UNIVERSIDAD NACIONAL AUTONOMA DE MÉXICO

FACULTAD DE CONTADURÍA Y ADMINISTRACIÓN

SISTEMA PARA LA ADMINISTRACIÓN DE MOVIMIENTOS DERIVADOS DEL FORMATO ÚNICO DE MOVIMIENTOS DE SERVICIOS PERSONALES

DISEÑO DE UN SISTEMA O PROYECTO PARA UNA ORGANIZACIÓN

ALEMÁN HERNÁNDEZ MARÍA DEL PILAR

MÉXICO, D.F. 2012

UNIVERSIDAD NACIONAL AUTONOMA DE MÉXICO

FACULTAD DE CONTADURÍA Y ADMINISTRACIÓN

SISTEMA PARA LA ADMINISTRACIÓN DE MOVIMIENTOS DERIVADOS DEL FORMATO ÚNICO DE MOVIMIENTOS DE SERVICIOS PERSONALES

DISEÑO DE UN SISTEMA O PROYECTO PARA UNA ORGANIZACIÓN QUE PARA OBTENER EL TÍTULO DE:

LICENCIADO EN INFORMÁTICA

PRESENTA:

ALEMÁN HERNÁNDEZ MARÍA DEL PILAR

ASESOR:

L.I. GABRIEL GUEVARA GUTIERREZ

MÉXICO, D.F. 2012

Tabla de contenido.

Agradecimientos

A mis padres:

Quiero agradecer en primer lugar a mis padres, por el infinito apoyo que me han brindado hasta ahora, quien han sido un pilar, un ejemplo a seguir y que día a día me han apoyado infinitamente, les agradezco su paciencia y su amor que me tienen, por ellos es que logré terminar mi carrera.

A mis tíos y a mis abuelitos que siempre estuvieron al pendiente de mí en todo momento, por todos sus consejos y amor que hasta ahora me han dado.

A mis hermanos sin ellos no sabría que hacer su apoyo incondicional, formaron un cimiento para poder sacar adelante este trabajo con mucho esfuerzo y apoyo por parte de ellos.

Agradezco a mi asesor por su paciencia que tuvo conmigo y por compartir parte de su aprendizaje, así como sus consejos, tareas, sugerencias y demás que me permitieron aprender cada día algo diferente.

A mis amigos que siempre estuvieron conmigo y nunca me dejaron de apoyar, porque cuando a ellos he recurrido, de una forma u otra me han ayudado, pero quisiera especialmente agradecerle inmensamente a dos amigos que siempre estuvieron conmigo apoyándome en todo momento, Marco César Camarillo Fuentes por sus buenas recomendaciones y aportaciones; y Julio César Soria Mani por brindarme ese apoyo incondicional, todos sus consejos y palabras de aliento para que yo continuara en el desarrollo de este trabajo, apoyarme y quererme siempre durante todo este tiempo.

Gracias a mis padres, abuelitos, hermanos, tíos y amigos he llegado hasta aquí, sé que no fue fácil el camino que tuve que pasar, me enfrente con varios obstáculos, tuve altas y bajas durante toda mi carrera, pero siempre contando con el apoyo de mi familia, solo me resta decir que aquí no termina esto, apenas es el principio de un nuevo comienzo en mi vida.

Introducción

El Consejo Nacional de Ciencia y Tecnología presenta una necesidad de información donde busca mejorar el servicio de la "Subdirección de Ejecución Presupuestal de los Centros Públicos CONACyT". Donde se lleva a cabo el proceso de movimientos de servicios personales, cuya información se encuentra dispersa y se pretende integrarla en un lugar para que dicha área pueda manipular, actualizar y editar.

En este proyecto se describe cual es su necesidad, como se va a satisfacer y los resultados obtenidos de dicho proceso.

El capitulo primero representa el marco referencial del trabajo, es decir abarca la historia, misión y visión del CONACyT, el proceso actual, los objetivos, el ambiente tecnológico, y la descripción de la necesidad de la "Subdirección de Ejecución Presupuestal de los Centros Públicos CONACyT".

En el segundo capítulo abarca el método de desarrollo de software, en el cual se describe el *Rational Unified Process*, sus fases, sus productos, las mejores prácticas, las disciplinas de proceso y de soporte. También se describe la *eXtreme Programming*, las fases con las que cuenta esta. Y una breve descripción de lo que es el *framework* de *Grails*.

El capítulo tercero se denomina desarrollo del sistema, en donde se describe los objetivos del proyecto, el alcance, la lista de stakeholders, la estructura de descomposición del trabajo, la lista de riesgos, la estimación tiempo-costo, la visión del sistema, la lista de requerimientos, la matriz de trazabilidad, los modelos y especificaciones de los casos de uso, el modelo de comportamiento y la interfaz grafica de usuario.

Y finalmente un anexo en el cual viene el formato único de movimientos de servicios personales que se utiliza en la "Subdirección de Ejecución Presupuestal de los Centros Públicos CONACyT", las conclusiones, un glosario y las referencias.

CAPÍTULO PRIMERO. MARCO REFERENCIAL

El trabajo que se presenta a continuación está dirigido al Consejo Nacional de Ciencia y Tecnología (CONACyT), el cual se describe a continuación una breve historia, su misión y visión.

1.1 Historia del CONACyT(CONACyT, Historia del CONACYT, 2011)

El Consejo Nacional de Ciencia y Tecnología (CONACyT), fue creado por disposición del H. Congreso de la Unión el 29 de diciembre de 1970, como un organismo público descentralizado de la Administración Pública Federal, integrante del sector educativo, con personalidad jurídica y patrimonio propio. También es responsable de elaborar las políticas de ciencia y tecnología en México. A partir de 1999 se presentaron dos reformas y una ley para coordinar y promover el desarrollo científico y tecnológico y en el año de 1970 se crea el CONACyT.

1.2 Misión(CONACyT, Historia del CONACYT, 2011)

Impulsar y fortalecer el desarrollo científico y la modernización tecnológica de México, mediante la formación de recursos humanos de alto nivel, la promoción y el sostenimiento de proyectos específicos de investigación y la difusión de la información científica y tecnológica.

1.3 Visión

Contribuir conjuntamente con otras dependencias y entidades del Gobierno Federal, así como del sector productivo a que México tenga una mayor participación en la generación, adquisición y difusión del conocimiento a nivel internacional, y a que la sociedad aumente considerablemente su cultura científica y tecnológica, disfrutando de los beneficios derivados de esta.

1.4 Objetivos

El sistema propuesto está dirigido al Consejo Nacional de Ciencia y Tecnología, en la "Subdirección de Ejecución Presupuestal de los Centros Públicos CONACyT"que pertenece a su vez a la "Dirección de Planeación, Programación y Presupuesto de los Centros Públicos CONACyT", la cual forma parte de la "Dirección Adjunta de Posgrado y becas".

La "Dirección Adjunta de Posgrado y becas" describe los siguientes objetivos generales:

- Conducir la planeación, programación y el presupuesto de los recursos de los programas de la Dirección Adjunta con el propósito de buscar su optimización.
- Dar seguimiento al ejercicio del presupuesto de los programas de la Dirección Adjunta y proponer acciones que permitan optimizar los recursos asignados a los programas de la misma.
- Establecer los lineamientos y criterios para la definición, identificación y medición de los programas de la Dirección Adjunta para evaluar su cumplimiento.
- Medir y monitorear el desempeño de los programas de la Dirección Adjunta en el marco del proceso de Planeación, Programación e Integración del Presupuesto de Egresos de la Federación para determinar avances en el cumplimiento de metas y objetivos.
- Proporcionar la información disponible e insumos necesarios, así como los lineamientos para la realización de diversos estudios en materia de fomento, formación, desarrollo y vinculación de recursos humanos de alto nivel, cuando sean requeridos.
- Coordinar la evaluación de los programas sujetos a reglas de operación de la Dirección Adjunta que realizan instancias externas, en términos de la normativa aplicable para, en su caso, reorientar sus políticas.
- Dar a conocer el impacto y los resultados de la Dirección Adjunta en diversos foros y medios con el fin de que sus efectos sean reconocidos en la sociedad.
- Proponer y operar las políticas y procesos para la generación de información estratégica de los programas de la Dirección Adjunta.
- Atender las demás funciones y tareas que le asignen las disposiciones legales y administrativas en el ámbito de su competencia, así como las que le sean encomendadas por la Dirección Adjunta de Posgrado y Becas.

La "Dirección de Planeación, Programación y Presupuesto de los Centros Públicos CONACyT", se encarga de integrar y proporcionar la información y documentación de los programas de la Dirección Adjunta que le sea requerida por las instancias, dependencias o instituciones, de acuerdo con las disposiciones legales aplicables(CONACyT, Manual de Organización, 2011).

1.5 Subdirección de Ejecución Presupuestal de los Centros Públicos CONACyT

La petición del sistema la da la "Subdirección de Ejecución Presupuestal de los Centros Públicos CONACyT"(CONACyT, Manual de Organización, 2011) la cual se encarga de:

• Supervisar, ministrar recursos, coordinar y dar seguimiento a la información programática, presupuestal y organizacional de servicios personales de los Centros Públicos de Investigación CONACyT.

• Verificar y proporcionar información programática y presupuestal así como de servicios personales a las áreas internas y externas (Contraloría Interna, Dirección Adjunta de Administración y Finanzas (DAAF), Secretaria de Hacienda y Crédito Público (SHCP), Secretaría de la Función Pública (SFP).

• Supervisar y coordinar el proceso de promoción del personal científico y tecnológico de las Entidades Coordinadas por el CONACyT de acuerdo con la normatividad aplicable para dar respuesta a las solicitudes generadas por las Comisiones de Evaluación Internas y Externas.

1.6 Procesos

La "Subdirección de Planeación, Programación y Presupuesto de los Centros Públicos CONACyT", es el área usuaria principal, la cual tiene los siguientes procesos:

- Cancelación de plazas,
- Creación de plazas,
- Incremento salarial,
- Conversión por promoción de plazas.

1.7 Ambiente tecnológico

El ambiente tecnológico con que cuenta la "Subdirección de Planeación, Programación y Presupuesto de los Centros Públicos CONACyT", representa una variable que está afectando los procesos actuales de esta subdirección ya que el personal a cargo de estos, solo maneja los conocimientos básicos en Excel por lo que en ocasiones ocurren en errores de captura.

Por otro lado las características de los equipos de cómputo que emplean son:

- Pentium 4 de 2.6 a 3.0 ghz.
- 80 GB de Disco Duro.
- 1 Gb de Memoria Ram.
- Unidad de Óptica CDRW/DVD.
- Monitor LCD 17 pulgadas AAA.

La "Subdirección de Ejecución Presupuestal de los Centros Públicos CONACyT", quien tiene a su cargo 26 centros, lleva a cabo los cálculos de las distintas prestaciones que tiene cada tipo de personal por cada centro al que administra. Así como, llevar un control de los tipos de movimientos que maneja mediante el Formato Único de Movimientos de Servicios Personales, los cuales son: incremento salarial, conversión por promoción, la creación y cancelación de una plaza.

1.8 Descripción de la necesidad

Las actividades anteriores se hacen de forma manual empleando formatos en Excel que no se puede compartir recurrentemente, no facilita la incorporación de nuevas plazas, tiene fallas en los cálculos. Lo anterior provoca que no haya eficiencia en el proceso por lo que se considera que un sistema ayuda a solucionar el problema, ya que la automatización reduce los tiempos y minimiza los errores que se presentan recurrentemente de captura.

El sistema tiene dos objetivos: agilizar los procesos reduciendo los errores de captura y reducir los tiempos por medio de la automatización, para proporcionar información programática y presupuestal de los servicios personales a las áreas internas y externas (Contraloría Interna, Dirección Adjunta de Administración y Finanzas (DAAF[1]), Secretaria de Hacienda y Crédito Público (SHCP), Secretaría de la Función Pública (SFP), emanada

[1] Ver glosario.

del ejercicio de los recursos autorizados a las Entidades Coordinadas por el CONACyT, con la finalidad de facilitar la toma de decisiones.

1.9 Diagrama de actividades

El diagrama de actividad(Pressman, 2004) es una especialización del diagrama de estado de UML, organizado respecto de las acciones y usado para especificar: un método, un caso de uso, un proceso de negocio (*Workflow*).

Las actividades se enlazan por transiciones automáticas. Cuando una actividad termina se desencadena el paso a la siguiente actividad. En el siguiente diagrama se muestra el proceso actual de la "Subdirección de Ejecución Presupuestal de los Centros Públicos CONACyT".

Figura 1.1 Diagrama de actividades calcular prestaciones. (Elaboración propia)

CAPÍTULO SEGUNDO. MÉTODO DE DESARROLLO DE SOFTWARE

Hoy en día es muy difícil implementar un método de desarrollo de software(María, 2011) que se adapte al tipo de proyecto porque en muchos casos suele haber grandes riesgos y es complicado el control sobre todo por la demanda actual por parte de los usuarios para desarrollar un software. Sin embargo, es de suma importancia contar con un método para poder desarrollar un software de calidad que cumpla los requerimientos del usuario.

2.1 RUP (Rational Unified Process)

El método a ocupar es el de RUP (IBM, Rational Unified Process:Best Practices for software development teams, 2011). *Rational Unifed Process* es un proceso de ingeniería de software, el cual proporciona un acercamiento disciplinado a la asignación de tareas y responsabilidades en una organización de desarrollo. Su propósito es asegurar la producción de software de alta calidad que se ajuste a las necesidades de los usuarios finales con un calendario y presupuesto predecibles.

La versión de RUP(Philippe, 2012) es la 2007, la cual incluye planes de trabajo que proporcionan una visión general de cómo aplicar el proceso a una amplia variedad de proyectos y tecnologías, análisis amplio de las pruebas, que abarcan el ciclo completo, en la mejora de la cobertura del diseño de interfaz de la aplicación - especialmente en su aplicación al desarrollo de aplicaciones web eficaces, así como ideas sobre el diseño de sistemas usando patrones y marcos que son diseñados con la intención de facilitar el desarrollo de software, permitiendo a los diseñadores y programadores pasar más tiempo identificando requerimientos de software(Wikipedia, Framework, 2011).

RUP es un proceso de estructura que puede ser adaptado y extendido para ajustarse a las necesidades que adopta una Organización. Se divide en 4 fases:

Inicio: Determinar la visión del proyecto.

Elaboración: Determinar la arquitectura óptima.

Construcción: Obtener la capacidad de operación inicial.

Transición: Liberar el proyecto y el producto.

Cada una de estas etapas es desarrollada mediante iteraciones, las cuales consisten en reproducir el ciclo de vida en cascada a menor escala. Los objetivos de una iteración se establecen en función de la evaluación de las iteraciones precedentes. Se desarrolla por cada iteración, llevada bajo dos disciplinas: 6 de proceso y 3 de soporte.

RUP rescata varias de las "mejores prácticas" en el desarrollo de software moderno en una forma que se adapta al tamaño de los proyectos y las organizaciones.

2.2 Mejores prácticas

El concepto de Mejores Prácticas -*Best Practices*- fue acuñado en 1993 por Hammer and Champy(Saffirio, 2012) para identificar siete reglas o indicaciones para modelar procesos de negocios.

Hoy la Mejor Práctica corresponde a un modelo completamente definido, cuyos buenos resultados operacionales están comprobados y que está disponible para ser utilizada. (Philippe, 2012)A continuación se detallan las 6 mejores prácticas para el desarrollo de software:

2.2.1 Desarrollo de software iterativo

En el *Rational Unified Process*, el enfoque iterativo es muy controlado; las iteraciones se han previsto en el número, duración y objetivos. Las tareas y responsabilidades de los participantes están definidas. Algún cambio se lleva a cabo de una iteración a otra, tratando de ser controladas.

2.2.2 Administración de requerimientos

La administración de requerimientos es un enfoque sistemático para promover, organizar, comunicar y gestionar las necesidades cambiantes de un sistema de software intensivo en una aplicación. Los beneficios de la administración de los requerimientos efectivos son los siguientes:

1. Un mejor control de proyectos complejos.
2. Mejora de la calidad del software y la satisfacción del cliente.
3. Reducción de los costos del proyecto y los retrasos.
4. Mejora de la comunicación en equipo.

2.2.3 Arquitectura de uso basada en componentes

Los casos de uso conducen el *Rational Unified Process* en todo el ciclo de vida, pero las actividades de diseño se centran en la noción de la arquitectura, ya sea la arquitectura del sistema (De acuerdo al *Software Engineering Institute* (SEI), la Arquitectura de Software(L. Bass, 2003) se refiere a "las estructuras de un sistema, compuestas de elementos con propiedades visibles de forma externa y las relaciones que existen entre ellos."), o para sistemas de software intensivos. El objetivo principal de las iteraciones iniciales del proceso es la de producir y validar una arquitectura de software que en el ciclo de desarrollo inicial, toma la forma de un prototipo de la arquitectura ejecutable que evoluciona gradualmente para convertirse en el sistema final en iteraciones posteriores.

2.2.4 Modelo visual de software

El Lenguaje de Modelado Unificado (UML) es un lenguaje gráfico para visualizar, especificar, construir y documentar los artefactos de un sistema de software intensivo. UML(IBM, Rational Unified Process:Best Practices for software development teams, 2011) proporciona una forma estándar de escribir los planos del sistema(IBM, UML, RUP, and the Zachman Framework: Better together), que abarca los elementos conceptuales tales como procesos de negocio y las funciones del sistema, así como elementos concretos, también como las clases escritas en un lenguaje de programación en específico, esquemas de bases de datos, y componentes de software reutilizables.

2.2.5 Verificación continúa de la calidad de software

1. La calidad del producto: La calidad de los productos principales que se producen (el software o el sistema) y todos los elementos que comprende (por ejemplo, componentes, subsistemas, arquitectura, entre otros).
2. Los procesos de calidad: El grado en que un proceso es aceptado (incluyendo las medidas y los criterios de calidad) se basa aplicando la observación en la fabricación del producto. Además, la calidad del proceso es intensivo por la calidad de los artefactos (como los planes de iteración, planes de prueba, las realizaciones de casos de uso, y el modelo de diseño) en apoyo al producto principal.

2.2.6 Control de cambios de software

En un desarrollo iterativo, muchos productos de trabajo son modificados a menudo. Al permitir la flexibilidad en la planeación y ejecución del desarrollo, permite que los requerimientos sean iterativos en el desarrollo y hace hincapié en las cuestiones esenciales de seguimiento de los cambios y asegura que todo esté sincronizado.

Se centra en las necesidades de desarrollo organizacional, la gestión del cambio es un proceso sistemático enfocado en los cambios de los requerimientos, el diseño e implementación. También cubre las actividades de seguimiento, los malentendidos y los compromisos de proyectos, así como la asociación de estas actividades con artefactos específicos.

2.3 Estructura estática: Descripción del proceso

Un proceso describe "quien", en este caso son los roles, está realizando "que" acciones para llegar al producto final, "como" lo llevan a cabo definiendo las actividades en las que están involucrados y "cuando" lo realizaran administrando de manera correcta el proyecto. *Rational Unified Process* se representa mediante cuatro elementos de modelado primarios:

2.3.1 Los roles: QUIENES

El concepto principal en un proceso son los roles. Un rol define los comportamientos y responsabilidades de un solo individuo o de un grupo de personas que trabajan en equipo, el comportamiento se representa en término de *actividades*, el desempeño del rol y cada uno de estos es asociado con un conjunto de actividades en común. La responsabilidad de cada rol normalmente se expresa en relación de cierto *artefacto*[2] que el rol crea, modifica o controla.

Los siguientes son ejemplos de los roles:

- Analista de Sistemas: Una persona que actúa como analista de sistemas dirige y coordina el planteamiento de requerimientos y casos de uso modelado para la funcionalidad y delimitación del sistema.

- Diseñador: Una persona que actúa como diseñador define las responsabilidades, operaciones, atributos y relaciones de una o más clases, también determina la forma en que se debe ajustar el entorno de la aplicación.

[2] Ver glosario.

- Diseñador de Pruebas: Una persona que actúa como un diseñador de pruebas es el responsable de la planeación, diseño, implementación y evaluación de las pruebas, incluida la generación del plan y el modelo de prueba, la aplicación de los procedimientos, y la evaluación que cubra las pruebas, los resultados y la eficacia.

2.3.2 Actividades: COMO

Los roles tienen las actividades que definen el trabajo que realizan. Una actividad es una unidad de trabajo que a una persona se le puede pedir para llevar a cabo y que produce un resultado significativo en el contexto del proyecto. La actividad tiene un propósito claro, por lo general muestra la creación o actualización de los artefactos, como un documento, un modelo, una clase o un plan.

La granularidad de una actividad por lo general dura de unas cuantas horas o hasta varios días, la actividad debería ser utilizada como elemento de planeación y de progreso, si es demasiado pequeño, será descartado, y si es demasiado grande, el progreso tendrá que ser dividido en varias actividades.

Las actividades pueden repetirse varias veces en el mismo artefacto, sobre todo de una iteración a otra en tanto el sistema vaya refinándose y expandiéndose. Las actividades repetitivas pueden ser realizadas por un mismo rol, pero no necesariamente el mismo individuo.

Las actividades están divididas en pasos. Los pasos se dividen en tres categorías principales:

- Medidas pensamiento: El trabajador entiende la naturaleza de la tarea, recopila y analiza los artefactos de entrada, y formula el resultado.
- Realizar los pasos: El trabajador crea o actualiza algunos artefactos.
- Revisar las medidas: El trabajador inspecciona los resultados con algunos criterios.

No todos los pasos se realizan necesariamente cada vez que se invoca una actividad, por lo que se puede expresar en forma de flujos alternativos.

2.3.3 Artefactos: QUE

Los artefactos pueden tomar varias formas o modalidades:

- Un documento, como un caso de negocio o documento de arquitectura de software.
- Un modelo, como el de casos de uso o el modelo de diseño.
- Un elemento se modela como una clase, un caso de uso, o como un subsistema.
- El código fuente.
- Los ejecutables.

Se debe tener en cuenta que el artefacto es el término utilizado por el *Rational Unified Process*. Otros procesos utilizan términos tales como los productos de trabajo, la unidad de trabajo, para denotar lo mismo. Los términos son sólo el subconjunto de todos los artefactos que terminan en las manos de los clientes y usuarios finales.

2.3.4 Los flujos de trabajo: CUANDO

La numeración de todos los roles, actividades y artefactos no constituyen el proceso. Se necesita una forma para describir secuencias significativas de las actividades que producen algún resultado valioso y para mostrar las interacciones entre los roles.

Un flujo de trabajo es una secuencia de actividades que produce un resultado de valor en términos de UML, un flujo de trabajo se puede representar con un diagrama de secuencia, un diagrama de colaboración, o un diagrama de actividades.

Se debe tener en cuenta que no siempre es posible representar todas las dependencias entre actividades. A menudo, dos actividades que tengan relación con el sistema se deben señalar, sobre todo cuando se refieren a un mismo rol.

Existen varias formas de organizar el conjunto de actividades en los flujos de trabajo. Sin embargo el *Rational Unified Process* utiliza lo siguiente:

- Flujos de trabajo básicos.
- Datos del flujo de trabajo.
- Planes de iteración mecánicos.

2.4 Estructura dinámica: Desarrollo iterativo

Si el modelo de ciclo de vida en cascada se considera para determinar el éxito de los proyectos a corto plazo o con una pequeña estimación del riesgo, ¿por qué no romper varios ciclos de vida en cascada? De esta manera, se pueden abordar algunos requerimientos y algunos riesgos, un poco de diseño, implementación, validaciones, para después recopilar más requisitos, hasta que se haya terminado esa iteración. Así es como funciona el método iterativo.

Desde la perspectiva de la administración de proyectos, se necesita una manera de evaluar el progreso para asegurarse de que no se está sin rumbo de iteración a iteración, pero que en realidad convergen en un producto. Desde el punto de vista de la administración, se debe definir los puntos en el tiempo para operar las funciones de las actividades periódicamente basadas en criterios claros. Estos hitos proporcionan puntos en los que se toma la decisión de seguir adelante, cancelar, o cambiar de rumbo. Por último, se debe dividir y organizar la secuencia de interacciones específicas de acuerdo a los objetivos a corto plazo. Los avances se miden por el número de casos de uso completos, casos de prueba, y un rendimiento satisfactorio.

2.4.1 Disciplinas de Proceso

2.4.1.1 Modelo de negocio

Los objetivos de los modelos de negocio son los siguientes:

- Comprender la estructura de la organización en la que un sistema se va a implementar (la organización destino).
- Comprender los problemas actuales de la organización objetivos e identificar posibilidades de mejora.
- Para que entre los clientes, usuarios finales y desarrolladores exista un entendimiento común de la meta organizacional.
- Obtener los requisitos del sistema necesarios para apoyar la organización destino.

Para alcanzar estos objetivos, el flujo de trabajo de modelado de negocio describe cómo desarrollar una visión de la nueva meta y en base a esta visión se definen los procesos, roles y responsabilidades de la organización en el modelo de negocio. Este modelo cuenta con casos de uso y un modelo de negocio.

2.4.1.2 Requisitos

Se define como un requisito: una condición o capacidad que un sistema debe cumplir. Los objetivos del flujo de trabajo son los siguientes:

- Establecer y mantener un acuerdo con los clientes y otras partes interesadas en lo que el sistema debe de hacer y ¿por qué?
- Proporcionar a los desarrolladores del sistema, una mejor comprensión de los requisitos del sistema.
- Definir los límites del sistema. Para proporcionar una base para la planeación de los contenidos técnicos de las iteraciones.
- Proporcionar una base en la estimación de costos y tiempo para desarrollar el sistema.
- Definir una interfaz de usuario del sistema, centrándose en las necesidades y objetivos de los usuarios.

2.4.1.3 Análisis y diseño

El propósito del flujo de trabajo de análisis y diseño es traducir los requisitos en una especificación que describe cómo implementar el sistema. Para hacer esta traducción, se deben comprender los requisitos y transformarlos en un diseño de sistema mediante la selección de la mejor estrategia de implementación. Al comienzo del proyecto, se debe establecer una arquitectura robusta para que se pueda diseñar un sistema que sea fácil de entender, construir y desarrollar. Se deberá ajustar el diseño para que coincida con el medio ambiente de aplicación, rendimiento, robustez, escalabilidad y capacidad de prueba, entre otras.

2.4.1.4 Implementación

El flujo de trabajo de implementación tiene cuatro propósitos:

- Definir la organización del código en términos de subsistemas de aplicación organizadas en capas para llevar a cabo las clases y objetos en componentes (archivos de código fuente, binarios, ejecutables, y otros).
- Probar los componentes desarrollados como unidades.
- Integrar en un sistema ejecutable los resultados producidos.
- El alcance de la prueba dentro del flujo de trabajo de aplicación se limita a la prueba de la unidad de los componentes individuales. El sistema de prueba y

ensayo de la integración se describen en el flujo de trabajo de pruebas. Esto se debe a que el implementador es responsable de la unidad de prueba.

2.4.1.5 Pruebas

El propósito de las pruebas es evaluar la calidad del producto. Esto implica no sólo el producto final, sino desde el principio del proyecto con la evaluación de la arquitectura continúa a través de la evaluación final, con los productos entregados a los clientes. Las pruebas consisten en lo siguiente:

- Verificación de las interacciones de los componentes.
- Verificación de la correcta integración de los componentes.
- Verificar que todos los requisitos se han aplicado correctamente.
- Identificar y asegurar que todos los defectos encontrados se traten antes de que el software sea implementado.

2.4.1.6 Despliegue

El propósito de la implementación es el producto de software terminado para los usuarios. El despliegue incluye los siguientes tipos de actividades:

- Pruebas del software en su entorno final (beta).
- Empaquetado del software para la entrega.
- Distribución del software.
- Instalación del software.
- Capacitación a los usuarios finales.
- Migración de software existente.

Estas actividades que se llevan a cabo varían mucho en la industria del software, dependiendo del tamaño de proyecto, el tipo de distribución y el contexto empresarial.

2.4.2 Disciplinas de Soporte

2.4.2.1 Administración del proyecto

La administración de proyectos es el arte de equilibrar los objetivos de la competencia, administrar los riesgos y las restricciones para entregar un producto que satisfaga las necesidades de los usuarios finales. El objetivo de los flujos de trabajo en la administración de proyectos del *Rational Unified Process* es hacer la tarea tan fácil como

sea posible y proporcionar orientación en esta área. No es un procedimiento que se deba llevar al pie de la letra, sino que representa un enfoque para la administración de proyectos que mejora las posibilidades de distribución del software con éxito. La administración del proyecto tiene tres propósitos:

- Proporcionar un marco para la administración de proyectos intensivos en software.
- Proporcionar directrices prácticas para la planeación, la selección de personal, ejecución y seguimiento del proyecto.
- Proporcionar un marco para la administración de riesgos.

2.4.2.2 Administración de la configuración y cambio

El propósito de la administración de la configuración y el cambio es seguir y mantener la integridad de la evolución de los activos del proyecto. Durante el transcurso del ciclo de desarrollo se crean muchos objetivos valiosos. El desarrollo de estos artefactos es un trabajo intensivo, ya que representan una inversión significativa. Estos artefactos evolucionan, sobre todo en el desarrollo iterativo, que se actualiza una y otra vez. Generalmente uno de los roles es responsable de un artefacto, aunque los miembros del equipo del proyecto deben ser capaces de identificar y localizar los artefactos, para seleccionar la versión adecuada de cada uno.

2.4.2.3 Entorno

El propósito del flujo de trabajo del entorno es el de apoyar a la organización de desarrollo con los procesos y herramientas necesarios. Este apoyo incluye lo siguiente:

- Configuración de las herramientas para adaptarse a la organización.
- Procesos de configuración.
- Procesos de mejora.
- Servicios técnicos para apoyar el proceso: las tecnologías de la información (TI), copias de seguridad, entre otros.

Fases del ciclo en RUP

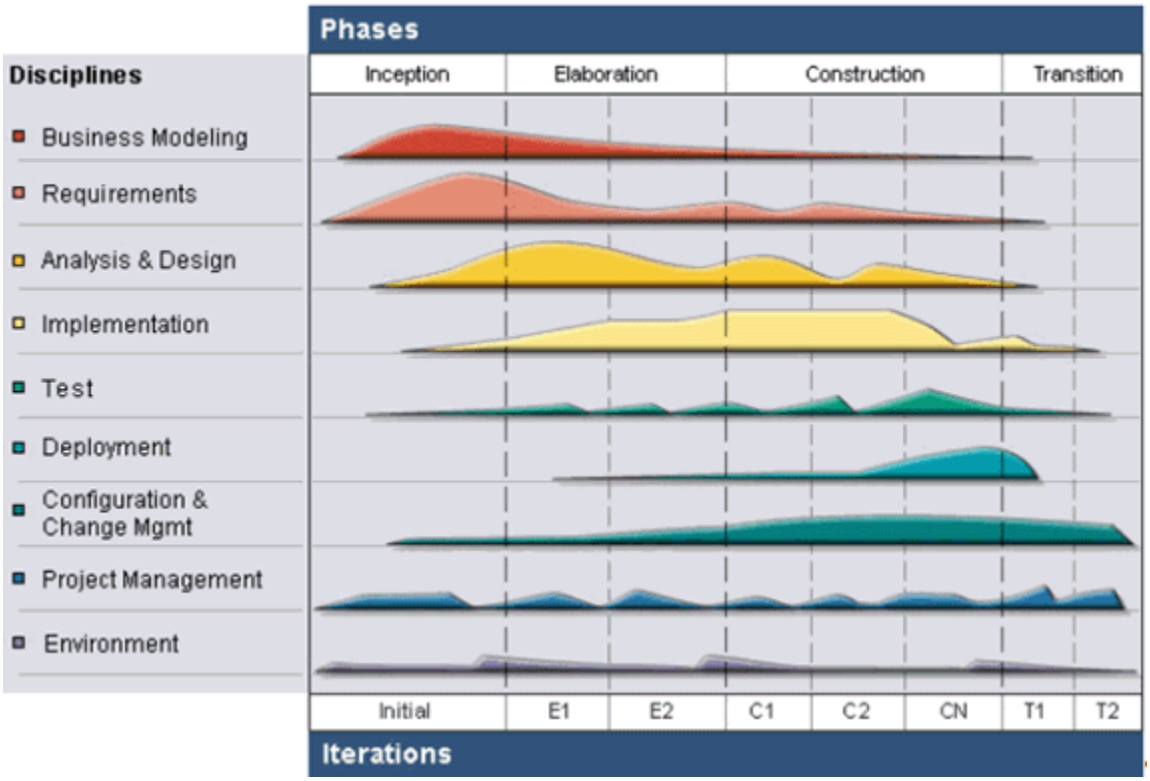


Figura 1.2 Fases del ciclo en RUP. (IBM, UML, RUP, and the Zachman Framework: Better together s.f.)

2.4.3.1 Fase de Inicio

Antes de iniciar un proyecto se requiere de la estimación de todos los requisitos, para determinar cuál es el objetivo, si es factible o no, cuánto va a costar, etcétera. Los objetivos principales de esta fase son:

- Establecer el ámbito del proyecto y sus límites.
- Encontrar los casos de uso críticos del sistema, los escenarios básicos que definen la funcionalidad.
- Mostrar al menos una arquitectura candidata para los escenarios principales.
- Estimar el costo en recursos y tiempo del proyecto.
- Estimar los riesgos, las fuentes de incertidumbre.

Los productos de la fase de inicio son los siguientes:

- Un documento de visión: visión general de los requerimientos del núcleo del proyecto.
- Un modelo de caso de uso inicial (completo).
- Un glosario inicial del proyecto.
- Un modelo de negocios inicial: el cual incluirá el contexto del producto, criterios de éxito y la valoración financiera.
- Evaluación de los riesgos.
- Plan de proyecto: mostrando las fases e iteraciones.

Hito de la fase de inicio: *Objetivos del ciclo de vida*

Al término de la fase de inicio el primer hito de importancia para el proyecto son: los objetivos del ciclo de vida.

Criterios de evaluación para la fase de inicio:

- Los participantes definen el alcance en cuanto al programa de estimación de costos.
- Requerimientos de entendimiento como evidencia de la fidelidad de los casos de uso.
- Credibilidad en la estimación de costos, prioridades, riesgos y procesos de desarrollo.
- Profundidad y amplitud de cualquier prototipo arquitectural que haya sido desarrollado.
- Comparar los gastos que se han requerido hasta ahora con los gastos planeados.

2.4.3.2 Fase de Elaboración

El propósito de esta fase es analizar el dominio del problema, establecer las bases de la arquitectura, actualizar el plan del proyecto y eliminar los riesgos mayores. Cuando termina esta fase se llega al punto de no retorno del proyecto. En esta fase se construye un prototipo de la arquitectura, que debe evolucionar en iteraciones sucesivas hasta convertirse en el sistema final. Este prototipo debe contener los casos de uso críticos identificados en la fase de inicio. También, debe demostrarse que se han evitado los riesgos más graves.

Los objetivos de esta fase son:

- Definir y validar la arquitectura.
- Completar la visión del sistema.
- Crear un plan fiable para la fase de construcción. Este plan puede evolucionar en sucesivas iteraciones. Debe incluir los costos si procede.
- Demostrar que la arquitectura propuesta soportará la visión con un costo y tiempo razonable.

Los productos de la fase de elaboración son los siguientes:

- Todos los modelos de casos de uso y actores deben de estar bien definidos, y las descripciones deben estar desarrolladas.
- Descripción de la arquitectura de software.
- Un prototipo ejecutable de la arquitectura.
- Una lista de riesgos al igual que el modelo de negocio.
- Un plan de desarrollo para el proyecto en general, incluyendo el plan de proyecto de granularidad, mostrando las iteraciones y el criterio de evaluación para cada una.
- Un caso de desarrollo actualizado especificando el proceso a usar.
- Un manual de usuario preliminar (opcional).

Hito de la fase de elaboración: *Ciclo de vida de la arquitectura*

Al final de la fase de elaboración es el segundo hito importante para el proyecto. En este punto se examinan los objetivos del sistema a detalle y el alcance, la arquitectura elegida y la resolución de los riesgos mayores.

El principal criterio de evaluación para esta fase implica la respuesta a las siguientes preguntas:

- ¿La visión del producto es estable?
- ¿La arquitectura es estable?
- ¿Existe alguna demostración ejecutable que muestre elementos de riesgo mayor que los elementos localizados y la credibilidad para resolverlos?
- ¿El plan para la fase de construcción es suficientemente detallado?

- ¿Todos los participantes están de acuerdo en que la visión actual puede ser lograda si el plan es ejecutado para realizar el sistema completo, en el contexto de la arquitectura actual?
- ¿Los gastos actuales en comparación a los gastos planeados son aceptables?

2.4.3.3 Fase de Construcción

La finalidad de esta fase es alcanzar la capacidad operacional del producto de forma incremental a través de las iteraciones sucesivas. Durante esta fase todos los componentes, características y requisitos, que no hayan sido realizadas hasta ahora, se deben implementar, integrar y probar, obteniendo una versión del producto que se ponga en manos de los usuarios (una versión beta).

Los objetivos concretos incluyen:

• Minimizar los costos de desarrollo mediante la optimización de recursos y evitando el tener que rehacer un trabajo o incluso desecharlo.

• Conseguir una calidad adecuada tan rápido como sea práctico.

• Conseguir versiones funcionales (alfa, beta, y otras versiones de prueba) tan rápido como sea práctico.

Hito de la fase de construcción*: Capacidad operativa inicial*

Al final de la fase de construcción viene el tercer hito de importancia, en este punto se decidirá si el software, la situación y los usuarios están listos para la fase de operación, sin exponer el proyecto a riesgos mayores. Este lanzamiento es llamado como “beta”.

El criterio de evaluación para esta fase implica la respuesta a lo siguiente:

- ¿El producto liberado es lo suficientemente estable para ser compartido con los usuarios?
- ¿Todos los participantes están listos para la transición en la comunidad del usuario?
- ¿Los gastos actuales en comparación a los gastos planeados son aceptables?

2.4.3.4 Fase de Transición

La finalidad de la fase de transición es poner el producto en manos de los usuarios finales, por lo que se requiere desarrollar nuevas versiones actualizadas del producto, completar la documentación, entrenar al usuario en el manejo del producto, y tareas en general relacionadas con el ajuste, configuración, instalación y uso del producto.

Los principales objetivos de esta fase son:

• Conseguir que el usuario se valga por sí mismo.

• Un producto final que cumpla los requisitos esperados, que funcione y satisfaga suficientemente al usuario.

Hito: Liberación del producto

Al finalizar la fase de transición se decidirá si los objetivos se cumplieron y si se debe empezar otro ciclo de desarrollo. En algunos casos este hito coincide al final de la fase de inicio para el siguiente ciclo. El criterio principal de evaluación para la fase de transición incluye la respuesta a las siguientes preguntas.

- ¿El usuario está satisfecho?
- ¿Se cumplieron los objetivos?
- ¿Los gastos actuales con los gastos planeados siguen siendo aceptables?

2.5 *eXtreme Programming (XP)*

El desarrollo de software ágil(PCMAG.com, Scrum Definition from PC Magazine Encyclopedia, 2012) es un término genérico para una variedad de mejores prácticas en desarrollo de software. Estos métodos han demostrado ser más eficaces en el tratamiento de las necesidades cambiantes durante la fase de desarrollo. Los métodos ágiles enfatizan el trabajo en equipo, la implicación del cliente y la creación frecuente de pequeñas piezas, del trabajo en todo el sistema.

EXtreme Programming(PCMAG.com, Definition of: XP, 2012) es un proceso ágil para el desarrollo de software ágil que hace énfasis en la implicación del cliente y el trabajo en equipo. Fue desarrollado por Kent Beck, Ward y Cunningham Ron Jeffries, que principalmente se basa en un conjunto formal de reglas sobre cómo se desarrolla la funcionalidad de una prueba antes de escribir el código y el diseño es sencillo, es decir, nunca se debe codificar más de lo necesario.

eXtreme Programming está diseñado para dirigir el proyecto correctamente en lugar de concentrarse en las fechas previstas de reuniones, que son a menudo un poco irreales en este negocio. Utiliza un enfoque orientado a objetos(Pressman, 2004) como su paradigma de desarrollo preferido. *EXtreme Programming* abarca un conjunto de reglas y prácticas que ocurren en el contexto de cuatro actividades del marco de trabajo: planeación, diseño, codificación y pruebas. En la figura 1.2 se ilustra el proceso de la e*Xtreme Programming (EP)*:

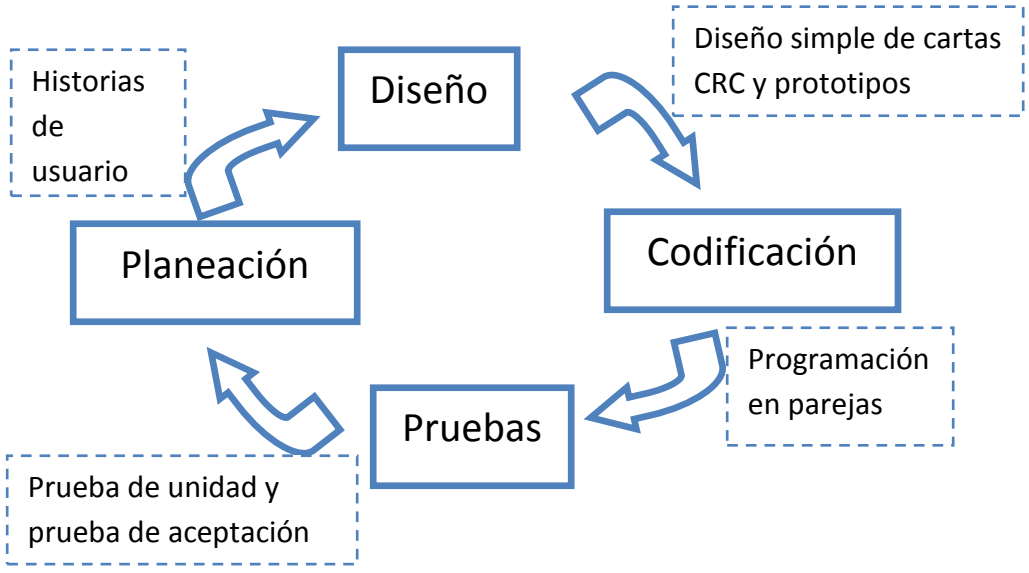


Figura 1.3 Proceso de la programación extrema. Adaptada de (Pressman 2004)

A continuación se describe las prácticas básicas de la eXtreme Programming:

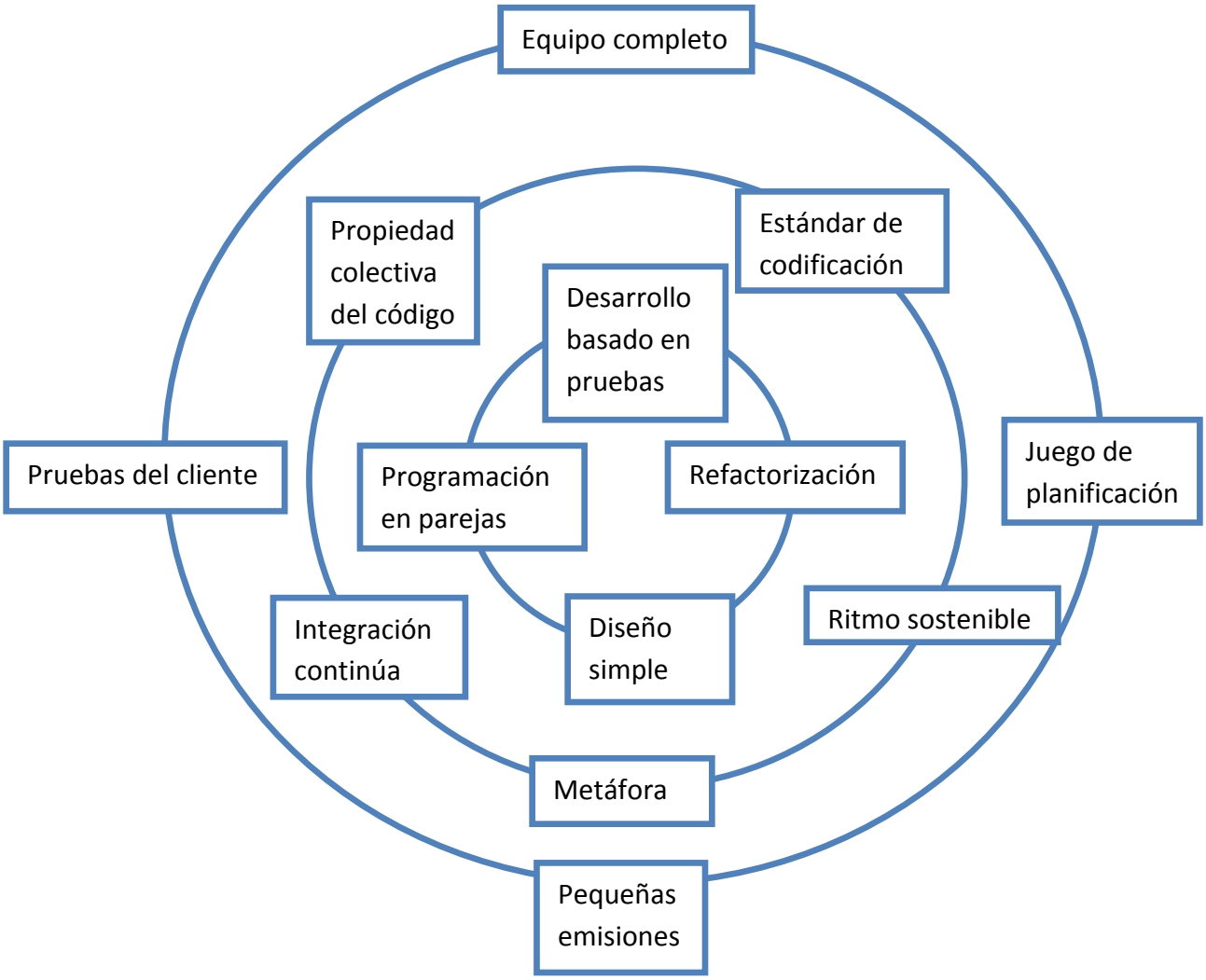


Figura 1.4 Basic *eXtreme Programming*. (XProgramming.com 2012)

Las actividades clave de la *eXtreme Programming* son:

2.5.1 Planeación

La actividad de planeación comienza creando una serie de historias (historias del usuario) que describen las características y la funcionalidad requeridas para el software que se construirá. Cada historia (similar a los casos de uso) la escribe el cliente y se coloca en una carta índice. El cliente le asigna un valor (una prioridad) a la historia basándose en los valores generales del negocio respecto de la característica o la función. El valor de una historia puede depender también de la presencia de otra historia.

Los miembros del equipo de la EP evalúan cada historia y le asignan un costo, el cual se mide en semanas de desarrollo. Si la historia requiere más de tres semanas de desarrollo, se le pide al cliente que la divida en historias menores, y se realiza de nuevo la asignación del valor y el costo. Es importante destacar que las historias nuevas pueden escribirse en cualquier momento.

Los clientes y el equipo de EP trabajan juntos para decidir cómo agrupar las historias hacia el próximo lanzamiento (el siguiente incremento de software) para que el equipo de la EP las desarrolle. Una vez establecido el compromiso básico para un lanzamiento, el equipo de la EP ordena las historias que se desarrollaran de una de las siguientes tres maneras: 1) todas las historias serán implementadas de modo inmediato (dentro de pocas semanas); 2) las historias con valor más alto se moverán en el programa y se implementarán al principio; o 3) las historias más riesgosas se moverán dentro del programa y se implementarán al principio(Pressman, 2004).

Después de que se ha entregado el primer lanzamiento del proyecto, el equipo de la EP calcula la velocidad del proyecto. La velocidad del proyecto es el número de historias de los clientes implementado durante el primer lanzamiento. La velocidad del proyecto puede utilizarse para 1)ayudar a estimar fechas de entrega y el programa para lanzamientos subsecuentes, y 2)determinar si se ha hecho un compromiso excesivo, el contenido de los lanzamientos se modifica o se cambian las fechas de las entregas finales.

Conforme avanza el trabajo de desarrollo, el cliente puede agregar historias, cambiar el valor de la historia existente, dividir historias o eliminarlas. El equipo de la EP considera de nuevo los lanzamientos restantes y modifica sus planes de acuerdo con ello.

2.5.2 Diseño

El diseño de la EP sigue de manera rigurosa el principio MS (mantenerlo simple). Siempre se prefiere un diseño simple respecto de una presentación más compleja. Además, el diseño ofrece una guía de implementación para una historia como está escrita, ni más ni menos. Se desaprueba el diseño de funcionalidad extra.

La EP apoya el uso de tarjetas CRC (Colaborador-Responsabilidad-Clase) como un mecanismo efectivo para pensar en el software en un contexto orientado a objetos. Las tarjetas CRC identifican y organizan las clases orientadas al objeto que son relevantes para el incremento del software actual. El equipo de EP conduce el ejercicio del diseño por medio de un análisis de requisitos, un modelado de datos, modelo basado en escenarios, modelo orientado al flujo, modelo basado en clases y el modelo de comportamiento. Sin embargo, las tarjetas CRC son el único producto de trabajo realizado como parte del proceso de la EP.

Si se encuentra un problema difícil de diseño como parte del diseño de la historia, la EP recomienda la creación inmediata de un prototipo operacional de esa porción del diseño. El prototipo del diseño, llamado la solución pico, se implementa y se evalúa. El propósito es reducir el riesgo cuando comience la verdadera implementación y validar las estimaciones originales en la historia que contiene el problema del diseño.

La EP apoya la refabricación, una técnica de construcción que también lo es de diseño. Fowler(Pressman, 2004) describe la refabricación de la siguiente manera:

> Refabricación es el proceso de cambiar un sistema de software de tal manera que no altere el comportamiento externo del código y que mejore la estructura interna. Es una manera disciplinada de limpiar el código [modificar/simplificar el diseño interno], lo que minimiza las oportunidades de introducir errores. En el refabricar, se mejora el diseño del código después de que se ha escrito.

El diseño se considera como un artefacto que puede y debe modificarse de manera continua a medida que prosigue la construcción. El propósito de refabricar es controlar estas modificaciones al sugerir pequeños cambios del diseño que “pueden mejorar de manera radical el diseño”. Refabricar significa que el diseño ocurre de manera continua a medida que se construye el sistema.

2.5.2.1 CRC

El modelado de Clase-Responsabilidad-Colaborador (CRC) proporciona un medio simple para identificar y organizar las clases relevantes para los requisitos del sistema o producto. Ambler(Pressman, 2004) describe el modelo CRC de la siguiente forma:

> Un modelo CRC en realidad es una colección de tarjetas índices estándar que representan clases. Las tarjetas se dividen en tres secciones. A lo largo del borde superior de la tarjeta se escribe el nombre de la clase. En el cuerpo de la tarjeta se listan las responsabilidades de la clase a la izquierda y los colaboradores a la derecha.

El modelo CRC puede utilizar tarjetas índices reales o virtuales. El objetivo es desarrollar una representación organizada de las clases. Las responsabilidades son los atributos y las operaciones relevantes para la clase. Dicho de manera más simple, una responsabilidad es "cualquier cosa que la clase sabe o hace."(Pressman, 2004)

Los colaboradores son aquellas clases que se requieren para que una clase reciba la información necesaria para completar una responsabilidad. Una colaboración implica ya sea una solicitud de información o la solicitud de alguna acción. A continuación se ilustra una tarjeta índice CRC:

Clase:	
Descripción:	
Responsabilidad	**Colaborador**

Tabla 1.1 Carta índice del modelo CRC.(Pressman, 2004)

Clase. Las clases se manifiestan en una de las siguientes formas: entidades externas, cosas, sucesos o eventos, roles, unidades organizacionales, sitios, estructuras(Pressman, 2004), clases de entidad, también llamadas clases de modelo o negocios, estas se extraen de manera directa del enunciado del problema, las clases de frontera que se utilizan para crear la interfaz y las clases de controlador que manejan una unidad de trabajo desde el inicio hasta el final.(Pressman, 2004)

Responsabilidades. Identifica responsabilidades (atributos y operaciones). Wirfs-Brock y sus colegas(Pressman, 2004) sugieren cinco directrices para determinar las responsabilidades de las clases:

1. La inteligencia del sistema se debe distribuir entre las clases para abordar de mejor manera las necesidades del problema.
2. Cada responsabilidad debe establecerse tan general como sea posible.
3. La información y el comportamiento relacionado con ella debe estar dentro de la misma clase.
4. La información relativa a una cosa debe localizarse con una sola clase, no distribuirse entre muchas de ellas.
5. Las responsabilidades pueden compartirse entre clases relacionadas cuando este es apropiado.

Colaboraciones. Wirfs-Brock y sus colegas definen las colaboraciones de la siguiente manera:

Las colaboraciones representan las solicitudes que un cliente hace a un servidor con el fin de cumplir una responsabilidad. Una colaboración es la materialización del contrato entre el cliente y el servidor. Se dice que un objeto colabora con otro objeto si, para cumplir con una responsabilidad, necesita enviar algunos mensajes al otro objeto. Una colaboración sencilla fluye en una dirección, lo que representa una solicitud del cliente al servidor. Desde el punto de vista del cliente, cada una de sus colaboraciones está asociada con una responsabilidad particular que ha implementado el servidor.

Las colaboraciones identifican las relaciones entre clases. Cuando un conjunto de clases colabora para lograr algún requisito, este puede organizarse en un subsistema (un aspecto de diseño).

Las colaboraciones se identifican al determinar si una clase puede cumplir cada responsabilidad por sí misma. Si no es así, entonces se requiere de la interacción con otra clase y, por ende, una colaboración.

2.5.3 Codificación

La EP recomienda que después de diseñar las historias y realizar el trabajo de diseño preliminar el equipo no debe moverse hacia la codificación, sino que debe desarrollar una serie de pruebas de unidad que ejerciten cada una de las historias que vayan a incluirse en el lanzamiento actual (incremento de software).

Una vez que el código esté completo, la unidad puede probarse de inmediato, y así proporcionar una retroalimentación instantánea a los desarrolladores.

Uno de los aspectos de la EP es la programación en pareja. La EP recomienda que dos personas trabajen juntas en cada estación de trabajo de computadora para crear el código de una historia. Esto proporciona un mecanismo para la resolución de problemas en tiempo real (dos cabezas piensan mejor que una) y el aseguramiento de la calidad en las mismas condiciones. También, alienta que los desarrolladores se mantengan centrados en el problema que se tiene a la mano.

La pareja de programadores es la responsable de la integración. Esta estrategia de "integración continua" ayuda a evitar problemas de compatibilidad e interfaz y proporciona un ambiente de "prueba de humo" que ayuda a identificar los errores desde el principio.

2.5.4 Pruebas

La creación de una prueba de unidad antes de comenzar la codificación es un elemento clave para el enfoque de la EP. Las pruebas de unidad que se crean deben implementarse con un marco de trabajo que permita automatizarlas. Esto apoya una estrategia de regresión de prueba cuando el código se modifica.

Cuando las unidades individuales de pruebas se organizan en un "conjunto universal de pruebas"(Pressman, 2004) se establece que: "Arreglar problemas pequeños cada pocas horas toma menos tiempo que arreglar problemas enormes justo antes de la fecha límite".

Las pruebas de aceptación de la *EP*, también llamadas pruebas del cliente, las especifica el cliente y se enfocan en las características generales y la funcionalidad del sistema, elementos visibles y revisables por el cliente. Las pruebas de aceptación se derivan de las historias del usuario que se han implementado como parte de un lanzamiento de software.

2.6 Integración *RUP* y *XP*

Se elige *Rational Unified Process* porque es un proceso de desarrollo de software completo, que a lo largo de sus fases tiene claramente definidas las actividades y los productos necesarios para la construcción del sistema. También, se va a emplear algunas características del proceso ágil *eXtreme Programming* (*XP*), ya que se va a ocupar el *framework* de *Grails* que utiliza *XP* para el desarrollo de software.

Entre *RUP* y *XP* existen las siguientes similitudes:

Rubros	***RUP***	***XP***
Requerimientos funcionales	Maneja las especificaciones de casos de uso.	Historias de usuario.
Clases	Diagrama de clases.	CRC: Clase-Responsabilidad-Colaborador.
Fases	Inicio, elaboración, construcción y transición.	Planeación, diseño, codificación, pruebas.
Iteraciones	Las fases las maneja en iteraciones.	Las agrupa en historias.

Tabla 1.2 Similitudes entre *RUP* y *XP.* (Elaboración propia)

A fin de evitar el re trabajo se tomaron las siguientes decisiones:

- Se elige emplear las especificaciones de casos de uso porque detallan requerimientos funcionales de manera completa a comparación de las historias de usuario de la *EP* que no permiten tal detalle porque son una descripción muy breve.
- *XP* emplea las tarjetas *CRC*, para representar clases relevantes para los requisitos del sistema, las cuales no se van a ocupar, ya que en *RUP* se tienen los diagramas de clases.
- Se descarta la actividad de XP de la programación en parejas, no es viable porque en este proyecto solo participa una persona.
- Se emplea *Grails* el cual es un *framework* para desarrollo web construido sobre una base sólida formada por proyectos como *Spring, Container, Hibernate*, entre otros.

2.7 *Grails*

Grails es un *framework* para desarrollo web que se parece mucho a *Rails* por fuera (incluso en el nombre, que originalmente era *Groovy on Rails* y tuvo que ser cambiado a petición de la gente de RoR(Brito, 2009)), pero que por dentro está construido sobre una base sólida formada por proyectos como *Spring container, Hibernate, SiteMesh, Log4J*, formado por *plugins* que permiten incorporar *Quartz, Compass/Lucene, JasperReports*.

Grails pretende ser un marco de trabajo altamente productivo siguiendo paradigmas tales como convención sobre configuración o no te repitas *DRY* (*Do not repeat yourself*), proporcionando un entorno de desarrollo estandarizado y ocultando gran parte de los detalles de configuración al programador. *Convention over Configuration* emplea convenciones para definir el tipo y las características de los artefactos, en lugar de hacerlo en archivos de configuración y emplea las virtudes de los lenguajes dinámicos para crear aplicaciones con menos código.

Grails ha sido impulsado principalmente por la empresa *G2One*(Wikipedia, Grails, 2012), la cual fue adquirida por la desarrolladora de software libre *SpringSource* en noviembre de 2008. En agosto de 2009 *SpringSource* fue a su vez adquirida por *VMWare*, empresa especializada en virtualización de sistemas.

Se inició en julio de 2005, con la versión 0.1 29 de marzo de 2006 y la versión 1.0 anunciada el 18 de febrero de 2008. En diciembre de 2009 se publicó la versión 1.2, y en mayo de 2010 la versión 1.3.

2.7.1 Características

- Ofrecer un *framework* web de alta productividad para la plataforma Java.
- Reutilizar tecnologías Java ya probadas como *Hibernate* y *Spring* bajo una interfaz simple y consistente.
- Ofrecer un *framework* consistente que reduzca la confusión y que sea fácil de aprender.
- Ofrecer documentación para las partes del *framework* relevantes para sus usuarios.

- Proporcionar lo que los usuarios necesitan en áreas que a menudo son complejas e inconsistentes:
    - *Framework* de persistencia potente y consistente.
    - Patrones de visualización potente y fácil de usar con *GSP* (*Groovy Server Pages).*
    - Librerías de etiquetas dinámicas para crear fácilmente componentes web.
    - Buen soporte de *Ajax* que sea fácil de extender y personalizar.
- Proporcionar aplicaciones ejemplo que muestren la potencia del *framework.*
- Proporcionar un entorno de desarrollo orientado a pruebas.
- Proporciona un entorno completo de desarrollo, incluyendo un servidor web y recarga automática de recursos.

Grails(Wikipedia, Grails, 2012) Se ha diseñado para ser fácil de aprender, fácil para desarrollar aplicaciones y extensible. Intenta ofrecer el equilibrio adecuado entre consistencia y funcionalidades potentes.

Grails tiene un servidor web integrado preparado para desplegar la aplicación desde el primer momento. Todas las bibliotecas requeridas son parte de la distribución de *Grails* y están preparadas para ser desplegadas automáticamente.

Grails soporta *scaffolding*[3] para implementar el patrón *CRUD* (*Create, Read, Update, Delete*).

2.7.2 Lenguaje Grovy

El uso de *Groovy* como punto de partida para el *framework* le proporciona una enorme ventaja inicial. El objetivo de *Groovy* era crear un lenguaje que permita a programadores Java una fácil transición al mundo de los lenguajes de script con tipado dinámico, proporcionando funcionalidades imposibles de implementar con lenguajes de tipado estático.

Groovy es un lenguaje dinámico desarrollado para Java, tiene la mejor integración con Java y probablemente la barrera más baja de entrada para los desarrolladores de Java.

[3] Ver Glosario

Realiza labores desde el más básico gestor de contenidos, hasta los sitios *MiddleWare* con servicios web *SOAP* y *REST*, pasando por plataformas *B2B* y sitios *WAP*(Klein, 2009).

A continuación se muestran las ventajas y desventajas de *Grails*:

Ventajas	**Desventajas**
Los lenguajes dinámicos, entre otras cosas, evitan la necesidad de especificar el tipo de dato de una variable en el momento de declararla en el código fuente, sino que el entorno de ejecución es capaz de determinarlo en tiempo de ejecución a partir de los datos que almacenemos en ella.	Su mayor ventaja es a la vez el origen de su mayor limitación: gran parte de lo que ocurre en un programa escrito en uno de estos lenguajes no figura por ninguna parte de su código fuente. Los tipos y funciones generados de forma dinámica no están descritos en ningún sitio, y por tanto los fallos que pueda contener son más difíciles de identificar y solucionar.
Permite definir clases en tiempo de ejecución para que los objetos se ajusten a un esquema que no puede conocerse de antemano, o añadir propiedades y métodos a un objeto para que haga más cosas de aquellas para las que fue inicialmente concebido.	Su misma naturaleza(Groovy, 2012) hace que sea más difícil escribir entornos de desarrollo para este tipo de lenguajes que incluyan ayudas al desarrollo como auto-completado o depuración.
Se necesita mucho menos código para realizar una determinada tarea que si la escribiésemos con un lenguaje estático. Su sintaxis suele ser mucho más expresiva y breve, lo cual redunda en una mayor productividad de los programadores y un código más fácil de comprender y mantener.	Se pierde una parte del control sobre lo que ocurre durante la ejecución del programa, lo que puede dificultar la resolución de incidencias en algunos tipos de aplicaciones.

Tabla 1.3 Ventajas y desventajas de *Grails.* (Elaboración propia)

CAPÍTULO TERCERO. DESARROLLO DEL SISTEMA

3.1 Objetivos

Desarrollar un sistema que administre los cuatro tipos de movimientos que realiza la "Subdirección de Planeación, Programación y Presupuesto de los Centros Públicos CONACyT", así como la realización del cálculo de cada una de las prestaciones que maneja para cada personal derivado del Formato Único de Movimientos de Servicios Personales de los Centros CONACYT, buscando mejorar la ejecución y cumplimiento de los procesos.

Este proyecto representa una oportunidad para mejorar la efectividad en aprovechamiento del tiempo y reducción de errores de captura durante los procesos a sistematizar.

3.2 Alcance

El sistema debe realizar los cálculos que se llevan a cabo en el área de la "Subdirección de Planeación, Programación y Presupuesto de los Centros Públicos CONACyT", utilizando el Formato Único de Movimientos de Servicios Personales para el cual se utilizan cuatro tipos de movimientos: incremento salarial, conversión por promoción, cancelación y creación de una plaza.

El sistema debe incluir el inicio de sesión de los usuarios, en la parte de usuario este calculará los movimientos básicos: incremento salarial, conversión por promoción, cancelación y creación de una plaza con la posibilidad de realizar un reporte al final de la operación. Por la parte de administrador se dará la posibilidad de manipular datos de crear puesto, modificar sueldo, crear nivel y código, administrar las prestaciones y modificar el sueldo de acuerdo al incremento salarial que cambia anualmente.

3.3 Lista de Stakeholders[4]

Nombre	Puesto	Tipo	Responsabilidad.
Federico Aradillas Ponce	Dirección de Planeación Programación y Ejecución de los Centros públicos CONACyT.	Patrocinador, usuario	• Autorizar los presupuestos para cada uno de los centros. • Recopilar información de los estados financieros. • Verificar la información programática que se genera en la Dirección de Planeación Programación y ejecución de los centros públicos CONACyT.
Fabiola Pérez Gómez	Subdirección de Ejecución Presupuestal de los Centros Públicos CONACyT.	Usuario	• Supervisar, ministrar recursos, coordinar y dar seguimiento a la información programática, presupuestal y organizacional de servicios personales de los Centros Públicos de Investigación CONACyT. • Verificar y proporcionar información programática y presupuestal así como de servicios personales a las áreas internas y externas (Contraloría Interna, Dirección Adjunta de Administración y Finanzas (DAAF), Secretaria de Hacienda y Crédito Público (SHCP), Secretaría de la Función Pública (SFP). • Supervisar y coordinar el proceso de promoción del personal científico y tecnológico de las Entidades Coordinadas por el CONACyT de acuerdo con la normatividad aplicable para dar respuesta a las solicitudes generadas por las Comisiones de Evaluación Internas y Externas.

[4] Recopilado del formato de perfiles de los empleados de la Subdirección de Ejecución Presupuestal de los Centros Públicos CONACyT.

Oscar Leonel Rodríguez Quiñones	Departamento en Materia de Remuneraciones de los Centros Públicos CONACyT.	Usuario	• Administrar y analizar la información programática, presupuestal así como de servicios personales de los Centros Públicos de Investigación CONACyT. • Analizar, verificar y ejecutar la ministración de recursos autorizados a las Entidades Coordinadas por el CONACYT. • Realizar estrategias y acciones de aplicación general en relación con la estructura orgánica, ocupacional, tabuladores de sueldos y prestaciones socioeconómicas, actualización de factores de prima de antigüedad, eventuales, honorarios, etc., establecidas por la Secretaría de Hacienda y Crédito Público así como la Secretaría de la Función Pública, con el fin de vigilar que se apeguen a la normatividad vigente.
Rosa Lilia Martínez Ramírez	Ministrador de los Centros Públicos de Investigación.	Usuario	• Capturar las ministraciones de recursos fiscales a los Centros Públicos de Investigación, mantener actualizados los montos presupuestales. • Analizar y consolidar información relativa a la calendarización financiera del presupuesto asignado a los Centros de Investigación con el fin de llevar el control de lo autorizado por la Secretaría de Hacienda y Crédito Público. • Ejecutar los movimientos presupuestales derivados de las transferencias, reducciones, movimientos compensados de recursos fiscales y propios que soliciten los Centros Públicos de Investigación.
Sandra Hernández Castro	Técnico especializado en materia de promociones.	Usuario	• Apoyar en el procedimiento de trámites para renivelación, creación y cancelación de plazas de los Centros Públicos de Investigación. • Preparar la información relativa a tabuladores de sueldos y prestaciones socioeconómicas, plantillas, eventuales, honorarios, etc., establecidas por la Secretaría de Hacienda y Crédito Público así como la Secretaría de la Función Pública, con el fin de aplicar la normatividad vigente.

			• Apoyar en el seguimiento a la entrega de los reportes de las Entidades Coordinadas por el CONACyT con el fin de dar cumplimiento a las fechas establecidas por las diferentes áreas normativas y vigilar que se apeguen a la normatividad vigente.
Luz María Morales Nava	Administrativo especializado.	Usuario	• Apoyar en el procedimiento de trámites para renivelación, creación y cancelación de plazas de los Centros Públicos de Investigación, con el objetivo de que sean autorizados y dictaminados por la SHCP, así como el envío oportuno de información impresa a diversas instancias. • Preparar la información relativa a los reportes de la Cuenta Pública. • Preparar la información relativa a tabuladores de sueldos y prestaciones socioeconómicas, plantillas, eventuales, honorarios. • Dar seguimiento a la entrega de los reportes de las Entidades Coordinadas por el CONACyT con el fin de dar cumplimiento a las fechas establecidas por las diferentes áreas normativas y vigilar que se apeguen a la normatividad vigente.
María del Pilar Alemán Hernández	Analista y Desarrollador de sistemas.	Equipo de proyecto	• Desarrollar la solución para mejorar el proceso actual derivado del Formato Único de Movimientos de Servicios Personales, y el proceso para su generación.

En la tabla siguiente se muestran marcados los productos de RUP, que se emplean para el desarrollo del sistema:

Inicio	Elaboración	Construcción	Transición
Documento de visión	Línea base de la arquitectura.	Producto de software.	Producto de software.
Modelo de casos de uso inicial (del 10% a 20%)	Modelo estático UML	El modelo de despliegue	Plan de soporte al usuario
Glosario del proyecto	Modelo dinámico UML	Conjunto de pruebas	Manual de usuario actualizado
Documento de evaluación del riesgo	Modelo de casos de uso	Manual de usuario	
Plan de proyecto inicial	Documento de visión actualizado	Descripción de las versiones	
Prototipos	Actualización de valoración del riesgo.	Plan de proyecto	
Documento inicial de la arquitectura	Actualización del modelo de negocio.		
Un modelo de negocio (caso de negocio)	Actualización del plan de proyecto		
	Firma del documento		

Tabla 1.4 Artefactos de RUP. (Elaboración propia)

3.4 EDT (Estructura para la División del Trabajo)

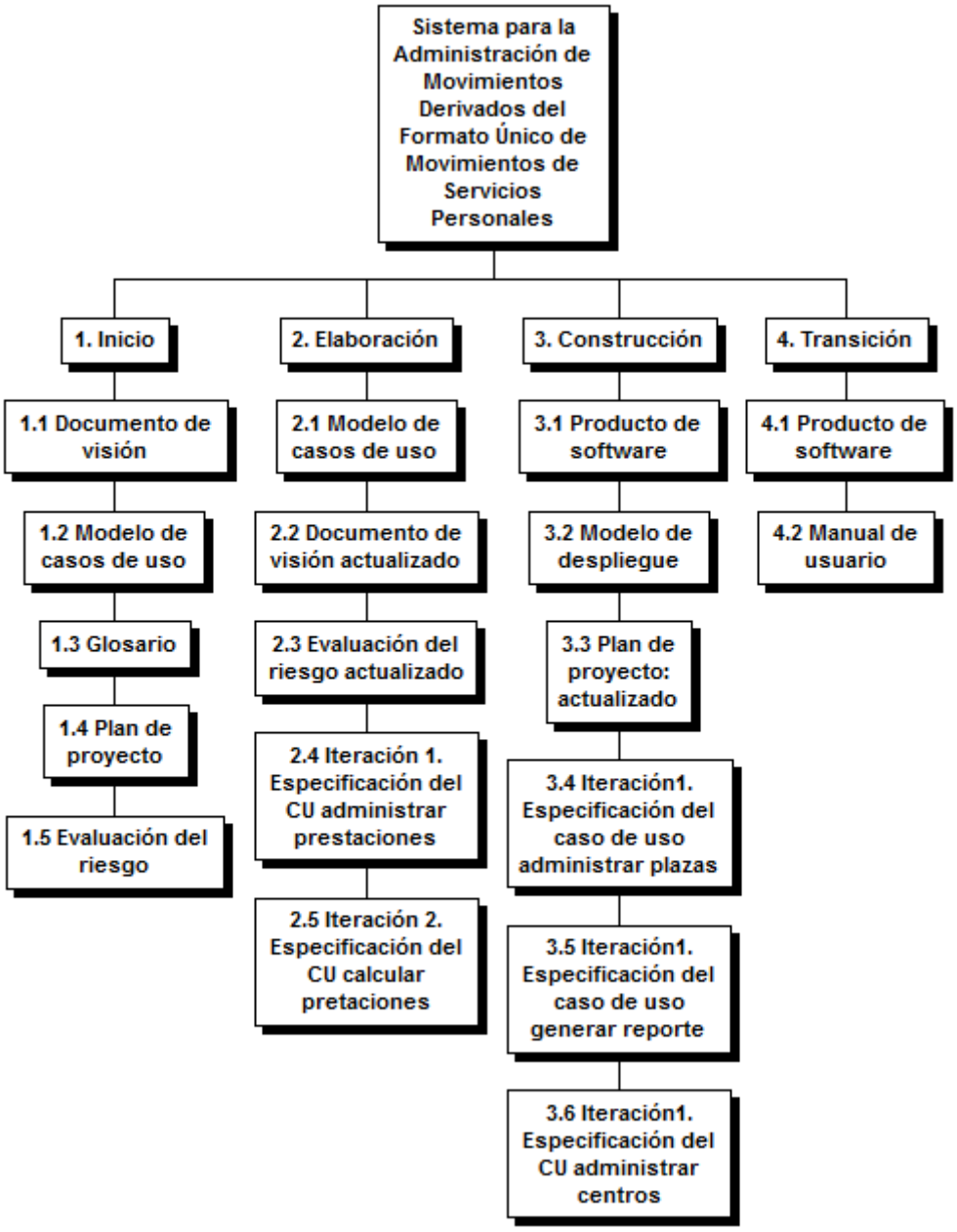


Tabla 1.5 EDT. (Elaboración propia)

3.5 Diagrama de Gantt

Id	Nombre de tarea	Duración	Comienzo
1	**Sistema para la Administración de Movimientos Derivados del Formato Único de Movimientos de Servicios**	0 días	lun 1/2/12
2	**1. Inicio**	0 días	lun 1/2/12
3	1.1 Documento de visión	15 días	lun 10/3/11
4	1.2 Modelo de casos de uso	3 días	lun 10/24/11
5	1.3 Glosario	3 días	jue 10/27/11
6	1.4 Plan de proyecto	5 días	mar 11/1/11
7	1.5 Evaluación del riesgo	5 días	mar 11/8/11
8	**2. Elaboración**	1 día	lun 11/14/11
9	2.1 Modelo de casos de uso	11 días	mié 11/16/11
10	2.2 Documento de visión actualizado	6 días	jue 12/1/11
11	2.3 Evaluación del riesgo actualizado	6 días	vie 12/9/11
12	2.4 Iteración 1. Especificación del CU administrar prestaciones	30 días	lun 12/19/11
13	2.6 Iteración 2. Especificación del CU calcular prestaciones	20 días	lun 1/30/12
14	**3. Construcción**	0 días	vie 2/24/12
15	3.1 Producto de software	60 días	vie 3/9/12
16	3.2 Modelo de despliegue	10 días	vie 6/1/12
17	3.3 Plan de proyecto: actualizado	10 días	vie 6/15/12
18	3.4 Iteración1. Especificación del caso de uso administrar plazas	20 días	vie 6/29/12
19	3.5 Iteración1. Especificación del caso de uso generar reporte	20 días	vie 7/27/12
20	3.6 Iteración1. Especificación del CU administrar centros	20 días	vie 8/24/12
21	**4. Transición**	1 día	jue 9/20/12
22	4.1 Producto de software	57 días	jue 9/20/12
23	4.2 Manual de usuario	3 días	lun 12/10/12

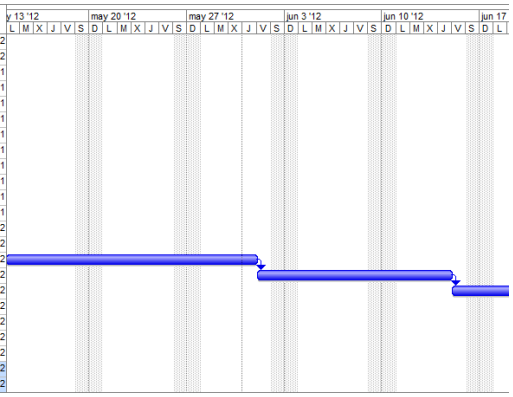


Tabla 1.6 Diagrama de Gantt. (Elaboración propia)

3.6 Milestones

Hitos de cada fase			
Inicio	**Elaboración**	**Construcción**	**Transición**
Hito: Objetivos del ciclo de vida.	Hito: Ciclo de vida de la arquitectura.	Hito: Capacidad operativa.	Hito Liberación del proyecto.
1 mes	3 meses	5 meses	2 meses

Tabla 1.7 Milestones. (Elaboración propia)

Categoría de riesgos

Un método para identificar riesgos(Pressman, 2004) consiste en crear una lista de verificación de riesgos. Con esta se pueden identificar riesgos y enfocarse en algún subconjunto de riesgos conocidos y predecibles en las siguientes subcategorías genéricas:

- Tamaño del producto: riesgos asociados con el tamaño global del software que se construirá o modificará.
- Impacto en el negocio: riesgos asociados con las restricciones que impone la gerencia o el mercado.
- Características del cliente: riesgos asociados con la sofisticación del cliente y la habilidad del desarrollador para comunicarse con él en una forma oportuna.

- Definición del proceso: riesgos asociados con el grado en el que se ha definido el proceso de software y en que le da seguimiento la organización que lo desarrolla.
- Entorno de desarrollo: riesgos asociados con la disponibilidad y calidad de las herramientas que se usaran en la construcción del producto.
- Tecnología que construir: riesgos asociados con la complejidad del sistema que se construirá y la "novedad" de la tecnología que está empaquetada en el sistema.
- Tamaño y experiencia de la plantilla de personal: riesgos asociados con la experiencia global técnica y en el proyecto de los ingenieros de software que harán el trabajo.

3.7 Lista de riesgos

De acuerdo a lo descrito anteriormente se determina la siguiente lista de riesgos de acuerdo a la clasificación mencionada:

Riesgos	Categoría	Probabilidad	Impacto
Los usuarios finales se resisten al sistema	Definición del Proceso **CO**	40%	3
La fecha límite de entrega está muy ajustada	**CO**	50%	2
La tecnología no cumple las expectativas	Tecnología que construir **RT**	30%	1
Falta de capacitación acerca del uso de las herramientas	Entorno de desarrollo **ED**	80%	3

Tabla 1.8 Lista de riesgos.(Pressman, 2004)

Valores de impacto:

1: catastrófico, 2: crítico, 3: marginal, 4: despreciable.

3.8 Estimación esfuerzo (tiempo-costo)

Se empleó la técnica de puntos de casos de uso para estimar el esfuerzo en tiempo y costo. A continuación se presentan los cálculos:

Factor de peso de los casos de uso

	Cantidad	Factor	UAW[5]	
Administrar usuarios	1	5	5	UUCP=UAW+UUCW
Administrar prestaciones	1	5	5	72
Administrar centros	1	5	5	
Administrar plazas	1	5	5	
Calcular prestaciones	1	15	15	
Administrar niveles	1	5	5	UCP=UUCP*TCF*EF
Administrar códigos	1	5	5	87.138
Administrar incremento salarial	1	5	5	
Administrar sueldos	1	5	5	
Generar reporte	1	5	5	Esfuerzo Horas-Humano
			60	E=UCP*Horas Humano
				4837.27

Suponiendo que al mes un programador cobre $9000, se divide entre 30 días el resultado son 300, se divide entre 8 horas dando como resultado 37.5. Al multiplicar el esfuerzo en horas 4837.27 por los 37.5 pesos la hora, da como resultado $181,386 pesos, el cual representa una parte del total del esfuerzo de todo el proyecto, generalmente el 40%. Este 40% se refiere al esfuerzo total para

[5] UAW: Factor de peso de los actores sin ajustar. UCP: Puntos de casos de uso ajustado. E: Esfuerzo Horas-Humano TCF: Factores técnicos. EF: Factores ambientales UUCP: Puntos de casos de uso sin ajustar. UUCW: Factor de peso de los casos de uso sin ajustar.

el desarrollo de las funcionalidades específicas en los casos de uso.

Factor de peso de los Actores sin ajustar

	Cantidad	Factor	UUCW
Subdirector	1	3	3
Jefe de departamento	1	3	3
Operativo	1	3	3
Director	1	3	3
			12

Factores de complejidad técnica

Factor	Descripción	Peso	Valor	Tfactor
T1	Sistema distribuido	2	1	2
T2	Objetivos de performance o tiempos de respuesta	1	5	5
T3	Eficiencia del usuario final	1	5	5
T4	Procesamiento interno complejo	1	3	3
T5	El código debe ser reutilizable	1	3	3
T6	Facilidad de instalación	0.5	3	1.5
T7	Facilidad de uso	0.5	5	2.5
T8	Portabilidad	2	3	6
T9	Facilidad de cambio	1	1	1
T10	Concurrencia	1	5	5
T11	Incluye objetivos especiales de seguridad	1	3	3
T12	Provee acceso directo a terceras partes	1	3	3
T13	Se requiere facilidad especiales de entrenamiento a usuario	1	3	3
				43
		TCF=	1.03	

Factores Ambientales

Factor	Descripción	Peso	Valor	Filtro
E1	Familiaridad con el modelo de proyecto utilizado	1.5	3	4.5
E2	Experiencia en la aplicación	0.5	1	0.5
E3	Experiencia en orientación a objetos	1	5	5
E4	Capacidad del analista líder	0.5	5	2.5
E5	Motivación	1	5	5
E6	Estabilidad de los requerimientos	2	0	0
E7	Personal part-time	-1	5	-5
E8	Dificultad del lenguaje de programación	-1	5	-5
				7.5
		EF=	1.115	

En la siguiente tabla se detallan la distribución en porcentaje, para el esfuerzo total en el desarrollo del proyecto:

Análisis	18139.7678
Diseño	36279.5355
Programación	72559.071
Pruebas	27209.6516
Despliegue	27209.6516

3.9 Visión del sistema

3.9.1 Objetivos

Desarrollar un sistema que administre los cuatro tipos de movimientos que realiza la "Subdirección de Planeación, Programación y Presupuesto de los Centros Públicos CONACyT", así como la realización del cálculo de cada una de las prestaciones que maneja para cada personal derivado del Formato Único de Movimientos de Servicios Personales de los Centros CONACYT, buscando mejorar la ejecución y cumplimiento de los procesos.

Este proyecto representa una oportunidad para mejorar la efectividad en aprovechamiento del tiempo y reducción de errores de captura durante los procesos a sistematizar.

3.9.2 Alcance

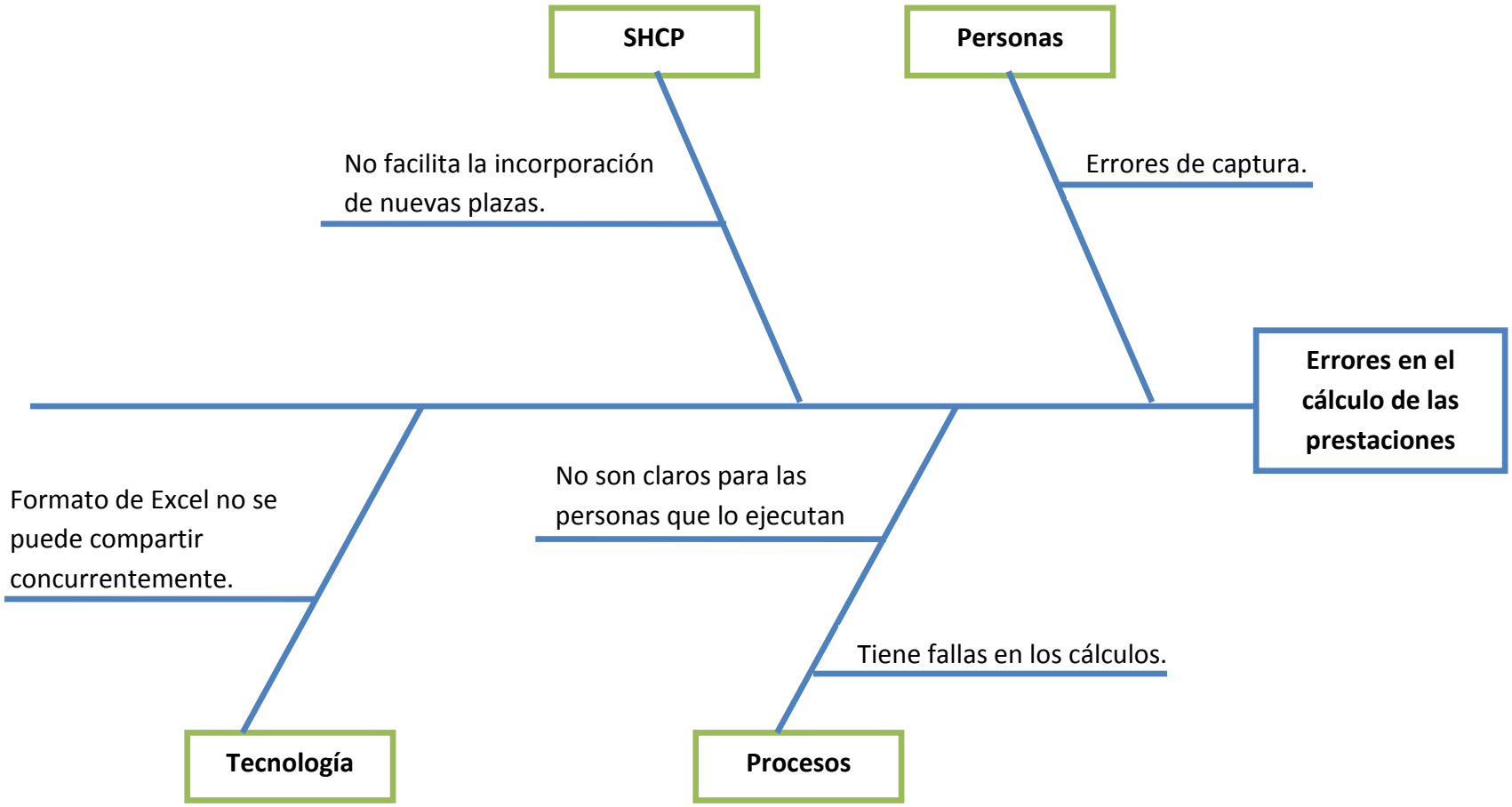
El sistema debe realizar los cálculos que se llevan a cabo en el área de la "Subdirección de Planeación, Programación y Presupuesto de los Centros Públicos CONACyT", utilizando el Formato Único de Movimientos de Servicios Personales para el cual se utilizan cuatro tipos de movimientos: incremento salarial, conversión por promoción, cancelación y creación de una plaza.

El sistema debe incluir el inicio de sesión de los usuarios, en la parte de usuario este calculará los movimientos básicos: incremento salarial, conversión por promoción, cancelación y creación de una plaza con la posibilidad de realizar un reporte al final de la operación. Por la parte de administrador se dará la posibilidad de manipular datos de crear puesto, modificar sueldo, crear nivel y código, administrar las prestaciones y modificar el sueldo de acuerdo al incremento salarial que cambia anualmente.

3.9.3 Descripción del problema

Problema	Los cálculos que se realizan utilizando el Formato Único de Movimientos de Servicios Personales no les permite a los usuarios obtener de manera ágil, rápida y sencilla la diferencia monetaria que existe entre los tipos de movimientos que realiza la "Subdirección de Ejecución Presupuestal de los Centros Públicos CONACyT". Todo se realiza de manera manual utilizando un formato en "Excel" de la paquetería de Office.
Afecta	A la "Dirección de Planeación Programación y Ejecución de los Centros públicos CONACyT", y a la "Subdirección de Ejecución Presupuestal de los Centros Públicos CONACyT", al "Departamento en Materia de Remuneraciones de los Centros Públicos CONACyT", al Ministrador de los Centros Públicos de Investigación, al Técnico especializado en materia de promociones y al Administrativo especializado.
Cuyo impacto es:	Tiempo: El tiempo que se invierte en ingresar cada una de las plazas y el sueldo implica un consumo excesivo de este recurso en la captura que se hace para cada centro. La información debe ser oportuna, ya que depende de otros centros a los que administra el CONACyT, para poder autorizar los montos que le corresponden a cada centro. Costo: Cada año la Secretaría de Hacienda y Crédito Público (SHCP) solicita que se suba a su sistema la conversión por promoción, el incremento salarial, la creación así como la cancelación de una plaza de los distintos centros a los que administra el CONACyT, para que el monto les sea autorizado, ya que la SHCP proporciona la diferencia para que se cubra ese monto, dependiendo del tipo de movimiento.
Una solución satisfactoria sería:	• Agilizar el proceso por medio del desarrollo de un sistema informático para que realice el cálculo del costo de las plazas en menos tiempo. • Es un sistema parametrizado o configurable ya que da la opción de cambiar los porcentajes, con permisos de usuarios, para cada una de las prestaciones que se requiera.

3.9.4 Diagrama causa-efecto (Ishikawa)



3.9.5 Descripción del producto

Para	"Subdirección de Ejecución Presupuestal de los Centros Públicos CONACyT"
¿Quién?	La "Subdirección de Ejecución Presupuestal de los Centros Públicos CONACyT" necesita un sistema que agilice el registro de los tipos de movimientos con los que cuenta esta dirección, valiéndose del tipo de personal y la vigencia para cada uno de los centros, el nivel, el puesto, la zona así como el tipo de personal. Que genere un reporte del resultado del registro de los tipos de movimientos realizados para cada uno de los centros a los que administra el CONACYT utilizando el Formato Único de Movimientos de Servicios Personales.
Nombre del producto	Sistema para la Administración de Movimientos Derivados del Formato Único de Movimientos de Servicios Personales.
Que	Va a permitir administrar los cuatro tipos de movimientos que realiza esta dirección para realizar el cálculo de las prestaciones de cada personal de los distintos centros a los que administra el CONACyT.
A diferencia de	A diferencia del proceso actual que se lleva acabo utilizando archivos en Excel. Este sistema administrará el proceso realizando los cálculos de las prestaciones para cada uno de los tipos de movimiento que maneja por cada centro.
El producto	Permitirá administrar los cuatro tipos de movimientos que maneja esta dirección y calculará las prestaciones correspondientes de cada uno de los centros a los que administra. Es un sistema parametrizado o configurable ya que da la opción de cambiar los porcentajes, con permisos de usuarios, para cada una de las prestaciones que se requiera.

3.9.6 Lista de usuarios

Usuario	Descripción	Responsabilidad.	Stakeholder.
Director	Responsable de la Dirección de Planeación, Programación y Presupuesto de los Centros Públicos CONACyT	Consultar las distintas plazas de los diferentes centros de los que está a cargo para generar reportes y poder autorizar los montos para cada centro.	Federico Aradillas Ponce.
Subdirector	Responsable de la Subdirección de Ejecución Presupuestal de los Centros Públicos CONACyT	Administrar el sistema y a la vez los usuarios del sistema. Así como de modificar las plazas.	Fabiola Pérez Gómez.
Jefe de Departamento	Responsable de la Jefatura de Departamento en Materia de Remuneraciones de los Centros Públicos CONACyT	Crear, cancelar realizar una conversión por promoción, aplicar un incremento salarial de una plaza por cada centro al que administra el CONACyT.	Oscar Leonel Rodríguez Quiñones.
Operativo	Apoyar en el procedimiento de trámites para renivelación, creación y cancelación de plazas de los Centros Públicos de Investigación	Realizar los distintos tipos de movimientos para todo el personal de cada uno de los centros a los que administra el CONACyT.	Rosa Lilia, Sandra, Luz María.

3.10 Lista de requerimientos

Clave	Fecha de levantamiento	Quien solicita	Descripción
FUMSP-01	16-Febrero-2011	Fabiola Pérez Gómez	El sistema debe permitir al subdirector elegir en un catálogo el centro, el tipo de personal y el tipo de movimiento que quiere realizar. Debe ingresar el número de plazas y el sueldo.
FUMSP-02	15-Marzo-2011	Oscar Rodríguez	El sistema debe calcular las prestaciones una vez que se haya elegido el tipo de movimiento y el tipo de personal que se quiere realizar con las plazas que le corresponden a ese centro.
FUMSP-03	23-Marzo-2011	Fabiola Pérez Gómez	El sistema debe generar un reporte con las características del Formato Único de Movimientos de Servicios Personales. (Anexo 1).
FUMSP- 04	10-Junio-2011	Fabiola Pérez Gómez	El sistema debe permitir al subdirector crear un puesto con sus respectivos sueldos.
FUMSP-05	10-Junio-2011	Fabiola Pérez	El sistema debe permitir realizar consultas de los tipos de movimientos efectuados para cada uno de los centros con su respectivo tipo de personal.
FUMSP-06	10-enero-2012	Pilar Alemán	El sistema debe tener un control de acceso o inicio de sesión solo para el personal que le compete.
FUMSP-07	10-enero-2012	Pilar Alemán	El sistema debe tener ciertos colores que favorezcan al ambiente y de acuerdo a los colores de la institución: azul y blanco.
FUMSP-08	10-enero-2012	Pilar Alemán	Las características del sistema deben funcionar 24X7.
FUMSP-09	10-enero-2012	Pilar Alemán	El sistema debe trabajar en un ambiente Web para que los usuarios puedan trabajar en forma remota, y también porque los equipos del personal no tienen los recursos necesarios para que sea una aplicación local.
FUMSP-10	15-Agosto-2011	Fabiola	En la prestación de ajuste calendario el sistema debe considerar los días 29 del año

		Pérez	bisiesto.
FUMSP-11	15-Agosto-2011	Fabiola Pérez Gómez	En la prestación de días económicos el sistema debe identificar el tipo de puesto para determinar los días que faltó el personal.
FUMSP-12	15-Agosto-2011	Fabiola Pérez Gómez	Al calcular la CESANTIA el sistema debe basarse en la antigüedad que tenga cada tipo de personal que cotiza al ISSSTE.
FUMSP-13	18-Agosto-2011	Fabiola Pérez Gómez	Para la prestación de material didáctico el sistema debe validar que solo se apliquen los cálculos para el tipo de personal de Científicos y Tecnológicos.
FUMSP-14	18-Agosto-2011	Fabiola Pérez Gómez	El sistema debe validar que centros cotizan al ISSSTE y cuales cotizan al IMSS para realizar sus respectivos cálculos de las prestaciones, si es del IMSS este no debe calcular la prestación de CESANTIA.
FUMSP-15	18-Agosto-2011	Fabiola Pérez	El sistema debe permitir al subdirector elegir el tipo de movimiento de conversión por promoción y realizar la re-nivelación calculando de acuerdo a la antigüedad del personal de científicos y tecnológicos.
FUMSP-16	18-Agosto-2011	Fabiola Pérez	El sistema debe realizar la diferencia de la promoción o re-nivelación entre la cancelación y creación del personal de científicos y tecnológicos, esa diferencia es la que se lleva cada investigador al subir de nivel.
FUMSP-17	18-Agosto-2011	Fabiola Pérez	El sistema debe calcular la prestación de compensación garantizada solo para el personal de mando.
FUMSP-18	18-Agosto-2011	Fabiola Pérez	El sistema debe clasificar mediante siglas CF (Plazas de confianza) que solo son para el personal de mando y CB (Plazas base) para todas las demás plazas. Las de confianza pertenecen al apartado A y las de base al apartado B. Estos aparatados se toman de los tabuladores.
FUMSP-19	15-Agosto-2011	Fabiola Pérez	El sistema debe clasificar mediante la vigencia de cada centro, conforme si la vigencia es de enero o es de febrero.
FUMSP-20	15-Agosto-2011	Fabiola	El sistema debe determinar el tipo de CESANTIA que tiene cada centro.

		Pérez Gómez	
FUMSP-21	15-Agosto-2011	Fabiola Pérez Gómez	El sistema debe permitir al subdirector elegir las categorías que para el personal de científicos y tecnológicos es fija (ITA, ITB, ITC).
FUMSP-22	18-Agosto-2011	Fabiola Pérez Gómez	El sistema debe realizar el cálculo anual de la creación de plazas.
FUMSP-23	15-Agosto-2011	Fabiola Pérez Gómez	El sistema debe permitir al subdirector crear un puesto cuando no exista en el centro con su respectivo código.
FUMSP-24	15-Agosto-2011	Fabiola Pérez Gómez	El sistema debe calcular las prestaciones del personal de mando tomando en cuenta que aparte de ser investigador se le asignan cargas o funciones administrativas por lo que tiene una remuneración adicional.
FUMSP-25	22-Enero-2012	Pilar Alemán	El sistema debe ser parametrizado para adecuar los cambios en los porcentajes de las prestaciones y los permisos de usuario para poder actualizar los porcentajes de las prestaciones que requiera.

En la siguiente tabla se muestran los cálculos para cada una de las prestaciones:

Prestación	Formula
Antigüedad.	Sueldo * 27.65%
Prima vacacional.	Sueldo / 330 días del año * 24 días * 1.15
Aguinaldo	Sueldo / 330 días * 20 días * 1.15
Fondo de ahorro	Sueldo * 13%
Compensación garantizada	sueldo unitario + compensación*12 meses
Ajuste calendario	Sueldo /330 días * 5 días
Días económicos	Sueldo / 330 días* 4.3
Estimación puntualidad y asistencia	Sueldo / 330 días*1.7
ISSSTE	Sueldo * 9.97% + 247.09 * 11 meses * total plazas
FOVISSSTE	Sueldo * 5%
SVI	Sueldo * 1.8%
SAR	Sueldo * 2.43%
CESANTIA	Sueldo * 3.175% + 97.77 * total plazas *11 meses
Material didáctico	Número plazas * sueldo * meses (11 o 12)
IMSS	Sueldo + prima vacacional + aguinaldo +ajuste calendario + antigüedad * 17%[6]
INFONAVIT	Sueldo + prima vacacional + aguinaldo + ajuste calendario + antigüedad * 5%

Tabla 1.9 Cálculos de cada prestación (Elaboración propia).

[6] Algunos números que se muestran en la tabla en algunos casos es dinero, en otros son días y salen de las repercusiones de seguridad social que brinda tanto el ISSSTE como las prestaciones que son autorizadas por la SHCP.

Comprobación de prestaciones	
ISSSTE 9.970	Si sueldo >15777,52.59*30*10*9.97%+247.09*total plazas)*periodo (11 meses) * sueldo *9.97% + 247.09 * total plazas * periodo (11 meses)
FOVISSSTE 5.0	Si sueldo > 15777, 52.59 * 30 * 10 * 0.05 * total plazas * periodo colectivo, (11 meses, sueldo * 0.05 * total plazas * periodo (11 meses)
Comprobación del ISSSTE y FOVISSSTE	52.59 * 30 * 10 * 0.0997 + 247.09 * total plazas * periodo (11 meses)
CESANTIA 3.175	Si sueldo > 15777, 52.59 * 30 * 10 * 0.03175 + 97.77 *total plazas * periodo (11 meses, sueldo * 0.03175 + 97.77 * total plazas * periodo (11 meses)
SAR 2.0	Si sueldo > 15777, 52.59 * 30 * 10 * 0.02 * total plazas *$ periodo (11 meses), sueldo * 0.02 * total plazas * periodo (11 meses)

Tabla 1.10 Comprobación de cálculo de prestaciones (Elaboración propia).

3.11 Matriz de trazabilidad

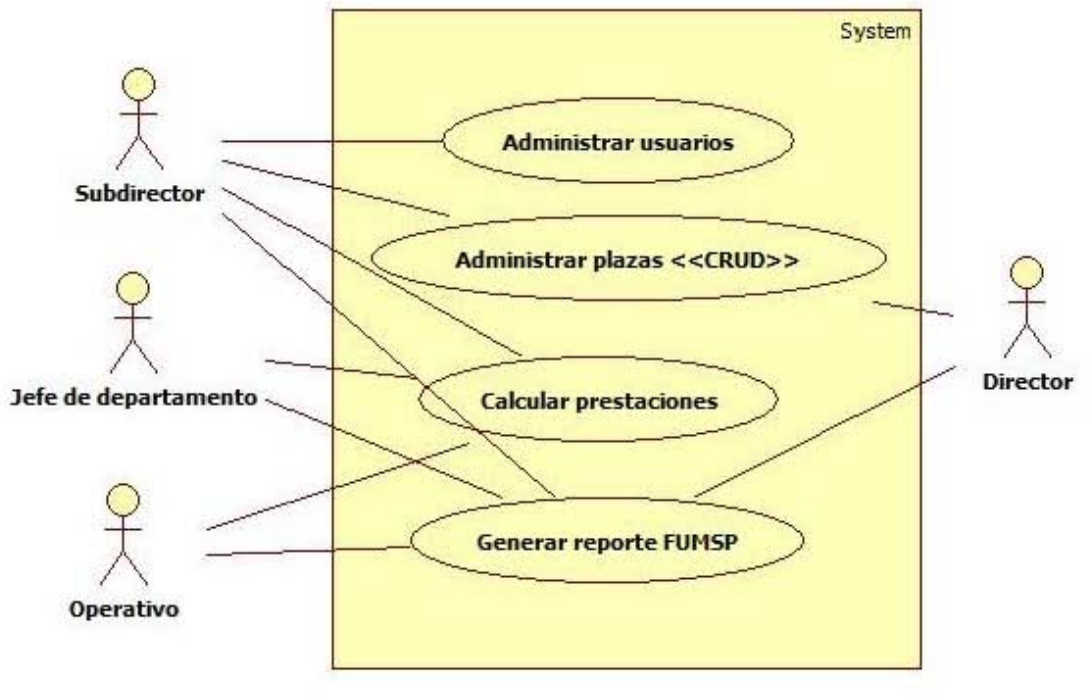
En la siguiente tabla la matriz de trazabilidad es en donde se confrontan los requerimientos y los casos de uso.

Caso de uso / Requerimiento	Calcular prestaciones	Administrar usuarios	Administrar prestaciones	Administrar plazas	Administrar niveles	Administrar códigos	Administrar incremento salarial	Administrar sueldos	Generar reporte	Administrar centros
FUMSP-01								✓		
FUMSP-02	✓									
FUMSP-03									✓	✓
FUMSP-04				✓				✓		
FUMSP-05										✓
FUMSP-06		✓								
FUMSP-07										
FUMSP-08		✓								
FUMSP-09										
FUMSP-10	✓		✓							
FUMSP-11	✓		✓							
FUMSP-12	✓									
FUMSP-13	✓									
FUMSP-14	✓		✓							

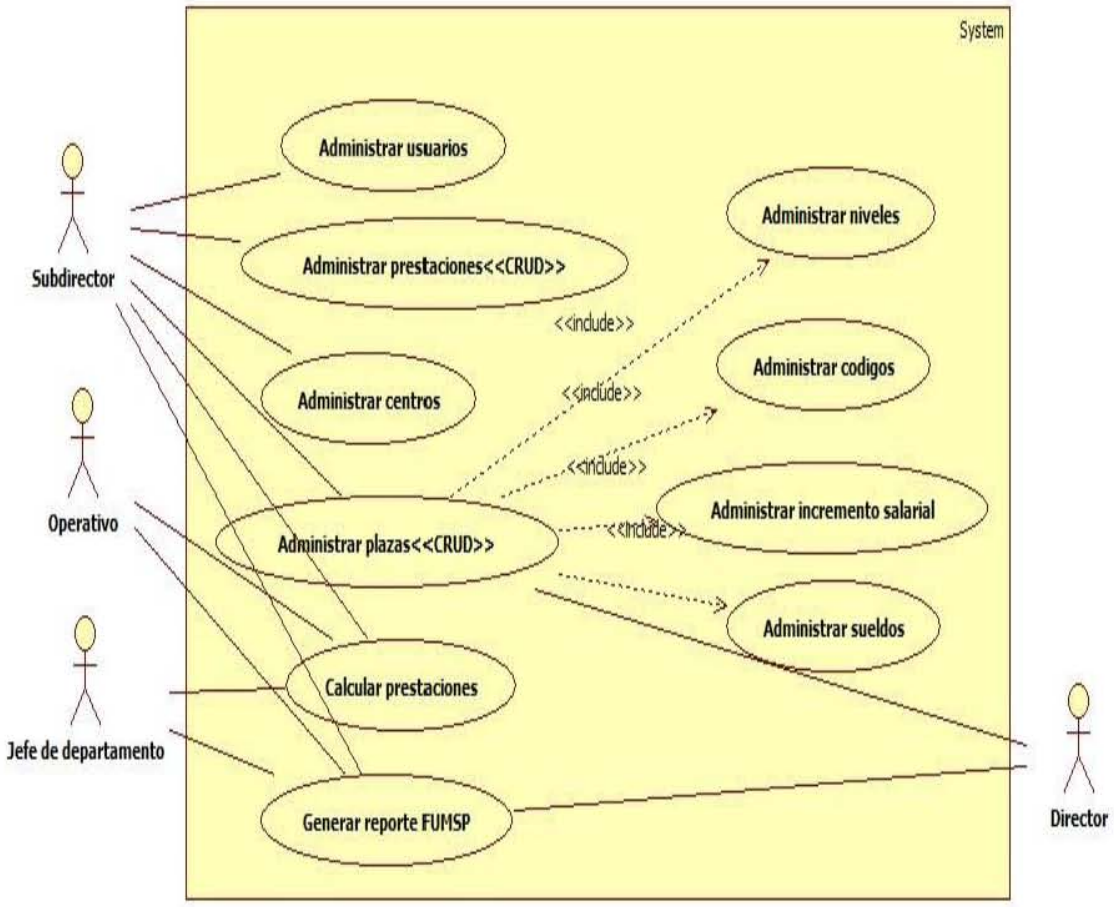
FUMSP-15	✓		✓							
FUMSP-16	✓		✓							
FUMSP-17	✓		✓							
FUMSP-18			✓							
FUMSP-19			✓							
FUMSP-20			✓							
FUMSP-21			✓							
FUMSP-22	✓									
FUMSP-23				✓	✓	✓	✓	✓		
FUMSP-24										
FUMSP-25	✓									

Tabla 1.11 Matriz de trazabilidad. (Elaboración propia)

3.12 Modelo de casos de uso calcular prestaciones



Modelo de casos de uso calcular prestaciones después de análisis de especificaciones de casos de uso.



Caso de Uso: Calcular prestaciones

1. Breve descripción

Este caso de uso se encarga de calcular las prestaciones con las que cuenta cada empleado del CONACyT.

2. Breve descripción del actor

1. Subdirector: Se encarga de ingresar, modificar cada una de las prestaciones en el sistema.
2. Director: Se encarga de realizar consultas de las prestaciones ingresadas para poder autorizar los montos para cada centro.

3. Precondiciones

El Subdirector debe ingresar los montos para cada tipo de empleado según el centro que elija.

4. Flujo básico

El caso de uso comienza cuando el Subdirector ingresa al sistema.

1. El sistema muestra el formulario de "consulta de centros", en donde se muestra la lista de centros, la lista del tipo de personal, la lista de plazas y la lista del tipo de movimiento, una opción de aceptar y una de cancelar.
2. El Subdirector selecciona el tipo de búsqueda que desea generar.
3. El sistema muestra el formulario de acuerdo al tipo de movimiento que haya elegido:
   - 3.1 Creación
   - 3.2 Cancelación
   - 3.3 Conversión por promoción
   - 3.4 Incremento salarial

   FLUJO DE CREACIÓN DE UNA PLAZA

   3.1.1 El sistema realiza la consulta a los datos de plazas y periodo insertados en el formulario de "cancelación de plaza sección 1" y realiza el cálculo de multiplicar las plazas seleccionadas, por el sueldo por 1.039 por la diferencia en meses de las fechas del período.

3.1.2 El sistema muestra el formulario denominado “creación de una plaza”, donde esta una tabla de “creación de plaza sección 1” con los datos de: nombre del centro, tipo de movimiento, tipo de personal, vigencia, plazas, período, número de meses, y sueldo colectivo por periodo. Con una opción de calcular prestaciones.

3.1.3 Calcular prestaciones

3.1.3.1 El sistema realiza una consulta para verificar las prestaciones que le corresponden al tipo de personal de la plaza creada.

3.1.3.2 El sistema realiza el cálculo de la antigüedad: multiplica el sueldo por el porcentaje de antigüedad (27.65%).

3.1.3.3 El sistema realiza el cálculo de la prima vacacional: divide el sueldo colectivo por periodo entre 330 días, el resultado lo multiplica por 24 días, el resultado lo multiplica por 1.15.

3.1.3.4 El sistema realiza el cálculo del aguinaldo dividiendo el sueldo colectivo por periodo entre 330 días, el resultado multiplicarlo por 20, y multiplicarlo por 1.15.

3.1.3.5 El sistema realiza el cálculo de fondo de ahorro: multiplicando el sueldo por 13%.

3.1.3.6 El sistema realiza el cálculo de ajuste calendario: multiplicando el sueldo y el resultado dividirlo entre 330 días y el resultado multiplicarlo por 5 días.

3.1.3.7 El sistema realiza el cálculo de días económicos: divide el sueldo entre 330 días, y el resultado lo multiplica por 4.3.

3.1.3.8 El sistema realiza el cálculo de la estimación puntualidad y asistencia: dividiendo el sueldo entre 330 días y el resultado lo multiplica por 1.7.

3.1.3.9 El sistema realiza el cálculo del ISSSTE: multiplicando el sueldo por 9.97%, al resultado le suma 247.09, al resultado

lo multiplica por 11 meses, al resultado lo multiplica por el total de plazas.

3.1.3.10 El sistema realiza el cálculo del FOVISSSTE: multiplicando el sueldo por 5%.

3.1.3.11 El sistema realiza el cálculo del SVI: multiplicando el sueldo por 1.8%.

3.1.3.12 El sistema realiza el cálculo de SAR: multiplicando el sueldo por 2.43.

3.1.3.13 El sistema realiza el cálculo de CESANTIA: multiplicando el sueldo por 3.175%, al resultado le suma 97.77, al resultado de la suma lo multiplica por el total de plazas, y lo multiplica por 11 meses.

3.1.3.14 El sistema realiza el cálculo de material didáctico: multiplicando el número de plazas por el sueldo, y por el número de meses, de acuerdo al centro.

3.1.3.15 El sistema realiza el cálculo del IMSS: sumando el sueldo, más la prima vacacional, más el aguinaldo, mas el ajuste calendario, más la antigüedad, y al resultado lo multiplica por 17%.

3.1.3.16 El sistema realiza el cálculo del INFONAVIT: sumando el sueldo, mas la prima vacacional, mas el aguinaldo, mas el ajuste calendario, mas la antigüedad y el resultado lo multiplica por 5%.

3.1.3.17 El sistema realiza la suma total de los resultados de los cálculos de las prestaciones.

3.1.3.18 El sistema muestra el formulario de “creación” con dos tablas, una de cancelación y otra de creación, con los datos de número de operación (id), la clave de prestación, el importe colectivo de cada una de las prestaciones, así como la suma total del importe colectivo, y realiza la diferencia entre el importe total de la tabla cancelación y el importe total de la tabla de creación, con opción de nueva consulta y generar reporte.

3.1.4 Nueva consulta.

3.1.4.1 El subdirector elige la opción de nueva consulta.

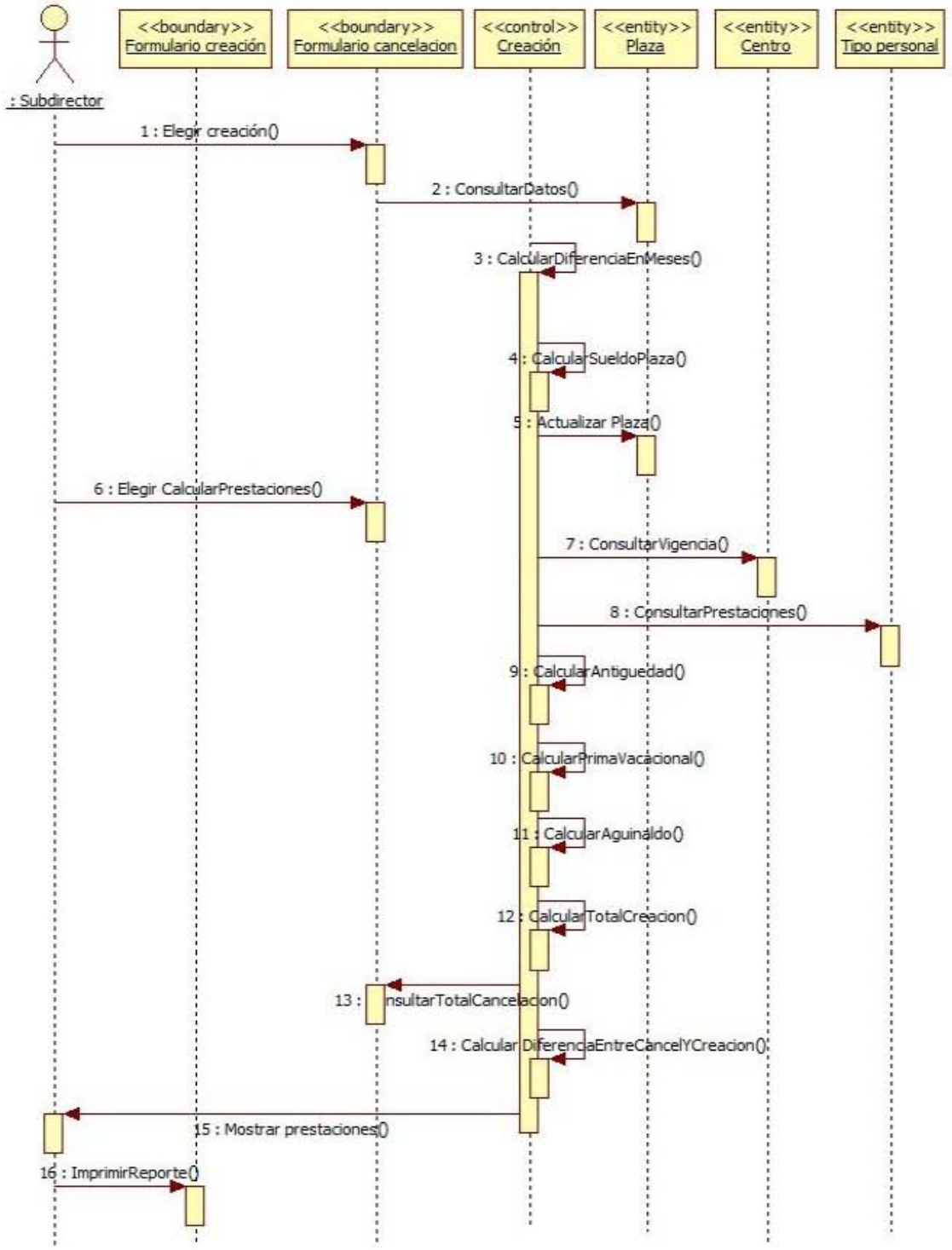
3.1.4.2 El caso de uso inicia de nuevo.

3.1.5 Reporte

3.1.5.1 El subdirector elige la opción de generar reporte.

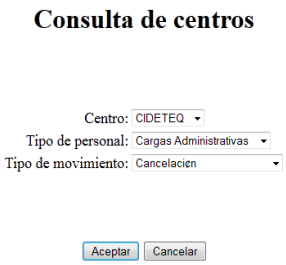
3.1.5.2 El sistema muestra el formato único de movimientos de servicios personales con los datos de los cálculos de las prestaciones realizadas. Con los datos de el nombre de la dependencia, el nombre del centro, el tipo de movimiento, el tipo de personal, la vigencia, el nivel, la zona, el código, el puesto, el total de plazas, el sueldo, el sueldo colectivo por período, la clave de la prestación, el nombre de la prestación, el importe colectivo y el total de todo el FUMSP. Y una opción de imprimir reporte.

Diagrama de secuencia: creación de una plaza.



Iniciar sesión

Formulario consulta de centros



FLUJO DE CANCELACIÓN DE UNA PLAZA

3.2.1 El sistema muestra el formulario denominado “cancelación de plaza” donde están las listas de las plazas, fechas del período, sueldo y una opción de calcular.

3.2.2 El subdirector selecciona las plazas, las fechas del período, ingresa el sueldo y selecciona la opción de calcular.

3.2.3 El sistema realiza la diferencia en meses de las fechas del periodo.

3.2.4 El sistema realiza la operación de multiplicar el número de plazas por el sueldo, por la diferencia en meses del cálculo anterior.

3.2.5 El sistema actualiza la plaza con los datos proporcionados y los cálculos realizados.

3.2.6 El sistema realiza una consulta para verificar la vigencia en la que cotiza el centro seleccionado.

3.2.7 El sistema muestra la tabla de “cancelación de plaza sección 1” con los datos de: nombre del centro, tipo de movimiento, tipo de personal y vigencia.

3.2.8 El sistema muestra la tabla de “cancelación de plaza sección 1” con los datos de las plazas canceladas en una tabla: nivel, zona, código, puesto, total de plazas, sueldo, y sueldo colectivo por periodo.

3.2.8.1 El subdirector selecciona en las listas del formulario “consulta de centros” el tipo de personal de mando o directivo.

3.2.8.2 Se inicia el caso de uso con la excepción de que introduce un dato más, la compensación garantizada, en el formulario de “cancelar plazas mando”.

3.2.8.3 El sistema realiza la diferencia en meses de las fechas del periodo.

3.2.8.4 El sistema realiza la operación de multiplicar el número de plazas por el sueldo, por la diferencia en meses

del cálculo anterior y le suma lo agregado en la compensación garantizada.

3.2.8.5 El sistema actualiza la plaza con los datos proporcionados y los cálculos realizados.

3.2.8.6 El sistema muestra la tabla de "cancelación de plazas mando sección 1" con los datos de: nombre del centro, tipo de movimiento, tipo de personal y vigencia.

3.2.8.7 El sistema muestra la tabla de "cancelación de plaza sección 1" con los datos de las plazas canceladas en una tabla: nivel, zona, código, puesto, total de plazas, sueldo, sueldo colectivo por periodo, compensación garantizada, compensación garantizada por periodo y el total.

3.2.8.8 El caso de uso continúa.

3.2.9 El sistema realiza una consulta para verificar las prestaciones que le corresponden al tipo de personal de la plaza cancelada.

3.2.10 El sistema realiza el cálculo de la antigüedad: multiplica el sueldo colectivo por periodo ingresado por el porcentaje de antigüedad (27.65%).

3.2.11 El sistema realiza el cálculo de la prima vacacional: divide el sueldo colectivo por periodo entre 330 días, el resultado lo multiplica por 24 días, el resultado lo multiplica por 1.15.

3.2.12 El sistema realiza el cálculo del aguinaldo dividiendo el sueldo colectivo por periodo entre 330 días, el resultado multiplicarlo por 20, y multiplicarlo por 1.15.

3.2.13 El sistema realiza la suma total de los resultados de los cálculos de las prestaciones.

3.2.14 El sistema muestra la tabla de "calcular prestaciones", con los datos de la clave de prestación, el nombre de la prestación, el importe colectivo de cada una de las prestaciones, así como la suma total del importe colectivo, con una opción de nueva consulta y generar reporte.

3.2.14.1 Nueva Consulta

3.2.14.1.1 El subdirector elige la opción de nueva consulta.

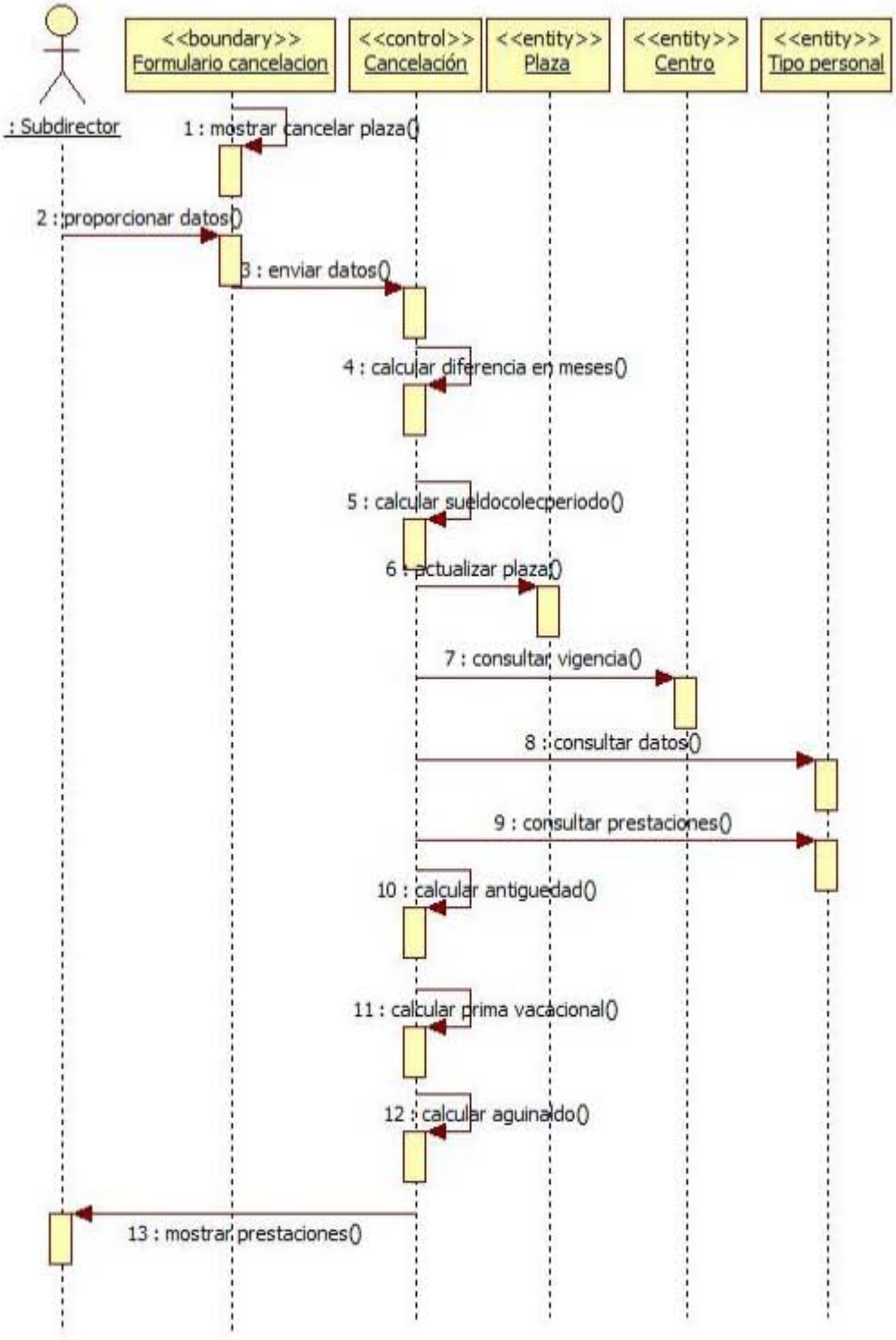
3.2.14.1.2 El caso de uso inicia de nuevo.

3.2.15.2 Reporte

3.2.15.2.1 El subdirector elige la opción de generar reporte.

3.2.15.2.2 El sistema muestra el formato único de movimientos de servicios personales con los datos de los cálculos de las prestaciones realizadas. Con los datos de el nombre de la dependencia, el nombre del centro, el tipo de movimiento, el tipo de personal, la vigencia, el nivel, la zona, el código, el puesto, el total de plazas, el sueldo, el sueldo colectivo por período, la clave de la prestación, el nombre de la prestación, el importe colectivo y el total de todo el FUMSP. Y una opción de imprimir reporte.

Diagrama de secuencia: cancelación de una plaza.



FLUJO DE CONVERSIÓN POR PROMOCIÓN

3.3.1 El subdirector elige en la lista del formulario "consulta de centros" la opción de conversión por promoción y en la lista de tipo de personal solo está la opción de Científicos y tecnológicos. Y muestra la lista de puestos de ese tipo de personal.

3.3.2 El subdirector elige el tipo de puesto.

3.3.3 El sistema muestra el formulario de "Conversión por promoción[7]".

3.3.4 El sistema realiza una consulta para verificar la vigencia en la que cotiza el centro seleccionado.

3.3.5 El sistema muestra el formulario de "conversión por promoción" con los datos de: nombre del centro, tipo de movimiento, tipo de personal y vigencia.

3.3.6 En donde despliega la tabla de "conversión por promoción cancelación" con los datos de puesto, nivel, zona, código, número de plazas y también una tabla denominada "conversión por promoción creación" con las listas de puesto, nivel, zona, código y una opción de agregar.

3.3.7 El subdirector ingresa las plazas, selecciona la lista de puesto, nivel, zona, código y selecciona la opción de agregar. Estas listas dependen una de la otra.

3.3.8 El sistema muestra dos tablas, denominada cancelación y creación. La tabla de cancelación y creación muestran los datos de puesto, nivel, zona, código, plazas, total de plazas, y una opción de calcular prestaciones y nueva consulta.

3.3.8.1 Calcular prestaciones

3.3.9 El sistema realiza el cálculo de la antigüedad: multiplica el sueldo colectivo por periodo ingresado por el porcentaje de antigüedad (27.65%).

3.3.10 El sistema realiza el cálculo de la prima vacacional: divide el sueldo colectivo por periodo entre 330 días, el resultado lo multiplica por 24 días, el resultado lo multiplica por 1.15.

3.3.11 El sistema realiza el cálculo del aguinaldo dividiendo el sueldo colectivo por periodo entre 330 días, el resultado multiplicarlo por 20, y multiplicarlo por 1.15.

3.3.12 El sistema realiza el cálculo de la prestación de material didáctico, ya que solo esta prestación se calcula para este

[7] El tipo de movimiento conversión por promoción solo se utiliza para el tipo de personal de Científicos y Tecnológicos.

tipo de personal, multiplica la plaza por el sueldo por los meses.

3.3.13 El sistema realiza la suma total de los resultados de los cálculos de las prestaciones.

3.3.14 El sistema muestra los datos de la cancelación y la creación en dos tablas, con los datos del número de operación (un id), la clave de la prestación, el importe, el subtotal, el total, y la diferencia de la renivelación de esa plaza. Y una opción de nueva consulta y generar reporte.

3.3.14.1 Nueva consulta

3.3.14.1.1 El subdirector elige la opción de nueva consulta.

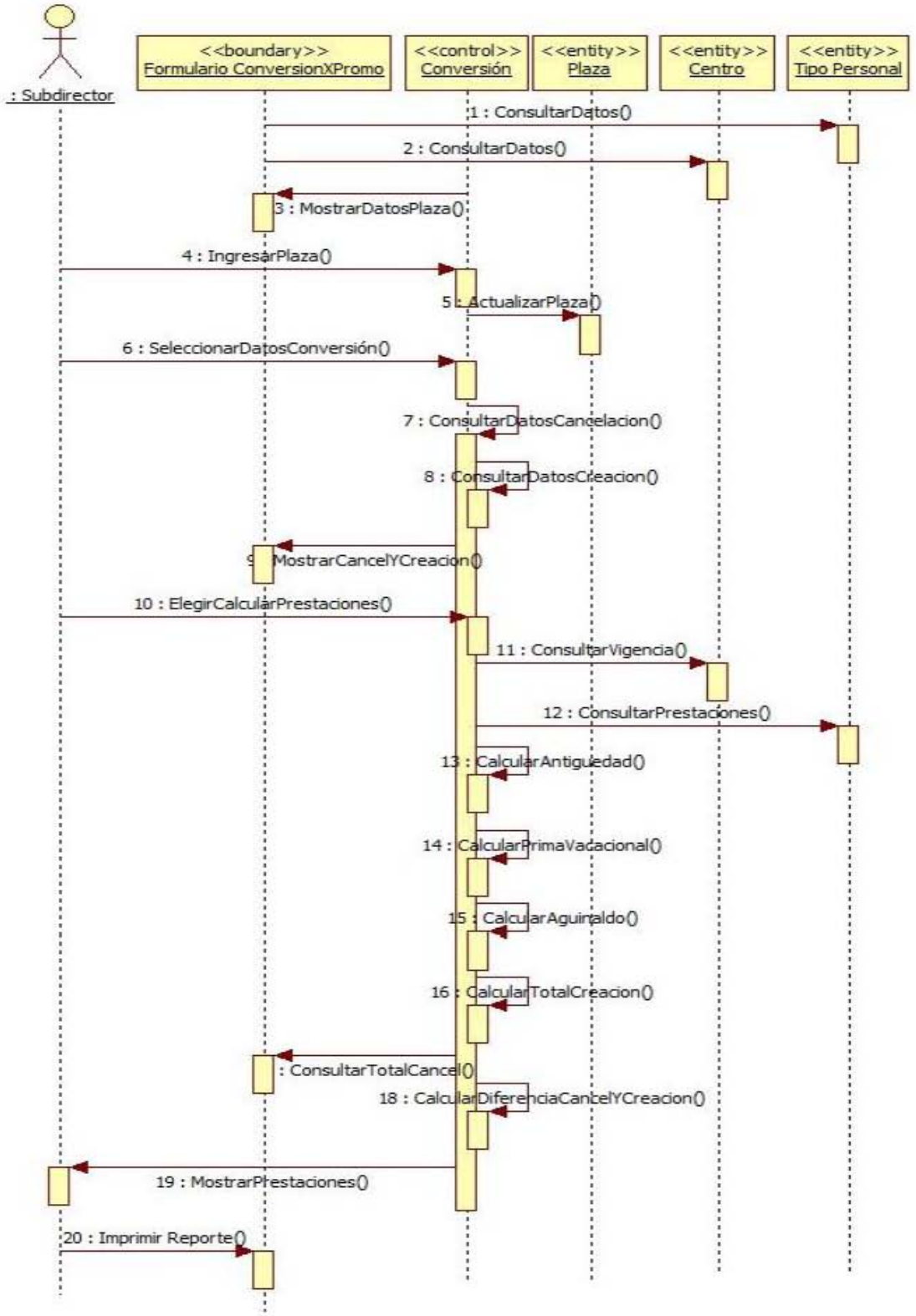
3.3.14.1.2 El caso de uso inicia de nuevo.

3.3.14.2 Reporte

3.3.14.3 El subdirector elige la opción de generar reporte.

El sistema muestra el formato único de movimientos de servicios personales con los datos de los cálculos de las prestaciones realizadas. Con los datos de el nombre de la dependencia, el nombre del centro, el tipo de movimiento, el tipo de personal, la vigencia, el nivel, la zona, el código, el puesto, el total de plazas, el sueldo, el sueldo colectivo por período, la clave de la prestación, el nombre de la prestación, el importe colectivo y el total de todo el FUMSP. Y una opción de imprimir reporte.

Diagrama de secuencia: conversión por promoción de una plaza.



FLUJO DE INCREMENTO SALARIAL

3.4.1 El sistema realiza una consulta para verificar la vigencia en la que cotiza el centro seleccionado.

3.4.2 El sistema muestra el formulario de “incremento salarial” con los datos de: nombre del centro, tipo de movimiento, tipo de personal y vigencia.

3.4.3 En donde despliega la tabla de “incremento salarial” con los datos de puesto, nivel, zona, código, sueldo, aumento y una opción de calcular.

3.4.4 El subdirector ingresa el aumento que se le hará a ese puesto y selecciona la opción de calcular.

3.4.5 El sistema realiza la multiplicación del sueldo por el porcentaje que ingresó y lo muestra en el formulario el incremento salarial con la opción de calcular prestaciones y nueva consulta.

3.4.5.1 Nueva consulta.

3.4.5.1.1 El subdirector elige la opción de nueva consulta.

3.4.5.1.2 El caso de uso inicia de nuevo.

3.4.5.2 Calcular prestaciones.

3.4.5.2.1 El sistema realiza una consulta para verificar las prestaciones que le corresponden a ese puesto con el sueldo del nuevo incremento.

3.4.5.2.2 El sistema realiza el cálculo de la antigüedad: multiplica el sueldo por el porcentaje de antigüedad (27.65%).

3.4.5.2.3 El sistema realiza el cálculo de la prima vacacional: divide el sueldo colectivo por periodo entre 330 días, el resultado lo multiplica por 24 días, el resultado lo multiplica por 1.15.

3.4.5.2.4 El sistema realiza el cálculo del aguinaldo dividiendo el sueldo colectivo por periodo entre 330 días, el resultado multiplicarlo por 20, y multiplicarlo por 1.15.

3.4.5.2.5 El sistema realiza la suma total de los resultados de los cálculos de las prestaciones.

3.4.5.2.6 El sistema muestra el formulario de "calcular prestaciones", en donde despliega la tabla de prestaciones, con los datos de la clave de la prestación, el nombre de la prestación, el importe colectivo de cada una de las prestaciones, así como la suma total del importe colectivo, con una opción de nueva consulta y generar reporte.

3.4.5.2.6.1 Nueva consulta.

3.4.5.2.6.1.1 El subdirector elige la opción de nueva consulta.

3.4.5.2.6.1.2 El caso de uso inicia de nuevo.

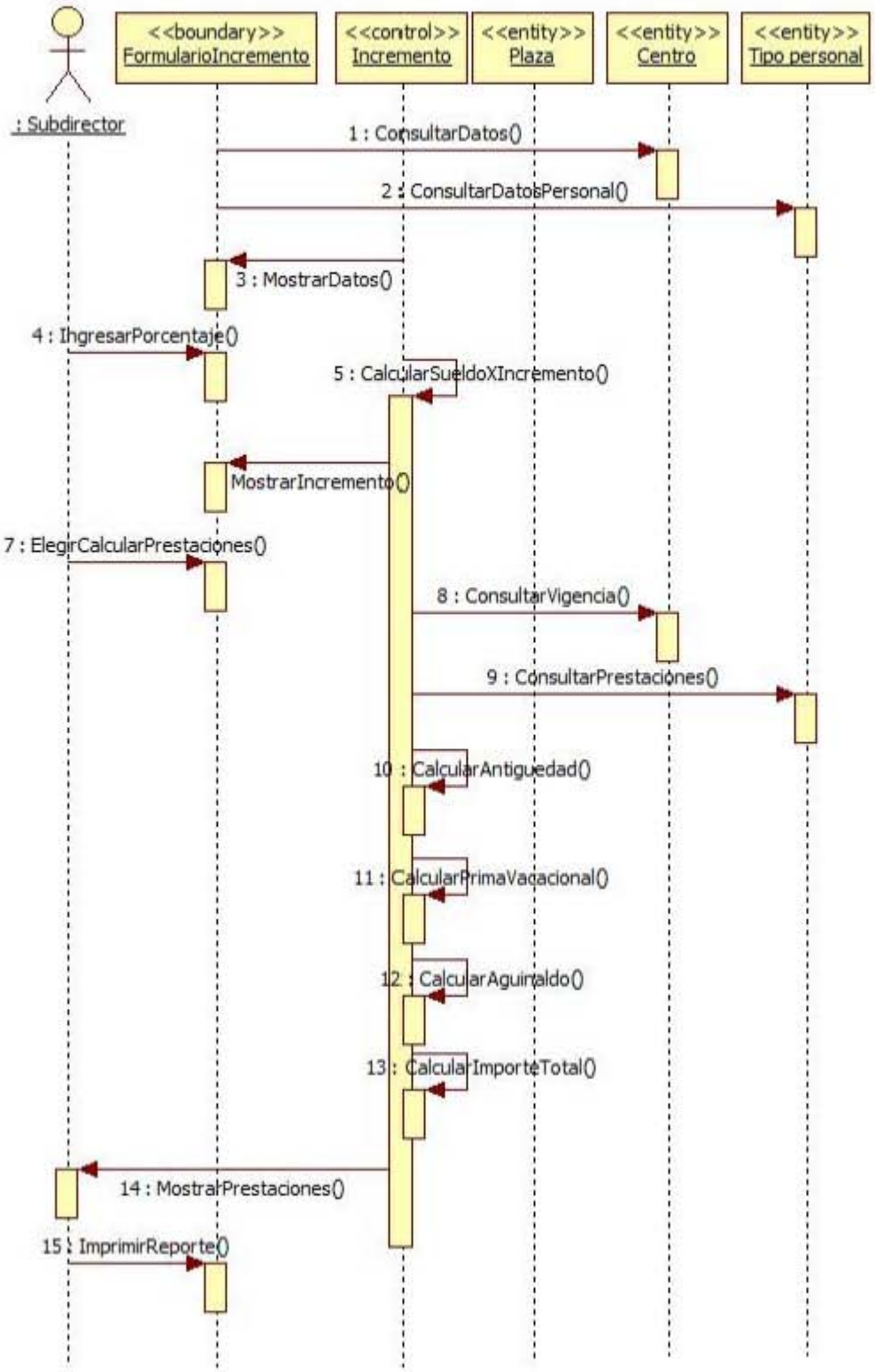
3.4.5.2.6.2 Reporte

3.4.5.2.6.2.1 El subdirector elige la opción de generar reporte.

3.4.5.2.6.2.2 El sistema muestra el formato único de movimientos de servicios personales con los datos de los cálculos de las prestaciones realizadas. Con los datos de el nombre de la dependencia, el nombre del centro, el tipo de movimiento, el tipo de personal, la vigencia, el nivel, la zona, el código, el puesto, el total de plazas, el sueldo, el sueldo colectivo por período, la clave de la prestación, el nombre de la prestación, el importe colectivo y el total de todo el FUMSP. Y una opción de imprimir reporte.

4. El caso de uso termina cuando realiza consultas y genera un reporte de los tipos de movimientos realizados.

Diagrama de secuencia: incremento salarial de una plaza



5. Flujos alternativos

a. El sistema realiza una consulta de acuerdo al tipo de personal para verificar las prestaciones que le corresponden a cada personal.

Para realizar la consulta correspondiente al tipo de personal el sistema debe verificar las prestaciones que le corresponden a ese tipo de personal.

1. El Subdirector verifica que las prestaciones sean las correctas del personal que eligió.
2. El Subdirector realiza una nueva consulta.

6. Subflujos

1. El sistema manejará un catalogo por cada uno de los rubros para realizar las consultas correspondientes.
2. Paso 1 del subflujo: Al realizar una consulta por centro este debe mostrar el personal con el que cuenta ese centro así como las prestaciones que tiene cada tipo de personal.
3. Paso 2 del subflujo: Al seleccionar las plazas e ingresar el sueldo el sistema debe calcular las prestaciones que le corresponde a ese tipo de personal.

8. Post-condiciones

1. Una vez que este generado el reporte, se imprime y se le muestra al Director para que lo autorice y lo firme.

Caso de Uso: Administrar plazas

1. Breve descripción

Este caso de uso se encarga de administrar los sueldos, las plazas, el incremento salarial y de agregar prestaciones.

2. Breve descripción del actor

1. Subdirector: Se encarga de ingresar, modificar, borrar y consultar cada una de las plazas, sueldos, y del incremento salarial del CONACyT en el sistema. Además de administrar las prestaciones por cada plaza.

3. Precondiciones

El Subdirector debe autenticarse o iniciar sesión en el sistema.

4. Flujo básico

El caso de uso comienza cuando el Subdirector ingresa al sistema.

1. El sistema muestra el formulario de “Administrar plazas”, con una opción de cambiar sueldo a la plaza, agregar plaza o puesto, cambiar porcentaje de incremento salarial, agregar prestación.
   FLUJO CAMBIAR SUELDO A PLAZA
2. Si el subdirector elige la opción de cambiar sueldo.
3. El sistema muestra el formulario de “cambiar sueldo a plaza”.
4. El sistema solicita los datos de centro, tipo de personal, puesto o plaza, nuevo sueldo y muestra el sueldo actual. Y una opción de guardar y una de cancelar.
5. El subdirector selecciona el centro, el tipo de personal, el puesto o plaza, e ingresa el nuevo sueldo y elige la opción de guardar.
6. El sistema registra el sueldo con los datos proporcionados.

FLUJO AGREGAR PLAZA

7. El sistema muestra el formulario “agregar puesto”, con una opción de tipo de personal, nivel, plaza, código y sueldo. Y una opción de guardar y una de cancelar.
8. El subdirector selecciona el tipo de personal, ingresa los datos de nivel, plaza, código y sueldo, y elige la opción de guardar.
9. El sistema registra el puesto con los datos proporcionados.

FLUJO CAMBIAR PORCENTAJE DE INCREMENTO SALARIAL

10. El sistema muestra el formulario de “cambiar porcentaje de incremento salarial”, con una opción de año, porcentaje actual y nuevo porcentaje. Y una opción de guardar y una de cancelar.
11. El subdirector selecciona el año, e ingresa el nuevo porcentaje. El sistema muestra el porcentaje actual. Y elige la opción de guardar.
12. El sistema registra el porcentaje del incremento salarial.
    FLUJO DE AGREGAR PRESTACIÓN
13. El sistema muestra el formulario de “agregar prestación” con una opción de clave, prestación, porcentaje. Y una opción de guardar y una de cancelar.
14. El subdirector ingresa la clave, la prestación y el porcentaje, y elige la opción de guardar.
15. El sistema registra la prestación.

6. Flujos alternativos

6.1 En la opción de incremento salarial, se debe ingresar el porcentaje del incremento salarial.

7. Subflujos

NA

9. Post-condiciones

NA

Formulario listar prestaciones

Formulario crear prestación

Formulario listar movimientos

Formulario listar códigos

3.15 Modelo de comportamiento

3.15.1 Diagrama de clases

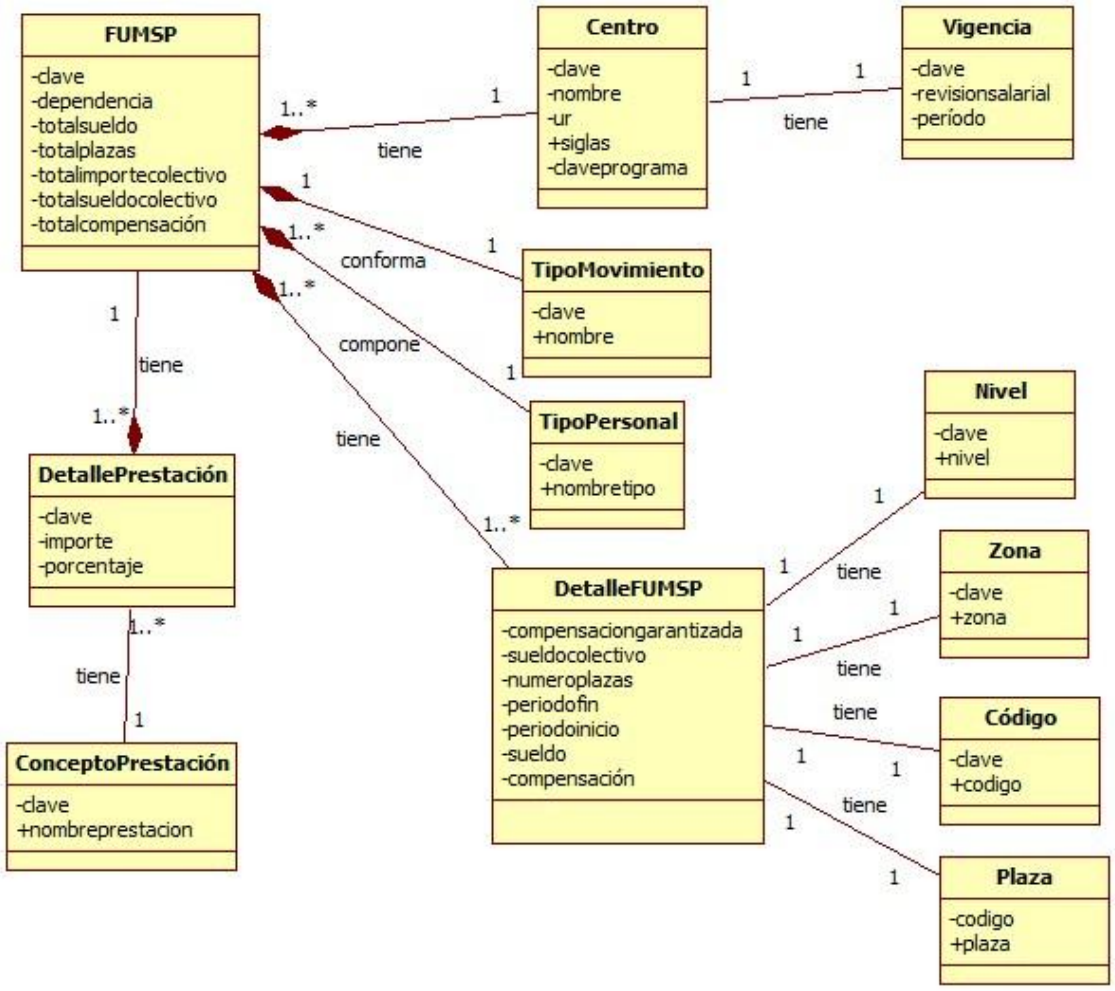


Figura 1.5 Diagrama de clases (Elaboración Propia)

3.16 Diccionario de datos

FUMSP		
Clave	Id del FUMSP.	1{0..9}100,000
Dependencia	Nombre de la institución, CONACyT.	5{a..z A..Z}45
Total sueldo	El total del sueldo por cada puesto.	1{0..9}70,000
Total plazas	Total de plazas de un puesto.	1{0..9}5,000
Total compensación	El total de la compensación por cada puesto de científicos y tecnológicos.	1{0..9}30,000
Total importe colectivo	Suma total del FUMSP, plazas por el sueldo.	1{0..9}500,000
Total sueldo colectivo	Total del sueldo colectivo de las plazas por periodo.	1{0..9}500,000,000

Detalle FUMSP		
Clave	Id del detalle FUMSP.	1{0..9}100,000
Compensación garantizada	Total de la compensación garantizada colectiva por periodo	1{0..9}500,000
Sueldo	El sueldo individual de cada una de las plazas.	1{0..9}10,000
Total plazas	Total de plazas de un puesto.	1{0..9}1000
Sueldo colectivo	Total del sueldo de varias plazas.	1{0..9}100,000
Compensación	Compensación individual de cada puesto del personal de científicos y tecnológicos.	1{0..9}3000
Periodo inicio	El tiempo inicial dentro de un periodo para obtener el inicio de una fecha elegida.	dd = {1..31} mm = {1..12} yyyy = {1900..2100}

Periodo final	El tiempo final dentro de un periodo.	dd = {1..31} mm = {1..12} yyyy = {1900..2100}
Número plazas	Número de plazas por cada centro.	1{0..9}80

Centro		
Clave	Id del centro.	1{0..9}100,000
Nombre	Nombre completo del centro.	1{a..z A..Z\.}100
Vigencia	Mes en el que cotiza el centro.	1{a..z A..Z}20
UR	Clave del centro de acuerdo al CONACyT.	1{0..9}5
Siglas	Siglas de cada centro.	1{A..Z}10
Clave programa	Id del centro de acuerdo a un tabulador del CONACyT.	1{0..9}5

Vigencia		
Clave	Id de la vigencia.	1{0..9}100,000
Revisión salarial	Nombre del mes en el que cotiza el centro.	1{a..z A..Z}20
Periodo	Total de meses del periodo del centro.	1{0..9}12

Tipo de movimiento		
Clave	Id del movimiento.	1{0..9}4
Nombre	Nombre del tipo de movimiento.	1{a..z A..Z}25

Tipo de personal		
Clave	Id del personal.	1{0..9}4
Nombre tipo	Nombre del tipo de personal.	1{a..z A..Z}25

Detalle Prestación		
Clave	Id de la prestación.	1{0..9}100,000
Importe	Importe individual de cada prestación.	1{0..9}5000
Porcentaje	Porcentaje que tiene cada prestación.	1{0..9}16

Concepto prestación		
Clave	Id de la prestación	1{0..9}5
Nombre prestación	Nombre de la prestación	1{a..z A..Z}80

Nivel		
Clave	Id del nivel.	1{0..9}100,000
Nivel	Nombre del nivel.	1{0..9 a..z A..Z\-}20

Zona		
Clave	Id de la zona.	1{0..9}3
Zona	Nombre de la zona.	1{I II III}3

Plaza		
Clave	Id de la plaza	1{0..9}100,000
Plaza	Nombre de la plaza	1{a..z A..Z \.}200

Código		
Clave	Id del código	1{0..9}100,000
Código	Nombre del código.	1{0..9 a..z A..Z}10

Conclusiones

En este trabajo se siguió el proceso de desarrollo de software de *Rational Unified Process* el cual indica que, como y quien tiene que hacer en cada una de las fases que este emplea.

Las fases de RUP definen un conjunto de productos que se tienen que realizar, los cuales garantizan que se diseñan y especifican todas las características del sistema para la administración de movimientos derivados del formato único de movimientos personales con base a los requerimientos que se identificaron que fueron un eje en el desarrollo del sistema porque: facilitó la comunicación con los *stakeholders*, la elaboración del plan de trabajo así como la estimación del esfuerzo del proyecto. Además, se generó un prototipo que ayudó a validar las entradas, procesos y salidas del sistema con los usuarios del sistema.

Durante el desarrollo del sistema se adapto *RUP* con *XP.* El primero ayudó a identificar de forma clara los entregables de cada una de las fases: plan de trabajo que sirvió como eje para construir el sistema, el documento de visión que ayudo a identificar los stakeholders, necesidades y características, la EDT ayudó a subdividir los entregables del proyecto principal y el proyecto de trabajo en componentes manejables, el diagrama de Gantt ayudó a mostrar el tiempo de dedicación previsto para diferentes tareas o actividades a lo largo de un tiempo total determinado, la lista de riesgos a identificar riesgos para poder mitigarlos a través de estrategias definidas, la lista de requerimientos que describe las propiedades o restricciones determinadas de forma precisa que deben satisfacerse, la matriz de trazabilidad que ayudó a confrontar los requerimientos con los casos de uso, el modelo de comportamiento y especificación de casos de uso que ayudó a generar un prototipo y modelar el comportamiento del sistema.

El segundo, *XP,* se utilizó para adoptar la filosofía del desarrollo de software ágil, y apoyo en el empleo del *framework* de *Grails*.

Grails implicó realizar un esfuerzo en su aprendizaje el cual se superó por los conocimientos de la asignatura de informática VI (Programación orientada a objetos) del plan de estudios de la carrera.

La velocidad en el desarrollo del proyecto fue lento porque no se tenía experiencia en la utilización de *RUP.* Gracias a que explica en forma clara las actividades a seguir en cada fase para construir el sistema fue sencillo su aprendizaje. Aunque implicó invertir tiempo en comprender como se hace cada producto.

Para el modelado del sistema se ocupó la notación estándar del *Unified Modeling Languaje*, este lenguaje visual permite especificar, construir y documentar los artefactos de los sistemas, el cual ayuda a comunicarse con cualquier persona que lo conozca.

En las iteraciones iniciales la estimación de tiempo fue errónea pero se tomaron como base para hacer los ajustes correspondientes para hacer las adecuaciones en las siguientes iteraciones de acuerdo a la complejidad del sistema.

Gracias al diseño y prototipos realizados se puede establecer que el sistema para la administración de movimientos derivados del formato único de movimientos personales cumple con los requerimientos y se puede constatar que los cálculos son realizados por el sistema. También, con la información registrada se pueden generar los reportes solicitados.

En pruebas preliminares que se hicieron del sistema para la administración de movimientos derivados del formato único de movimientos de servicios personales se detecta que una persona con conocimientos básicos de computación reduce el tiempo de una hora a media hora.

Este sistema solo automatiza un conjunto de procesos, sin embargo no está contemplando otros procesos y características de la organización, será conveniente que se analice la factibilidad de automatizar los procesos que intervienen en este sistema para que el CONACyT decida si se implementa.

Gracias a este trabajo se aprendió a llevar a cabo la administración de un proyecto para desarrollar un sistema lo cual requiere que por lo menos se haga lo siguiente:

- Identificación de los requerimientos.
- Comunicación constante con los stakeholders.
- Mostrar avances de la propuesta.
- Administración de tiempos.
- Análisis y diseño en forma.
- Estimación del costo del proyecto.

Referencias

Alejandro Martínez, R. M. (2007). *Guía a Rational Unified Process*. Recuperado el 11 de Septiembre de 2011, de http://docs.google.com/viewer?a=v&q=cache:fY8O_03_OMgJ:www.dsi.uclm.es/asignaturas/42551/trabajosAnteriores/Trabajo-Guia%2520RUP.pdf+documentacion+rup&hl=es&gl=mx&pid=bl&srcid=ADGEEShr1NYYhHe_y2x6_wC_hoO2t642eNOCt9CUESPtYtAVazQYbiy64vsC_SVBgzOGGEafEZ5tr2a

Benjamín C, G. (2012). *SOAP (Simple Object Access Protocol)*. Recuperado el 31 de Enero de 2012, de http://www.desarrolloweb.com/articulos/1557.php

Brito, N. (2009). *Manual de desarrollo Web con Grails.* Ediciones ágiles.

CONACyT. (2011). *Historia del CONACYT*. Recuperado el Octubre de 18 de 2011, de http://www.conacyt.mx/Acerca/Paginas/default.aspx

CONACyT. (25 de Marzo de 2011). Manual de Organización. *Manual de Organización del CONACYT*. Distrito Federal: Diario Oficial.

E-Comercio. (2011). *Bisness To Bisness*. Recuperado el 31 de Enero de 2012, de http://ecomercio.uo.edu.cu/downloads/Libros%20y%20cursos/Ebusiness%20Espana/U3B2B.pdf

Francisco, T. A. (2011). *Un algoritmo GRASP para resolver el problema de la programación de tareas dependientes en máquinas diferentes*. Recuperado el 18 de Septiembre de 2011, de http://www.cybertesis.edu.pe/sisbib/2005/tupia_am/html/sdx/tupia_am.html

Freitag, P. (2011). *REST vs SOAP Web Services*. Recuperado el 31 de Enero de 2012, de http://www.petefreitag.com/item/431.cfm

Groovy, E. d. (2012). *¿Groovy? ¿Grails?* Recuperado el 14 de Enero de 2012, de http://www.escueladegroovy.com/informacion/groovy-grails

IBM. (2011). *Rational Unified Process:Best Practices for software development teams*. Recuperado el 11 de Septiembre de 2011, de http://www.ibm.com/developerworks/rational/library/content/03July/1000/1251/1251_bestpractices_TP026B.pdf

IBM. (s.f.). *UML, RUP, and the Zachman Framework: Better together*. Recuperado el 23 de enero de 2012, de http://www.ibm.com/developerworks/rational/library/nov06/temnenco/index.html

Jack, F. (2000). *Negocios exitosos.* McGraw- Hill.

Klein, D. (2009). *The pragmatic Programmers, Grails a quick-start guide.* Collen Toporek.

L. Bass, P. C. (2003). *Software Architecture in Practice.* Adisson Wesley.

María, M. S. (2011). *Metodologías de desarrollo de Software*. Recuperado el 17 de Septiembre de 2011, de http://www.informatizate.net/articulos/metodologias_de_desarrollo_de_software_07062004.html

Paul, K. (2007). *Manual de UML.* México: Mc Graw Hill.

PCMag.com. (2012). *Agile software development*. Recuperado el 21 de Enero de 2012, de http://www.pcmag.com/encyclopedia_term/0,2542,t=agile+software+development&i=37607,00.asp

PCMAG.com. (2012). *Definition of: XP*. Recuperado el 31 de Enero de 2012, de http://www.pcmag.com/encyclopedia_term/0,2542,t=XP&i=55075,00.asp

PCMag.com. (2012). *EXtreme Programming*. Recuperado el 21 de Enero de 2012, de http://www.pcmag.com/encyclopedia_term/0,2542,t=XP&i=55075,00.asp

PCMAG.com. (2012). *Scrum Definition from PC Magazine Encyclopedia*. Recuperado el 31 de Enero de 2012, de http://www.pcmag.com/encyclopedia_term/0,2542,t=Scrum&i=50946,00.asp

Pérez, I. N. (2012). *¿Qué es la tecnología WAP?* Recuperado el 31 de Enero de 2012, de http://www.elcodigo.com/tutoriales/wap/wap1.html

Philippe, K. (2012). *The Rational Unified Process An Introduction.*

Pressman, R. S. (2004). *Ingeniería de Software, Un enfoque práctico.* Mc Graw Hill.

Programming, E. (1999). *Unit Tests*. Recuperado el 12 de Enero de 2012, de http://www.extremeprogramming.org/rules/unittests.html

Programming, X. (2012). *Basic Extreme Programming*. Recuperado el 31 de Enero de 2012, de http://xprogramming.com/what-is-extreme-programming/

Rumbaugh James, J. I. (2002). *El lenguaje unificado de modelado: Manual de referencia.* España: Addison Wesley.

Saffirio, M. (2012). *Mejores Prácticas*. Recuperado el 3 de Enero de 2012, de http://msaffirio.wordpress.com/2009/06/06/mejores-practicas-best-prectices-practicas-propias-own-practices/

Sanchez, M. A. (2011). *Metodologías de desarrollo de software*. Recuperado el 24 de Octubre de 2011, de http://www.informatizate.net/articulos/metodologias_de_desarrollo_de_software_07062004.html

Scielo. (2012). *Aplicación de las metodologías ágiles en el proceso de producción de piezas de arte de nuevos medios: Bio-lencia como caso de estudio*. Recuperado el 22 de Enero de 2012, de http://www.scielo.br/scielo.php?lng=en

UTA. (2012). *Optimizador de medios*. Recuperado el 31 de Enero de 2012, de subversion.assembla.com/.../Plan%20de%20Proyecto%2020090430a

Wikipedia. (2011). *Framework*. Recuperado el 30 de Octubre de 2011, de http://es.wikipedia.org/wiki/Framework

Wikipedia. (2012). *Grails*. Recuperado el 12 de Enero de 2012, de http://es.wikipedia.org/wiki/Grails

XProgramming.com. (2012). *What is Extreme Programming?* Recuperado el 22 de Enero de 2012, de http://xprogramming.com/what-is-extreme-programming/

Glosario

CONACyT. Consejo Nacional de Ciencia y Tecnología.

SHCP. Secretaria de Hacienda y Crédito Público.

CPI's. Centros Públicos de Investigación.

SFP. Secretaría de la Función Pública.

FUMSP. Formato Único de Movimientos de Servicios Personales.

DAAF. Dirección Adjunta de Administración y Finanzas.

IMSS. Instituto Mexicano del Seguro Social.

ISSSTE. Instituto de Seguridad y Servicios Sociales de los Trabajadores del Estado.

CRC. Colaborador-Responsabilidad-Clase.

CESANTIA. Prestación laboral consistente en un mes de salario por cada año laborado.

Incremento salarial: Este movimiento se realiza una vez al año e involucra a todos los Centros CONACYT, únicamente para el personal Administrativo, Científico y Tecnológico y Cargas Administrativas si cuentan con ellas. En este movimiento no se toma en cuenta para el costeo la partida de gasto Despensa. Para el caso del personal Científico y Tecnológico se debe incluir la partida Material Didáctico, dependiendo el Centro se utiliza un porcentaje fijo del 6% sobre sueldo o una tabla de montos fijos por categoría. Para este caso se utilizan los FUMSP de cancelación y creación simultáneamente, haciendo la diferencia el FUMSP de creación con el porcentaje de incremento en la columna de sueldo.

Conversión por promoción: Se utiliza cuando se va a renivelar una plaza y su costeo varía de acuerdo al periodo, tipo de plaza y Centro que lo solicita. En este movimiento se toman en cuenta todas las partidas de gasto. Para este caso se utilizan los FUMSP de cancelación y creación simultáneamente, haciendo la diferencia el FUMSP de creación con el nuevo puesto que se solicita renivelar.

Creación plaza: Se utiliza cuando se solicita la creación de una plaza y su costeo varía de acuerdo al periodo, tipo de plaza y Centro que lo solicita. En este movimiento no se toma en cuenta para el costeo la partida de gasto Prima de antigüedad. Sólo se utiliza el FUMSP creación.

Cancelación plaza: Se utiliza cuando se va a cancelar una plaza y su costeo varía de acuerdo al periodo, tipo de plaza y Centro que lo solicita. En este movimiento se toman en cuenta todas las partidas de gasto. Sólo se utiliza el FUMSP cancelación.

Producto/artefacto: Pieza de información producida, modificada o utilizada por un proceso, es decir son las cosas que el proyecto produce o usa mientras se trabaja hacia el

producto final. Los artefactos son utilizados como insumo por los roles para llevar a cabo una actividad.

Scaffolding: Una vez que se crea el modelo de datos, *Grails* puede generar el controlador y las vistas necesarias para realizar operaciones CRUD con estas entidades.

SOAP: *Simple Object Access Protocol*. Proporciona un mecanismo estándar para empaquetar mensajes, facilita una comunicación del estilo RPC (remote procedure calls) entre un cliente y un servidor remoto.

REST: *Representational State Transfer.* Cada URL es única, representa un objeto. Se puede obtener el contenido de ese objeto mediante un HTTP GET para eliminarlo, después se puede utilizar un POST, PUT o DELETE para modificar el objeto (en la práctica la mayoría de los servicios utilizan un post para esto).

B2B: *Business to business*. Es la relación que se establece entre empresas. La relación puede ir del fabricante a la empresa que le proporciona las materias primas o del fabricante al distribuidor.

WAP: *Wireless Application Protocol.* La tecnología WAP es realmente un estándar impulsado por la industria del sector de las telecomunicaciones con el objetivo de proporcionar un sistema avanzado de servicios de internet para dispositivos móviles.

Lenguaje dinámico: clase de lenguajes de programación de alto nivel que se ejecutan en tiempo de ejecución de muchos de los comportamientos comunes que otros lenguajes pueden realizar durante la compilación. Estos comportamientos pueden incluir la extensión del programa, mediante la adición de nuevos códigos, mediante la extensión de los objetos y las definiciones, o mediante la modificación del sistema de tipos, todos durante la ejecución del programa. Los lenguajes dinámicos proporcionan herramientas directas para hacer uso de ellos.

Tipado: un lenguaje de programación es dinámicamente tipado si una misma variable puede tomar valores de distinto tipo en distintos momentos. La mayoría de lenguajes de tipado dinámico son lenguajes interpretados, como Python o Ruby.

Anexo 1. Formato Único de Movimientos de Servicios Personales

308-A.1.0.0/10.005

FORMATO UNICO DE MOVIMIENTOS DE SERVICIOS PERSONALES

SUBSECRETARIA DE EGRESOS

HOJA 1 DE 1

DEPENDENCIA: CONSEJO NACIONAL DE CIENCIA Y TECNOLOGIA
ENTIDAD: CENTRO DE INVESTIGACIONES EN ÓPTICA, A.C. (CIO)
TIPO DE MOVIMIENTO: INCREMENTO SALARIAL 2011 — VIGENCIA: 1o DE ENERO DE 2011
TIPO DE PERSONAL: CIENTÍFICO Y TECNOLÓGICO

SITUACION PROPUESTA (CREACION DE PLAZAS)

U.R.	ACT. INST.	F.	S.F.	NIVEL	ZONA	CODIGO	PUESTO O CATEGORIA	TOTAL PLAZAS	SUELDO	SUELDO COLECTIVO POR PERIODO	COMPENSACION GARANTIZADA O DE ENLACES	COMP. GARANT. O DE ENLACES COLECTIVA POR PERIODO
*	*	*	*	IITC	II	IITC	PROF. INV. ING. TECNOL. TITULAR "C"	21	28,443.77	7,167,829.51		
				IITB	II	IITB	PROF. INV. ING. TECNOL. TITULAR "B"	23	27,274.53	7,527,770.07		
				IITA	II	IITA	PROF. INV. ING. TECNOL. TITULAR "A"	33	26,105.76	10,337,880.23		
				IIAC	II	IIAC	PROF. INV. ING. TECNOL. ASOCIADO	3	23,768.01	855,648.29		
				IIAB	II	IIAB	PROF. INV. ING. TECNOL. ASOCIADO	11	21,040.48	2,777,343.00		
				IIAA	II	IIAA	PROF. INV. ING. TECNOL. ASOCIADO	3	20,105.53	723,799.19		
				TTC	II	TTC	TÉCNICO TITULAR "C"	10	19,248.10	2,309,771.81		
				TTB	II	TTB	TÉCNICO TITULAR "B"	21	18,398.61	4,634,434.15		
				TTA	II	TTA	TÉCNICO TITULAR "A"	16	17,260.91	3,314,894.14		
				TAC	II	TAC	TÉCNICO ASOCIADO "C"	7	13,017.79	1,093,494.10		
				TAB	II	TAB	TÉCNICO ASOCIADO "B"	8	11,591.97	1,112,828.85		
				TAA	II	TAA	TÉCNICO ASOCIADO "A"	5	10,461.28	627,676.52		
				TAUC	II	TAUC	TECNICO AUXILIAR "C"	2	8,350.29	200,406.89		
				TAUB	II	TAUB	TECNICO AUXILIAR "B"	1	6,682.22	80,186.70		
TOTAL								164	251,741.23	42,763,163.45	0.00	0.00

CAPITULO CONCEPTO	IMPTE. PROP. COLECTIVO
11301 SUELDO	42,763,163.45
13102 ANTIGÜEDAD	9,074,343.29
13201 PRIM. VACACIONAL	3,278,509.20
13202 AGUINALDO	5,464,182.00
13409 MATERIALES	2,565,789.81
15101 FONDO DE AHORRO	5,559,211.25
15401 AJUS. CALENDARIO	593,932.83
14103 IMSS	10,835,786.50
14202 INFONAVIT	3,186,996.03
14401 SVI	675,657.98
14301 SAR	1,274,798.41
TOTAL	85,272,370.74

Índice de cuadros y figuras.