidad de ejecución por parte de los planificadores en el proceso de planificación vigente en las entidades cubanas.

Quedando desglosado en los siguientes **objetivos específicos**:

1 Fundamentar la investigación sobre el proceso de intercambio y actualización de la información, mediante la elaboración del Marco Teórico.
2 Diagnosticar la situación actual en que se encuentra el proceso de intercambio y actualización de la información en las entidades cubanas.
3 Realizar el análisis y diseño del componente Gestor de Intercambio de Información JIT.
4 Implementar el componente Gestor de Intercambio de Información JIT y efectuar pruebas de validación al mismo.

Por consiguiente la presente investigación se encuentra enmarcada en el **campo de acción**: La gestión del intercambio y actualización de la información generada dentro del proceso de planificación de actividades en las entidades cubanas.

Teniendo en cuenta los elementos expuestos anteriormente, se define como **idea a defender**: El desarrollo de una solución para el intercambio y actualización de la información en el proceso de planificación en las entidades cubanas con el Sistema para la Planificación de Actividades (SIPAC), contribuirá a que se realice en menos tiempo, con el volumen de información deseado y una mayor capacidad de ejecución por parte de los planificadores.

Esto permitirá como **posible resultado** de la presente investigación la construcción del componente Gestor de Intercambio de Información JIT para el Sistema para la Planificación de Actividades (SIPAC).

La **estrategia de investigación** empleada es la referida como Experimental o Explicativa, pues permite establecer los vínculos causales, leyes y mecanismos internos del objeto de investigación, partiendo de conocimientos suficientes acerca del problema que posibilitan plantear una hipótesis (en este caso se traduce a idea a defender) a nivel explicativo.

Los **métodos de investigación** científica empleados en la investigación son: dentro del conjunto de **Métodos Teóricos**, el de **Modelación**, donde se lleva a cabo una modelación del proceso de

actualización e intercambio de información, permitiendo predecir la respuesta de dicho proceso a variaciones de algunos de sus parámetros, sin tener que ejecutar el proceso en la realidad; el **Analítico-Sintético** para analizar los hechos partiendo de la descomposición del objeto de estudio en cada una de sus partes y estudiarlas en forma individual, luego integrarlas y reconstruir un todo a partir de los elementos estudiados en el análisis y el **Histórico-Lógico** para la administración de los datos relacionados con la instauración del proceso de Planificación por Objetivos en Cuba.

Para obtener los datos se utilizó el **Método Empírico** de **Entrevista**, con el propósito de construir una concepción inicial de la problemática, estas fueron realizadas a funcionales o especialistas de SIPAC mediante despachos frecuentes.

Estructura del documento:

Capítulo 1: Fundamentación Teórica:

Se expone el estado del arte referente al proceso de planificación de actividades vigente en las entidades cubanas, sus principales características y su origen, así como el proceso de intercambio y actualización de la información generada dentro del proceso de planificación. Se realiza la fundamentación teórica del tema. Se describen y valoran algunas de las soluciones existentes. Se señalan las características principales del modelo de desarrollo propuesto y se justifican las herramientas y tecnologías para la modelación e implementación de la solución que se propone.

Capítulo 2: Análisis y diseño de la solución:

En este capítulo se describen las características esenciales que debe tener la solución propuesta, sustentadas por la modelación de los procesos de negocio y la definición de los requisitos. Se plasman algunos de los artefactos propuestos en la fase de Modelación definida en el modelo de desarrollo aplicado, el diseño del componente y la arquitectura que soporta los módulos de software.

Capítulo 3: Implementación y validación de la solución propuesta:

Este capítulo incluye lo referente a la implementación del componente y a la validación del sistema, garantizándose este último a través de pruebas de calidad.

Capítulo I: Fundamentación Teórica

1.1 Introducción

En el presente capítulo se brinda información sobre las principales características y el origen del proceso de planificación de actividades vigente en las entidades cubanas, así como una panorámica del proceso de intercambio y actualización de la información generada dentro de dicho proceso. Se ofrece, de manera breve, una visión de las herramientas utilizadas hasta la actualidad para la informatización del proceso de planificación de actividades en dichas entidades y un resumen de las técnicas y tecnologías a emplear en el desarrollo de la solución propuesta.

1.2 Estudio del estado del arte

1.2.1 Planificación, Dirección por Objetivos y Planeación Estratégica

La planificación como evento o hecho toma lugar en cada momento de la vida de cualquier persona, desde el pensamiento más simple para saber qué hacer ante la básica necesidad de alimentarse, o ante la proyección a largo alcance de metas más ambiciosas según los intereses que se persiguen.

Muchos autores dan su punto de vista y enfocan de diferentes maneras el concepto de planificación, James Arthur Finch en 1996 planteó que la planificación *“Es el proceso de establecer metas y elegir medios para alcanzar dichas metas”* (2) y en 1998 el Dr. Leonard Goodstein planteó que *"Es el proceso de establecer objetivos y escoger el medio más apropiado para el logro de los mismos antes de emprender la acción"* (3). Efectivamente no es más que trazarse estrategias con el fin de cumplir con las metas deseadas para alcanzar los resultados esperados en el tiempo acordado y en concordancia con los recursos que se disponen. Es necesario hacer una planificación eficiente para evitar y predecir, en la medida de lo posible, futuros inconvenientes en el plan; para lograr esto se deben responder algunas preguntas específicas: ¿qué debe hacerse?, ¿quién va a ser el responsable? y ¿dónde, cuándo y cómo debe hacerse?

Tomando como punto de partida las definiciones anteriores, se asume en el presente trabajo la definición de planificación como un proceso cuyo propósito básico es lograr una adecuada coordinación de personas, recursos y mecanismos de una organización mediante planes, actividades y tareas para

obtener resultados relevantes sobre la base de la amplia y efectiva participación, aumentando las probabilidades de éxito.

Como resultado del proceso de la evolución de la planificación, surge la Dirección por Objetivos, determinando la proyección y consecución de las metas a alcanzar en la organización. Esta se encarga de establecer objetivos de acuerdo con los de la unidad jerárquica superior, siguiendo una lógica descendente y estos objetivos deben ser llevados a cabo por equipos y personas concretas. Se encarga del desarrollo institucional mediante la descripción de lo que se espera de cada cual, así como la medición y evaluación individual y colectiva, demandando la participación de todos, ayudando así a la motivación y a la responsabilidad de las personas involucradas, en términos de objetivos verdaderamente cumplidos. (4) Es una técnica sistemática que permite documentar los objetivos y darles seguimiento de manera formal posibilitando su mejora continua. En muchos países se combina la Dirección por Objetivos con la Planeación Estratégica, este último es el proceso a través del cual se declara la visión y la misión de la empresa, se estudia la situación interna y externa de ésta, se establecen los objetivos generales y se elaboran las estrategias y planes estratégicos necesarios para alcanzar dichos objetivos. (5) Esta se realiza por los directivos a nivel de organización, o sea, se enfoca en una visión general de la empresa y se proyecta a largo plazo basándose en planes, objetivos y estrategias generales que parecen sencillos pero que afectan un gran número de actividades; debe poseer un carácter flexible que permita cambios y ser analizado cada cierto tiempo como toda planeación.

1.2.2 Planificación por Objetivos

Es precisamente la Planificación por Objetivos el modelo definido a implementarse en Cuba, pero un modelo más personalizado, como resultado de la combinación de la Dirección por Objetivos y la Planeación Estratégica, buscando una adaptación teniendo en cuenta el estatus económico y social y las futuras proyecciones estratégicas del país. Este es un modelo asumido por los especialistas encargados del análisis del comportamiento de la planificación internacionalmente, sus tendencias, la influencia de sus aristas y su comportamiento a nivel nacional. Eduardo Martínez[2] en su “Evaluación de la Educación Superior” (6) plantea que comprende técnicas analíticas de planificación, empleo de presupuestos y

[2] Especialista Regional en Planificación y Gestión de Ciencia y Tecnología, UNESCO. Av. Brasil 2697, 11300, Montevideo, Uruguay; fax (598-2) 7072140; tel. 7072023;e-mail:emartinez@unesco.org.uy

técnicas de programación lineal e investigación de operaciones para controlar la producción y los procesos de distribución; es una forma particular de funcionamiento en torno a las tareas directivas. Esta propone una adecuada combinación de recursos, tiempo y personas como parte del mecanismo de la entidad u organización para obtener resultados relevantes sobre la base de la participación de todos y de forma efectiva. Esta definición es acogida por la presente investigación, agregándole también la definición de metas o fines estratégicos pero implementados operativamente mediante planes proyectados sobre áreas de resultados claves las cuáles enmarcan los objetivos a alcanzar y actividades a cumplir con su funcionamiento estatal[3] o propósito por el cual existe.

1.2.3 Medios y herramientas para planificar actividades y objetivos

Existen numerosas herramientas para efectuar la elaboración del plan de trabajo, administración y seguimiento de actividades regulares. Se pueden mencionar muchas opciones de sistemas destinados a la planificación de actividades, en ambientes escritorio, web, dirigidos a agendas electrónicas y a la telefonía móvil, entre otros. Algunas de estas herramientas son:

- **GanttProject:**

Una iniciativa de código abierto con ambiente escritorio para representaciones esquemáticas de la distribución de las tareas a corto, mediano y largo plazo. Permite distribuir las actividades por personas o recursos enmarcados en metas o proyectos. Esencialmente está destinada para la administración de proyectos con la amplia utilización de diagramas Gantt. No permite planificar por objetivos de acuerdo al modelo de planificación cubano, pues aunque permite la representación de metas, no contiene el potencial para representar los objetivos con sus criterios de medidas y las áreas de resultados claves.

- **OpenProj:**

También es de código abierto y ambiente escritorio en las que se pueden planificar detalladamente cada tarea, con tareas sucesoras y predecesoras, destinada a la gestión de proyectos de forma fácil, con buena interoperabilidad con Microsoft Project. Al igual que GanttProject tampoco se ajusta al modelo cubano de planificación.

[3] **Funcionamiento Estatal**: Actividades por la cual responde una persona u organización al estado.

❖**Excel, Outlook y Access:**

Herramientas del paquete Office de Microsoft, de fácil manejo. Utilizadas en entidades como los Organismos de Administración Central del Estado (OACE) y el Comité Ejecutivo del Consejo de Ministros (CECM). En el caso de Outlook gestionaban las actividades como eventos, luego de ser recibidas las mismas en una plantilla Excel con un formato determinado con tablas y filtros. Muchas veces su utilización, lejos de ser una solución, creaba un problema más grande ya que la información planificada que se recibía no coincidía con las clasificaciones establecidas y el filtrado de la información no se podía ejecutar adecuadamente. Como medio de planificación, tiene sus ventajas comparado al proceso manual en hojas, pero no permite elaborar la organización y menos llevar el seguimiento y control de la planificación como debe de ser.

❖**Redmine:**

Es una herramienta para la gestión de proyectos que incluye un sistema de seguimiento de incidentes con seguimiento de errores. Otras herramientas que incluye son calendario de actividades, diagramas de Gantt para la representación visual de la línea del tiempo de los proyectos, wiki, foro, visor del repositorio de control de versiones, control de flujo de trabajo basado en roles, integración con correo electrónico, entre otros. Es utilizado ampliamente en la Universidad de las Ciencias Informáticas para la planificación de proyectos y el plan de trabajo. A pesar de numerosas personalizaciones que pueden realizársele no responde al modelo de Planificación por Objetivos ni a sus principales procesos, no permite modelar las estructuras, trabaja directamente con usuarios, grupos y roles de acceso a funcionalidades.

❖**P-TRAB:**

Primera herramienta orientada al modelo de Planificación por Objetivos cubano. Toda una revelación en su momento ya que resolvía perfectamente las necesidades de los planificadores según la forma en que aplicaban para entonces la Planificación por Objetivos. El P-TRAB fue y es utilizado aun principalmente por las entidades pertenecientes a las FAR, mientras que otras tantas se auxiliaron y aun lo hacen en herramientas como Outlook y Excel del paquete de Herramientas Office. El P-TRAB diseñado solamente para MS-DOS, permite el registro de los objetivos con sus actividades de aseguramiento, pero presenta muchas desventajas en cuanto a escalabilidad del sistema, accesibilidad y disposición de la información. Este sistema no se pensó para brindar la información generada en el proceso de planificación mediante una red. No es posible acceder a la información deseada en el momento deseado y aunque cuenta con un

mecanismo de intercambio de información, basado en la generación de ficheros con extensión **dbf** a partir de la base de datos descentralizada que presenta, el solo hecho de no estar disponible en una red se salta una de las principales cuestiones con que debe contar un sistema informático para la planificación por objetivos, como el proceso de aprobación-conciliación, el cual regula totalmente el flujo de la información planificada, así como el intercambio de la misma.

❖**SIPAC**

Las herramientas anteriormente mencionadas no cumplen con todas las exigencias y expectativas para la informatización de la planificación por objetivos en las entidades, SIPAC mantiene un alcance bastante amplio en estas cuestiones, pensado como sistema multiplataforma, escalable y con prestaciones para funcionar en red. SIPAC cuenta con un mecanismo para efectuar el intercambio de información de diversas maneras, desde la más sencilla donde exista una red y una base de datos centralizada donde el acceso a la información está restringido totalmente por la compartimentación de la misma que ofrece el sistema, así como diferentes bases de datos tributando información entre ellas mediante réplicas online, o mediante la réplica offline para el caso donde no exista una red entre las bases de datos de las partes que participan en el intercambio de información. Pero, ¿acaso estos tipos de réplica permiten el intercambio de información de manera plena según todos los escenarios donde se solicite conocer determinada información? Precisamente, SIPAC no abarca el escenario donde el usuario o planificador común desea generar determinada información de manera sencilla, rápida, según sus criterios, necesidades y sin esperar por un súper usuario con privilegios de administración encargado de generar una réplica offline para obtener la información existente, esta situación genera cierto descontento por parte de los planificadores y de los usuarios generales del sistema.

1.2.4 El método Justo a Tiempo (JIT) como filosofía base para el intercambio de información

En la presente investigación se define el intercambio de información como el proceso en el que participan dos o más partes, clasificadas como partes generadoras o fuentes y partes receptoras o destino, donde una parte generadora puede convertirse también en receptora. Se puede definir como parte a una persona, entidad, empresa u organismo que participe en el proceso de planificación de actividades.

Si el intercambio de información tiene lugar en SIPAC e inclusive en las herramientas utilizadas anteriormente como P-TRAB, entonces ¿por qué los clientes o usuarios de dicho sistema aún quedan

insatisfechos con la informatización de este proceso mediante la programación de réplicas?, ¿cómo y cuándo tiene lugar este tipo de intercambio de información entre las partes necesitadas y qué hace necesario que SIPAC proporcione nuevas prestaciones con el objetivo de responder ante ellas? La Figura 1.1 muestra un ejemplo sencillo de cómo tiene lugar este proceso.

Figura 1.1 Esquema que representa de forma sencilla el intercambio de la información planificada entre dos entidades

En palabras, el planificador perteneciente a la Entidad 1 necesita hacerle conocer a su subordinado, el planificador de la Entidad 2, la planificación base por la cual la Entidad 2 debe de regir su plan ese año. Esta planificación es la agrupación de varios elementos, o sea, planes, algunos objetivos y actividades que involucran a la Entidad 2. Regido por un plazo de tiempo determinado, la información debe llegar a la Entidad 2 con tiempo para que sea analizada, consolidada y asegurada mediante la implementación de objetivos y actividades de aseguramiento. Una vez generada la planificación para el aseguramiento de la planificación del nivel superior, esta debe de ser aprobada por el jefe de la Entidad 2 y hacerla llegar a la Entidad 1 como parte del proceso de retroalimentación de la planificación entre un nivel superior (Entidad 1) y un nivel inferior (Entidad 2). (Fig.1.1). En resumen, existen varios factores que rigen este proceso de

intercambio de información entre entidades sin conexión mediante una red: qué tipo y cantidad de información se desea intercambiar, para quién o quiénes y en qué momento debe de ocurrir el intercambio. Teniendo en cuenta que no se puede utilizar el término “en tiempo real” ya que este término se emplea fundamentalmente para nombrar a los sistemas informáticos que interaccionan con su entorno físico, responden a los estímulos del entorno dentro de un plazo de tiempo determinado y no basta con que las acciones del sistema sean correctas, sino que, además, tienen que ejecutarse dentro de un intervalo de tiempo determinado. (7) Realmente este es un término que aunque pueda resultar versátil, no responde correctamente a esta situación; más bien responde al término “justo a tiempo”.

La terminología “Justo a Tiempo” o “Just in Time” (JIT por sus siglas en Inglés), responde a un método de organización de la producción, empleado por primera vez por las empresas japonesas durante los años que siguieron a la crisis de los setenta. Su idea esencial se basa en “producir los elementos que se necesitan, en las cantidades que se necesitan y en el momento en que se necesitan”. (7) En esta investigación se adoptan los principios de este método como filosofía base para responder ante la necesidad del usuario con una propuesta basada en decisiones inteligentes, ágiles, de rápido resultado, fácil entendimiento y operables. El método JIT plantea, como filosofía, dos cuestiones importantes: el hábito de ir mejorando y el de eliminar las prácticas desperdiciadoras; en otras palabras, el JIT persigue la continuidad en hacer las cosas de la mejor forma ya que más que un método productivo que se debe implantar, es una filosofía que se debe enseñar, con sus virtudes e inconvenientes. Por tanto, su adopción en esta investigación está apegada a la necesidad de poder brindarle al usuario un mecanismo de intercambio de información donde el mismo defina qué información (tipo y cantidad) desea generar o recibir y el momento en que le conviene hacerlo, haciéndolo el principal protagonista de sus acciones, minimizando así el tiempo de operación (es mucho más lento generar o copiar toda la información que existe en una base de datos, que parte de ella), minimizando la saturación de las bases de datos con información totalmente innecesaria para el proceso de planificación de dicho usuario, aunque esto siempre va a estar regido por el rol que juegue el usuario a la hora de hacer la extracción o recepción de la información y por último, donde dicho usuario pueda definir la vía de intercambio de información según sus intereses, pues la realidad de las entidades cubanas es contar con un mecanismo de intercambio de información sistema-sistema, sistema-sistema auxiliar y viceversa.

Se define como sistema en este caso a SIPAC y la comunicación sistema-sistema sería mediante un archivo generado por dicho sistema y que solo será legible la interpretación de la información mediante

SIPAC, un archivo que puede ser cifrado o no, para que en caso de que se desee visualizar la información del mismo con un editor de texto la información esté protegida. El sistema auxiliar sería cualquier herramienta como Outlook y Excel ya que SIPAC en su primer despliegue no llegará a todos los niveles, lo que traerá consigo la necesidad de contar con un mecanismo que permita generar la información y que pueda ser interpretada por herramientas auxiliares. En el caso de Excel, como herramienta auxiliar, que permita que la información sea visualizada de forma natural por el planificador y que de esa misma forma pueda ser interpretada por SIPAC a través de un modelo diseñado, a petición de los clientes, por parte del equipo de desarrollo. Este modelo se definió para que existiera una plantilla única que fuese capaz de brindar información legible y entendible a los usuarios y que al mismo tiempo pudiese ser interpretado por el sistema ya que las plantillas que existían anteriormente solo tenían significado para los panificadores. En el caso de Outlook para manejar estándares internacionales como los archivos con extensión “ics” o “vcs”. La Figura 1.2 muestra lo planteado anteriormente.

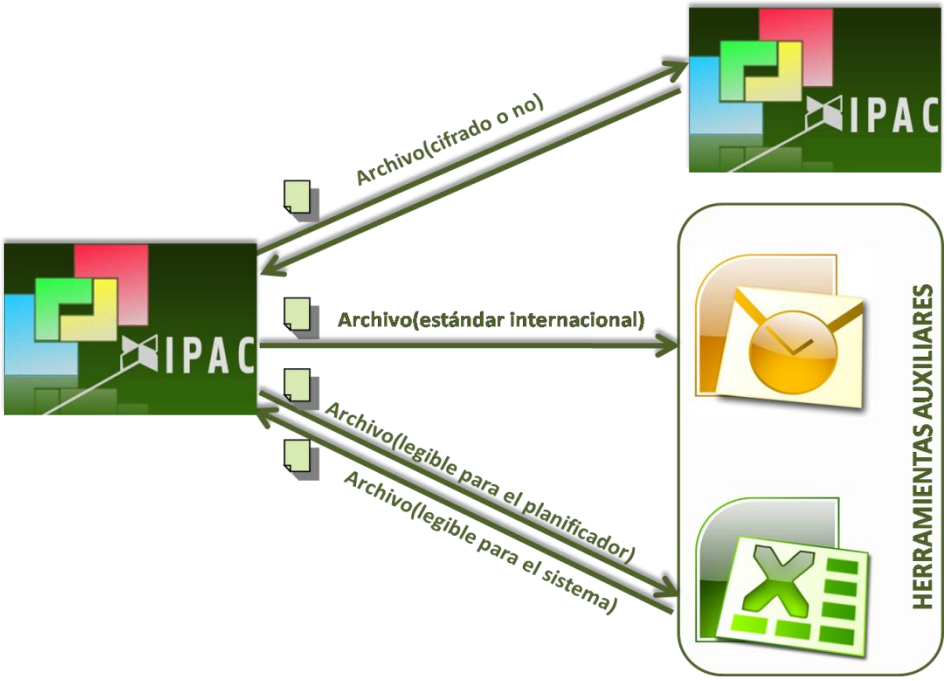


Figura 1.2 Interacción sistema-sistema y sistema-sistema auxiliar

A modo de resumen, SIPAC necesita contar con prestaciones que le permitan al usuario intercambiar la información planificada de forma sencilla y rápida, que le permita seleccionar la información deseada en el momento deseado, teniendo en cuenta que la realidad de las entidades cubanas hoy en día es que no

cuentan en su gran mayoría con la infraestructura necesaria para proporcionar una red que posibilite el intercambio de información en tiempo real. Todo esto acentúa la necesidad de contar con la informatización del proceso de intercambio y actualización de la información planificada en las entidades cubanas, en el menor tiempo posible, con el volumen de información deseado y donde los planificadores sean los principales protagonistas.

1.3 Lenguajes a utilizar en el desarrollo de la solución

Uno de los aspectos más importantes dentro del desarrollo de la solución consiste en los lenguajes de modelado y desarrollo utilizados para el tratamiento de la solución al problema planteado, estos han sido propuestos y aprobados en consecuencia con las realidades del país por los organismos encargados UCID y CEIGE. De igual manera sucedió con las herramientas, tecnologías y el marco de trabajo para el desarrollo del Sistema para la Planificación de Actividades.

1.3.1 Lenguaje y Notación de Modelado

Según José Enrique González Cornejo, Gerente General de DocIRS, “El lenguaje de modelado es la notación (principalmente gráfica) que usan los métodos para expresar un diseño. El proceso indica los pasos que se deben seguir para llegar a un diseño.” (8) La estandarización es sumamente importante pues permite la comunicación entre los agentes involucrados en el desarrollo; si se deseara discutir un diseño con una tercera persona este debe conocer el lenguaje y no el proceso a seguir para obtenerlo. Los lenguajes de modelado propuestos para la modelación de la solución son:

- **UML**

El Lenguaje de Modelado Unificado (Unified Modeling Language) aparece a fines de la década de 1980 y principios de la década de 1990 como una serie de métodos de análisis y diseño orientados a objetos. Pero no fue hasta octubre de 1995 que surgió el primer borrador, luego de que se unieran Grady Booch, Jim Rumbaugh e Ivar Jacobson, “los tres amigos”. (9) Según Booch, UML es un lenguaje para modelar, que es el procedimiento que emplean los ingenieros para el diseño de software antes de pasar a su construcción. Este lenguaje no solo es efectivo para sistemas informáticos, al contrario, es capaz de adaptarse a cualquier proyecto en el que se emplee. (10) Este lenguaje divide cada proyecto en un número de diagramas que representan las diferentes vistas del proyecto.

❖**BPMN**

Bussines Process Modeling Notation (BPMN) es la notación empleada para representar gráficamente las etapas del proceso de desarrollo de la solución que se propone y coordinar la secuencia de los procesos y los mensajes que fluyen entre los participantes de las diferentes actividades. Al contrario de UML, esta notación es orientada a procesos pero combinándolos entre sí se puede lograr una mayor exactitud a la hora de modelar la situación existente.

1.3.2 Lenguaje de Marcado

❖**XML**

El Extensive Markup Language se trata de un estándar del W3C[4] que posibilita compartir la información de una manera segura, fiable y fácil. Además, permite compartir los datos con los que se trabaja a todos los niveles, por todas las aplicaciones y soportes. (11) Se usa para la creación de reglas básicas que facilitan el intercambio de información estructurada entre aplicaciones.

1.3.3 Lenguajes del Lado del Cliente

Los lenguajes del lado del cliente son aquellos lenguajes que son asimilados directamente por el navegador y no necesitan pre tratamiento. Un lenguaje de lado cliente es totalmente independiente del servidor, lo cual permite que la página pueda ser albergada en cualquier sitio. Su principal ventaja es que se delega en el cliente la ejecución del trabajo evitando recargar al servidor. (12)

❖**HTML**

Hyper Text Markup Language (HTML) es decir, Lenguaje de Marcas de Hipertexto, es el lenguaje de programación utilizado para el desarrollo de las páginas clientes de la aplicación. Es sencillo y muy amigable para el desarrollador, está estructurado a través de etiquetas (tags) que permiten definir y organizar los elementos de las páginas. Permite la integración con otros lenguajes de programación como JavaScript y PHP mediante etiquetas. La interpretación del código de dicho lenguaje varía según el navegador web a emplear. (13)

[4]**El Consorcio World Wide Web (W3C):** comunidad internacional que guía la Web hacia su máximo potencial a través del desarrollo de protocolos y pautas.

❖**JavaScript**

Es un lenguaje de programación que se utiliza para crear páginas web dinámicas. Al ser un lenguaje interpretado brinda la posibilidad de probar lo implementado directamente en el navegador, sin necesidad de algún lenguaje externo para compilarlo. (14) Se usará mayormente para acceder a los elementos de la capa de presentación de la solución, para mejorar las interfaces de usuarios y para implementar instrucciones de respuesta a las posibles acciones de los usuarios.

❖**CSS**

Las Hojas de Estilo en Cascada (Cascading Style Sheet) serán utilizadas para todo lo referente a los estilos permitiendo un total control sobre el formato a emplear en la capa de presentación. Constituyen el estándar para la inserción de estilos a documentos estructurados, como por ejemplo, páginas HTML o XML. El objetivo de la definición de este estándar del W3C es permitir la separación entre las normas de presentación y el propio contenido a mostrar. (11)

1.3.4 Lenguajes del Lado del Servidor

Los lenguajes del lado del servidor son aquellos lenguajes que son reconocidos, ejecutados e interpretados por el propio servidor y que se envían al cliente en un formato comprensible para él. No es más que el procesamiento de una petición de un usuario mediante la interpretación de un script en el servidor web para generar páginas HTML dinámicamente como respuesta. (15)

❖**PHPv5.2**

PHP Hypertext Pre-Processor es un lenguaje de programación interpretado, usado para la creación de aplicaciones web dinámicas. PHP permite la conexión a numerosas bases de datos, incluyendo MySQL, Oracle, ODBC, etc. Puede ser ejecutado en la mayoría de los sistemas operativos (Windows, Mac OS, Linux, Unix), es de código abierto, gratuito y fácil de usar, que permite involucrarse con aplicaciones de contenido dinámico sin tener que aprender todo un nuevo grupo de funciones y prácticas. Su interpretación y ejecución se da en el servidor, en el cual se encuentra almacenado el script y el cliente sólo recibe el resultado de la ejecución. (16)

1.3.5 Marcos de Trabajo

Un marco de trabajo no es más que un conjunto de bibliotecas orientadas a la reutilización a gran escala de componentes software y código para el desarrollo rápido de aplicaciones. Logra que los componentes sean clases que pertenezcan a una gran jerarquía de clases, lo que resulta en bibliotecas más fáciles de aprender a usar. Son de gran utilidad para mantener una alta cohesión entre métodos de programación similares. (17)

❖**Sauxe**

Para el desarrollo de la solución que se propone se empleará el marco de trabajo Sauxe. Es una tecnología desarrollada en la UCI, hecha para garantizar aspectos como la seguridad, la multi-entidad, el multi-tema, el multi-idioma, la auditoría, la integración, la interoperabilidad, la concurrencia, la administración de transacciones, entre otros. Para las restricciones de diseño que asume este marco tecnológico, reutiliza las tecnologías como Zend Framework para el manejo de la lógica de negocio, Doctrine para el acceso a datos y ExtJS para la capa de presentación, además está fusionado bajo tecnología totalmente libre (PHP, Postgresql, Apache, entre otras). Este marco de trabajo permite desarrollar sistemas con calidad y estandarizar el proceso de desarrollo. (18)

❖**ExtJS 2.2**

ExtJS es una librería JavaScript que permite construir aplicaciones complejas en internet, además de flexibilizar el manejo de componentes de la página como el DOM[5], comunicación con el servidor usando AJAX, permite crear interfaces de usuario bastante funcionales que es principalmente para lo cual se emplea, posee numerosas funcionalidades que permiten añadir interactividad a las páginas HTML. Una de las grandes ventajas de utilizar ExtJS es que permite crear aplicaciones complejas utilizando componentes predefinidos. Se distribuye la carga de procesamiento Cliente- Servidor, permitiendo que el servidor pueda atender más clientes al mismo tiempo. (19)

❖**Zend Framework**

[5] **DOM:** Document Object Model es una interfaz de plataforma y lenguaje neutral que permitirá a los programas y scripts acceder y actualizar dinámicamente el contenido, estructura y estilo de los documentos. (48)

Para el manejo de la lógica del negocio se utilizará el marco de trabajo Zend Framework. Es un marco de trabajo de código abierto, orientado a objeto; está diseñado para la versión 5 de PHP. Posee muchos componentes acoplados que se pueden utilizar o no independientes. Establece una estructura básica utilizando los componentes del patrón Modelo-Vista-Controlador (MVC). Posee componentes con estructuras para AJAX, para facilitar peticiones y dar respuestas basadas en AJAX y otros componentes para facilitar la composición de vistas. (20)

❖**Doctrine**

Doctrine se utilizará para trabajar con un esquema de base de datos como si fuese un conjunto de objetos y no de tablas y registros. Está inspirado en Hibernate, que es uno de los ORM[6] (en inglés Object Relational Mapper) más populares y grandes que existen; brinda una capa de abstracción de la base de datos muy completa, sin necesidad de duplicar códigos, facilitando un mejor trabajo a los diseñadores SQL. (21)

1.3.6 Tecnologías y Herramientas de Desarrollo

❖**AJAX**

Asynchronous JavaScript And XML permite mejorar completamente la interacción del usuario con la aplicación, evitando las recargas constantes de la página, ya que el intercambio de información con el servidor se produce en un segundo plano. En sí misma no es una tecnología, son varias tecnologías independientes unidas entre sí. Se puede nombrar a XHTML[7] y CSS para la capa de presentación, DOM para la interacción y manipulación dinámica de la presentación, XML, XSLT[8] y JSON[9] para intercambiar y manipular información y JavaScript para unir estas tecnologías. (22)

❖**SQLite**

[6] **ORM**: Componente de software que permite trabajar con los datos persistidos como si fueran parte de una base de datos orientada a objetos.
[7]**XHTML:** Lenguaje de Marcado de Hipertexto Extensible, extiende HTML 4.0 combinando la sintaxis de HTML, diseñado para mostrar datos, con la de XML, diseñado para describir los datos.
[8]**XSLT**: (Extensible Stylesheet Language Transformations o lenguaje de hojas extensibles de transformación), que permite convertir documentos XML de una sintaxis a otra (por ejemplo, de un XML a otro o a un documento HTML)
[9]**JSON:** acrónimo de JavaScript Object Notation, es un formato ligero para el intercambio de datos.

SQLite es un proyecto de dominio público creado por Dwayne Richard Hipp que implementa una pequeña librería de aproximadamente 500Kb programada en lenguaje C, que funciona como un sistema de gestión de base de datos relacionales. A diferencia de otros motores de base de datos este es independiente, pues no se comunica con un motor de base de datos, sino que sus librerías pasan a integrar la aplicación. Es más rápido que MySQL y PostgreSQL y cuenta con diferentes interfaces que permiten trabajar con C++, PHP, Perl, etc. Otra de las ventajas que posee es que es portable, se ejecuta en muchas plataformas y sus bases de datos pueden ser fácilmente portadas sin ninguna configuración o administración. (23)

❖**Herramienta CASE: Visual Paradigm for UML**

Se utilizará como herramienta CASE[10] Visual Paradigm for UML 6.0 que es una herramienta de modelado visual y utiliza como el lenguaje de modelado UML 2.1. Esta herramienta tiene unas características gráficas muy cómodas que facilitan la realización de diagramas de modelado que siguen el estándar de UML integrando sus tutoriales, demostraciones interactivas y proyectos UML; lo que ayuda a construir la aplicación mejor, con calidad y rapidez. Permite diseñar código inverso, generar código desde diagramas y generar documentación. (24)

❖**Sistema de control de Versiones: Subversion**

La herramienta para el control de versiones que se utilizará es Subversion (SVN) en su versión 1.6.5. Es el sistema más utilizado para el control de versiones ya sea de código fuente, páginas o documentación. Maneja los archivos, las carpetas y sus modificaciones en el transcurso del tiempo. SVN es libre y Open Source[11] ya que está distribuido bajo la licencia Apache. Cada vez que se modifique un archivo y se envíe al repositorio, este creará una copia de la versión anterior y así ninguna de las copias anteriores se perderá. (25)

❖**Entorno Integrado de Desarrollo(IDE): NetBeans**

La solución de este problema será desarrollada sobre el IDE multipropósito NetBeans 6.9 integrado a la versión 5 de PHP, ExtJS, el diseño de Hojas de Estilo (CSS) y el lenguaje HTML, este IDE examina todos

[10]**CASE:** es un conjunto de programas y ayudas que dan asistencia a los analistas, ingenieros de software y desarrolladores, durante todos los pasos del Ciclo de Vida de desarrollo de un sistema informático.
[11]**Open Source:** término con el que se conoce al software distribuido y desarrollado libremente.

los directorios, haciendo carga de clases, métodos y objetivos, para acelerar la programación. Desde el editor es posible realizar la administración del sistema de control de versiones Subversion. Todos sus beneficios y funcionalidades son obtenidos gratuitamente pues es un producto de código abierto. (26)

❖**Servidor Web: Apache**

Como servidor Web se utilizará Apache 2.0. Es una versión más rápida y más estable que tiene la infraestructura necesaria para servir distintos protocolos y está estructurado en módulos. Ofrece la posibilidad de que los webmasters[12] puedan configurar las respuestas que muestra el servidor Apache cuando se producen algunos errores o problemas. Es uno de los servidores más ampliamente usados, es de código abierto y multiplataforma. (27)

❖**Sistema de Gestor de Bases de Datos: PostgreSQL**

Como sistema manejador de Base de Datos se usará la herramienta multiplataforma PostgreSQL en su versión 8.3.3. El código fuente de esta herramienta está disponible bajo una licencia de código abierto, publicado bajo la licencia BSD[13] (Berkeley Software Distribution). Es una herramienta que tiene soporte completo para subconsultas, ejecutar consultas complejas y consultas sobre vistas. Su integridad de datos incluye claves principales y claves externas, además tiene soporte para uniones, vistas y procedimientos almacenados de gran tamaño. Cuenta con varias herramientas gráficas de alta calidad para administrar las bases de datos (pgAdmin) y para hacer diseño de bases de datos. (28)

1.3.7 Modelo de Desarrollo

La presente investigación se acoge al empleo del modelo de desarrollo utilizado en la UCID. Este modelo se encuentra descrito en el documento Proceso de Desarrollo y Gestión de Proyectos de Software versión primera y es el resultado de la combinación de modelos que poseen como características principales su basamento en Componentes, Iterativo e Incremental. Dentro de sus aspectos más importante presenta el incremento del resultado parcial del sistema ya probado, integrado y estable, así como una mayor reutilización de las soluciones. Los procesos de este modelo se representan a continuación:

[12]**Webmaster:** persona responsable de mantenimiento o programación de un sitio web.
[13]**BSD**: Es la licencia de software otorgada principalmente para los sistemas BSD (Berkeley Software Distribution o Distribución de Software Berkeley).

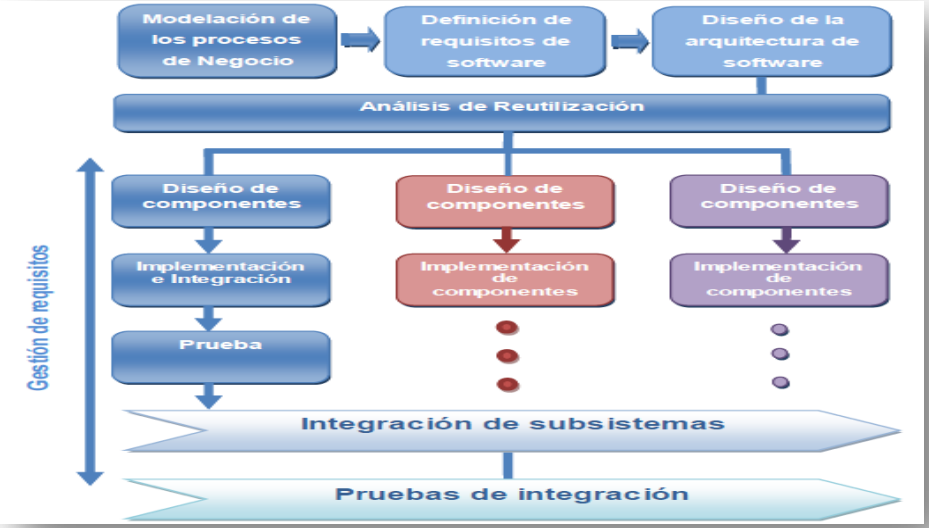


Figura 1.3 Procesos del Modelo de Desarrollo a utilizar

Cada proyecto tiene un ciclo de vida regido por una serie de fases en forma de secuencia, una a continuación de la otra:

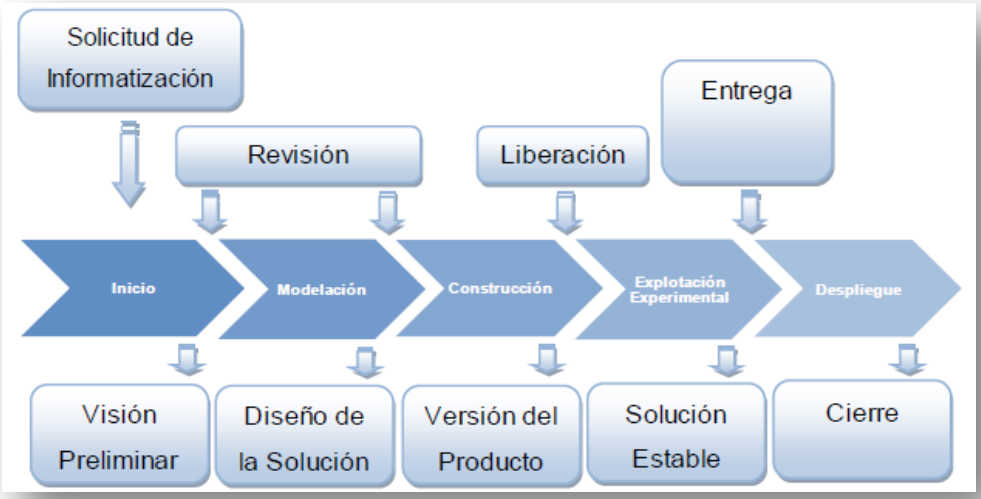


Figura 1.4 Fases o etapas del ciclo de vida de un proyecto

1.4 Conclusiones del capítulo

En este capítulo se expuso el estado del arte referente al surgimiento de la planificación, su evolución, adaptación y aplicación en las entidades cubanas. Se conceptualizaron algunos métodos de planificación empleados en el mundo como la Dirección por Objetivos y la Planeación Estratégica, identificándose a la Planificación por Objetivos como el modelo creado en Cuba para llevar a cabo la planificación de actividades. Se nombraron las principales herramientas destinadas a la planificación en el mundo pero que no están acorde al modelo de planificación cubano y las deficiencias de las empleadas en Cuba. Se identificó la carencia de un proceso de intercambio y actualización de la información de manera ágil y sencilla acorde a las necesidades de los clientes.

Capítulo II: Análisis y diseño de la solución

2.1 Introducción

La descripción de las características esenciales que debe tener la solución que se propone es el punto de partida del presente capítulo, las cuales se encuentran respaldadas por la modelación de los procesos del negocio; se muestran algunos de los artefactos propuestos en la fase de Modelación definida en el modelo de desarrollo aplicado. Se describe y fundamenta la línea base de la arquitectura, se precisan los estilos y corrientes arquitectónicas adoptadas y la estructura de empaquetamiento sobre los cuales se concebirá la solución propuesta. Los requerimientos funcionales se llevan a procedimientos minuciosamente diseñados para su posterior implementación. Se fundamentan los patrones empleados en la modelación del diseño y se modelan los diagramas que describen las relaciones entre las clases del diseño, el modelo de datos y la comunicación entre los componentes de SIPAC.

2.2 Descripción del Negocio

Dada las características del modelo económico cubano es de suma importancia mantener una planificación consecuente con los recursos de los que se dispone. Este proceso está regido por lo planteado en la Instrucción No.1 del Presidente de los Consejos de Estado y de Ministros para la planificación de los objetivos y actividades en los órganos, organismos de la administración central del estado, entidades nacionales y las administraciones locales del Poder Popular, donde se plasma la trayectoria que debe seguir toda planificación y se llama a la necesidad de ser rigurosos en el cumplimiento de la misma. Los procesos concepción y ejecución del plan juegan un papel fundamental ya que mediante ellos se elaboran todos los elementos a cumplir y se asignan a áreas específicas, se elaboran objetivos y para darle cumplimiento a estos se crean actividades. Acto seguido viene un proceso aprobación-conciliación que consiste en pasar a través de los diferentes niveles dichos elementos para aprobar los que sean posibles de cumplir o se rechazan los que no. Estos rechazados se vuelven a reformular hasta su posterior aprobación o se desechan completamente en caso de no ser factible su ejecución. Los aprobados pasan a formar parte inmediatamente de la planificación iniciando así el proceso de ejecución. Luego queda velar por el correcto cumplimiento de dichos elementos para lograr que los objetivos se alcancen y evitar percances o malas implementaciones.

El intercambio y actualización de la información entre los diferentes niveles dentro de los órganos de planificación se encuentra relacionado con la necesidad intrínseca que debe tener cada herramienta informática, destinada a la planificación, de permitir este intercambio y en especial en el país que no cuenta con una infraestructura entre todas las entidades para mantener una comunicación efectiva y ágil.

Una vez que comienza la etapa de planificación se analizan estrategias para la planificación y se evalúan los riesgos y las posibilidades que pueden afectarla, se definen la misión, visión y la proyección de la misma; se obtiene una propuesta de Documento Rector[14] que luego de ser conciliada, se reformula y se emite de manera oficial el Documento Rector que va a regir el proceso de planificación en ese período. Luego se hace necesario intercambiar dicha información con otros niveles. Cuando en los otros niveles es recibida dicha información se estudian las proyecciones de la planificación que se deben cumplir y si es posible cumplirlas para poder concebir, en respuesta a lo definido por el nivel superior y las necesidades internas del nivel, los elementos de la planificación. Esta propuesta se envía nuevamente a conciliar o aprobar para los niveles superiores, en caso de existir, para ser analizada y si es aprobada y es relevante se incorpora dicha información a la planificación sobrescribiendo la existente o agregándola en caso de que sea nueva. En caso de no ser posible cumplir con las proyecciones propuestas en el Documento Rector se elabora una propuesta de proyección de planificación que se envía para que sea evaluada por el nivel superior. Como se puede apreciar, se hace muy engorroso llevar a cabo el proceso de intercambio y actualización de la información entre los distintos niveles y es por eso que se hace sumamente necesario contar con una herramienta que se encargue de informatizar este intercambio.

2.2.1 Propuesta de Solución

SIPAC no posee aún una funcionalidad que permita el intercambio de información de manera ágil, segura y eficiente en el momento justo en que el usuario lo requiera. Con el presente trabajo de diploma se aboga por incorporar el componente Gestor de Intercambio de Información JIT, cumpliendo con los principios de integridad, confidencialidad y disponibilidad de la información.

El desarrollo de esta solución garantiza que se mantenga la escalabilidad de SIPAC ya que es orientada a componentes y permitirá al usuario o planificador común generar o hacerse de determinada información

[14] **Documento Rector**: Disposición de rango superior que contiene las medidas organizativas, áreas de resultados claves, objetivos, actividades principales y otros, que sirven de base para que los niveles inferiores elaboren el plan.

de manera sencilla, rápida, según sus criterios, necesidades y sin esperar por una persona con determinados privilegios de administración que genere una réplica offline para obtener la información existente.

2.3 Modelación del Negocio

2.3.1 Diagramas de Proceso de Negocio (BPMN)

Se modela el proceso de negocio que da lugar al componente Gestor de Intercambio de Información JIT usando la notación para la representación de procesos de negocio (BPMN), permitiendo representar gráficamente las etapas del proceso de desarrollo de la solución que se propone y coordinar la secuencia de los procesos entre los participantes de las diferentes actividades.

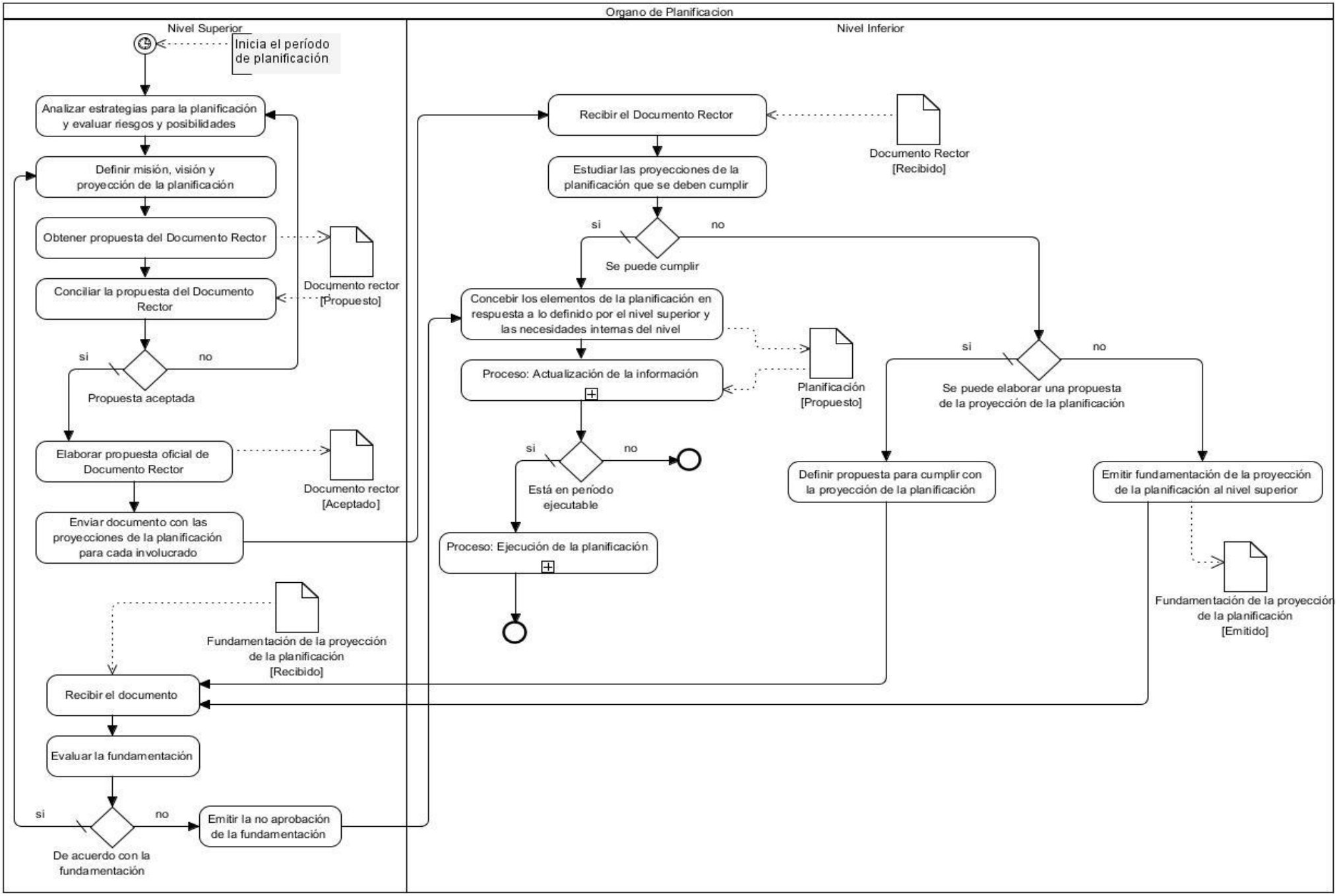


Figura 2.1 Diagrama de Proceso de Negocio Intercambio de Información

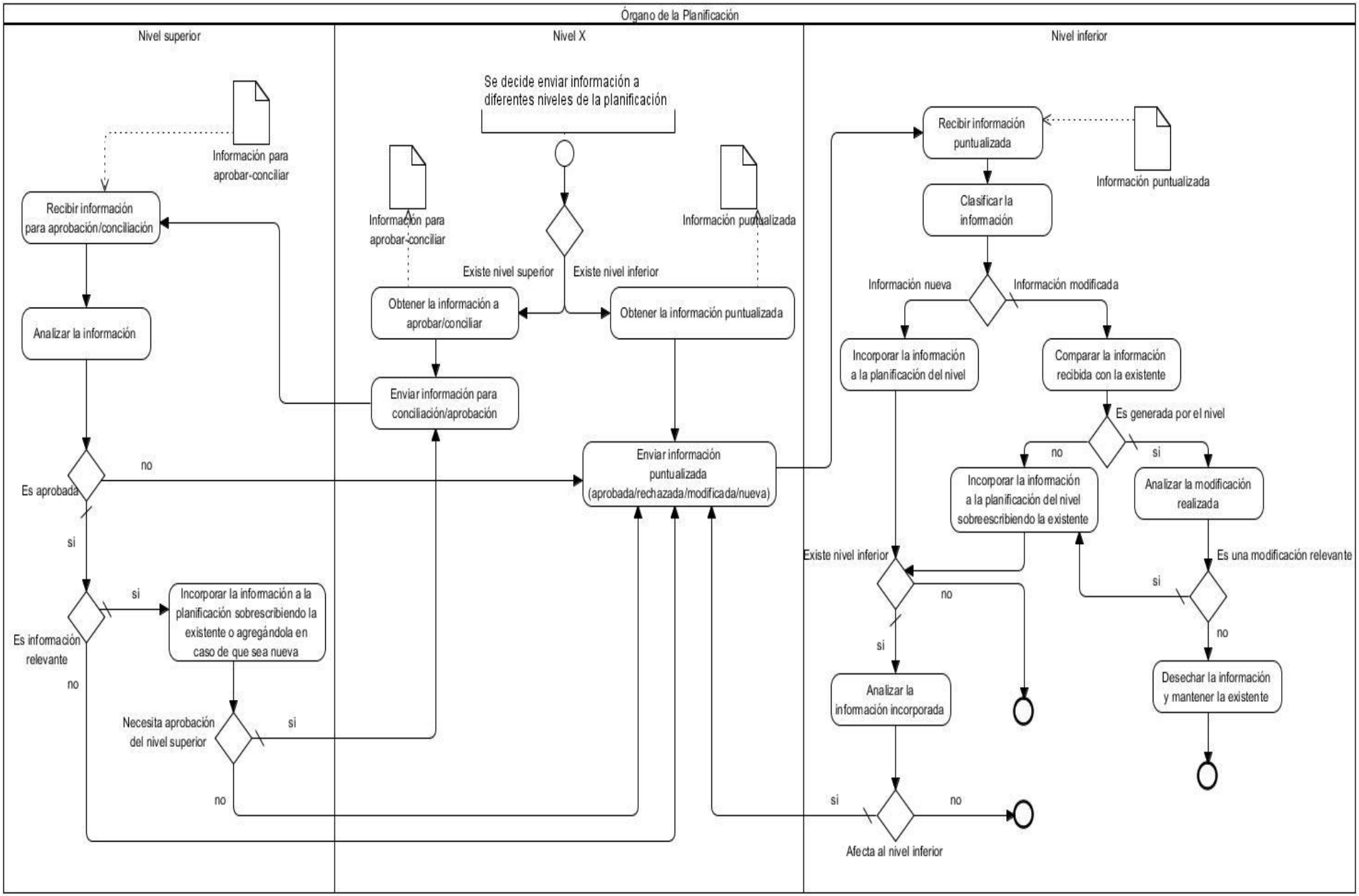


Figura 2.2 Diagrama de Proceso de Negocio Actualización de la Información

2.3.2 Especificación de procesos de negocio

Tabla 1: Proceso de Intercambio de Información

Objetivo	Definir y aprobar la información que tributa al plan que define la entidad.
Evento(s) que lo genera(n)	Inicia el período de planificación.
Pre condiciones	Se ha emitido una documentación que orienta el inicio del período de planificación.
Marco legal	N/A.
Clientes internos	Ejecución de la planificación. Actualización de la Información.
Clientes externos	N/A.
Entradas	Documento Rector (Documento Word). Propuesta de Planificación (Documento Word). Fundamentación de la Proyección de la Planificación (Documento Word).
Flujo de eventos	
Flujo básico	
1.	Analizar estrategias para la planificación y evaluar riesgos y posibilidades. Se analiza cómo explotar las fortalezas que posee la organización para cumplir con el plan a establecer.
2.	Definir misión, visión y proyección de la planificación. Se definen cuáles van a ser los principales objetivos y el alcance de la planificación a implementar.
3.	Obtener propuesta del Documento Rector. El documento en formato impreso o digital se circula a todos los involucrados del mismo nivel de dirección que participarán en el intercambio de información.
4.	Conciliar la propuesta del Documento Rector. Este se concilia para analizar si es aceptado o no por los encargados de elaborar el Documento Rector que guiará el proceso de planificación. En caso de que la propuesta no sea aceptada se procederá a ejecutar la actividad 1 del flujo básico: Analizar estrategias para la planificación y evaluar riesgos y posibilidades.
5.	Elaborar propuesta oficial de Documento Rector. Se procede a hacer oficial dicha propuesta.
6.	Enviar documento con las proyecciones de la planificación para cada involucrado. El Documento Rector oficial se envía a los niveles inferiores.
7.	Recibir el Documento Rector. Es recibido el documento rector en el nivel inferior.
8.	Estudiar las proyecciones de la planificación que se deben cumplir. El documento es estudiado minuciosamente por parte de todos los involucrados del nivel para ver si se pueden cumplir todas las proyecciones de la planificación. Si no se pueden cumplir las proyecciones de la planificación ver el flujo alternativo 8.a.
9.	Concebir los elementos de la planificación en respuesta a lo definido por el nivel superior y las necesidades internas del nivel.
10.	Proceso: Actualización de la información. Después de llevado a cabo el subproceso Actualización de la información, se verifica si está en tiempo del período ejecutable. Si no está en tiempo del período ejecutable ver la actividad 12 del flujo básico.

11.	Proceso: Ejecución de la planificación. Donde se ejecuta la planificación.
12.	Concluye el proceso.
Pos-condiciones	
1.	N/A.
Salidas	
1.	N/A.
Flujos paralelos	
1.	N/A.
Flujos alternos	
Flujo alternativo 8.a No se cumplen todas las proyecciones de la planificación	
1.	Se verifica si se puede elaborar una propuesta de la proyección de la planificación.
2.	Definir propuesta para cumplir con la proyección de la planificación.
3.	Recibir documento.
4.	Evaluar la fundamentación.
5.	Se verifica si está de acuerdo con la fundamentación. Si está de acuerdo se ejecuta la actividad 2 del flujo básico: Definir misión, visión y proyección de la planificación.
Pos-condiciones	
1.	N/A.
Salidas	
1.	N/A.
Flujo alternativo 8.a.1 No se puede elaborar una propuesta de la proyecciones de la planificación	
1.	Emitir fundamentación de la proyección de la planificación al nivel superior. Se procede a ejecutar la actividad 3: Recibir documento del flujo alternativo 8.a No se cumplen todas las proyecciones de la planificación.
Pos-condiciones	
1.	N/A.
Salidas	
1.	Fundamentación de la proyección de la planificación.
Flujo alternativo 8.a.5 No está de acuerdo con la fundamentación	
1.	Emitir la no aprobación de la fundamentación. Se procede a ejecutar la actividad 9 del flujo básico: Concebir los elementos de la planificación en respuesta a lo definido por el nivel superior y las necesidades internas del nivel.
Pos-condiciones	
1.	N/A.
Salidas	
1.	N/A.
Asuntos pendientes	
1.	N/A.

Descripción del Proceso de Actualización de la Información ver Anexo 1.

2.3.3 Modelo Conceptual

La siguiente figura representa el modelo conceptual del componente Gestor de Intercambio de Información el cual se basa en el modelo conceptual de SIPAC, los conceptos fundamentales con los que se va a trabajar se encuentran descritos en el expediente de proyecto en el documento CIG-ERP-N-DPO-i2201.

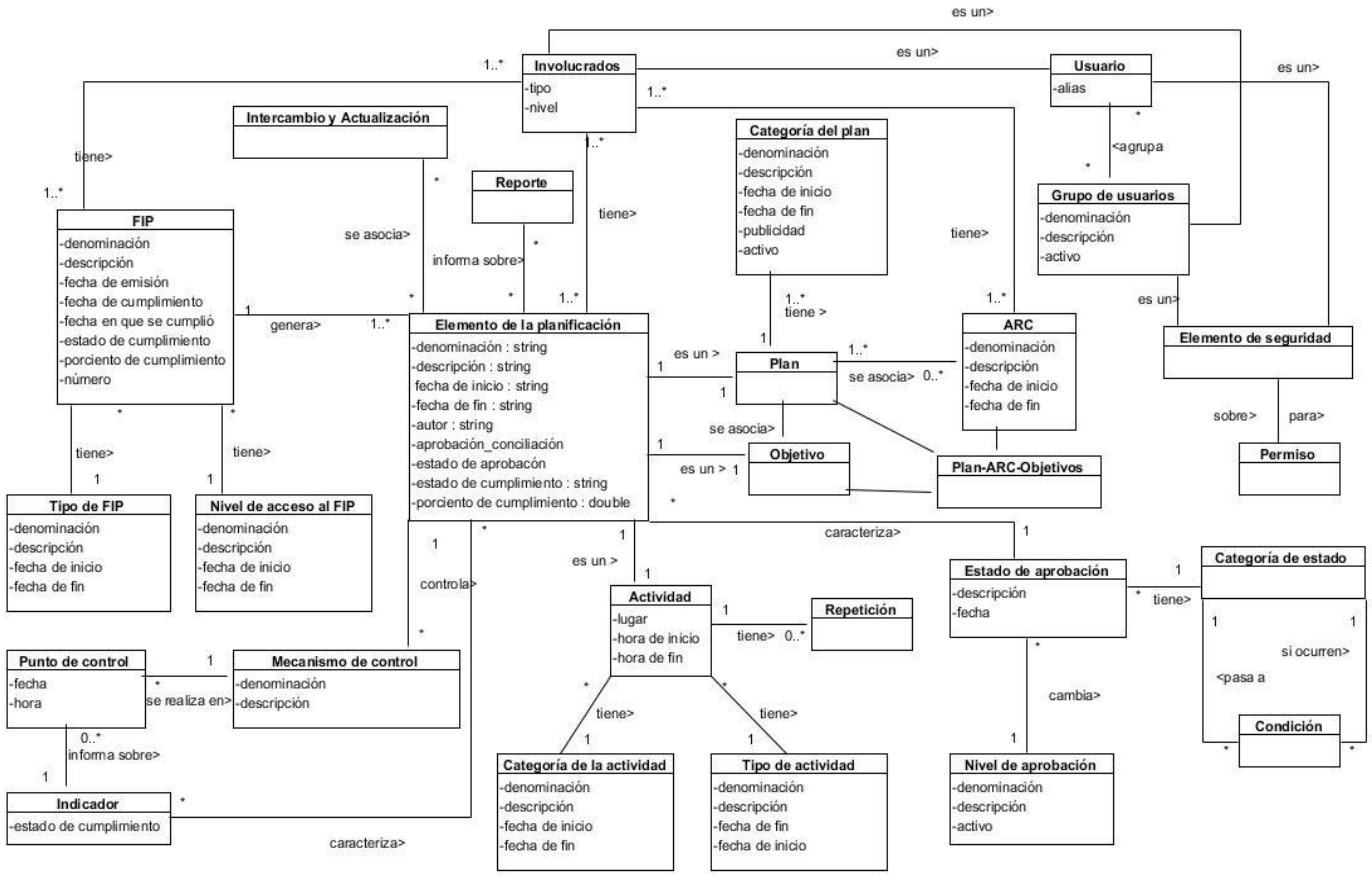


Figura 2.3 Modelo Conceptual

2.3.4 Requerimientos

Un Requerimiento es una condición o capacidad requerida por un usuario para resolver un problema o alcanzar un objetivo y que debe ser poseída por un sistema o componente del sistema para satisfacer un contrato, estándar, especificación u otro documento formal. (29) Los requerimientos deben ser apropiadamente documentados para su posterior cumplimiento.

2.3.5 Técnicas de Captura de Requisitos

La captura de requisitos es una actividad más humana que técnica, mediante la que el equipo de desarrollo de un sistema de software extrae de cualquier fuente de información disponible, las necesidades que debe cubrir dicho sistema. (30)

Para extraer los requisitos del sistema se utilizaron las siguientes técnicas:

Entrevistas: Se utiliza de manera frecuente en los diferentes encuentros con el cliente, dígase funcionales o especialistas de SIPAC. Se realizaron preguntas con el objetivo de obtener toda la información posible sobre la visión que el entrevistado tiene de los requisitos y comprender los propósitos de la solución buscada.

Tormenta de ideas: Se realizan reuniones y encuentros con todos los involucrados en el desarrollo del sistema donde cada cual expresa sus ideas y criterios. Su objetivo fundamental es dar una visión general de las necesidades del sistema.

Talleres: También consiste en la realización de reuniones con los involucrados en el desarrollo del sistema, estos encuentros son más organizados donde se realiza una preparación previa y es dirigida por un experto (en este caso el funcional o jefe del proyecto SPA). Su objetivo fundamental es detallar cada requisito y sirve como base para la posterior especificación de los mismos.

2.3.6 Listado de Requisitos Funcionales Identificados

Requerimientos funcionales: Son declaraciones de los servicios que proveerá el sistema, de la manera en que éste reaccionará a entradas particulares. En algunos casos, los requerimientos funcionales de los sistemas también declaran explícitamente lo que el sistema no debe hacer. (31)

A continuación se muestran los requisitos capturados para la primera etapa del desarrollo de la solución que se propone.

Tabla 2: Requisitos Funcionales capturados para la primera etapa de desarrollo

RF1. Importar elementos:	**RF2. Exportar elementos:**	**RF3. Insertar elementos:**
1.1 Importar Plan.	2.1 Exportar Plan.	3.1 Insertar Plan.
1.2 Importar Objetivo.	2.2 Exportar Objetivo.	3.2 Insertar Objetivo.
1.3 Importar Actividad.	2.3 Exportar Actividad.	3.3 Insertar Actividad.
1.4 Importar ARC.	2.4 Exportar ARC.	3.4 Insertar ARC.
1.5 Importar FIP.	2.5 Exportar FIP.	3.5 Insertar FIP.

RF4. Actualizar elementos:	RF5. Comparar elementos:	RF6. Listar Formato
4.1 Actualizar Plan.	5.1 Comparar Plan.	
4.2 Actualizar Objetivo.	5.2 Comparar Objetivo.	
4.3 Actualizar Actividad.	5.3 Comparar Actividad.	
4.4 Actualizar ARC.	5.4 Comparar ARC.	
4.5 Actualizar FIP.	5.5 Comparar FIP.	

2.3.7 Especificación de Requisitos Funcionales

Para la especificación de requisitos se utiliza la plantilla definida por la dirección del proyecto ERP-Cuba, la cual presenta diferentes secciones para describir los flujos de eventos que conllevan a la ejecución de dicha funcionalidad en el sistema.

A continuación se exponen la especificación de los requisitos Exportar Plan a un archivo legible al sistema (.SIPAC, .XLS, .VCS o .ICS), e Importar Plan de un archivo legible al sistema (.SIPAC o .XLS).

Tabla 3: Especificación de Requisito Exportar Plan

Precondiciones	El usuario ha registrado al menos un plan en el sistema.
Flujo de eventos	
Flujo básico	
1	Se seleccionan los planes a exportar.
2	Se selecciona el formato al cual desea exportar.
3	Se selecciona si desea exportar el plan o los planes seleccionados con sus relaciones.
4	El sistema permite seleccionar la ubicación hacia donde se exportará la información y registrar el nombre.
5	El sistema valida (ver Validaciones) los datos introducidos.
6	El sistema exporta los datos del plan seleccionado con sus relaciones.
7	Concluye el requisito.
Pos-condiciones	
1	Se ha creado un archivo con los datos de los planes seleccionados y sus respectivas relaciones.
Flujos alternativos	
Flujo alternativo 2.a Información incompleta	
1	El sistema señala los datos vacíos y permite corregirlos.
2	El usuario corrige los datos.
3	Volver al paso 2 del flujo básico.
Flujo alternativo 3.a El usuario confirma que desea exportar el plan o los planes seleccionados sin sus relaciones.	
1	El sistema permite seleccionar la ubicación hacia donde se exportará la información y registrar el nombre.

2	El sistema valida (ver Validaciones) los datos introducidos.	
3	El sistema exporta los datos del plan seleccionado sin sus relaciones.	
4	Concluye el requisito.	
Pos-condiciones		
1	Se ha creado un archivo con los datos de los planes seleccionados sin sus relaciones.	
Flujo alternativo *.a El usuario cancela la acción.		
1	Concluye el requisito.	
Pos-condiciones		
1	No se crea el fichero.	
Validaciones		
1	Se debe exportar en formato .SIPAC, .XLS, .VCS o .ICS.	
2	La dirección hacia donde se exportará el fichero debe ser correcta.	
Relaciones	**Requisitos Incluidos**	N/A.
	Extensiones	N/A.
Conceptos	**Elemento de la planificación**	Visibles en la interfaz: Denominación Fecha de inicio Fecha de fin Estado de aprobación Estado de cumplimiento Observ. Utilizados internamente: N/A.
	Plan	Visibles en la interfaz: N/A. Utilizados internamente: N/A.
Requisitos especiales	N/A.	
Asuntos pendientes	N/A.	

Tabla 4: Especificación de Requisito Importar Plan

Precondiciones	N/A.
Flujo de eventos	
Flujo básico	
1	El sistema permite seleccionar el fichero a importar.
3	El sistema valida (ver Validaciones) que sea el fichero correcto.
4	El sistema importa los datos: Denominación Categoría del plan Descripción

	Fecha de inicio Fecha de fin Autor Estado de aprobación Estado de cumplimiento Porciento de cumplimiento Observ.
5	El sistema muestra los datos de la importación en la pestaña “Elementos que coinciden con la información existente”.
6	Datos que se muestran en la pestaña “Elementos que coinciden con la información existente”: CAR Denominación Autor
7	Se selecciona uno de los elementos listados.
8	El sistema muestra una comparación entre los datos a importar que coincidan con los datos que existen en el sistema.
9	El sistema solicita confirmación si se desea sobrescribir la información de los elementos del fichero.
10	El usuario confirma si desea sobrescribir la información de los elementos del fichero.
11	El sistema sobrescribe la información de los elementos del fichero.
12	El sistema importa los datos del plan con la información de los elementos del fichero.
13	El sistema muestra el listado de elementos importados.
14	Concluye el requisito.
Pos-condiciones	
1	Se sobrescribieron los datos registrados en el sistema por los datos contenidos en el fichero.
Flujos alternativos	
Flujo alternativo 10.a El usuario no desea sobrescribir la información de los elementos del fichero.	
1	El sistema importa los datos del plan totalmente nuevos.
2	El sistema muestra el listado importado.
3	Concluye el requisito.
Pos-condiciones	
1	Se registraron nuevos planes en el sistema.
Flujo alternativo *.a El usuario cancela la acción.	
1	Concluye el requisito.
Pos-condiciones	
1	N/A.
Validaciones	
1	Se puede importar desde los formatos .SIPAC o .XLS.
2	La dirección desde donde se importará el fichero debe ser correcta.

Relaciones	**Requisitos Incluidos**	El sistema muestra el listado importado: Listar planes, en la agrupación Gestionar plan.
	Extensiones	N/A
Conceptos	**Elemento de la planificación**	Visibles en la interfaz: Denominación Fecha de inicio Fecha de fin Estado de aprobación Estado de cumplimiento Observ. Utilizados internamente: N/A.
	Plan	Visibles en la interfaz: N/A. Utilizados internamente: N/A.
Requisitos especiales	N/A.	
Asuntos pendientes	N/A.	

2.3.8 Requerimientos no funcionales

Los requerimientos no funcionales son aquellos que no se refieren directamente a las funciones específicas que entrega el sistema, sino a las propiedades emergentes de éste como la fiabilidad, la respuesta en el tiempo y la capacidad de almacenamiento. Los requerimientos no funcionales a menudo se aplican al sistema en su totalidad. (32)

Requisitos no funcionales del componente implementado:

1. Usabilidad (USB)

1.1. El sistema podrá ser usado por cualquier usuario con conocimientos básicos de computación.

2. Rendimiento (REN)

2.1. Se intercambian datos con el sistema ya sea mediante acciones de inserción, búsqueda, eliminación o modificación de datos, en un servidor de 1 GB de memoria RAM y se recibe la notificación de la acción realizada en un período de 0.1 a 0.7 segundos.

3. Seguridad (SEG)

3.1. La información que se maneje en el sistema estará protegida de acceso no autorizado, a partir de los diferentes roles de los usuarios que empleen el sistema.

3.2. Si se desea eliminar o modificar un elemento de la base de datos que está siendo utilizado por otro elemento que depende de él; el sistema no permite que este elemento sea eliminado. (Borrar en cascada[15]) (Seguridad[16]/ Integridad[17]).

4. Interfaz (INU)

4.1. El sistema debe contar con una interfaz fácil, amigable y sencilla que permita a los usuarios finales del mismo interactuar con este aun teniendo conocimientos básicos.

2.4 Modelación del Sistema

2.4.1 Arquitectura de la Solución

La arquitectura de la solución propuesta se basa en la del sistema Cedrux, definida en las orientaciones del departamento de Tecnología de CEIGE y está estructurada fundamentalmente en tres vistas que pueden descomponerse en otras. La siguiente jerarquía muestra la organización utilizada para la representación de la arquitectura de software de la solución propuesta:

- **Tecnología**: Define la plataforma tecnológica sobre la cual se desarrolla el sistema, se especifican los marcos de trabajo y herramientas bases.
- **Seguridad**: Chequea e implementa todos los aspectos relacionados con el acceso a la aplicación, la modificación, lectura o eliminación de la información.
- **Presentación**: Conforma el aspecto visual del sistema, dígase colores, botones, vínculos y todos los elementos significativos de vista de la presentación.
- **Sistema**: Define los tipos de componentes que conforman el sistema, así como sus características y composición estructural interna (vista vertical de la arquitectura).
- **Integración**: Se ocupa de los procesos de integración interna (entre componentes de un mismo proyecto) y externa (entre proyectos distintos), establece las definiciones, estándares, protocolos de comunicación y reglas de intercambio de información.

[15] **Borrar en cascada:** Esta opción le indica al sistema gestor de la base de datos que cuando se elimina un registro de la tabla principal automáticamente se borran también los registros relacionados en la tabla secundaria.
[16] **Seguridad:** Atributo de calidad que indica el grado en que un acceso no autorizado (accidental o deliberado) se prevenga.
[17] **Integridad:** Atributo de calidad que indica la ausencia de alteraciones inapropiadas de la información.

- **Datos**: Elabora todas las definiciones a nivel de datos, la integración de los distintos modelos, de los patrones, estándares y definiciones a este nivel.
- **Desarrollo**: Define la plataforma tecnológica a utilizar en la confección del producto, así como la disponibilidad de los distintos servicios telemáticos.
- **Despliegue**: Identifica los requerimientos de dispositivos necesarios para la implantación del sistema y los distintos escenarios de despliegue posibles. (33)

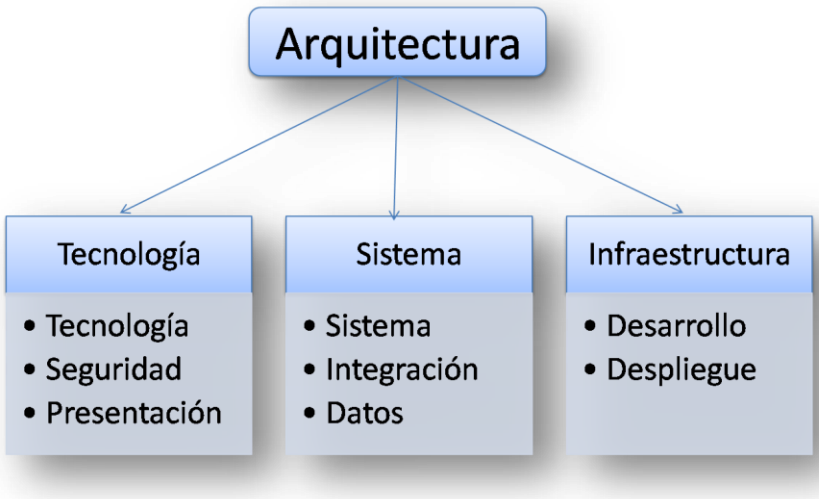


Figura 2.4 Organización de las vistas de la arquitectura del sistema Cedrux

2.4.2 Estilos arquitectónicos aplicados

SIPAC se basa en la arquitectura del sistema de gestión empresarial Cedrux, la cual surgió de la unión de varios estilos arquitectónicos. Tiene un estilo orientado a componentes para poder llevar a cabo el desarrollo horizontal, disminuyendo la complejidad del dominio de los negocios que influyen en el mismo. También se posibilita un alto nivel de reutilización y una independencia a la hora de realizar pruebas a los diferentes componentes. Se decidió mejorar esta arquitectura incorporando elementos propios de las Arquitecturas Orientadas a Servicios (SOA) tales como Registro de Servicios, Servicios, Descripción de servicios, que dan la posibilidad de ajustarse eficazmente a las condiciones cambiantes del negocio empresarial, promoviendo y permitiendo la reutilización, la ejecución de desarrollos paralelos, la interconexión de tecnologías existentes en vez de consumir tiempo y costos en la reinvención y una mayor flexibilidad ya que se pueden añadir nuevos módulos para dotar al sistema de

nuevas funcionalidades. (34) Esta arquitectura se basa en capas las cuales están compuestas fundamentalmente por:

1. **Capa de Presentación:** En esta capa se aprovechan las bondades que brinda el Marco de Trabajo ExtJS para la implementación de interfaces amigables.
2. **Capa de Negocio:** En esta capa se usa el patrón arquitectónico Modell View Controller o Modelo Vista Controlador para gestionar la seguridad a nivel de aplicación.
3. **Capa de Acceso a Datos:** En esta capa se usará Doctrine, como Marco de Trabajo de persistencia para la comunicación con el servidor de datos mediante PDO[18], también consta de un Persistidor de Configuración que es el encargado de comunicarse vía XML con los Ficheros de Configuración del sistema denominado Fast Response.
4. **Capa de Dato:** En esta capa estará ubicado como servidor de base de datos PostgreSQL y un conjunto de Ficheros de Configuración de la arquitectura tecnológica.
5. **Capa de Servicio:** En esta última capa se encuentran todos los subsistemas que prestan y consumen servicios entre sí. (18)

2.5 Patrones de diseño empleados en la solución propuesta

Los patrones de diseño brindan una solución ya probada y documentada a problemas de desarrollo de software que están sujetos a contextos similares, especifican la estructura y el comportamiento de una sociedad de clases. Estos son directrices y principios estructurados que describen un problema común y presentan soluciones simples. (35) Los Patrones de Diseño empleados se concretaron durante la elaboración del diseño de las clases y las relaciones entre ellas, para garantizar un bajo acoplamiento y alta cohesión entre las clases.

2.5.1 Patrones GRASP

GRASP: Es el acrónimo de General Responsability Assignment Software Patterns y se encargan de asignar responsabilidades a los objetos, cuenta con dos responsabilidades fundamentales, conocer (atributos, relaciones con otros objetos) y hacer (tareas que debe cumplir cada objeto). (36)

[18]**PDO**: (en inglés: PHP Data Objects) es una extensión de PHP que define una interfaz de acceso a datos que permite la conexión a diferentes bases de datos utilizando tecnología orientada a objetos. (31)

Entre los principales patrones GRASP utilizados se encuentran:

- **Experto**

Este patrón indica a qué clase pertenece la responsabilidad (clase experta) de realizar una labor en dependencia de la información que posea la clase y la labor que tiene encomendada. Esta se evidencia claramente en la definición de las clases en dependencia de las funciones que ejecuta. Un ejemplo es la clase IntercambioExcelModel que se encarga de exportar los diferentes elementos de la planeación a una hoja de cálculo Excel.

- **Creador**

El patrón Creador orienta quién debe ser el responsable de crear una nueva instancia de alguna clase, hay algunos casos en los que se recomienda que una clase **B** deba crear una instancia de **A** si:

- ✓ B agrega los objetos de A.
- ✓ B contiene a los objetos de A.
- ✓ B registra las instancias de los objetos de A o B utiliza particularmente los objetos de A.
- ✓ B tiene los datos de inicialización que serán transmitidos a A cuando este objeto sea creado. (37)

Este patrón se evidencia en la clase CmpIntercambioController**,** la cual es la responsable de crear instancias de la clase CmpIntercambioModel para así utilizar las funcionalidades de esta.

- **Bajo Acoplamiento**

Este patrón es fundamental pues logra una baja dependencia permitiendo que una clase no dependa de muchas otras, alta reutilización entre las clases y reducir el impacto de los cambios. La siguiente figura ejemplifica cómo se empleó este patrón en el diseño de clases.

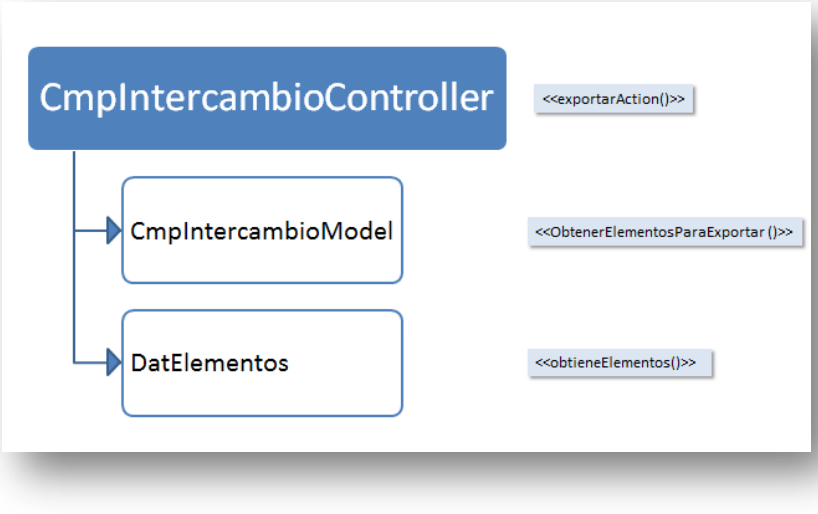


Figura 2.5 Ejemplo de uso de Bajo Acoplamiento

Se puede evidenciar que solo se le atribuye la responsabilidad de crear objetos de DatElementos a la clase CmpIntercambioModel evitando así que el número de recurrencias de la clase CmpIntercambioController a otras clases, sea mayor.

- **Alta Cohesión**

La cohesión es una medida de cuán relacionadas y enfocadas están las responsabilidades de una clase. Una alta cohesión caracteriza a las clases con responsabilidades estrechamente relacionadas que no realicen un trabajo enorme. Este patrón se aplica de manera general en el diseño de SIPAC ya que se agrupan las clases en dependencia de los requerimientos, por ejemplo IntercambioExcelModel solo se encarga de exportar cualquier elemento de la planificación a formato Excel.

- **Controlador**

Un Controlador es un objeto de interfaz no destinado al usuario que se encarga de manejar los eventos en el componente, separando así la lógica de negocios de la capa de presentación. De esta manera el controlador delega a las clases modelo las actividades a realizar manteniendo así una alta cohesión. La clase CmpIntercambioController es un ejemplo de la aplicación de este patrón.

2.5.2 Patrones GoF

GoF: Es el acrónimo de Gang of Four o Banda de Cuatro por sus siglas en español llamados así por los cuatro autores del libro Patrones de Diseño que recoge alrededor de 23 patrones de los más utilizados. Están clasificados según su propósito:

- ✓ Creacional: Resuelven problemas relativos a la creación de objetos.
- ✓ Estructural: Resuelven problemas relativos a la composición de objetos, la relación entre clases, la combinación de clases y la formación de estructuras de mayor complejidad.
- ✓ Comportamiento: Tratan la interacción y cooperación entre clases. (38)

Entre los principales patrones GoF utilizados se encuentran:

- ❖ **Fachada**

Este patrón proporciona una interfaz unificada de alto nivel para un subsistema, que oculta las interfaces de bajo nivel de las clases que lo implementan simplificando así la interacción con el subsistema. Se utiliza para proporcionar un fácil acceso a subsistemas complejos. (39) Este se usa fundamentalmente en los servicios, donde la relación existente entre las clases controladoras y los servicios, permite acceder a métodos que no están implementados en el componente y que se encuentran, tanto en otros componentes pertenecientes a SIPAC, como en otros subsistemas externos. La utilización de la clase PlanServices como fachada para el acceso a determinadas funcionalidades del SIPAC es una muestra del empleo de este patrón.

- ❖ **Cadena de Responsabilidades**

Este patrón consiste en que cada objeto debe saber valorar la petición para una acción y en caso de no poderla controlar sabe exactamente cómo pasarla a otro objeto. Este es principalmente usado en el tratamiento de Excepciones, si se produce un error en una consulta a la base de datos esta es manejada por el Modelo quien crea una excepción de tipo ZendExt_Exception que debe ser enviada al Controlador, quien la envía a la Vista ya traducida y esta muestra en un lenguaje entendible al usuario la notificación del error.

2.6 Diseño de la solución

2.6.1 Diseño de la solución en términos de componentes

El Sistema para la Planificación de Actividades es el encargado de interrelacionar objetivos de trabajo, actividades y recursos en tiempo real; garantizando el seguimiento del desarrollo y cumplimiento de los

objetivos y tareas principales en las entidades. El Modelo de Componentes de dicho sistema se integra por el componente Configuración que es el responsable de la gestión de los grupos de usuarios del sistema, de los permisos respectivos y de la gestión de los nomencladores; el componente Planeación que abarca la relación entre todos los elementos de la planificación y de establecer las relaciones entre ellos y el componente Gestor de Intercambio de Información que permite el intercambio y actualización de la información deseada en el momento deseado; los cuales a través de sus respectivas interfaces de comunicación establecen un estrecho vínculo.

La siguiente figura responde al diagrama de componentes del Sistema para la Planificación de Actividades en el que se resaltan los dos componentes que conforman la solución que se propone. Este diagrama posibilita representar la relación entre los componentes y sus respectivas interfaces.

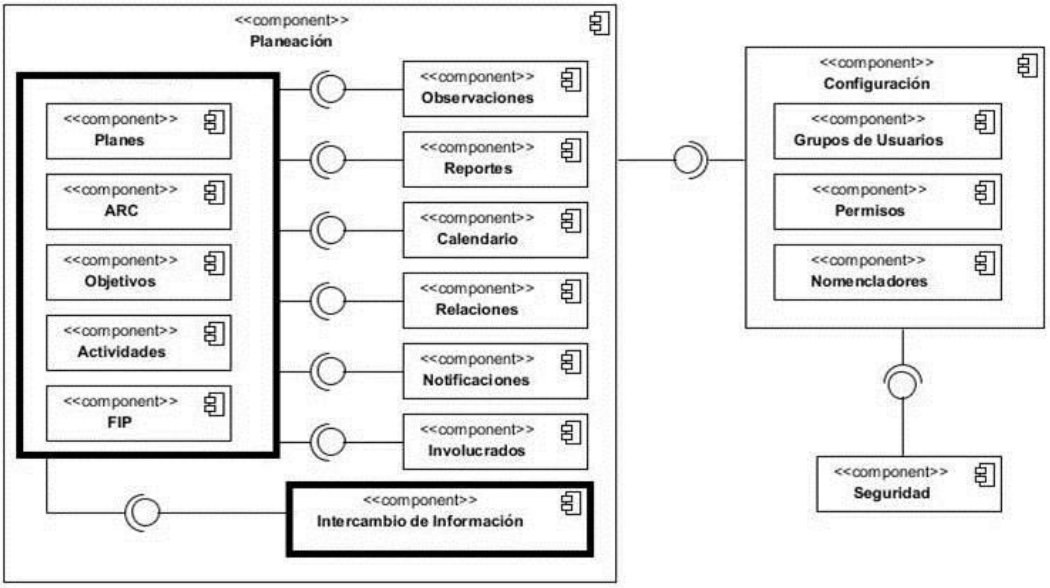


Figura 2.6 Diagrama de componentes del Gestor de Intercambio de Información

2.6.2 Diseño de clases

Para la transición a la fase de implementación es de suprema importancia tener definidas las clases que intervendrán y las relaciones entre ellas. Es por esta razón que a continuación se presenta el diagrama de clases donde se representan las clases del componente y sus interrelaciones.

2.6.2.1 Diagramas de clases de diseño

A continuación se muestra el diagrama de clases de diseño del componente Gestor de Intercambio de Información con las propiedades y operaciones asociadas a cada una de sus clases cuya especificación se encuentra en Anexo 2.

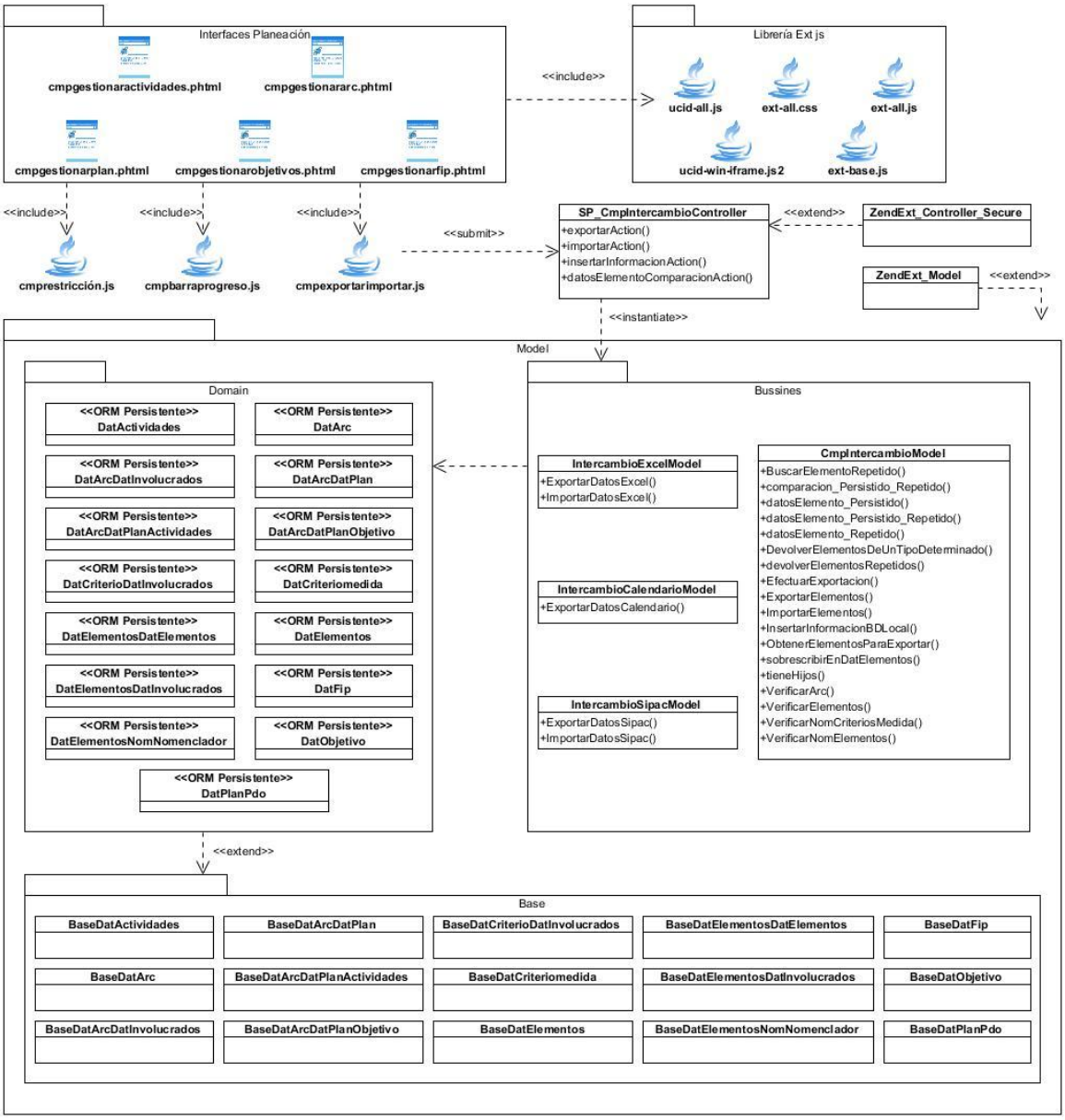


Figura 2.7 Diagrama de Clases del diseño del componente Intercambio de Información

2.6.2.2 Modelos de datos

El modelo de datos de SIPAC fue diseñado de manera que fuera escalable, resumiendo los conceptos y definiendo las relaciones, ajustándose perfectamente a las necesidades de almacenamiento de datos del sistema. Posee tercera forma normal y cuenta con 39 tablas, se tienen en cuenta además algunas formas de persistir información similar a la estructura de grafos y árboles, esto se evidencia en la persistencia de campos resúmenes para agilizar las recuperaciones frecuentes de los datos de los nomencladores con los campos **ordizq** y **orderch**, estos a través de una fórmula determinan si un elemento de la tabla tiene descendencia y cuáles son. Posee una estructura relacional que permite la desagregación de los datos manejados en el proceso. En la Figura 2.8 se muestra el modelo de datos en cuestión y en la Figura 2.9 se detallan las tablas encargadas de almacenar los datos correspondientes al módulo Planeación.

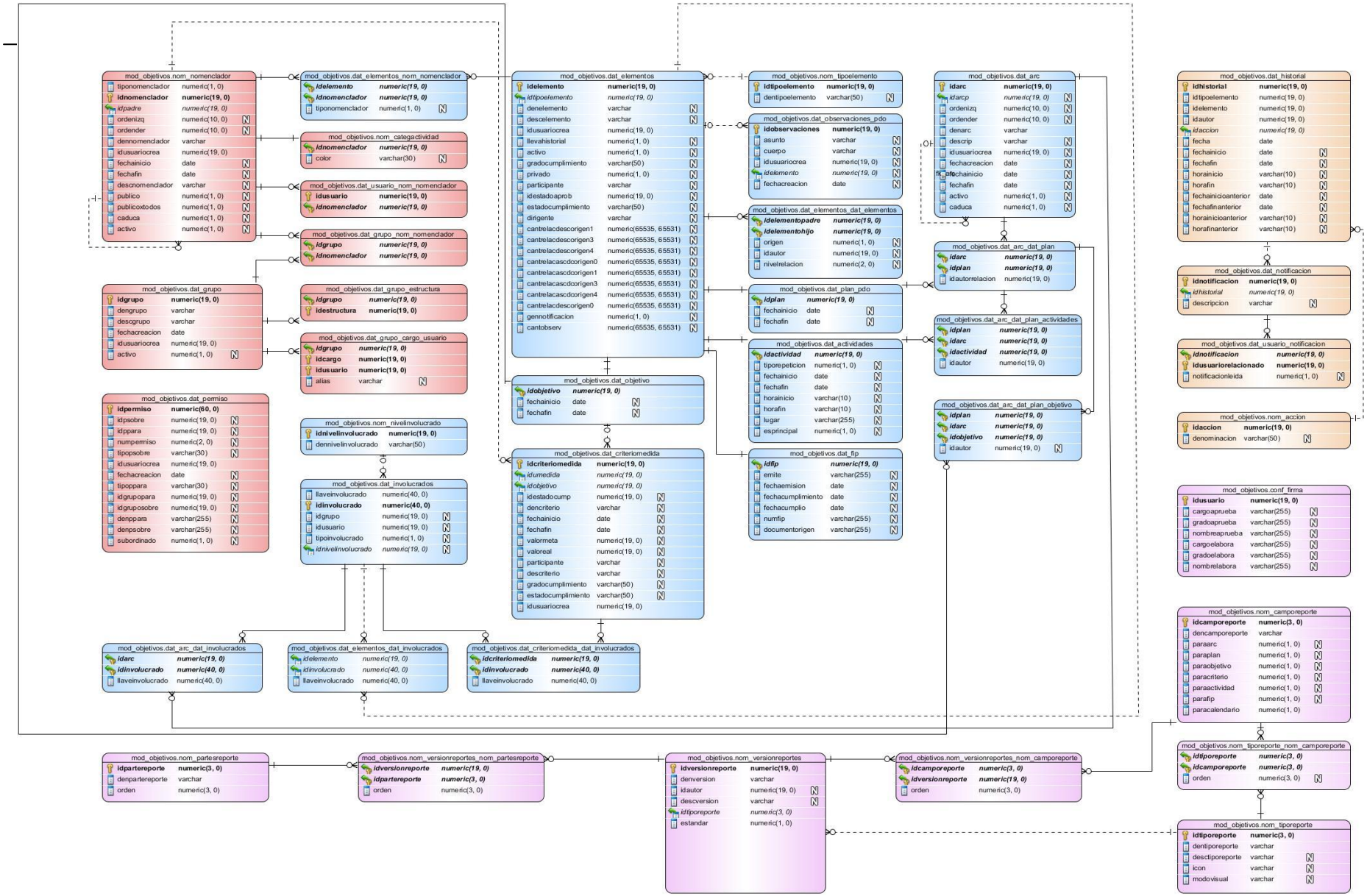


Figura 2.8 Modelos de Datos

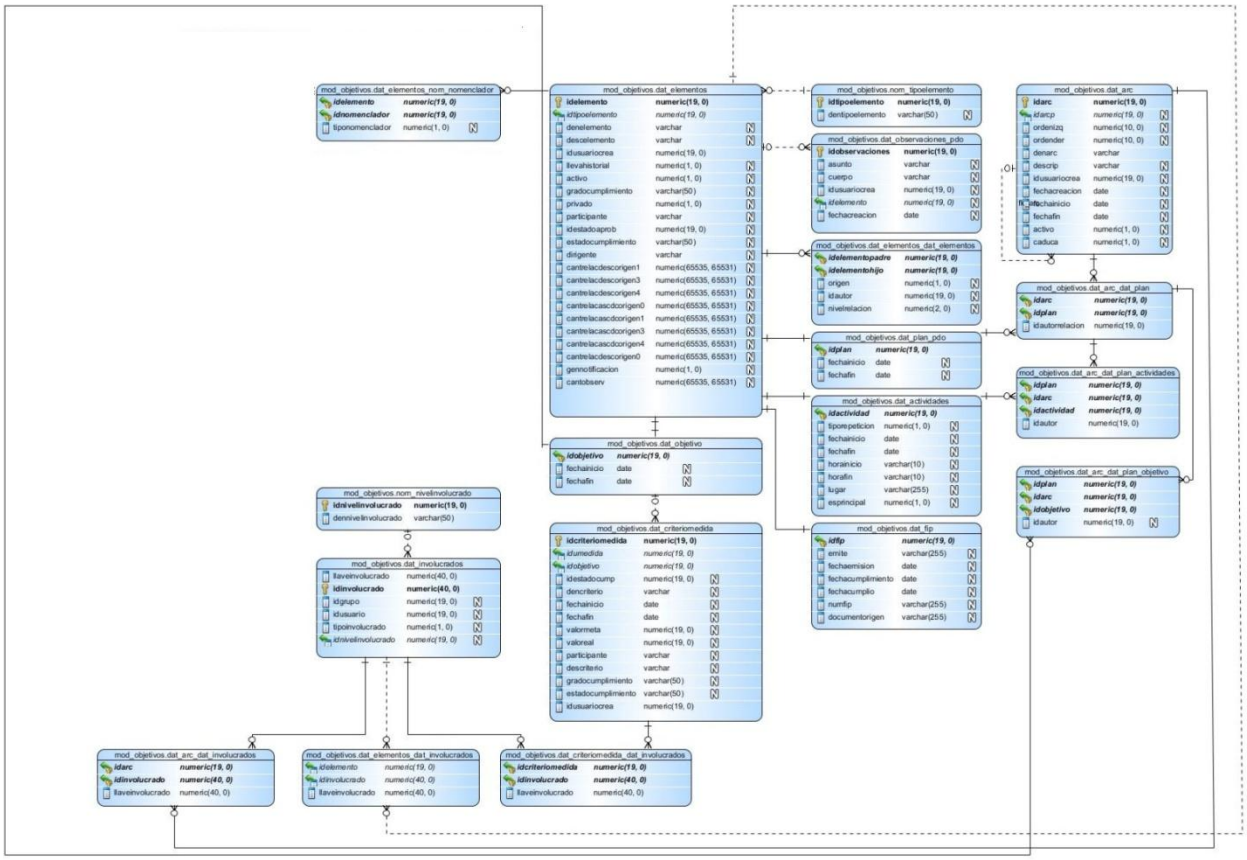


Figura 2.9 Tablas asociadas a la persistencia de datos del módulo Planeación

2.6.2.3 SQLite como solución alterna

En el proceso de actualización de la información existe un momento crítico a tener en cuenta, es cuando es necesario comparar si la información que llega para actualizar ya existe en ese nivel o entidad, pues en caso de que sea así el planificador debe elegir con cuál información se queda, si es con la que recibe o con la que ya existe, en dependencia de sus intereses. Inicialmente, la propuesta para esta situación fue poder efectuar la búsqueda y comparación de la información a nivel de fichero o formato de intercambio, pero esto no es factible debido a que, si el volumen de información contenida en el fichero es muy grande, resulta engorroso efectuar búsquedas sobre un determinado elemento, haciendo recorridos innecesarios y atentando contra el rendimiento de la solución.

Con la premisa de acceder a la información de forma rápida y minimizar el tiempo de búsqueda y de comparación, surgieron nuevas variantes. Primero, era crear una estructura de persistencia exactamente

igual a la que maneja el módulo Planeación, con el objetivo de persistir temporalmente la información recibida para poder comparar y acceder a la información de forma rápida mediante consultas, pero la sola idea de duplicar o crear un espejo dentro del mismo esquema de determinadas tablas o estructuras de persistencia, atenta contra la normalización y la búsqueda del diseño óptimo de la Base de Datos ya que el resultado sería tener dentro del mismo esquema, tablas con objetivos iguales destinadas a persistir datos exactamente iguales, con la única diferencia de que, el espejo creado solo contendría información por un breve tiempo y estaría vacía en el momento en que dicha información se eliminase al concluir el proceso de comparación. Por otra parte, también se manejó la variante de utilizar el esquema destinado a persistir el historial de la información para guardar estos datos pero de forma temporal y luego eliminarlos, opción que no fue viable debido a que chocaba contra las políticas de seguridad definidas para el manejo y administración de la Base de Datos ya que el esquema historial solo permite insertar y leer la información, pero nunca modificarla o eliminarla.

Es aquí cuando surge la solución de utilizar SQLite, una pequeña librería de aproximadamente 500kb, programado en el lenguaje C, de dominio público, totalmente libre y que tiene como función hacer un sistema de bases de datos relacional. Con su arquitectura cliente/servidor, es independiente totalmente, simplemente se realizan llamadas a funciones de las propias librerías de SQLite, lo cual reduce ampliamente la latencia en cuanto al acceso a las bases de datos y realiza operaciones de manera eficiente y es más rápido que MySQL y PostgreSQL. De esta manera se logra contar con un mecanismo que brinda la sencillez y agilidad de un fichero y la robustez de un gestor de base de datos.

2.6.3 Estrategia de integración

Para garantizar la integridad y seguridad del flujo de datos que sigue un flujo común, desde la vista hacia el modelo y viceversa, se definió la comunicación entre las diferentes partes del sistema, así como la integración entre subsistemas y componentes concebida sobre 4 nodos y a partir de cada uno de los elementos arquitectónicos definidos.

A continuación se describe en cada uno de los lugares donde se ubican dichos nodos:

Nodo 1 (Vista – Controlador): Este se encuentra entre la vista y el controlador. Los datos recogidos en un formulario son enviados hacia el Controlador mediante el protocolo HTTP usando el método “post” y

los resultados arrojados son nuevamente enviados por este a la vista en un JSON a través del método “echo”.

Nodo 2 (Controlador – Modelo): Este está entre el controlador y el modelo. El Controlador toma los datos recibidos desde la vista, instancia una determinada clase del modelo y llama a uno de sus métodos, pasándole como parámetros los datos recibidos.

Nodo 3 (Modelo – Doctrine): Vincula el modelo con el marco de trabajo Doctrine. El Modelo utiliza llamadas a métodos de Doctrine que le permitan crear, modificar, eliminar o actualizar los datos almacenados en la Base de Datos.

Nodo 4 (Doctrine – Base de Datos): Se encuentra entre el Doctrine y la Base de Datos. Doctrine ejecuta las consultas a la Base de Datos utilizando programación orientada a objetos.

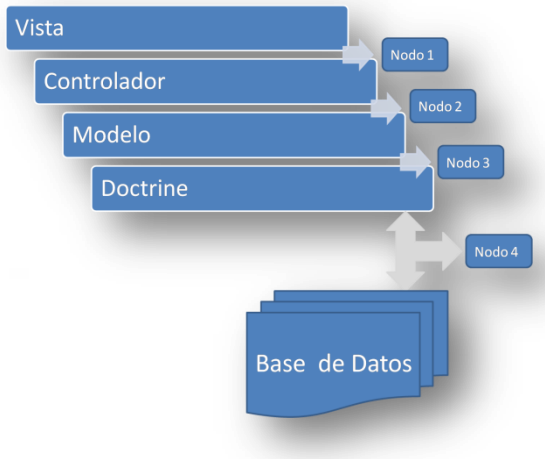


Figura 2.10 Estrategia de integración

2.6.4 Prototipos de interfaz de usuario

A continuación los prototipos de Interfaz de usuarios asociados al componente Gestor de Intercambio de Información.

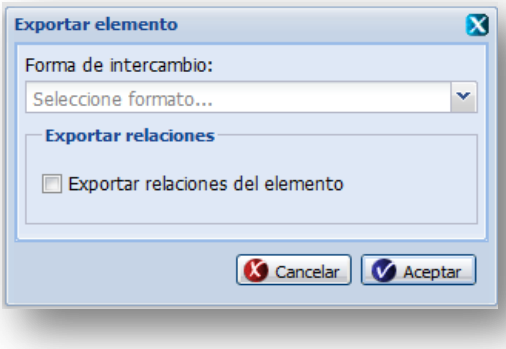


Figura 2.11 Decisión de exportación con o sin relaciones

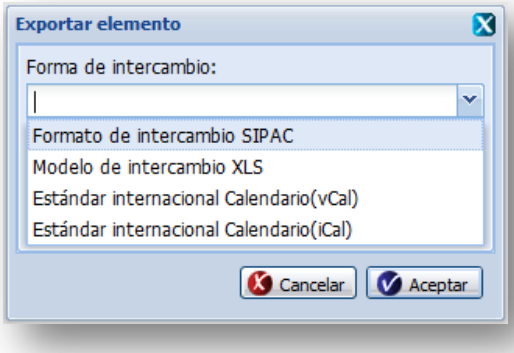


Figura 2.12 Exportar elemento al formato deseado

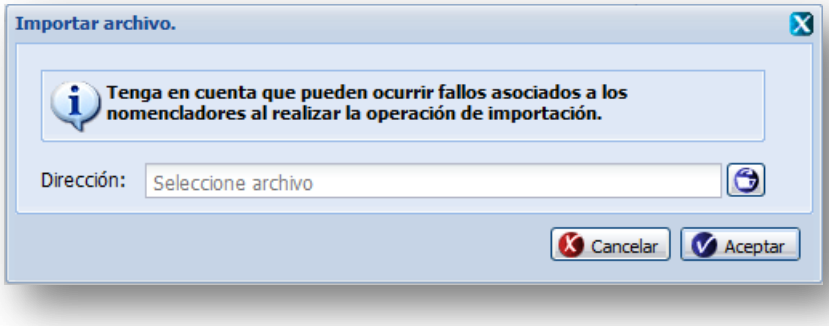


Figura 2. 13 Importar archivo

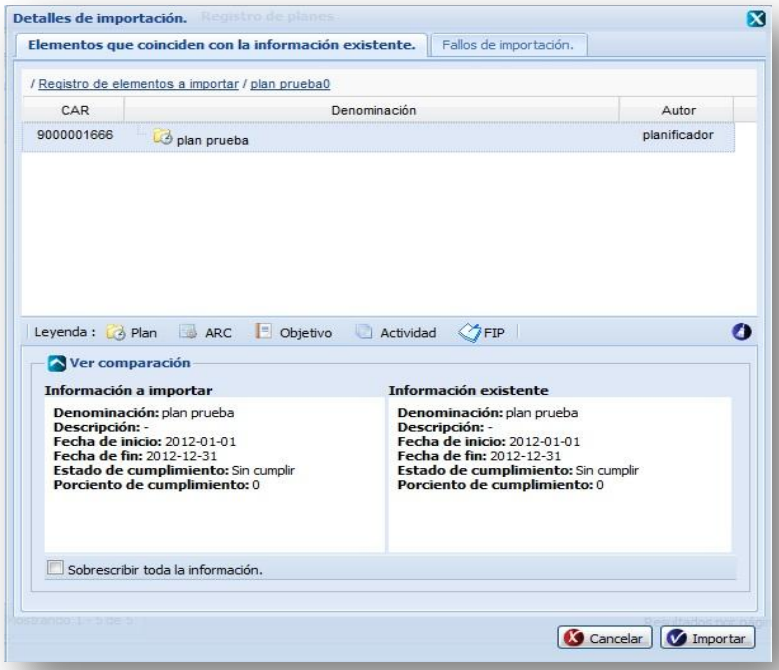


Figura 2.14 Detalles de Importación

2.7 Conclusiones del capítulo

En el presente capítulo se obtuvieron los artefactos relacionados con el análisis y diseño de la solución propuesta a partir de la descripción verbal del proceso de negocio con sus principales actividades. Se identificaron 26 requisitos funcionales a implementar y se definieron los comportamientos a tener en cuenta para la elaboración de las clases. Se establece una visión profunda en términos de componentes y la estrategia de integración de los mismos dando paso a la definición de las funcionalidades y la estructura de la solución, por lo que se puede concluir que es posible el comienzo de la construcción del componente.

Capítulo III: Implementación y validación de la solución propuesta

3.1 Introducción

En el presente capítulo se exponen los estándares a seguir en la implementación del código fuente y en los componentes para el despliegue. También se describe la estructuración del código fuente empaquetado en módulos y componentes, las métricas aplicadas al diseño para asegurar la calidad y por último se realizan una serie de pruebas con el objetivo de validar el cumplimiento de la idea a defender expuesta en el Capítulo 1 del presente trabajo.

3.2 Implementación

La implementación es un proceso de varias fases que empieza una vez obtenido el diseño. Su principal propósito es lograr el desarrollo de la arquitectura y del sistema como un todo. De forma más específica, los propósitos de la Implementación son:

- Planificar las integraciones del sistema necesarias en cada iteración. Siguiendo para ello un enfoque incremental.
- Implementar clases, componentes y subsistemas encontrados durante el diseño.
- Integrar componentes. (40)

A continuación se presenta la estructura física de la solución propuesta en concordancia con el marco de trabajo Sauxe y el diagrama de despliegue asociado a la misma.

3.2.1 Estructura física del sistema

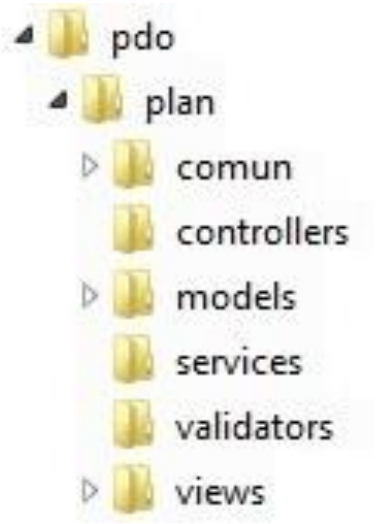
El Departamento de Tecnología de CEIGE toma las decisiones para definir la estructura del marco de trabajo la cual es diseñada para satisfacer los estilos arquitectónicos y niveles de empaquetamiento adoptados para la estructura del sistema; para una mejor organización y claridad durante el desarrollo se van ubicando cada uno de los subsistemas implementados dentro de la aplicación. La estructura física de SIPAC se rige por la estructura del marco de trabajo definida para Cedrux. A continuación se presenta la estructura del componente Intercambio de Información perteneciente SIPAC.

En la carpeta *apps* se encuentra la lógica del negocio. Esta almacena los controladores y el modelo de cada uno de los componentes pertenecientes a los subsistemas. En la siguiente figura se muestra el contenido del subsistema planificación.

comun: Comprende la configuración concreta para el subsistema, dentro de ella está ubicado el fichero modelo_intercambio.xls usado como plantilla estándar para el intercambio con esta herramienta, también se encuentra la carpeta **recursos** y en esta una denominada *xml*. Esta última incluye los ficheros *ioc y* posiciones_modelo_intercambio_xls, en el primero se publican los servicios que brindan cada uno de los componentes de SIPAC mediante un lenguaje de descripción de servicios web (WSDL) y el segundo contiene las posiciones exactas donde se encuentran los atributos de los elementos de la planificación en el fichero con extensión.

temp: Contiene ficheros temporales que se utilizan para el proceso de intercambio y actualización de la información, estos ficheros a su vez poseen las estructuras físicas de la base de datos heredada de SQLite. Además contiene el fichero temporal con extensión xls que es utilizado para intercambiar información mediante el uso de la herramienta Microsoft Excel.

pdo: Contiene al Subsistema de Planificación por Objetivos que incluye el componente Gestor de Intercambio de Información, el cual está estructurado por un conjunto de paquetes que se especifican a continuación:

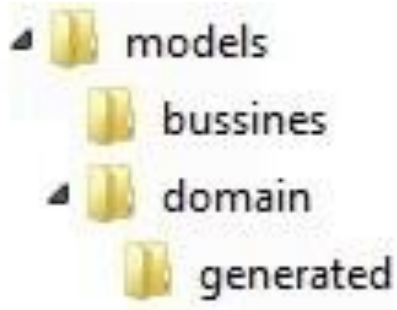


comun: almacena la configuración utilizada por el componente, incluye la carpeta ***recursos*** y dentro de esta una denominada *xml* que contiene los ficheros: *validation y exception.*

El fichero *validation* chequea las precondiciones antes de ejecutar una determinada función en el servidor según el tipo de parámetros, la acción y el usuario que la realice. Mientras que en *exception* se encuentran las descripciones de los mensajes de las excepciones que pueden ser lanzadas por el servidor.

controllers: Este fichero contiene las clases controladoras encargadas de gestionar las funcionalidades de sistema.

models: Este está estructurado de la siguiente manera:



Esta carpeta contiene otras dos, las cuales a su vez agrupan clases y otras subcarpetas. El paquete *bussines* contendrá las clases necesarias para acceder a los datos que persisten en la base de datos. El

paquete *domain* debe contener las clases generadas por el marco de trabajo Doctrine a partir de cada una de las tablas existentes en la base de datos. Cada una de estas clases heredará de clases bases que contienen los atributos y dependencias entre las tablas y se encuentran en el paquete llamado *generated* y son generadas igualmente por Doctrine.

services: Este paquete incluye todas las clases y funcionalidades de los servicios que va a ofrecer el componente.

validators: Agrupa las clases y funcionalidades correspondientes a las validaciones que realiza el servidor antes de ejecutar una determinada función. Estas son llamadas en el xml *validation* perteneciente a los recursos del componente en cuestión.

views: Contiene los paquetes idioma y scripts, que contiene el idioma en que se va a mostrar la aplicación y las páginas clientes.

La carpeta *web* que contiene las vistas de todos los subsistemas y componentes de la aplicación, contiene un fichero *index.php* que almacena la dirección del archivo de configuración y a través de este inicializa la aplicación para que se carguen en la misma un conjunto de componentes necesarios para su funcionamiento. Su código es el mismo para todos los componentes.

pdo: Contiene todas las vistas del Subsistema de Planificación por Objetivos, incluye el componente Gestor de Intercambio de Información, que igualmente incorpora el fichero *index.php*. También contiene el paquete *views* que se especifica a continuación.

views: Esta contiene los *css* y *js* necesarios para visualizar todo el contenido de la presentación del componente Gestor de Intercambio de Información.

css: Contiene las clases necesarias para darle la estructura gráfica al componente, separando así el estilo del contenido.

js: Comprende los ficheros JavaScript necesarios para que el usuario interactúe con el sistema y obtenga los resultados esperados. Está compuesto por paquetes con los nombres de las funcionalidades que contiene.

3.2.2 Estándares de código

Un estándar en el código de programación es muy importante principalmente para que el desarrollador entienda y puedan mantener la aplicación. Los estándares de programación mejoran la legibilidad del código, al mismo tiempo que facilitan su compresión y se asegura un código de alta calidad y fácil de mantener. (41) Se definen tres partes fundamentales dentro de un estándar de programación:

- Convención de nomenclatura: Define cómo nombrar variables, funciones y clases.
- Convenciones de legibilidad de código: Es la forma de organizar el código y lograr que independientemente de quien desarrolle se entienda como un todo.
- Convenciones de documentación: Define cómo establecer comentarios, archivos de ayuda, entre otros.

❖**Nomenclatura de las clases**

Los nombres de las clases deben comenzar con mayúscula y el resto en minúscula, en caso de que sea un nombre compuesto se empleará notación *PascalCasing*, la cual define que los identificadores y nombres de variables, métodos y clases que están compuestos por múltiples palabras juntas, inicia cada palabra con letra mayúscula y sin usar ningún artículo posibilitando que con solo leer el nombre de la clase ya se reconozca la función de la misma.

❖**Nomenclatura según el tipo de clases**

Clases controladoras: Las clases controladoras después del nombre llevan la palabra “Controller”. Ejemplo: CmpIntercambioController.

Clases de los modelos:

- Bussines (Negocio): Las clases que se encuentran dentro del paquete Bussines después del nombre llevan la palabra “Model”. Ejemplo: CmpIntercambioModel.
- Domain (Dominio): Las clases que se encuentran dentro de Domain reciben el nombre de las tablas de la base de datos. Ejemplo: “DatPlanPdo”.
- Generated (Dominio base): El nombre de las clases que se encuentran dentro de Generated comienza con la palabra: “Base” y seguido el nombre de la tabla en la base de datos. Ejemplo: BaseDatElementos.

❖**Nomenclatura de las funcionalidades y los atributos**

El nombre de las funciones y atributos se comienza con letra inicial minúscula, en caso de que sea un nombre compuesto se empleará la notación *CamelCasing* que es similar a la *PascalCasing* pero con la diferencia de la primera letra.

A las funcionalidades que se encuentran en las clases controladoras se les agrega al final del nombre la palabra “Action”. Ejemplo: exportarAction().

A las que se encuentran en las clases de lo modelos se nombran de manera que con solo leerlas se identifique su objetivo. Ejemplo: exportarDatosExcel().

❖**Nomenclatura de los comentarios**

Los comentarios deben ser claros y concisos de manera que se entienda el objetivo de lo que se está programando. Para las funciones más complejas se debe comentar a un nivel de detalle más alto para alcanzar un óptimo entendimiento.

3.2.3 Diagrama de despliegue

A continuación de describe cómo queda distribuido físicamente el sistema entre los diferentes nodos. Seguidamente se describen los diagramas de despliegue en sus dos variantes principales, el primer escenario para PC cliente con disco duro (Figura 3.1) y el segundo escenario para PC sin disco duro también conocidas como clientes ligeros[19] (Figura 3.2).

Figura 3.1 Diagrama de despliegue de escenario para PC cliente con disco

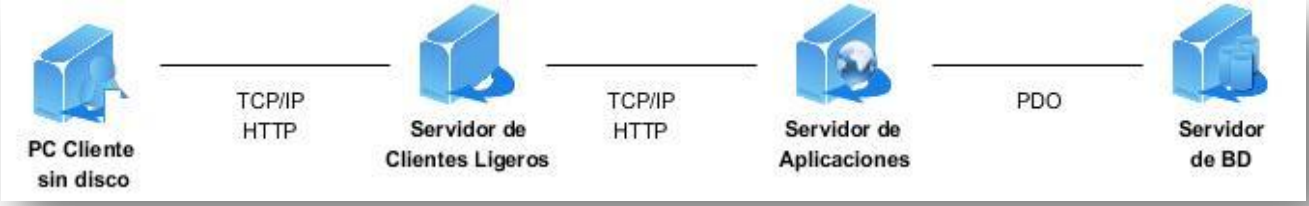


Figura 3.2 Diagrama de despliegue para clientes ligeros

[19]**Cliente ligero o cliente liviano:** es una computadora cliente en una arquitectura de red cliente-servidor que depende primariamente del servidor central para las tareas de procesamiento, y principalmente se enfoca en transportar la entrada y la salida entre el usuario y el servidor remoto.

3.3 Validación de la solución propuesta

El conjunto de cualidades que caracterizan y que determinan la utilidad y existencia de un software se avala por la calidad del mismo; este parámetro se ha convertido en un elemento muy importante para las grandes organizaciones debido a que influye directamente en la competitividad entre las empresas. Durante el proceso de desarrollo de software los errores son múltiples y se pueden manifestar desde la misma especificación de requisitos donde se define lo que el sistema debe hacer. Para detectar y corregir el máximo de estos errores se expone a continuación la validación de la solución propuesta, se presentan las métricas que permiten juzgar la calidad del diseño y se describen las pruebas de aplicación realizadas.

3.3.1 Validación del diseño propuesto

Las métricas son un medidor bastante eficiente que se utiliza para validar la calidad del diseño implementado; a pesar de necesitarse práctica para el correcto empleo de las mismas, deben realizarse a cabalidad, pues no se concibe el diseño de las aplicaciones informáticas sin la realización de mediciones como una alternativa viable. (42) A continuación se presentarán las métricas Tamaño Operacional de Clase (TOC) y Relaciones entre Clases (RC), necesarias para evaluar la calidad del diseño propuesto a nivel de componentes. Ninguna de las dos métricas fue aplicada durante el diseño procedimental, sino que fueron aplicadas cuando estuvo disponible el código fuente, ya que la complejidad del negocio exigía la implementación de numerosas funciones secundarias necesarias para la implementación de cada procedimiento descrito en el diseño de clases. También como métrica asociada a la herencia se utilizó Profundidad de Herencia (PH) que indica el número de definiciones de clase que se extienden a la raíz de la jerarquía de clases, cuanto más profunda es la jerarquía, más difícil entender dónde se definen determinados métodos. (43) Esta métrica está relacionada con la reusabilidad desde el punto de vista de que las clases que son más profundas en el árbol de herencia son más complejas y presentan predisposición al ser reutilizadas. En este caso se comporta de manera diferente ya que las clases definidas en el diseño presentan Nivel 1 del árbol de profundidad para un 100 % de las clases en este nivel lo que favorece a la reutilización. Ver Anexo 3.

Las métricas TOC y RC incluyen medidas de los siguientes atributos de calidad:

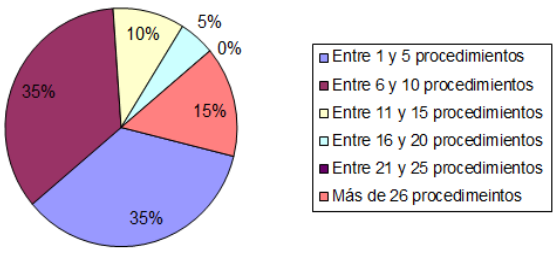
- **Responsabilidad:** Responsabilidad asignada a una clase en un marco conceptual correspondiente al modelado de la solución propuesta.

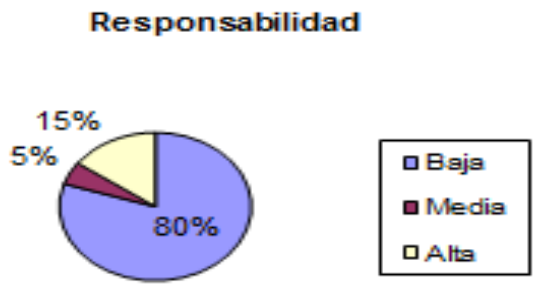
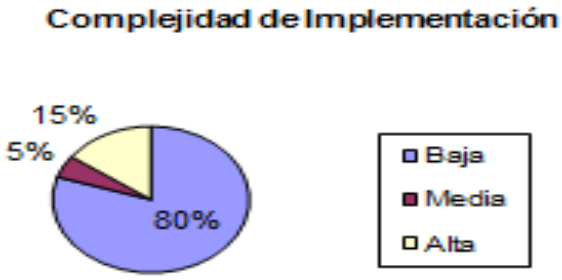
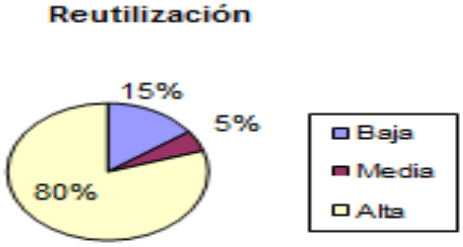
- **Complejidad del mantenimiento:** Nivel de esfuerzo necesario para sustentar, mejorar o corregir el diseño de software propuesto. Puede influir significativamente en los costes y la planificación del proyecto.
- **Complejidad de implementación:** Grado de dificultad que representa implementar un diseño de clases determinado.
- **Reutilización:** Significa cuán reutilizada es una clase o estructura de clase dentro de un diseño de software.
- **Acoplamiento:** Dependencia o interconexión de una clase o estructura de clase respecto a otras.
- **Cantidad de pruebas:** Número o grado de esfuerzo necesario para realizar las pruebas de calidad al producto (componente) diseñado. (1)

❖**Métrica TOC: Tamaño operacional de clase.**

Con la evaluación de la métrica TOC se obtuvo como resultado lo siguiente:

Tabla 5: Métrica Tamaño Operacional de Clases(TOC)

Tamaño Operacional de Clases (TOC).
Descripción: Está dado por la cantidad de funciones asignados a una clase. 

Atributo que evalúa: **Responsabilidad:** Un aumento del TOC provoca un aumento de la responsabilidad asignada a la clase.	
Atributo que evalúa: **Complejidad de implementación**: El aumento del TOC provoca un aumento de la Complejidad de implementación.	
Atributo que evalúa: **Reutilización:** Un aumento del TOC provoca una disminución en el grado de reutilización de la clase.	

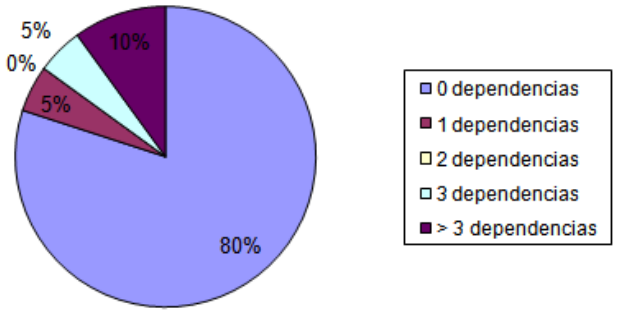
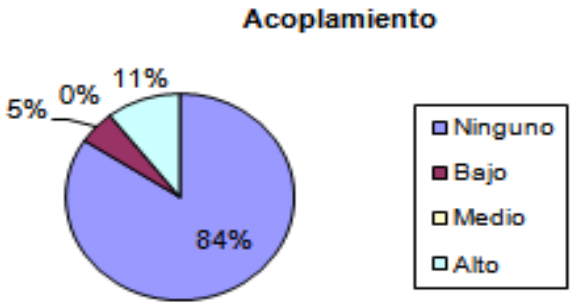
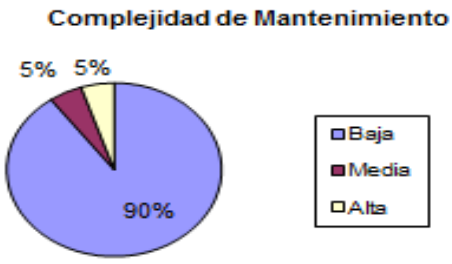
Ver instrumentos y tabla de resultados en:

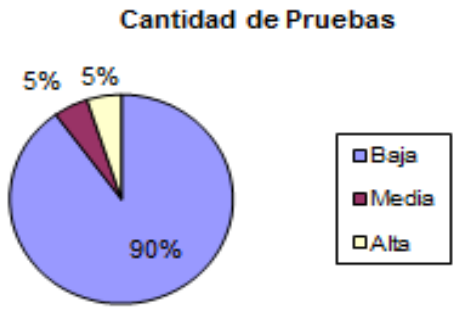
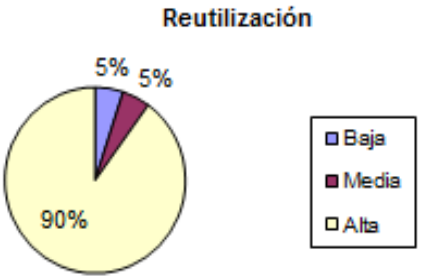
Anexo 4: Instrumento de medición de la métrica Tamaño Operacional de Clase (TOC).

❖**Métrica RC: Relaciones entre Clases.**

Con la evaluación de la métrica RC se obtuvo como resultado lo siguiente:

Tabla 6: Métrica Relaciones entre Clases (RC)

<table>
<tr><th colspan="2">Relaciones entre Clases (RC).</th></tr>
<tr><td colspan="2">Descripción: Está dado por el número de relaciones de uso de una clase con otras.
 
</td></tr>
<tr><td>Atributo que evalúa:
Acoplamiento: El aumento de las RC provoca un aumento del acoplamiento de la clase.</td><td> 
</td></tr>
<tr><td>Atributo que evalúa:
Complejidad del mantenimiento: El aumento de las RC provoca un aumento de la complejidad del mantenimiento de la clase.</td><td> 
</td></tr>
</table>

Atributo que evalúa: **Cantidad de Pruebas:** El aumento de las RC provoca un aumento de la cantidad de pruebas de unidad necesarias para probar una clase.	
Atributo que evalúa: **Reutilización:** El aumento de las RC provoca una disminución en el grado de reutilización de la clase.	

Ver instrumentos y tabla de resultados en:

Anexo 5: Instrumento de medición de la métrica Relaciones entre clases (RC).

Análisis de los resultados obtenidos en la evaluación de las métricas TOC y RC

Después de analizar los resultados alcanzados en la aplicación de las métricas aplicadas, se llega a la conclusión de que la métrica TOC evidenció que los niveles de responsabilidad y la complejidad de implementación son bajos con un valor de un 80%. Por su parte la métrica RC demostró que el 11% de las clases presenta un alto nivel de acoplamiento, por consiguiente menor es la complejidad de mantenimiento y la cantidad de comprobación que se produce cuando se efectúen modificaciones. Además ambas métricas indican un alto nivel de reutilización con valores que oscilan entre el 80% y el 90%.

3.3.2 Pruebas de Aplicación

Para determinar la calidad es muy importante el proceso de pruebas. Este es parte del ciclo de desarrollo de software, como uno de los tantos procesos que intervienen en la creación de un producto. En este proceso se ejecutan pruebas dirigidas a componentes del software o al sistema de software en su totalidad, con el objetivo de medir el grado en que el software cumple con los requerimientos y eliminar los posibles errores encontrados. Entre las bondades de una prueba están:

- Debe tener una alta probabilidad de encontrar un error.
- No debe ser redundante.
- Debe ser la mejor de todas las posibles.
- No debe ser ni demasiado sencilla ni demasiado compleja.

❖Tipos de Pruebas de Software

Las pruebas en conjunto tienen como objetivo general verificar y validar un software, independientemente de las características y el entorno donde se desarrollen, además de los recursos y los factores vinculados al proceso de desarrollo. Según en el nivel de trabajo en que se encuentre un software se usa un tipo de prueba, entre algunos de estos se definen:

Pruebas de Unidad: Se focaliza en ejecutar cada módulo, lo que proporciona un mejor modo de manejar la integración de las unidades en componentes mayores. Busca asegurar que el código funciona de acuerdo con las especificaciones.

Pruebas de Integración: Se especializa en identificar errores introducidos por la combinación de programas probados unitariamente y verifica que las interfaces entre las entidades externas y las aplicaciones funcionan correctamente.

Pruebas del Sistema: Se encarga de asegurar la apropiada navegación dentro del sistema, ingreso de datos, procesamiento y recuperación.

Pruebas de Aceptación: Ayuda a la determinación por parte del cliente de la aceptación o rechazo del sistema desarrollado, es ejecutada antes de que la aplicación sea instalada dentro de un ambiente de producción. (44)

❖Métodos de prueba

Los métodos de prueba poseen un enfoque sistemático y se utilizan para ayudar a definir conjuntos de casos de prueba[20] para identificar diferentes tipos de errores basados en un cierto criterio. Estos casos de prueba quedarán determinados por los valores a asignar a las entradas en su ejecución y son independientes del nivel donde se enmarque la prueba. Posee dos enfoques alternativos:

Pruebas de Caja Blanca: Sus criterios de selección de casos de prueba son basados en el contenido de los módulos y buscará cierta cobertura entre caminos independientes, valores de las condiciones, bucles dentro y fuera de sus límites operacionales y estructuras internas de datos.

Pruebas de Caja Negra: Estas pruebas se realizan para comprobar que las funcionalidades son operativas sin importar el código. Se enfocan en encontrar fallas como las que se exponen a continuación. (45)

- ❖ Funciones incorrectas o ausentes.
- ❖ Errores de interfaz.
- ❖ Errores en estructuras de datos o en accesos a las bases de datos externas.
- ❖ Errores de rendimiento.
- ❖ Errores de inicialización y terminación.

Para llevar a cabo las pruebas de caja negra existen varias técnicas, tales como:

- ✓ **Partición de Equivalencia:** Divide el campo de entrada de un programa en clases de datos de los que se pueden derivar casos de prueba y descubre de forma inmediata una clase de errores.
- ✓ **Análisis de Valores Límites:** Lleva a una elección de casos de prueba que ejerciten los valores límite y en los extremos de la clase ya que los errores tienden a darse más en los límites del campo de entrada que en el centro.
- ✓ **Grafos de Causa-Efecto:** Selecciona un gran conjunto de casos de prueba de manera sistemática con una ayuda gráfica. Tiene un efecto secundario beneficioso en precisar estados incompletos y ambigüedades en la especificación. (46)

Pruebas realizadas al componente:

[20] **Casos de prueba:** especifican una forma de probar el componente, incluye: la entrada, las condiciones bajo las cuales ha de probarse y los resultados esperados. (51)

Para llevar a cabo el proceso de pruebas y verificar el funcionamiento del componente Gestor de Intercambio de Información se empleó el método de caja negra, específicamente la técnica de partición de equivalencia. Con esta técnica se reduce el posible conjunto de casos de prueba en uno más pequeño, un conjunto manejable que evalúa bien el software. Con el diseño de casos de prueba para partición equivalente se identifican las clases de equivalencia tomando cada condición de entrada y repartiéndola en dos o más grupos: las clases de equivalencia válidas que representan entradas válidas al programa y las clases de equivalencia inválidas que representan el resto de los estados posibles de la condición, que son los valores erróneos de la entrada.

Casos de pruebas de caja negra generados al aplicar dicha técnica:

Requisito a probar: Exportar Plan

Tabla 7: Caso de prueba “Exportar Plan”

Nombre del requisito	Descripción general	Escenarios de pruebas	Flujo del escenario
1: Exportar Plan.	Se exporta(n) el/los planes hacia un fichero.	EP 1.1: Exportar el/los planes con las relaciones a otros elementos.	– Se selecciona(n) el/los planes que se desea(n) exportar. – Se presiona el botón **Exportar plan(es)** o se presiona clic derecho y la opción **Exportar plan.** – Se selecciona la extensión a que se desea exportar. – Se marca la opción “Exportar relaciones del elemento”. – Se presiona el botón **Aceptar.** – Se selecciona la opción Guardar archivo. – Se presiona el botón **Aceptar**.
		EP 1.2: Exportar el/los planes sin las relaciones a otros elementos.	– Se selecciona(n) el/los planes que se desea(n) exportar. – Se presiona el botón **Exportar plan(es)** o se presiona clic derecho y la opción **Exportar plan.** – Se selecciona la extensión a que se desea exportar. – No se marca la opción “Exportar relaciones del elemento”. – Se presiona el botón **Aceptar.** – Se selecciona la opción Guardar archivo. – Se presiona el botón **Aceptar**.

	EP 1.3: Exportar el/los planes sin seleccionar el formato.	– Se selecciona(n) el/los planes que se desea(n) exportar. – Se presiona el botón **Exportar plan(es)** o se presiona clic derecho y la opción **Exportar plan.** – No se selecciona la extensión a que se desea exportar. – Se marca o no la opción “Exportar relaciones del elemento”. – Se presiona el botón **Aceptar.** – El sistema muestra el mensaje de información. – Se presiona el botón **Aceptar** del mensaje.
	EP 1.4: Cancelar	– Se selecciona(n) el/los planes que se desea(n) exportar. – Se presiona el botón **Exportar plan(es)** o se presiona clic derecho y la opción **Exportar plan**. – Se presiona el botón **Cancelar**.

Tabla 8: Descripción de variable

No	Nombre de campo	Tipo	Válido	Inválido
NA	NA	NA	NA	NA

Tabla 9: Juegos de datos a probar

Id del escenario	Escenario	Respuesta del sistema	Resultado de la prueba
EP 1.1	Exportar el/los planes con las relaciones a otros elementos.	Se guarda la información del plan o los planes seleccionados y sus relaciones en un archivo con extensión *.xls, *.sipac, *.ics o *.vcs.	
EP 1.2	Exportar el/los planes sin las relaciones a otros elementos.	Se guarda la información del plan o los planes seleccionados sin sus relaciones en un archivo con extensión *.xls, *.sipac, *.ics o *.vcs.	
EP 1.3	Exportar el/los planes sin seleccionar el formato.	El sistema muestra el mensaje: “Por favor verifique que hay campo(s) con valor(es) incorrecto(s)”.	
EP 1.4	Cancelar.	Se cancela la operación.	

Resultados de las pruebas realizadas

Luego de aplicados los métodos de prueba al componente implementado, es válido señalar que los resultados obtenidos hasta el momento han sido satisfactorios desde el punto de vista interno y funcional del mismo. No fueron detectadas no conformidades logrando un correcto comportamiento del componente ante disímiles escenarios (entradas válidas y no válidas).

El equipo de Calisoft ha llevado a cabo las pruebas de calidad de una primera versión de SIPAC, dentro de ella el componente Gestor de Intercambio de Información y se han validado los requerimientos identificados en la solución propuesta, certificándose en el Acta de Liberación y proyectándose el avance en la ejecución del Intercambio y Actualización de la información. Ver Anexo 6. Actualmente se están realizando pruebas de aceptación al componente desarrollado en una primera versión junto a SIPAC en distintos ministerios del país, con el propósito de confirmar que el sistema desarrolla puntualmente las necesidades de la entidad y en opinión del usuario satisface la ejecución del proceso.

3.3.3 Validación de la idea a defender

Todas las pruebas realizadas hasta el momento y las métricas de calidad aplicadas al diseño del componente han arrojado resultados satisfactorios y se puede afirmar que el sistema cumple con los estándares de calidad de diseño requeridos y que los requisitos funcionales han sido correctamente implementados. ¿Pero se puede afirmar realmente que el resultado final valida la idea a defender planteada en la introducción del presente trabajo? Para responder esta pregunta se decidió realizar el proceso de intercambio y actualización de la información en un ambiente real, usando los mecanismos antiguos y la solución propuesta en este trabajo de diploma. Se usaron algunas variables para comparar los resultados obtenidos, las cuales se detallan a continuación:

Tiempo: Se refiere al tiempo total necesitado para efectuar el intercambio y actualización de información entre los niveles.

Volumen de información: Se refiere a la cantidad de información extraída de un origen e insertada en un destino.

Capacidad de ejecución: Se refiere a la capacidad que puede tener cada usuario de efectuar el proceso de intercambio y actualización de la información en las herramientas comparadas.

Tabla 10: Unidad de medida de las variables

Variable	Tiempo	Volumen de información	Capacidad de ejecución
Unidad de Medida	Minutos(min)	Cantidad de elementos(Planes, Áreas de Resultados Claves(ARC), Objetivos y Actividades)	Cantidad de pasos o procesos que no dependen del usuario.

El escenario seleccionado fue el proceso de intercambio de la información durante la planificación del año venidero en el Ministerio de las Fuerzas Armadas Revolucionarias (MINFAR). Se seleccionaron tres usuarios que representan dos niveles de planificación, un jefe que representaba al nivel superior y dos subordinados representando al nivel inferior. Esta planificación se ordena efectuar el día 1 de Mayo de cada año y tiene como fecha tope de entrega el día 15 del mismo mes para enviar dicha información.

Tiempo:

Esta variable se midió contabilizando el tiempo empleado desde que el planificador decide comenzar a planificar y necesita localizar al informático encargado de efectuar la réplica offline de la base de datos, este proceso puede demorar en dependencia de donde se localice al informático alrededor de 12 minutos, cuando se le asigna la tarea al informático este pide un autorizo necesario para efectuar la operación que demora cerca de 15 minutos, cuando recibe el permiso se dirige hacia el local del servidor demorándose 2 minutos aproximadamente, donde efectúa la réplica para obtener los datos necesarios para el planificador, en caso de que no esté diseñada la réplica debe ser configurada para que devuelva los datos requeridos, la completa ejecución tarda de 4 a 10 minutos en dependencia de si tiene que hacer configuraciones o no. Cuando la réplica está completa se lleva en una memoria flash[21] al departamento de la Oficina secreta donde se escanea para verificar que el archivo no contenga virus, este es un proceso muy común y necesario en las entidades militares pero que no se tomó en cuenta para la medición de la variable Tiempo ya que en el resto de las entidades este departamento no existe, una vez escaneada y revisada la réplica, le es devuelta al informático y este regresa a su local de donde le envía la réplica al planificador vía STI[22], esta demora alrededor de 1 minuto en llegar para comenzar la planificación. Una vez concluida es enviada al planificador del nivel superior mediante la misma vía demorando el mismo tiempo que le tomó al planificador del nivel inferior recibirla y en este nivel se carga en el sistema para verificarla. Todo este proceso demora 41 minutos aproximadamente sin contar el

[21] **Memoria flash**: Dispositivo para almacenar información
[22] **STI**: Sistema de Transmisión de Información. Herramienta para enviar información empleado por las unidades militares en Cuba.

tiempo que se demora la réplica en el Departamento de la oficina secreta, por otro lado usando el Gestor de Intercambio de Información este tiempo se reduce considerablemente ya que no se necesita contar con réplicas offline ni con terceros usuarios que intervengan en el proceso, a través de SIPAC se obtienen los datos necesarios para realizar la planificación la cual una vez es concluida se selecciona exportar a uno de los formatos brindados, proceso que demora menos de 1 minuto, se envía vía STI al planificador del nivel superior el cual la importa en SIPAC y comienza a trabajar sobre ella. Se puede apreciar la considerable disminución de tiempo que representó el uso de la solución implementada en el presente trabajo para el proceso de intercambio de información en el MINFAR.

Tabla 11: Desglose de la variable Tiempo

Proceso	Réplica Offline	Gestor de Intercambio de Información
Localizar al informático	12 min	0 min
Solicitar permiso necesario para efectuar la réplica offline	15 min	0 min
Dirigirse hacia el local del servidor	2 min	0 min
Configurar la réplica para obtener los datos requeridos	6 min	0 min
Efectuar réplica offline	4 min	0 min
Enviar réplica offline al planificador vía STI	1 min	0 min
Enviar la planificación al planificador del nivel superior vía STI	1 min	2 min
Tiempo Total aproximado	**41 min**	**2 min**

Volumen de información

Para efectuar las réplicas offline se necesita especificar la cantidad y el tipo de información deseada pero igualmente no se garantiza que se obtendrá esa información solamente ya que se replican otros datos de configuración que generalmente no son necesarios. En la réplica efectuada en la prueba real se obtuvo un volumen medio de información innecesaria ya que se le solicitó el informático que extrajera los datos referentes a un Plan, cuatro Aéreas de Resultados Claves, diez Objetivos y quince Actividades; cuando se realizó la réplica esta devolvió las tablas de los datos solicitados junto a todas las relaciones que tenían los determinados elementos en ambas direcciones cuando solo se necesitaban las relaciones descendentes, además se obtuvieron los nomencladores y las unidades de medida. Por otro lado el gestor

de intercambio arrojó exactamente los datos seleccionados por el propio planificador con las relaciones descendentes solamente.

Tabla 12: Desglose de la variable Volumen de Información

Elementos solicitados	**Elementos devueltos por la Réplica Offline**	**Elementos devueltos por el Gestor de Intercambio de Información**
1 Plan	1 Tabla con el plan solicitado más todos los nomencladores definidos para todos los elementos del plan	El plan solicitado
4 ARC	1 Tabla con los 4 ARC solicitados más 5 actividades y 2 Planes con las que estaba relacionado	Los 4 ARC solicitados más 5 actividades con las que tenía relación
10 Objetivos	1 Tabla con los 10 objetivos necesarios más 12 actividades y 3 ARC con las que estaba relacionado	Los 10 objetivos necesarios más 12 actividades con las que estaban relacionados
15 Actividades	1 Tabla con las 15 actividades solicitadas más 3 ARC y 4 Objetivos con los que estaba relacionado	Las 15 actividades solicitadas

Capacidad de ejecución

Para esta variable se contabilizaron la cantidad de actividades que necesitaban de terceros usuarios en el proceso de intercambio de información en el escenario seleccionado mediante la réplica offline y el gestor de intercambio de información. Se demostró que el planificador necesitaba de un informático con permisos administrativos para efectuar la réplica, este a su vez requería de una autorización de otra persona para poder crear la réplica, una vez comenzado el proceso para efectuar la réplica se hicieron determinadas configuraciones para obtener los datos esperados obteniéndose un 15 % de capacidad de ejecución. Por otro lado el gestor de intercambio pone al planificador como principal protagonista de sus acciones ya que evita muchos de los procesos y actividades secundarias necesarias para efectuar dicho intercambio contando con un 100% de capacidad de ejecución. La siguiente tabla demuestra lo antes expuesto.

Tabla 13: Desglose de la variable Capacidad de ejecución

Actividades o procesos	**Réplicas Offline**	**Gestor de Intercambio de Información**

Localizar al informático	Depende de terceras personas	No se realiza la actividad
Solicitar permiso necesario para efectuar la réplica offline	Depende de terceras personas	No se realiza la actividad
Dirigirse hacia el local del servidor	Depende de terceras personas	No se realiza la actividad
Configurar la réplica para obtener los datos requeridos	Depende de terceras personas	No se realiza la actividad
Efectuar réplica offline	Depende de terceras personas	No se realiza la actividad
Enviar réplica offline al planificador vía STI	Depende de terceras personas	No se realiza la actividad
Enviar la planificación al planificador del nivel superior vía STI	No depende de terceras personas	No depende de terceras personas
Capacidad de ejecución	**15%**	**100%**

Luego de lo antes expuesto se puede sustentar que la solución propuesta valida la idea a defender expresada en la introducción del presente trabajo.

3.4 Conclusiones del capítulo

En este capítulo se presentó la estructura física de componente Gestor de Intercambio de Información JIT, se precisaron los estándares de código para su implementación y los componentes necesarios para su despliegue. Se aplicaron métricas para la evaluación del diseño. Se efectuaron pruebas al componente para verificar su correcto funcionamiento, aplicándose pruebas de caja negra, realizándose cada uno de los casos de prueba correspondientes a la técnica de partición de equivalencia. Se realizó una prueba para sustentar la idea a defender planteada, utilizando algunas variables para comparar los resultados obtenidos con los mecanismos usados anteriormente para llevar a cabo el proceso de intercambio y actualización de la información y con la solución propuesta en este trabajo de diploma; entre estas variables están el Tiempo, Volumen de información y Capacidad de ejecución.

Conclusiones Generales

Los resultados obtenidos durante el desarrollo de la presente solución arrojaron las siguientes conclusiones:

- Con la investigación de las tecnologías que se utilizaban y se usan para manejar la información generada en los procesos de la Planificación por Objetivos se detectaron numerosas deficiencias que no favorecían el proceso de intercambio y actualización de la información en las entidades cubanas. Detectándose la carencia de un mecanismo que permitiera favorecer este intercambio.
- El análisis y diseño de la solución propuesta permitió la obtención de los artefactos requeridos por el modelo de desarrollo de software por el que se rigió el desarrollo de la misma, especificando cada uno de los requerimientos que llevaron a la integridad, confidencialidad y disponibilidad de la información.
- Se obtuvo la estructura física del sistema y se definieron los estándares de código. Se logró implementar el componente Gestor de Intercambio de Información, una herramienta sencilla, eficiente y capaz de brindarle al usuario la posibilidad de obtener los elementos deseados justo a tiempo, donde por primera vez los planificadores son los principales protagonistas del intercambio de información.
- Las pruebas de caja negra sirvieron para demostrar el correcto funcionamiento de la solución ante los diferentes escenarios. Con la prueba realizada en el MINFAR se corroboró que la solución desarrollada contribuye a que el proceso de intercambio y actualización de la información se realice en menos tiempo, con el volumen de información deseado y una mayor capacidad de ejecución por parte de los planificadores en SIPAC, por lo tanto constituye una mejora en el proceso de planificación vigente en las entidades cubanas.

Recomendaciones

Declarados como cumplidos los objetivos propuestos, se recomienda:

- Implementar la opción de generar claves públicas y privadas para el cifrado de la información a intercambiar.
- Utilizar la solución propuesta como base para el desarrollo de un modulo dirigido a la interoperabilidad, donde se implemente de forma dinámica las dependencias entre SIPAC y cualquier otro sistema para la planificación de actividades.

Referencias Bibliográficas

1. **Artela, Ariadna Rendón y Pérez, Manuel Alejandro Castellanos.** *Modelación y construcción de los componentes.* La Habana : UCI, 2010.
2. **Stoner, James Arthur Finch, y otros.** *Administración.* 1996.
3. **Goodstein, Leonard, Nolan, Timothy y Pfeiffer, J. William.** *Planeación Estratégica Aplicada.* 1998.
4. **PRÁCTICA., LA DIRECCIÓN POR OBJETIVOS. TEORÍA Y.** [En línea]
5. **David, Fred R.** *Conceptos De Administración Estratégica.* Mexico : Pearson Education, 2008.
6. **Bernal Vidal, Néstor y Galán Ramírez, Yuliet.** *Proceso de Planificación por Objetivos en las entidades de las FAR.* La Habana : UCI, 2008.
7. **Salcedo, José Luis Villarroel.** Centro Politécnico Superior Universidad de Zaragoza. *Departamento de Informática e Ingeniería de Sistemas Centro Politécnico Superior Universidad de Zaragoza.* [En línea] www.lcc.uma.es/~luisll/Introduccion_teoria.pdf.
8. **Confederación Granadina de Empresarios.** CGE. *CGE.* [En línea] http://www.cge.es/portalcge/tecnologia/innovacion/4115sistemajust.aspx.
9. **Cornejo, José Enrique González.** DocIRS . [En línea] http://www.dociris.cl.
10. *El Lenguaje Unificado de Modelado(UML).* **Orallo, Enrique Hernández.**
11. **Rumbaugh, James, Jacobson, Ivar y Booch, Grady.** *El Lenguaje Unificado de Modelado.* s.l. : Addison Wesley.
12. **W3C.** World Wide Web Consortium. *W3C.* [En línea] http://www.w3c.es/divulgacion/guiasbreves/hojasestilo.
13. **Ecured.** Ecured. [En línea] http://www.ecured.cu/index.php?title=Especial:Pdfprint&page=Lenguaje_de_Programaci%C3%B3n_Web.
14. **BizAgi.** Bizagi. [En línea] http://www.bizagi.com. 4.
15. **Pérez, Javier Eguíluz.** *Introducción a JavaScript.*
16. **Desarrollo Web.** Desarrollo Web. [En línea] http://www.desarrolloweb.com/articulos/243.php.
17. **Herrera, Ing. Gerardo.** Vaslibre. *Vaslibre.* [En línea] 2006. http://www.vaslibre.org.ve/publicaciones/phpflisol2006.pdf.
18. *Marco de Trabajo para la Definición y Desarrollo de un Data Warehouse Difuso.* **Matamala, Carolina Zambrano y Contreras, Marcela Varas.** s.l. : Universidad de Atacama;Universidad de Concepción.
19. **Gómez Baryolo, Oiner, Morejón Borbón, Yoandry y García Tejo, Darien.** *ARQUITECTURA TECNOLÓGICA PARA EL DESARROLLO DE SOFTWARE.* La Habana : UCI.
20. **Ecured.cu.** Ecured.cu. *Ecured.cu.* [En línea] 1 de Diciembre de 2011. http://www.ecured.cu/index.php/Sencha_Ext_JS.
21. **Framework.zend.** Framework.zend. *Framework.zend.* [En línea] 21 de Abril de 2011. http://framework.zend.com/manual/en/learning.quickstart.intro.html.
22. **Ontuts.** Ontuts. [En línea] http://web.ontuts.com/tutoriales/utilizando-doctrine-como-orm-en-php/.
23. **Pérez, Javier Eguíluz.** Introducción a AJAX. *Librosweb.es.* [En línea]
24. **Cabero, Gerardo Antonio y Maldonado, Daniel.** *Sqlite: Rápido, ágil,liviano y robusto.* s.l. : http://sqlite-latino.blogspot.com/.
25. **Altamirano, Ing. Alfonso Valdez.** Comparativo de Entornos de Desarrollo Integrados. *ubicuos.* [En línea] Mayo de 2009. http://www.ubicuos.com.
26. **Galindez, Rodrigo.** Rodrigogalindez. [En línea] 16 de Mayo de 2008. http://www.rodrigogalindez.com/files/14.pdf.
27. **Maldonado, Daniel M.** Elcodigok. *Elcodigok.* [En línea] 30 de 9 de 2010. http://www.elcodigok.com.ar.
28. **Apache.org.** Apache. *Apache.* [En línea] http://httpd.apache.org/docs/2.0/es/new_features_2_0.html.
29. **Postgresql.** postgresql.org. *postgresql.org.* [En línea] http://www.postgresql.org/about/.
30. **Rotta, Ing. Luis Zuloaga.** Galeon. *Galeon.* [En línea] http://www.galeon.com/zuloaga/Doc/AnalisisRequer.pdf.
31. **Méndez, Gonzalo.** Facultad de Informática Universidad Complutense de Madrid. *Facultad de Informática Universidad Complutense de Madrid.* [En línea] 2008. http://www.fdi.ucm.es/profesor/gmendez/docs/is0809/03-Requisitos.pdf.

32. **Armas, Dayamy Linares.** EMVI. *EMVI.* [En línea] http://www.eumed.net/libros/2010b/698/Requisitos%20funcionales.htm.
33. **Sommerville, Ian.** *Ingeniería del Software.* s.l. : Prentice Hall, 2005.
34. **González, Dianelys Brito.** *ARQUITECTURA DE SOFTWARE.* Habana : CEIGE, 2011. CEIGE-N-00-i001.
35. **Chaviano Gómez, Enrique y Carrascoso Puebla, Yoan Arlet.** *Propuesta de arquitectura orientada a servicios para el módulo de inventario del ERP cubano.* Habana : UCI, 2009.
36. **Tedeschi, Nicolás.** Microsoft. *Microsoft.* [En línea] http://msdn.microsoft.com/es-es/library/bb972240.aspx.
37. **Dpto. de Ingeniería de Sistemas y Computación.** Universidad de los Andes. *Uniandes.* [En línea] http://sistemas.uniandes.edu.co/~isis2701/dokuwiki/lib/exe/fetch.php?media=isis2701-patronesgrasp.pdf.
38. **Visconti, Marcello y Astudillo, Hernán.** Departamento de Informática Universidad Técnica Federico Santa María. [En línea] http://www.inf.utfsm.cl/~visconti/ili236/Documentos/08-Patrones.pdf.
39. **Universidad de Valladolid.** Departamento de Informática . *Departamento de Informática.* [En línea] 2009. http://www.infor.uva.es/~felix/datos/priii/tr_patrones-2x4.pdf.
40. **Diego, Javier Nieto y Fernández, Pablo Ramos.** Departamento de Informática y Automática. [En línea]
41. **Universidad Tecnológica de Aguascalientes.** UTAGS. *UTAGS.* [En línea] http://materias.utags.edu.mx.
42. Java Foundations. [En línea] http://javafoundations.blogspot.com/2010/07/java-estandares-de-programacion.html#1_estandares.
43. **Negro, Ing. Pablo Ariel.** *Umbrales para Métricas Orientadas a Objetos.* s.l. : Universidad Abierta Interamericana, 2008.
44. **Microsoft.** MSDN.Microsoft. *MSDN.Microsoft.* [En línea] 2010. http://msdn.microsoft.com/es-es/library/bb385914.aspx.
45. **Cárdenas, Maria Clara Choucair.** Asociación Colombiana de Ingenieros de Sistemas. *ASCIS.* [En línea] 21 de Septiembre de 2007. http://www.acis.org.co/fileadmin/Base_de_Conocimiento/XXVII_Salon_Informatica/MariaClaraChoucair-PruebasDeSoftware.pdf.
46. **Myers, Glenford J.** *The art of software testing. Third Edition.* 2011 : John Wiley & Sons.
47. **Rojas, Johanna y Barrios, Emilio.** Universidad Distrital Francisco José de Caldas. [En línea] 2007. http://gemini.udistrital.edu.co/comunidad/grupos/arquisoft/fileadmin/Estudiantes/Pruebas/HTML%20-%20Pruebas%20de%20software/node28.html.
48. Mastermagazine. [En línea] 2005. http://www.mastermagazine.info/termino/5286.php.
49. **W3C.org.** W3C.org. *W3C.org.* [En línea] http://www.w3.org/DOM/.
50. **Gómez Baryolo, Oiner, Morejón Borbón, Yoandry y García Tejo, Darien.** *ARQUITECTURA TECNOLÓGICA PARA EL DESARROLLO DE SOFTWARE.* Habana : UCI.
51. **Pressman, Roger.** *Ingeniería de Software. Un enfoque práctico.* 1998.
52. **González, Mairelys Fernández y Rivera, Osley Zorrilla.** *Diseño e implementación del componente Ajuste al Costo del Subsistema Costos y Procesos del Sistema Integral de Gestión de Entidades CEDRUX.* La Habana : UCI, 2010.
53. **Hay, Edward J.** *Just in time: la técnica japonesa que genera mayor ventaja competitiva.* Bogota : Norma, 1989.