UNIVERSIDAD NACIONAL AUTÓNOMA DE MÉXICO

FACULTAD DE CIENCIAS

CIMS, UN SISTEMA DE ADMINISTRACIÓN DE SOFOMES PARA LA ORGANIZACIÓN CRÉDITO INTEGRAL

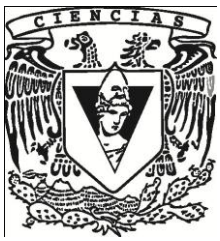
T E S I S

QUE PARA OBTENER EL TÍTULO DE:

LICENCIADO EN CIENCIAS DE LA COMPUTACIÓN

P R E S E N T A :

JULIO GRANADOS MORENO



DIRECTORA DE TESIS:
L. EN C.C. KARLA RAMÍREZ PULIDO
2011

1. Datos del alumno
Granados
Moreno
Julio
56 31 15 36
Universidad Nacional Autónoma de México
Facultad de Ciencias
Ciencias de la Computación
302623365

2. Datos del tutor
L. en C.C.
Karla
Ramírez
Pulido

3. Datos del sinodal 1
Dra.
Amparo
López
Gaona

4. Datos del sinodal 2
Act.
Calos Ernesto
López
Nataren

5. Datos del sinodal 3
M. en C.
María Guadalupe Elena
Ibargüengoitia
González

6. Datos del sinodal 4
Dr.
Jorge Luis
Ortega
Arjona

7.Datos del trabajo escrito.
CIMS, un sistema de administración de SOFOMes para la organización Crédito Integral
163p.
2011

Agradecimientos

Quiero agradecer a mi asesora, Karla Ramírez Pulido, por brindarme la oportunidad de trabajar con ella durante todo este tiempo, otorgándome su confianza y apoyo para desarrollar este proyecto. También por su amplia tolerancia que ha tenido hacia mi persona.

Dedicatoria

A mis padres por apoyarme y aguantar mi mal humor en esos momentos en donde no estoy sonriendo como siempre.

Índice general

Introducción

El interés en la economía surge de un concepto bastante sencillo: el hombre siempre ha querido más de lo que tiene. Deseamos poder vivir en un mundo seguro, lleno de paz, vidas largas, sanas y divertidas. Queremos tener un amplio poder adquisitivo; suficiente para poder comprar de todo: ropa, viajes, automóviles, etcétera; todo lo anterior obtenido siempre por medio del trabajo legal que realizamos, y a su vez teniendo tiempo libre suficiente para realizar actividades lúdicas, así como poder convivir con la familia y amigos. La realidad para muchos es que para poder obtener todo lo anterior necesitamos trabajar bastante. Además nuestra capacidad adquisitiva está limitada por varios factores como el tiempo que tenemos disponible. Estas limitaciones nos dejan naturalmente con una cantidad determinada de pretensiones insatisfechas, las cuales somos incapaces de cumplir **[1]**.

El dinero es un medio de intercambio que nos puede ayudar a cumplir nuestras pretensiones, prácticamente todas nuestras necesidades materiales pueden ser solventadas por este medio. Citando a Adam Smith, pionero de la economía política, el tener dinero otorga a uno la habilidad de "tener el mando" del trabajo de otros, por lo que el poder adquisitivo hasta cierto punto es poder sobre otras personas, hasta el punto en que se está dispuesto a negociar trabajo o bienes por dinero **[119]**. Sin embargo, como lo indica el párrafo anterior, nuestro tiempo y capacidad adquisitiva limitados nos coloca dentro de una economía sin empuje, repleta de pretensiones insatisfechas.

Probablemente no exista un motor más importante en la economía moderna que el otorgamiento de créditos. Comprar un coche, comprar una casa, emprender un negocio, entre otras actividades importantes muy seguramente requerirán que nos otorguen un crédito, para tratar de solventar nuestras pretensiones insatisfechas. Claro está que dicho préstamo debe contar con ciertas características, a fin de que se ajuste a nuestras necesidades y lo podamos liquidar en algún momento.

Nuestro país, México, no es ajeno a la situación antes descrita; sin embargo las "características adecuadas", como tasas de interés, plazo y estabilidad del instrumento, con las que debe contar un crédito no siempre están disponibles para el consumidor. Respecto a las opciones que tenemos los mexicanos para cubrir nuestras necesidades económicas por medio de deudas, prácticamente hemos contado con dos opciones

(desde el 2006): los bancos y las instituciones tipo SOFOL (Sociedad Financiera de Objeto Limitado), parecidas a los bancos pero con menos control gubernamental **[121]**. No obstante, la situación política y económica nacional de principios del siglo XXI propiciaron la creación de un nuevo modelo de negocios facultado para otorgar créditos. Fue así como nació la SOFOM (Sociedad Financiera de Objeto Múltiple). Ésta posee múltiples ventajas operativas y fiscales responsables de su éxito rotundo, dentro de las que podemos mencionar la capacidad para proveer servicios de arrendamiento financiero y factoraje, además de otorgar créditos, carga fiscal reducida (pues el capital operativo no genera impuestos directamente) y bajo control gubernamental, al ser una entidad no regulada directamente por la CNBV (Comisión Nacional Bancaria de Valores) **[13]**.

El auge de las instituciones financieras tipo SOFOM significó no sólo retos a nivel operativo sino una necesidad imperante de contar con sistemas de cómputo eficientes que ayudasen a la gestión de estas instituciones financieras. CIMS (Crédito Integral Management Suite) es un sistema que nace como respuesta a dicha necesidad. Está basado en las operaciones de una exitosa SOFOM real, llamada Crédito Integral; CIMS administra los aspectos principales del otorgamiento de créditos incluyendo gestión de créditos, gestión de clientes y generación de múltiples reportes alusivos.

Los lineamientos y requerimientos del CIMS fueron capturados y analizados al mismo tiempo que Crédito Integral ha ido madurando. Lo cual ha ayudado en gran medida a la estandarización de procesos de la empresa misma y del sistema que la gobierna. Un sistema hecho a la medida, sin pérdida de generalidad que ha ido evolucionando hasta convertirse en una poderosa y eficiente herramienta, la cual se amolda a la empresa y no al revés.

Las funciones que contempla el CIMS son variadas y sobre todo fáciles de usar. Éstas incluyen la generación de tablas de amortización, estados de cuenta, reportes ejecutivos y reportes fiscales. Cabe mencionar que los reportes anteriores son altamente personalizables y versátiles, al mismo tiempo que son compatibles con el software líder en el mercado: Microsoft Excel®[1]. CIMS fue desarrollado estrictamente dentro del marco de la ingeniería de software, siempre alineado con las mejores

[1] Excel: Software capaz de crear y editar hojas de cálculo creado por Microsoft como parte del conjunto de aplicaciones de Office **[78]**. Es el más popular a nivel mundial en plataformas Mac y PC hasta el momento (agosto de 2010) **[131]**.

prácticas de desarrollo y administración de proyectos dictadas por el *Project Management Institute*. Ejemplos de éstas son segmentar el proyecto en fases (inicio, planeación, ejecución, control y monitoreo; pruebas y cierre), realizar el análisis de requerimientos, diseñar el sistema por medio de diagramas UML, crear y ejecutar distintos tipos de pruebas, analizar los riesgos implicados, entre otras.

El CIMS se ha ido convirtiendo en la herramienta principal que apoya a toda la empresa en sus tareas cotidianas y críticas; lo anterior debido a que es capaz de hacer más eficiente la labor del personal operativo, al ocuparse del cálculo de los indicadores vitales de todos los créditos, así como de la gestión completa de clientes, créditos y transacciones automáticamente. CIMS también auxilia al área directiva al proporcionar la información oportuna, pues puede calcular la rentabilidad del negocio por medio de la TIR (Tasa Interna de Retorno), el CAT (Costo Anual Total) y otras medidas importantes de manera automática por el sistema.

El sistema fue desarrollado en Java, un lenguaje de programación orientado a objetos, robusto, popular, libre, distribuido y multiplataforma. Resulta también importante destacar que el software está cimentado en una base de datos cuyo sistema manejador es MySQL, es rápido, gratuito y confiable; éste se ha convertido en el sistema manejador de bases de datos de código abierto más usado alrededor del mundo. Además, el CIMS cuenta con TopLink de Oracle para gestionar la comunicación entre la aplicación de escritorio y la base de datos.

Aunado al objetivo de negocio orientado a cubrir la necesidad de contar con un software de gestión de SOFOMes, se encuentra el objetivo técnico indispensable. La naturaleza colectiva de los equipos de desarrollo de software resulta muy conveniente debido al alto grado de especialización requerido. Por lo tanto, las metodologías de desarrollo de software están orientadas a equipos de más de una persona. Incluyendo las que se describen en este documento. Probablemente la que está más pensada para equipos pequeños, dado que lo mencionan en el libro principal que la describe **[24]**, es Crystal Clear. Sin embargo, el equipo de desarrollo del CIMS está compuesto por una sola persona. Adaptar dicha metodología a equipos de desarrollo de una sola persona es el objetivo técnico de este proyecto.

Todas las herramientas y conceptos de Crystal Clear, sobre todo las que incluyen dinámicas gregarias, tendrán que ser en mayor o menor medida adaptadas para que

funcionen en este particular equipo de desarrollo. Esto no significa simplemente omitir todo lo que requiera la participación de varias personas, sino realmente identificar y abstraer las ventajas de las actividades colectivas de la metodología. Posteriormente, diseñar actividades pensadas para equipos de una persona que conserven los objetivos y la esencia de las que fueron diseñadas para equipos de más de una.

Además, el proyecto prestará atención especial a la experiencia del usuario. El desarrollo del proyecto en el mismo lugar en el que se encuentra el usuario se traduce en una retroalimentación fluida **[24]**. Misma que será aprovechada para analizar la opinión del usuario en términos de facilidad de uso al mismo tiempo que el proyecto avanza.

El CIMS pretende ayudar a las nuevas instituciones financieras tipo SOFOM en algunos de los aspectos operativos y directivos (gestión de clientes, administración de créditos, generación de reportes, apoyo en la toma de decisiones, análisis de rentabilidad de inversiones, automatización de procesos, entre otros). Antes, las empresas de este tipo no contaban con una herramienta especializada, madura y apegada a las mejores prácticas de desarrollo de software. El CIMS, una herramienta eficiente, poderosa, robusta, segura, extensible y fácil de usar; ha ido entendiendo las necesidades del personal y de los clientes de este tipo de instituciones financieras ya que evoluciona de la mano de una de ellas. CIMS trabaja sinérgicamente con las SOFOMes, de esta forma permite al personal directivo enfocarse en lo más importante: sus clientes.

Capítulo 1

SOFOMes y sistemas de crédito

El acrónimo SOFOM significa **Sociedad Financiera de Objeto Múltiple**. Una SOFOM es una institución financiera que está facultada para otorgar créditos a personas físicas y morales así como realizar operaciones de factoraje [2] y de arrendamiento financiero [3] **[2]**. Es preciso aclarar que una SOFOM no puede recibir recursos del público en general y no necesita de autorización del gobierno federal para iniciar operaciones.

Una institución financiera de este tipo goza de ciertos beneficios fiscales, procesales y civiles mismos que se enuncian a continuación:

- El monto de los fondos colocados como créditos, no son tomados en cuenta en cálculos del impuesto al activo.
- Los intereses que hayan sido generados en transacciones de la cartera crediticia con otras instituciones financieras no será objeto de IVA[4].
- Los derechos de créditos con garantía hipotecaria, pueden ser cedidos a un intermediario sin notificación al deudor, ni de escritura pública, ni de inscripción en el Registro Público de la Propiedad y del Comercio. Lo anterior posibilita y facilita la venta de la cartera de créditos.

Las SOFOMes se clasifican en dos grandes ramas: entidades reguladas y no reguladas. La diferencia principal consiste en el grado de control que tendrá la CNBV sobre la institución y la procedencia del capital a colocar en créditos. Las características principales de cada uno de los tipos de SOFOMes antes mencionados se enlistan a continuación:

[2] Factoraje: Es la actividad de comprar a un prestador de servicios o empresa sus facturas y/o documentos por cobrar a cambio de liquidez inmediata **[2]**.
[3] Arrendamiento financiero: Operación financiera en la que una arrendadora adquiere un bien cuyo usufructo será cedido a una persona física o moral siendo este último el arrendatario **[2]**.
[4] IVA: Impuesto al Valor Agregado. A la fecha de la redacción de este texto, este impuesto es del 15% sobre el precio total del bien o servicio **[3]**.

- SOFOM E.R. (Sociedad Financiera de Objeto Múltiple Entidad Regulada): Es aquella que posee vínculos patrimoniales con instituciones de crédito o bien sociedades controladoras de grupos financieros que tengan como integrantes a otras instituciones de crédito **[4]**.

- SOFOM E.N.R (Sociedad Financiera de Objeto Múltiple Entidad No Regulada): Es la institución que no posee vínculos patrimoniales con instituciones de crédito o sociedades controladoras de grupos financieros que tengan como integrantes a otras instituciones de crédito **[4].**

Se dice que una institución tiene vínculos patrimoniales con otra cuando la primera tiene participación en el capital social de la otra, cuando la primera ejerce control sobre la segunda o cuando la segunda tenga accionistas en común con la primera.

Ejercer control se define de la siguiente manera: *"una institución ejerce control sobre la otra cuando la primera posea por lo menos el veinte por ciento de las acciones representativas del capital social de la segunda, controle la asamblea general de accionistas, pueda designar a la mayoría de los integrantes del consejo administrativo o controle a la segunda por medio de otra forma"* **[50]***.*

Por otro lado, se dice que una institución tiene accionistas en común con otra cuando existe una persona o grupo de personas accionistas de una institución que tiene acuerdos con accionistas de la otra institución para tomar decisiones colectivamente y a su vez éstos mantengan una participación mayoritaria en el capital social de la institución en cuestión o puedan ejercer control de la misma.

Resulta importante observar que las SOFOMes no pueden atraer capital directamente de la gente en general **[5]**. Además varios órganos gubernamentales estarán monitoreando las actividades de las SOFOMes en busca de acciones ilícitas como evasión de impuestos y lavado de dinero **[6].**

La disponibilidad de crédito es vital para el desarrollo de un país porque es un factor preciso dentro de la inversión. La existencia de las SOFOMes ayuda a diversificar las opciones que tenemos como personas físicas y morales al adquirir créditos al mismo tiempo que fomentan las bajas tasas de interés y la competitividad en el mercado financiero. Sólo es cuestión de que escojamos la opción que más nos convenga **[7].**

1.1 ¿Cómo funciona una SOFOM?

Ya hablamos de lo que es una SOFOM y del importante papel que juega dentro del sistema financiero de un país y de cómo éstas impulsan el desarrollo de un país. Ahora nos enfocaremos en la forma en la que funcionan las SOFOMes, sin olvidar la estrecha relación con el sistema de administración que presentaremos más adelante. El funcionamiento básico de una SOFOM se ejemplifica en la figura 1.1.

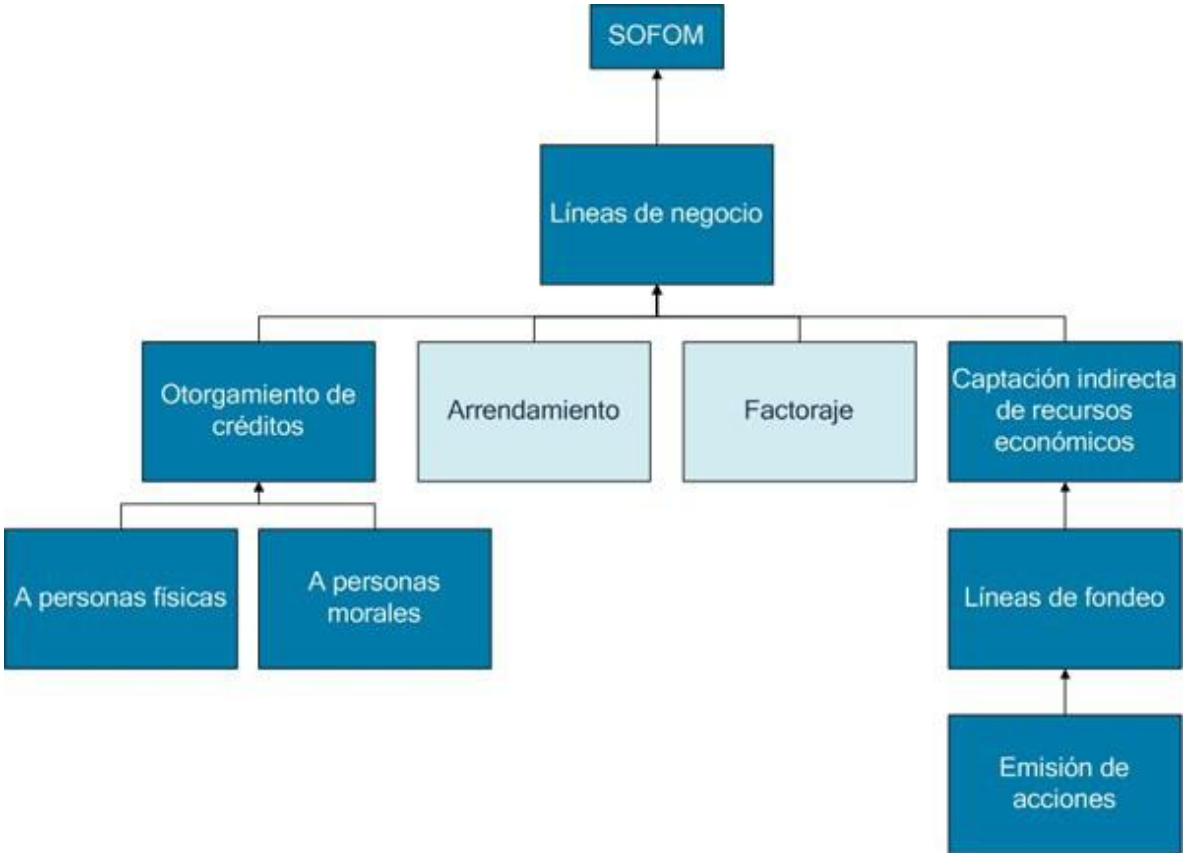


Figura 1.1 – Actividades de una SOFOM (simplificado) **[7]**

A lo largo de este documento estaremos tratando con una SOFOM en particular. Ésta es una institución de crédito real funcionando en Interlomas, Estado de México y cuya razón social es Crédito Integral SOFOM E.N.R. El proyecto de tesis actual está cimentado en el software que ayudará a administrar a esta empresa en particular. La aplicación que se desarrollará es llamada Crédito Integral Management Suite (CIMS).

Cabe aclarar que Crédito Integral actualmente se encuentra en crecimiento, es por eso que actualmente sólo brinda los servicios de otorgamiento de créditos. La figura 1.1 muestra las actividades de arrendamiento y factoraje en un tono más ligero debido a lo anterior.

1.2 ¿Qué es un sistema de crédito (software)?

Los sistemas de crédito son paquetes de software sencillos, en su mayoría diseñados principalmente para aumentar la productividad de una institución de crédito al mismo tiempo que aseguran la consistencia y perfección en varios procesos vitales de la misma **[8]**. Independientemente del aumento de la productividad, se puede percibir que sería catastrófico un error de dedo en los cálculos relacionados con los recursos económicos.

Existen varios tipos de sistemas de crédito dependiendo de su alcance, implementados en varias plataformas y algunos con funciones especiales muy útiles; aunque realmente todos hacen básicamente lo mismo. También, como gran parte del desarrollo de sistemas, la creación de este tipo de software ha evolucionado hacia las plataformas Web **[9]**. Actualmente, casi todas las opciones que se encuentran en el mercado ofrecen una arquitectura de dos o tres capas (ver capítulo 5); lo cual, está relacionado de una u otra forma con la tendencia a aumentar el número de computadoras por persona **[11]**. Lo anterior, dificulta la sincronización de los datos consultados y modificados por las distintas computadoras en caso de que no se tenga una base de datos centralizada y múltiples equipos usando el software **[9]**.

Respecto a las funciones especiales, al usuario final se le ofrecen varias opciones:

- Llenado de formularios vía web.
- Análisis de solicitudes de créditos en aras de la automatización de aprobación o rechazo de las mismas.
- Acceso directo a interfaces bancarias para poder sincronizar estados de cuenta bancarios y automatizar la notificación de movimientos.
- Automatización de generación de documentación legal.
- Repositorio de documentación digital.

- Reportes contables.

1.3 Rol de un sistema de crédito dentro de una SOFOM

Un sistema de crédito puede ofrecer virtualmente toda la funcionalidad relacionada al otorgamiento de crédito que necesitemos. Sin embargo, resulta natural pensar que el software comercial está orientado a un negocio en específico; y por lo tanto es más sencillo analizar las aplicaciones tomando en cuenta a quien se quiere vender el producto. Ahora bien, podemos comparar dos tipos de empresas que otorgan créditos en México: la banca (como grupo financiero[5]) y las SOFOMes.

Nos referimos a la banca como un grupo financiero. Un conjunto de empresas que tienen un fin común; por un lado recabar fondos del público ya sea de forma directa o indirecta y por otro lado colocar esos fondos en forma de deuda. Para fines prácticos, nos enfocaremos en la estructura de un banco en particular: IXE Banco. Su estructura principal se ejemplifica en la figura 1.2.

Tomando esto en cuenta, si quisiéramos diseñar un sistema de crédito que ayudase en la gestión del grupo financiero completo, sería un sistema bastante complejo que cubriría una infinidad de funciones. Para poder enfrentar dicho problema en nuestro caso particular, la dirección de IXE dividió el funcionamiento de sus empresas en varios sistemas que se comunican entre sí para compartir datos de índole común como lo es el saldo total o la cartera de clientes global. Por ejemplo, para el caso de la consulta de saldos para todo el grupo financiero, IXE diseñó un sistema denominado "Sistema de Administración de Liquidez". El diagrama 1.3 muestra su funcionamiento.

[5] Grupo financiero: Conjunto de empresas que proveen servicios en el sector financiero. Los integrantes de este grupo poseen ligas de control como son accionistas y/o personal directivo en común. **[12]**

Figura 1.2 – Funcionamiento de un grupo financiero en particular: IXE **[122]**

La figura anterior muestra las distintas partes de IXE "banco", aunque realmente se le denomina grupo financiero a lo que nosotros conocemos comúnmente como banco. IXE se compone de varias instituciones que se dedican prácticamente a lo mismo: captar recursos económicos del público en general y proporcionarlos a quien los necesite. La diferencia entre cada uno de los integrantes del grupo financiero reside en el fragmento de la población de quien pretenden captar recursos así como el sector a quien le procuran prestar. Por ejemplo, la casa de bolsa y los fondos captan recursos del público inversionista, gente que busca colocar sus excedentes financieros. El dinero se puede invertir directamente en empresas que quieran cubrir sus necesidades de liquidez o prestar por medio de Fincasa[6] en forma de hipotecas. También se puede prestar a través del banco usando tarjetas de crédito o préstamos personales.

[6] Fincasa: Empresa integrante del grupo financiero IXE dedicada al otorgamiento de créditos hipotecarios **[133]**.

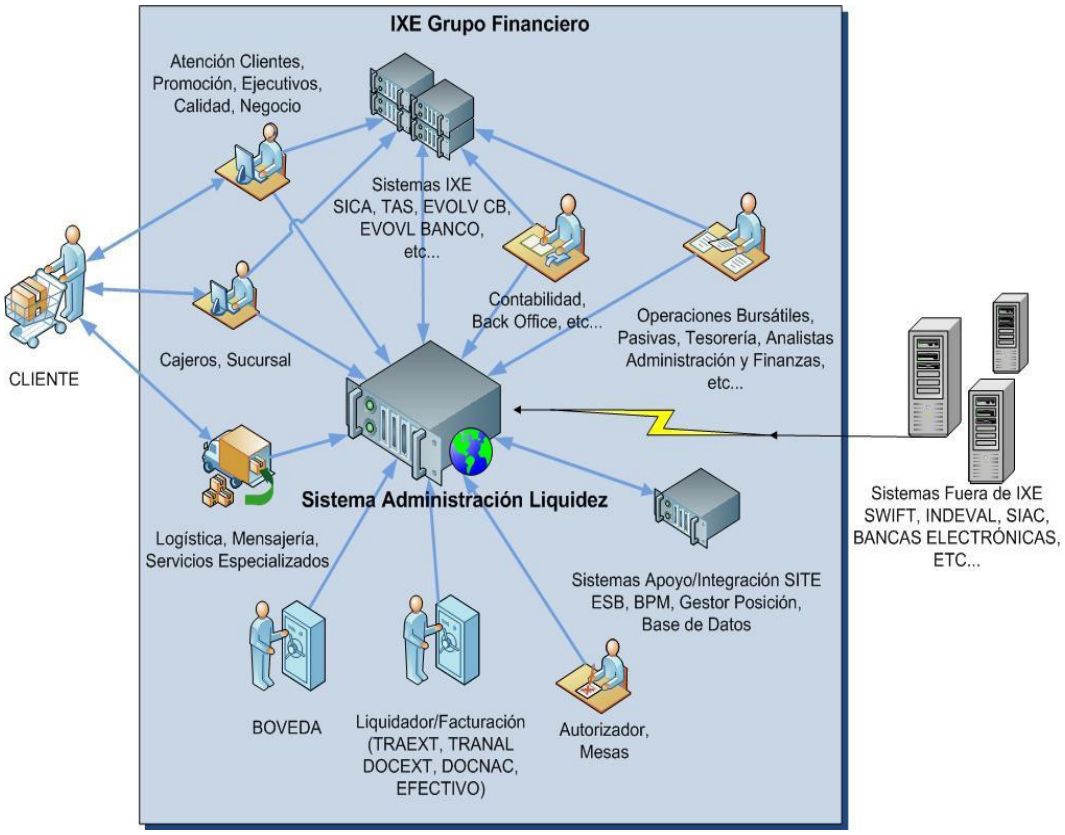


Figura 1.3 – Sistema de Administración de Liquidez (SALI) - IXE ***[123]***

Por otro lado, aunque el grupo financiero esté integrado por varias entidades independientes, la comunicación entre las distintas partes de IXE o cualquier otro grupo financiero es indispensable para su bien funcionamiento. La figura 1.3 expone la interacción entre distintos elementos de IXE y un sistema que revisa los niveles de liquidez de todo el grupo financiero. Prácticamente, todos deben reportarle directamente al SALI. Tan sólo saber la cantidad de efectivo con la que cuenta IXE en un determinado momento resulta una actividad sumamente compleja. Las sucursales bancarias, los servicios de mensajería, los elementos bursátiles, el departamento de contabilidad y muchas unidades más deben de ser coordinadas correctamente a fin de proporciona información veraz y oportuna.

Como podemos ver en las figuras 1.2 y 1.3, hablar de un sistema de crédito puede resultar un poco ambiguo porque existen muchas actividades alrededor del otorgamiento de créditos. Además, también se pueden englobar dichas actividades en diferentes modelos de negocio para satisfacer requerimientos fiscales específicos que podrían traer ventajas competitivas. Por ejemplo, grupos financieros como IXE han

aprovechado el modelo de negocio de SOFOM al convertir negocios ya funcionales de otorgamiento de créditos en SOFOMes pequeñas, para así aprovechar las ventajas de control mínimo por parte de autoridades gubernamentales como la CONDUSEF. **[13]** Lo anterior nos lleva a tener SOFOMes dentro de grupos financieros y combinaciones un poco más complejas.

1.4 Riesgos implicados

Los proyectos de desarrollo de software son inherentemente complicados. Existen muchos factores que tienen que estar coordinados a fin de que éstos sean exitosos. Dichos factores son a la vez fuentes de riesgo, es decir posibilidades de error que deben ser controladas.

La conciencia del riesgo en proyectos de desarrollo de software apareció de varias formas. Al principio surgió simplemente como la demanda del mercado por productos que estuvieran a prueba de fallos. Los desarrolladores entonces buscaron diseños más robustos que permitieran cumplir con el objetivo anterior. Al hacer esto estaban reduciendo lo que hoy llamamos "riesgos del producto". De esta forma y otras parecidas fue que la cultura del riesgo fue siendo adoptada por el gobierno de TI[7]. Sin embargo, en otras disciplinas fue distinto debido a que muchas veces se pensaba que controlar las fallas era aceptar que éstas existían en la dirección de los proyectos desde un principio **[14]**.

El riesgo significaba distintas cosas dependiendo de la perspectiva en la que se viera. Para la dirección en finanzas, el riesgo significaba una pregunta: ¿qué tan rentable era una inversión? El riesgo crédito era estudiado, definido y medido puntualmente dentro de las instituciones financieras. La variación en la retribución de la inversión era una medida de riesgo, lo cual empataba perfectamente con el concepto viejo de variación de la realidad equivale a problema.

Administrar el riesgo se fue convirtiendo, en las distintas disciplinas, en una parte integral de la administración de proyectos. Los líderes de proyecto veían el riesgo de una forma mucho más clara que cualquier otra persona. Los proyectos claramente eran riesgosos. Por ejemplo, construir una presa en la selva es claramente riesgoso;

[7] Gobierno de TI (Tecnologías de la Información): Es una disciplina perteneciente al gobierno corporativo enfocada en los sistemas de las tecnologías de la información, su desempeño y la gestión de sus riesgos **[124]**.

construir ductos subterráneos de petróleo también, en éstos y muchos casos más el proyecto mismo se volvía sinónimo de riesgo; por lo que, administrar un proyecto implicaba manejar riesgos.

Finalmente todas estas influencias fueron contribuyendo a que el manejo del riesgo en la administración de proyectos de TI fuera indispensable. El manejo del riesgo es una parte esencial en la vida de un proyecto y se ha convertido en algo básico para la supervivencia del mismo. Pensar en identificar riesgos y manejarlos ya no es potencialmente visto como algo negativo, ésta es la nueva cultura del riesgo y la dirección que TI ha adoptado.

1.5 Taxonomía de los riesgos

La clasificación de los riesgos en Ingeniería de Software no resulta una tarea sencilla. Uno puede comprobar esto simplemente al tratar de hacerlo dentro de un proyecto de TI. Resulta mucho más sencillo que una persona que se encuentra más cerca de los riesgos responda por ellos de forma individual, debido a que supuestamente los conoce mejor. La forma más práctica y "científica" de manejar los riesgos es clasificarlos de acuerdo con sus atributos, por lo que primero se clasifican los riesgos y después se procede a escoger cuales serán asumidos en el manejo de nuestro proyecto.

La taxonomía de los riesgos involucra estudiar las características de todos los riesgos encontrados para posteriormente revisar las características que comparten. Basados en dichas características es que se definen clases de riesgos que compartan ciertos atributos. Las clases se pueden obtener de una base de datos de clases de riesgos o simplemente definirse sobre la marcha basadas en una estructura lógica de características.

Cuando pensamos en la clasificación de riesgos por lo general pensamos inmediatamente en un árbol de riesgos. La figura 4.1 muestra parte de un árbol de riesgos del "Software Engineering Institute" de la universidad "Carnegie Mellon". La idea es tener una forma de clasificar los riesgos, la cual haga énfasis en tipos y subtipos de riesgos.

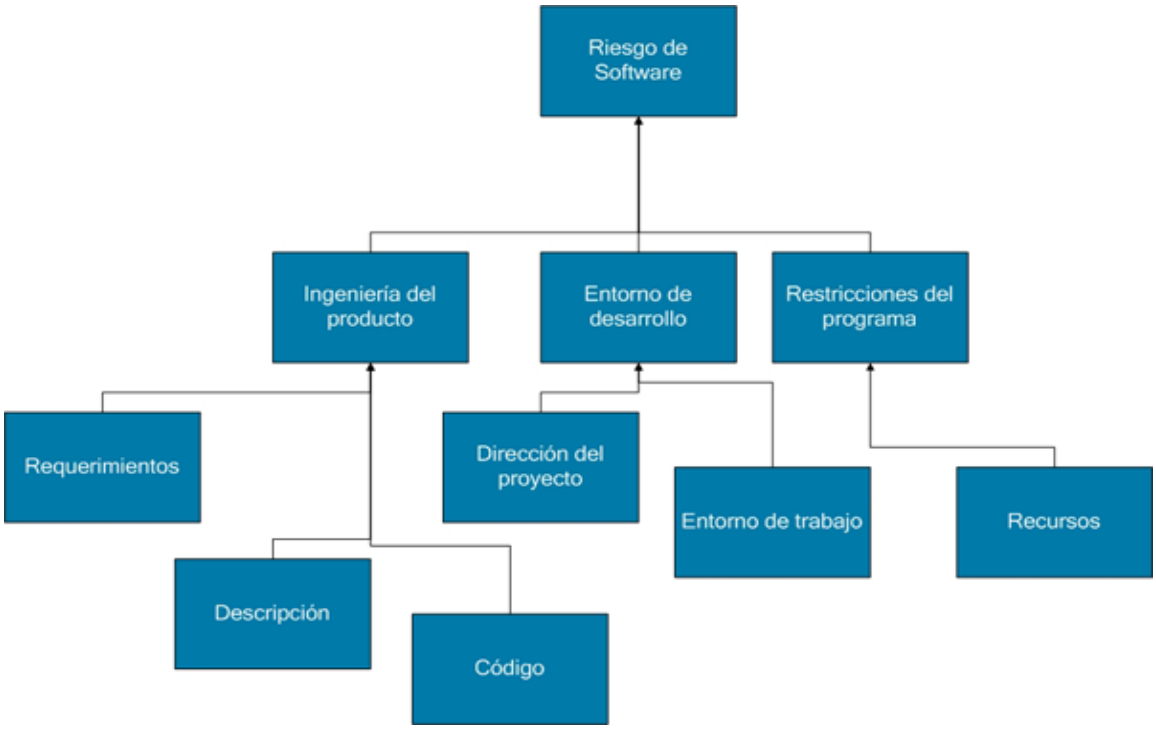


Figura 1.4 – Parte de un árbol de clasificación de riesgos **[14]**

La figura anterior nos expone a los riesgos de software divididos en tres principales ramas: **ingeniería del producto, entorno de desarrollo y restricciones del programa**. Posteriormente las ramas principales se fragmentan en hojas mucho más manejables. Esta clasificación facilita el análisis de los riesgos debido a que nos permite estudiarlos basados en la actividad y/o área de conocimiento en cuestión. Además comprender el área de conocimiento nos permite consultar a un experto específico en la materia para ayudarnos a mitigar el riesgo **[14]**.

La clasificación de riesgos es una poderosa herramienta que tenemos disponible dentro del proceso de gestión de los mismos. Sin embargo, las acciones que tomemos para mitigarlos serán las que realmente nos ayudarán a mantener nuestro proyecto dentro de los márgenes de tiempo, costo, alcance y calidad **[14]**.

Capítulo 2

Metodologías de desarrollo de software

Los programas de cómputo se han convertido en la piedra angular en la toma de decisiones en las empresas. Forman parte de la base de la investigación científica moderna así como la solución de problemas relacionados con la ingeniería en general. Lo anterior nos preocupa debido a que el software, como cualquier cosa creada por el hombre, es propenso a errores. La alta complejidad de los proyectos de software nos ha dejado claro que es indispensable adoptar un enfoque sistemático para abordar el problema. Las metodologías de desarrollo de software son presentadas como una solución a dicho problema.

Las metodologías de desarrollo de software constituyen los procesos, métodos y capas de herramientas, que los ingenieros de software utilizan para poder abordar el problema de desarrollo de software de una manera disciplinada y metódica. En el proceso de software, se selecciona una de éstas según sea la naturaleza del proyecto base para después aplicarla.

En las siguientes secciones se presentan varias metodologías de desarrollo de software. Cada una se encuentra agrupada de acuerdo al tamaño del proyecto, las necesidades del cliente en cuestión, así como los recursos disponibles para el desarrollo del mismo. Dentro de las metodologías se encuentran las siguientes:

- Metodología lineal secuencial.
- Metodología de construcción de prototipos.
- Metodología de desarrollo ágil.
- Metodología incremental.
- Metodología espiral.
- Metodología de métodos formales.

2.1 Metodología lineal secuencial

Los orígenes de esta metodología están ligados a procesos de ingeniería más generales **[15]**. Ha sido llamado también "metodología en cascada" debido a la forma

en la que están organizadas las partes del mismo. Un diagrama que ilustra los pasos involucrados en el proceso se encuentra a continuación (figura 2.1):

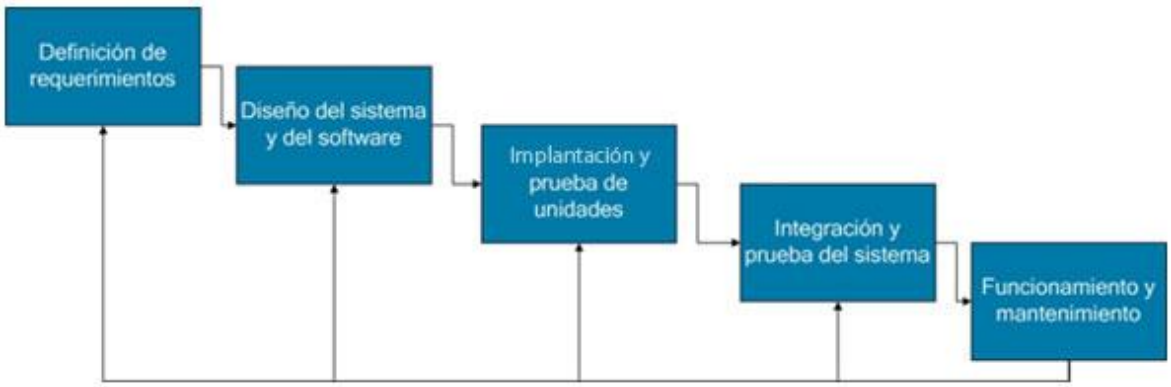


Figura 2.1 – Metodología en cascada **[46]**

Esta metodología comprende, como se muestra en la figura 2.1, una serie de pasos separados y distintos entre sí que están relacionados con el desarrollo del software y las especificaciones del mismo. La descripción de los pasos anteriores se encuentra a continuación.

Análisis y definición de requerimientos

Las consultas con el usuario final son la fuente más importante de información en cuanto a la recopilación de servicios, restricciones y metas del sistema. Es muy importante detallar cada uno de los pasos para evitar problemas derivados de confusiones o malos entendidos.

Diseño del sistema y del software

Lo primero que se hace es dividir los requerimientos en sistemas de hardware o de software. Una arquitectura completa del sistema debe de estar terminada al final de este paso, habiendo comprendido las abstracciones principales del sistema de software y sus relaciones.

Implantación y prueba de unidades

En este punto se debe revisar que cada una de las partes del sistema cumpla con las especificaciones iniciales.

Integración y pruebas del sistema

Como su nombre lo indica, este paso comprende la integración de los componentes del sistema para que se conformen en una sola pieza. Además el sistema, como un componente entero, se debe probar para poder asegurar que éste cumpla con los requerimientos originales. Posteriormente, una vez que el sistema ha superado las pruebas, se entrega al cliente.

Funcionamiento y mantenimiento

Esta es la fase más larga en la vida del sistema (por lo general). Comprende el descubrimiento, la corrección de los errores que no habían sido detectados con anterioridad así como la mejora de la implementación de las unidades del sistema.

Es necesario tomar en cuenta que el resultado final de cada uno de los pasos por lo general es un documento firmado por el cliente (representante legal). La fase siguiente no debe de empezar sin terminar la actual. Sin embargo, en la práctica las fases se enciman y es bastante común que dentro de una fase sea forzoso corregir errores de una fase anterior. Lo anterior resulta complicado debido a que la metodología dicta que el proyecto sea dividido rígidamente desde un principio en partes distintas.

Cabe mencionar que todos los modelos tienen sus ventajas y desventajas, en particular éste tiene la ventaja de que la documentación es realizada al terminar cada una de las fases de la metodología. No obstante, la restricción que prohibe empezar la siguiente fase sin terminar la actual resulta ser una desventaja, debido a que en la práctica es común que nos veamos obligados a corregir errores de fases anteriores antes de continuar. La conclusión es que esta metodología resulta muy útil sólo cuando los requerimientos del sistema son bastante claros desde un principio.

2.2 Metodología de construcción de prototipos

La idea de que un cliente tenga todos los requerimientos de un sistema total y éstos sean completamente claros desde el principio del desarrollo, es un poco pretenciosa y lejos de la realidad. Lo cierto es que en la práctica es bastante común que un cliente tenga muy presente los objetivos generales del software, pero no los requerimientos detallados de entrada, proceso y salida. Además, también existen ciertas dificultades a

las que se pueden enfrentar los desarrolladores, por ejemplo a la efectividad de algún algoritmo o a la interacción entre el usuario final y el sistema. La problemática descrita sugiere que es necesario utilizar una metodología que se pueda adaptar a los cambios en los requerimientos. La metodología de construcción de prototipos es introducida como una opción efectiva en situaciones como la anterior.

El paradigma de construcción de prototipos se encuentra estructurado básicamente de la siguiente forma (figura 2.2):

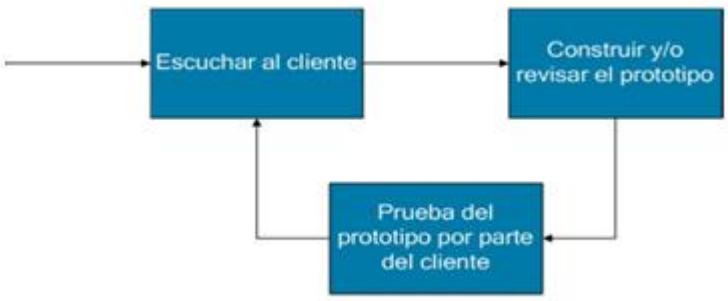


Figura 2.2 – Metodología de construcción de prototipos **[46]**

Analizando la figura anterior, podemos suponer que la base de esta metodología de desarrollo es la construcción de prototipos. Primero recopilamos los requisitos, después elaboramos un diseño rápido y finalmente construimos el prototipo, el cual es sometido a pruebas por parte del cliente y posteriormente aprobado. Por último el prototipo es usado para refinar los requisitos del software a desarrollar.

Parece bastante elemental la metodología, pero la construcción de prototipos nos lleva hacia la siguiente pregunta: ¿qué debemos hacer con el prototipo una vez que éste ha sido aprobado y revisado por el cliente? La respuesta resulta sencilla. Aunque nos veamos tentados a darle una especie de mantenimiento al prototipo, es muy importante que lo tiremos debido a que es muy probable que éste sea muy lento, grande, torpe o una combinación de las anteriores **[16].**

Es importante recordar que, como cualquier metodología de desarrollo de software, éste tiene sus ventajas y sus desventajas. Las ventajas ya han sido discutidas y las desventajas se detallan a continuación. Siempre existe la tentación y la presión del cliente por usar y tratarle de dar mantenimiento a un prototipo prácticamente inservible. Una vez que el desarrollador entiende que no hay una buena razón para

hacer eso, hay que superar el hecho de que el cliente siempre va a querer usar el prototipo, pues podría llegar a pensar que está todo listo. Sin embargo el prototipo puede no cumplir con los requisitos de calidad y de mantenimiento del software. También está el riesgo que, por parte del programador, exista un mal uso de las tecnologías involucradas en el sistema por la presión de terminar en muy poco tiempo el prototipo (algoritmos, lenguajes de programación, sistema operativo de implementación, entre otros).

Como hemos observado, la metodología de construcción de prototipos es presentada como una excelente alternativa para poder planear el desarrollo de un sistema. No obstante hay que tener clara una restricción fundamental: los prototipos desarrollados son mostrados sólo como una forma de definición de requerimientos y éstos no podrán ser usados en la versión final. De lo contrario, los problemas mencionados en el párrafo anterior se podrían presentar.

2.3 Metodología DRA (Desarrollo Rápido de Aplicaciones)

Esta es una metodología basada en la metodología secuencial (o de cascada) descrita anteriormente. El punto central de esta metodología es la corta duración en los periodos de desarrollo. El sistema original está dividido en varios componentes que son construidos en ciclos de desarrollo pequeños, y en caso de que se conozcan perfectamente los requisitos y se tengan bien delimitados en el proyecto, esta metodología nos puede ayudar a desarrollar software funcional y de alta calidad en periodos de 60 a 90 días (muy cortos) **[17].** Esta metodología está compuesta de las siguientes fases (figura 2.3):

Modelado de Gestión

La información generada por las funciones de gestión se presenta de tal forma que responda a las preguntas de ¿qué información se maneja?, ¿quién la genera?, ¿a dónde va? y ¿quién la procesa?

Modelado de datos

Primero se definen un conjunto de objetos de datos necesarios para dar soporte a la empresa. Posteriormente se definen los atributos de estos objetos y la relación que hay entre ellos.

Modelado del proceso

Se crean las interfaces necesarias para poder implementar las funciones relacionadas con la creación, modificación, eliminación o recuperación del modelo de datos.

Generación de aplicaciones

La generación de código gira alrededor del uso de lenguajes de programación y técnicas para ahorrar tiempo como es la reutilización de componentes.

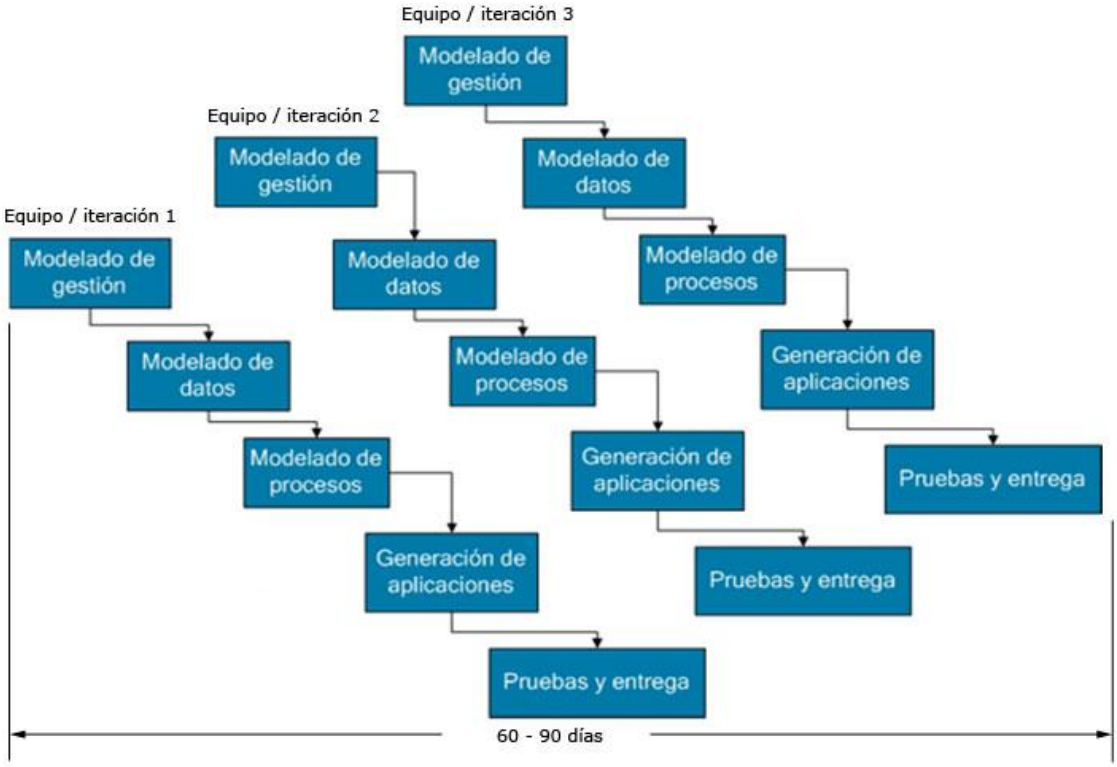


Figura 2.3 – Desarrollo Rápido de Aplicaciones **[46]**

Pruebas y entrega

Una vez más, los componentes se tendrán que probar para asegurar su calidad y cumplimiento con el alcance definido. Sin embargo hay que recordar que ya tenemos

probados varios componentes y que en teoría deberíamos reutilizar algunos. La única diferencia es que los componentes nuevos tendrán que ser probados por separado y a su vez todos los componentes se tendrán que probar juntos.

Hay que tener en cuenta que esta metodología también tiene ciertas desventajas **[18]**:

- Los proyectos grandes necesitan bastantes recursos humanos, pero esta metodología no ofrece reducciones en términos de recursos humanos.
- El hecho de que los proyectos puedan ser realizados en períodos cortos de tiempo necesitan un esfuerzo y compromiso extra por parte de los desarrolladores y demás integrantes del equipo.
- Esta metodología sólo es conveniente cuando el sistema total se puede modularizar eficientemente. La alta dificultad de fragmentar el sistema o la ineficiencia al hacerlo por lo general nos indican que tal vez esta metodología no sea la óptima.
- El desarrollo rápido de aplicaciones puede no ser apto para proyectos que requieren interacción alta con sistemas de legacía (existentes). Tampoco es una buena idea usar esta metodología cuando estamos haciendo uso de nuevas tecnologías que no han sido probadas a fondo.

Metodología "Crystal Clear"

Crystal Clear es una metodología parte de la familia Crystal y del grupo de metodologías de desarrollo ágil. Todas ellas han sido desarrolladas por Alistar Cockburn y son parte de las metodologías ágiles. Esta metodología está pensada para equipos pequeños de hasta seis u ocho personas. Los proyectos en donde se aplique esta metodología no son de misión crítica, pues el objetivo principal de Crystal Clear es la eficiencia. La razón por la cual esta metodología es popular en equipos pequeños de desarrollo, es que ésta se encuentra pensada para equipos poco numerosos desde el principio. Sus creadores argumentan que es más fácil empezar con una metodología que está diseñada para equipos chicos, en vez de usar una que está diseñada para equipos grandes y después recortarla para adaptarla a las necesidades del equipo en específico **[24].**

Los aspectos generales de Crystal Clear se enuncian a continuación:

- Está basado principalmente en la observación de un gran número de equipos exitosos. Muchas otras metodologías están basadas en un número mucho

menor de proyectos. Por ejemplo Extreme Programming nació del proyecto C3 de Chrysler **[24]**.

- El enfoque que tiene Crystal Clear hacia los procesos relacionados con el desarrollo de software es muy distinta a la de las otras metodologías. Esto nace del principio de que estamos acostumbrados a la falsa creencia que al seguir cierto proceso obtendremos software de calidad. Dicha meta teóricamente se alcanzaría por medio de la adopción de notaciones de diagramación y flujos de trabajo. Sin embargo, según Crystal Clear, esto no es lo que más importa. Esta metodología se centra en aislar los factores que más están involucrados en el éxito de un proyecto **[24]**.
- Está diseñada para funcionar bajo esquemas de presupuesto acotado **[24]**.
- Crystal Clear funciona para equipos que quieran adoptar sólo una fracción de toda la metodología. De hecho se puede combinar con otra metodología y sólo tomar usar ciertos aspectos de Crystal Clear **[24]**.
- Incluye instrucciones para poder adoptar la metodología y para adaptarlo a las necesidades de la organización y del proyecto **[24]**.

Últimamente Crystal Clear ha ganado popularidad sobre todo dentro de grupos pequeños de programación. Sin embargo, como todos los métodos nuevos, no ha sido adoptado en muchos proyectos debido a su reciente aparición; pero el hecho de que esta metodología pueda ser usada por partes le da mucha más versatilidad.

2.4 Metodología incremental

Es bien sabido que todos los sistemas complicados evolucionan con el tiempo. El software no es la excepción **[19]**. Es bastante común que las presiones del mercado junto con la forma en la que los requerimientos del proyecto están organizados provoquen una entrega de producto muy sencilla como primera versión. Esta primera versión cambiará después y se irá desarrollando al mismo tiempo que nuevas funcionalidades sean añadidas al proyecto.

Las metodologías de desarrollo descritas con anterioridad no toman en cuenta la naturaleza evolutiva de un sistema en producción, y es en este caso cuando debemos considerar a las metodologías de desarrollo iterativas, las cuales giran alrededor del principio de ir agregando gradualmente más funciones al proyecto de software.

La metodología incremental es la conjunción de elementos de la metodología lineal secuencial con ideas de la metodología de construcción de prototipos. Como se muestra en la figura 2.4, cada incremento de la metodología es un ciclo de la metodología lineal secuencial **[20]**.

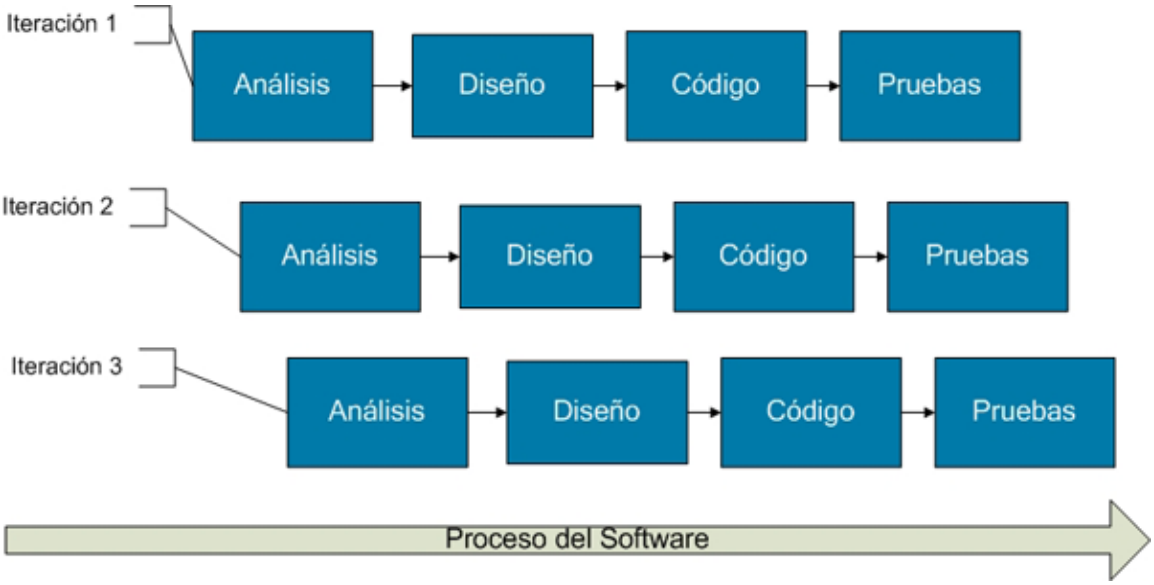


Figura 2.4 – Metodología incremental **[46]**

Como ejemplo podríamos tomar un programa sencillo de edición de imágenes. En la primera versión se incorporarían funciones de apertura y edición de archivos. En una segunda versión podríamos agregar funciones propias de la edición de imágenes. Para finalizar, en una tercera iteración veríamos la posibilidad de perfeccionar las funciones existentes. Hay que tomar en cuenta que, para ayudar en la definición de requerimientos, en cualquier ciclo podemos hacer uso de técnicas de construcción de prototipos.

Hay que tener en cuenta que aunque esta metodología incorpora funciones a la metodología lineal secuencial y de la de construcción de prototipos, ésta difiere de las últimas dos en que posee una naturaleza iterativa y que cada incremento pretende entregar una versión funcional del proyecto. Lo anterior, contrasta con la metodología de construcción de prototipos específicamente con la parte que especifica que cada prototipo está destinado a ser desechado.

Una de las características más importantes de la metodología incremental reside en que los requerimientos de personal puedan variar a través de los incrementos. Por ejemplo, el primer incremento podría requerir mucho menos recursos humanos que el último incremento en donde el desarrollo se complica.

2.5 Metodología en espiral

Esta metodología fue originalmente propuesta por Bohem **[21]**, se caracteriza por tener como base una espiral en vez de una secuencia de pasos con regresión hacia una actividad anterior. Cada parte del desarrollo del proyecto es representada por un ciclo de la espiral. Entonces tendríamos que los ciclos más internos están relacionados obviamente con las primeras fases del desarrollo y los ciclos más externos con las fases más cercanas al final.

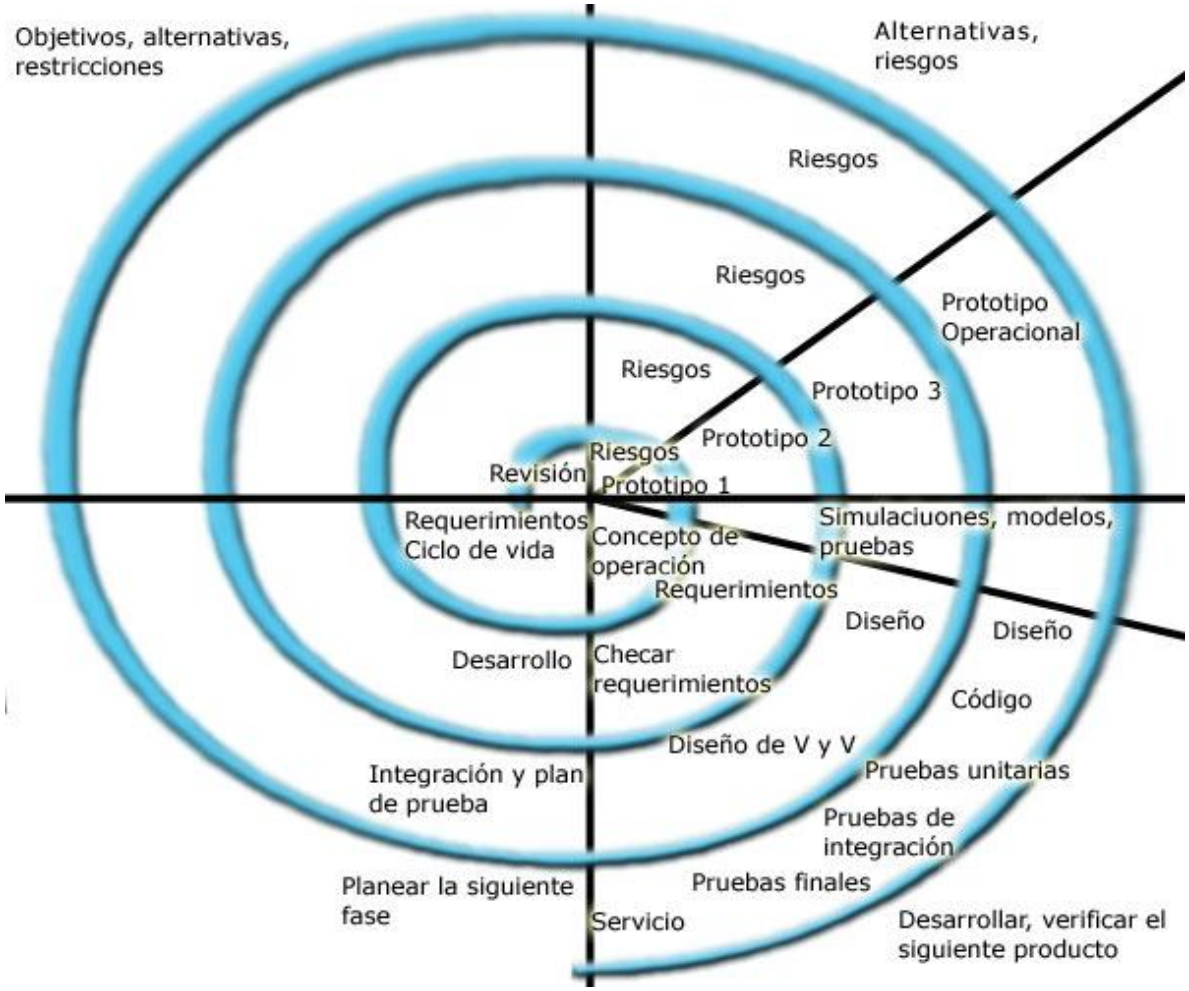


Figura 2.5 – Metodología espiral **[46]**

La figura 2.5 menciona los conceptos básicos de la metodología en espiral. Ésta consta de cuatro secciones principales:

- Definición de objetivos: Aquí los objetivos y restricciones del proyecto en general se detallan, así como los riesgos o asociados.
- Análisis y reducción de riesgos: Aquí se identifican y mitigan las condiciones peligrosas encontradas.
- Desarrollo y validación: En este rubro se desarrolla el código y se hacen las pruebas pertinentes para asegurar la calidad del producto.
- Planeación: En este paso se evalúa la necesidad de un siguiente ciclo en el espiral o si es posible dar por terminado el proyecto.

Una de las características principales que distinguen a esta metodología de las demás es que el riesgo se identifica y se maneja en cada ciclo. También hay que notar que no existen fases definidas como especificaciones o diseño. Los ciclos en la espiral son escogidos con base en lo requerido.

Esta metodología ofrece numerosas ventajas, sin embargo hay que recordar que en la ingeniería de software no hay una solución que sea la adecuada para todos los casos. La metodología en espiral puede presentar problemas en esquemas bajo contrato en proyectos grandes. El cliente podría llegar a pensar que se carece de formalidad y que puede ser difícil o imposible de controlar. Además no posee tanto tiempo de uso como las metodologías más viejas como la de construcción de prototipos o la lineal secuencial.

2.6 Metodología de métodos formales

Esta metodología está basada en la utilización de técnicas orientadas a la especificación matemática del software. Está cimentada en el concepto de transformar una descripción matemática a través de diferentes representaciones de un programa ejecutable. Estas transformaciones preservan la correctud debido a su naturaleza matemática. Esto hace inmediato probar que el programa cumple con las especificaciones originales.

Esta metodología se basa en un principio llamado "ambiente cleanroom". El proceso de ingeniería de software *cleanroom* (cuarto limpio) es un proceso de desarrollo de software que tiene como fin principal desarrollar software con un nivel de confiabilidad certificable. Cleanroom fue desarrollado por Harlan Mills y su grupo de colaboradores en IBM **[22].** El principio básico de cleanroom gira alrededor de la prevención de los errores en vez de la corrección de los mismos. De hecho el nombre viene de los cleanrooms[8] que usan la industria de electrónicos en la fabricación de semiconductores en un esfuerzo por producirlos sin defectos en su manufactura. Por lo general los proyectos de software basados en este principio son usados en proyectos de misión crítica **[23].**

El principio de cleanroom está basado en las siguientes normas:

- Desarrollo de software basado en métodos formales.

  Se caracteriza por el uso de la estructura del método de caja[9] (*Box Structure Method*) para hacer las especificaciones y el diseño de un producto de software. El hecho de que el producto concuerde con las especificaciones originales es verificado a través de revisiones en equipo.

- Implementación incremental a través del control de calidad estadístico.

  Cleanroom usa un enfoque iterativo en materia de desarrollo. Cada iteración es probada para corroborar que posee la calidad requerida. En caso contrario, regresamos a la fase de diseño para corregir el error.

- Pruebas.

  Una muestra representativa de entradas y salidas es elegida y probada, las cuales están basadas en la especificación formal. Después, esta muestra es analizada estadísticamente para poder calcular la confiabilidad del producto.

Volviendo al tema de los métodos formales, presentamos la figura 2.6 que enuncia las fases que componen a dicho método.

[8] Cleanroom: Un cleanroom es un ambiente que posee muy bajo nivel de contaminantes atmosféricos como polvo, microbios, vapores químicos entre otros. Estos ambientes son muy usados en la industria de la manufactura y la investigación en un esfuerzo por mantener el objetivo principal libre de defectos y contaminantes **[22]**.

[9] Estructura del método de caja: Método usado para la especificación y diseño del software dentro de la metodología Cleanroom que consiste en expresar las aplicaciones como funciones matemáticas a fin de poder verificarlas formalmente **[22]**.

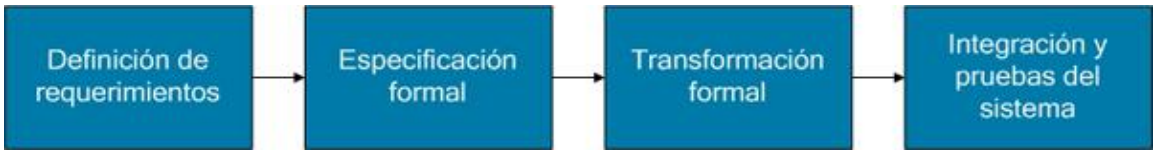


Figura 2.6 – Metodología de métodos formales **[22]**

Como se muestra en la figura anterior, el proceso es lineal. Primero se definen los requerimientos, después se genera una especificación formal[10], posteriormente se elabora una transformación formal y finalmente integramos y probamos el sistema. La parte central de esta metodología es que está basada completamente en especificaciones y verificaciones formales.

Los métodos formales podrían ser la metodología que nos lleve a un software sin errores, sin embargo su aplicación presenta varios problemas, algunos de ellos se enuncian a continuación:

- El desarrollo de código usando este método es muy caro y tardado.
- El personal necesita entrenamiento especializado.
- Resulta complicado usar métodos formales como método de comunicación con el cliente debido a la orientación técnica de la metodología.

2.7 Putos clave referente a cualquier metodología

- Los procesos del software incluyen las actividades involucradas en la producción y la evolución de un sistema de software. Todas estas actividades están representadas en una metodología específica de desarrollo de software.
- Las actividades que, por lo general, tienen en común todos los modelos son especificación, diseño, implantación, pruebas y mantenimiento.
- Las metodologías genéricas de procesos describen la organización de los procesos de software.
- Las metodologías iterativas describen el proceso de software como un ciclo de actividades.

[10] Especificación formal: Descripción que tiene como objetivo producir un conjunto de especificaciones del producto no ambiguas a fin de que los requerimientos, restricciones y supuestos del proyecto se definan precisamente por medio del uso de notación matemática **[109]**.

- Cada una de las metodologías están orientadas hacia las distintas necesidades de la organización que las implementa.
- No hay una solución que se adapte a todos los proyectos. Es necesario decidir cuál es la que más se acomoda al desarrollo en particular.

2.8 CIMS y Crystal Clear

El proyecto CIMS fue creado en un entorno peculiar con sus ventajas y desventajas. Por un lado el ambiente requería de una metodología de desarrollo que fuera extremadamente flexible, pues a pesar de que el modelo de negocio se tenía perfectamente claro, la implementación concreta carecía de una definición clara y concisa. Sin embargo, las limitaciones en cuestión de tiempo hacían indispensable que dicha flexibilidad fuera regulada y que el cliente fuera disciplinado debido a que se corría el riesgo de que se diera lo que comúnmente se conoce como *scope creep*[11].

Crystal Clear es la metodología elegida para llevar a cabo este proyecto debido a que provee justo la flexibilidad necesaria. Una de sus características principales es que no es una metodología que obliga a adoptarla por completo o en lo absoluto. Ahora bien, la cuestión es encontrar qué partes de Crystal Clear adaptaremos a las necesidades particulares del proyecto.

La adaptación de la metodología básicamente consistió en la realización de los ajustes necesarios para que ésta funcionara en un equipo de desarrollo de una sola persona. Dada la importancia del Personal Software Process (PSP) desarrollado por el Software Engineering Insitute de Carnegie Mellon, uno podría pensar que el problema del equipo de una persona se resuelve adoptando esta *metodología*, el problema es que el PSP no es una metodología; es un proceso. Las herramientas descritas por éste son sumamente útiles y de hecho se usan en otros estándares como el PMBOK[12], pero sigue haciendo falta adoptar una metodología.

[11] *Scope creep*: Término en inglés que se refiere a la situación en la que continuamente se añaden funciones extras a un producto. Estas funciones van más allá de las funciones básicas del producto y pueden causar complicaciones innecesarias **[125]**.
[12] PMBOK (*Project Management Body of Knowledge*): Estándar internacionalmente reconocido que provee los fundamentos de la administración de proyectos aplicadas a varios tipos de proyectos incluyendo los de Tecnologías de la Información **[134]**.

De los principios básicos de Crystal Clear, los sujetos a modificaciones fueron mejora reflexiva, comunicación *por ósmosis* y en menor medida seguridad personal. Justo estos tres son los más orientados a grupos. Los detalles de las variaciones se encuentran a continuación:

- Mejora reflexiva

  En un equipo plural, esta característica implica juntarse por lo menos una vez cada tres meses un par de horas para poder encontrar las virtudes y defectos del equipo entero en términos particulares. Por ejemplo, ¿qué es lo que te ha ayudado a trabajar más rápido/lento?, ¿dónde podemos mejorar?, ¿cuáles son nuestros hábitos de trabajo?

  Sin embargo, en un equipo singular resultaba bastante difícil llegar a la autocrítica constructiva pues no había punto de comparación o retroalimentación externa objetiva. La solución fue dedicar una fracción del tiempo de trabajo por día en el cálculo de métricas de eficiencia como la de valor ganado al mismo tiempo que se registraban los factores externos que podían influir en los resultados. Al término de cada semana reflexionaba sobre preguntas del estilo del párrafo anterior, pero conmigo mismo.

- Comunicación *por ósmosis*

  En un equipo de más de una persona, esta propiedad involucra el flujo de información pasivo hacia los miembros del equipo. Éstos escuchan lo que está sucediendo a su alrededor sin necesariamente estar participando de forma activa en una conversación. Lo anterior se logra por lo general teniendo al equipo entero en un mismo cuarto.

  En el caso de un equipo integrado por una sola persona, esta característica fue implementada concretamente mandando mensajes de correo electrónico periódicos sobre mis actividades al personal de Crédito Integral con el que levantaba los requerimientos. Lo anterior, haciendo énfasis especial en las cosas que tenía duda sin necesariamente estarles haciendo preguntas. Los destinatarios de los mensajes sabían que no era estrictamente necesario que supieran la respuesta a mis enunciados. Simplemente comentaban o respondían si es que consideraban que tenían algo importante que decir al respecto.

- Seguridad personal

  Este punto está enfocado en la confianza que tienen los miembros del equipo para poder expresar sus inquietudes, errores, deseos, entre otros aún cuando no es sencillo hacerlo. Por ejemplo, decirle a tu jefe que sub estimaste el proyecto por más del 50%.

  En mi equipo unitario, no hubo mucho cambio en este punto, sólo que la confianza que tuvo que ser cultivada fue entre los directivos de Crédito Integral y yo, así como entre mis sinodales/tutora y yo.

Capítulo 3

Gestión y desarrollo de proyectos de software

3.1 Definición de proyecto de software

Lo primero que tenemos que hacer para poder estudiar las características de un proyecto de desarrollo de software es definir proyecto de una manera genérica. Un proyecto es un plan específico para lograr ciertos objetivos **[25]**. Dicho plan consta de los siguientes puntos:

- Involucra tareas no rutinarias.
- Requiere planeación.
- Persigue objetivos específicos.
- Es finito en tiempo, alcance y costo.
- Involucra varias fases del mismo.

Uno de los aspectos más importantes de un proyecto es su tamaño pues la dificultad del mismo dependerá de dicha característica. A continuación se identifican las diferencias que existen entre un proyecto convencional y un proyecto de desarrollo de software.

Progreso invisible

El progreso al construir un objeto físico es siempre visible, esto no ocurre con el software, porque éste no se encuentra inherentemente incrustado en el espacio. Por lo tanto no tiene representación geométrica en la manera en la que la superficie terrestre tiene mapas, o los edificios tienen planos. Además los diagramas de software de una aplicación de tamaño moderado pueden llegar a ser demasiado numerosos, lo que dificulta las estimaciones y la definición de hitos una vez que el proyecto está siendo construido **[16]**.

Complejidad

Por cada unidad monetaria gastada, los proyectos de desarrollo de software son más complicados que un proyecto convencional debido a que no existen dos partes iguales. Si es que existen, convertimos las dos partes similares en una subrutina, abierta o

cerrada. Además los sistemas de software poseen varias órdenes de magnitud más estados[13] que la mayoría de las cosas que construyen los humanos **[16]**.

Flexibilidad

El software está sujeto a muchos cambios debido a que siempre se está adaptando a los procesos que sustenta. También lo están otros objetos producidos por el hombre como son los coches o las computadoras. Sin embargo, los productos construidos raramente cambian después de su manufactura; por lo general las nuevas funciones a incorporar en estos productos se incluyen en modelos posteriores. Esto no ocurre con los proyectos de software debido a que aún aquél que es exitoso está sujeto a cambios. Una vez que un producto de software resulta útil, la gente trata de usarlo en nuevas aplicaciones. Por otro lado el software con errores también es cambiado para corregir sus funcionamiento erróneo **[16]**.

3.2 Preámbulo al desarrollo de software

Un proyecto de software involucra muchas tareas que no necesariamente tienen que ver con la escritura de código. De hecho, existen proyectos de software en donde la aplicación de software es de tipo COTS[14]. Las tres actividades que sirven como preámbulo al desarrollo de software son:

Estudio de factibilidad

Es una investigación en donde decidimos si vale la pena llevar a cabo el proyecto o no. Se evalúa la relación costo–beneficio del proyecto en general y se toma una decisión.

Planeación

Se define la estrategia que vamos a seguir para poder llevar a cabo el proyecto. Es importante notar que en proyectos lo suficientemente grandes, sólo la primera etapa se detalla debido a que ésta sentará las bases para desarrollar las siguientes fases.

[13] Estado: Una configuración única de información en una máquina **[131]**
[14] COTS *(Commercial Off The Shelf)*: Objeto comercial del anaquel. Acrónimo que caracteriza a un artículo que fue desarrollado previamente y está listo para venderse o rentarse sin necesidad de mantenimiento a través de su ciclo de vida **[26]**.

Ejecución

El proyecto se ejecuta dependiendo del tipo del mismo y de la metodología que hayamos escogido para llevarlo a cabo. En el capítulo dos se describen varias metodologías de desarrollo.

3.3 Tipos de proyectos de software

Existen varias formas de categorizar los proyectos de software. Es muy importante hacerlo de forma correcta ya que las herramientas específicas que tengamos disponibles dependerán de la categoría en la que nuestro proyecto se encuentre. Por ejemplo el método SSADM[15], creado para el análisis y diseño de sistemas, es exclusivo para sistemas de información **[28]**.

Categorización basada en objeto de interacción

Los proyectos se pueden clasificar en dos partes dependiendo de su objeto de interacción. Estas dos partes son sistemas de información y sistemas integrados. La principal diferencia entre ellas es que en los primeros se interactúa con personas y los segundos con máquinas. Un ejemplo de un sistema incrustado es un sistema operativo ya que su principal objetivo es interactuar con los programas que se ejecutan sobre él a fin de administrar los recursos de una computadora. Sin embargo el sistema operativo también interactúa con el usuario.

Categorización basada en intensión

Un proyecto puede también estar categorizado tomando en cuenta el fin que éste persigue, de esta forma podemos definir dos categorizaciones mútuamente excluyentes:

1. Proyectos que pretendan cumplir con ciertos objetivos sin realmente desarrollar un producto.
2. Proyectos cuyo fin es desarrollar un producto específico.

Un ejemplo de la primera categorización podría ser la automatización del manejo de las tareas relacionadas con los recursos humanos. En este caso, dado nuestro objetivo,

[15] SSADM *(Structured Systems Analysis and Design Method)*: Método estructurado de análisis y diseño de sistemas **[27]**.

podríamos simplemente adquirir un producto COTS e implementarlo en nuestra empresa y no necesariamente desarrollar uno propio. Después de todo nuestro objetivo era automatizar los procesos de recursos humanos y mientras lo estemos cumpliendo, no será forzoso desarrollar una aplicación nueva. Por otro lado también podemos estar involucrados en la petición de un cliente que consista en desarrollar un nuevo sistema. En este caso sí estaríamos tratando con un proyecto que pertenece a la segunda categorización, donde los requerimientos del proyecto correrían a cargo del cliente.

3.4 Posibles problemas relacionados con proyectos de software

Las tareas de administración de un proyecto de software están relacionadas con la coordinación de los diferentes elementos que integran al equipo de trabajo en búsqueda del éxito del proyecto. Los problemas se suscitan cuando se presentan agentes que amenazan el cumplimiento del plan original. Como se observa a continuación, las tareas más importantes de un administrador de proyectos están relacionadas con reducir los problemas existentes y/o probables. Las responsabilidades más comúnes de un líder de proyectos se enlistan a continuación:

- Manejar y ajustarse a las fechas límite de entrega.
- Adaptarse a recursos limitados.
- Asegurar la comunicación eficiente entre equipos y dentro de los mismos.
- Motivar al equipo y asegurar su compromiso.
- Establecer metas.
- Gestionar efectivamente cambios en el proyecto.
- Negociar el plan del proyecto con las diferentes partes del equipo.
- Motivar a la gerencia y asegurar su compromiso.
- Resolver conflictos.
- Administrar al personal externo.

  Fuente datos: Encuesta del Project Management Journal **[29]**.

Por otro lado, al hacer una encuesta a los líderes de proyectos respecto a los problemas más comúnes que enfrentaban, los resultados fueron los siguientes:

- Cálculos erróneos relacionados con las fechas límite y la cantidad de recursos requeridos.
- Falta de estándares de calidad y métricas.
- Falta de orientación relacionada con decisiones de la organización.
- Falta de medios para hacer el progreso visible.
- Mala definición de roles.
- Criterios de éxito incorrectos.

  Fuente de datos: Encuesta publicada en IEEE Transactions on Software Engineering **[30]**.

Esta encuesta refleja el punto de vista de los líderes de proyectos. Sin embargo, en aras de la objetividad, también hace falta tomar en cuenta el punto de vista de los demás miembros del equipo de trabajo. Los resultados de una encuesta similar, pero hecha a miembros de un equipo de desarrollo que no tienen que ver con gestión fueron los siguientes:

- Especificaciones de trabajo incorrectas.
- Dirección ignorante respecto tecnologías de la información.
- Ignorancia respecto al área de aplicación de ciertas tecnologías.
- Falta de estándares.
- Falta de documentación actualizada.
- Actividades inconclusas de las cuales dependen otras.
- Falta o mala comunicación entre usuarios y técnicos.
- Falta de dedicación.
- Falta de capacitación técnica del equipo.
- Cambios en las bases y/o especificaciones del proyecto.
- Cambios en el ambiente de desarrollo.
- Presiones respecto a las fechas límite.
- Falta de control de calidad.
- Gestión a distancia.

  Fuente de datos: Encuesta publicada en IEEE Transactions on Software Engineering **[30]**.

Cabe notar que muchos de los problemas denotados con anterioridad tienen como raíz común la falta de comunicación. El alto grado de especialización en TI es señalado como posible culpable muchas veces en encuestas de este tipo.

3.5 Estimaciones de un proyecto de software

Cuando hablamos de un proyecto de software exitoso, por lo general estaremos hablando de uno que fue entregado "bien y a tiempo". También es indispensable que nuestro proyecto se haya desarrollado con los recursos económicos planeados. Tomando esto en cuenta, es fácil ver que las estimaciones sobre los tiempos y los recursos juegan un papel clave en el aseguramiento del mismo.

Las estimaciones de un proyecto de software son complicadas. Muchas de las dificultades son inherentes al desarrollo de software. Además, el desarrollo de software está relacionado con personas las cuales no pueden ser tratadas de forma mecánica. Otros riesgos que enfrenta la buena estimación de recursos en proyectos de software se encuentran a continuación:

Proyectos virtualmente nuevos

En proyectos de ingeniería no relacionados con el software, es muy común que un proyecto a desarrollar sea muy parecido a uno desarrollado previamente pero con un cliente distinto y en otro lugar. Por lo tanto, resultaría bastante tentador usar parte de las estimaciones de dichos proyectos anteriores. En el mundo del software, siempre existen ciertos factores que hacen único a nuestro proyecto; lo que llena a nuestras estimaciones de incertidumbre y duda **[28]**.

Tecnología cambiante

La tecnología usada para desarrollar proyectos de software hoy es distinta a la usada hace 20 años. Por ejemplo en un banco muchos de los sistemas de legacía están desarrollados en Cobol y hoy en día pueden estar desarrollando sistemas con tecnologías nuevas como Java ó Flex.

Falta de homogeneidad en experiencia

Las estimaciones de recursos suelen estar basadas en proyectos anteriores. Esto se expondrá más adelante en este capítulo. Sin embargo por una razón u otra es común ver que las empresas no comparten la información acerca de sus proyectos anteriores a sus empleados **[28]**.

Partiendo del hecho de que se estiman los proyectos actuales dados los anteriores, resulta interesante analizar la siguiente tabla (3.1) que relaciona esfuerzo aplicado con fases de desarrollo y eficiencia. Los datos de ésta fueron obtenidos en una investigación empírica publicada en el Journal of Systems and Software de 1985 involucrando proyectos de software de tamaño similar, basados en el número de líneas de código empleadas y el tiempo dedicado al desarrollo de los proyectos **[32]**. Los porcentajes son respecto al total de meses de trabajo. Por ejemplo, para el proyecto "A", 23% = 3.9/16.7.

	Diseño		Código		Pruebas		Total		Eficiencia
Proyecto	**mt**	**%**	**mt**	**%**	**mt**	**%**	**mt^{tot}**	**lc^{tot}**	**lc^{tot}/mt^{tot}**
A	3.9	23	5.3	32	7.4	44	16.7	6050	362.28
B	2.7	12	13.4	59	6.5	26	22.6	8363	370.04
C	3.5	11	26.8	83	1.9	6	32.2	13334	414.1
D	0.8	21	2.4	62	0.7	18	3.9	5942	1523.59
E	1.8	10	7.7	44	7.8	45	17.3	3315	191.62
F	19	28	29.7	44	19	28	67.7	38988	575.89
G	2.1	21	1.4	74	0.5	5	10.1	38614	3823.17
H	1.3	7	12.7	66	5.3	27	19.3	12762	661.24
I	8.5	14	22.7	38	28.2	47	59.5	26500	445.38

Figura 3.1 – Meses de trabajo (mt) en relación a líneas de código (lc) y eficiencia.

Analizando la tabla 3.1, nos podemos dar cuenta de que algunos proyectos se salen totalmente del promedio en términos de eficiencia. Por ejemplo, el proyecto "G" mostró una eficiencia de aproximadamente diez veces más que el proyecto "A". Esta tabla también refuerza la idea de que los proyectos de software poseen un carácter casi único con respecto a sus antecesores. Sin embargo la herramienta más poderosa en estimaciones es la estadística.

3.5.1 Subestimaciones y sobrestimaciones

Cualquier líder de proyecto debe estar consciente de que las estimaciones del proyecto mismas, si son expuestas al equipo de trabajo, tendrán influencia en los tiempos en los que el proyecto deberá terminarse. Una sobrestimación causaría que el proyecto sea finalizado en más tiempo del necesario. Esto puede ser explicado por las siguientes leyes:

Ley de Parkinson

El trabajo se expande hasta abarcar todo el tiempo disponible. Dicho de otra manera, una fecha límite muy holgada hará que el equipo se confíe y trabaje más lento. **[33]**

Ley de Brooks

El esfuerzo requerido para implementar un proyecto crece rápidamente junto con la cantidad de gente asignada al mismo. El trabajo extra necesitado en comunicación y coordinación del equipo crece al mismo tiempo que se agrega más gente al proyecto. Esto refuerza el hecho de que agregar más gente a un proyecto que ya presenta resultados tardíos sólo lo retrasa más. **[16]**

Lo anterior nos podría indicar que la moraleja es simplemente tomar la cota inferior de las estimaciones de esfuerzo. El problema con lo anterior es que la calidad se vería afectada debido a las presiones de tiempo. Lo más dañino es que los problemas que traen los proyectos de calidad sub-estándar sólo son visibles en las fases posteriores al desarrollo. Además la cantidad de re-trabajo requerida en estas fases puede tener un efecto dramático en el éxito del proyecto; lo que hace indispensable contar con modelos de estimación confiables que serán descritos a continuación.

3.5.2 Técnicas de estimación de trabajo de software

Las principales formas de estimar el esfuerzo en proyectos de software se enlistan a continuación **[34]**:

- Modelos algorítmicos – Usan objetos que representan las características del sistema a desarrollar en la plataforma elegida.
- Juicio experto – Básicamente implica que alguien con experiencia en el área aconseje al líder de proyecto en la materia.
- Analogía – Sistemas ya desarrollados se pueden usar como base para estimar el esfuerzo en el sistema a desarrollar.
- Parkinson – Usa todo el esfuerzo disponible de un equipo y lo trata como medida de estimación.
- Precio más bajo para ganar – La estimación es simplemente suficientemente baja como para ganar un contrato o licitación.

- De arriba abajo – Primero se hace un estimado para el proyecto en general y después se estudian los componentes individualmente.
- De abajo hacia arriba – Primero se estiman los esfuerzos requeridos para los componentes individuales y después simplemente se van sumando.

Es muy importante tomar en cuenta que un proyecto de desarrollo de software requiere mucho más que simplemente escribir código. Existen muchos productos de trabajo que son producidos junto con el proceso de desarrollo del software como lo son documentos, agendas, planes, reportes de errores, entre otros. Ningún producto es más importante que el otro; por ejemplo los errores de un producto tendrán impacto en los demás. Es responsabilidad del líder de proyecto coordinar a los miembros del equipo para asegurarse que absolutamente todos los productos desarrollados estén correctos a fin de que el proyecto en general sea exitoso **[125]**.

Capítulo 4

Riesgos asociados a los sistemas de crédito

Introducción al riesgo

Definiciones de riesgo podemos encontrar muchas. El significado de la palabra varía dependiendo del contexto en el que la estemos usando. Sin embargo, todas las definiciones de riesgo analizadas coinciden en lo siguiente: los riesgos son peligros potenciales **[41]**. Todos nos enfrentamos a distintos riesgos todos los días, aunque también hemos desarrollado habilidades que nos permiten enfrentarlos o reducir la probabilidad de que ocurran. No obstante, los riesgos siempre estarán presentes en nuestras actividades y está en nosotros la decisión de realizar o no la actividad que conlleva éstos. El problema es evitar el riesgo, lo que también significa privarnos de la actividad asociada. Actividades cotidianas como manejar conlleva el riesgo de un accidente automovilístico, cursar una carrera implica el riesgo de no cumplir las metas personales, hasta salir a la calle nos pone en peligro de que nos asalten. Una posible solución es decidir cuáles riesgos tomar y cuáles no **[69]**.

Sin embargo hasta ahora sólo hemos estado hablando de riesgos relacionados a personas físicas. Las empresas también están sujetas a éstos. Los procesos dentro de cualquier organización siempre están sujetos a dificultades y tropiezos de distintos tipos, los cuales juntos pueden ser la causa de diversos problemas. Éstos pueden provocarnos graves pérdidas y situaciones desfavorables en caso de no ser detectados y atendidos con anticipación para prevenirlos. Cabe mencionar que la mayoría de estas situaciones se pueden predecir y/o mitigar evitando así que los problemas se propaguen, aumenten o en su defecto se creen nuevos. Es por eso que con la ayuda de la probabilidad y la estadística, podemos tener una idea bastante clara de qué tanto y con qué frecuencia nos afectan los posibles riesgos [16] detectados en alguna organización.

El colapso del banco Barings en 1995 y las pérdidas de más de $691.2 (millones de dólares estadounidenses) por parte de Allied Irish Banks en 2002 son ejemplos de las

[16] Riesgo: Se refiere a un concepto que especifica la probabilidad de ocurrir de ciertos eventos adversos a cierto objetivo u organización **[41]**

consecuencias que puede ocasionar el riesgo financiero no controlado, a causa de transacciones no autorizadas realizadas por empleados **[70]**.

4.1 Riesgos relacionados al desarrollo de software

Los proyectos de desarrollo de software son inherentemente complicados. Esto es porque existen muchos factores que tienen que ser coordinados de manera casi perfecta a fin de que éstos sean exitosos. Dichos factores son a la vez fuentes de riesgo, las cuales deben ser controladas.

La conciencia del riesgo en proyectos de desarrollo de software apareció de varias formas. Al principio surgió simplemente como la demanda del mercado por productos que estuvieran a prueba de fallos. Los desarrolladores entonces buscaron diseños más robustos que permitieran mitigar dichos fallos. Al hacer esto estaban reduciendo lo que hoy llamamos "riesgos del producto". Fue así que la cultura del riesgo fue siendo adoptada por el gobierno de TI. Sin embargo en otras disciplinas fue un poco distinto debido a que muchas veces se pensaba que el controlar las fallas era aceptar que existían en la dirección de los proyectos desde un principio **[14]**.

El riesgo tenía distintas acepciones dependiendo de la perspectiva con la que se viera. Para la dirección en finanzas, el riesgo significaba responder la pregunta de ¿qué tan rentable es una inversión? El riesgo del crédito era estudiado, definido y medido puntualmente dentro de las instituciones financieras. La variación en la retribución de la inversión era una medida de riesgo.

Administrar el riesgo se fue convirtiendo, en distintas disciplinas, en una parte integral para la administración de proyectos. Los líderes de proyectos perciben el riesgo de una forma mucho más clara que cualquier otra persona, ya que su tarea incluye monitorear el mismo. Existen proyectos que son claramente riesgosos. Por ejemplo construir una presa en la selva es claramente riesgoso debido a lo hostil del ambiente de trabajo. Construir ductos subterráneos de petróleo es claramente riesgoso también debido a lo hostil del ambiente de trabajo. En estos y muchos casos más el proyecto mismo se volvía sinónimo de riesgo.

Todas estas influencias contribuyeron a que el manejo del riesgo en la administración de proyectos de TI fuera indispensable. El manejo del riesgo es una parte esencial de la vida de un proyecto y se ha convertido en algo indispensable para la supervivencia del mismo. Pensar en identificar riesgos y manejarlos ya no es visto como algo negativo, pues está demostrado según estadísticas de desempeño de proyectos que la gestión de riesgo permite desarrollar proyectos mejores y con menos recursos **[72]**.

4.2 Características de los riesgos y su categorización

Un problema o dificultad que podemos enfrentar al momento de analizar las características de los riesgos es que los atributos analizados pueden depender del enfoque que estemos utilizando. Por ejemplo si nuestra perspectiva es la económica se pueden clasificar los riesgos dependiendo del impacto económico que se tendría en caso de que el deudor dejara de pagar. Por otro lado si la perspectiva es la de planeación, podríamos pensar en clasificar los riesgos dependiendo del tiempo o recursos económicos que podríamos perder si ocurrieran.

El número de atributos a analizar en la taxonomía de los riesgos debe de reducirse en aras de la simplicidad, de lo contrario dificultaríamos su clasificación y gestión. De hecho es preciso escogerlos dependiendo de nuestro enfoque hacia el control y mitigación de los mismos. A continuación se enunciarán y describirán ciertas perspectivas de atributos de los riesgos.

Origen

Teniendo el origen del riesgo como perspectiva, tenemos dos categorías:

- Internos: Como su nombre lo indica, los riesgos internos provienen desde adentro de la organización que gestiona el proyecto. Debido a esto, los riesgos internos son más fáciles de identificar, controlar, medir y revisar que los externos, ya que tenemos mucho más control sobre ellos.
- Externos: Sobre estos riesgos tenemos poco o nulo control pues son originados por entes ajenos a los desarrolladores. Por ejemplo el retraso del proyecto debido a una epidemia de influenza. Lo mejor que podemos hacer es estar preparados por medio de un plan de contingencia y analizar puntualmente la probabilidad de ocurrencia junto con el impacto de estos riesgos.

Impacto

El impacto que tendría el riesgo si se convirtiera en realidad es extremadamente importante, porque es justo lo que queremos minimizar en caso de que ocurran los problemas potenciales. Las clases o categorías que podemos derivar de este punto de vista son:

- Catastróficos: Riesgos que podrían causar daño máximo a nuestro proyecto en caso de que ocurrieran. Éstos tienen que ser tratados de forma muy especial. De hecho se supone que lo mejor que podemos hacer es suponer que van a ocurrir y no tomar el riesgo. En el caso en el que sí decidamos tomar riesgos de este tipo, modelarlos de la mejor manera posible y tratar de estar preparados antes de que ocurran los daños.
- Limitantes: Estos riesgos son aquellos que estamos muy seguros que van a ocurrir. De hecho estos no se llaman riesgos sino limitantes debido a que estamos seguros que van a ocurrir. Tratar de mitigar sus consecuencias es lo que se recomienda pues no hay muchas opciones.
- Nominales: Éstos no poseen una clasificación particular. No son ni muy probables de que ocurran, ni implican un alto impacto en el caso en el que sí llegasen a suceder.
- Triviales: Las consecuencias de este tipo de riesgos son insignificantes. El hecho de que ocurran no afectan ni el tiempo, ni el alcance, ni el costo del proyecto. Debido a lo anterior son considerados como insignificantes.

Las recomendaciones sobre qué hacer respecto a los distintos tipos de riesgo basados en los criterios de clasificación especificados ya han sido descritas.

Para poder ejemplificar la categorización anterior, vamos a presentar los riesgos que se encontraron al desarrollar el CIMS. Tomando en cuenta el hecho de que estamos desarrollando un sistema de crédito para una SOFOM, resulta interesante analizar las consecuencias o el impacto que tendrían los riesgos en el caso de que se hicieran realidad. A continuación se muestra la tabla 4.1 con todos los riesgos principales junto con su impacto y probabilidad de ocurrencia.

Número	Riesgo	Impacto (1 al 10 teniendo mayor impacto el 10)	Probabilidad de ocurrencia (0 a 100%)	Clasificación
1	Fuga de personal involucrado	7	20%	Nominal
2	Ausencias de personal involucrado	3	60%	Nominal
3	Pérdida de información	10	20%	Catastrófico
4	Cambios en alcance	5	40%	Nominal
5	Fallas en el suministro eléctrico	1	10%	Trivial

Tabla 4.1 Características de riesgos del CIMS

4.3 Taxonomía del SEI (Software Engineering Institute)

El SEI de Carnegie Mellon propuso una clasificación de riesgos, en la cual la estructura principal está organizada en clases, subclases y elementos. A continuación se enuncian las principales estructuras de esta clasificación **[14]**.

- Riesgos de ingeniería del producto **[14]**.
  - Requerimientos.
    - Estabilidad.
    - Completez.
    - Claridad.
    - Validez.
    - Factibilidad.
    - Precendencia.
    - Escala.
  - Diseño.
    - Funcionalidad.
    - Dificultad.
    - Interfaces.
    - Rendimiento.
    - Capacidad de ser probado.

        - Restricciones de hardware.
        - Software difícil de desarrollar.
    - Código y pruebas unitarias.
        - Factibilidad.
        - Pruebas.
        - Programación ó implementación.
    - Integraciones y pruebas.
        - Ambiente.
        - Producto.
        - Sistema.
    - Especialidades de ingeniería.
        - Mantenibilidad.
        - Confiabilidad.
        - Seguridad.
        - Factores humanos.
        - Especificaciones.
- Entorno de desarrollo **[14]**.
    - Proceso de desarrollo.
        - Formalidad.
        - Control de procesos.
        - Familiaridad.
        - Control de producto.
    - Sistema de desarrollo.
        - Capacidad.
        - Usabilidad.
        - Confiabilidad.
        - Soporte de sistema.
        - Capacidad de ser entregado.
        - Familiaridad.
    - Proceso de gestión.
        - Planeación.
        - Organización de proyectos.
        - Experiencia en la administración.
        - Interfaz del programa.
    - Métodos de gestión.
        - Monitoreo.

        - Recursos humanos.
        - Aseguramiento de calidad.
        - Gestión de la configuración.
    - Ambiente de trabajo.
        - Actitud de calidad.
        - Cooperación.
        - Comunicación.
        - Moral.
- Restricciones del programa **[14]**.
    - Recursos.
        - Agenda o calendarización.
        - Personal.
        - Presupuesto.
        - Facilidades.
    - Contrato.
        - Tipo de contrato.
        - Restricciones.
        - Dependencias.
    - Interfaces.
        - Clientes.
        - Contratistas.
        - Sub – contratistas.
        - Contratista principal.
        - Dirección corporativa.
        - Vendedores.
        - Política.

4.4 Riesgo crédito

Otro de los riesgos que se deben estimar y tomar en el giro de las SOFOMes es el riesgo crédito. A grandes rasgos, el riesgo crédito es el riesgo de pérdida económica originada por la falta de pago de una deuda por parte del deudor de la misma. El deudor pasa por las siguientes fases antes de suspender por completo los pagos de la deuda: retraso en los pagos, reestructuración de la deuda con términos y condiciones más atractivas para el deudor, y finalmente caer en la bancarrota. Para evitar que esto

pase, las instituciones de crédito tiene varias herramientas disponibles para estimar el riesgo crédito de los deudores, mismas que serán expuestas más adelante.

Los métodos de estimación y gestión del riesgo crédito dependen principalmente del tipo de crédito otorgado y del tipo de deudor. Respecto al tipo de crédito podemos afirmar que los créditos revolventes, como los de las tarjetas de crédito, son más fáciles de controlar porque la línea de crédito cuenta con un límite de crédito previamente establecido que puede ser aumentado o disminuido dependiendo de la disciplina de los pagos de los deudores. Por otro lado los créditos personales, como los que otorga Crédito Integral, son más difíciles de gestionar ya que una vez transferido el dinero al deudor no queda otra opción que esperar a que los pagos se efectúen **[49]**.

Sin embargo, las instituciones sí poseen métodos de estimación del riesgo de los deudores antes de otorgar cualquier crédito. El tipo de deudor es un factor muy importante a tomar en cuenta, ya que éste dicta las herramientas de estimación que tienen las instituciones de créditos disponibles. Por ejemplo en el caso de deudores tipo personas físicas, las instituciones de crédito pueden consultar en México el Buró de Crédito **[74]**. Ésta es una empresa privada facultada para proveer información histórica acerca de nuestro desempeño como deudores. Esto incluye detalles sobre la puntualidad de los pagos efectuados como deudores, así como el monto histórico de nuestras deudas vigentes y liquidadas. Respecto a créditos personales, de auto o hipotecarios, la regla principal en algunas SOFOMES para decidir si el crédito es otorgado o no es muy sencilla: el monto de los pagos mensuales que debemos efectuar para liquidar créditos vigentes no debe ser mayor a tres veces los ingresos del deudor **[75]**.

Respecto a los deudores tipo personas morales también está disponible el Buró de Crédito, sin embargo los créditos otorgados a personas morales son generalmente más riesgosos porque involucran montos más altos. Dentro de Crédito Integral, a las personas morales se les requiere que endosen bienes cuyo valor sea tres veces superior al monto de la deuda. Esto es además de la condición de contar con un buen historial crediticio dentro del Buró de Crédito. No obstante, los otorgadores de créditos tienen recursos más sofisticados para conocer el riesgo crédito de los deudores potenciales. Empresas privadas como Standard & Poor, Fitch Ratings y Moody's proporcionan información acerca del riesgo crédito de empresas. Estas compañías

también califican fondos de inversión y realizan tareas diversas de consultoría financiera **[76]**.

Capítulo 5

Sistema CIMS (*Crédito Integral Management Suite*)

5.1 Antecedentes

La empresa Crédito Integral S.A. de C.V., SOFOM [17] E.N.R. [18] tiene como negocio principal otorgar créditos a clientes según la viabilidad del negocio. Esto es a través del análisis de la solicitud de crédito que los clientes presenten cubriendo una serie de requisitos previamente establecidos.

Actualmente, la empresa lleva el registro, control, seguimiento y facturación de los créditos que autoriza por medio de la sub-utilización de las herramientas básicas de Microsoft Office.[19] Lo anterior provoca limitantes evidentes en las distintas áreas de negocio de la empresa al elevar la posibilidad de error dramáticamente en los procesos cotidianos y disminuir la visión en las oportunidades de negocio.

5.2 Descripción

El sistema CIMS es una aplicación completa de escritorio diseñada específicamente para la administración de una SOFOM. El software fue diseñado para poder expandirse a medida que la empresa amplíe la gama de servicios que ofrece. Actualmente sólo cubre la parte de otorgamiento de créditos, sin embargo se tiene pensado que a mediano plazo se incorporen las funciones de factoraje y arrendamiento. Lo anterior, reutilizando la estructura de la base de datos de clientes ya sea de personas físicas como de morales.

Dada la situación dentro de Crédito Integral descrita en el apartado 5.1 aunado a la prosperidad de la empresa, fue que el consejo administrativo decidió diseñar un software que le permitiera tomar control de la empresa. El creciente número de préstamos y la forma manual en la que éstos eran manejados provocaba un gran número de errores "de dedo" que se propagaban con pérdidas monetarias casi

[17] SOFOM: Sociedad financiera de Objeto Múltiple. Es una entidad financiera que, por medio de la captación de recursos externos, otorga créditos a personas morales y físicas. **[77]**
[18] E.N.R. Entidad no regulada. Es un calificativo que aplica a SOFOMes que no tienen accionistas en común con una institución de crédito **[77]**
[19] Microsoft Office: Es un conjunto de aplicaciones de escritorio interrelacionadas, servidores y servicios desarrollados para los sistemas operativos Windows y Mac OS X. **[78]**

catastróficas. Conscientes del grave problema que enfrentaba el funcionamiento de la SOFOM, la empresa empezó por estandarizar sus procesos internos. Resultaba impresionante ver como procesos aparentemente sencillos, como el llenado de formularios, presentaban múltiples inconsistencias y diversas variantes que obstaculizaban el análisis de los clientes. Una amplia lista de irregularidades y errores, podría ser llenada tan sólo con los errores humanos en cálculos de intereses y diversos montos. Lo anterior afectaba la credibilidad de la empresa ante el cliente al mismo tiempo que nublaba las oportunidades de negocio.

El desarrollo del sistema beneficia a toda persona física y moral relacionada con la empresa. Esto incluye a las siguientes figuras:

Accionistas

Los accionistas son probablemente el grupo de personas más beneficiado debido a su estrecha relación con la SOFOM. Con la ayuda de CIMS, ellos pueden tener mucho mayor control de la forma en la que es manejada su inversión. La rendición de cuentas por parte de la administración de la empresa hacia los accionistas se puede hacer por primera vez de manera transparente y puntual. Esto acrecenta enormemente la confianza entre los accionistas mismos y promueve directamente la entrada de mayor capital hacia la empresa. Por otra parte, la alta disponibilidad de capital líquido dentro de la empresa posibilita la colocación de mayores recursos ampliando la actividad de la SOFOM y la rentabilidad de la empresa en general. **[79]**

Directivos

Los directivos también son un grupo de personas altamente beneficiados por la implementación del CIMS. Ellos gozan de múltiples ganancias al simplemente poder hacer mejor su trabajo. Los directivos se beneficiaron de la implementación del sistema al poder recibir información consistente y confiable relativa al funcionamiento de la institución. Tareas comúnes de la dirección como son el análisis de información y la toma de decisiones cruciales mejoraron dramáticamente debido a que anteriormente los reportes ejecutivos tardaban hasta una semana en ser generados manualmente. **[80]** Hoy, gracias a la implementación del CIMS, dichos reportes tardan tan sólo un par de segundos. Como resultado de la alta disponibilidad de información confiable, las decisiones de alto nivel son tomadas con certeza y de manera asertiva.

Clientes

Este grupo de personas, también integrantes de Crédito Integral, son la parte medular de cualquier negocio **[80]**. La reducción de errores en los procesos mejora la salud de la empresa. Esto permite tener un costo de operación más bajo, consecuentemente más clientes y esto finalmente se traduce en mejores servicios a menor costo **[80]**. Las ventajas mencionadas con anterioridad son sólo las ventajas directas. Analizando las ventajas indirectas, es fácil ver que los créditos a menor costo se traducen en una gran ventaja: más dinero a menos precio. No es sorpresa que una de las mejores formas de reactivar la economía sea bajar las tasas de interés, lo cual propicia directamente que las empresas tengan más dinero para invertir y mayores oportunidades para generar fuentes de empleo. Al entender estos simples conceptos es que nos preguntamos por qué en la actual desaceleración económica mundial Estados Unidos de América baja a niveles ridículos sus tasas de interés y en México no **[81]**. Aunque el pasado 19 de junio de 2009 el Banco de México anunció que bajaría su tasa de interés a 4.75% **[82]**, es una medida tardía.

Personal administrativo de baja jerarquía

El personal no directivo que trabaja para Crédito Integral es quien finalmente realizaba a mano todos los procesos de bajo nivel dentro de la empresa. Ellos, como humanos, son propensos a cometer errores. El problema aquí es que a ningún cuentahabiente le gustaría que el error se cometiera en su cuenta, ya que por alguna razón las empresas tienden a equivocarse a favor de ellas mismas y no a favor del cliente. **[83]** Además debemos de recordar que estamos lidiando con una empresa que otorga créditos y particularmente en este giro los errores de dedo son catastróficos. Los procesos dentro de la empresa dependían de entidades altamente propensas a errores. El resultado era el esperado: una tasa de error considerable y personal sufriendo por constates castigos derivados de éstos. El impacto después de implementar el CIMS fue que el personal administrativo de esta naturaleza incrementó su productividad al hacer uso del sistema por medio de una computadora. La dirección mejoró su satisfacción respecto a su personal y la empresa en general obtuvo un clima organizacional más saludable.

CNBV[20]

Esta institución gubernamental es básicamente la autoridad en el mundo de las finanzas. En un país plagado de narcotraficantes y demás organizaciones delictivas como México, no nos sorprende que el lavado de dinero sea uno de los delitos más comúnes. **[84]** Una de las tareas de la CNBV es rastrear los recursos económicos del crimen organizado por medio de las instituciones financieras. La implementación del CIMS ayuda a la CNBV a realizar dicha tarea al contar con información acumulada acerca de los clientes de Crédito Integral. Esta información después es analizada en busca de patrones que indiquen o sugieran el lavado de dinero. **[85]**

De una forma u otra todas las entidades relacionadas con Crédito Integral SOFOM E.N.R. resultaron beneficiadas. Tanto clientes como accionistas y personal en cualquier empresa resultan favorecidos de su buen funcionamiento.

5.3 Objetivos principales

CIMS fue creado con un objetivo en mente: automatizar y estandarizar los procesos dentro de Crédito Integral. Éstos se encuentran contemplados dentro del análisis de requerimientos (capítulo 6). Obviamente este objetivo es algo ambiguo y las juntas que dieron como resultado la creación de CIMS fueron mucho más descriptivas, sin embargo la idea de desarrollar un sistema de cómputo siempre estuvo presente en las mentes de los directivos de la empresa. El listado de los objetivos principales se encuentran en un diagrama que ilustra su jerarquía y se pueden observar en la figura 5.1. Los detalles se encuentran posteriormente.

[20] CNBV: Comisión Nacional Bancaria de Valores. Organismo público federal encargado de regular y supervisar el funcionamiento de las instituciones financieras. **[86]**

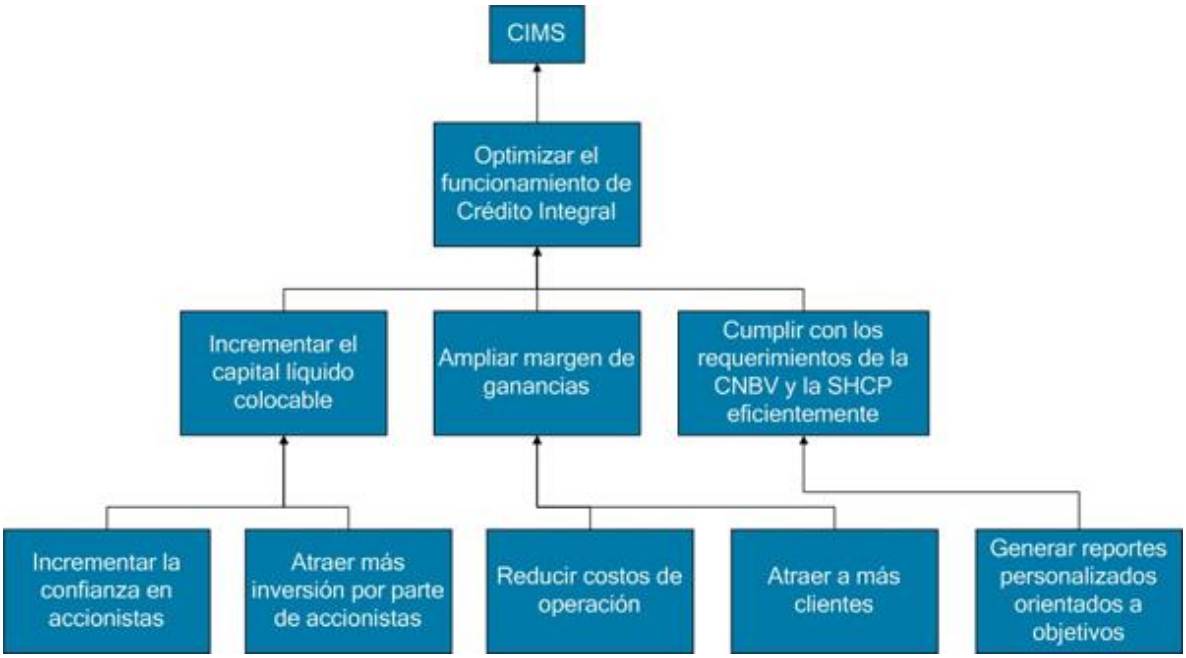


Figura 5.1 – Objetivos principales del desarrollo del CIMS

Todos los objetivos de la creación del CIMS giran alrededor de la optimización del funcionamiento de la SOFOM. Sin embargo, los podemos ir desglosando en busca de metas un poco más concretas y realizables. La optimización del funcionamiento de Crédito Integral se pudo hacer realidad sólo a través del mejoramiento de las distintas áreas de negocio y/o responsabilidades de la empresa, esto incluye:

Incrementar el capital líquido a colocar

Crédito Integral requiere de un constante flujo de recursos económicos para poder colocar ese dinero en créditos para sus clientes. Dichos fondos provienen principalmente de recursos de los accionistas. De estos últimos hay dos tipos: los nuevos y los existentes. Los accionistas existentes requieren de ciertos informes que les aseguren que su dinero estará en buenas manos. Por otro lado, la captación de nuevos accionistas se puede lograr a través del contagio de la confianza de los accionistas existentes. Todo esto se resume en un sólo propósito: transparencia. La creación de reportes ejecutivos confiables y a tiempo, es la clave para poder lograr dicha meta. El desarrollo de CIMS facilitó notoriamente la creación de los reportes ejecutivos existentes al mismo tiempo que generó nuevas formas de visualización de los datos que permitieran el análisis de los mismos. Además, contribuyó al desarrollo de nuevos reportes que incrementaron la confianza en la empresa por parte de los accionistas. Toda esta confianza y transparencia hacen de Crédito Integral el mejor

lugar para invertir ya que los accionistas gozan de una tasa de interés de Cetes por dos. Esto implica un muy atractivo rendimiento, tomando en cuenta que la inversión más fructuosa en el mercado mexicano dentro de las SOFOMes reguladas (bancos) sólo da una tasa de Cetes por uno. La inversión antes mencionada es Inbursa CT. **[87]**

Ampliar margen de ganancias

El margen de ganancias de una empresa es básicamente lo que ésta obtiene después de realizar sus ventas y después de solventar sus gastos de operación. **[80]** En el caso de Crédito Integral, esto significa reducir costos de operación al mismo tiempo que se amplía la cantidad de recursos en créditos otorgados. Al implementar el CIMS, se logró una reducción de costos de operación de al menos 20%, la cual se puede constatar dentro de los reportes ejecutivos mensuales de Crédito Integral. Esto fue posible debido a que fue posible continuar con la misma cantidad de trabajadores asignados a realizar el cálculo de intereses y cifras a reportar al incrementarse el número de créditos otorgados. Por otro lado, un mayor margen de ganancias permite invertir en elementos que incrementen la cartera de clientes como lo es la publicidad. Además, se ha mejorado el buen funcionamiento de la empresa, lo cual fomenta de manera directa la lealtad en los clientes existentes.

Cubrir los requerimientos de la CNBV y la SHCP[21] eficientemente

La CNBV, como fue mencionado con anterioridad, es quien le exige rendición de cuentas a las instituciones financieras. También se encarga, en coordinación con la PGR[22], de investigar los recursos financieros de presuntos lavadores de dinero. Para poder cumplir con este requerimiento, el personal de Crédito Integral analizaba a cada uno de sus clientes en busca de este tipo de conductas delictivas. Dado al tamaño de la cartera de clientes, esta tarea no tomaba mucho tiempo aún siendo realizada a mano. Sin embargo, conforme la cartera de clientes creció, la realización de ésta se volvió cada vez más tardada y propensa a errores. La implementación del CIMS pretende simplificar esta tarea y eliminar los errores relacionados con la ejecución de la misma. En este caso los errores podrían llevar hasta la cárcel al personal de Crédito Integral al ser culpados por encubrimiento a miembros del crimen organizado. Por su parte, la SHCP requiere de declaraciones de impuestos mes con mes y a pesar de la

[21] SHCP: Secretaría de Hacienda y Crédito Público. Secretaría de Estado encargada de calcular los ingresos del país, manejar los recursos del mismo, regular el sistema bancario del país, cobrar impuestos, dirigir los servicios aduanales entre otras responsabilidades. **[88]**

[22] PGR: Procuraduría General de Justicia: Organismo federal encargado de investigar y perseguir los delitos de orden federal. **[89]**

actual implementación de sistemas de cómputo para el cálculo de impuestos por parte del departamento de contaduría de Crédito Integral, la base para el cálculo de impuestos todavía se tiene que calcular a mano. El CIMS tiene contemplado dentro de sus funciones la generación de reportes fiscales y la exportación de los mismos a medios compatibles con software popular de contaduría dentro del mercado mexicano como es el COI®[23].

5.4 Funciones esenciales

Casi toda empresa tiene procesos particulares. Los directivos, en búsqueda de la eficiencia y la estandarización de dichos procesos, a veces recurren a paquetes de software. Estos últimos, en caso de que estén bien hechos y que se adapten perfectamente a los procesos de la empresa, resultan ser de gran utilidad pues aumentan en gran medida la productividad de ésta **[80]**. La dirección de la empresa en cuestión estaba consciente de ello y buscó el desarrollo de un sistema hecho a la medida que le ayudara a manejar sus procesos. El resultado fue la creación del CIMS y sus funciones principales giran alrededor de los procesos de alto nivel de la ya mencionada SOFOM Crédito Integral. A continuación se describirán los procesos principales que son parte de Crédito Integral y de CIMS.

Captura y mantenimiento de cartera de clientes

Las funciones relativas a la captura de clientes son las típicas que podemos encontrar en un sistema de información. Estas incluyen altas, bajas, cambios y despliegue de la información de la base de datos. Cabe recalcar que, debido a la necesidad de obtener la mayor cantidad de información del posible cliente, la captura de clientes es bastante extensa. Durante la creación de las diferentes pantallas de captura de información se buscó siempre la reutilización de componentes al analizar los requerimientos que compartían los distintos tipos de clientes. Los tipos principales de éstos son persona física y persona moral que corresponden a particulares y empresas respectivamente.

[23] COI: Contabilidad Integral. Paquete de software creado por la compañía mexicana Aspel diseñado para facilitar las tareas de captura, proceso y mantenimiento de la información contable de la empresa que lo implemente. **[90]**

Generación de tablas de amortización

Las tablas de amortización son parte clave de cualquier institución de crédito y obviamente esta empresa no es la excepción. CIMS contempla, dentro de sus funciones, la generación de una tabla de amortización un poco atípica que cumple con los requisitos peculiares del director general. Las tablas son parametrizables y pueden ser generadas tomando como semilla un rango de fechas sobre el cual se han realizado todas las transacciones relativas al crédito en cuestión.

Generación de estados de cuenta

Un estado de cuenta le informa al cliente el resumen de los movimientos realizados en un periodo de tiempo determinado, éste es por lo general mensual. El sistema en cuestión genera de forma automática estados de cuenta y los envía por correo electrónico a los clientes de manera oportuna antes de su fecha de corte. Esto a fin de fomentar el pago puntual por parte del cliente. Es prudente mencionar que se realizaron tareas de estandarización de estados de cuenta; debido a que los anteriores se hicieron a mano por lo que tenían varias inconsistencias entre ellos, lo cual provocaba confusión en el cliente, éste procedía a quejarse una vez que veía su estado de cuenta y no lograba entenderlo. Una forma más en la que CIMS mejoró el funcionamiento de Crédito Integral.

Generación de reportes personalizados dinámicos

Durante el tiempo en el que se estaban definiendo los requerimientos, nació un fenómeno llamado "reportitis" (aunque es probable que alguien ya haya nombrado algo de esta manera). Este mal contempla múltiples peticiones de reportes para cualquier idea que se le ocurra al personal. El problema es que los ajustes al presupuesto derivados de este mal causaban frecuentemente disgustos a la dirección que se "sorprendía" por dichos ajustes. La solución fue ingeniosa y a la vez algo sencilla: hacer una gran lista de todos los reportes y diseñar una manera de generar reportes virtualmente infinitos. Un motor de reportes fue la implementación de esta solución. El motor recibe una amplia gama de parámetros mismos que utiliza para la creación de los reportes que son exportados a distintas plataformas para su análisis.

5.5 Arquitectura

El CIMS es un sistema implementado como una aplicación de escritorio. Fue diseñado para ser usado en un ambiente cerrado, pero con la posibilidad de ser consultado y alimentado a la vez por varias personas trabajando en distintas máquinas. La arquitectura básica del CIMS gira alrededor de la simplicidad. Tomando en cuenta el hecho de que el sistema está diseñado para un ambiente cerrado, fue que se escogió la arquitectura de dos capas. La figura 5.2 ejemplifica dicha arquitectura.

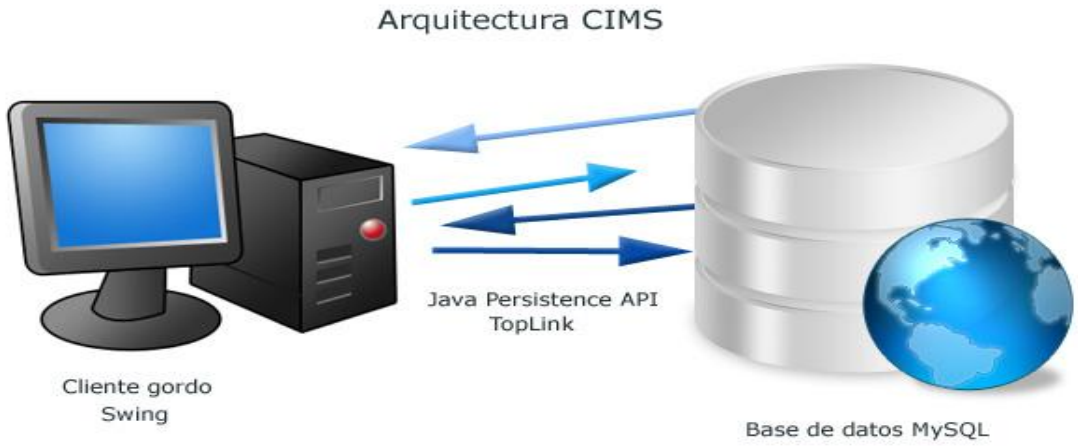


Figura 5.2 – Arquitectura CIMS

Una aplicación de dos capas por lo general incluye un cliente escrito en Java, en este caso usando Swing[24], que se conecta directamente a una base de datos usando algún paquete de conectividad a bases de datos. CIMS usa TopLink[25] para lograr dicho fin. Los detalles de la manera en la que la aplicación interactúa con la base de datos se encuentran la sección 8.21.

Dentro de las ventajas que ofrece este tipo de arquitectura podemos mencionar la experiencia con el usuario enriquecida y sencilla, debido a que los usuarios actuales ya están acostumbrados a los componentes de las interfaces gráficas, así como al estilo de los menús dentro de los clientes gordos. Además la información del sistema reside en una base de datos central a la que acceden todos los clientes gordos evitando así la duplicidad de información y/o la necesidad de sincronización de datos.

[24] Swing: Es un conjunto de herramientas para interfaces gráficas escrita en Java. **[53]**
[25] TopLink: Un conjunto de APIs y herramientas desarrolladas por Oracle para proveer persistencia de objetos y transformación de los mismos. Su objetivo principal es reducir el tiempo de desarrollo y los esfuerzos de mantenimiento de una aplicación por parte de los programadores al mismo tiempo que se incrementa su productividad. **[91]**

Capítulo 6
CIMS – Análisis de requerimientos

Probablemente uno de los problemas más importantes a los que nos enfrentamos al momento de desarrollar sistemas complejos es el de la análisis de requerimientos. Éste se encarga de dictar la funcionalidad del sistema junto con sus características específicas. Así mismo, debe de indicar las restricciones impuestas al sistema en términos del entorno en donde funcionará. Tomando en cuenta lo anterior es que podemos definir al análisis de requerimientos como el proceso de comunicación entre clientes, usuarios y desarrolladores de sistemas **[92]**.

La dificultad de los requerimientos y los procesos asociados a éstos, está directamente relacionada a su estrecho lazo con los humanos. La ingeniería de requerimientos involucra procesos técnicos y sociales. Estos últimos son los que nos dan problemas porque intervienen preferencias, problemas sociales, caprichos, intereses políticos y muchas otras cosas más inherentes a la convivencia humana. Hoy tenemos varias herramientas como los métodos formales y los casos de uso que nos ayudan a mitigar dichos problemas.

6.1 Introducción al documento de requerimientos

El documento de especificación de requerimientos de software engloba las necesidades y requerimientos demandados por la dirección general de Crédito Integral así como las restricciones y necesidades técnicas expuestas por el área de Desarrollo de Sistemas en relación a la construcción del Sistema Integral denominado CIMS (Crédito Integral Management Suite).

Requerimientos

A continuación se contemplan los requerimientos de usuario, tanto funcionales como no funcionales los cuales nos servirán como base para el desarrollo del proyecto "Crédito Integral Management Suite" el cual estará compuesto por los siguientes módulos:

- Módulo de gestión de entidades operables.
    - Gestión de personas físicas.

    - Gestión de personas morales.
    - Gestión de créditos.
    - Gestión de clientes.
- Motor de información básica.
    - Módulo de generación de tablas de amortización.
    - Módulo de generación de estados de cuenta.
- Motor de generación reportes (Multireportes).
- Herramientas diversas como generación de documentos.

Definiciones, acrónimos y abreviaturas

Término	**Definición corta**
Crédito Integral	Empresa en crecimiento dedicada al otorgamiento de créditos y actividades relacionadas bajo el novedoso modelo de SOFOM.
Involucrado clave A	Director general de Crédito Integral.
Involucrado clave B	Director operativo de Crédito Integral.
Tabla de amortización	Herramienta usada normalmente en diversos tipos de crédito con plazo fijo que contiene información sobre el número de pagos, el monto de los mismos y la cantidad de dinero aplicada a interés y a capital **[93]**.
Estado de cuenta	Documento oficial enviado por la institución financiera al deudor (en este contexto) en donde se desglosan todos los movimientos realizados en la cuenta durante cierto periodo de tiempo **[94]**.
Accionista	Persona moral o física que posee acciones
UDI	Unidad de Inversión. Medida económica usada como referencia en varias transacciones monetarias. Está basada en indicadores económicos como la inflación y, hasta el 6 de marzo de 2010, valía $4.268 pesos **[95]**.
IPAB	Instituto de Protección al Ahorro Bancario. Institución creada por el gobierno mexicano a fin de garantizar los ahorros de los mexicanos depositados en bancos **[95]**.
Multireportes	Herramienta del CIMS capaz de crear reportes acumulados que reciben diferentes parámetros.

6.2 Requerimientos funcionales

Listado de actores

A continuación se mostrará un diagrama que ejemplifica la interacción de los distintos usuarios del CIMS de manera conjunta con el software mismo. Más adelante se encuentra una descripción detallada de la labor de cada uno de los actores con el sistema.

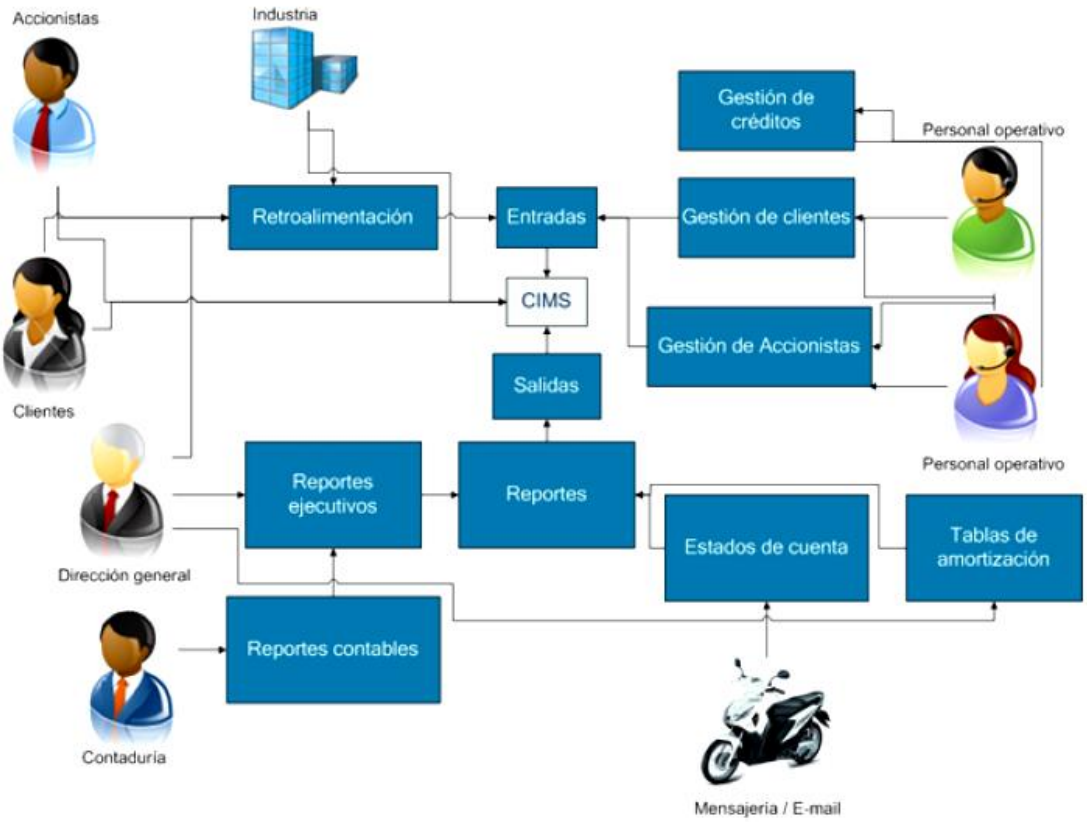


Figura 6.1 – Diagrama actores CIMS

Accionistas

Estos actores tienen una interacción indirecta con el CIMS. Si bien están muy interesados en lo relacionado con la empresa, realmente nunca hacen uso del sistema. En el futuro cercano, se tiene pensado crear interfaces para que los accionistas y usuarios consulten vía web el estado de sus transacciones. Sin embargo, por el momento el contacto con el sistema es a través de los reportes ejecutivos creados por el módulo **`Multireportes`** y entregados vía terrestre por mensajería o vía electrónica (correo electrónico). Además, como cualquier accionista, su labor dentro de la empresa

es proveer a la misma de capital líquido a cambio de un rendimiento en intereses. Cabe aclarar que la razón por la cual las SOFOMes captan recursos por medio de accionistas es debido al alto nivel de riesgo al que están expuestos dichos recursos. El problema aquí es el hecho de que ni las SOFOMes reguladas, ni las no reguladas, tienen protección del IPAB.

Clientes

Los clientes tampoco tienen una interacción directa con el CIMS. Sin embargo, son la parte medular del negocio al ser quienes aportan ganancias al mismo. El cliente nunca hace uso del sistema a pesar de su vital importancia dentro de la compañía. Éste simplemente solicita préstamos por medio de operadores de Crédito Integral, hace sus depósitos de la forma en la que más le convenga y recibe sus estados de cuenta para poder entender por qué siempre debe de pagar más de lo que pidió originalmente. El cliente juega un papel importante en el mejoramiento de la calidad en el servicio de la empresa, al proveer retroalimentación en el servicio. Y dado que el funcionamiento de la institución está íntimamente ligado al CIMS, el contar con esta información podría modificar también el funcionamiento del sistema.

Dirección general

La dirección general es el actor principal en la toma de decisiones de la empresa. El CIMS fungirá como herramienta primordial en el soporte de dicha actividad. Las distintas funciones analíticas del sistema como lo es el **Multireportes**, proveerán una perspectiva global del estado actual de la empresa de una forma rápida y sencilla. Además de utilizar el sistema como herramienta directiva, la dirección general también será pieza clave en la obtención de retroalimentación sobre el desempeño del CIMS.

Contaduría

Este actor es en cierta forma independiente del sistema, pues si bien los intereses generados por el capital colocado en créditos genera impuestos, el cálculo de la contabilidad es realizado por el sistema COI® (Contabilidad Integral). Los contadores sólo tienen que exportar la información de los créditos vigentes haciendo uso del CIMS para después importar dichos datos al COI®. Cabe recalcar que la figura de SOFOM es altamente ventajosa en comparación con la de un banco tradicional en términos

fiscales, pues el capital colocado en créditos no genera impuestos sino sólo los intereses del capital mismo **[13]**.

Industria

Al citar a la industria como actor nos referimos al medio en el cual se encuentran todos los actores del CIMS. Este personaje, aunque indirecto, es el que realmente dicta el rumbo de la empresa de una manera abstracta y general. Por ejemplo si la industria mexicana no hubiera hecho evidente la necesidad de cambio en el sector financiero respecto al otorgamiento de créditos, el gobierno mexicano nunca hubiera creado las SOFOMes. Como consecuencia directa de lo anterior, Crédito Integral probablemente no se hubiera creado.

Personal operativo

Este actor es quien más tiene contacto con el CIMS. Su labor diaria va desde capturar nuevos clientes y créditos hasta hacer llamadas de cobranza a clientes morosos. Dentro de las herramientas que le proporciona el sistema se encuentran pantallas de captura, edición y visualización de clientes y créditos. El registro de los pagos ejecutados por los clientes también es tarea de este actor. El personal operativo es probablemente uno de los actores que resultan más beneficiados al realizar todas sus actividades a través del CIMS, debido a que la responsabilidad de realizar los cálculos dentro de los reportes es delegada al software. Además sus funciones serán efectuadas con mayor rapidez porque el CIMS incluye atajos para ejecutar acciones comúnes dentro de Crédito Integral.

Mensajería / medios electrónicos

Este actor es simplemente un vehículo para la información. Toda la información contenida en la base de datos viaja hacia los clientes complejos o gordos (*fat clients*) a través de una red local. Por otro lado, los estados de cuentas son enviados vía correo electrónico a fin de que los clientes tengan la información de manera oportuna y puedan realizar su pago de la misma forma[26].

[26] También se usa el correo postal convencional para poder enviar una copia de los estados de cuenta a los clientes, ya que muchos de ellos los requieren como comprobantes fiscales pues son documentos oficiales que pueden ser usados en deducciones de impuestos.

Requerimientos

Los requerimientos funcionales que se presentarán a continuación estarán separados por módulos del sistema y/o actividad a realizar por parte del CIMS. Estos se encuentran estrechamente alineados con los objetivos principales del proyecto, dentro de los cuales se encuentra la automatización de tareas de la SOFOM, junto con las características del medio en donde se va a implementar (oficinas centrales de Crédito Integral). En seguida se encuentra la figura 6.2 en donde se enlistan los requerimientos y sus características

#	Descripción	Detalles / Restricciones	Módulo asociado
1	Restringir el acceso y uso del CIMS por medio del uso de identificación y autenticación.	Existirán dos tipos de usuarios: administrador y restringido. El primero tendrá acceso total al sistema incluyendo la modificación de los datos, y el segundo sólo podrá crear reportes y leer datos.	Identificación y autenticación.
2	Gestionar usuarios. Esto incluye altas, bajas, lectura y cambios de los mismos.	Sólo el usuario tipo administrador podrá realizar dichas tareas.	Identificación y autenticación. Gestión de usuarios.
3	Modificar privilegios de usuarios.	Un usuario con cierta prioridad no puede asignar a otro prioridad más alta que la suya.	Identificación y autenticación. Gestión de usuarios.
4	Restringir las contraseñas para que no sean muy fáciles de adivinar.	La clave no debe contener números ni letras consecutivas y debe ser una mezcla entre mayúsculas, minúsculas y números. Además debe de ser de mínimo 8 caracteres.	Identificación y autenticación. Gestión de usuarios.
5	El CIMS deberá mostrar una pantalla al inicio mientras el sistema se esté cargando.	La pantalla debe mostrarse lo más pronto posible.	Principal.
6	Administrar clientes. Altas, bajas, cambios y lectura de clientes debe ser posible.	Se deben diferenciar dos tipos de clientes principalmente: persona moral y persona física.	Gestión de clientes, Principal.
7	La actividad que involucra dar	Los pasos deben de agrupar	Gestión de clientes.

	de alta clientes debe de estar dividida en varios pasos debido a que muchos datos son recabados tanto de persona moral como de persona física.	datos a recabar del cliente similares entre sí.	
8	Las validaciones del paso actual deben de cumplirse para pasar al siguiente en alta / edición de clientes.	Cada paso contará con una serie de validaciones las cuales deberán de cumplirse para continuar con el proceso.	Gestión de clientes.
9	Los clientes deben poder ser buscados.	Los criterios de búsqueda incluyen nombre, razón social, tipo de cliente e identificador del cliente.	Gestión de clientes.
10	Gestionar créditos. El usuario debe poder dar de alta, mostrar, dar de baja y editar créditos.	Un crédito debe forzosamente estar asociado a un cliente. El usuario deberá poder escoger entre calcular las fechas de amortización automáticamente o manualmente. El crédito a dar de alta o modificar debe ser validado a fin de checar que contenga todos los datos indispensables.	Gestión de créditos, Principal.
11	No permitir borrar un cliente que todavía tenga créditos.	Para borrar un cliente primero se deberán borrar sus créditos asociados.	Gestión de clientes, Gestión de créditos.
12	Administrar transacciones. Dentro de las operaciones incluidas se encuentran listados, altas, bajas y cambios.	Una transacción debe estar asociada a un crédito. Validaciones para la edición y alta deben de estar incluidas en las pantallas.	Gestión de transacciones.
13	No permitir borrar un crédito que cuente con transacciones asociadas.	Primero deberán ser borradas todas las transacciones de dicho crédito.	Gestión de créditos, Gestión de transacciones.
14	Generación de tablas de amortización.	Este reporte debe ser personalizable. Le fecha final, el cliente y el crédito deben estar incluidos como parámetros.	Tablas de amortización.
15	Generación de estados de	Dependiendo del mes, el	Estados de cuenta.

	cuenta.	crédito y el cliente CIMS debe poder generar un estado de cuenta basado en una plantilla gráfica de la empresa.	
16	Creación de reportes acumulativos en donde los créditos de entrada y las características a mostrar de los mismos sean parametrizables.	El usuario deberá poder crear un reporte común con el menor número de "clicks" posible.	**Multireportes**.
17	Los créditos a incluir en el Multireportes deben de poder ser agregados desde la pantalla en donde se muestren los resultados de la búsqueda de clientes.		**Multireportes**, Principal.
18	Los reportes acumulativos deben poderse exportar a un formato popular.	Dada la amplia aceptación del Office de Microsoft dentro y fuera de Crédito Integral **[56]** se pensó que sería buena idea exportar reportes a Excel[27].	**Multireportes**.

Figura 6.2 – Requerimientos funcionales

6.3 Requerimientos no funcionales

Los requerimientos no funcionales incluyen los márgenes sobre los cuales el sistema tendrá que funcionar y se enlistan a continuación:

- Confiabilidad.

  En este contexto, se define como la habilidad del sistema para realizar sus funciones bajo ciertas condiciones previamente definidas por un periodo determinado de tiempo. Dentro de esta categoría, el CIMS debía de cumplir con los siguientes requisitos:

[27] Excel: Software capaz de crear y editar hojas de cálculo creado por Microsoft® como parte del conjunto de aplicaciones de Office **[78]**.

- Disponibilidad.

El sistema debe de estar disponible en el horario de operaciones de Crédito Integral, éste es de lunes a viernes de 9:00 – 18:00 hrs. La interrupción en el funcionamiento del sistema en ese horario sería muy problemática, por lo que las reparaciones y cambios al sistema deberán hacerse fuera de ese horario.

- Tiempo promedio de reparación (MTTR[28]).

El sistema no debe de estar fuera de línea más de un día laboral. Durante ese tiempo, el personal estará operando de forma manual; por lo que el sistema deberá ser sincronizado una vez sea reparado. Un aumento en la cantidad de este tiempo atrasaría demasiado las operaciones de Crédito Integral.

- Exactitud.

Debido a que una exactitud distinta al 100% en las cifras que reporta el CIMS tendría un impacto catastrófico sobre la confianza de los clientes en la empresa, los datos generados por el software no deberán de presentar errores.

- Rendimiento.

Cualquier operación realizada dentro del CIMS debe de realizarse en a lo más diez segundos, debido a que una vez que haya transcurrido este tiempo, la operación normal de la empresa se vería retrasada **[106]**. A continuación se muestran algunos casos en donde, debido a la naturaleza del proceso, la restricción anterior no aplica.

- Búsquedas.

Estas operaciones no deben de tardar más de veinte segundos tomando en cuenta el volumen de clientes y créditos que actualmente se manejan y los proyectados en el futuro. Cabe señalar que actualmente existen sólo 173 créditos dados de alta y actualmente la dirección de Crédito Integral no espera que la cifra rebase las mil unidades para fines del 2010 **[107]**.

- Generación de reportes individuales.

[28] MTTR (*Mean time to repair*): Tiempo promedio de reparación. Medida básica de mantenimiento de elementos reparables, representa el tiempo promedio que se requiere para reparar un componente que ha fallado **[105]**.

Excepcionalmente se podrá requerir la elaboración de reportes individuales por un periodo muy extenso de tiempo (más de veinte años), para estos casos el tiempo de respuesta del CIMS podrá extenderse. El proceso estará limitado sólo por la cantidad de memoria disponible.

- Operación del **`Multireportes.`**

El resultado final de este módulo en el peor de los casos no deberá superar los dos minutos de tiempo **[107]**. En caso contrario, se permitirá (en un futuro) implementar funciones de optimización como memoria caché de ciertas operaciones sobre créditos viejos o inactivos. La implementación de dichas funciones no está pensada para esta versión debido a que el volumen de créditos manejado actualmente por Crédito Integral está dentro de los márgenes establecidos por esta sección.

6.4 Pantallas y diagramas

De manera ilustrativa a continuación se encuentran algunas pantallas y diagramas interesantes pertenecientes al CIMS.

Pantallas

Los requerimientos son claros, pero resulta atractivo ver algunas pantallas del sistema funcionando.

Módulo: Gestión de clientes.
Pantalla: Alta de Persona Moral.

En la figura 6.3 se muestra la pantalla en donde se dan de alta los clientes tipo "Persona Moral". Podemos observar que el proceso está divido en varios pasos en forma de *wizard*[29] y que cada paso cuenta con un pequeño formulario que debe de ser llenado. Elegimos usar un *wizard* como contenedor de los formularios debido a que es un componente gráfico común en las interfaces gráficas modernas **[61]**, por lo tanto contribuye a que el CIMS sea fácil de usar. También evita que el usuario cometa

[29] Wizard: Es un conjunto de pasos representados por paneles en la interfaz gráfica de un software **[61]**.

menos errores en el proceso de alta debido a que los datos a recabar están organizados basados en el elemento a describir. Además creemos, aunque es nuestra suposición solamente, que el uso de este componente puede hacer creer al usuario que el proceso de alta es más ligero, porque nunca llega a ver el formulario completo.

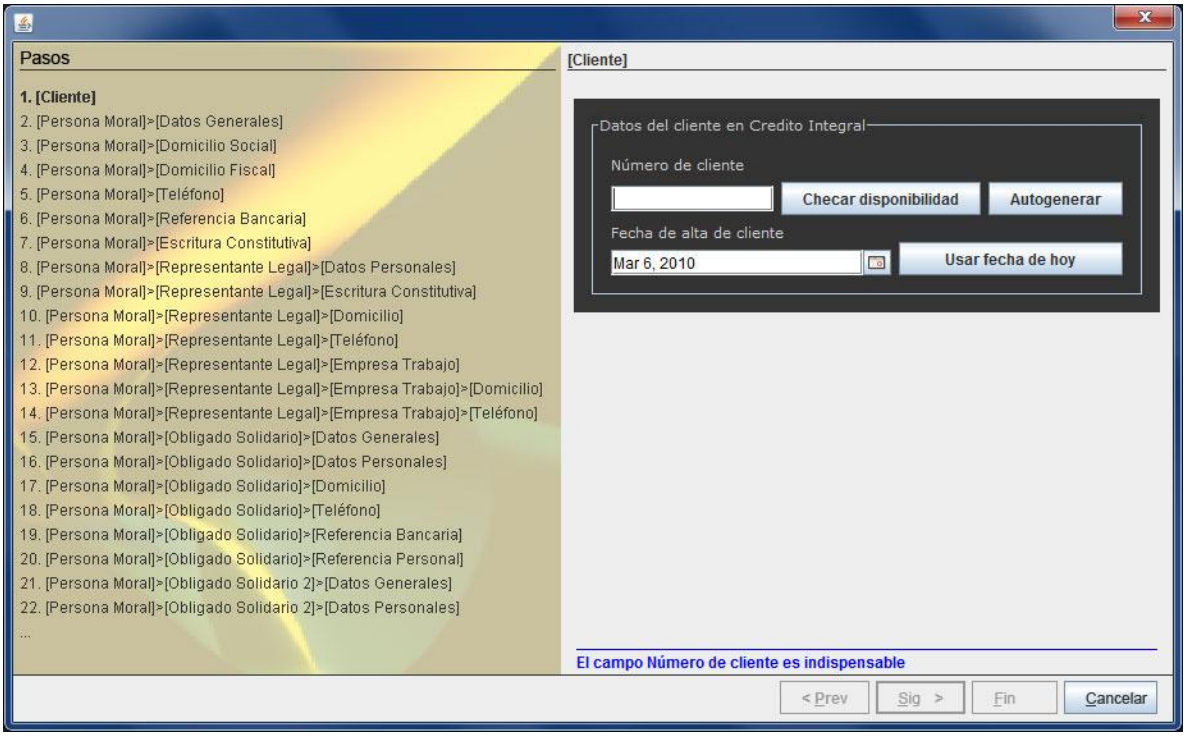


Figura 6.3 – Alta de una Persona Moral en el CIMS

La figura anterior sólo muestra dos campos: número de cliente y fecha de alta de cliente. Resulta importante destacar que ambos pueden ser generados automáticamente por el CIMS a fin de facilitar la tarea de darlas de alta. También, si el usuario lo desea, los campos pueden ser editados manualmente y validados posteriormente por el sistema evitando así almacenar información inválida.

Módulo: **`Multireportes`**.
Pantalla: Vista final del reporte generado.

Después de haber recabado todas las características que el usuario desea que se muestren en su reporte, CIMS procede a generarlo y mostrarlo dentro de una "tabla jerárquica". Este componente facilita la visualización de información subordinada como

es el caso de los créditos y clientes. De esta forma el usuario puede rápidamente encontrar los datos que necesita. La persona viendo el reporte es capaz de contraer o expandir los clientes dependiendo del detalle de los créditos que requiera analizar. La figura 6.4 nos muestra la vista del **`Multireportes`** en acción.

MultiReportes

Archivo Ayuda

Empresa/Crédito	1 Pago Realizado[Capital]	1 Pago Realizado[Interes Moratorio]	1 Pago Realizado[Interes Vencido]	1 Pago Realizado[Interes]	1 Pago Realizado[Total Interes]
DISEÑO Y CONSTRUCDCIONES ALKA, S.A. DE C.V.	$3,248,099.99	$1,999,155.82	$374,412.67	$0.00	$374,412.67
GRUPO IMPULSA, DISEÑO Y CONSTRUCCION SA DE CV	$2,506,200.00	$486,373.95	$359,347.25	$494.14	$359,841.40
PROMOCIONES EN PLAZA, S.A. DE C.V.	$5,363,934.12	$558,711.20	$1,082,177.48	$308,664.40	$1,390,841.88
PLAYING DISPLAY, S.A. DE C.V.	$320,000.00	$2,062.64	$32,652.58	$7,370.10	$40,022.68
004-01 // 10.Oct.2007 // $320,000.00 // 5% // 4 meses	$320,000.00	$2,062.64	$32,652.58	$7,370.10	$40,022.68
004-02 // 16.Oct.2009 // $150,000.00 // 10% // 2 meses	$0.00	$0.00	$0.00	$0.00	$0.00
004-03 // 06.Nov.2009 // $150,000.00 // 10% // 2 meses	$0.00	$0.00	$0.00	$0.00	$0.00
GRANDES IDEAS ACTIVAS, S.A. DE C.V.	$1,334,999.99	$74,760.07	$80,333.25	$79,650.00	$159,983.26
EASY COMM COMUNICACIONES, S.A. DE C.V.	$2,102,999.98	$713,628.96	$374,501.97	$76,779.99	$451,281.96
CONTINENTAL DE SRVICIOS, S.A. DE C.V.	$2,575,499.99	$434,896.02	$265,652.29	$6,200.15	$271,852.44
CONSOLIDADORA TECNOLOGICA PROYECTOS SA DE CV	$603,000.00	$20,153.33	$26,620.00	$0.00	$26,620.00
MONICA MARIA PEÑA PEÑA	$94,000.00	$0.00	$480.50	$729.60	$1,210.10
009-01 // 22.Oct.2007 // $57,000.00 // 1.2% // 1 meses	$57,000.00	$0.00	$0.00	$729.60	$729.60
009-02 // 03.Mar.2008 // $12,000.00 // 1.2% // 1 meses	$12,000.00	$0.00	$155.00	$0.00	$155.00
009-03 // 23.Oct.2008 // $25,000.00 // 1.3% // 1 meses	$25,000.00	$0.00	$325.50	$0.00	$325.50
009-04 // 24.Aug.2009 // $33,000.00 // 1.2% // 1 meses	$0.00	$0.00	$0.00	$0.00	$0.00
ANTONIO VALDERABANO VALDERABANO	$80,000.00	$26,804.73	$4,000.00	$0.00	$4,000.00
FERNANDO DIAZ DIAZ	$82,900.00	$2,103.45	$8,030.42	$0.00	$8,030.42
MIGUEL ANGEL GLORIA GLORIA	$1,055,000.00	$418,903.65	$121,365.30	$0.00	$121,365.30
DESARROLLOS TURISTIVCOS BLANCO, S.A. DE C.V.	$1,299,999.97	$202,131.46	$137,148.11	$43,055.56	$180,203.67
FILTRECO, S.A. DE C.V.	$301,399.94	$37,219.42	$47,979.43	$24,659.83	$72,639.27
SARA MAGDALENA GALINDO GALINDO	$165,000.00	$5,337.10	$717.92	$0.00	$717.92
SISTEMAS CONSTRUCTIVOS DE GEOTECNIA SA DE CV	$3,928,999.98	$413,364.92	$716,932.19	$138,466.67	$855,398.86
CONTRATISTAS DE OBRAS CIVILES, S.A. DE C.V.	$1,981,500.00	$7,373.50	$107,106.80	$0.00	$107,106.80
JAVIER ARTURO VILCHIS VILCHIS	$50,000.00	$4,875.00	$4,929.17	$0.00	$4,929.17
EDUARDO LUISILLO LUISILLO	$15,550.00	$0.00	$0.00	$0.00	$0.00
Fuerza de Trabajo Industrial SA de CV	$644,999.98	$183,487.71	$124,105.84	$52,140.00	$176,245.84
RUBEN CIELAK CIELAK	$162,000.00	$29,648.95	$4,776.53	$0.00	$4,776.53
LIZETH NANCY RODRIGUEZ RODRIGUEZ	$18,300.00	$0.00	$148.63	$1,181.07	$1,329.70
LEONTE RAMON GUZMAN GUZMAN	$499,000.00	$0.00	$2,540.92	$3,583.60	$6,124.52
Jorge Castellanos Castellanos	$1,686,073.75	$459.00	$57,837.92	$0.00	$57,837.92
Gran Total	$30,119,457.67	$5,621,450.89	$3,933,797.19	$742,975.11	$4,676,772.30
Reporte al 06.Mar.2010					

Figura 6.4 – Vista final del **Multireportes**

En la vista final del **Multireportes** podemos observar varios clientes enlistados con su indicador general: pagos realizados. Algunos cuentahabientes, como es el caso de "Playing Display S.A. de C.V.", muestran sus respectivos créditos junto con sus detalles vinculados: monto, tasa ordinaria, tasa moratoria, y plazo. Las columnas mostradas corresponden a los pagos realizados divididos en lo que fueron aplicados: capital, interés ordinario, interés moratorio e interés a capital vencido. Además se muestra una columna adicional que corresponde al pago total de intereses.

Diagramas

Los "planos" usados para desarrollar el CIMS son parte clave en su creación. Veamos un ejemplo.

Módulo: Gestión de clientes.
Clase principal: **PersonaFisicaView.**

Tipo de diagrama: Diagrama de clases.
La figura 6.5 muestra un diagrama de clases centrado en la clase **PersonaFisicaView**. Esta clase es parte de la interfaz gráfica y hereda de **AbstractEntityClassView**, la cual posee funciones útiles e indispensables para todas las pantallas. Por ejemplo, la persistencia de datos es implantada en esta clase debido a que todas las pantallas que contienen formularios deberán guardar la información editada en la base de datos. Por su parte, las clases **EmpresaTrabajoView** y **DomicilioView** también heredan de **AbstractEntityClassView** para poder reutilizar los métodos relativos a la gestión de datos como **bind()** y **update()**. Individualmente, la clases en el diagrama poseen las siguientes objetivos:

- **PersonaFisicaView**: Capturar, editar y mostrar datos únicos y exclusivos de los clientes tipo persona física.
- **EmpresaTrabajoView**: Gestionar información relativa a la empresa en donde trabaja el cliente. La empresa donde trabaja el representante legal de las personas morales también aplica.
- **DomicilioView**: Administrar los datos de cualquier domicilio incluyendo el domicilio de las personas físicas y morales.

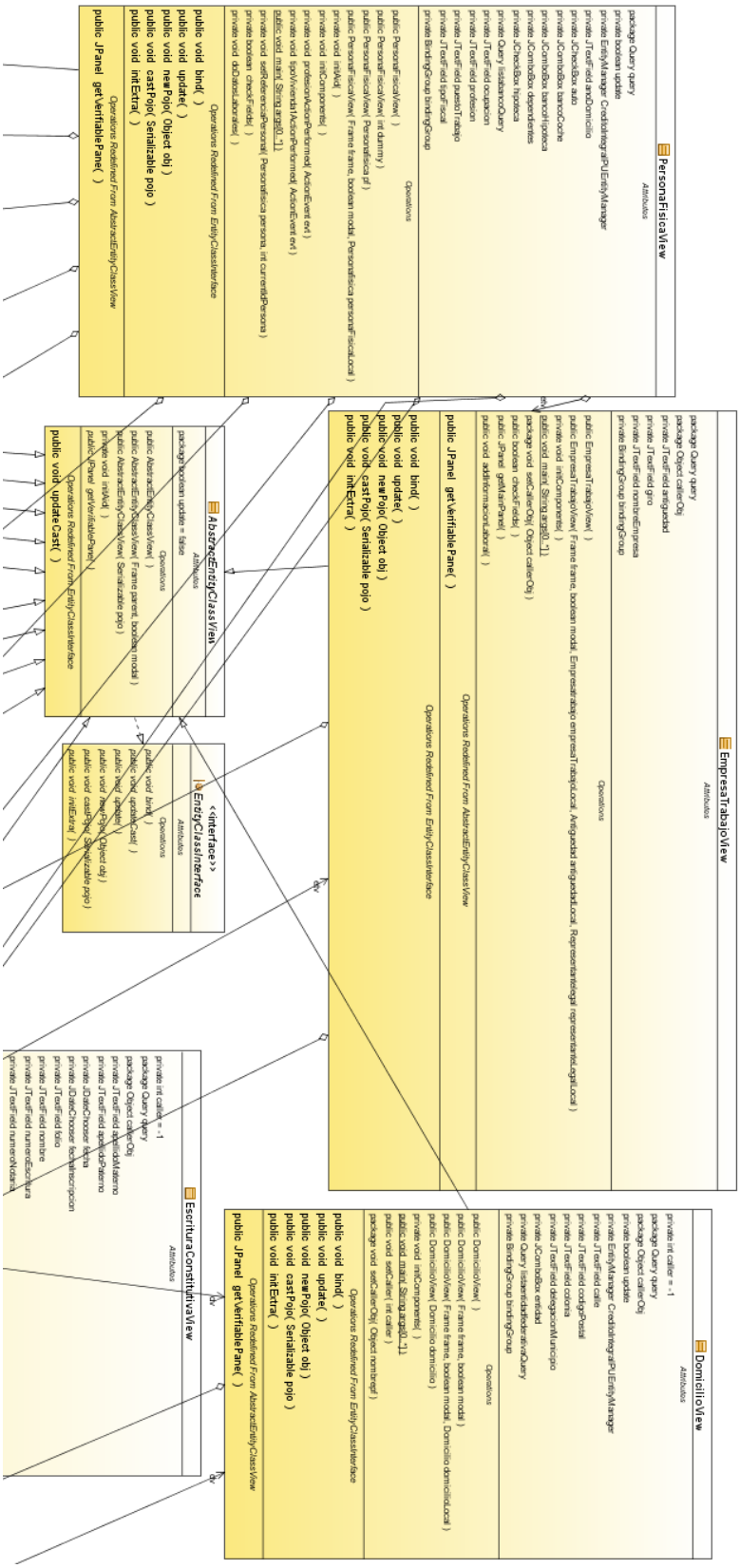


Figura 6.5 - Parte de un diagrama de clases del CIMS

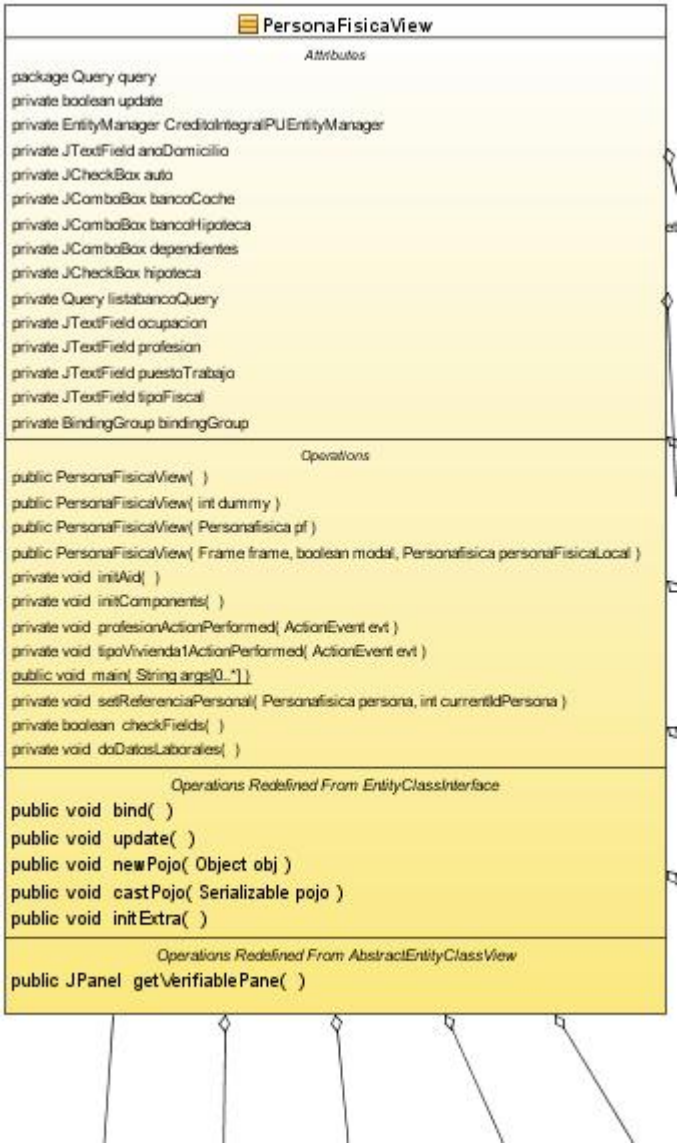


Figura 6.5-1 - Parte de un diagrama de clases del CIMS (ampliación parte 1)

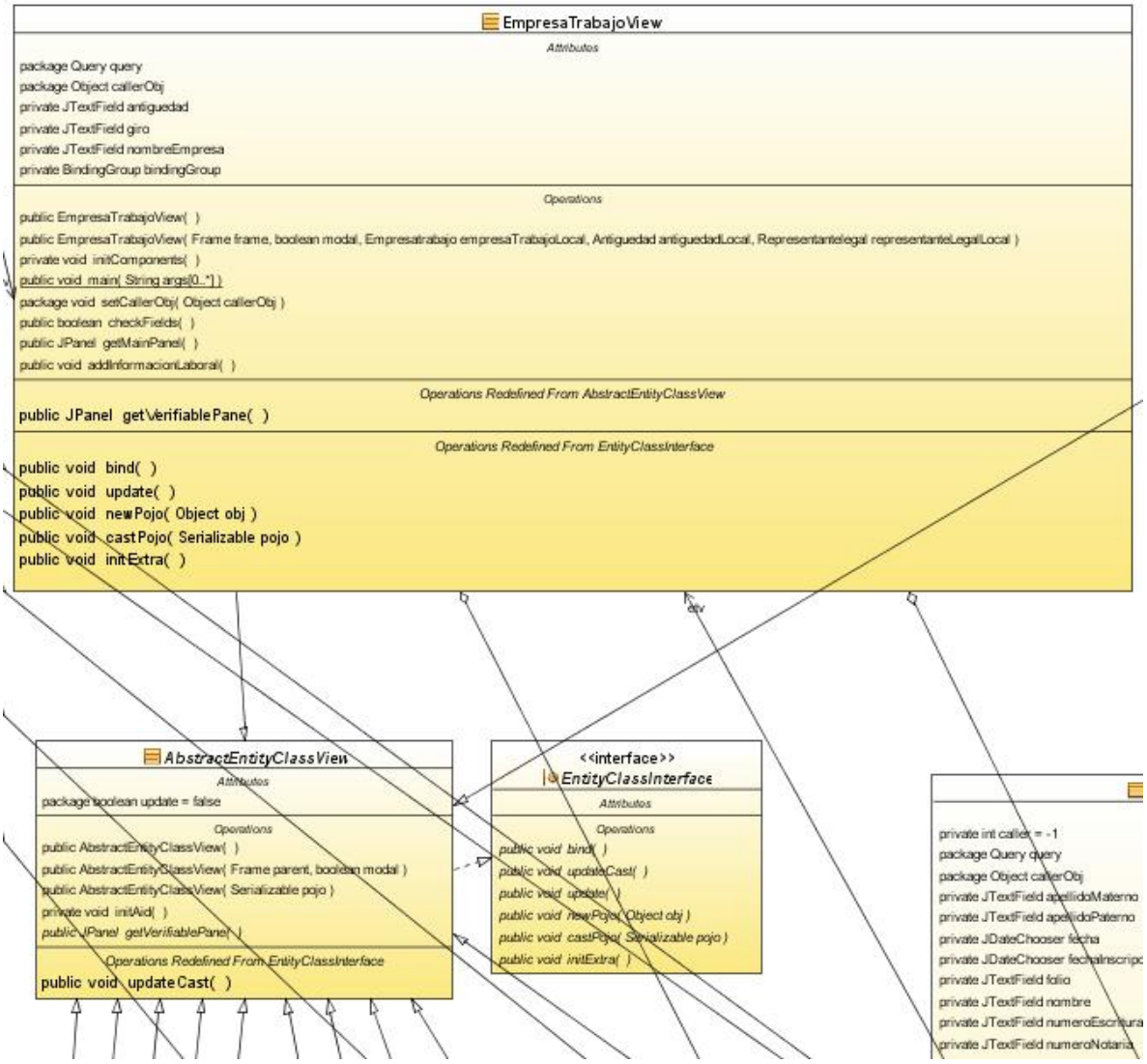


Figura 6.5-2 - Parte de un diagrama de clases del CIMS (ampliación parte 2)

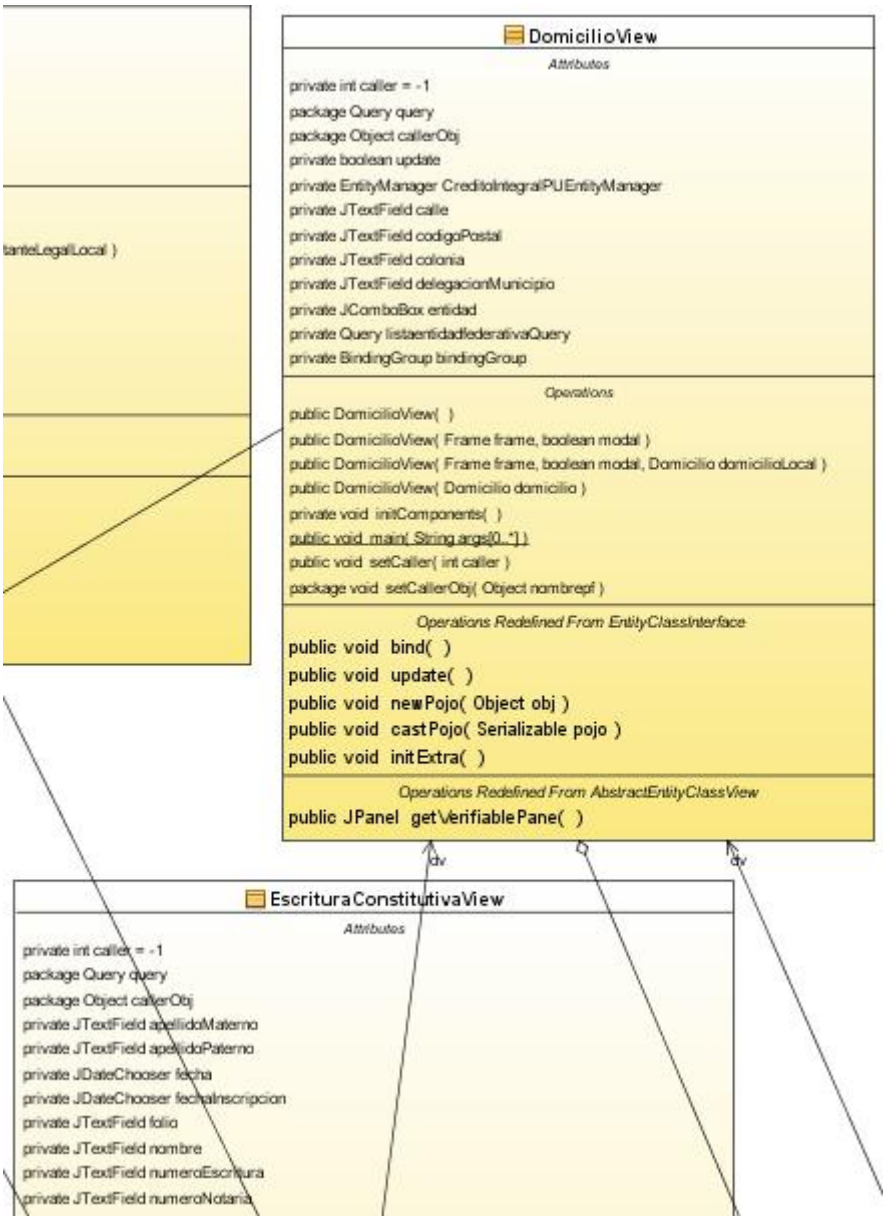


Figura 6.5-3 - Parte de un diagrama de clases del CIMS (ampliación parte 3)

El diagrama de clases anterior muestra cuatro clases que representan pantallas de la interfaz gráfica: **`PersonaFisicaVew`**, **`EmpresaTrabajoView`**, **`DomicilioView`** y **`EscrituraConstitutivaView`**. Todas las anteriores cuentan con formularios para visualizar y editar información de los clientes. Dichos formularios están compuestos por elementos gráficos pertenecientes a las bibliotecas de Java. Podemos observar en la figura anterior que la mayoría de las variables globales como **`ocupación`**, **`giro`** y **`calle`**, son justo esos elementos gráficos, por ejemplo **`JTextField`** y **`JComboBox`**. Los

métodos son simplemente constructores e inicializadores de los elementos gráficos, como **`init()`** que es generado automáticamente por Netbeans. Casi todo el código de las clases gráficas está centrado en la interacción con el usuario para aislar lo más posible el código de negocio de las clases gráficas. Las funciones que comparten todas las clases gráficas se encuentran en la clase **`AbstractEntityClassView`**, por ejemplo aquellas relacionadas con la gestión de datos. Cabe mencionar que esta clase es de tipo abstracta y por lo tanto no se crean instancias de ésta, es necesario heredar de ella para poder usar sus funciones, las cuales se enfocan principalmente en la sincronización entre los componentes gráficos y la base de datos, así como la validación de información. Ejemplo de éstas es el método **`bind()`** cuya mecánica principal se explica detalladamente en la sección 8.2.5. Aquí también se encuentran detalles de las labores de gestión de datos incluidas en la mencionada clase.

Capítulo 7

Bases teóricas del diseño del CIMS

7.1 Diseño de la base de datos

La información es probablemente el recurso más importante dentro de una organización **[35].** La adquisición y el correcto manejo de información de manera oportuna y segura, es la forma en la que las empresas crean ventajas competitivas dentro del mercado. Esto no debe de sorprendernos pues, según el documento "*Key Information Security Measures*" de la empresa "Gartner" **[36]**, las empresas gastan una parte muy importante de su presupuesto en medios para poder proteger su información y explotarla correctamente. Dentro de los medios que podemos mencionar se encuentran herramientas como respaldos automáticos de información, firewalls, tecnología de filtrado de gateways, sistemas de detección de intrusos y demás. Por otro lado, el análisis profundo de los datos protegidos como propiedad intelectual e indicadores del mercado representan una poderosa herramienta que ayuda en la toma de decisiones. Es tal la importancia de la información corporativa que la implementación de herramientas eficientes para el manejo de datos, como el llamado "Enterprise software [30] ", ha dado resultados drásticos en el mejoramiento del funcionamiento de algunas empresas **[37]**.

Un sistema manejador de bases de datos (SMBD) es un grupo conformado por datos relacionados entre sí y un conjunto de programas para poder consultar y manipular dichos datos. Coloquialmente, escuchamos el término Base de Datos y no Sistema Manejador de Bases de Datos, la diferencia entre los dos términos reside en que la información que manipula el sistema manejador de bases de datos es denominada comúnmente base de datos y el sistema manejador sólo opera sobre dicha información **[39]**. El objetivo principal de un Sistema Manejador de Bases de Datos es proveer un medio para guardar y recuperar datos de forma conveniente y eficiente. Para poder

[30] Enterprise software: Software que tiene como propósito resolver un problema empresarial. Ejemplos de este tipo de aplicaciones son ERPs (Enterprise Resource Planning) y CRMs (Customer Relationship Management). **[38]**

entender la importancia de un Sistema Manejador de Bases de Datos, veamos los problemas que resuelve:

- Redundancia de información e inconsistencia.
  Con el paso del tiempo la misma información puede ser capturada y almacenada de distintas maneras y varias veces. Los cambios en el software que usamos para gestionar dicha información, aunados a cambios en el mercado y tendencias son sólo algunas de las causas del mencionado problema **[39]**.

- Dificultad de acceso a la información.
  Los datos almacenados en un sistema de información[31] serán consultados en cierto momento con un propósito específico. El problema aparece al tratar de consultar la información de manera eficiente, ya que a menos de que los datos se encuentren organizados de manera óptima para la consulta[32] específica en cuestión, la operación se vuelve lenta y problemática **[39]**.

- Aislamiento de la información.
  Los datos almacenados en distintos archivos con diferentes formatos son difíciles de consultar debido a la falta de homogeneidad en el formato entre los mismos. Desarrollar software nuevo para tener acceso a dicha información es difícil y costoso. Por ejemplo la utilización de herramientas como la creación de índices se vuelve prácticamente imposible por falta de criterios de comparación y referencia **[39]**.

- Problemas de integridad.
  Dependiendo del tipo de datos a almacenar y de la naturaleza del negocio en cuestión, ciertas restricciones deben imponerse con el fin de asegurar la integridad de la información. Por ejemplo, dentro de Crédito Integral, requerimos que fuera imposible tener créditos sin dueño. Para poder asegurarlo pudimos haber escrito el código necesario dentro de la aplicación. Sin embargo, añadir nuevas restricciones requeriría añadir más código lo cual es laborioso y

[31] Sistema de información: En este contexto, nos referimos a sistema de información como cualquier combinación de TI y las actividades humanas que usan dichas tecnologías para apoyar actividades en general como gestión, toma de decisiones, entre otras operaciones. No necesariamente bases de datos **[126]**.
[32] Consulta: En este contexto, nos referimos a consulta como la acción y el efecto de consultar. No necesariamente nos estamos refiriendo a una consulta en una base de datos **[127]**.

difícil. Dentro de un sistema manejador de bases de datos agregar nuevas restricciones requiere de mucho menos esfuerzo **[39]**. Por ejemplo sumar la restricción anterior en MySQL[33], el sistema manejador de bases de datos del CIMS, requiere de sólo las siguientes instrucciones:

```
  CONSTRAINT `clienteFKcreditoc` FOREIGN KEY (`idClienteF`) REFERENCES
`cliente` (`idCliente`),
  CONSTRAINT `creditoFKcreditoc` FOREIGN KEY (`idCreditoF`) REFERENCES
`credito` (`idCredito`)
```

- Problemas de atomicidad.
  Debido a que los sistemas de software están sujetos a fallas en sus procesos resulta indispensable, dada la importancia de algunos, poder regresar a un estado anterior consistente **[39]**. Sería catastrófico en muchas aplicaciones, como por ejemplo las bancarias, que una operación compuesta por varias operaciones más pequeñas quedara incompleta después de un fallo de comunicación.

- Problemas de acceso concurrente.
  Resulta sencillo imaginar situaciones en las que el acceso concurrente sobre la misma información pueda causar serios problemas **[39]**. Vamos a ilustrar lo anterior con un ejemplo típico. Supongamos que tenemos una misma cuenta de banco en la que se retiran dos cantidades más o menos al mismo tiempo. El problema ocurre cuando un retiro se ejecuta primero y el saldo no es actualizado antes de que la otra transacción retire fondos también porque si la suma de los dos retiros fuera mayor al saldo de la cuenta, habremos logrado retirar más dinero del que realmente tenemos.

- Problemas de seguridad.
  Las restricciones sobre quién puede o no consultar y/o modificar la información en un sistema se pueden implementar de forma sencilla en un SMBD. Un método sencillo y comúnmente usado para lograr dicho objetivo es la creación

[33] MySQL: Sistema manejador de bases de datos multiplataforma, libre, gratuito y con más de 6 millones de instalaciones a nivel mundial **[128]**.

de las llamadas "*access control lists*"[34] **[40]**. Éstas permiten definir que procesos y/o usuarios tengan acceso al objeto en cuestión. También es posible especificar cuáles son las operaciones permitidas por cada uno de los usuarios o procesos.

Modelos de datos

Definimos un modelo de datos como un grupo de herramientas conceptuales usadas para describir éstos, su relación, y semántica, así como posibles restricciones de consistencia. A continuación se describirán los dos modelos de datos más comúnes:

7.1.1 Modelo Entidad-Relación:

Este modelo está basado principalmente en la representación de conceptos a través de objetos llamados entidades y la relación entre ellos. Es utilizado principalmente para diseñar bases de datos de manera gráfica. Veamos los conceptos que integran al modelo:

Entidad: Objeto del "mundo real" único que posee ciertos atributos que lo hacen particular.

Relación: Se refiere a una asociación entre dos o más entidades. Una relación expresa la cantidad de entidades del mismo tipo que se asocian a otro tipo de entidad a través de la cardinalidad[35] de esta asociación.

Las entidades y sus tipos, se distinguen entre ellas por medio de una super llave. Una super llave es uno o más atributos que, al ser usados en conjunto, identifican como única a una entidad dentro de un conjunto de entidades del mismo tipo. Una llave primaria puede crearse si tomamos sólo los atributos de la super llave necesarios para identificar únicamente a cualquier entidad, pero ni uno más.

A continuación presentamos un ejemplo muy sencillo que nos ayudará a comprender mejor los conceptos enunciados. El diagrama 7.1 es una representación gráfica del modelo Entidad-Relación. Con objeto de mantenernos en el ámbito financiero, la figura describe un cliente bancario con sus respectiva cuenta de débito y de crédito. Las

[34] Access Control List (ACL). Se define como una lista de permisos adjuntos a un objeto sujeto a ser leído o modificado **[40]**.
[35] Cardinalidad: El número de elementos en un grupo o conjunto. **[41]**

entidades son representadas por un cuadrado y las relaciones por un rombo. También, podemos observar que un óvalo simboliza a los atributos. Observemos que existen algunos que están subrayados. Éstos representan llaves primarias.

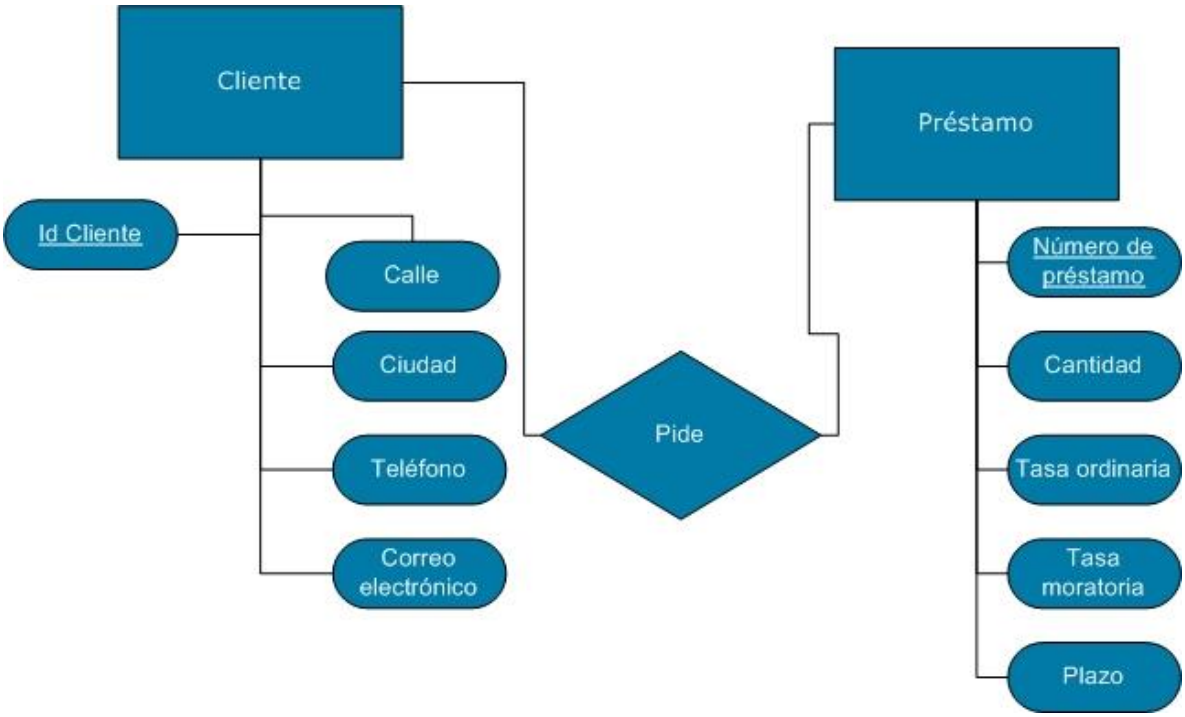


Diagrama 7.1 – Ejemplo simplificado de modelo entidad-relación de la función de otorgamiento de créditos del CIMS

La actividad preponderante del CIMS, y de Crédito Integral, es el otorgamiento de créditos. Resulta sorprendente ver que tanto un banco tradicional como una SOFOM funcionan de la misma sencilla manera, ilustrada en el diagrama anterior. Básicamente un cliente cuyos datos han sido recabados con anterioridad, pide un préstamo de ciertas características.

En el diagrama anterior, el **`Cliente`** es la figura central. Éste posee los atributos **`ciudad`**, **`calle`**, **`teléfono`**, **`correo electrónico`** e **`id Cliente`**, su llave primaria. El cuentahabiente es capaz de pedir **`Préstamos`** por cierta **`cantidad`**. Dicha entidad sólo tiene los atributos **`cantidad`** y **`número de préstamo`** que usamos como llave primaria. El modelo Entidad-Relación no es la única forma de representar una base de datos, también podemos usar el modelo relacional el cual se describe en la siguiente sección.

7.1.2 Modelo relacional

El modelo relacional está basado principalmente en tablas conectadas de forma específica haciendo uso de la lógica de primer orden[36] y la teoría de conjuntos. Se ha convertido en el modelo principal de representación de bases de datos debido a su simplicidad ya que el desarrollador puede después de crear el modelo, hacer operaciones de creación, lectura, actualización y eliminación de tuplas[37] directamente sobre las tablas del modelo.

El diagrama 7.2 es un ejemplo del modelo relacional. Cada relación es representada por una caja, los atributos se encuentran dentro de la misma y el nombre de la relación está en la parte superior. Los atributos que son llaves primarias, están en negritas y subrayados adentro de una caja pequeña. Las relaciones se asocian entre sí por otras que usan llaves de las relaciones a asociar. Por ejemplo los clientes se unen a sus cuentas por la relación *cuenta-cliente*. Lo mismo ocurre con los *préstamos* en donde la relación *préstamo-cliente* realiza dicho trabajo.

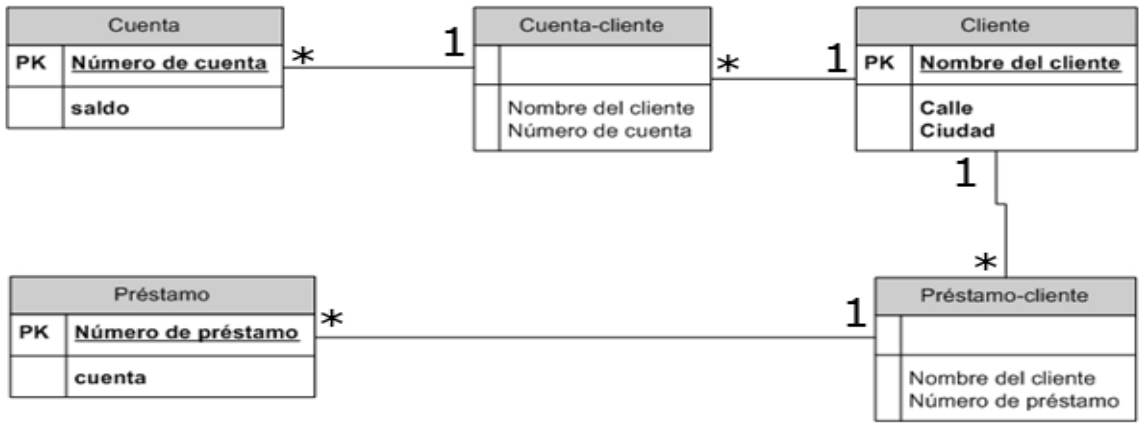


Diagrama 7.2 – Ejemplo de modelo relacional acerca de un banco simple

En el diagrama anterior podemos observar la relación cliente así como sus cuentas de débito y crédito. La explicación de sus detalles gráficos se encuentran en el párrafo

[36] Lógica de primer orden: Sistema formal deductivo derivado de la lógica proposicional, que se caracteriza por el uso de cuantificadores para medir la cantidad de individuos del dominio sobre el cual se está actuando **[42]**.
[37] Tuplas: Matemáticamente es una secuencia finita de elementos. Hablando de una base de datos es un registro individual. Si estamos tratando con una tabla con renglones y columnas, podríamos decir que una tupla es un renglón **[39]**.

anterior. También podemos ver que un cliente puede tener muchas cuentas y préstamos, pero éstas sólo serán poseídas por él mismo. Lo anterior es expresado por el número uno y los asteriscos, que significan muchos, pegados a las relaciones. Las diferentes combinaciones con respecto a la cantidad de elementos del mismo tipo que le corresponden a otro se enlistan a continuación:

Símbolo en un extremo	Símbolo en el otro extremo	Significado
1	1	Uno a uno
1	*	Uno a muchos
*****	*	Muchos a muchos
*****	1	Muchos a uno

Tabla 7.3 – Cardinalidad de asociaciones

El modelo relacional se define como un conjunto de operaciones algebráicas que actúan sobre tablas. Dentro de las operaciones podemos encontrar las que son comúnes sobre conjuntos como unión, intersección y resta. También se encuentran selecciones, proyecciones y combinaciones de operandos. Las consultas a la base de datos se hacen a través de varias operaciones algebráicas de este tipo las cuales pueden ser simples o compuestas. Los operadores en el álgebra relacional tienen un propósito específico, de hecho acciones como inserción, eliminación y actualización de tuplas que se realizan a través del operador de asignación.

Al escribir expresiones de álgebra relacional, componemos una secuencia de procedimientos que generan una respuesta a nuestra consulta. Esto convierte al álgebra relacional en un lenguaje imperativo en donde las expresiones están compuestas por una secuencia de instrucciones que indican paso a paso como resolver un problema. Sin embargo, existe otra forma de escribir consultas en las tablas del modelo relacional. El cálculo relacional contrasta con el concepto imperativo al ser un lenguaje declarativo que sólo describe la información deseada sin proporcionar un procedimiento específico para obtener dicha información **[39]**. Para poder comprender mejor lo anterior, a continuación presentamos una consulta como ejemplo.

$$\{ x \mid C(x) \}$$

Lo anterior regresa como resultado el conjunto de tuplas x tales que el predicado C se cumple para x. Es posible crear consultas mucho más poderosas y versátiles al componer condiciones más complejas en el cálculo relacional. Aunque existen maneras mucho más sencillas de escribir consultas. Por ejemplo a través del uso del lenguaje SQL[38], el cual surge de la necesidad de simplificar al álgebra relacional.

Por otro lado, es importante tener en cuenta que además de contar con la representación a manera de tablas concretas, tenemos a nuestra disposición más herramientas para componer nuestras consultas. Las vistas son una de ellas. Éstas son tablas virtuales creadas por una consulta específica para poder simplificar un objetivo concreto. La nueva tabla virtual puede atender consultas tal y como lo haría una tabla concreta.

7.2 Diseño del sistema

Un diseño es una imagen representativa de lo que necesitamos construir **[46]**. En el contexto del diseño de un sistema, el diseño se divide en cuatro grandes rubros:

- Diseño de datos: Se refiere al diseño del modelo de datos, o dicho de otra forma el diseño de la base de datos.

- Arquitectura: Incluye cual será la estructura más abstracta del funcionamiento del sistema.

- Interfaces: El diseño de interfaces se enfoca directamente a los medios de comunicación entre el sistema y el exterior.

- Diseño de componentes: Incluye a los elementos fundamentales del sistema. En el caso particular del CIMS, todo gira alrededor de los siguientes elementos:

  1. Gestor de clientes. Administra todos los datos de los clientes. Permite las operaciones de vista, altas, cambios y bajas de éstos.

[38] SQL (*Structured Query Language*): Lenguaje estructurado de consultas. Es el lenguaje de consultas más popular del mercado debido a su facilidad de uso y versatilidad **[39]**.

2. Gestor de créditos y movimientos. Es el componente encargado de relacionar los movimientos con los créditos correspondientes al mismo tiempo que administra el vínculo entre ellos.

3. Generador de tablas de amortización. Provee un reporte detallado del crédito en cuestión junto con los movimientos correspondientes.

4. Multireportes. Es el nombre particular del generador de reportes acumulativos parametrizable del CIMS.

En nuestro esfuerzo por diseñar los componentes del sistema nos encontramos valiosas herramientas como UML[39].

Lenguaje de Modelado Unificado (*Unified Modeling Language* UML)

UML es un lenguaje que reúne todas las mejores prácticas de la ingeniería de software para el modelado de sistemas **[44]**. Sus características se presentan a continuación:

- Es un lenguaje completo. Posee la capacidad de capturar y transmitir información sobre un tema con fines de comunicación haciendo uso de su propia sintaxis y semántica.

- Sus usos incluyen visualización, especificación, construcción y documentación de sistemas.

- Está basado en el paradigma orientado a objetos.

- Puede extenderse, es versátil y se rige bajo los estándares actuales de la industria.

- Fomenta y facilita la captura, análisis y difusión de la información en cuestión. Esto permite aumentar la calidad del producto, reducir costos y disminuir el tiempo de comercialización.

[39] UML (*Unified Modeling Language)*: Lenguaje Unificado de Modelado, un lenguaje ampliamente usado en descripciones gráficas de sistemas. Está compuesto por varios tipos de diagramas, reglas semánticas y sintácticas.

Dada la versatilidad del UML, los tipos de diagramas que se pueden crear por medio de este lenguaje son muy diversos. La clasificación de los mismos se encuentra ejemplificada en el diagrama 7.4.

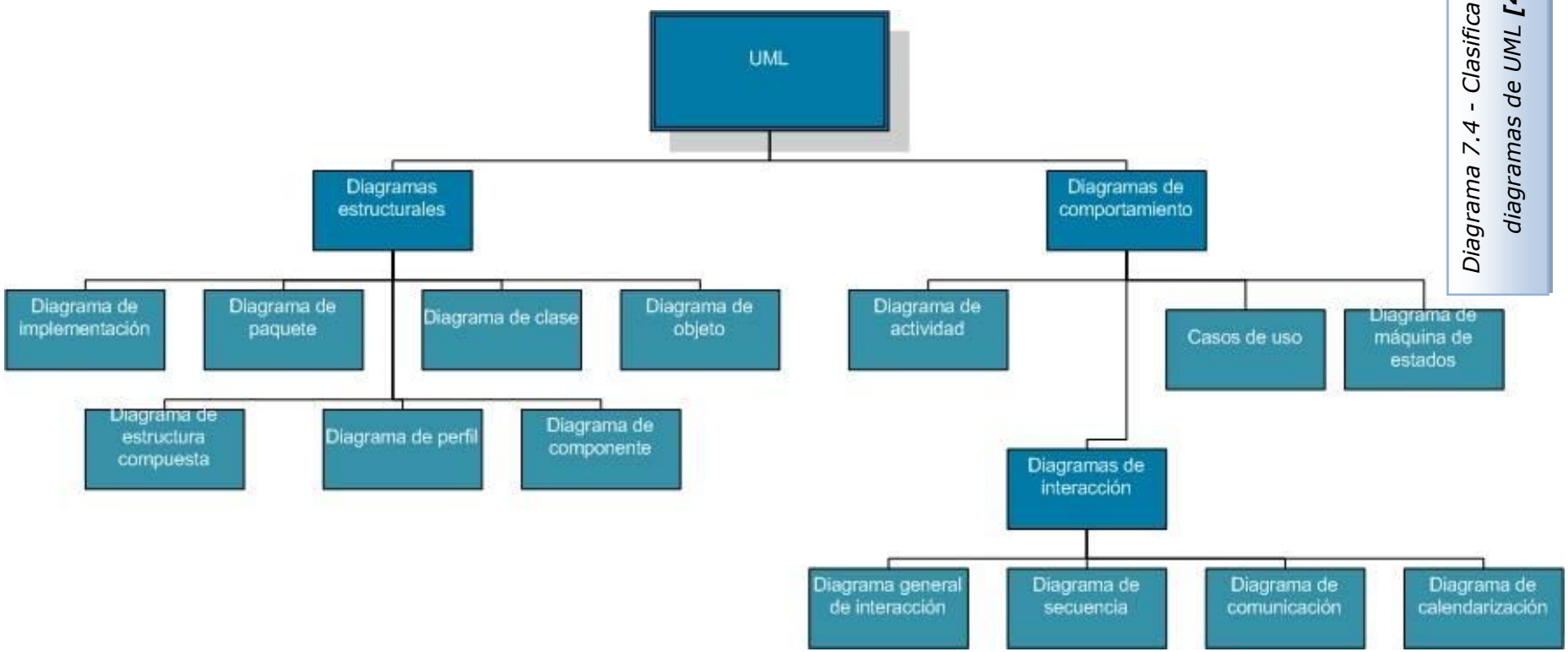


Diagrama 7.4 - Clasificación de diagramas de UML ***[45]***

En la figura anterior podemos apreciar que los diagramas UML pueden ser clasificados principalmente en dos ramas: estructurales y de comportamiento. Dentro de la primera podemos encontrar los diagramas de clases, de paquetes, de componentes, entre otros. Por otro lado, diagramas como los casos de uso y las máquinas de estado se encuentran clasificados como de comportamiento. Resulta interesante observar que la figura anterior expone una subdivisión de los diagramas de comportamiento denominada diagramas de interacción. En esta última podemos encontrar a diagramas de secuencia, de calendarización, entre otros. Como recurso adicional, a continuación se describen los más importantes:

Diagrama de clase

Nos ayudan a describir la estructura de un sistema al representar las clases que integran el mismo. Dicha representación incluye los atributos de las clases así como la relación entre ellas. A groso modo, este tipo de diagramas modelan la estructura estática de un sistema en vez de mostrar cómo se comporta realmente. A continuación se enlistan las propiedades más importantes de los diagramas de clase:

- Contienen clasificadores. Los clasificadores son elementos que describen las funciones estructurales y de comportamiento. Las funciones estructurales incluyen atributos, mientras que las de comportamiento incluyen operaciones y métodos. También es preciso denotar que los clasificadores pueden estar involucrados en asociaciones entre elementos de su mismo tipo.

- Pueden contener objetos, siendo éstos instancias de la misma clase, las cuales expresan configuraciones y atributos específicos. Un diagrama de clase que contiene solamente objetos sin la clase que los representa, es llamado a veces diagrama de objeto.

- Refiriéndonos a la descripción física de los diagramas, podemos decir que un diagrama de clase:

    - Denota a una clase como un rectángulo con un contorno sólido. Dicho rectángulo contiene pequeños "compartimentos" que son usados como contenedores de las características del diagrama en sí.

- Debe poseer un compartimento en el cual se exprese el nombre de la clase (en caso de que la clase sea abstracta, el texto debe de ir en *itálicas).*
- Puede contener varios compartimientos con los atributos de la clase.
- Puede estar compuesta por varios compartimientos con las operaciones de la clase junto con sus métodos.
- Puede comprender más compartimientos que contengan otras propiedades como excepciones y otras reglas de negocio.

En el diagrama 7.5 podemos observar un diagrama de clase que representa la clase del CIMS que calcula el IVA de la comisión del crédito otorgado. El nombre de la misma se encuentra en la parte superior de manera aislada y en negritas. Su único atributo, mostrado justo debajo junto con su tipo es **`resultado`**. Éste guarda el IVA de la comisión una vez calculada. También están los métodos en la parte inferior del diagrama precediendo al tipo de dato que regresan. Éstos son **`operation`** y **`getFormat`**. que calculan el IVA de la comisión y regresan el formato en el cual se debe mostrar la información respectivamente. La clase completa **`ComisionIVA`** así como la explicación detallada de sus dos métodos principales mencionados anteriormente se encuentran en el fragmento 8.42 del capítulo 8.

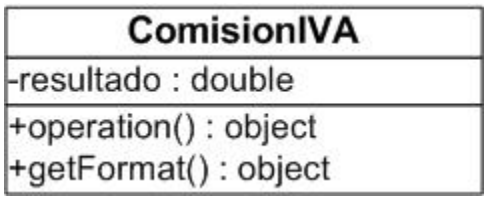


Diagrama 7.5 – Ejemplo de diagrama de clase del CIMS que calcula el IVA de la comisión del crédito otorgado

Diagrama de caso de uso

Este tipo de diagrama muestra la funcionalidad de un sistema desde el punto de vista de actores[40]. Dichos actores realizan actividades dentro del sistema orientadas a objetivos determinados. Los actores pueden estar relacionados con otros objetivos o

[40] Actor: En este contexto, un actor se refiere a un rol que puede ser asumido por usuarios del sistema o por otros sistemas.

actividades. Los casos de uso muestran al sistema desde el punto de vista de un usuario y sus acciones dentro del sistema.

Los casos de uso están integrados por varios elementos gráficos, mismos que serán descritos a continuación:

- Actores

  Definen roles que pueden ser asumidos por usuarios del sistema o por otros sistemas. Las características principales de estos objetos son:

  - Forman parte de interacciones que involucran intercambio de mensajes y acciones con otros sistemas.

  - Físicamente, dentro del diagrama, los actores están representados por el dibujo de una persona.

  - Poseen un nombre o identificador relacionado con el objetivo a realizar dentro del sistema.

  - Pueden estar relacionados con otros actores de características similares siendo éstos una generalización o especialización de otros actores.

- Casos de uso (dentro de un diagrama de caso de uso)

  Son clases que definen unidades de funcionalidad o comportamiento proporcionadas por un sistema. Físicamente están representadas por un óvalo y agrupados dentro de un rectángulo que engloba todos los casos de uso de un sistema o de un subsistema en particular. Los casos de uso poseen las siguientes características:

  - Pueden tener operaciones y atributos.

  - Pueden ser descritos en texto plano o por medio de diagramas de comportamiento.

  - Pueden estar expresados en niveles jerárquicos dependiendo de la importancia del caso de uso.

- Relaciones

  Los casos de uso (dentro de un diagrama) se expresan como líneas entre ellos mismos. Relaciones, las hay de varios tipos, los cuales se enlistan a continuación:

  - Incluyente – Un caso de uso puede incluir a otro. Esto significa que el comportamiento del caso de uso que es incluído se encuentra dentro del otro caso de uso en donde se incluyó al primero.

  - Extensión - Un caso de uso puede extender a otro. Decimos que un caso de uso A extiende al caso de uso B cuando A es una forma especial o extendida de B.

A continuación se encuentra el diagrama 7.6 el cual está representando un caso de uso basado en la operación de Crédito Integral. Podemos observar los elementos principales de este tipo de esquemas como lo son actores, operaciones y relaciones. Ejemplos de actores son el **cliente**, el **promotor** y el **ejecutivo**; mientras que dentro de las operaciones mostradas se encuentran: **pedir crédito**, **realizar transacción**, **dar de alta crédito en el CIMS** e **investigar en buró de crédito**. Las relaciones son flechas que unen acciones entre sí. Por ejemplo el **cliente** es capaz de **pedir crédito** al **promotor**. También puede **pagar amortización**, operación que extiende a **realizar transacción**. Por su parte, el **ejecutivo** capta inversionistas por medio de la operación **captar inversionistas** y obtiene reportes ejecutivos a través de la operación **obtener del CIMS reportes ejecutivos**.

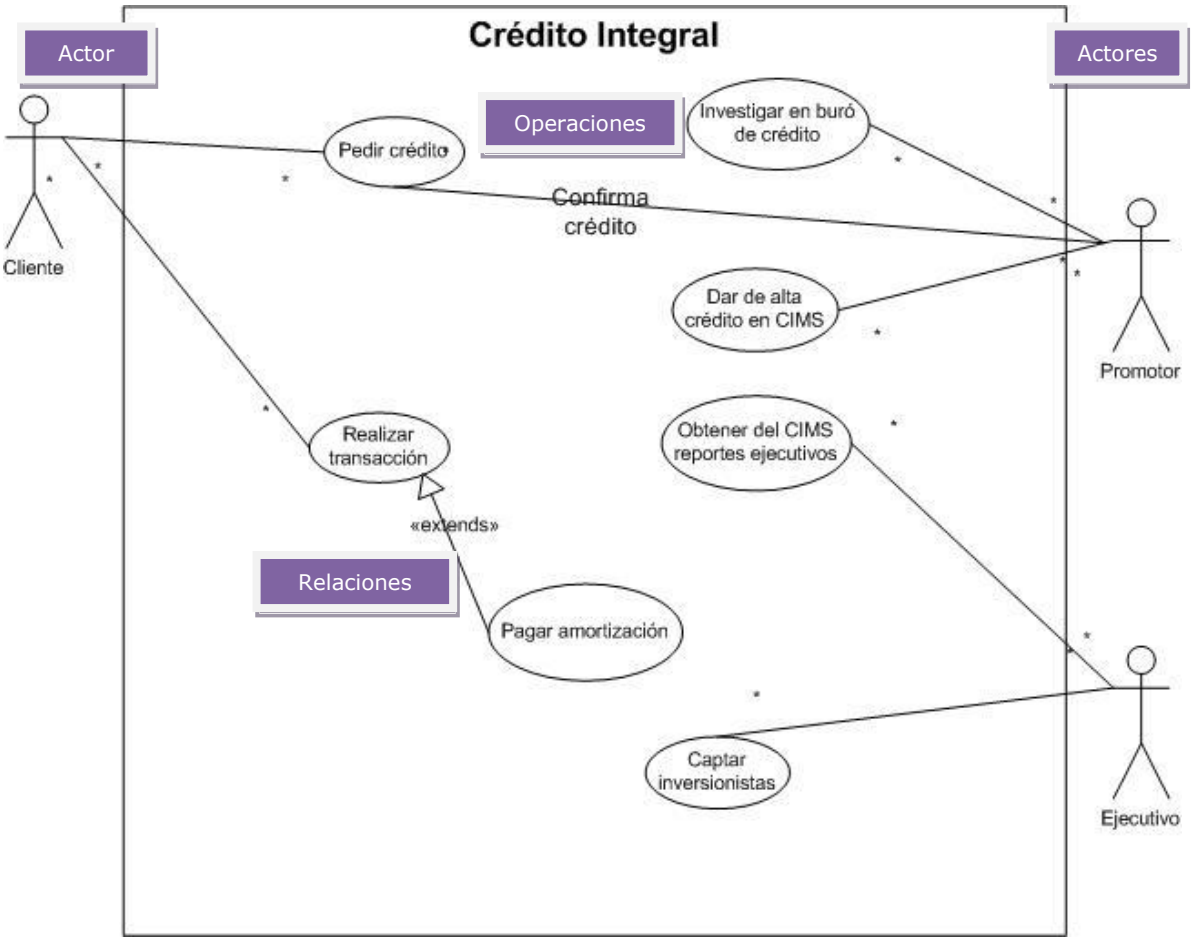


Diagrama 7.6 – Ejemplo de diagrama de caso de uso de Crédito Integral

Diagrama de secuencia

Un diagrama de secuencia es un tipo de diagrama de interacción que muestra como los procesos se relacionan entre sí tomando en cuenta el momento y el orden en el que lo hacen. Los diagramas de secuencia poseen las siguientes características:

- Pueden existir en forma genérica o de instancia. En la primera forma describen el intercambio de mensajes entre clases; mientras que en la segunda, describen un intercambio de mensajes particulares entre objetos que son instancias de esa clase.

- Poseen un elemento llamado "línea de vida" que representa el tiempo en el que un proceso se encuentra activo dentro de la operación.

- Denotan interacción entre los procesos por medio de mensajes. Éstos gráficamente se representan como líneas horizontales. En el caso de que los mensajes sean síncronos las líneas serán sólidas y punteadas en el caso de que sean asíncronos.

- Usan "cajas de activación" para denotar que un proceso es ejecutado como respuesta a un mensaje. Gráficamente estos elementos se encuentran representados por rectángulos.

- Usan una "X" dibujada debajo de la línea de vida para denotar que la vida del proceso en cuestión ha terminado.

El diagrama 7.7 es un ejemplo de este tipo de diagramas, en el cual se pueden ver las actividades que se realizan dentro del CIMS al generar una tabla de amortización. Resulta importante observar la clase **`Gui`**, mostrada en el diagrama 7.7, y sus interacciones con las demás clases. Podemos ver las cajas de activación relativas a distintas interacciones y como éstas se encuentran dentro de la línea de vida. Lo anterior nos ayuda a entender el momento relativo en el que ocurren las acciones y hasta cuando una clase es importante en un contexto determinado.

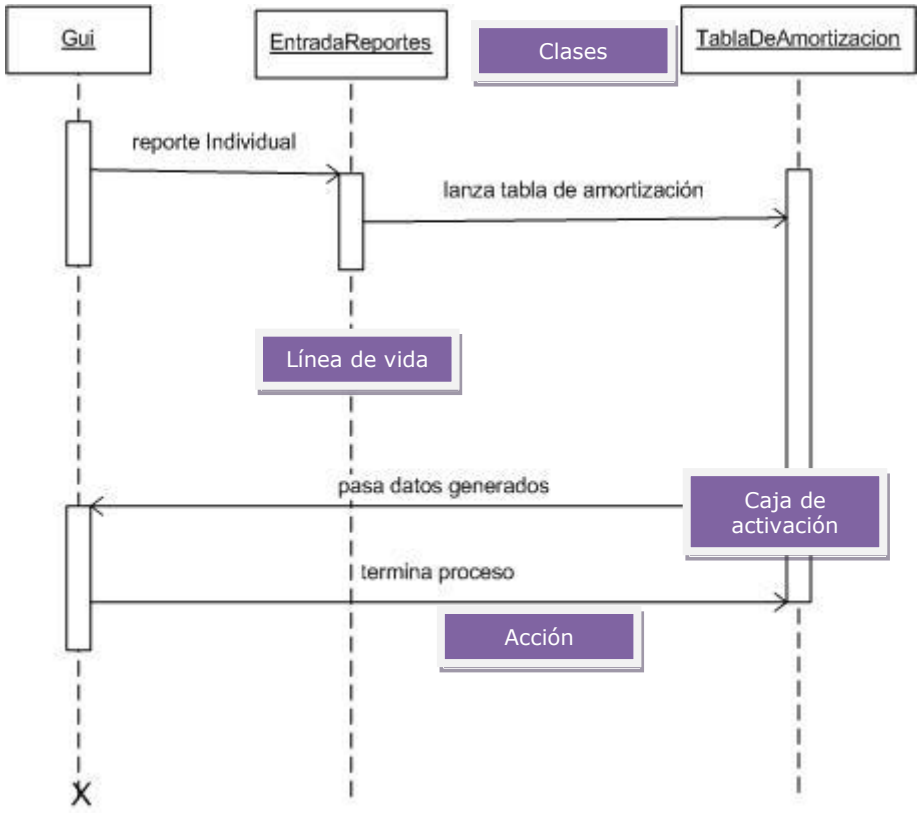


Diagrama 7.7 – Ejemplo de diagrama del CIMS en donde se genera una tabla de amortización

En el diagrama anterior podemos observar lo siguiente: la clase **`Gui`** invoca a la clase **`EntradaReportes`** indicándole que requiere un "reporte individual". Por su lado, esta clase recaba datos del usuario relativos al tipo de reporte deseado, para posteriormente llamar a la clase **`TablaDeAmortizacion`** la cual se encargará de recabar y procesar los datos del reporte. Finalmente la clase **`Gui`** recibe de la clase **`TablaDeAmortizacion`** el reporte y lo muestra.

Capítulo 8

Implementación del CIMS

La creación del CIMS se llevó a cabo a través de un periodo de más de un año. Durante este periodo, varios aspectos del sistema y de la empresa Crédito Integral® fueron madurando conforme el mercado dictaba los lineamientos sobre los cuales debía de operar la empresa. A continuación se presentan los detalles de la implementación del CIMS.

8.1 Implementación de la base de datos

La base de datos fue implementada tomando en cuenta un modelo Entidad-Relación, el cual fue descrito en el capítulo anterior. A fin de poder explicar este proceso, los diagramas más importantes se encuentran a continuación. Cabe mencionar que esta sección no incluye todos los diagramas de la base de datos debido a la extensión limitada de este documento.

El CIMS gira alrededor de una SOFOM y la actividad preponderante de ésta es el otorgamiento de créditos **[13]**. Por lo tanto, no debe resultar sorprendente el hecho de que el elemento principal de la base de datos y de la empresa en general es el cliente. La tabla que representa a la clase Cliente se encuentra ilustrada en el diagrama 8.1.

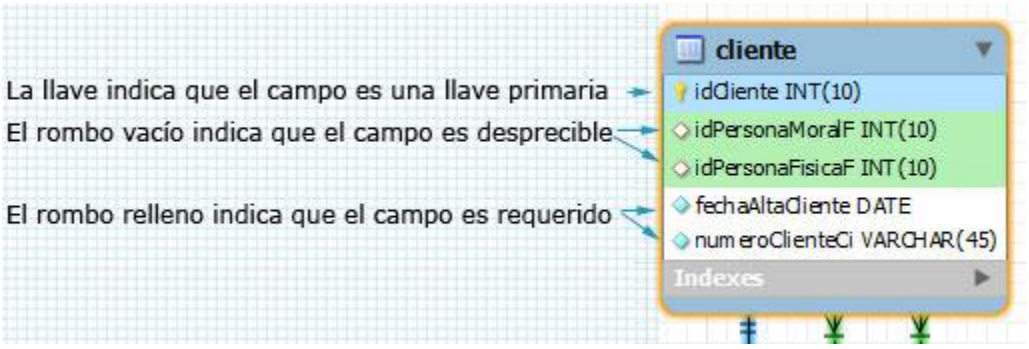


Diagrama 8.1 – Tabla cliente

El cliente puede ser persona física o persona moral; ambas son tablas en la base de datos. Resulta importante aclarar que el cliente tiene dos formas de ser identificado, una es por medio del identificador interno de la base de datos (**`idCliente`**) y la otra es a través del identificador propio de Crédito Integral® (**`numeroClienteCi`**).

Teniendo en cuenta lo anterior, procedemos a explicar un poco más a fondo los dos elementos principales que proveen funcionalidad de recabar información acerca de los principales tipos de clientes: persona moral y persona física.

8.1.1 Persona moral

Persona moral se refiere a una empresa y, dado que el otorgamiento de créditos es inherentemente riesgoso para cualquier institución que preste este servicio **[49]**, nos interesa recabar la mayor cantidad de información útil acerca de nuestro cliente a fin de evitar prestarle a quien no debemos. Claro está que además de los privilegios fiscales con los que cuenta una SOFOM (explicado en capítulos anteriores), ésta se encuentra facultada para consultar en el buró de crédito la información del cliente y a su vez procesar legalmente a aquellos morosos. A fin de poder proceder legalmente en contra de los integrantes de la cartera vencida y de calcular con exactitud el riesgo crédito, se decidió recabar bastante información de los clientes. El diagrama 8.2 es un ejemplo de todos los datos que se recaban acerca de ellos. Particularmente de los clientes de tipo persona moral.

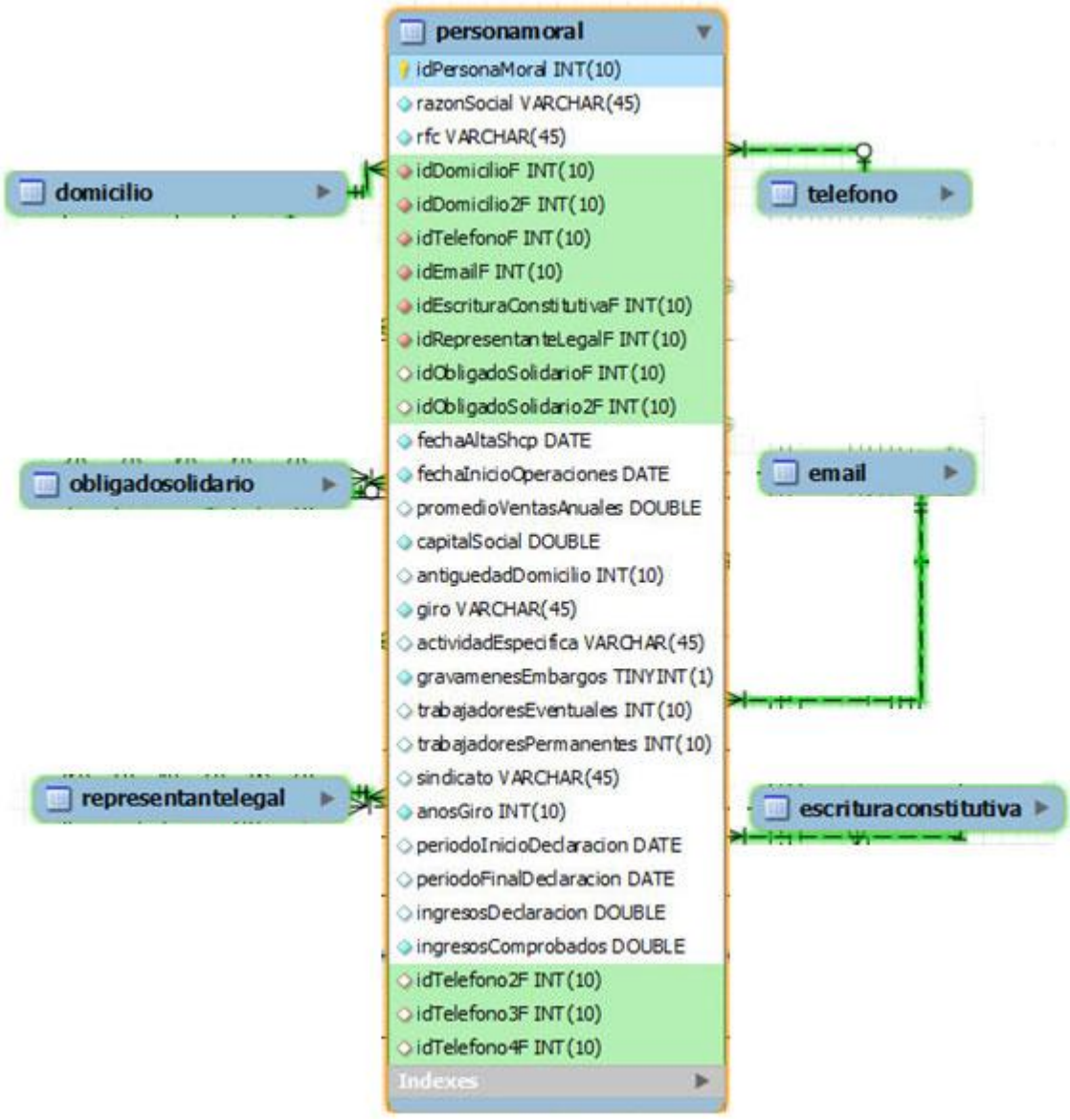
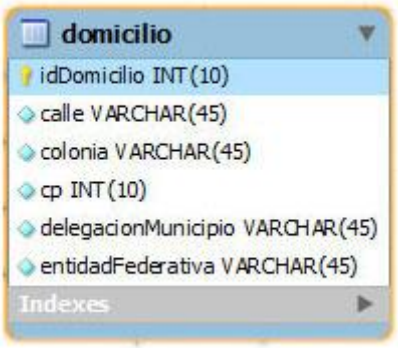


Diagrama 8.2 – Persona moral

El diagrama anterior muestra como algunos datos recabados de las personas morales son parte de la tabla misma. En la figura anterior y en todas las demás de base de datos son mostrados con fondo blanco. Este es el caso de la razón social (**`razonSocial`**), el RFC (**`rfc`**), la fecha de alta en la SHCP (**`fechaAltaShcp`**), entre otros. Sin embargo existen otros datos que son agrupados como una tabla más en la base de datos con el fin de volverlos a usar, en caso de ser necesario. Estos datos son los siguientes:

Domicilio fiscal

Domicilio registrado ante la SHCP.

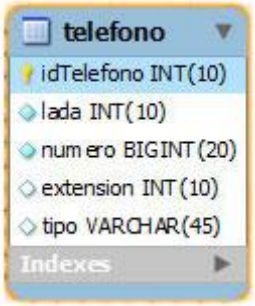


- Calle.
- Colonia.
- Código postal.
- Delegación o municipio dependiendo de si se encuentra el domicilio en el Distrito Federal o no.
- Entidad federativa.

Diagrama 8.3 – Domicilio

Teléfono

Número telefónico con lada y extensión.



- Lada.
- Número.
- Extensión.
- Tipo.

Diagrama 8.4 – Teléfono

Correo electrónico

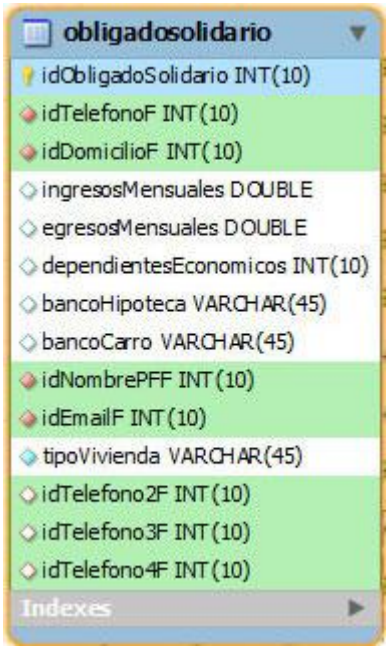
Dirección de correo electrónico con campo para notas.

- Correo electrónico (**email**)
- Nota.

Diagrama 8.5 – Correo electrónico

Obligado solidario

Aval en caso de que el crédito y sus intereses no sean cubiertos por el deudor.



- Teléfono. Usamos la entidad del diagrama 8.4.
- Domicilio. Usamos la entidad del diagrama 8.3.
- Ingresos mensuales.
- Egresos mensuales.
- Dependientes económicos
- Banco en donde el obligado solidario tiene una hipoteca.
- Banco en donde el obligado solidario adeuda un auto.
- Datos complementarios de un ciudadano[41] como sexo, estado civil, entre otros (**idNombrePFF**).
- Correo electrónico (**idEmailF**).
- Tipo de vivienda. Esta puede ser rentada, propia o hipotecada.
- Teléfono.

Diagrama 8.6 – Obligado solidario

[41] Ciudadano (como término jurídico): Hombres y mujeres mexicanos que tengan un modo honesto de vivir y hayan cumplido 18 años **[129]**.

Representante legal

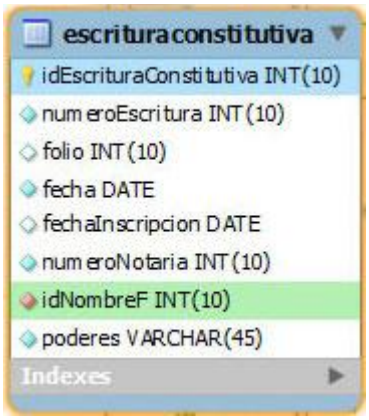
Persona física que es responsable legalmente de la empresa.

- Datos complementarios de un ciudadano como sexo, estado civil, entre otros (**idNombrePFF**).
- Teléfono.
- Correo electrónico (**idEmailF**).
- Escritura constitutiva.

Diagrama 8.7 – Representante legal

Escritura constitutiva

Persona física que es responsable legalmente de la empresa.



- Número de escritura constitutiva.
- Folio.
- Fecha en la que se expidió el documento.
- Fecha de inscripción en la SHCP.
- Número de notaría.
- Nombre del notario (**idNombreF**).
- Poderes / facultades otorgadas por medio de dicho documento (**poderes**).

Diagrama 8.8 – Escritura constitutiva

Es importante destacar que la persona moral, puede tener varios teléfonos, un domicilio fiscal y otro físico, y un correo electrónico, también posee (como figura central y responsable de la empresa) un representante legal. Sin embargo, en caso de que esta persona sea incapaz de responsabilizarse por algún inconveniente jurídico,

legalmente podemos proceder en contra de los dos obligados solidarios[42], que en este caso fungen como avales de la empresa. Cabe mencionar que tenemos una referencia al "acta de nacimiento" de la empresa que realmente es la escritura constitutiva[43]. Este documento nos confirma quién es el representante legal de la empresa, así como otros datos importantes como lo son los integrantes de la empresa, el objeto de la misma, duración, importe exacto de su capital social, las aportaciones de cada socio, facultades de cada uno, domicilio, forma en la que se distribuyen pérdidas y ganancias entre cada integrante, importe del fondo de reserva, casos en los que la sociedad se disolvería anticipadamente, entre otros **[51]**. Por otro lado, Crédito Integral® también otorga créditos a personas físicas. Veamos las características de este tipo de cliente en la siguiente sección.

8.1.2 Persona física

Persona física se refiere a un individuo capaz de adquirir derechos y obligaciones de manera individual y no como sociedad o asociación **[52]**. Dicho de otra forma, una persona física es una persona individual o natural que es sujeto de derecho. A pesar de los esfuerzos del personal directivo de Crédito Integral® por simplificar trámites para el otorgamiento de créditos para personas físicas, la cantidad de información requerida a este tipo de clientes también es cuantiosa, lo anterior es debido a la naturaleza del riesgo crédito (desarrollado en el capítulo 4). La información requerida a las personas físicas se encuentra resumida en el diagrama 8.9.

[42] Obligado solidario: Aval que se solidariza con un deudor en el contexto de otorgamiento de un crédito al cubrir el monto adeudado en caso de que el deudor no lo haga **[50]**.
[43] Escritura constitutiva: Documento público establecido permanentemente por un notario público dentro del registro del mismo que contiene el contrato social y estatutos de la sociedad o asociación en cuestión **[51]**.

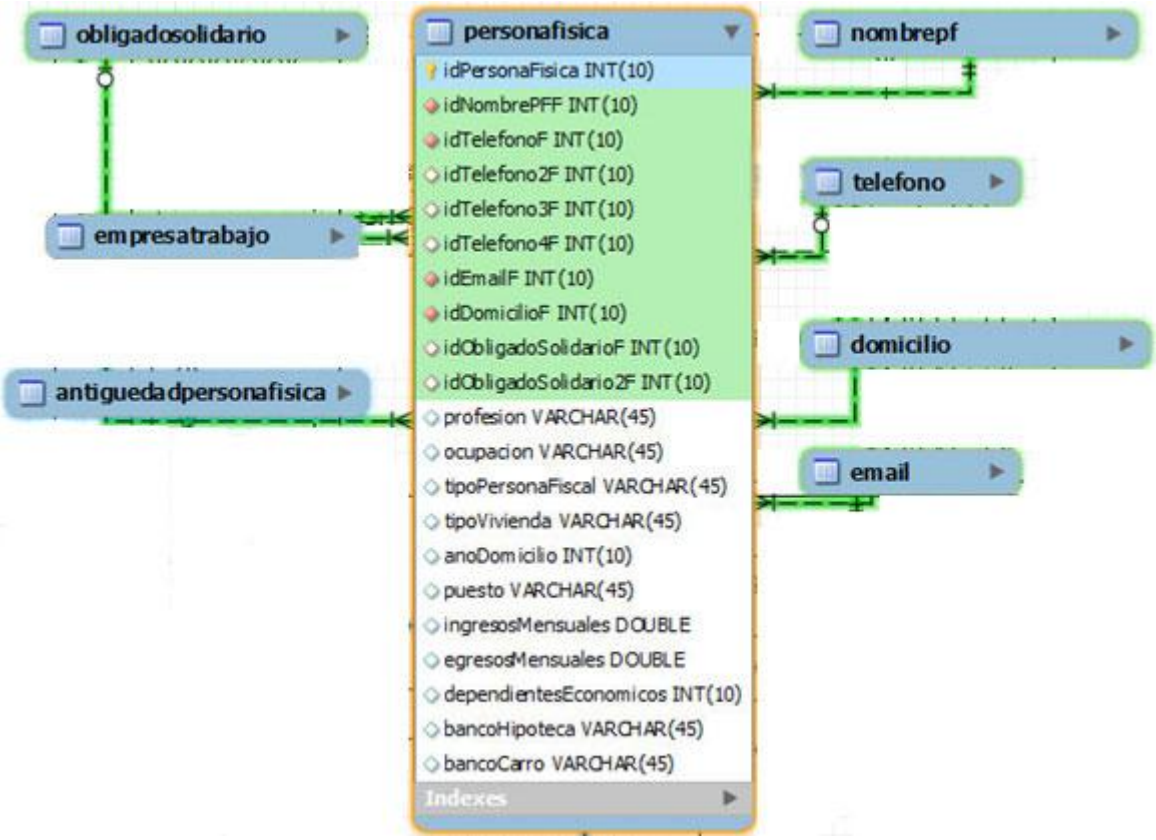


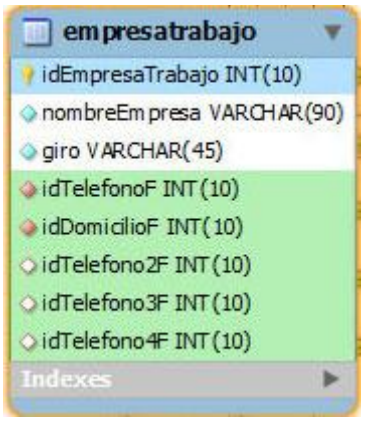
Diagrama 8.9 – Persona física

Observemos que la clase **persona física** contiene varios datos importantes y básicos como lo son teléfonos, correo electrónico, domicilio, nacionalidad, RFC, CURP, entre otros. También, al igual que la persona moral, existen dos obligados solidarios (**idObligadoSolidarioF** y **idObligadoSolidario2F**) que cumplen con la función de responsabilizarse del crédito en caso de que el deudor principal no lo haga.

Resulta importante tomar en cuenta que algunos datos como domicilio, obligado solidario y teléfono son comúnes tanto para persona física como para persona moral. Siguiendo los principios de reutilización de componentes, los datos que son comúnes en las dos clases de clientes son representados por las mismas tablas. Este es el caso del domicilio (diagrama 8.3), el teléfono (diagrama 8.4), el correo electrónico (diagrama 8.5) y del obligado solidario (diagrama 8.6). De esta forma evitamos repetir información y creamos una base de datos normalizada. Los que no son requeridos por **Persona moral**, pero sí por **Persona física**, se detallan a continuación:

Empresa trabajo

Es la empresa en donde labora el deudor.



- Nombre de la empresa.
- Giro.
- Teléfono.
- Domicilio.

Diagrama 8.10 – Empresa trabajo

Antigüedad Persona física

Es una tabla que une a la Persona física en cuestión con la empresa en donde labora.

- Llave foránea que hace referencia a la empresa en donde trabaja el deudor (**`idEmpresaTrabajoF`**).
- Llave foránea que hace referencia a la Persona física que labora en la empresa en cuestión (**`idPersonaFisicaF`**).
- Tiempo en meses de haber laborado.

Diagrama 8.11 – Antigüedad Persona física

Datos complementarios de un ciudadano

Información adicional de un ciudadano en términos jurídicos.

- Estado civil.
- Fecha de nacimiento.
- Sexo.
- Nacionalidad.
- Identificación (por ejemplo credencial de elector, pasaporte vigente o cédula profesional).
- Número / folio de identificación.
- Registro Federal de Contribuyentes (**`rfc`**).
- Clave Única de Registro de Población (**`curp`**).
- Nombre completo (**`idNombreF`**).
- Régimen bajo el cual se encuentra casado en caso de que lo esté. Puede ser separación de bienes, en donde cada quien es propietario de lo que adquiere, o bienes mancomunados, en donde la propiedad de los bienes se reparte equitativamente.

Diagrama 8.12 – Datos complementarios de un ciudadano

Habiendo detallado los datos requeridos para los distintos tipos de clientes, veamos los pormenores de su antecesor: cliente.

8.1.3 Cliente

Como ya lo habíamos mencionado anteriormente, el cliente es la entidad más importante alrededor de la cual giran todos los procesos importantes de la SOFOM. De hecho Crédito Integral® maneja a personas morales y físicas como un mismo objeto a través de la entidad cliente. El diagrama 8.13 nos muestra lo descrito anteriormente.

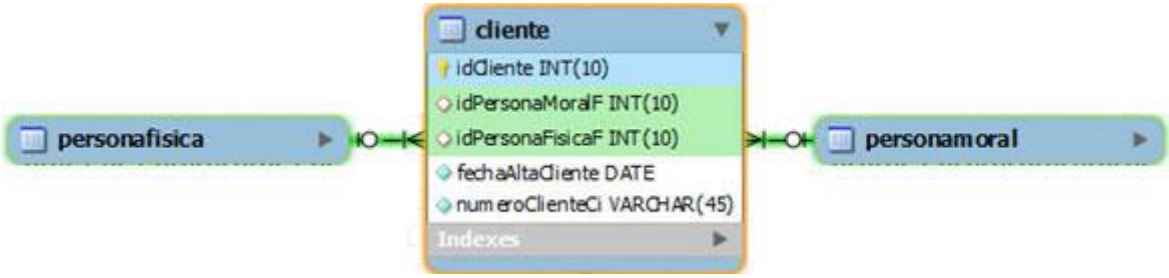


Diagrama 8.13 – Entidad cliente y su relación con persona moral y física

La entidad cliente posee dos llaves foráneas **`idPersonaMoralF`** e **`idPersonaFisicaF`** que representan las llaves primarias de **`personaMoral`** y **`personaFisica`** respectivamente. Lo anterior crea una liga que permite manejar a todos los diferentes tipos de clientes como si estuvieran en la misma categoría (haciendo uso de sus llaves foráneas que hacen referencia al tipo de cliente específico). La tabla **`cliente`** posee los siguientes atributos que muestra la tabla 8.14:

Atributo	Descripción
idCliente	Llave primaria interna
idPersonaMoralF	Llave foránea de persona moral
idPersonaFisicaF	Llave foránea de persona física
fechaAltaCliente	Fecha en la que se dio de alta el cliente en el CIMS
numeroClienteCI	Identificador de cliente dentro de Crédito Integral®

Tabla 8.14 – Atributos de la entidad cliente

8.1.4 Crédito

La actividad preponderante de Crédito Integral® es el otorgamiento de créditos a sus clientes. Alrededor de la entidad homóloga giran las acciones relacionadas con dicha actividad. Dentro del conjunto de acciones asociadas podemos mencionar abonos al crédito y cargos extras al mismo. También están ligados a la entidad las fechas de amortización correspondientes que nos indican cuándo es el momento en que el cliente debe de abonar la amortización correspondiente. El diagrama 8.15 expone a la entidad crédito y a sus principales entidades relacionadas.

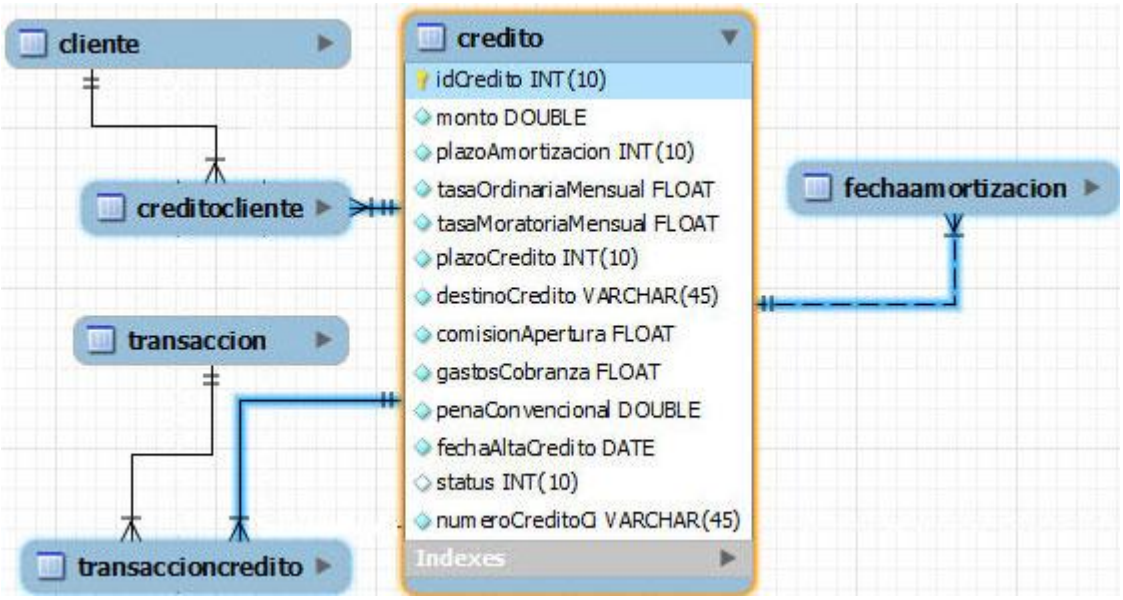
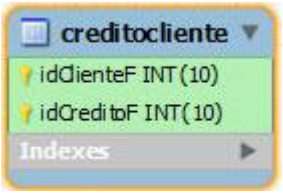


Diagrama 8.15 – Entidad crédito y principales entidades relacionadas

Analizando el diagrama anterior podemos ver cómo, el crédito está relacionado a sus fechas de amortización y sus transacciones como lo mencionábamos en el párrafo anterior (además de estar ligado a su deudor). Los detalles de estas entidades relacionadas se encuentran descritas a continuación:

Crédito cliente

Es una tabla que une al crédito en cuestión con su acreedor.



- Llave foránea que hace referencia al acreedor del crédito (**`idClienteF`**).
- Llave foránea que hace referencia al crédito en cuestión (**`idCreditoF`**).

Diagrama 8.16 – Crédito cliente

Transacción crédito

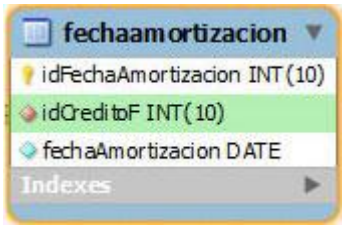
Es una tabla que une al crédito en cuestión con sus transacciones asociadas. Por ejemplo pagos, cargos extras o ajustes.

- Llave foránea que hace referencia a una transacción en específico (**`idTransaccionF`**).
- Llave foránea que hace referencia al crédito en cuestión (**`idCreditoF`**).

Diagrama 8.17 – Transacción crédito

Fecha de amortización

Es una tabla que une al crédito en cuestión con la fecha de amortización actual.



- Llave foránea que hace referencia al crédito en cuestión (**`idCreditoF`**).
- Día, mes y año de la fecha de amortización.

Diagrama 8.18 – Fecha de amortización

Transacción

Representa una operación financiera aplicada a un crédito en específico.

- Monto.
- Concepto.
- Tipo: (1) abono (2) cargo.
- Fecha en la que ocurre la transacción.

Diagrama 8.19 – Transacción

Básicamente sólo tomando en cuenta las fechas de amortización del crédito y sus transacciones, podemos crear casi todos los reportes del CIMS. La tabla de amortización, de la cual hablaremos más adelante, toma como entrada principalmente los datos mencionados con anterioridad. Posteriormente la tabla de amortización se consulta de distintas maneras para poder crear reportes más complejos que forman parte del **`Multireportes`** (descrito en capítulos anteriores).

8.2 Implementación de componentes principales del CIMS

El CIMS fue desarrollado totalmente en Java, un lenguaje de programación orientado a objetos, robusto, popular, libre, distribuido y multiplataforma. Java y sus diversas bibliotecas han posibilitado y facilitado enormemente el desarrollo del CIMS al proveer una plataforma robusta, confiable y fácil de usar. También proporcionan múltiples elementos reutilizables que aceleran el proceso de desarrollo. A continuación se enlistan los componentes principales y una breve descripción de la implementación de las rutinas más importantes.

8.2.1 Pantalla de bienvenida e inicialización

Al ejecutar el CIMS, lo primero que se muestra es la pantalla de bienvenida. Sin embargo, antes de que esta pantalla se muestre, el CIMS ejecuta ciertas rutinas de inicialización. Éstas realizan tareas diversas como conexión a base de datos y dibujado de interfaces gráficas. A continuación se muestra dichas tareas de una manera un poco más detallada.

Tareas de inicialización.

1. Inicialización de componentes gráficos.

```
1     Splash__Screen sc = new Splash__Screen(new Frame());
2     initComponents();
```

Fragmento 8.20 – Clase **Gui**

Respecto a estas dos últimas líneas de código, cabe observar que antes de que se inicialicen los componentes gráficos por medio del método `initComponents()`, se crea un *splash screen*[44]. Esto nos permite mostrar una imagen al usuario antes de que se carguen todos los componentes gráficos de la aplicación. Sin embargo, hay que recordar que Java corre sobre una máquina virtual la cual debe estar cargada. También deben ser cargadas bibliotecas como Swing[45] o AWT[46] dependiendo de la

[44] Splash screen: Una ventana simple que aparece brevemente en el tiempo en el que una aplicación es lanzada por completo y la ventana principal de la misma es mostrada **[53]**.
[45] Swing: Un conjunto de componentes de interfaces gráficas que permiten desarrollar una apariencia parametrizable **[53]** .

aplicación **[54]**. Todo lo anterior toma tiempo y el objetivo principal de un *splash screen* es poderle mostrar al usuario una imagen que le indique que su programa ya se está ejecutando. Afortunadamente este problema ya ha sido resuelto por Sun® a partir de la versión SE 6 de Java. La solución consiste en la incorporación de un lanzador de aplicaciones Java capaz de decodificar imágenes aún antes de que se cargue la máquina virtual. El resultado es que el usuario puede visualizar el *splash screen* casi de inmediato. La mejor parte es que es bastante sencillo añadir esta función a una aplicación. Existen varias maneras de hacerlo, sin embargo CIMS implementa esta característica por medio del archivo manifest.mf de la siguiente manera:

```
1    Manifest-Version: 1.0
2    X-COMMENT: Main-Class will be added automatically by build
3    SplashScreen-Image: img/cims.jpg
```

Fragmento 8.21 – Archivo **`manifest.mf`**

Sí, basta con decirle donde está la imagen que quieres usar como *splash screen* para que funcione. Tras bambalinas, esta implementación del *splash screen* es más sofisticada porque es capaz de mostrar la imagen de espera mucho antes de que se cargue la máquina virtual de Java. Operación que puede tardar varios segundos dependiendo de la máquina ejecutando el programa. Finalizamos este tema aclarando que la línea 1 del fragmento 8.20 de las tareas de inicialización, la cual hace referencia a un objeto de tipo `Splash_Screen`, funciona en caso de que no estemos usando la versión SE 6 de Java o superior **[54]**. La figura 8.22 nos muestra como se ve la imagen.

[46] AWT *Abstract Window Toolkit*: Una biblioteca de clases que permite crear interfaces gráficas para aplicaciones de Java. Muchos de los componentes en ese conjunto de clases han sido reemplazados por aquellos de Swing **[53]**.

*Figura 8.22 – Splash screen (**cims.jpg**)*

La línea 107 del mismo fragmento, `initComponents()`, es un método típico dentro de NetBeans[47] que inicializa los componentes gráficos. De hecho, como programadores, no tenemos el control directo del contenido de dicho método. Las instrucciones se agregan por medio de Matisse, el editor de GUI[48]s de NetBeans.

2. Conexión a bases de datos.

```
1131   session = new Session();
1132   session.logIn();
1133   if (session.entityManager!=null) {
```

[47] NetBeans: un *Integrated Development Environment* o entorno de desarrollo integrado para desarrollar Java, JavaScript, PHP, Python, Ruby, Groovy, C, C++, Scala y Clojure **[55]**.
[48] GUI *Graphical User Interface*: Interfaz Gráfica del Usuario. Una interfaz para el usuario de un software basada en íconos, imágenes y menús en vez de texto puro **[41]**.

```
1134        Utilities.aviso(Aviso.LOGINBIEN,true);
1135    }
```

Fragmento 8.23 – Clase **Gui**

Ahora veamos el método `logIn()`

```
24      public void logIn(){
25            entityManager =
        javax.persistence.Persistence.createEntityManagerFactory("CreditoIntegral
        PU").createEntityManager();
26      }
```

Fragmento 8.24 – Clase **Session**

Respecto a las líneas anteriores, resulta importante notar que CIMS usa el JPA[49]. Este API [50] permite a los desarrolladores gestionar información relacional usando las plataformas Standard y Enterprise de Java. Existen varias implementaciones del JPA actualmente en el mercado, siendo las más populares Hibernate de RedHat y TopLink Essentials de Oracle ambas de código abierto **[56]**. Estas dos últimas son más que implementaciones del JPA, también cumplen la función de ORM[51]. Este tipo de software actúa como intérprete al transformar automáticamente objetos de datos dentro de la aplicación a grupos de valores más sencillos para ser manipulados directamente dentro del SMBD **[60]**. CIMS particularmente usa TopLink Essentials debido a su facilidad de uso e integración transparente con NetBeans. Es tan fácil de usar que basta con escribir un archivo de configuración XML para que funcione. A continuación se encuentra un fragmento de dicho archivo en el cual se declaran las clases que van a ser administradas por Toplink y cuál será la forma en la que vamos a establecer la conexión a la base de datos.

[49] JPA *Java Persistence API*: Un *framework* de Java que permite gestionar información relacional **[57]**.
[50] API *Application Programming Interface*: Una interfaz implementada por un software para permitir la interacción con otro **[58]**.
[51] ORM *Object Relational Mapping*: Mapeo relacional de objetos. Técnica de programación usada para convertir información incompatible dentro de lenguajes de programación basados en objetos y Sistemas Manejadores de Bases de Datos **[60]**.

```
<persistence version="1.0" xmlns="http://java.sun.com/xml/ns/persistence"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="http://java.sun.com/xml/ns/persistence
http://java.sun.com/xml/ns/persistence/persistence_1_0.xsd">
<persistence-unit name="CreditoIntegralPU" transaction-
type="RESOURCE_LOCAL">
    <provider>oracle.toplink.essentials.PersistenceProvider</provider>
    <class>entityClasses.Antiguedad</class>
    <class>entityClasses.Antiguedadpersonafisica</class>
    <class>entityClasses.Cliente</class>
    <class>entityClasses.Credito</class>
    <class>entityClasses.Creditocliente</class>
    <class>entityClasses.Domicilio</class>
    <class>entityClasses.Email</class>
    <class>entityClasses.Empresatrabajo</class>
    <class>entityClasses.Escrituraconstitutiva</class>
    <class>entityClasses.Fechaamortizacion</class>
    <class>entityClasses.Nombrepf</class>
    <class>entityClasses.Nombresimple</class>
    <class>entityClasses.Obligadosolidario</class>
    <class>entityClasses.Personafisica</class>
    <class>entityClasses.Personamoral</class>
    <class>entityClasses.Referenciabancaria</class>
    <class>entityClasses.Referenciapersonal</class>
    <class>entityClasses.Representantelegal</class>
    <class>entityClasses.Telefono</class>
    <class>entityClasses.Transaccion</class>
    <class>entityClasses.Transaccioncredito</class>
    <class>entityClasses.Listaentidadfederativa</class>
    <class>entityClasses.Usuario</class>
    <property name="toplink.jdbc.url"
value="jdbc:mysql://localhost:3306/cims"/>
      <property name="toplink.jdbc.driver" value="com.mysql.jdbc.Driver"/>
    </properties>
```

Fragmento 8.25 – Archivo **persistence.xml**

A continuación se encuentra la explicación del fragmento anterior, las líneas 1-4 fungen como encabezado del archivo XML. En las líneas 5-6 se declara el nombre con el que se va a hacer referencia a este archivo dentro de las clases de Java. La línea 7 expresa la clase del proveedor de persistencia. Dentro de las líneas 8-13 se encuentran los nombres de algunas de las clases más importantes que serán gestionadas por Toplink. Las líneas 29-32 declaran ciertos parámetros de la conexión como la dirección del servidor y el nombre del controlador de la base de datos que se encargará de habilitar el enlace. Prácticamente la líneas anteriores son suficientes para que TopLink gestione los objetos relacionados a la persistencia de datos.

Al terminar las rutinas de inicialización, CIMS nos enseña el menú principal de forma convencional. Las operaciones principales del software son accesibles desde este menú. La siguiente figura (8.24) nos muestra como se ve esta pantalla.

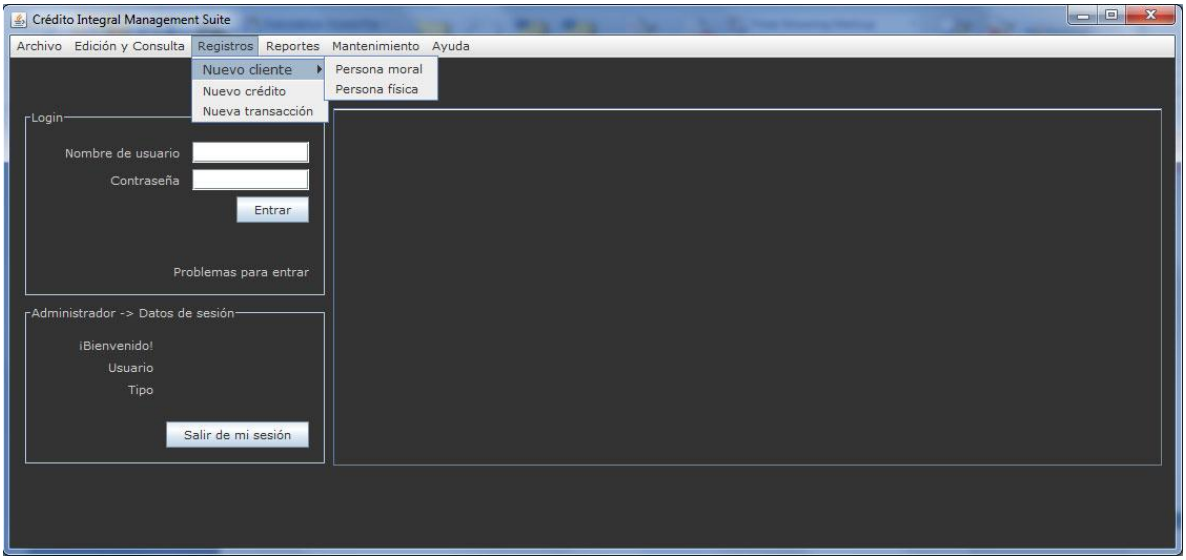


Figura 8.26 – Pantalla principal del CIMS

8.2.2 Captura y mantenimiento de cartera de clientes

La gestión de la cartera de clientes es una parte importante del CIMS. Los procesos de apertura y mantenimiento de clientes se realizan de forma muy similar debido a que ambos utilizan las mismas pantallas. Las actividades anteriores están divididas en varios pasos. Mismos que están agrupados por medio de un componente llamado

wizard[52]*.* Respecto a las opciones de implementación, el *wizard* provee distintas opciones dependiendo del nivel de personalización que requiramos. Dentro de CIMS, usamos la implementación que involucraba la utilización de un *factory*[53] que recibía como parámetros las clases de las pantallas involucradas en cada paso del proceso. A continuación se encuentra una parte del código que realiza justo esto.

```
90        wiz = WizardPage.createWizard(new Class[] {
91        wizardpages.personamoral.Cliente.class,
92        wizardpages.personamoral.PersonaMoral.class,
93        wizardpages.personamoral.DomicilioSocial.class,
94        wizardpages.personamoral.DomicilioFiscal.class,
95        wizardpages.personamoral.Telefono.class,
96        wizardpages.personamoral.ReferenciaBancaria.class,
```

Fragmento 8.27 – Clase **PersonaMoralWizard**

Del fragmento anterior, la línea 90 crea una instancia de tipo **Wizard** a la que se van a agregar las pantallas del componente. Cada una de éstas representa un paso en el proceso de alta y mantenimiento de clientes. Las líneas 91-96 son simplemente algunos de los nombres de las clases que representan las pantallas de dichos pasos.

Al final, sólo hacía falta llamar otro método del *wizard* que gestiona las pantallas de acuerdo al paso en cuestión dependiendo del proceso específico que esté administrando el componente . Éste es el siguiente.

```
120     result = WizardDisplayer.showWizard(wiz);
```

Fragmento 8.28 – Clase **PersonaMoralWizard**

Cada pantalla recibe un objeto de datos POJO[54], realiza las modificaciones pertinentes dependiendo de la información que el usuario haya proporcionado y finalmente lo manda a una instancia de la clase **EntityManager** de TopLink, la cual gestiona la

[52] Wizard: Es un conjunto de pasos representados por paneles en la interfaz gráfica de un software **[61]**.
[53] Factory: Un patrón de diseño usado en el desarrollo de software para encapsular el proceso de creación de objetos **[62]**.
[54] POJO *Plain Old Java Object*: Objeto de Java simple. Es un nombre que se le da a los objetos de Java que no siguen ninguna convención, modelo o framework **[63]**.

interacción con el Sistema Manejador de Bases de Datos, para que se mantengan los cambios en el SMBD.

8.2.3 Búsqueda de clientes

Los clientes, además de ser dados de alta y editables, deben poder ser buscados por medio de distintos criterios. Dentro de CIMS, se creó una pantalla que captura distintos datos, los cuales son usados dentro de consultas al SMBD. En la siguiente figura se muestra dicha pantalla.

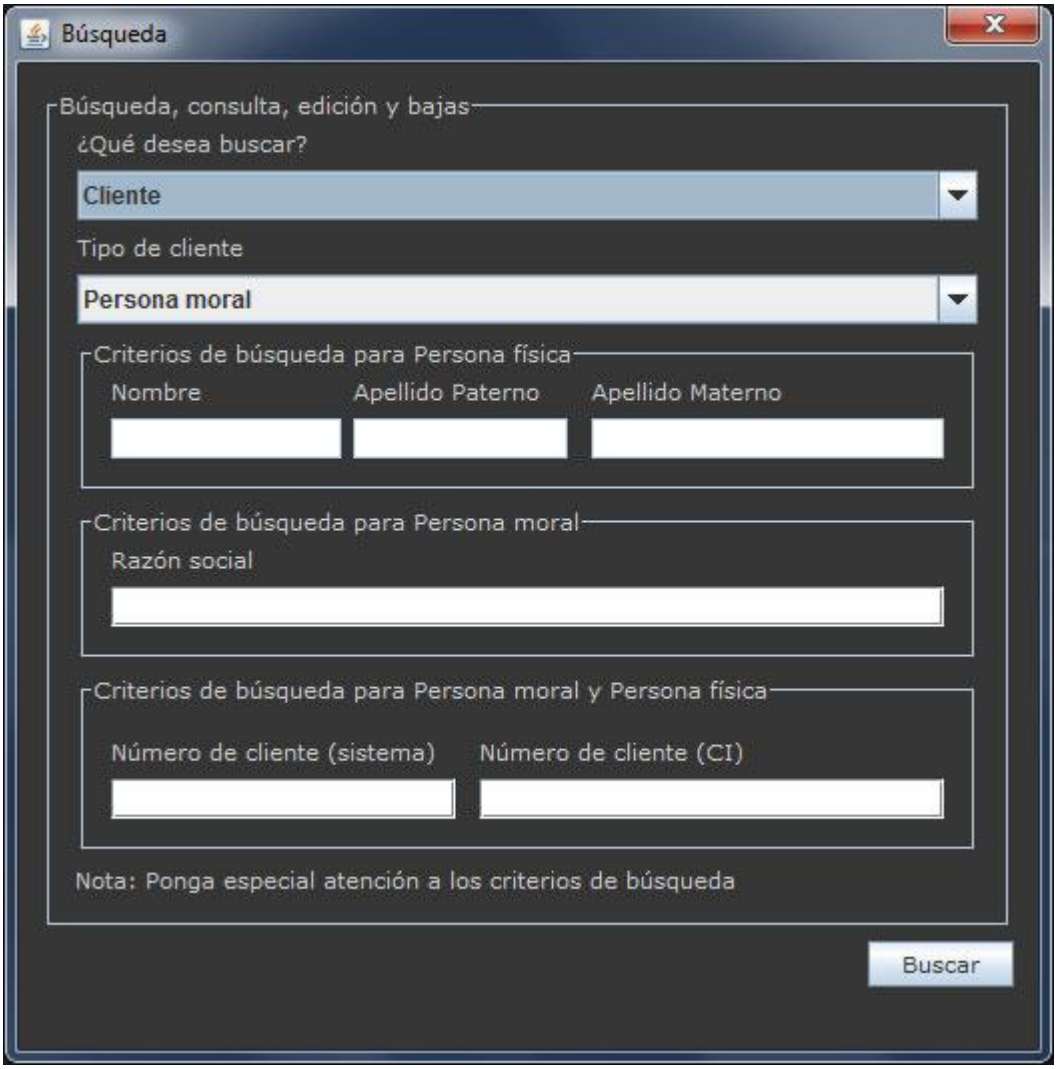


Figura 8.29 – Pantalla de búsqueda

Las consultas ejecutadas por la pantalla anterior no son escritas en SQL tradicional, sino en JPQL[55] que es el lenguaje de consultas a bases de datos que el JPA usa y TopLink entiende **[64]**. A continuación mostramos parte del código que genera las mencionadas consultas:

Búsqueda de persona moral por razón social:

```
431   List<Personamoral> personaMoralList =
      session.entityManager.createQuery("SELECT p FROM Personamoral p WHERE
      p.razonSocial LIKE :razonSocial")
432                       .setParameter("razonSocial","%"+nombre.getText()+"%")
433                       .getResultList();
```

Fragmento 8.30 – Clase **Busqueda**

A continuación se encuentra un ejemplo de lo que regresa la consulta anterior con el parámetro "Alka".

Parámetro	Resultado
Alka	**Razón social - Diseño y Construcciones Alka**

Tabla 8.31 – Consulta por razón social

Búsqueda de cliente por su identificador, **id Cliente**, independientemente de si es persona moral o física:

```
456  cliente = (Cliente)
     session.entityManager.createNamedQuery("Cliente.findByIdCliente")
457  .setParameter("idCliente",Integer.parseInt(numeroClienteSistema.getText()))
458  .getSingleResult();
```

Fragmento 8.32 – Clase **Busqueda**

La tabla 8.32 nos muestra un ejemplo de la consulta anterior.

[55] JPQL *Java Persistence Query Language*: Lenguaje de consultas orientado a objetos, parte del JPA **[64].**

Parámetro	Resultado
15	**15 - Consolidadora Tecnológica de Proyectos SA de CV**

Tabla 8.33 – Consulta ***id Cliente***

Consulta que regresa una lista con todos los clientes sin filtro alguno:

```
536   List<Cliente> clienteList = session.entityManager.createQuery("SELECT c
      FROM Cliente c").getResultList();
```

Fragmento 8.34 – Clase **Busqueda**

En la siguiente tabla podemos observar cómo funciona la consulta anterior.

Parámetro	Resultado
- No recibe parámetros -	- Lista completa de clientes -

Tabla 8.35 – Consulta de todos los clientes

Independientemente del método de búsqueda usado, la lista de clientes se liga a otra pantalla en donde se muestran los resultados. Misma que se muestra en la figura 8.34. Respecto al JPQL, podemos observar que es muy similar al SQL pero con algunas características extras. Por ejemplo, en la búsqueda por identificador de **Cliente**, no se crea la consulta en ese momento; en realidad lo que se hace es acceder a una previamente elaborada por medio de su identificador y luego ajustamos los parámetros de la misma. El fragmento 8.32 ilustra este proceso. En la línea 456 de dicho fragmento se busca la consulta llamada **Cliente.findByIdCliente** y en la 457 le pasamos el parámetro deseado. La consulta en cuestión es la siguiente: **SELECT c FROM Cliente c WHERE c.idCliente = :idCliente**, la cual regresa el cliente cuyo identificador sea igual al que le estamos pasando como parámetro. La tabla 8.33 contiene un ejemplo de un parámetro pasado a esa consulta junto con su resultado.

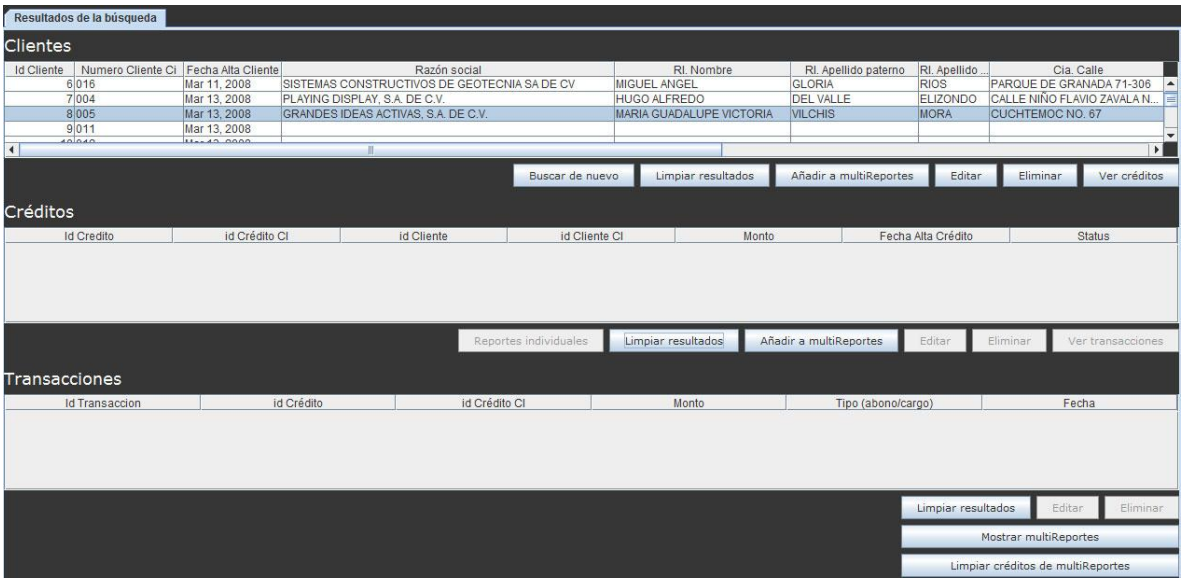


Figura 8.36 – Resultados de la búsqueda

8.2.4 Captura y mantenimiento de créditos

El proceso de apertura de créditos no es tan complejo como el de apertura de clientes. Sin embargo, es de suma importancia pues representa la base de la actividad principal de Crédito Integral®. Dar de alta un crédito nuevo involucra llenar una sencilla forma que recaba los datos del mismo. Dentro de estos datos se encuentran; la tasa de interés, el monto del crédito y las fechas de amortización. Estas últimas pueden ser calculadas automáticamente por el CIMS de acuerdo al plazo del crédito, y también pueden ser capturadas por el usuario. La parte del código que realiza dichas funciones se encuentra a continuación:

```
527         if (autoFechas) {
528             int plazo = creditoBind.getPlazoCredito();
529             Calendar calendar = new GregorianCalendar();
530             calendar.setTime(creditoBind.getFechaAltaCredito());
531             for (int i=0; i<plazo; i++) {
532                 calendar.add(calendar.MONTH,1);
533                 fecha = new Fechaamortizacion(null,calendar.getTime());
534                 fecha.setIdCreditoF(creditoBind);
535                 Session.entityManager.persist(fecha);
536             }
```

```
537         }
538         fechasAdded=true;
```

Fragmento 8.37 – Clase **`CreditoView`**

En el fragmento anterior, la línea 527 verifica si las fechas de amortización del crédito deben ser añadidas automáticamente. Las líneas 528-530 fijan la fecha y el plazo que serán usados en el ciclo de las líneas 531-536 en donde se van agregando las fechas de amortización con base en el plazo del crédito. Al final, la línea 538 fija una variable global con el valor **`true`** que indica que las fechas de amortización ya han sido agregadas.

Observando el código podemos notar que es bastante sencillo pues simplemente usa el plazo del crédito para agregar una fecha de amortización cada cierto periodo de tiempo. No obstante, el cálculo y/o alta de fechas de amortización de forma manual, son una parte indispensable del proceso de alta o modificación de un crédito debido a que el cálculo de intereses ordinarios y moratorios se realizarán a partir de estos elementos. El plazo de la amortización, el plazo del crédito, la tasa moratoria y ordinaria son suficientes datos para que el CIMS nos dé la información acerca de quién debe pagar cuánto y cuándo.

8.2.5 Captura y mantenimiento de transacciones y pagos

Una vez dado de alta un crédito, los usuarios del CIMS podrán dar de alta transacciones fechadas que pueden aumentar o disminuir el monto adeudado por el cliente. Esta operación consiste en llenar una forma que nos pide los detalles de la transacción en cuestión, mismos que se encuentran en el diagrama 8.19. El código que realiza dicha función no resulta muy interesante pues es algo sencillo, salvo por la parte que realiza la liga entre objetos POJO, cuya función se describe en la explicación del fragmento 8.39, y los componentes gráficos. Es una característica relativamente nueva introducida al mercado a mediados del 2006 y que lleva por nombre "Beans Binding (JSR[56] 295)" parte también de los swinglabs. Es un paquete de funciones muy

[56] JSR *Java Specification Request*: Es el nombre que se le da a los documentos formales generados por la Comunidad de Procesos de Java (JCP por sus siglas en inglés) que describen especificaciones y tecnologías que son candidatos a ser añadidas a la plataforma de Java **[66]**.

útil que puede ser usado sin ningún problema con tal sólo prestar atención a los errores[57]. La pantalla se expone en la siguiente figura.

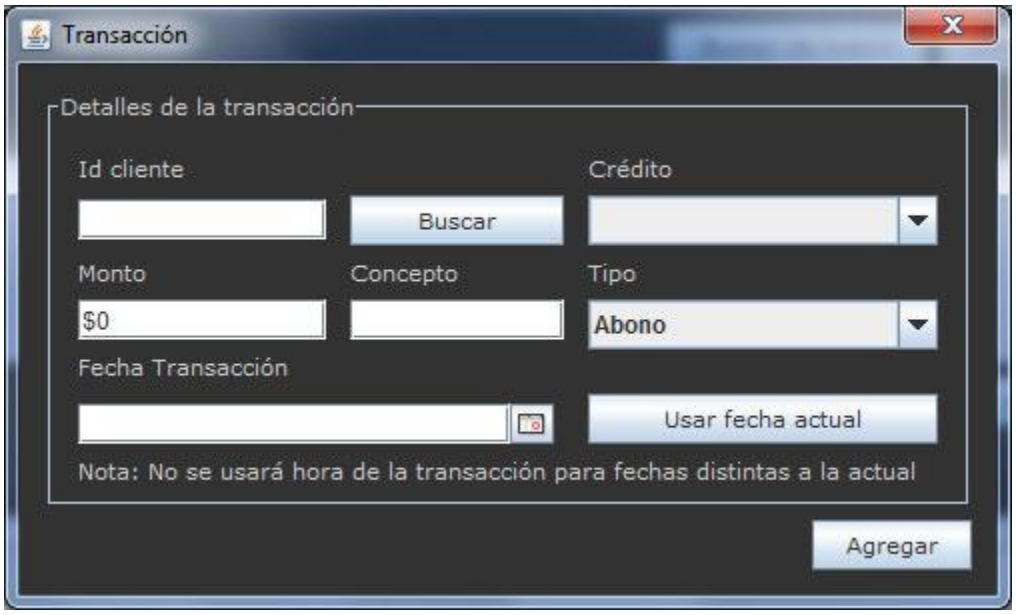


Figura 8.38 – Pantalla de transacciones

En la pantalla anterior podemos ver los distintos parámetros de una transacción que pueden ser capturas por el usuario. La parte del código de esa pantalla que usa el JSR 295 se encuentra a continuación:

```
80    bindingGroup = new org.jdesktop.beansbinding.BindingGroup();
      -- líneas omitidas por falta de importancia en este contexto --
134   org.jdesktop.beansbinding.Binding binding =
      org.jdesktop.beansbinding.Bindings.createAutoBinding(org.jdesktop.beansbin
      ding.AutoBinding.UpdateStrategy.READ_WRITE, transaccion1,
      org.jdesktop.beansbinding.ELProperty.create("${concepto}"),
      conceptoTextField, org.jdesktop.beansbinding.BeanProperty.create("text"));
135   bindingGroup.addBinding(binding);
      -- líneas omitidas por falta de importancia en este contexto --
192   binding =
      org.jdesktop.beansbinding.Bindings.createAutoBinding(org.jdesktop.beansbin
      ding.AutoBinding.UpdateStrategy.READ_WRITE, transaccion1,
      org.jdesktop.beansbinding.ELProperty.create("${fecha}"),
```

[57] Error: Se refiere a una falla o defecto en un programa de computadora, sistema o máquina **[41]**.

```
    fechaTransaccionDate,
    org.jdesktop.beansbinding.BeanProperty.create("date"));
193 bindingGroup.addBinding(binding);
    -- líneas omitidas por falta de importancia en este contexto --
312 bindingGroup.bind();
```

Fragmento 8.39 – Clase **TransaccionView**

Del fragmento anterior, en la línea 80 se crea un objeto de tipo **BindingGroup**, denominado **bindingGroup**, al cual le vamos a agregar los objetos que deben de estar ligados. En las líneas 134 y 192 se asignan al objeto **binding** de tipo **Binding** parejas compuestas por un componente gráfico a ligar con una propiedad de un POJO. Por ejemplo en la línea 134 se liga el componente gráfico **conceptoTextField** con la propiedad **concepto** del POJO **transacicon1**. En las líneas 135 y 193 simplemente se agrega el objeto **Binding** al **BindingGroup** de la línea 80. Finalmente, en la línea 312 se ejecuta el método **bind()** que finaliza el enlace. A continuación, dentro del fragmento 8.39-1, se muestra dicho método.

```
142 public void bind() {
143   List<Binding> toBind = new ArrayList<Binding>(unbound);
144   for (Binding binding : toBind) {
145           binding.bind();
146   }
147 }
```

Fragmento 8.39-1 – Método **BindingGroup.bind()**

La línea más importante del fragmento anterior es la 145 que básicamente aplica otro método **bind()**, pero de la clase **Binding**. Veamos este método en el fragmento 8.39-2.

```
442 public final void bind() {
443       throwIfManaged();
444       bindUnmanaged();
445 }
```

Fragmento 8.39-2 – Método **Binding.bind()**

Del fragmento 8.39-2 podemos recalcar lo siguiente, la línea 443 lanza una excepción si los elementos a ligar (*bind*) fueron unidos previamente, de lo contrario la línea 444 los une. El detalle del método **bindUnmanaged()**, de la línea anterior, se encuentra a continuación en el fragmento 8.39-3.

```
466   protected final void bindUnmanaged() {
467         throwIfBound();
468
469         bindImpl();
470
471         psl = new PSL();
472         sourceProperty.addPropertyStateListener(sourceObject, psl);
473         targetProperty.addPropertyStateListener(targetObject, psl);
474
475         isBound = true;
476
477         if (listeners != null) {
478         for (BindingListener listener : listeners) {
479               listener.bindingBecameBound(this);
480         }
481    }
482
483         firePropertyChange("bound", false, true);
484    }
```

Fragmento 8.39-3 – Método **Binding.bindUnmanaged()**

Del fragmento anterior sobresalen las líneas 472 y 473 las cuales ligan la propiedad fuente con la propiedad destino. Los cambios de la propiedad fuente se reflejarán en la propiedad destino. Además desde la líneas 477 hasta la 484 se notifica a las partes involucradas que la propiedad en cuestión ha sido ligada.

8.2.6 Generador de tablas de amortización

Las tablas de amortización son el reporte principal de los créditos otorgados. Representa la base sobre la cual se construyen otros reportes, mucho más complicados como los del **Multireportes**. Cada tabla de amortización recibe como parámetro un

crédito junto con la fecha inicial y final del reporte. Las columnas a mostrar son fijas. El proceso de generación involucra el cálculo de los indicadores del crédito (monto adeudado, intereses ordinarios, intereses moratorios, entre otros) cada día. El código dentro del ciclo principal se encuentra a continuación:

```
190    for (int i = 0; i < this.loopGlobal; ++i) {
191          this.iGlobal = i;
192          this.fechaCorriendo.add(5, 1);
193
194          if (i != 0) {
195                resetCifras();
196                aplicaMontosVencidos();
197                aplicaPagos();
198                hazMontos();
199                hazAbonos();
200                calculaMontosVencidos();
201          }
202          else {
203                hazMontos();
204                hazAbonos();
205          }
206                hazExtras();
207                colocaInfo();
208    }
```

Fragmento 8.40 – Clase **TransaccionView**

Las líneas 190-208 aplican los montos vencidos, aplican los pagos, calculan los nuevos montos, realizan abonos y calculan los montos vencidos del siguiente día. Dichas líneas se ejecutan una vez por cada día de la tabla de amortización. El detalle de los métodos **hazMontos()** y **hazAbonos()** se encuentran en los fragmentos 8.40-1 y 8.40-2 respectivamente.

```
269  private void hazMontos() {
270    adeudoMonto = calculaAdeudoMonto();
```

```
271    intereses = monto * (credito.getTasaOrdinariaMensual() / (100*30));
272    saldoIntereses = saldoIntereses + intereses;
273
274    interesesMoratorios = calculaInteresesMoratorios();
275   saldoInteresesMoratorios = saldoInteresesMoratorios+interesesMoratorios;
276
277    interesesMensualesFinMesAid += intereses;
278    interesesMensualesCorteAid += intereses;
279    interesesMoratoriosCorteAid += interesesMoratorios;
280  }
```

Fragmento 8.40-1 – Método **TransaccionView.hazMontos()**

```
283 private void hazAbonos() {
284 for (Transaccioncredito trcr:credito.getTransaccioncreditoCollection()) {
285  Transaccion tr = trcr.getTransaccion();
286  Double pago = 0.0;
287  if (!tr.getTipo()) {
288   pago = tr.getMonto();
289  }
290  Calendar fechaTransaccion = new GregorianCalendar();
291  fechaTransaccion.setTime(tr.getFecha());
292
293 if (Utilities.compareCalendarsSimple(fechaTransaccion,fechaCorriendo)) {
294 pago = abona(pago,saldoInteresesMoratorios,ABONOSALDOINTERESESMORATORIOS);
295 pago = abona(pago,interesesVencidos,ABONOINTERESESVENCIDOS);
296 pago = abona(pago,montoVencido,ABONOMONTOVENCIDO);
297 boolean paid = false;
298 for (Fechaamortizacion fa:credito.getFechaamortizacionCollection()) {
299  Calendar fechaAmortizacion = new GregorianCalendar();
300  fechaAmortizacion.setTime(fa.getFechaAmortizacion());
301  if(Utilities.compareCalendarsSimple(fechaAmortizacion,fechaCorriendo)){
302  pago = abona(pago,saldoIntereses,ABONOSALDOINTERESES);
303  pago = abona(pago,monto,ABONOMONTO);
304  paid = true;
305
306 }
```

```
307 }
308 if (!paid) {
309  pago = abona(pago,saldoIntereses,ABONOANTICIPADOINTERESES);
310  pago = abona(pago,monto,ABONOANTICIPADOMONTO);
311  }
312 }
313 }
314 abonoMontoAcumulado += abonoAnticipadoMonto;
315 abonoMontoAcumulado += abonoMonto;
316 abonoMontoAcumulado += abonoMontoVencido;
317 }
```

Fragmento 8.40-2 – Método **`TransaccionView.hazAbonos()`**

Desde el principio del fragmento 8.40-2 hasta la línea 292 se obtienen las transacciones de tipo pago realizadas para el crédito en cuestión. También se convierte la fecha de las transacciones de tipo **`Date`** a **`Calendar`** para su fácil manejo. Desde las líneas 293 hasta la 313 se divide el pago en los distintos conceptos del crédito. Primero, se pagan todos intereses moratorios (línea 294), después los intereses vencidos (línea 295) y al final el monto vencido (línea 296). Posteriormente, en las fechas de amortización correspondientes, se abonan los intereses ordinarios (línea 302) y después la amortización correspondiente al periodo (línea 303). Finalmente se abonan intereses anticipados (línea 309) y abonos a capital anticipados (310). Las últimas 4 líneas simplemente suman todo el capital (monto) pagado.

Cabe destacar que los pagos se realizan siempre y cuando exista un adeudo al concepto en cuestión y el monto de los pagos sean suficientes para cubrirlos. Por ejemplo, si es que el monto de los intereses moratorios (1er concepto que se paga) es igual al monto del pago, entonces ya no se abona al monto vencido aunque éste exista.

Al final de todo el proceso se coloca la información dentro de la tabla de amortización y se despliega al usuario. Recordemos que la tabla de amortización nos muestran una monografía del crédito en cuestión en un periodo de tiempo determinado. La siguiente figura nos muestra como se ve una tabla de amortización en el CIMS.

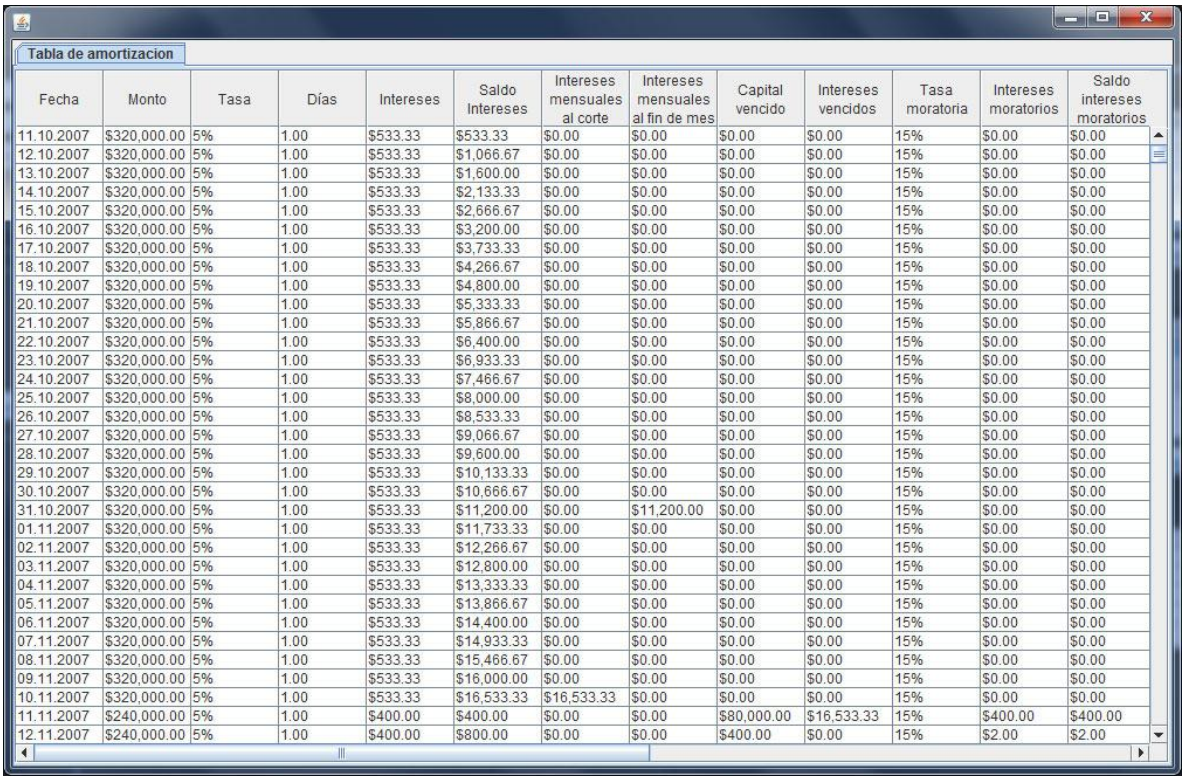


Tabla de amortizacion

Fecha	Monto	Tasa	Días	Intereses	Saldo Intereses	Intereses mensuales al corte	Intereses mensuales al fin de mes	Capital vencido	Intereses vencidos	Tasa moratoria	Intereses moratorios	Saldo intereses moratorios
11.10.2007	$320,000.00	5%	1.00	$533.33	$533.33	$0.00	$0.00	$0.00	$0.00	15%	$0.00	$0.00
12.10.2007	$320,000.00	5%	1.00	$533.33	$1,066.67	$0.00	$0.00	$0.00	$0.00	15%	$0.00	$0.00
13.10.2007	$320,000.00	5%	1.00	$533.33	$1,600.00	$0.00	$0.00	$0.00	$0.00	15%	$0.00	$0.00
14.10.2007	$320,000.00	5%	1.00	$533.33	$2,133.33	$0.00	$0.00	$0.00	$0.00	15%	$0.00	$0.00
15.10.2007	$320,000.00	5%	1.00	$533.33	$2,666.67	$0.00	$0.00	$0.00	$0.00	15%	$0.00	$0.00
16.10.2007	$320,000.00	5%	1.00	$533.33	$3,200.00	$0.00	$0.00	$0.00	$0.00	15%	$0.00	$0.00
17.10.2007	$320,000.00	5%	1.00	$533.33	$3,733.33	$0.00	$0.00	$0.00	$0.00	15%	$0.00	$0.00
18.10.2007	$320,000.00	5%	1.00	$533.33	$4,266.67	$0.00	$0.00	$0.00	$0.00	15%	$0.00	$0.00
19.10.2007	$320,000.00	5%	1.00	$533.33	$4,800.00	$0.00	$0.00	$0.00	$0.00	15%	$0.00	$0.00
20.10.2007	$320,000.00	5%	1.00	$533.33	$5,333.33	$0.00	$0.00	$0.00	$0.00	15%	$0.00	$0.00
21.10.2007	$320,000.00	5%	1.00	$533.33	$5,866.67	$0.00	$0.00	$0.00	$0.00	15%	$0.00	$0.00
22.10.2007	$320,000.00	5%	1.00	$533.33	$6,400.00	$0.00	$0.00	$0.00	$0.00	15%	$0.00	$0.00
23.10.2007	$320,000.00	5%	1.00	$533.33	$6,933.33	$0.00	$0.00	$0.00	$0.00	15%	$0.00	$0.00
24.10.2007	$320,000.00	5%	1.00	$533.33	$7,466.67	$0.00	$0.00	$0.00	$0.00	15%	$0.00	$0.00
25.10.2007	$320,000.00	5%	1.00	$533.33	$8,000.00	$0.00	$0.00	$0.00	$0.00	15%	$0.00	$0.00
26.10.2007	$320,000.00	5%	1.00	$533.33	$8,533.33	$0.00	$0.00	$0.00	$0.00	15%	$0.00	$0.00
27.10.2007	$320,000.00	5%	1.00	$533.33	$9,066.67	$0.00	$0.00	$0.00	$0.00	15%	$0.00	$0.00
28.10.2007	$320,000.00	5%	1.00	$533.33	$9,600.00	$0.00	$0.00	$0.00	$0.00	15%	$0.00	$0.00
29.10.2007	$320,000.00	5%	1.00	$533.33	$10,133.33	$0.00	$0.00	$0.00	$0.00	15%	$0.00	$0.00
30.10.2007	$320,000.00	5%	1.00	$533.33	$10,666.67	$0.00	$0.00	$0.00	$0.00	15%	$0.00	$0.00
31.10.2007	$320,000.00	5%	1.00	$533.33	$11,200.00	$0.00	$11,200.00	$0.00	$0.00	15%	$0.00	$0.00
01.11.2007	$320,000.00	5%	1.00	$533.33	$11,733.33	$0.00	$0.00	$0.00	$0.00	15%	$0.00	$0.00
02.11.2007	$320,000.00	5%	1.00	$533.33	$12,266.67	$0.00	$0.00	$0.00	$0.00	15%	$0.00	$0.00
03.11.2007	$320,000.00	5%	1.00	$533.33	$12,800.00	$0.00	$0.00	$0.00	$0.00	15%	$0.00	$0.00
04.11.2007	$320,000.00	5%	1.00	$533.33	$13,333.33	$0.00	$0.00	$0.00	$0.00	15%	$0.00	$0.00
05.11.2007	$320,000.00	5%	1.00	$533.33	$13,866.67	$0.00	$0.00	$0.00	$0.00	15%	$0.00	$0.00
06.11.2007	$320,000.00	5%	1.00	$533.33	$14,400.00	$0.00	$0.00	$0.00	$0.00	15%	$0.00	$0.00
07.11.2007	$320,000.00	5%	1.00	$533.33	$14,933.33	$0.00	$0.00	$0.00	$0.00	15%	$0.00	$0.00
08.11.2007	$320,000.00	5%	1.00	$533.33	$15,466.67	$0.00	$0.00	$0.00	$0.00	15%	$0.00	$0.00
09.11.2007	$320,000.00	5%	1.00	$533.33	$16,000.00	$0.00	$0.00	$0.00	$0.00	15%	$0.00	$0.00
10.11.2007	$320,000.00	5%	1.00	$533.33	$16,533.33	$16,533.33	$0.00	$0.00	$0.00	15%	$0.00	$0.00
11.11.2007	$240,000.00	5%	1.00	$400.00	$400.00	$0.00	$0.00	$80,000.00	$16,533.33	15%	$400.00	$400.00
12.11.2007	$240,000.00	5%	1.00	$400.00	$800.00	$0.00	$0.00	$400.00	$0.00	15%	$2.00	$2.00

Figura 8.41 – Tabla de amortización

En la figura anterior podemos observar los detalles de un crédito en particular. Los pormenores incluyen fecha, monto, tasa, días por renglón de la tabla, intereses por periodo, saldo de intereses, intereses mensuales al corte, entre otros.

8.2.7 Multireportes

El **`Multireportes`** es el motor generador de reportes del CIMS. Es totalmente parametrizable y el usuario puede indicar los créditos a incluir en el reporte así como las características de los mismos que serán tomadas en cuenta. Además, cuenta con botones que agrupan características según su función a fin de poder crear reportes comúnes. También encontramos filtros que nos ayudan a mostrar en el reporte final sólo los créditos en el estado requerido como vigente o liquidado. También, resulta importante mostrar el código que implementa cada función dentro del motor. Parte de dicho código se encuentra a continuación:

```
1    class ComisionIva extends Actions {
2           static Map<Credito, Double> result;
3
4           public Map<Credito, Double> operation(List<Map> modelList){
5                 Map resultLocal = new HashMap();
6                 if (ComisionCobrada.result == null) {
7                       ComisionCobrada cc = new ComisionCobrada();
8                       cc.operation(modelList);
9                 }
10                Set<Credito> creditoKeys = ComisionCobrada.result.keySet();
11                for (Credito cr : creditoKeys) {
12   resultLocal.put(cr,Double.valueOf(((Double)ComisionCobrada.result.get(cr))
13   .doubleValue() * CENTESIMA * cr.getMonto() * IVA));
14                }
15                result = resultLocal;
16                return result;
17          }
18
19          public Format getFormat() {
20                return Utilities.money;
21          }
22   }
```

Fragmento 8.42 – Clase **`ComisionIva`**

Del fragmento anterior, la línea 1 declara el nombre de la clase, la 2 el mapa en donde se guardarán los resultados del cálculo. Las líneas 4-17 calculan el IVA de la comisión, aunque realmente las líneas 12 y 13 son las que realizan las operaciones aritméticas. Finalmente las líneas 19-21 regresan el formato en el que debe ser mostrado el resultado del cálculo.

Cada característica es una clase aparte que realiza cierta operación específica sobre las tablas de amortización. Por ejemplo la clase **`ComisionIva`** calcula el IVA de la comisión del crédito que le es pasado como parámetro. De esta manera podemos implementar nuevas funciones, por ejemplo calcular el valor presente del crédito, con tan sólo crear nuevas clases y desarrollar la operación en cuestión.

8.3 Experiencia del usuario

Una de las características con las que CIMS debe indispensablemente contar, dado que fue explícitamente estipulado al comenzar el proyecto, es facilidad de uso. El personal en contacto constante con el software debía poder aprender a usarlo intuitivamente y en poco tiempo. Acoplar esta herramienta a los procesos habituales de la empresa tenía que ser un proceso transparente y no un problema adicional que resolver.

Después de analizar la funcionalidad provista por CIMS y de múltiples episodios de lo que parecía *scope creep*, se concluyó que el proceso más propenso a extender indefinidamente el proyecto fue la generación de reportes. Los numerosos reportes "diferentes" que debía ser capaz de generar el software demoraron por varias semanas la culminación del módulo involucrado. Paradójicamente, cumplir con las peticiones del usuario de agregar más reportes lejos de generar confianza en el progreso del proyecto, causaba sensación de el proyecto estaba más distante a su culminación. Además, resultaba complicado clasificar los reportes e integrarlos en la interfaz gráfica. Lo que generaba confusión y deterioraba la experiencia del usuario.

La solución a dicho problema fue el **`Multireportes`**. La idea de este componente surgió de la necesidad de evitar la constante necesidad de "nuevos reportes" del sistema que resurgía cada junta de seguimiento con los directivos de Crédito Integral. La respuesta a situaciones de este tipo es por lo general la aplicación de límites estrictos respecto a las funciones que sí se añadirán y a las que no. Sin embargo, muchos de los "nuevos reportes" resultaban muy útiles. Lo que propició que dichos reportes fueran añadidos paulatinamente a las funciones del CIMS, pero por medio del **`Multireportes`** que permite la creación de prácticamente cualquier reporte basado en parámetros y funciones previamente definidas. Este componente no sólo resolvió el problema de los reportes infinitos sino que también mejoró la experiencia del usuario al simplificar el acceso a ellos.

Capítulo 9

Pruebas y trabajo a futuro

Las pruebas realizadas a los paquetes de software representan una manera de comunicarle a toda la gente involucrada la calidad de los productos finales. También son una forma de validar que las aplicaciones fueron desarrolladas conforme a los requerimientos definidos previamente en los planes de proyecto **[108]**.

Por otro lado, el trabajo a futuro engloba todas las funciones que podrían ser implementadas dentro del CIMS (arrendamiento financiero, factoraje, diversificación de reportes, entre otras), pero que debido al alcance de este trabajo, no podrán ser desarrolladas para esta versión en particular. Dichas funciones se encuentran explicadas detalladamente en la sección 9.2 de este capítulo.

9.1 Pruebas

"*El desarrollo de sistemas de software implica una serie de actividades de producción en las que las posibilidades de que aparezca el fallo humano son enormes. Los errores pueden empezar a darse desde el primer momento del proceso, en el que los objetivos pueden estar especificados de forma errónea o imperfecta, así como en posteriores pasos de diseño y desarrollo. Debido a la imposibilidad humana de trabajar y comunicarse en forma perfecta, el desarrollo de software debe de ir acompañado de una actividad que garantice la calidad*" **[109]**.

El proceso de pruebas es muy importante ya que en muchas organizaciones que fabrican software dedican hasta el 40% del tiempo total de desarrollo de un producto solamente en cuestiones relativas a las pruebas **[46]**.

Existen numerosos tipos de pruebas en el ámbito del desarrollo de software **[110]**; sin embargo al CIMS se le aplicaron pruebas unitarias, pruebas de integración y pruebas de aceptación del usuario. A continuación se exponen cada una de esas pruebas.

9.1.1 Pruebas unitarias

Las pruebas unitarias son un método para verificar el software en donde el desarrollador prueba unidades individuales de código. Una unidad de código se define como la parte más pequeña de una aplicación que se puede probar. Este tipo de pruebas pueden estar diseñadas de forma manual o automática con la ayuda de un software que las genere.

Dentro de las ventajas que ofrecen las pruebas unitarias se encuentran las siguientes:

- Permite realizar cambios a código existente o añadir nueva funcionalidad y verifica que dichos cambios no hayan generado nuevos errores.

- Facilita la integración de componentes unitarios nuevos sin probar y elementos existentes probados al permitir realizar pruebas de abajo hacia arriba[58].

- Simplifica la documentación al proporcionar más información sobre la acción que realizan las unidades de código que están siendo probadas.

Actualmente existen en el mercado varias herramientas que permiten construir y ejecutar pruebas unitarias de forma automática. Para el caso del CIMS que está escrito en lenguaje de programación Java, JUnit[59], la cual es una herramienta fácil de usar, libre y gratuita **[111]**.

A continuación se encuentra la figura 9.1 donde se enlistan las pruebas unitarias diseñadas para el CIMS junto con el resultado de su ejecución.

#	Operación a realizar	Entrada	Resultado esperado	Estatus
1	Generar correctamente las fechas de amortización de	Crédito con: plazo = 10 (meses)	10 fechas de amortización	Aprobada

[58] De abajo hacia arriba: estrategia para procesar información en donde se parte de unidades pequeñas hacia unidades más grandes. Las partes pequeñas se convierten en integrantes de las más grandes **[41]**.
[59] JUnit: *Framework* libre y gratuito escrito en el lenguaje de programación Java para realizar pruebas unitarias de código escrito en el mismo lenguaje **[111]**.

	un crédito otorgado a un plazo determinado	periodicidad = 30 (días) ningún día inhábil		
2	Cálculo de interés ordinario en la tabla de amortización	Crédito con: monto = 150,000 (pesos) plazo = 12 (meses) tasa ordinaria = 5 % periodicidad = 30 (días)	Interés diario = 250 (pesos)	Aprobada
3	Cálculo de interés moratorio en la tabla de amortización	Crédito con: monto = 33,000 (pesos) plazo = 6 meses tasa moratoria = 15 % periodicidad = 60 (días) fecha de alta de crédito = 1 febrero 2010 fecha actual = 3 abril de 2010 transacciones = []	Interés moratorio diario = 55 (pesos)	Aprobada
4	Cálculo de interés total en la tabla de amortización de un crédito blando sin penalización por amortizaciones vencidas	Crédito con: monto = 1,100,000 (pesos) plazo = 1 (mes) tasa ordinaria = 3 % tasa moratoria = 0 % periodicidad = 30 (días) fecha alta de crédito = 21 marzo de 2008 fecha actual = 21 de mayo de 2008 ninguna transacción registrada	Interés total a la fecha tomando en cuenta fecha actual = 33,000 (pesos)	Reprobada inicialmente aunque aprobada actualmente*
5	Eliminación de un crédito por parte de un usuario limitado	Usuario limitado	Operación denegada	Aprobada
6	Cálculo del CAT de un crédito particular dentro del Multireportes	Crédito con: monto =10,000 (pesos) plazo = 12 (meses) tasa ordinaria = 5 % tasa moratoria = 15 %	CAT = 71.92 %	Aprobada

		periodicidad = 30 (días) comisión por apertura = 2 %		
*La prueba 4 fue inicialmente reprobada porque la tabla de amortización dejaba de calcular intereses una vez que el crédito vencía por completo. Esto es debido a que el monto total del crédito era tomado como entrada para el cálculo de intereses moratorios al vencerse la amortización única. Sin embargo la tasa de interés moratorio era cero y no se calculaban intereses ni ordinarios ni moratorios ocasionando que el crédito fuera "gratuito". Se modificó el motor de gestión de créditos a fin de que los créditos blandos sin tasa moratoria tuvieran automáticamente un interés igual al de la tasa de interés ordinario.				

Figura 9.1 - Pruebas unitarias ejecutadas por el CIMS

La mecánica de cada una de las seis pruebas expuestas en la figura 9.1 fue la siguiente:

1. Un crédito con un plazo de 10 meses que amortizaba cada 30 días fue dado de alta. El CIMS debía calcular las 10 fechas de amortización automáticamente basado en los parámetros anteriores. El resultado fue exitoso.
2. Un crédito por $150,000.00 que vencía mensualmente fue dado de alta con un plazo de 12 meses y una tasa ordinaria de intereses de 5%. El sistema debía de calcular el interés diario de $250.00 (durante el primer mes). La prueba fue exitosa.
3. Un crédito por $33,000.00 que amortizaba bimestralmente fue dado de alta con un plazo de 6 meses y una tasa de intereses moratorios del 15%. CIMS calcularía entonces el monto de los intereses moratorios diarios durante el primer periodo de morosidad ($55.00). El resultado fue exitoso.
4. Otro crédito fue dado de alta por $1,100,000.00 con un plazo de un mes, vencimiento mensual, tasa de interés ordinario del 3% y moratorio del 0%. Habiendo transcurrido dos meses de la apertura del crédito sin ningún pago, el CIMS debía de calcular $33,000.00 de intereses totales. La prueba fue inicialmente reprobada, aunque después aprobada. Los detalles de este caso están al final de la figura 9.1.
5. Se intentó eliminar un crédito por medio con permisos de un usuario limitado, la operación fue denegada.
6. El CIMS debía de calcular el CAT a un crédito por $10,000.00 pesos con plazo de 12 meses, amortización mensual, comisión por apertura del 2%, tasa de

intereses ordinarios del 5% y moratorios del 15%. El resultado fue el esperado: 71.92%.

9.1.2 Pruebas de integración

Este tipo de pruebas contempla la combinación de módulos individuales para ser evaluados como un todo. Las pruebas de integración suponen que las pruebas unitarias ya han sido realizadas, pues la estrategia usa el método de abajo hacia arriba para atacar el problema.

Las pruebas de integración verifican el desempeño de los componentes al mismo tiempo que verifican la confiabilidad de los mismos. Los casos de prueba son diseñados y construidos de tal forma que se permita corroborar la correcta interacción de los componentes.

El CIMS fue evaluado a través de varias pruebas de integración que contemplaban métodos previamente verificados a través de pruebas unitarias como las de la figura 9.1. Dada la naturaleza de módulos como el **`Multireportes`** (ver figura 6.4 del capítulo sexto) y los **`estados de cuenta`** (ver figura 9.2) que generan información a partir de datos de la **`tabla de amortización`**, la mayor parte de los casos de prueba de este tipo fueron aplicados a dichos módulos.

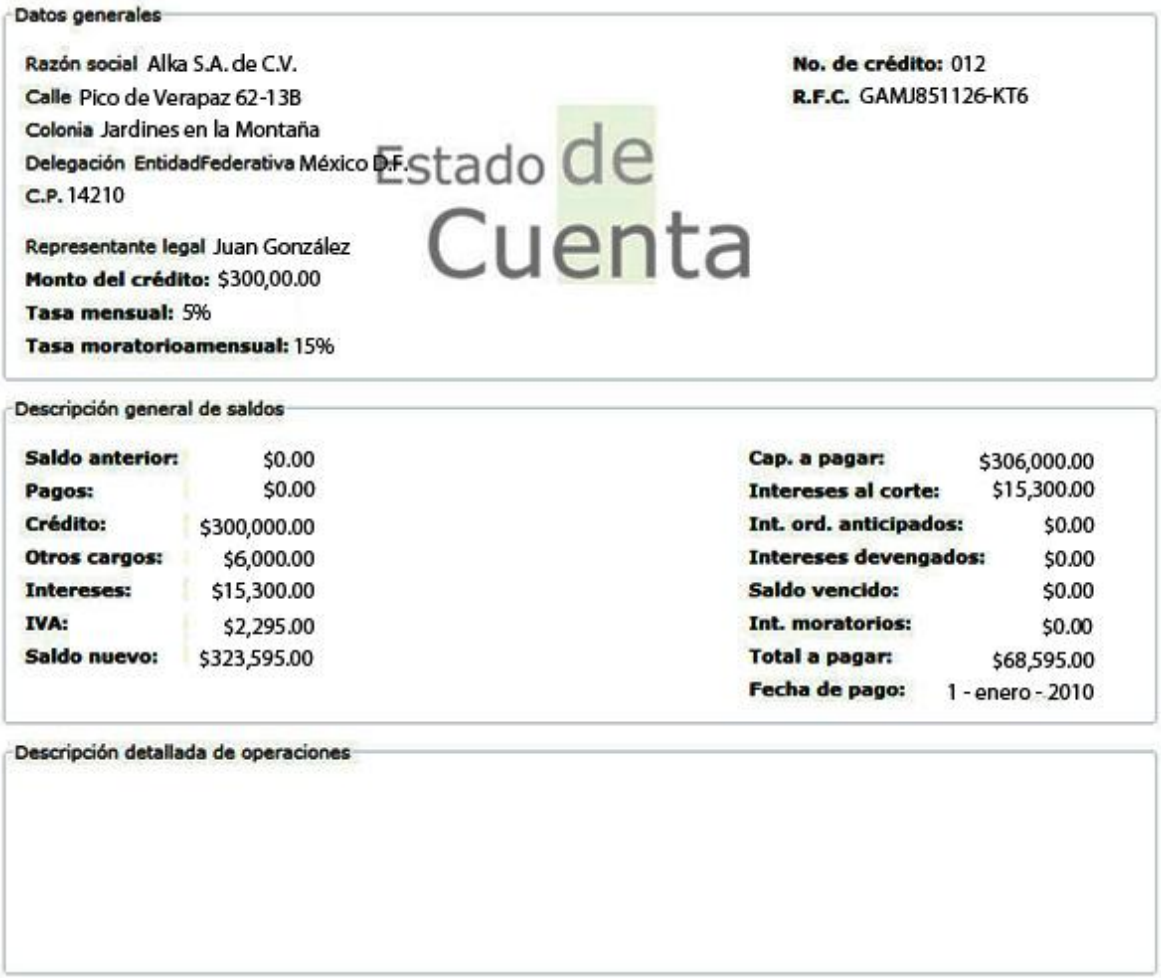


Figura 9.2 - Plantilla del estado de cuenta

El módulo de **`estados de cuenta`**, mostrado en la figura anterior, está divido en tres secciones. La denominada "Datos generales" (primer rectángulo de arriba hacia abajo) muestra los datos generales del cliente en cuestión junto con las características principales del crédito que le fue otorgado. La siguiente, "Descripción general de saldos", muestra los detalles del comportamiento del crédito durante el periodo que abarca el estado de cuenta. Finalmente se encuentra la "Descripción detallada de operaciones" (último rectángulo de arriba hacia abajo) en donde se enlistan las transacciones aplicadas. Todos los datos mostrados en el estado de cuenta, salvo los del primer rectángulo, dependen directamente del resultado de los cálculos del módulo **`tabla de amortización`**. Esto refuerza la idea de basar la mayoría de la pruebas en dicho módulo.

9.2 Trabajo a futuro

Respecto al trabajo a futuro existen puntos de suma importancia sobre los cuales podríamos trabajar en un corto y mediano plazo. A fin de poder examinarlos fácilmente, los hemos dividido con base en el objetivo que cada uno de éstos persiguen. A continuación se enlistan dichos objetivos.

9.2.1 Crecimiento de Crédito Integral dentro de la figura de SOFOM

Varios de los puntos a desarrollar están relacionados con la línea de negocio que Crédito Integral® aún no ha explotado. Recordemos que una SOFOM está facultada por la CNBV para poder realizar las siguientes actividades:

1. Otorgamiento de créditos a plazo fijo (por ejemplo un crédito automotriz o hipotecario) o por medio de una línea de crédito (p. ej. una tarjeta de crédito).
2. Operaciones de factoraje con garantía (el cliente está obligado a liquidar la cuenta por cobrar si la SOFOM no puede hacerlo) y sin ella (la SOFOM asume todo el riesgo de que la cuenta no pueda ser cobrada).
3. Arrendamiento financiero a plazo variable.

Actualmente solo realiza el primero de los puntos antes mencionados. Se espera que Crédito Integral® crezca y diversifique sus servicios. Una forma de hacerlo, aprovechando por completo el modelo de negocio de SOFOM, sería ofreciendo los servicios de factoraje y arrendamiento (puntos 2 y 3 del listado anterior).

Los dos puntos restantes en esencia son otorgamiento de créditos. Lo único que cambia es el fin del "crédito" y la forma en la que se está otorgando. Esto implica que acciones que ya se realizan en el otorgamiento de créditos, como la generación de tablas de amortización y estados de cuenta, también se encuentran involucradas en el factoraje y el arrendamiento financiero, lo que simplifica en gran medida la implementación técnica de estas líneas de negocio. Para cubrir las dos líneas de negocio restantes en un futuro, será necesario la elaboración de nuevas pantallas para capturar información; por ejemplo una para capturar transacciones de tipo factoraje y otra para las de tipo arrendamiento.

9.2.2 Generalización del CIMS para poder ser usado en otras SOFOMes

Esto también podría ser una opción muy atractiva a desarrollar debido a que nos permitiría comercializar el CIMS en el floreciente mercado de las SOFOMes. Generalizar el CIMS implicaría trabajar en los siguientes puntos:

- Generalización del módulo(s) de reportes.
  Probablemente los módulos relacionados con los reportes son los que están más personalizados para funcionar dentro de Crédito Integral®. Las tablas de amortización y los estados de cuenta son los reportes que se encuentran en esta situación. Generalizarlos implicaría la creación de un panel de control en donde se pudieran configurar ampliamente estos reportes.

- Agregar funciones que comúnmente se encuentran en paquetes de software similares. A continuación de mencionan algunas.
  Actualmente el CIMS sólo contempla la existencia de dos tipos de usuarios, el primero que es el restringido[60] y el segundo relacionado con el administrador[61]. Por otro lado, se han encontrado paquetes de software para administración de SOFOMes que incluyen la creación de usuarios con cualquier combinación de privilegios existentes en el sistema **[112]** sin restringir los tipos de usuarios a sólo dos a diferencia del CIMS. También se ha encontrado que algunas aplicaciones parecidas incluyen funciones tampoco provistas por el CIMS, como la prevención del lavado de dinero **[113]**.

9.2.3 Creación de una interfaz web

Crédito Integral® solicitó que el sistema fuera desarrollado con una interfaz basada en un "cliente gordo". En este caso la implementación involucró una aplicación de escritorio y una conexión a la base de datos. Sin embargo, a fin de poder usar modelos de negocio altamente redituables como el denominado *Software As A Service*[62], resulta casi indispensable usar una interfaz Web.

---

[60] Usuario tipo restringido (según sección 6.2 del capítulo sexto): Tipo de usuario del CIMS solamente con capacidad de consultar la base de datos. La generación de reportes y exportación de los mismos está también permitida pues cumple la restricción de no modificar la base de datos impuesta a esta clase de usuario.

[61] Usuario tipo administrador (según sección 6.2 del capítulo sexto): Tipo de usuario del CIMS que tiene acceso total al sistema incluyendo lectura, modificación de datos y creación de reportes.

[62] *Software As A Service:* Modelo de venta de software en el que la licencia del mismo es otorgada al cliente a manera de servicio y por petición (*on demmand*). El modelo de negocio es parecido al pago por evento y por lo general la aplicación en cuestión es ejecutada y almacenada en servidores de la empresa que lo distribuye **[114]**.

9.2.4 Actualización automática con bancos

Actualmente, todas las actividades dentro de Crédito Integral® que podrían ser usadas como fuente de información para la generación de reportes, tienen que ser dadas de alta manualmente. Para el caso de los créditos y los clientes, esto podría no representar mayor problema teniendo en cuenta que el personal operativo de la empresa es capaz de realizar dicha tarea en un tiempo razonable. Sin embargo, si Crédito Integral® creciera sería poco práctico dar de alta las transacciones de varios créditos manualmente, la solución sería utilizar una conexión automática con los bancos involucrados. Software de finanzas personales como Quicken®[63] ya incluye dicha funcionalidad para bancos populares en el mundo **[115]** como HSBC® de E.U.A. o Citibank® también de E.U.A. Desafortunadamente esta función no está disponible para bancos mexicanos, lo que implicaría su implementación. Las razones por las cuales no está disponible para México podrían estar relacionadas al hecho de que Quicken® usa un protocolo propietario para comunicarse con los bancos y les cobra una cuota por utilizarlo **[130]**.

9.2.5 Actualización automática de sistemas contables

Actualmente, el área de contabilidad de Crédito Integral® obtiene información contable por medio del CIMS de forma manual. Posteriormente dicha información es capturada dentro de los sistemas propios del área, como el COI®. Este proceso se realiza una parte manual y otra parte de manera automática ya que el CIMS es capaz de exportar reportes en formato Excel® el COI®; sin embargo los reportes deben de ser generados e importados manualmente. Resultaría deseable que el CIMS contara con la capacidad de sincronización de información con el COI® de forma automática.

9.2.6 Factibilidad y estatus actual de funcionalidad nueva propuesta

Las funciones mencionadas en los párrafos anteriores son deseables y considerablemente factibles. A fin de poder contender en una mercado altamente competitivo, el CIMS tendrá que continuar evolucionando. Satisfacer y exceder las expectativas de los usuarios será clave en el éxito del software. Ya se está pensando poner en marcha futuras fases de desarrollo en las que se llevarán a cabo dichas mejoras. Todas ellas están enfocadas en la facilidad de uso y la versatilidad.

[63] Quicken®: Popular software de finanzas personales multiplataforma (Windows y Macintosh), versátil y propietario desarrollado por la compañía Intuit. Incluye funciones en línea como sincronización automática con bancos **[115]**.

Además están estrechamente ligadas al exitoso modelo económico de SOFOM; el cual sigue creciendo y expandiéndose dentro de nuestro país **[13]**.

Capítulo 10

Conclusiones

Este proyecto de tesis fue desarrollado a fin de proveer al sector financiero un sistema de gestión que automatice las tareas más comúnes y propensas a errores dentro de una SOFOM. El CIMS, piedra angular de esta tesis, es la herramienta que minimizará errores y acelerará procesos en las instituciones financieras de este tipo. Durante su desarrollo se obtuvieron las siguientes conclusiones:

El uso de un IDE como NetBeans para desarrollar proyectos en Java, facilita y acelera el proceso de creación de código. También provee un entorno amigable en donde convergen armoniosamente varios marcos de trabajo populares, como Spring y Struts, lo que permite su utilización con una curva de aprendizaje mínima. No obstante el manejo de ciertas características del IDE, como la generación de código que aparentemente ahorran tiempo y minimizan errores, pueden convertirse en problemas, ya que al final resulta necesario modificar el código generado a fin de obtener la funcionalidad deseada por falta de flexibilidad. Este es el caso del JSR 295 del cual se trató en el capítulo octavo.

Las herramientas visuales para el desarrollo de interfaces gráficas, como el Matisse, permiten acelerar el desarrollo de pantallas. Son instrumentos de tipo *What You See Is What You Get* (lo que ves es lo que tienes) que habilitan observar casi de manera instantánea el resultado gráfico final de la pantalla, mientras está siendo editada. La desventaja que presentan es que, como en el caso del editor gráfico de NetBeans, dificultan o imposibilitan editar el proyecto en otro software distinto al que lo creó: nula portabilidad.

Las herramientas ORM que ligan tablas en una base de datos con objetos de Java hacen prácticamente transparente el proceso de persistencia de información. En el caso del CIMS, las necesidades convencionales de almacenamiento de información permitieron usar el ORM TopLink con excelentes resultados. Algunas de las ventajas observadas al usar TopLink fueron las siguientes:

- Las líneas de código relacionadas a la persistencia se reducen enormemente.

- Las entidades a persistir se modelan con base en los conceptos reales de negocio.
- Crear consultas más complejas es más sencillo pues éstas se escriben en su propio lenguaje que es una implementación del JPQL descrito en la sección 8.2.3.
- La concurrencia alta es soportada a través del **`ConcurrencyManager`** que funciona de manera automática gestionando las transacciones de la base de datos.

El proceso de planeación del proyecto es de suma importancia. Analizar los requerimientos del software dentro de la administración del mismo es un proceso vital que puede llegar a convertirse en una importante fuente de problemas. Definir clara y meticulosamente cada uno de los puntos dentro de los requerimientos es clave para evitar malos entendidos y retrasos en el desarrollo del proyecto. Durante el periodo del desarrollo del CIMS, las fuentes de riesgos con mayor impacto fueron los requerimientos.

La herramienta utilizada para la recolección de requerimientos, denominada *job shadowing* consiste en observar los procesos que se pretenden automatizar para posteriormente documentarlos y estandarizarlos. Su uso es un método eficaz para detallar requerimientos y encontrar pormenores que los usuarios no consideran en un principio. Así podemos atacar de manera proactiva conflictos originados por solicitudes de cambio en el alcance del sistema. Varios de los detalles del CIMS, por ejemplo el *wizard* del **`Multireportes`**, derivan del uso de esta valiosa herramienta.

El CIMS es capaz de obtener reportes acumulativos compuestos a partir del resultado de los datos más simples. Por ejemplo el valor actual de la cartera procede del monto adeudado al día de hoy de todos los créditos, dicho proceso puede ejecutarse más rápido si el CIMS almacena en una base de datos los montos adeudados de créditos sin relevancia en el contexto actual.

Con la ayuda del CIMS, el personal operativo de Crédito Integral logró lo siguiente:

- Calcular rápidamente y con facilidad la rentabilidad del capital colocado como instrumentos de deuda.
- Atraer inversión externa en forma de accionistas (poco más de $5,000,000.00 durante el 2009).

- Inspirar confianza a los inversionistas pese al hecho de que el capital invertido en las SOFOMes no está asegurado por el IPAB.
- Incrementar la eficiencia de toda la empresa pues durante el 2009 se duplicaron los créditos otorgados, sin la contratación de más personal.

Cabe mencionar que, mientras los objetivos anteriores se cumplieron, algunas actividades no fueron afectadas positivamente por la implantación del CIMS dentro de Crédito Integral. El otorgamiento de créditos por parte del equipo de ventas es una de las operaciones que no se vieron beneficiadas, aunque tampoco perjudicadas. El éxito de dicho equipo siguió dependiendo directamente de la habilidad de los vendedores.

El número de SOFOMes en México siguió aumentando durante el año 2009, de hecho, varias instituciones de banca múltiple ya establecidas en el mercado actualmente otorgan créditos bajo el modelo de SOFOM, sin importar el hecho de que originalmente se constituyeron como empresas con un giro distinto.

Los modelos de negocio del sector financiero relacionados con el otorgamiento de créditos comparten los métodos de cálculo de amortizaciones. Extender el CIMS hacia otras instituciones de este tipo resulta sencillo, dada la forma modular en la que está construido.

La adecuación de los tres principios de Crystal Clear "mejora reflexiva", "comunicación por ósmosis" y "seguridad personal" generó resultados favorables reflejados en indicadores del proyecto. Las ventajas clasificadas de acuerdo al principio de la metodología adaptado se encuentra a continuación:

- Mejora reflexiva.
  Aunada a las métricas de eficiencia como valor ganado permitieron identificar los factores distractores clave en el desarrollo del software. También denotaron la baja efectividad de las juntas en las que participaron más personas de las que realmente estaban involucradas en los temas a tratar.

- Comunicación por ósmosis.
  Permitió resolver varios malos entendidos y confusiones derivados de requerimientos que antes consideraba perfectamente claros.

- Seguridad personal.
  Mejoró la comunicación y aceleró el proyecto al evitar el repetir trabajo y errores.

Este objetivo, adaptar Crystal Clear a equipos de una sola persona, se cumplió en su totalidad. Los principios anteriores de la metodología fueron modificados de tal forma que conservaran su esencia respecto a su aplicación al mismo tiempo que dieran como resultado ventajas similares a las otorgadas por versiones originales **[24]**.

El fenómeno de *scope creep*, relacionado a la generación de reportes, fue eliminado tras la llegada del **`Multireportes`**. Antes, conforme el proyecto avanzaba, la aparición de nuevos reportes continuaba sin mostrar signos de que algún día acabaría. No obstante, al analizarlos todos, noté que todos los reportes poseían muchas características en común. De una forma muy abstracta, todos los reportes son datos derivados de las tablas de amortización, filtrados por un criterio específico y mostrados de una forma en particular. Por lo tanto la solución a los reportes infinitos debía incluir una forma en la que el usuario pudiera escoger rápidamente las propiedades anteriores, al mismo tiempo que el sistema recordara sus reportes favoritos. Todo esto sin requerir de muchos clics a fin de mejorar la experiencia del usuario.

El componente gráfico que ayudó enormemente a solucionar el problema fue un *Wizard* (descripción a fondo en el capítulo 6). Normalmente los encontramos en las instalaciones de paquetes de software o en la configuración de algunas características de Windows por ejemplo la configuración de Internet. Sin embargo, no los encontramos en procesos comunes porque resulta tardado y tedioso tener que pasar por todos los pasos del *Wizard* una y otra vez para realizar tareas recurrentes. Aplicado a los reportes del CIMS, resultaría fastidioso recorrer todo el *Wizard* para elaborar diariamente el reporte de "valor de la cartera". Sin embargo, también habría que permitir al usuario modificar ciertas características de los reportes comunes.

Finalmente el problema fue resuelto al implementar un *Wizard* que fuera capaz de crear reportes desde cero, pero que permitiera guardar el tipo de reporte mientras éste era creado. El componente también permite recuperar tipos de reporte guardados y modificar sólo las características deseadas. Además, la búsqueda de clientes y créditos permite agregar con sólo un clic todos los créditos de un grupo de clientes al

Multireportes para poder generar tipos de reportes ya guardados o nuevos. Además los tipos de reportes pueden ser exportados entre los distintos clientes siempre respetando el nivel de acceso que tienen los usuarios.

Versiones futuras del CIMS estarán diseñadas para ser usadas por múltiples instituciones financieras. Actividades como distribuir actualizaciones de software y control de licencias de uso, son simplificadas considerablemente por modelos de negocio en línea como el *Software As a Service.* Además este modelo permite usar el sistema en cuestión desde cualquier computadora lo suficientemente poderosa como para ejecutar la versión de escritorio. Por estas razones es que próximamente el CIMS incluirá una interfaz Web para poder interactuar con el usuario en línea.

La facilidad de uso del CIMS, aunado a su diseño modular extensible y al hecho de que fue específicamente diseñado para las SOFOMes, da como resultado un sistema altamente especializado y a la vez pensado para el cambiante mundo financiero.

Capítulo 11

Fuentes bibliográficas, hemerográficas y electrónicas

11.1 Bibliografía

[1] Parkin, Michael, ***Economía***, 6a. edición, 2007, Prentice Hall, México.

[8] Javaswal, Samir & Shetty, Yogesh, ***Practical .NET for financial markets***, 6a. edición, 2006, Apress, EUA.

[9] Governor, James; Hinchcliffe, Dion; Nickull, Duane, ***Web 2.0 Architectures: What entrepreneurs and information architects need to know***, 1a. edición, 2009, O'Reilly, EUA.

[14] C. Ravindranath, Pandian, ***Applied Software risk management***, 1a. edición, 2007, Auerbach Publications, EUA.

[15] Royce, **Managing the development of large software systems: concepts and techniques**, 1a. edición, 1970, IEEE Computer Society Press, Los Angeles CA.

[16] Brooks,F., ***The Mythical Man-Month***, 1a. edición, 1975, Addison-Wesley, EUA.

[17] Martin,J., ***Rapid Application Development***, 1a. edición, 1991, Addison-Wesley, EUA.

[19] Gilb, T., ***Principles of Software Engineering***, 1a. edición, 1988, Addison-Wesley, Gran Bretaña.

[20] Mc Dermid, J & P Rook, ***Software Development Process Models***, 1a. edición, 1993, CRC Press, EUA.

[23] Foreman, John., ***Cleanroom software engineering, Software Technology Roadmap***, 1a. edición, 2005, Software Engineering Institute, EUA.

[24] Cockburn, Alistair, ***Crystal Clear: A human powered methodology for small teams***, 1a. edición, 2004, Adisson-Wesley professional, EUA.

[25] Procter, Paul, ***Longman Concise English Dictionary***, 1a. edición, 1985, Longman, Inglaterra.

[28] Hughes, Bob & Cotterell, Mike, ***Software Project Management***, 2a. edición, 1999, Longman, Inglaterra.

[33] C. Northcote, Parkinson, ***Parkisnon's Law: And Other Studies in Administration (Mass Market Paperback)***, 1a. edición, 1987, Ballantine Books, EUA.

[34] Bohem, Barry W., ***Software Engineering Economics***, 1a. edición, 1981, Prentice Hall, EUA.

[69] Arnoldi, Jakob, **Risk (Key Concepts)** , 1a. edición, 2009, Polity Press, EUA.

[80] Charam, Ram, **What the CEO Wants You to Know: How Your Company Really Works**, 1a. edición, 2001, Crown Business, EUA.

[92] Sommerville, Ian, **Ingeniería del software**, 7a. edición, 2005, Pearson Educación, España.

[94] Castruita, César, **ABC de productos financieros**, 1a. edición, 2008, CONDUSEF, México.

[106] Nielson, Jakob, **Usability Engineering**, 1a. edición, 1993, Morgan Kaufmann, EUA.

[38] Kalakota, Ravin, **e-Business 2.0**, 2a. edición, 2001, Addison Wesley, EUA.

[39] Silberschatz, Abraham; Korth, Henry F.; Sudarshan, S., **Database System Concepts**, 5a. edición, 2005, McGraw-Hill, EUA.

[42] Russell, Stuart & Norvig, Peter, **Inteligencia Artificial, un enfoque moderno**, 2a. edición, 2004, Pearson Education, España (versión traducida al Español).

[43] Jech, Thomas J., **Set Theory**, 3a. edición, 2006, Springer, EUA.

[44] Alhir, Sinan Si, **UML in a Nutshell**, 1a. edición, 1998, O'Reilly, EUA.

[46] Pressman, Roger S., **Ingeniería del Software**, 5a. edición, 2002, McGraw-Hill, España.

[49] Duffie, Darrel & Singleton, Kenneth J., **Credit Risk**, 1a. edición, 2003, Princeton University Press, Reino Unido.

[50] Estados Unidos Mexicanos, **Código Civil Federal**, 1a. edición, 2001, Porrúa, México.

[62] Freeman, Elisabeth & Freeman, Eric, **Head First Design Patterns**, 1a. edición, 2004, O'Reilly, EUA.

[109] Deutsch, M., **Software Quality Engineering**, 1a. edición, 1998, Prentice Hall, EUA.

[110] Davis, Alan M., **201 Principles of Software Development**, 1a. edición, 1995, McGraw-Hill, EUA.

[117] Tian, Jeff, **Software Quality Engineering - testing, quality assurance and quantifiable improvement**, 1a. edición, 2005, John Wiley & Sons, Canada.

[102] Ledgerwood, Joanna. **Microfinance Handbook: an Institutional and Financial Perspective**., 1a. edición, 2001, The World Bank, EUA.

[119] Smith, Adam, **The Wealth of Nations**, 1a. edición, 1999, Digireads.com, EUA.

[124] Weill, Peter & Ross, Jeanne W., **IT Governance: How Top Performers Manage IT Decision Rights for Superior Results**, 1a. edición, 2004, Harvard Business Press, EUA.

[125] Stellman, Andrew & Greene, Jennifer, **Applied Software Project Management**, 1a. edición, 2005, O'Reilly Media, EUA.

[129] Gámiz, Máximo, **Constitución política de los Estados Unidos Mexicanos: comentada**, 6a. edición, 2004, Limusa, México.

[132] Cohen, Daniel I.A., **Introduction to Computer Theory**, 2a. edición, 1996, Wiley, EUA.

11.2 Hemerografía

[11] Editor de la revista 2008, Getting Wired, ***The Economist***, de http://www.economist.com/node/12798277, 19 de diciembre de 2009 23:44 hrs.

[13] Quieren todos su SOFOM 2008, ***El norte (Negocios)***, http://www.elnorte.com/negocios/articulo/429/856668/, 19 de mayo de 2009 05:55 hrs.

[18] Butler,J. 1995, Rapid Application Development in Action, ***Applied Computer Research***, vol 14 no 5.

[21] Boehm, B. 1988, A spiral model for software development and enhancement, ***Computer***, vol 21 no 5.

[22] Mills, H; M. Dyer y R.Linger 1987, ***Cleanroom software engineering***, IEEE Software, IEEE Software, vol 4 no 5 pp 19–25.

[29] H, J Thamhain & D. L. Wilemon 1986, Criteria for controlling software according to plan, ***Project Management Journal***, publicación de junio de 1986

[30] Editores de la revista 1981**,** Major issues in software engineering project management, ***IEEE Transactions on Software Engineering***, Vol. 7 pp 333-342.

[32] B. A. Kitchenman y N. R. Taylor, Software Project Development Cost, ***Journal of Systems and Software***, Vol 5, I 4, noviembre de 1985, pp 267-278

[70] Maike Sundmacher, Operational Risk Capital Charges for Banks: Consideration and Consequences, **University of Western Sidney**, publicado en mayo de 2004 y consultado por última vez el 2 de febrero de 2010 a las 11:09PM, de http://papers.ssrn.com/sol3/papers.cfm?abstract_id=963227.

[72] Editores del Quality Assurance Institute, Boletín mensual del comando estratégico de los E.U.A, **Quality Assurance Institute**, Publicación de enero de 1995.

[81] **BBC de Londres**, http://news.bbc.co.uk/2/hi/business/7696766.stm, 30 de junio de 2009 19:30 hrs.

[82] Banxico baja su tasa de referencia a 4.75% 2009, **El Universal (Finanzas)**, 19 de junio de 2009.

[84] México, muy vulnerable al lavado de dinero, dice el procurador fiscal, **La Jornada (Economía)**, 5 de octubre de 2007.

[85] **SAS**, http://www.sas.com/offices/latinamerica/mexico/news/alta_resolucion.pdf, 30 de junio de 2009 19:51 hrs.

[87] Nueva alternativa para inversión en cetes, **El Economista (Finanzas personales)**, 20 de diciembre de 2001.

[107] Leonte Guzmán, **Reporte de resultados 2009 y metas próximas**, 3a edición, *No publicado en medios*, México, 2009, 22p.

[36] Editor de Gartner 2009, IT Key Metrics Data 2009: Key Information Security Measures: Organizations Including Disaster Recovery: by Region, **Gartner**, de http://www.gartner.com/DisplayDocument?doc_cd=163876, 10 de noviembre de 2009 14:35 hrs.

[51] Estados Unidos Mexicanos 2009, **Ley General de Sociedades Mercantiles**, publicada en el Diario Oficial de la Federación el 2 de mayo de 2009.

[96] **CNN Expansión**, http://www.cnnexpansion.com/expansion/2009/08/24/Desembarca-en-Mexico, 1 de marzo de 2010 3:11 hrs.

[97] **Business Week**, http://www.businessweek.com/magazine/content/07_52/b4064038915009.htm?chan=magazine+channel_in+depth&chan=top+news_top+news+index_top+story, 1 de marzo de 2010 3:12 hrs.

[99] **Condusef**, http://www.condusef.gob.mx/PDF-s/Comunicados/2010/CAT_%20tarjeta_cr%C3%A9dito_cl%C3%A1sica.pdf, 1 de marzo de 2010 3:15 hrs.

[100] **Condusef**, http://www.condusef.gob.mx/Estadisticas/boletines/Boletin0309/Bol_0309.pdf, 1 de marzo de 2010 3:16 hrs.

[122] **IXE Grupo Financiero**, SITE (Sistema Integral de Tesorería), Diagrama principal de empresas integrantes de IXE Grupo Financiero, 15 de abril de 2009.

[123] **IXE Grupo Financiero**, SALI (Sistema de Administración de Liquidez), Diagrama principal del funcionamiento del SALI, 15 de abril de 2009.

11.3 Fuentes electrónicas

[2] **Sitio oficial de la Comisión Nacional Bancaria de Valores (CNBV)**, http://www.cnbv.gob.mx/recursos/Glosario1F.htm, 12 de julio de 2009 00:10 hrs.

[3] **Sitio oficial de la Secretaría de Administración Tributaria (SAT)**, http://www.sat.gob.mx/sitio_internet/asistencia_contribuyente/principiantes/comun/23_707.html, 12 de julio de 2009 00:45 hrs.

[4] **Sitio oficial de la Secretaría de Administración Tributaria (SAT)**, http://www.sat.gob.mx/sitio_internet/informacion_fiscal/tramites_fiscales/tramites_financieros/106_8635.html, 12 de julio de 2009 01:55 hrs.

[5] **Sitio oficial de la Comisión Nacional para la Protección y Defensa de Usuarios de Servicios Financieros (CONDUSEF)**, http://www.condusef.gob.mx/index.php?option=com_content&view=article&id=693&Itemid=160, 12 de julio de 2009 03:30 hrs.

[6] **Sitio Oficial de la Secretaría de Hacienda y Crédito Público (SHCP)**, http://www.apartados.hacienda.gob.mx/sitios_de_interes/html/imagenes/banners/sofomes/guiaparaconstitucion_sofomes.pdf, 12 de julio de 2009 03:42 hrs.

[7] **Sitio oficial de la Comisión Nacional para la Protección y Defensa de Usuarios de Servicios Financieros (CONDUSEF)**, http://e-portalif.condusef.gob.mx/Revista/2008/proteja_95/reportaje_95.html, 20 de marzo de 2010 13:09 hrs.

[12] **Gestiópolis**, http://www.gestiopolis.com/recursos/experto/catsexp/pagans/fin/19/polfro.htm, 12 de julio de 2009 06:00 hrs.

[26] **Business Dictionary**, http://www.businessdictionary.com/definition/commercial-off-the-shelf-COTS.html, 11 de junio de 2009 18:44 hrs.

[27] **Sitio oficial de University of Glamorgan**, http://www.comp.glam.ac.uk/pages/staff/tdhutchings/chapter4.html, 11 de junio de 2009 23:00 hrs.

[74] **Buró de Crédito**, http://www.burodecredito.com.mx, 22 de febrero de 2010 2:14 hrs.

[75] **VW Bank**, http://www.volkswagenbank.com.mx, 22 de febrero de 2010 2:18 hrs.

[76] **Credit Rating Agencies**, http://www.defaultrisk.com/rating_agencies.htm, 22 de febrero de 2010 2:27 hrs.

[77] **Sitio oficial de la Comisión Nacional para la Protección y Defensa de Usuarios de Servicios Financieros (CONDUSEF)**, http://www.condusef.gob.mx/index.php?option=com_content&view=article&id=693&Itemid=160, 20 de junio de 2009 22:15 hrs.

[78] **Microsoft**, http://office.microsoft.com, 8 de junio de 2009 19:50 hrs.

[79] **Confederación Patronal de la República Mexicana (COPARMEX)**, http://www.coparmex.org.mx/upload/comisionesDocs/Presentacion%2520Sofom%2520B.ppt, 30 de junio de 2009 18:45 hrs.

[83] **Apestan.com**, http://apestan.com, 30 de junio de 2009 19:44 hrs.

[86] **Sitio oficial de la Comisión Nacional Bancaria de Valores (CNBV)**, http://www.cnbv.gob.mx/seccion.asp?sec_id=1&com_id=0, 30 de junio de 2009 19:54 hrs.

[88] **Sitio oficial de la Secretaría de Administración Tributaria (SAT)**, http://www.sat.gob.mx/sitio_internet/quienes_somos/127_16480.html, 22 de febrero de 2010 4:01 hrs.

[89] **Sitio oficial de la Procuraduría General de la República (PGR)**, http://www.pgr.gob.mx/Que%20es%20PGR/presentacion.asp, 30 de junio de 2009 20:00 hrs.

[90] **Sitio oficial de ASPEL**, http://www.aspel.com.mx/mx/productos/coi1.html, 30 de junio de 2009 20:15 hrs.

[91] **Oracle**, http://download.oracle.com/docs/cd/B32110_01/web.1013/b28218/undtl.htm, 30 de junio de 2009 20:16 hrs.

[93] **Primera hora**, http://www.primerahora.com/XStatic/primerahora/template/content.aspx?se=supnota&n=239147&ms=hogaryconstruccion, 06 de marzo de 2010 15:30 hrs.

[95] **Sitio oficial del Instituto de Protección al Ahorro Bancario (IPAB)**, http://www.ipab.org.mx, 06 de marzo de 2010 15:35 hrs.

[105] **Business Dictionary**, http://www.businessdictionary.com/definition/mean-time-to-repair-MTTR.html, 06 de marzo de 2010 15:35 hrs.

[35] **Techsoup**, http://www.techsoup.org/learningcenter/techplan/archives/page9763.cfm, 12 de octubre de 2009 04:33 hrs.

[37] **Oracle**, http://www.oracle.com/customers/index.html, 29 de enero de 2010 01:45 hrs.

[40] **Microsoft**,
http://msdn.microsoft.com/en-us/library/aa374872(VS.85).aspx, 12 de octubre de 2009 18:30 hrs.

[41] **Princeton**,
http://wordnetweb.princeton.edu/perl/webwn, 13 de octubre de 2009 19:30 hrs.

[45] **Object Management Group**,
http://www.omg.org/spec/UML/2.0/, 30 de enero de 2009 06:03 hrs.

[47] **MySQL**,
http://wb.mysql.com/?page_id=6, 8 de febrero de 2010 22:42 hrs.

[48] **Microsoft**,
http://office.microsoft.com/en-us/visio/HA101656401033.aspx, 8 de febrero de 2010 23:01 hrs.

[52] **Sitio oficial de la Secretaría de Hacienda y Crédito Público (SHCP)**,
http://www.sat.gob.mx/sitio_internet/princ_fisc_p/131_9895.html, 10 de febrero de 2010 2:53 hrs.

[53] **Sun Microsystems**,
http://java.sun.com/products/jlf/ed2/book/HIG.Glossary.html#51241, 10 de febrero de 2010 3:17 hrs.

[54] **Sun Microsystems**,
http://java.sun.com/docs/books/tutorial/uiswing/misc/splashscreen.html, 10 de febrero de 2010 3:22 hrs.

[55] **NetBeans**,
http://netbeans.org/, 10 de febrero de 2010 3:35 hrs.

[56] **Google**,
http://www.google.com/trends, 10 de febrero de 2010 4:03 hrs.

[57] **Sun Microsystems**,
http://java.sun.com/javaee/technologies/persistence.jsp, 10 de febrero de 2010 4:09 hrs.

[58] **Free On-Line Dictionary of Computing**,
http://foldoc.org/Application+Program+Interface, 10 de febrero de 2010 4:09 hrs.

[59] **Free On-Line Dictionary of Computing**,
http://foldoc.org/framework, 10 de febrero de 2010 4:09 hrs.

[60] **Agile Data**,
http://www.agiledata.org/essays/mappingObjects.html, 10 de febrero de 2010 4:10 hrs.

[61] **Java.Net**,
https://wizard.dev.java.net/javadoc/org/netbeans/spi/wizard/Wizard.html, 11 de febrero de 2010 22:44 hrs.

[63] **IBM**,
http://www.ibm.com/developerworks/wikis/display/xdcomputegrid/Glossary, 12 de febrero de 2010 12:56 hrs.

[64] **Sun Microsystems**,
http://java.sun.com/javaee/5/docs/tutorial/backup/update3/doc/QueryLanguage.html, 12 de febrero de 2010 12:56 hrs.

[65] **NetBeans**,
http://forums.netbeans.org/topic8317.html, 12 de febrero de 2010 12:57 hrs.

[66] **Javvin**,
http://www.javvin.com/softwareglossary/JSR.html, 12 de febrero de 2010 12:58 hrs.

[67] **Sitio oficial de University of Wollongong**,
http://staff.uow.edu.au/audit/termsandconcepts/index.html, 12 de febrero de 2010 12:59 hrs.

[68] **Java.Net**,
https://beansbinding.dev.java.net/, 12 de febrero de 2010 1:00 hrs.

[108] **Cem Kaner Ph.D.**,
http://www.kaner.com/pdfs/ETatQAI.pdf, 14 de marzo de 2010 19:00 hrs.

[111] **JUnit**,
http://www.junit.org/about, 14 de marzo de 2010 19:10 hrs.

[112] **FiNoBank**,
http://www.dycsi.com.mx/, 14 de marzo de 2010 19:11 hrs.

[113] **Anfexi**,
http://www.anfexi.com/AnfexiProducto.aspx?IdProducto=14, 14 de marzo de 2010 19:11 hrs.

[114] **SYS-CON Media**,
http://cloudcomputing.sys-con.com/?q=node/1047073, 14 de marzo de 2010 19:14 hrs.

[115] **Quicken**,
http://quicken.intuit.com/compare-quicken-personal-finance-software-products.jsp, 14 de marzo de 2010 19:14 hrs.

[116] **Microsoft**,
http://www.microsoft.com/money/faq.mspx, 14 de marzo de 2010 19:14 hrs.

[118] **Sitio oficial de Boston University**,
http://www.bu.edu/buworks/glossary/STU/, 14 de marzo de 2010 19:19 hrs.

[98] **Banco Azteca**,
http://www.bancoazteca.com/PortalBancoAzteca/publica/credito/consumo/simula_consumo.jsp,
1 de marzo de 2010 3:14 hrs.

[101] **Google**,
http://www.google.com/#hl=en&source=hp&q=sap+implementation+failures&btnG=Google+Search&aq=2&aqi=g10&aql=f&oq=sap+impl&fp=c26c79a56c95bda8,
1 de marzo de 2010 3:16 hrs.

[103] **EsMas**,
http://www.esmas.com/emprendedores/pymesint/pymechangarro/493439.html,
19 de marzo de 2010 2:03 hrs.

[104] **SAP**,
http://www.sap.com, 1 de marzo de 2010 3:18 hrs.

[120] **Sitio oficial de la Comisión Nacional para la Protección y Defensa de Usuarios de Servicios Financieros (CONDUSEF)**,
http://www.condusef.gob.mx/, 20 de marzo de 2010 22:09 hrs.

[121] **Sitio oficial de la Comisión Nacional para la Protección y Defensa de Usuarios de Servicios Financieros (CONDUSEF)**,
http://sipres.condusef.gob.mx/home/, 20 de marzo de 2010 22:09 hrs.

[126] **Internet Archive**,
http://web.archive.org/web/20070903115947/http://www.sei.cmu.edu/publications/documents/03.reports/03tr002/03tr002glossary.html, 1° de abril de 2010 21:02 hrs.

[127] **Diccionario de la Real Academia Española**,
http://buscon.rae.es/draeI/SrvltConsulta?TIPO_BUS=3&LEMA=consulta, 1° de abril de 2010 21:03 hrs.

[128] **MySQL**,
http:// www.mysql.com, 1° de abril de 2010 21:05 hrs.

[130] **About.com**,
http://financialsoft.about.com/od/quickenforbeginners/f/Q_BankList.htm, 21 de junio de 2010 6:20 hrs.

[131] **Sitio oficial de University of Texas at Dallas**,
http://www.utdallas.edu/~liebowit/book/wordprocessor/word.html, 8 de agosto de 2010 1:24 hrs.

[133] **Fincasa**,
http://www.fincasa.com.mx, 7 de diciembre de 2010 20:28 hrs.

[134] **Project Management Institute**,
http://www.pmi.org, 21 de febrero de 2011 19:00 hrs.