MODELADO DE UN SISTEMA DE PUNTO DE VENTA HACIENDO USO DE UN CATÁLOGO DE PATRONES DE ANÁLISIS BASADO EN MORENA

TANIA ARAÚJO MEJÍA
YURANI BOLAÑOS CARATAR

UNIVERSIDAD DE NARIÑO
FACULTAD DE INGENIERÍA
PROGRAMA DE INGENIERÍA DE SISTEMAS
SAN JUAN DE PASTO
2010

MODELADO DE UN SISTEMA DE PUNTO DE VENTA HACIENDO USO DE UN CATÁLOGO DE PATRONES DE ANÁLISIS BASADO EN MORENA

TANIA ARAÚJO MEJÍA
YURANI BOLAÑOS CARATAR

TRABAJO DE GRADO PRESENTADO COMO REQUISITO PARCIAL PARA OPTAR AL TÍTULO DE INGENIEROS DE SISTEMAS

ING. ALEXANDER BARÓN SALAZAR
DIRECTOR

UNIVERSIDAD DE NARIÑO
FACULTAD DE INGENIERÍA
PROGRAMA DE INGENIERÍA DE SISTEMAS
SAN JUAN DE PASTO
2010

“Las ideas y conclusiones aportadas en el trabajo de grado son responsabilidad exclusiva de sus autores”.

Artículo 1º. Del acuerdo No. 324 del 11 de Octubre de 1966 emanado del Honorable Consejo Directivo de la Universidad de Nariño.

NOTA DE ACEPTACIÓN

_____

_____

_____

_____

_____

Director: Ing. Alexander Barón

_____

_____

_____

_____

Jurado:

_____

_____

_____

_____

Jurado:

San Juan de Pasto, Noviembre de 2010

AGRADECIMIENTOS

Al amigo que nunca falla, nada sería posible sin su compañía de cada día.

Al Ingeniero Alexander Barón, quién nos ha ofrecido su conocimiento y apoyo desinteresado y nos acogió en este interesante proceso investigativo. Mil gracias Profe!!!

A nuestras familias que comparten cada vivencia, cada sentimiento, cada esfuerzo, tropiezos y aciertos y siguen ahí presentes con su amor que es el apoyo moral que nos permite continuar en el camino a recorrer.

A ti que has estado ahí para acompañarme y trasnocharte aún cuando no te correspondía… Hasta la última palabra escrita en este documento contó con tu compañía aún en la distancia y eso vale más que el oro… Gracias Corazón!!!

A mi madre que me acompaña y me guía desde el cielo, gracias a sus esfuerzos, lucha e incondicional apoyo, se hizo realidad este y muchos sueños más, tú eres la razón de este proyecto y de toda mi vida.

Gracias Totales a la vida, a la música, al amor, a la familia, a los amigos, pero siempre y ante todo gracias totales a ti mi Dios…

RESUMEN

La Reutilización de Software aparece como una alternativa para desarrollar aplicaciones y sistemas Software de una manera más eficiente, productiva y rápida. La idea es reutilizar elementos y componentes de Software en lugar de tener que desarrollarlos desde el principio.

ESKEMA tiene como objetivo proveer mecanismos que permitan introducir de manera natural en las tareas del desarrollador, la reutilización como práctica sistémica. ESKEMA facilita a las empresas la incorporación de la reutilización en su proceso de desarrollo de software, permitiendo estandarizar las tareas de reutilización (identificación, definición, construcción, almacenamiento, etiquetado, documentación, búsqueda y gestión de activos de software reutilizables).

Para el cumplimiento de este objetivo, ESKEMA provee el catálogo de patrones de análisis con una estructura que facilita los mecanismos de búsqueda, recuperación e instanciación, además de un componente gestor de experiencias de uso de los patrones de análisis.

ESKEMA incluye el Modelo de Representación de Patrones Aspectuales de Análisis – MORENA, el cual fue validado con TRIPODE que es un catálogo de patrones de análisis para el dominio POS basado en MORENA.

En consecuencia surge este proyecto de grado, el cual hace una aplicación del catálogo de patrones que propone TRIPODE, en un caso de estudio definido dentro del dominio POS, Supermercados YUCATA provee las características necesarias para que la aplicación se pueda realizar con éxito.

ABSTRACT

Software Reuse appears as an alternative to develop software applications and systems more efficiently, productively and quickly. The idea is to reuse components and software components instead of having to develop them from scratch.

ESKEMA aims to provide mechanisms to enter, naturally in the work of developers, reuse and systemic practice. ESKEMA helps businesses to incorporate reuse in their software development process, allowing reuse standardized tasks (identification, definition, construction, storage, labeling, documentation, search process and asset management of reusable software.)

To fulfill this goal, ESKEMA provides the analysis pattern catalog with a structure that provides mechanisms for search, retrieval and instantiation, and a component that manages experiences in using test patterns.

ESKEMA includes the Representation Model of Aspectual Patterns for Analysis - |MORENA, which was validated with TRIPODE which is a catalog of analysis patterns for the POS domain based on MORENA.

Consequently this final research project arises, making an application of the pattern catalog proposed by TRIPODE, a case of study defined within the POS domain. Supermarkets YUCATA provides the features needed in order to successfully create the application.

TABLA DE CONTENIDO

LISTA DE TABLAS

Pág.

LISTA DE FIGURAS

Pág.

Figura 75: Signature patrón estrategia

GLOSARIO

ACTIVO DE SOFTWARE: elementos reutilizables de software que ayuden a detectar, describir y compartir las soluciones para los problemas que se presentan en el desarrollo de software.

ACTOR: conjunto coherente de roles que juegan los usuarios de los casos de uso cuando interactúan con éstos.

AOD: Aspect-Oriented Design – Diseño orientado por aspectos

AOSD: Aspect Oriented Software Development - Desarrollo de Software Orientado por Aspectos.

ARTEFACTO: Pieza de información que es utilizada o producida por un proceso de desarrollo de software.

ASPECTO: Módulo que encapsula la implementación de un crosscutting concern.

ATRIBUTO: Propiedad con nombre de un clasificador que describe un rango de valores que pueden contener las instancias de la propiedad.

BACKUP: Copia de seguridad, con el fin de que estas copias adicionales puedan utilizarse para restaurar el original después de una eventual pérdida de datos.

BONO: Documento valor intercambiable por mercancía sobre el valor del mismo.

CAMBIO DURANTE LA VENTA: Hace referencia a los posibles flujos alternativos que se pueden presentar durante la venta (devolución, cancelación y/o adición de productos).

CARDINALIDAD: Número de elementos en un conjunto.

CASE: Computer Aided Software Engineering

CASO DE ESTUDIO: Es un método particular de investigación cualitativa. Se usa en grandes muestras, siguiendo un rígido protocolo para examinar un número limitado de variables. Los casos de estudio incluyen la profundización y examen longitudinal de un caso o evento específicos.

CÁTALOGO: Es el componente que permite la gestión de persistencia de los activos software en el cual éstos se compilan, relacionan y organizan.

CLASE: Descripción de un conjunto de objetos que comparten los mismos atributos, operaciones, relaciones y semántica.

CLIENTE: Es la persona que compra uno o más productos.

COLABORACIÓN: Representa una interacción entre objetos cuyo propósito es la realización de alguna actividad, y en donde cada objeto participante juega un determinado rol.

CONCERN: Diferentes temas o asuntos de los que es necesario ocuparse para resolver el problema.

CONTEXTO: Conjunto de elementos relacionados para un objetivo particular, como especificar una operación.

CROSSCUTING CONCERNS: Conceptos encapsulados que originalmente estaban repartidos por todo o parte de la aplicación.

CROSSCUTTING: Comportamiento transversal.

DARE: Domain Analysis and Reuse Environment

DEPÓSITO: Similar a un atributo de cantidad, con una entrada añadida para cada cambio a su nivel.

DIAGRAMA: Representación gráfica de un conjunto de elementos; representado la mayoría de las veces como un grafo conexo de nodos (elementos) y arcos (relaciones).

DIAGRAMA DE CLASES: Muestra un conjunto de clases, interfaces y colaboraciones, así como sus relaciones. Se utiliza para modelar la vista de diseño estática de un sistema.

DIAGRAMA DE SECUENCIA: Muestra la interacción de un conjunto de objetos mediante mensajes en una aplicación a través del tiempo y se modela para cada método de la clase.

DOMINIO: Sector económico donde se presentan problemas que las aplicaciones resuelven. Un sector económico puede ser dado por la actividad económica a la cual se dedica la organización

ELICITACION: Proceso mediante el cual se obtiene información del sistema que se deseado por el usuario final.

ENTRADA: Mantiene un registro de cambios en un depósito. Acumula el valor de las transacciones realizadas para luego registrarlas en un depósito.

ESKEMA: Un esquema para la gestión de catálogos de patrones de análisis.

FACTURA: Documento que refleja la entrega de un producto o la provisión de un servicio, junto a la fecha de atención, además de indicar la cantidad a pagar como contraprestación.

FACTURA TEMPORAL: Información correspondiente a la venta que se guarda en RAM hasta que se haga la ejecución de la venta

FODA: Feature Oriented Domain Analysis

INGENIERÍA DE DOMINIO: Proceso empleado para el desarrollo de aplicaciones para una familia de sistemas similares.

INSTANCIACION: Acción y efecto de crear una instancia. El crear en memoria un ejemplar de un conjunto de datos y código definido por una clase o estructura.

INTERFAZ: Capa intermedia entre el sistema POS y los sistemas con los cuales interactúa.

JAD: Joint Application Design – Desarrollo Conjunto de Aplicaciones. Esta técnica se desarrolla a lo largo de un conjunto de reuniones en grupo durante un periodo de 2 a 4 días. En estas reuniones se ayuda a los clientes y usuarios a formular problemas y explorar posibles soluciones, involucrándolos y haciéndolos sentirse partícipes del desarrollo.

MENSAJE: Especificación de una comunicación entre objeto que transmite información con la expectativa de que se desencadenará actividad.

MÉTODO: Implementación de una operación.

MODELO: Simplificación de la realidad, creada para comprender mejor el sistema que se está creando.

MORENA: Modelo de Representación de Patrones Aspectuales de Análisis.

NIVEL: Representa el valor actual del depósito, es el efecto neto de todas las entradas vinculadas al depósito.

OBJETO: Manifestación concreta de una abstracción; entidad con unos límites bien definidos e identidad que encapsula estado y comportamiento; instancia de una clase.

OPERACIÓN: Implementación de un servicio que puede ser requerido a cualquier objeto de la clase para que muestre un comportamiento.

ODM: Organization Domain Modelling

PARÁMETRO: Especificación de una variable que puede cambiarse, pasarse o ser devuelta.

PATRÓN: Activo de software que ha sido útil en un contexto práctico y probablemente será útil en los demás.

PATRÓN DE ANÁLISIS: Activo software que sirve como elemento de representación durante la fase de análisis de un sistema.

PERSISTENCIA: Se refiere a la propiedad de los datos para que estos sobrevivan de alguna manera

POS: (Point of Sale) ó Punto de Venta. Hace referencia al lugar dentro de una empresa o negocio en donde se efectúa el proceso de salida y cobro de la mercancía o productos que ofrece

PROCESO DE VENTA: En el proceso de venta se identifican tres tiempos, a saber:
Identificación y registro de productos: en donde se van enlistando los productos en la factura temporal, actualizando el total de la venta.
Pago: transacción económica que depende de la forma de pago.
Ejecución de venta: se hace persistencia de la factura temporal y se envía la información a los demás sistemas relacionados a través de la gestión de interfaces.

PROGRAMAS DE LEALTAD: Son estrategias comerciales para la adquisición de clientes y programas de fidelización de los supermercados.

RELACIÓN: Conexión semántica entre elementos.

REQUISITO: Característica, propiedad o comportamiento deseado de un sistema.

REQUISITO FUNCIONAL: Define el comportamiento interno del software cálculos, detalles técnicos, manipulación de datos y otras funcionalidades.

REQUISITO NO FUNCIONAL: Describen aquellas características que los clientes y usuarios desean que tenga el sistema a desarrollar y que de no ser cumplidas no afectan la funcionalidad básica del sistema.

REQUISITO DE INFORMACIÓN: Describen qué información debe almacenar el sistema para satisfacer las necesidades de clientes y usuarios. Identifican los conceptos relevantes sobre los que se debe almacenar información y los datos específicos que son de interés.

REPOSITORIO: Contenedor de componentes o artefactos reutilizables.

ROL: Describe las propiedades estructurales y de comportamiento de un objeto en un contexto.

ROLLBACK: En base de datos, es una operación que devuelve a la base de datos a algún estado previo. Los Rollbacks son importantes para la integridad de la base de datos, a causa de que significan que la base de datos puede ser restaurada a una copia limpia incluso después de que se han realizado operaciones erróneas.

SIGNATURE: Permite especificar la parametrización un elemento Template.

SISTEMA POS: Es un sistema compuesto por software y hardware, creado con el fin de automatizar y agilizar los procesos relacionados con ventas y las demás actividades que puedan depender de ella.

TARJETAS CRC (Clase, Responsabilidad y Colaboración) Son una metodología para el diseño de software orientado por objetos.

TEMPLATE: Es un constructor descrito en la especificación de Superestructura de UML 2.0, este es un elemento parametrizado, es denotado con un cuadro punteado a lado superior derecho dentro del cual de describen los parámetros mediante el uso de un signature.

THEME: Constructores conceptuales y de diseño propios de la aproximación Theme/UML.

THEME/UML: Extensión de UML que enfatiza en apoyar al diseñador en la especificación de Themes que se traslapan unos con otros, y en las capacidades de composición, que están definidas.

TRANSACCIÓN: Todo proceso que genera intercambio de información entre depósitos de información. Puede estar compuesta por otras transacciones.

TRIPODE: Catálogo de patrones aspectuales de análisis para el dominio POS basado en MORENA

UML: Unified Modeling Language. Lenguaje de modelado visual que se usa para especificar, visualizar, construir y documentar artefactos de un sistema de software.

UPS: Uninterrupted Power System, que en español significa Sistema de Potencia Ininterrumpida, es un dispositivo que gracias a sus baterías, puede proporcionar energía eléctrica tras un apagón a todos los dispositivos que tenga conectados.

USUARIO: La persona encargada que interactúa con el sistema POS.

VALIDACIÓN: Construcción de un modelo correcto; es el proceso de determinar si el modelo, como abstracción, es una buena representación para resolver los problemas que lo fundamentaron.

VENTA: Transferencia de un producto o servicio a un comprador mediante el pago de un precio convenido. Conjunto de transacciones.

VINCULAR: Indica que un ítem, cliente, cajero, caja, empresa o pago se agrega a la lista de la factura temporal.

1. INTRODUCCIÓN

El avance en la Ingeniería de Software se evidencia en la generación constante de nuevas propuestas metodológicas, técnicas y estrategias, como el Desarrollo Orientado por Aspectos – AOSD (Aspect Oriented Software Development), la aplicación de patrones y la reutilización de artefactos software, las cuales buscan mejorar el proceso de desarrollo y la producción de software de calidad que responda a las exigencias organizacionales y sociales.

En conjunto, estas propuestas (AOSD, patrones de análisis y reutilización de artefactos), permiten construir aplicaciones de una manera más eficiente, económica, productiva y rápida. Sin embargo, aplicadas de forma aislada contribuyen parcialmente a la solución integral de los problemas que actualmente enfrenta la industria de desarrollo de software.

Para dar respuesta a la necesidad de aplicación integrada de estos enfoques, se define ESKEMA: Esquema para la Gestión de Catálogos de Patrones de Análisis Utilizando el Enfoque Aspectual [1], liderado por el Ingeniero Alexander Barón.
Este esquema incluye el modelo de representación de patrones de análisis – MORENA, el cual provee mecanismos que permiten introducir de manera natural en las tareas del desarrollador, la reutilización como práctica sistémica a partir del uso de catálogos de patrones de análisis.

MORENA se somete a un proceso de evaluación, este proceso se divide en dos etapas, la primera etapa es la validación del modelo mediante su aplicación en la construcción de un catálogo de patrones de análisis para el dominio POS, y la segunda etapa es demostrar la funcionalidad de este catálogo, en el análisis de un caso de estudio de un sistema POS.

El resultado de la primera etapa de la validación del modelo, fue un catálogo de Patrones de Análisis, llamado TRIPODE - Catálogo de patrones aspectuales de análisis para el dominio POS basado en MORENA.

Este proyecto de investigación se encarga de realizar la segunda etapa del proceso. Para esto, se realizará el análisis de sistemas POS existentes y su clasificación, a partir del cual se determina el caso de estudio, posteriormente se desarrolla la ingeniería de requisitos del caso de estudio y finalmente se modela en la etapa de análisis, el sistema POS del caso de estudio definido, aplicando los patrones de análisis propuestos en el catálogo TRIPODE de acuerdo al modelo de Ingeniería de Aplicación propuesto por ESKEMA.

Hecha la aplicación de TRIPODE en el modelado del sistema POS para el caso de estudio definido durante la etapa de análisis, se presentan las propuestas de mejoramiento para MORENA, TRIPODE y ESKEMA.

DESCRIPCIÓN DEL PROBLEMA

La investigación en el campo de la Ingeniería de Software desde hace un tiempo nos enfrenta a una nueva aproximación para el desarrollo de software: El desarrollo basado en aspectos (AO). Buena parte de las experiencias de aplicación de esta aproximación se limitan a ejercicios de tipo académico y por lo tanto existen expectativas a nivel mundial acerca del impacto real de AO en la productividad y calidad de una solución software a escala industrial.

En consecuencia de esto, se desarrollan aplicaciones que permiten hacer un uso integral de lo que propone el Desarrollo Orientado por Aspectos, surgiendo así diversas áreas de acción, entre ellas el uso de catálogos para la reutilización de componentes por lo que genera la necesidad de un esquema para la gestión de catálogos. El esquema requerido es propuesto en el proyecto de investigación, ESKEMA: Esquema para la Gestión de Catálogos de Patrones de Análisis Utilizando el Enfoque Aspectual [1], liderado por el Ingeniero Alexander Barón.

ESKEMA incluye el Modelo de Representación de Patrones de Análisis – MORENA. Para que la potencialidad de MORENA pueda ser aplicada en contextos reales se hace necesario un proceso de evaluación que implica resolver las siguiente problemática:

Validar el modelo mediante su aplicación en la construcción de un catálogo de patrones de análisis para el dominio POS.

Demostrar la funcionalidad del catálogo, en el análisis de un sistema POS.

Para la resolución de esta problemática el grupo de investigación GALERAS – LIS de la Universidad de Nariño, propone el desarrollo de dos proyectos de investigación cada uno de los cuales es asumido por un grupo de trabajo.

Los proyectos referidos a los ítems 1 y 2 de la problemática citada se desarrollan como trabajos de grado en el Programa de Ingeniería de Sistemas en la Universidad de Nariño.

Este proyecto de investigación busca dar solución al ítem 2: Demostrar la funcionalidad de un catálogo de patrones de análisis basado en MORENA, en el análisis de un sistema POS, para tal fin, se hace uso del catálogo producto de la primera etapa - TRIPODE.

Para garantizar el desarrollo sincronizado de los dos proyectos, el grupo de investigación ha definido la siguiente relación de actividades:

Tabla 1. Relación de actividades grupo Galeras LIS

	Grupo 1	Grupo 2
Actividades	Lectura de documentos referentes de MORENA	Realizar el estudio de los POS existentes.
	Estudio de MORENA	Determinar las características de los casos de estudio POS existentes
	Taller de Identificación de elementos de Validación de MORENA	Analizar las características de los casos de estudio POS
	Búsqueda y Lectura de documentos referentes de validación de modelos	Especificar el caso de estudio POS
	Taller de diseño para el plan de validación	Realizar la elicitación del caso de estudio POS seleccionado
	Lectura de documentos referentes del dominio	Realizar la clasificación de los requisitos.
	Exploración de sistemas de información asociados al dominio	Realizar el modelo conceptual de los requisitos
	Identificar los requisitos de aplicación en el dominio	Realizar la negociación de requisitos
	Construcción inicial del catálogo TRIPODE	Realizar la validación de requisitos
	Simulación del uso del catálogo TRIPODE	Realizar el estudio de herramientas CASE
	Simulación de retroalimentación del catálogo TRIPODE	
	Identificar los elementos de mejoramiento	
	Definir la estrategia de retroalimentación al modelo	
	Construir el plan de mejoramiento de MORENA	
		Realizar el estudio del catálogo de patrones de análisis: TRIPODE
		Realizar el modelado del sistema POS haciendo uso de TRIPODE.
		Analizar los resultados obtenidos de la aplicación de TRIPODE

FORMULACIÓN DEL PROBLEMA

¿Cómo validar la aplicabilidad del catálogo de patrones de análisis basado en MORENA en un contexto real?

SISTEMATIZACIÓN DEL PROBLEMA

¿Qué caso de estudio POS es el más adecuado para aplicar el catálogo de patrones de análisis basado en MORENA?

¿Cómo establecer los requisitos que el caso de estudio POS plantea para el desarrollo del producto software?

¿Cómo aplicar el catálogo de patrones de análisis basado en MORENA en el caso de estudio POS definido?

ALCANCE

El proceso de evaluación de MORENA, se divide en dos etapas, la primera etapa es la validación del modelo mediante su aplicación en la construcción de un catálogo de patrones de análisis basado en MORENA, para el dominio POS, la segunda etapa consiste en demostrar la funcionalidad de dicho catálogo, en el análisis de un sistema POS.

Este proyecto se encarga del desarrollo de la segunda etapa, utilizando el catálogo producto de la primera etapa, que para este caso corresponde a TRIPODE. Para esto, se realizará el análisis de sistemas POS existentes y su clasificación, a partir del cual se determina el caso de estudio, posteriormente se desarrolla la Ingeniería de Requisitos para el caso de estudio y finalmente se modela el sistema POS deseado, haciendo uso de los patrones de análisis propuestos en TRIPODE.

JUSTIFICACIÓN

Para que el esquema de gestión de catálogos de patrones de análisis sea aceptado por la comunidad académica e industrial del software, se hace necesario aplicarlo en un contexto real, que permita demostrar su usabilidad en un proceso de desarrollo de software. Para tal propósito, este proyecto de investigación aplica TRIPODE que es un catálogo de patrones de análisis basado en MORENA, para realizar el modelado de la etapa de análisis de un sistema POS.

El POS (Point of Sale) hace referencia al lugar dentro de una empresa o negocio en donde se efectúan los procesos de salida y cobro de los productos o servicios que ofrece. Actualmente los sistemas POS son factor clave de éxito en el

desempeño de una organización, ya que los movimientos que aquí se generan afectan otros procesos organizacionales.

Como se indicó en apartados anteriores de este documento, el presente trabajo de investigación hace parte de un proyecto más amplio, en el cual, dada la importancia del sistema POS para las organizaciones comerciales, se ha definido utilizarlo como espacio de validación y aplicación práctica de los productos de la investigación.

Referirse a un sistema POS en general implica la observación de muchos y muy variados requisitos de software, por lo tanto para el desarrollo de esta investigación es necesario determinar cuál es el caso de estudio POS más adecuado y cuáles son sus características y condiciones específicas para el uso del catálogo, en tal sentido, este proyecto incluye la definición y especificación del caso de estudio POS a partir del cual se desarrolla la Ingeniería de Requisitos.

El caso de estudio y sus correspondientes requisitos establecen el espacio de trabajo para la aplicación del catálogo de patrones de análisis basado en MORENA, denominado TRIPODE, por lo tanto es en este espacio en donde se realiza el modelado del sistema POS para el caso de estudio definido a través de herramientas CASE.

Para hacer uso del catálogo, se aplican los procesos correspondientes a la Ingeniería de Aplicación propuestos por ESKEMA [1].

2. OBJETIVOS

OBJETIVO GENERAL

Aplicar el catálogo de patrones de análisis basado en el Modelo de Representación de Patrones de Análisis - MORENA, en el análisis de un sistema POS.

OBJETIVOS ESPECIFICOS

Definir y especificar el caso de estudio POS para aplicar el catálogo de patrones de análisis basado en MORENA.

Desarrollar la Ingeniería de Requisitos del caso de estudio definido, para determinar las características deseables del producto software.

Modelar el sistema POS para el caso de estudio definido, para validar la aplicación del catálogo.

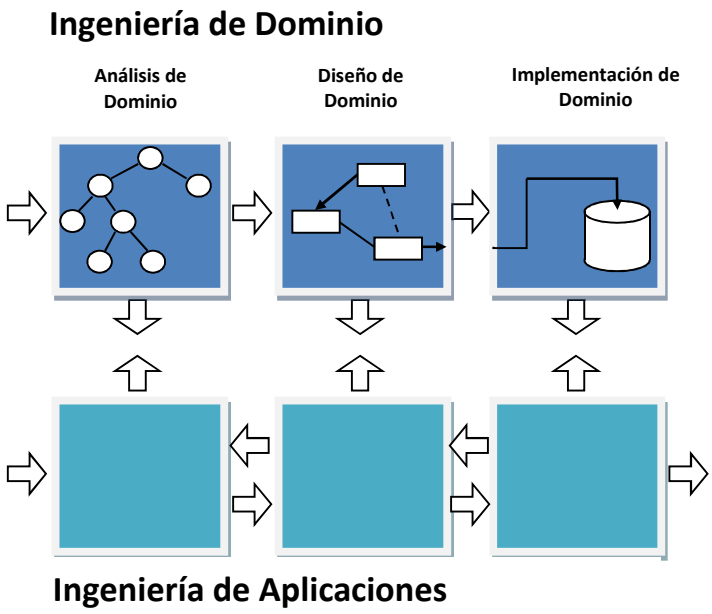
3. MARCO TEÓRICO

En este capítulo se presentan los elementos teóricos requeridos para el desarrollo de la investigación los cuales son:

3.1. INGENIERÍA DE DOMINIO

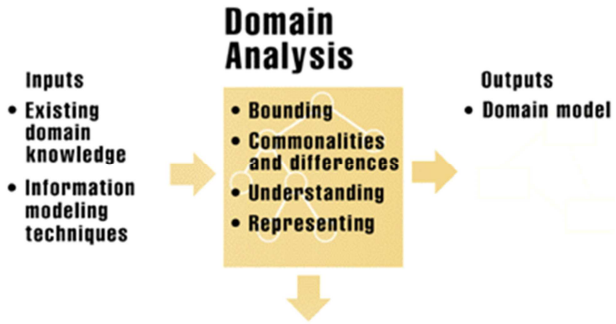
La ingeniería de dominio es el proceso empleado para el desarrollo de aplicaciones para una familia de sistemas similares. La ingeniería de dominio incluye todas las actividades requeridas para la construcción de los activos principales del software. Estas actividades son: el análisis de dominio (identificación de uno o más dominios, capturar la variación dentro de un dominio), el diseño de dominio (construcción de un diseño adaptable) y la implementación del dominio (definir los mecanismos para trasladar los requisitos a sistemas creados para la reutilización).como se distingue en la **Fig.1** Los productos o activos de software de estas actividades son modelo del dominio, modelo de diseño del dominio, lenguajes de dominio especifico, código generado y componentes de código.

Figura 1. Proceso de ingeniería de dominio



3.1.1 Análisis de dominio. El análisis de dominio es el proceso de identificación, colección, organización y representación de la información relevante en un dominio, basado en el estudio de sistemas existentes y sus desarrollos históricos, conocimiento capturado de expertos del dominio, la teoría relacionada y la tecnología emergente dentro de un dominio [2].

Figura 2. Proceso de análisis de dominio

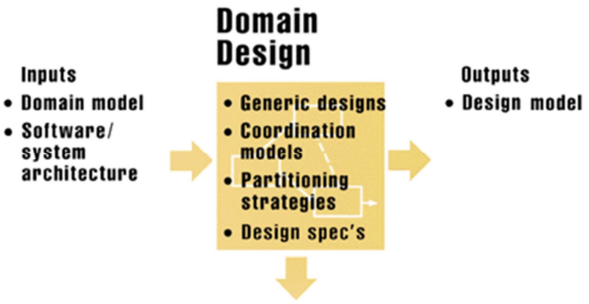


El análisis de dominio describe detalladamente el dominio estudiado, considerando los elementos comunes y los elementos diferenciales, de los sistemas en el dominio, permite la organización y la comprensión de las relaciones entre los elementos del dominio y representa la información recolectada de manera que sea útil para el desarrollo de la aplicación actual y de las aplicaciones futuras que pertenezcan a la misma familia o dominio.

En la actualidad existen numerosas técnicas de análisis de dominio. Cada una se centra en incrementar el nivel de comprensión del dominio capturando la información y representándola en modelos formales.

3.1.2 Diseño de dominio. Actualmente muchos investigadores están estudiando y clasificando arquitecturas de software, con el fin de identificar estilos de arquitectura y patrones de sistemas que permitan la reutilización en el desarrollo de nuevas aplicaciones dentro de un dominio.

Figura 3. Proceso de diseño de dominio



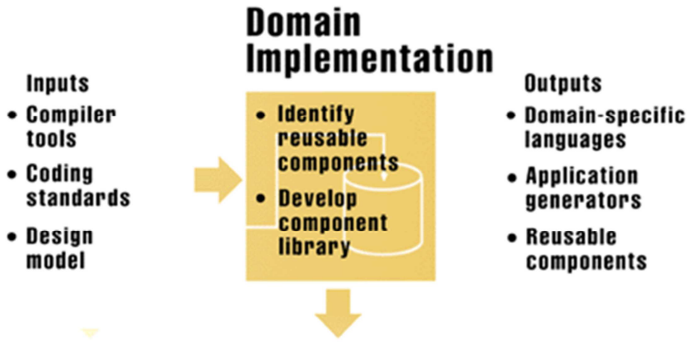
El diseño de dominio es el proceso de desarrollo de un modelo de diseño a partir de los productos del análisis del dominio y del conocimiento adquirido a partir del estudio del reuso de los requisitos, diseños y arquitecturas genéricas de software de software [2].

La clave para la reutilización es el desarrollo y documentación de arquitecturas genéricas para las aplicaciones de un dominio. El modelo de diseño representa la arquitectura genérica desarrollada para el dominio analizado y provee el marco de trabajo para el desarrollo de los componentes reutilizables en la implementación del dominio.

La arquitectura genérica es representada por un estilo arquitectónico que es el más apropiado para la familia de productos. Esta arquitectura define la plataforma computacional para todas las aplicaciones de la familia y provee las pautas para la construcción de un diseño genérico. El resultado es una estrategia de partición para la asignación de características del dominio a elementos del software y un modelo de coordinación que describe como los elementos son activados y como comparten información.

3.1.3 Implementación de dominio. La creación de componentes reutilizables ha cobrado gran trascendencia en la comunidad de ingeniería de software actual, principalmente con el uso de librerías de componentes de código. Muchos de estos repositorios no son el resultado de esfuerzos en torno a la ingeniería de dominio, por tanto, activos reutilizables de alto nivel tales como modelos de dominio o arquitecturas de referencia. El trabajo en el área de librerías de componentes tiende a centrarse en cuestiones de cualificación, clasificación, recuperación y retiro de componentes.

Figura 4. Proceso de implementación de dominio



La implementación del dominio es el proceso de identificar componentes reutilizables basados en el modelo del dominio y una arquitectura genérica [2]. Usando el conocimiento obtenido durante el análisis del dominio, y la arquitectura genérica desarrollada durante el diseño del dominio, los ingenieros del dominio adquieren y, donde sea necesario, crean los activos reutilizables que son catalogados en una librería de componentes para el uso de los ingenieros de aplicación. Estos componentes reutilizables, así como los generadores de aplicación y los lenguajes de dominio son los principales productos de esta fase en la ingeniería de dominio. La creación, gestión, y mantenimiento de un repositorio de activos reutilizables, son también tareas de las cuales se encarga la implementación de dominio.

3.2. CATÁLOGOS DE ACTIVOS DE SOFTWARE

Los catálogos han sido definidos de diversas formas, aquí se presentan algunas de ellas:

El catálogo es una lista en la que se registran, describen y ordenan, siguiendo determinadas normas, personas, cosas o sucesos que tienen algún punto en común1.

El catálogo está diseñado para potenciar la reutilización de activos prefabricados en el desarrollo de nuevas aplicaciones

El catálogo de activos se desarrolla principalmente debido a la necesidad de almacenar y recuperar la definición e implementación de estos y para automatizar el proceso de búsqueda de los mismos.

[1] BIBLIOTECAS VIRTUALES. Comunidad Virtual Literaria. [en línea] http://es.thefreedictionary.com/cat%C3%A1logo) [Citado el 21 de Febrero de 2010]

Para facilitar la aplicación de los activos, éstos se compilan, organizan y relacionan en lo que se conoce como catálogo de activos de software [3].

De lo anterior se puede deducir que un catálogo es una colección de activos de software en la cual éstos se compilan, organizan y relacionan, siguiendo determinadas normas; dotándoles de alguna estructura u organización para facilitar su selección.

Los activos de software que pueden contener los catálogos son elementos reutilizables de software que ayuden a detectar, describir y compartir las soluciones para los problemas que se presentan en el desarrollo de software. Estos elementos pueden ser: componentes, aspectos, patrones, concerns, etc.

Para la construcción de un catálogo de activos de software se debe establecer una estructura jerárquica similar a la de un sistema de ficheros que puede ser determinada por el tipo de activos de software que este contendrá, además se debe implementar un conjunto de reglas que permitan realizar operaciones como: búsqueda, eliminación, adiciones y modificaciones sobre estos activos.

3.3. PATRONES DE SOFTWARE

Los desarrolladores de software, normalmente se enfrentan a problemas que ocurren reiteradamente. En estos casos, contar con algún instrumento que ayude a detectar, describir y compartir las soluciones más apropiadas para estos problemas sería de una gran utilidad. Los patrones de software desempeñan dicha misión en el marco de la ingeniería del software.

Antes de la aparición de los patrones de software, la reutilización se había centrado en el código. Los patrones permiten la reutilización del conocimiento empírico que implica la resolución de los problemas que reiteradamente se presentan en cada etapa del desarrollo de software.

Para obtener las ventajas que genera la reutilización a partir de los patrones de software, es primordial integrarlos a los procesos de desarrollo de software. Para ello, es necesario considerar los patrones como elementos de modelado de primer orden, representarlos y describirlos de manera adecuada, y contar con un catálogo de patrones que abarque todo el proceso y facilite la selección del más apropiado en cada momento.

3.3.1. Definición. Aunque el concepto de patrón puede ser más o menos intuitivo, ciertamente es difícil encontrar una definición comúnmente aceptada para el término patrón de software. Esto es así debido a que existen muchas comunidades interesadas en su estudio y con perspectivas muy diferentes, las cuales dependen fundamentalmente de su ámbito de aplicación.

Para nuestro caso retomamos una definición genérica propuesta por Martin Fowler en [3]: “una idea que ha sido útil en un contexto práctico y probablemente será útil en los demás”. El termino idea se emplea para poner de manifiesto el hecho que un patrón puede ser cualquier cosa. Puede ser un grupo de objetos que colaboran, como los patrones del "Grupo de los Cuatro" [4] o los principios para el proyecto de organización de Coplien [5]. La frase contexto práctico refleja el hecho que los patrones son desarrollados a partir de la experiencia práctica de un proyecto real.

A menudo se dice que los patrones son más descubiertos que inventados. Esto es cierto en el sentido de que los modelos se convierten en patrones sólo cuando se da cuenta que pueden tener una cierta utilidad común. No todas las ideas de un proyecto son patrones; patrones son esas cosas que los desarrolladores piensan que pueden ser útiles en otros contextos.

3.3.2. Clasificación. Hay muchas formas posibles de clasificar los patrones, prácticamente tantas como tipos diferentes de problemas podemos encontrar a lo largo del ciclo de vida del software. No obstante, existen algunas taxonomías ampliamente conocidas que merece la pena comentar.

En primer lugar se puede hablar de la clasificación realizada por Buschmann et al. [6]. En ésta se diferencian tres tipos relacionados de patrones:

- Patrones de arquitectura. Expresan un esquema u organización básica de la estructura de un sistema software. Proporcionan un conjunto de subsistemas predefinidos, especifican sus responsabilidades e incluyen reglas y pautas para organizar las relaciones entre ellos.

- Patrones de diseño. Proporcionan un esquema para refinar los subsistemas o componentes de un sistema software, o las relaciones entre ellos. Describen estructuras recurrentes de componentes que se comunican y resuelven un problema de diseño dado en un contexto particular.

- Patrones de código (Idioms). Son los patrones de más bajo nivel, específicos para un lenguaje de programación determinado. Un "idiom" describe cómo implementar aspectos de componentes o de relaciones entre ellos usando las características de un lenguaje dado [5].

Otra importante categorización en la que, hasta donde se sabe, se introduce por primera vez el término patrón conceptual, es la que realizan Riehle y Züllighoven [6]. Aquí los autores distinguen entre:

Patrones conceptuales. Usados para facilitar la construcción del modelo conceptual de un sistema. Se describen por medio de términos y conceptos de un dominio de aplicación particular (dominio del problema). Guían la percepción que

se tiene de un dominio de aplicación y ayudan tanto a comprenderlo como a describirlo. Según los autores, éstos deberían apoyarse en metáforas del dominio de aplicación.

Patrones de diseño. Usados en la construcción del modelo de diseño. Emplean conceptos pertenecientes al dominio de la solución. Varían su escala desde "macro-arquitecturas", como pueden ser los patrones de arquitectura según Buschmann et al. [6], hasta "microarquitecturas", como los de Gamma en [4]. Los autores sostienen que el paso del modelo conceptual al modelo de diseño se puede facilitar si es posible establecer una relación clara entre patrones de diseño y patrones conceptuales.

Patrones de programación. Descrito por medio de constructores pertenecientes a algún lenguaje de programación concreto. Similar a los patrones de código o idioms [5]. Asimismo, si éstos se relacionan con los de diseño, la etapa de implementación se podría agilizar.

En consonancia con la idea de patrón conceptual, Fowler [3] acuña el término patrón de análisis para referirse a aquellos patrones que son aplicables durante la fase de análisis de un sistema.

No obstante, los patrones también pueden ser clasificados en virtud de su dominio de aplicación, dando lugar, como es lógico, a una clasificación mucho más extensa. Aunque hay patrones que son de carácter general, esto es, independientes del dominio, muchos otros son creados para resolver problemas en ámbitos concretos de aplicación.

Otra interesante tipificación es la que atiende a la etapa o aspecto que se considera dentro del proceso de desarrollo. La estructuración de un catálogo a partir de este criterio podría facilitar en gran medida la selección y aplicación del patrón más adecuado en cada instante del proceso.

3.3.3. Descripción. Los patrones van más allá de la mera exposición de una solución para un problema conocido. Un patrón debe reflejar convenientemente las razones por las cuales nos puede interesar su uso. Además, para facilitar su aplicación y comparación, la descripción debería ser realizada de una manera comprensible, clara y estructurada.

Meszaros y Doble [6], presentaron un original e interesante trabajo sobre las recomendaciones que se deben seguir para la descripción de patrones. A través de un lenguaje de patrones, los autores describen y ayudan a aplicar las mejores prácticas utilizadas para su escritura. Según estos autores, la descripción de un patrón está formada por una serie de elementos obligatorios, junto con otros opcionales. Éstos se materializan en un conjunto de secciones que dotan de

estructura la descripción. Así, los aspectos esenciales que deberían contemplarse son: nombre, contexto, problema, fuerzas, solución.

Entre los elementos considerados opcionales, los cuales pueden ser más o menos útiles dependiendo del patrón concreto que se describe, están: indicios, contexto resultante, patrones relacionados, ejemplos, código ejemplo, razón, alias, agradecimientos.

Coincidiendo con Riehle y Züllighoven [6], se piensa que la descripción de un patrón debería adaptarse a su tipo y dominio de aplicación específico, presentando de forma explícita sólo aquellas secciones que ayuden realmente a usar y entender mejor el patrón, y eliminando aquellas que se consideren superfluas o redundantes.

Uno de los formatos de descripción más famosos, conocido como formato del GoF, es el que proporciona Gamma2. Su estructura cuenta con las siguientes secciones: nombre, clasificación, intención, alias, motivación, aplicabilidad, estructura, participantes, colaboraciones, consecuencias, implementación, código ejemplo, usos conocidos y patrones relacionados.

En términos generales, la mayoría de los formatos empleados para escribir patrones constan de cuatro partes esenciales: una declaración del contexto en el que el patrón es útil, el problema que el patrón soluciona, las fuerzas que participan en la solución, y la solución que resuelve el problema. Este formato es conocido como el formato fijo Contexto-Problema-Fuerza-Solución. La estructura de este formato apoya la definición de un patrón como "una solución a un problema en su contexto". Existe otro formato más simple que incluye únicamente el par problema-solución.

Para muchos analistas el uso de un formato fijo, es un determinador de un patrón. El uso de un formato de patrón aceptado claramente marca al patrón como algo diferente a un objeto de escritura técnica.

El formato con el cual se deben escribir los patrones es un debate abierto en la comunidad de ingeniería del software, sin embargo hay consenso en que los patrones deben ser nombrados. Una de las ventajas de trabajar con patrones es la forma en que puede enriquecer el vocabulario del desarrollo y comunicar ideas de diseño muy efectivamente a través de las nomenclaturas.

[2] COLLAZOS SERRANO, VÍCTOR E. Refactorización arquitectónica de software orientada a la detección de patrones de diseño J2EE, Maestría en Sistemas y ciencias de la computación, Universidad Nacional de Colombia – UNAL, pdf. [Citado el 15 de Enero de 2009]

3.3.4 Integración en los procesos de software. Desarrollar software es un proceso indudablemente complejo. Esto es así por varias razones:

El software es complejo por naturaleza.

Normalmente es necesario trabajar en grupo.

Suelen existir problemas de comunicación con los usuarios, incluso entre los propios miembros del equipo de desarrollo.

El software evoluciona y se debe facilitar su mantenimiento.

Los proyectos pueden tener un tamaño considerable.

Al mismo tiempo hay un creciente interés por construir software fiable, gastando la menor cantidad de tiempo y dinero posible. La ingeniería del software pretende precisamente satisfacer dicha demanda, estableciendo y aplicando principios y métodos propios de la ingeniería. La reutilización de software es una de las actividades encaminadas a la consecución de dicho objetivo.

Tradicionalmente esta reutilización ha estado basada fundamentalmente en el código. Sin embargo, otros tipos de artefactos (especificaciones y modelos de requisitos, diseños, documentación, etc.), no han sido reutilizados sistemáticamente; principalmente sabiendo que la construcción de éstos repercute enormemente sobre la calidad del producto final y el tiempo de desarrollo. La intención de los patrones es dar un paso más allá a través de la reutilización sistemática del conocimiento experto empleado en la creación de todos estos artefactos.

Aparte de la ventaja que supone la reutilización del conocimiento que el experto emplea para afrontar más ágilmente los problemas que se encuentra durante la construcción de todos estos artefactos, el equipo de desarrollo se beneficia, por otro lado, de la posibilidad de compartir un vocabulario común, compuesto por los nombres de los patrones, que favorece el entendimiento entre ellos y con los usuarios finales. Adicionalmente, la inclusión de patrones mejora la documentación y, por ende, los modelos que se generan, desde el punto de vista de su legibilidad, comprensión y mantenimiento.

En definitiva, el desarrollo de software en base a patrones permite aliviar la complejidad intrínseca de su desarrollo y contribuye en la obtención de software más rentable y de mejor calidad.

3.3.5 Patrones de análisis. Durante las etapas tempranas, los desarrolladores elaboran modelos conceptuales que ayudan a entender los problemas que aparecen en el dominio de aplicación. Para agilizar y mejorar la construcción de estos modelos, es posible aplicar un tipo de patrones, conocidos como patrones de análisis, cuyo objetivo es facilitar la comprensión, comunicación y reutilización de aquellas abstracciones claves recurrentes en un determinado dominio del problema.

Fowler [3], acuña el término patrón de análisis para referirse a aquellos patrones que son aplicables durante la fase de análisis de un sistema. Los patrones de análisis capturan modelos conceptuales, vinculados a un dominio de aplicación concreto, que pueden ser reutilizados en distintas aplicaciones (reutilización del conocimiento del dominio). Su utilización tiene un objetivo doble, por una parte, acelerar el desarrollo de los modelos de análisis abstractos que capturan los principales requisitos del problema, y por otra, facilitar la transformación del modelo de análisis en el modelo de diseño.

- Aplicación de patrones de análisis.

Entre los problemas más complejos y decisivos que es necesario abordar durante el desarrollo de un sistema software están la obtención, análisis y especificación de sus requisitos.

A lo largo de este proceso, el analista desarrolla uno o varios modelos, denominados modelos de análisis o conceptuales, que ayudan a entender y simplificar los problemas que aparecen en el dominio de aplicación. Éstos complementan la especificación de requisitos realizada en lenguaje natural, representando de una forma más técnica y detallada, el contexto, estructura y comportamiento del sistema.

Estos modelos tienen un gran valor, principalmente debido a que, por un lado, son una abstracción que representa la visión (modelo mental) que tiene el analista sobre el dominio de aplicación, y por otro, son usados como puente entre el análisis y el diseño del sistema a desarrollar. Por ello, es de máxima importancia que dichos modelos estén bien construidos y sean válidos, es decir, que definan correctamente el sistema que desea el cliente.

Una prometedora técnica que permite acelerar y mejorar el desarrollo de estos modelos, a la vez que ayuda a reducir el número de errores cometidos, consiste en la aplicación de patrones de análisis.

Según Martin Fowler, “los patrones de análisis son grupos de conceptos que representan una construcción común en el modelado del negocio. Éstos pueden ser relevantes para un sólo dominio, o pueden abarcar muchos dominios” [3].

Los patrones de análisis son también ampliamente conocidos como patrones conceptuales. Riehle y Züllighoven [6] introdujeron este término para referirse a aquellos patrones aplicables en la construcción de un modelo conceptual para un dominio de aplicación concreto.

Claramente, los patrones conceptuales están centrados en el qué (dominio del problema) y no en el cómo (dominio de la solución). Representan y documentan escenarios comunes que aparecen en el dominio que comparten.

- Beneficios que aporta la reutilización de patrones de análisis

Las ventajas que supone la aplicación de patrones de análisis durante las etapas tempranas de modelado de un sistema software se pueden resumir en las siguientes:

Guían la percepción que se tiene de un dominio de aplicación y ayudan a encontrar, comprender y describir los problemas comunes que aparecen dentro de éste.

Proporcionan un lenguaje de comunicación o lengua franca [6] que facilita el entendimiento entre el personal técnico y las personas relacionadas con el dominio del problema.

Optimizan el análisis por medio de la reutilización de modelos conceptuales que ya han demostrado su validez en dominios semejantes, mejorando la calidad del software producido. Los modeladores sólo tienen que modificar y combinar estructuras existentes, más que crear nuevas desde cero [7].

Simplifican la construcción de los modelos conceptuales y facilitan su validación. Los patrones ayudan a reducir o eliminar los errores de modelado más gruesos, puesto que los elementos básicos del modelo están ya definidos [7].

Facilitan la comprensión, comunicación, documentación y mantenimiento de los modelos generados cuando los patrones utilizados se identifican claramente dentro de éstos.

Reducen el tiempo de desarrollo significativamente y mejoran la interfaz entre la etapa de análisis y de diseño [6].

A pesar de todo, los patrones no están todavía ampliamente usados durante las etapas tempranas de modelado, a diferencia de lo que ocurre en etapas posteriores.

- Catálogo de patrones de análisis de Martín Fowler

Es necesario señalar que el número de publicaciones relacionadas con este tema es de momento escaso, sobre todo en comparación con la atención que se ha prestado a otros tipos de patrones, como por ejemplo los de diseño. Además, los trabajos existentes están muy disgregados, abordando la aplicación temprana de patrones desde muy diversas perspectivas. Aumentar y aunar los esfuerzos en este terreno es una tarea que no se puede eludir. No obstante, se han desarrollado algunos trabajos significativos referentes a esta temática. Así, por ejemplo, hay autores que han concentrado sus energías en la exploración de métodos y herramientas para dar soporte al proceso de modelado conceptual en base a patrones.

Este es el caso de Fowler [3], quizás la obra más relevante e influyente dentro del movimiento que está relacionado con el estudio de los patrones de análisis. Este libro presenta una colección de 85 modelos de objetos del negocio reutilizables en diferentes ámbitos (relaciones organizacionales, cantidades, medidas, responsabilidades, análisis financiero, inventarios, contabilidad, planificación, comercio, etc.). No obstante, gran parte de estos patrones no son exclusivos de un sólo dominio, pudiendo ser aplicados en varios de ellos, lo que eleva aún más su capacidad de reutilización.

Martin Fowler presenta un catálogo de patrones de análisis para grupos de dominios de negocio. Muestra patrones para: contabilidad, observaciones y mediciones, observaciones para las finanzas corporativas, refiriéndose a los objetos, inventario y contabilidad, planificación, comercio, contratos, paquetes de comercio.

Modelos Conceptuales. El modelo conceptual es la base conceptual del catálogo de patrones de análisis de Fowler [3]. El modelo conceptual es una abstracción mental que permite entender y simplificar los problemas que se presentan en las organizaciones. Un modelo conceptual es un artefacto humano que representa una situación del mundo real. Los modelos conceptuales son efectivos, en términos de ingeniería, porque nos permiten entender fácil y rápidamente lo que está pasando en el mundo real. Para representar una situación real se puede usar más de un modelo, esto implica un proceso de análisis para elegir la mejor opción, encontrando modelos que son más apropiados para determinadas circunstancias. Lo anterior ilustra un principio importante: Los modelos no son buenos o malos; son más o menos útiles[3].

[3] Alexander, C., S. Ishikawa, MSilverstein, M. Jacobson, I. Fiksdahl'King, and S. Angel. "A Pattern Language". New York: Oxford University Press, 1977.

Para desarrollar software que cumpla con los requisitos que lo motivaron, es necesario un modelo conceptual apropiado. Normalmente el modelo más simple es el más recomendado. La elección del modelo afecta la flexibilidad y la reutilización del sistema. La tendencia es a construir modelos que provean máxima flexibilidad, esto puede ser contraproducente, ya que construir demasiada flexibilidad en un sistema puede hacerlo muy complejo, difícil de mantener y costoso, aspectos entre los cuales la ingeniería busca balance. En tal sentido los desarrolladores deben tener en cuenta que la flexibilidad debe ser añadida únicamente si es estrictamente necesaria y de frecuente uso.

3.4 DESARROLLO DE SOFTWARE ORIENTADO POR ASPECTOS - AOSD

Actualmente, el Object Oriented Software Development (OOSD) es el modelo más utilizado en el desarrollo de software. Este paradigma ha permitido un gran avance de la ingeniería del software, haciendo posible el desarrollo de sistemas y aplicaciones de alta complejidad. El paradigma orientado a objetos da solución adecuada al comportamiento funcional, pero la solución ofrecida para el comportamiento no funcional no es la más apropiada. El paradigma objetual permite realizar una separación adecuada del comportamiento no transversal, pero el comportamiento transversal no es tratado de manera apropiada.

El AOSD (Aspect Oriented Software Development) se constituye en la alternativa más visionaria para el mejoramiento del proceso de desarrollo del software, su evolución ha permitido identificar las falencias de los paradigmas precedentes y proponer estrategias técnicas de solución que se incorporan en todo el ciclo de vida del software. La orientación a aspectos se presenta como un modelo que soporta la separación de los comportamientos que representan la funcionalidad base de aquellos comportamientos que pueden ser transversales a más de un módulo. La implementación de aplicaciones de software utilizando técnicas de ASOD permite mejores estructuras de implementación que tienen impacto positivo sobre características de calidad del software tales como escalabilidad, reusabilidad y la reducción de complejidad.

3.4.1 Separación de concerns. AOSD es un paradigma de ingeniería del software que conduce a reducir la complejidad de los sistemas software a través de la separación de concerns o intereses. Los concerns son los diferentes temas o asuntos de los que es necesario ocuparse para resolver el problema. Por ejemplo una función específica que debe realizar una aplicación, pero también surgen otras como por ejemplo distribución, persistencia, replicación, sincronización, etc. La separación de concerns se refiere a la habilidad para identificar, encapsular y manipular de forma aislada aquellas partes del software que son relevantes para

un concepto dado, objetivo o propósito4, centrada en un principio en la etapa de implementación y posteriormente aplicada a todas las etapas del ciclo de vida del software.

Una de las corrientes más relevantes dentro de AOSD es MDSOC5 (Multidimensional Separation of Concerns) que promueve la encapsulación de diferentes tipos de concerns organizados en dimensiones y la integración de las mismas para formar el sistema completo. Cada una de estas dimensiones está compuesta por concerns del mismo tipo encapsulados de forma independiente. Estas piezas separadas (artefactos que encapsulan concerns) serán integradas a través de una serie de reglas de composición para formar el sistema completo. Todas las dimensiones son tratadas de forma arbitraria, posibilitando que un desarrollador se centre en determinadas características del software sin tener que conocer las demás. Por concerns del mismo tipo se pueden entender propiedades que descomponen el software atendiendo a un determinado criterio.

3.4.2 Crosscutting concerns. Son concerns cuya funcionalidad afecta a varios módulos ya definidos. Estos concerns requerirán su implementación en muchas clases o módulos, sin la técnica apropiada, produce un código enmarañado (tangling) y/o mezclado (scattering), el cual será difícil de modificar y entender.

Se entiende el scattering como la necesidad de diseminar un conjunto de requisitos a través de muchos componentes del sistema. El tangling, por el contrario, consiste en la necesidad de hacer que un sólo componente del sistema tenga que realizar todo un conjunto de requisitos.

Estos dos fenómenos, que se hacen presentes en el paradigma orientado a objetos, dificultan la reutilización, aumentan el acoplamiento entre componentes al tiempo que aumentan la complejidad del sistema y hacen que el software pierda capacidad para evolucionar.

En pocas palabras, tangling y scattering, atentan contra los atributos de calidad a los que apunta la ingeniería de software: aumentar la calidad del software, reducir sus costos y facilitar su mantenimiento y evolución.

AOSD a través de la separación Crosscutting Concerns o concerns transversales o cruzados desde las etapas tempranas del ciclo de vida del software hasta la implementación, elimina los fenómenos de tangling y scattering permitiendo la

[4] González, C. Murillo, J. Amaya, P. Un modelo de propiedades y dependencias para el análisis orientado a aspectos en MDA. Quercus Software Engineering Group. Departamento de Informática. Universidad de Extremadura. 2004.

[5] Ossher, H. Tarr, P.: "Multi-Dimensional Separation of Concerns and The Hyperspace Approach." In Proceedings of the Symposium on Software Architectures and Component Technology: The State of the Art in Software Development. Kluwer, 2000.

trazabilidad del sistema, el análisis del impacto de los cambios y la evolución del mismo, al tiempo que disminuye la complejidad e incrementa la comprensión de los diversos artefactos de requisitos, análisis, arquitectura diseño y código.

3.4.3. Descomposición aspectual. Para modularizar los crosscutting concern, los desarrolladores de software necesitan conocer y aplicar diferentes técnicas de descomposición. Muchos lenguajes de programación existentes, incluidos los orientados a objetos, los procedurales y los lenguajes funcionales, tienen en común que sus mecanismos claves de abstracción y composición, se basan en una forma de procedimiento.

La orientación a aspectos propone un nuevo tipo de modularización, que va más allá que los procedimientos: El aspecto. Un aspecto es un módulo que encapsula la implementación de un crosscutting concern. La clave de esta técnica de modularización radica en los mecanismos de composición modular. Contrario a los procedimientos que invocan explícitamente el comportamiento implementado por otros procedimientos, los aspectos tiene un mecanismo de invocación implícito. El comportamiento de un aspecto es invocado implícitamente en la implementación de otros módulos. Consecuentemente, los desarrolladores de estos otros módulos pueden ser inconscientes del crosscutting concern.

3.4.4 La base del paradigma: clases y aspectos. Un sistema orientado a aspectos puede verse como una combinación de la funcionalidad básica y el comportamiento aspectual, los cuales pueden ser combinados por medio de puntos de enlace.

Funcionalidad básica. El paradigma orientado a aspectos utiliza toda la potencia de la orientación a objetos para soportar la funcionalidad base del sistema. Por medio de objetos se implementa la funcionalidad principal del sistema, tal como puede ser la gestión de inventarios, el pago de una nómina, entre otros. La funcionalidad básica está determinada por el dominio del problema.

Comportamiento aspectual. El enfoque original con el cual surgió la orientación a aspectos identificaba el comportamiento aspectual sólo con conceptos técnicos tales como la persistencia, la gestión de errores, la sincronización o la comunicación de procesos. Hoy, el comportamiento aspectual es llevado más allá de las características técnicas del sistema y empiezan a identificarse conceptos del dominio del problema que tiene comportamiento aspectual y que cruzan el sistema.

3.4.5 Pointcut y advice. El mecanismo de invocación implícito requiere que el aspecto especifique por sí mismo donde o cuando necesita ser invocado. La implementación de un aspecto consiste en dos partes conceptualmente diferentes: El código de la funcionalidad aspectual y el código de la aplicación aspectual. El código de la funcionalidad aspectual no es esencialmente diferente del código regular y es ejecutado donde el aspecto es invocado. Esta invocación del aspecto es determinada por el código de aplicabilidad del aspecto. Este código contiene las declaraciones donde el aspecto necesita ser invocado. En la terminología de aspectos, el código de aplicabilidad del aspecto es referido como un pointcut y el código de la funcionalidad es referido como el advice del aspecto. Un simple aspecto puede consistir de múltiples y diferentes funcionalidades que necesitan ser invocados desde diferentes sitios en el código. La implementación de un aspecto puede consistir de varios segmentos de código de pointcuts y advices.

3.4.6. Tejedor (weaver). La interacción entre clases y aspectos se hace posible a través de un tercer componente, el cual conocemos como tejedor. El tejedor es el encargado de realizar la mezcla de la funcionalidad base con el comportamiento aspectual. Las clases y los aspectos se pueden mezclar de dos formas distintas: de manera estática y de manera dinámica.

- Tejido estático. Implica modificar el código fuente de una clase insertando sentencias en estos puntos de enlace. Es decir, que el código del aspecto se introduce en el de la clase. Dos características importantes del tejido estático, son: se evita un impacto negativo en el rendimiento de las aplicaciones, pero se hace difícil identificar los aspectos en el código una vez ya se ha tejido.

- Tejido dinámico. El entrelazado dinámico requiere que los aspectos existan y estén presentes de forma explícita tanto en tiempo de compilación como en tiempo de ejecución. A partir de una interfaz de reflexión, el tejedor es capaz de añadir, adaptar y borrar aspectos de forma dinámica, si así se desea, durante la ejecución.

3.4.7. Aplicaciones bajo el paradigma orientado a aspectos. Para especificar un sistema orientado a aspectos se tiene tres elementos principales principales:

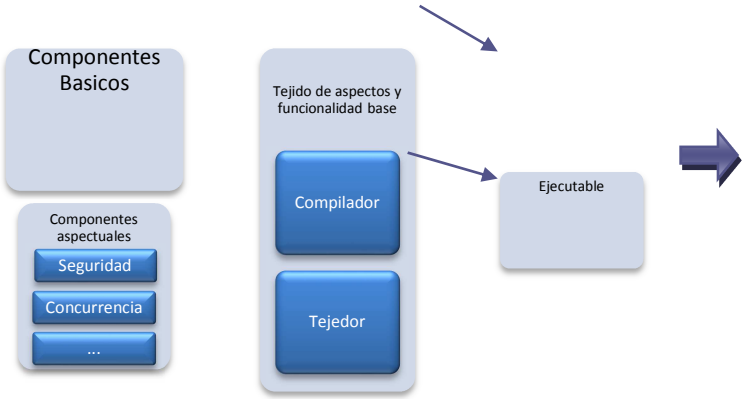
- Clases, las cuales modelan la funcionalidad base
- Aspectos
- Tejedor

En el esquema tradicional, la construcción de la aplicación se realiza a través de un compilador o intérprete, el cual simplemente toma el código de la funcionalidad base y lo traduce a un lenguaje entendible por la máquina. La construcción de aplicaciones bajo este esquema se muestra en la Figura 5.

Figura 5. Construcción de aplicaciones enfoque tradicional

La construcción de una aplicación bajo el enfoque orientado a aspectos requiere la combinación del código de la funcionalidad base con los distintos módulos que implementan los aspectos. Esta combinación es realizada por el tejedor. Cada aspecto codificado con un lenguaje distinto. La construcción de aplicaciones bajo este esquema se muestra en la Figura 6.

Figura 6. Construcción de aplicaciones Orientado por aspectos



El paradigma aspectual permite ver el sistema como un conjunto conformado por un modelo de objetos y varios modelos de aspectos. El modelo de objetos es el encargado de implementar la funcionalidad base del sistema, mientras que el modelo de aspectos permite implementar las características no funcionales (distribución, manejo de errores, sincronización entre otras) y según los enfoques más recientes, también permite implementar aspectos del dominio del problema.

3.5. LENGUAJE UNIFICADO DE MODELADO UML

Un lenguaje proporciona un vocabulario y reglas para combinar palabras de ese vocabulario con el objetivo de posibilitar la comunicación. Un lenguaje de modelado es un lenguaje cuyo vocabulario y reglas se centran en la representación conceptual, lógica y física un sistema. El lenguaje unificado de modelado UML es un lenguaje estándar para escribir planos de software. UML

puede utilizarse para visualizar, especificar, construir y documentar los artefactos de un sistema que involucra una gran cantidad de software.

El modelado proporciona una comprensión de un sistema. Nunca es suficiente con un único modelo. Para sistemas con gran cantidad de software se requiere un lenguaje que cubra las diferentes vistas de arquitectura de un sistema mientras evoluciona a través de un ciclo de vida de software.

UML es solo un lenguaje y por lo tanto es tan solo una parte de un método de desarrollo de software. UML es independiente del proceso, aunque su origen está estrechamente relacionado con RUP, el cual es un marco metodológico dirigido por los casos de uso, centrado es arquitectura, interactivo e incremental. Un proceso bien definido guiará a los usuarios al decidir los artefactos a producir, que actividades y que personal que se emplea para crearlos y gestionarlos, y como usar estos artefactos para medir y controlar el proyecto de forma global. Sin embargo UML es algo más que un conjunto de símbolos gráficos. Detrás de cada símbolo hay una notación UML bien definida. De esa manera un desarrollador puede escribir un modelo, y otro desarrollador o incluso una herramienta puede interpretar el modelo sin ambigüedad e implementar la solución. [8]

3.5.1. UML 2.0. Las características de las especificaciones UML 1.x, llevaban a problemas como excesivo tamaño, alta complejidad de uso debido a la amplia cantidad de conceptos, semántica imprecisa no apta para la generación de código ejecutable o modelos, soporte inadecuado para desarrollos basados en componentes, falta de soporte para intercambio de diagramas. Como respuesta a estos problemas OMG realiza la versión UML 2.0 que tiene como objetivos principales hacer el lenguaje más flexible, y permitir la validación y ejecución de modelos. La Versión UML 2.0 se compone de 4 estándares, como se observa en la Figura [7].

Figura 7: Paquete completo UML 2.0

- UML 2.0 superestructura. Contiene la definición formal de los constructores de UML. En este documento se especifican nuevas características y diagramas con el fin de mejorar los modelos para responder a las necesidades de mejorar el soporte para desarrollos basados en componentes, mejorar el modelado de procesos de negocios y aumentar el soporte para arquitecturas de tiempo de ejecución

(comparar modelos ejecutables) incluyendo la especificación de estructuras jerárquicas y comportamientos dinámicos.

- UML 2.0 infraestructura. Contiene los constructores básicos para la definición y adaptación de UML, y proporciona los mecanismos de extensión de UML. Surge como respuesta a las propuestas para mejorar las bases arquitectónicas de UML, incluyendo su núcleo y sus mecanismos de extensión.

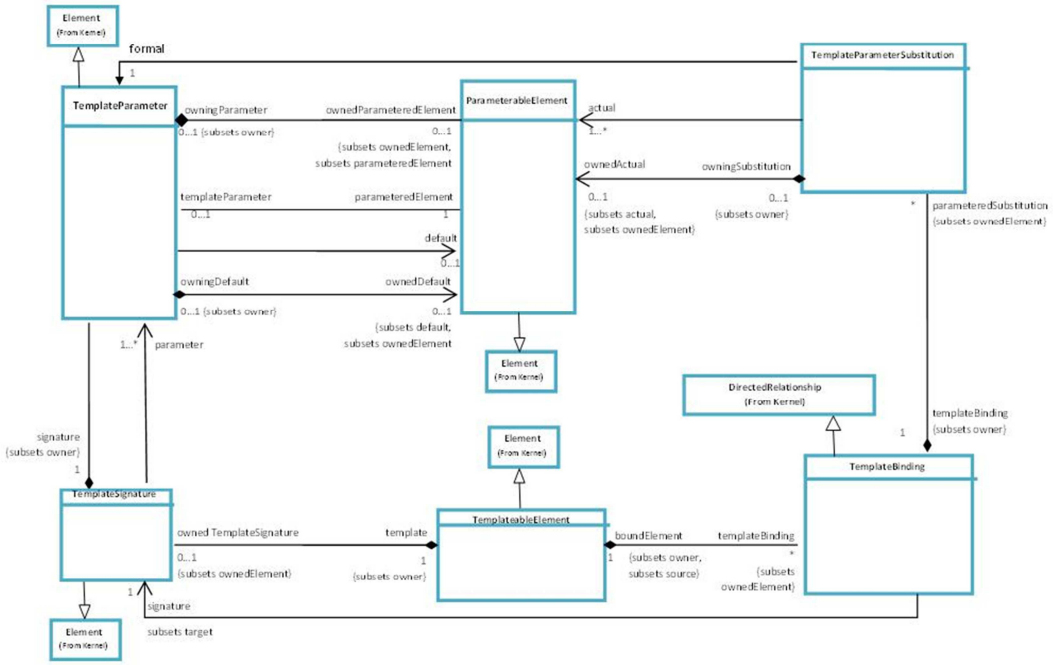
- Estándar para intercambio de diagramas. Permite compartir diagramas entre diferentes herramientas de modelado. Surge como respuesta a las propuestas que definieran un metamodelo para el intercambio de elementos de diagramas entre herramientas UML. Este metamodelo soporta el intercambio de características tales como la posición de los elementos, el agrupamiento de elementos, la alineación de elementos, las configuraciones de las fuentes, los caracteres y los colores.

- Lenguaje de restricción de objetos (OCL). Contiene la definición de un metamodelo de Lenguaje de Restricciones de Objetos (OCL) acorde al metamodelo de UML, para definir en un diagrama invariantes, precondiciones, poscondiciones y restricciones. Con esto de incrementa la precisión y consistencia de las implementaciones OCL y facilita el intercambio de constructores OCL entre distintas herramientas.

3.5.2. Templates de la especificación UML 2.0. Un Template es un constructor descrito en la especificación de Superestructura de UML 2.0, como un elemento de modelado el cual es parametrizado por otros elementos del modelo y pueden ser de tipo clasificador (Classifiers), paquete (Packages), y operación (Operations). Para especificar su parametrización un elemento Template posee una firma o Signature. Un Template también describe e identifica un patrón para un grupo particular de elementos. Los Templates se pueden usar para diseñar un único elemento de modelado que puede funcionar con diferentes tipos de datos. Un Template puede ser utilizado para generar otro modelo de elementos utilizando relaciones TemplateBinding. En esta relación los parámetros del Signature del Template que se especifican como parámetros formales serán sustituidos por parámetros actuales o los parámetros predeterminados en el enlace.

Dentro de la especificación de Superestructura de UML los constructores Templates usan 6 clases, estas determinan sus características, a continuación se hará una descripción de estas clases y se presentan ejemplos gráficos que se toman de [9].

Figura 8: Clases que brindan las características de los constructores Template



- ParameterableElement

Es un elemento de tipo clasificador, valor especifico, Propiedad u Operación que puede ser expuesto como un parámetro formal de un Template, o especificado como parámetro actual en un TemplateBinding dependiendo de la clase que lo referencie. Esta clase de define como una metaclase abstracta.

Un ParameterableElement puede ser referenciado por la clase TemplateParameter cuando éste va a ser definido como un parámetro formal de un Template, en este caso este ParameterableElement es usado como restricción para los argumentos actuales que se especificarán en la relación TemplateBinding de modo que los parámetros actuales que esta relación contenga deberán respetar el tipo de elemento que representa el ParameterableElement.

El ParameterableElement expuesto como un TemplateParameter puede ser usado en el Template como cualquier otro elemento del tipo definido en el espacio de nombres del Template y el ParameterableElement no podrá ser utilizado en otras partes del modelo.

Un ParameterableElement también puede ser referenciado por la clase TemplateParameterSubstitution donde será usado como un parámetro actual dentro de una relación TemplateBinding.

- TemplateableElement

Se describe como un elemento que puede ser definido opcionalmente como una plantilla (Template) o en un elemento vinculado (Bound Element). Un TemplateableElement puede contener un Templatesignature, el cual especifica los parámetros formales de un Template. Un TemplateableElement que contiene un Templatesignature después es referenciado como plantilla o Template.

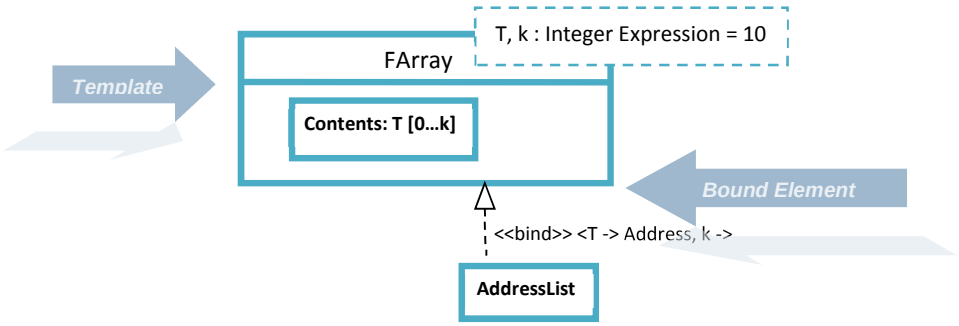
Un TemplateableElement puede contener un TemplateBinding, el cual describe como los parámetros formales son remplazados por los parámetros actuales. Un TemplateableElement que contenga un TemplateBinding es después referenciado como un elemento vinculado o Bound Element.

Si un TemplateableElement tiene parámetros de Template (TemplateParameter), un pequeño rectángulo de línea punteada se superpone en el símbolo del TemplateableElement, normalmente en la esquina superior derecha. El rectángulo punteado contiene una lista de parámetros de Template denominados formales. La lista de parámetros no puede estar vacía, aunque podría ser suprimida en la presentación. La lista de parámetros formales del Template puede ser mostrada como una lista separada por comas, o se puede presentar un parámetro formal de Template por línea. Un elemento vinculado tiene la misma notación gráfica que otros elementos del tipo al que corresponda por ejemplo una clase, un paquete etc. (ver Figura 9)

Un elemento vinculado puede tener múltiples TemplateBinding, posiblemente al mismo Template. Adicionalmente, un elemento vinculado puede contener elementos diferentes a los del Binding, cada relación TemplateBinding es mostrada usando la notación que se describe en la especificación para ella.

Un TemplateableElement puede contener al mismo tiempo un TemplateBinding como un Templatesignature. Así un TemplateableElement puede ser al mismo tiempo un Template y un elemento vinculado.

Figura 9: Templateable element



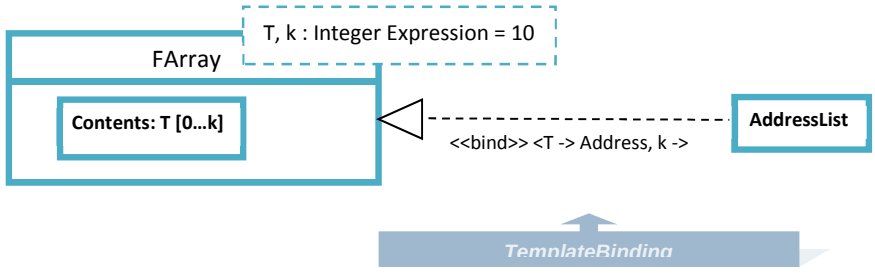
- Template Binding

Representa una relación entre un elemento vinculado y un Template. Un TempleteBinding especifica la sustitución de los parámetros actuales por los parámetros formales de un Template. A un TemplateBinding le pertenece un conjunto de TemplateParameterSubstitutions.

La semántica de una relación Template Binding es equivalente al modelo de elementos que se derivan de copiar el contenido del Template dentro de un elemento vinculado, remplazando todos los elementos expuestos como TemplatesParameter o parámetros formales con los elementos correspondientes especificados como parámetros actuales en el TemplateBinding. Si no se especifica el parámetro actual en el TemplateBinding para un parámetro formal, entonces el valor por defecto definido en el parámetro formal del Template es usado automáticamente. (Ver Figura 10)

Un TemplateBinding es mostrado como una flecha punteada con la punta en el Template y la cola en el elemento vinculado y el estereotipo <<bind>>. La información de la relación se muestra como una lista separada por comas.

Figura 10: Template binding



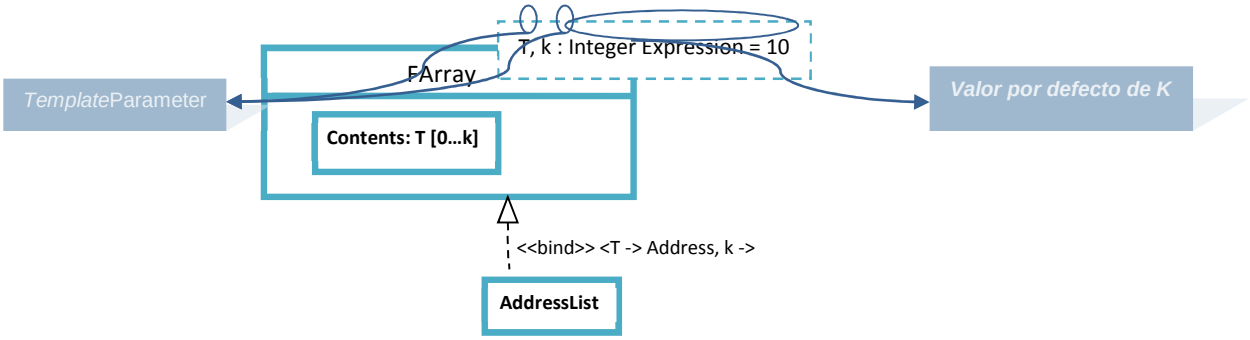
- TemplateParameter

Un TemplateParameter expone un ParameterableElement como un parámetro formal de Templates. Este ParameterableElement sólo tiene sentido dentro del Template o de otras Templates que pueden tener acceso a su funcionamiento interno (por ejemplo, si el Template soporta especialización). Los ParameterableElement expuestos como TemplateParameter no pueden ser utilizados en otras partes del modelo. (Ver Figura 11)

Cada elemento expuesto restringe a los elementos que puedan ser sustituidos por los parámetros actuales en una relación de Binding.

A un TemplateParameter se le asignar un valor por defecto que sustituirá el parámetro formal en una relación Binding, en caso que esta relación no prevea de manera explícita la sustitución el parámetro formal.

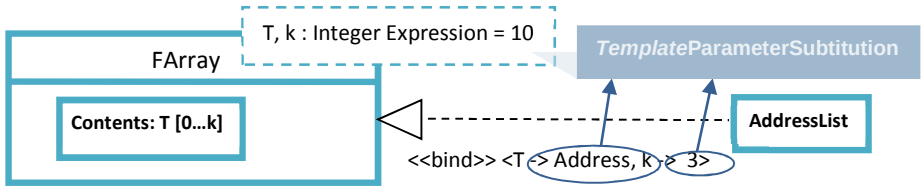
Figura 11: Template parameter



- Template Parameter Substitution

Un TemplateParameterSubstitution asocia uno o más parámetros actuales con un TemplateParameter o parámetro formal de un Template en el contexto de un TemplateBinding.

Figura 12: Temple parameter substitution

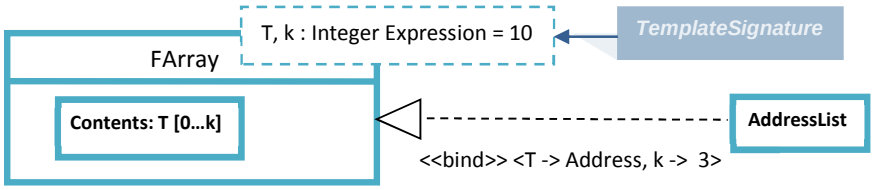


- Templatesignature

El Templatesignature es una propiedad de un TemplateableElement y la cual contiene uno o más TemplateParameters que definen el Signature para la relación entre Templates y elementos vinculados.

Un Templatesiganature especifica un conjunto TemplateParameters o parámetros formales que están asociados a un TempleteableElement. Los parámetros formales especifican los elementos que pueden ser sustituidos en un TemplateBinding.

Figura 13: Template signature



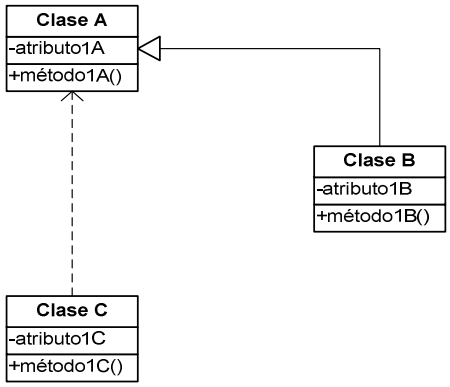
Teniendo en cuenta las clases anteriores, se puede decir que: un Template es un TempleteableElement, el cual define una lista de uno o más TemplateParameter que se especifican para producir un elemento de modelado para un sistema puntual. Este conjunto de TemplateParameter que se conocen como parámetros formales, son presentados dentro de un Templatesignature, el cual se representa con un rectángulo puenteado al borde superior izquierdo del TempleteableElement.

Un elemento vinculado también es un TemplateableElement. Al relacionar un elemento vinculado con un Template, se genera una relación de tipo TemplateBinding. Esta relación contiene los valores específicos que remplazaran a cada uno de los parámetros formales que se encuentran en el

Templatesignature del Template. Estos valores recibe el nombre de TemplateParameterSubstitution.

3.5.3 Diagramas de clases. Los diagramas de clases son los más utilizados en el modelado de sistemas orientados a objetos. Un diagrama de clases muestra un conjunto de clases, interfaces y colaboraciones, así como sus relaciones. Gráficamente, un diagrama de clases es una colección de nodos y arcos, como se muestra en la figura 14.

Figura 14: Diagrama de clases



Los diagramas de clases se utilizan para modelar la vista de diseño estática de un sistema. Principalmente, esto incluye modelar el vocabulario del sistema, modelar las colaboraciones o modelar esquemas. Los diagramas de clases también son la base para un par de diagramas relacionados: los diagramas de componentes y los diagramas de despliegue.

Los diagramas de clases son importantes no sólo para visualizar, especificar y documentar modelos estructurales, sino también para construir sistemas ejecutables, aplicando ingeniería directa o inversa.

Los diagramas de clases contienen normalmente los siguientes elementos:
Clases
Interfaces
Relaciones de dependencia, generalización y asociación

Al igual que los demás diagramas, los diagramas de clases pueden contener notas y restricciones.

Los diagramas de clases también pueden contener paquetes o subsistemas, los cuales se usan para agrupar los elementos de un modelo en partes más grandes. A veces se colocarán instancias en los diagramas de clases, especialmente cuando se quiera mostrar el tipo (posiblemente dinámico) de una instancia [3].

- Clases. Las clases son los bloques de construcción más importantes de cualquier sistema orientado a objetos. Una clase es una descripción de un conjunto de objetos que comparten los mismos atributos, operaciones, relaciones y semántica. Una clase implementa una o más interfaces. Gráficamente, se representa como un rectángulo.

Las clases se pueden utilizar para capturar el vocabulario del sistema que se está desarrollando. Estas clases pueden incluir abstracciones que formen parte del dominio del problema, así como las clases que constituyan una implementación. Se pueden utilizar las clases para representar cosas que sean software, software o puramente conceptuales.

Las clases bien estructuradas están bien delimitadas y forman parte de una distribución equilibrada de responsabilidades en el sistema [8].

- Las partes más importantes de una clase son:

Nombre: cada clase ha de tener un nombre que la distinga de otras clases. Un nombre es una cadena de texto. Ese nombre sólo se denomina nombre simple; un nombre de camino consta del nombre de la clase precedido por el nombre del paquete en el que se encuentra. El nombre de una clase debe ser único en su paquete.

- Atributos: un atributo es una propiedad de una clase identificada con un nombre, que describe un rango de valores que pueden tomar las instancias de la propiedad. Una clase puede tener cualquier número de atributos o no tener ninguno. Un atributo representa alguna propiedad del elemento que está modelando que es compartida por todos los objetos de esa clase.

- Operaciones: una operación es la implementación de un servicio que puede ser requerido a cualquier objeto de la clase para que muestre un comportamiento. En otras palabras, una operación es una abstracción de algo que se puede hacer a un objeto y que es compartido por todos los objetos de la clase. Una clase puede tener cualquier número de operaciones o ninguna [8].

- Interfaces. Las interfaces definen una línea entre la especificación de lo que una abstracción hace y la implementación de cómo lo hace. Una interfaz es una colección de operaciones que sirven para especificar un servicio de una clase o de un componente. Gráficamente, una interfaz se representa como un círculo; de forma expandida, una interfaz se puede ver como una clase estereotipada para mostrar sus operaciones y otras propiedades.

Las interfaces se utilizan para visualizar, especificar, construir y documentar las líneas de separación dentro de un sistema. Una interfaz bien estructurada

proporciona una clara separación entre las vistas externa e interna de una abstracción, haciendo posible comprender y abordar una abstracción sin tener que sumergirse en los detalles de su implementación [8].

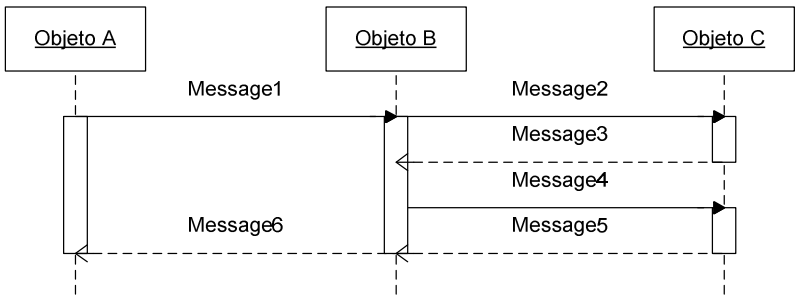
- Relaciones. Al realizar abstracciones, uno se da cuenta de que muy pocas clases se encuentran aisladas. En vez de ello, la mayoría colaboran con otras de varias maneras. Por lo tanto, al modelar un sistema, no sólo hay que identificar los elementos que conforman el vocabulario del sistema, también hay que modelar cómo se relacionan estos elementos entre sí.

Una relación es una conexión entre elementos. Gráficamente, una relación se representa como una línea, usándose diferentes tipos de línea para diferencia los tipos de relaciones.

En el modelado orientado a objetos hay tres tipos de relaciones especialmente importantes: dependencias, que representan relaciones de uso entre las clases (incluyendo refinamiento, traza y ligadura); generalizaciones, que conectan clases generales con especializaciones, y asociaciones, que representan relaciones estructurales entre los objetos. Cada una de estas relaciones proporciona una forma diferente de combinar las abstracciones [8].

3.5.4. Diagramas de secuencia. Un diagrama de secuencia destaca la ordenación temporal de los mensajes. Como se muestra en la figura 15, se forma colocando en primer lugar los objetos que participan en la interacción en la parte superior del diagrama, a lo largo del eje X. Normalmente, se coloca a la izquierda el objeto que inicia la interacción, y los objetos subordinados a la derecha. A continuación, se colocan los mensajes que estos objetos envían y reciben a lo largo del eje Y, en orden de sucesión en el tiempo, desde arriba hasta abajo. Esto ofrece al lector una señal visual clara del flujo de control a lo largo del tiempo.

Figura 15: Diagrama de secuencia

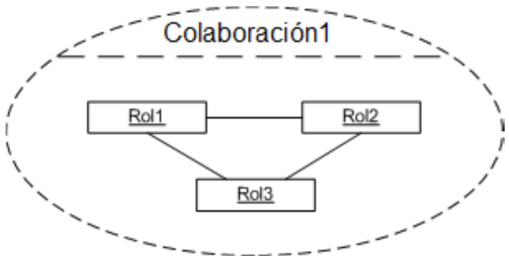


Los diagramas de secuencia tienen dos características que los distinguen. En primer lugar, está la línea de vida. La línea de vida de un objeto es la línea discontinua vertical que representa la existencia de un objeto a lo largo de un periodo de tiempo.

En segundo lugar, está el foco de control. El foco de control es un rectángulo delgado y estrecho que representa el periodo de tiempo durante el cual un objeto ejecuta una acción, bien sea directamente o a través de un procedimiento subordinado. La parte superior del rectángulo se alinea con el comienzo de la acción; la inferior se alinea con su terminación [8].

3.5.5. Colaboraciones. Una colaboración representa una interacción entre objetos cuyo propósito es la realización de alguna actividad, y en donde cada objeto participante juega un determinado rol. Gráficamente, una colaboración se representa como una elipse con el borde discontinuo, como lo muestra la figura 16.

Figura 16: Colaboración



Una colaboración no es propietaria de ninguno de sus elementos estructurales, sino que los referencia o usa. Por esto un mismo elemento puede formar parte de más de una colaboración.

Una colaboración tiene dos aspectos: el aspecto estructural o estático consiste en un conjunto de roles y las relaciones estructurales entre ellos; el aspecto dinámico o de comportamiento consiste en una o más interacciones entre los roles participantes en la colaboración.

Un rol describe las propiedades estructurales y de comportamiento de un objeto en un contexto. Un objeto juega uno o más roles a lo largo de su existencia y un mismo rol puede ser jugado por diferentes objetos. Dado que un objeto finalmente será instancia de una clase, un rol es una proyección (vista parcial) de dicha clase, de modo que las propiedades del rol serán un subconjunto de las propiedades de la clase del objeto. Una clase implementará uno o más roles y un rol podrá ser implementado por una o más clases [10].

3.6 THEME/UML

Theme/UML es una aproximación de diseño orientada por aspectos. Es una extensión de UML que enfatiza en apoyar al diseñador en la especificación de Themes que se traslapan unos con otros, y en las capacidades de composición, que están definidas.

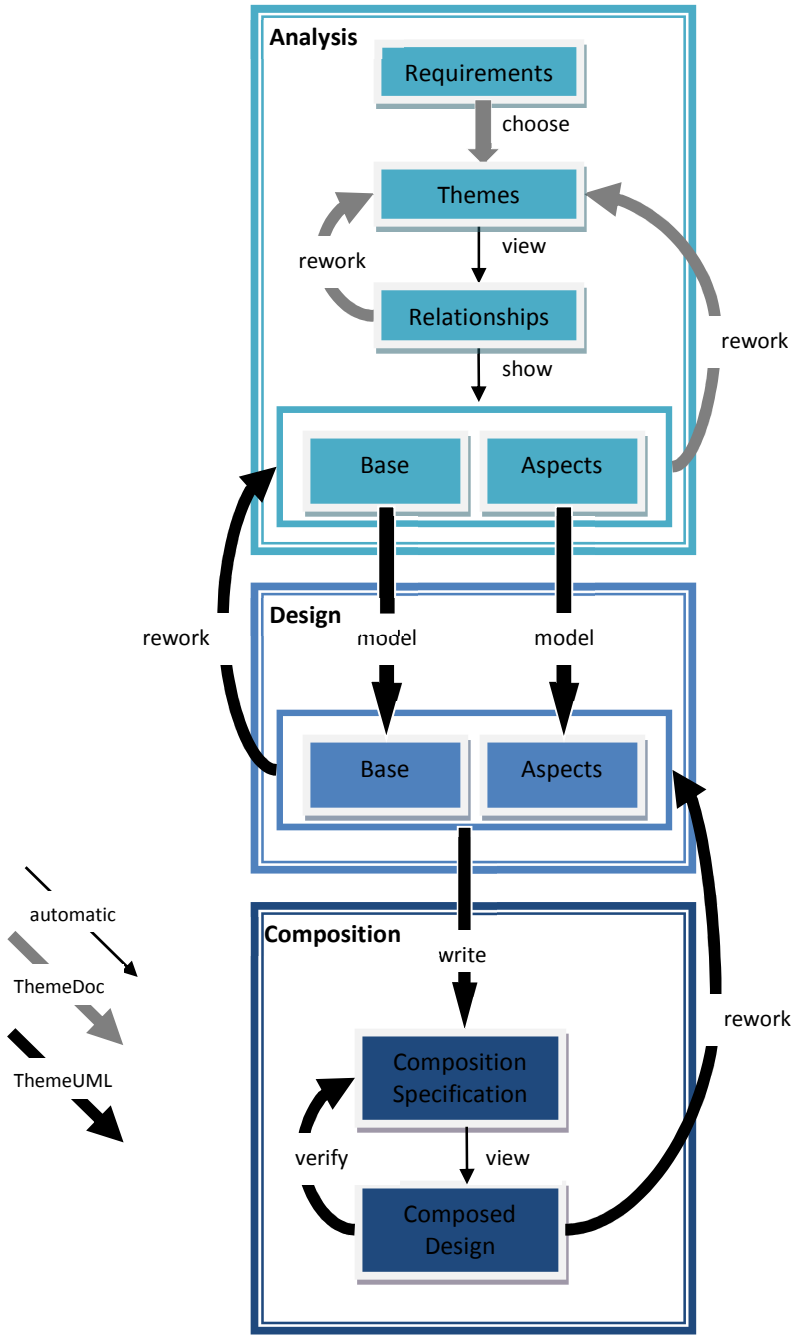
En todas las aproximaciones orientadas a aspectos, debe haber una manera para definir como relacionar los aspectos con el resto del sistema, para proveer esta capacidad, Theme/UML tiene definida una nueva clase de relaciones, llamada una relación de composición, que permite al diseñador identificar estas partes en el diseño del Theme y por lo tanto como debería ser compuesto. Para los Themes que entrecruzan otros, estos medios identifican cuando y donde en estos otros Themes, debe ocurrir el comportamiento adicional, para otras clases de traslapo, estos medios identifican elementos de diseño en el Theme que correspondan el uno con el otro y dicen cómo deben ser integrados.

Theme/UML también provee semánticas para composición de modelos basados en la especificación de relaciones de composición, este soporta un nivel de verificación que permite que el diseñador considere el diseño del sistema total, incluyendo la especificación de composición, para asegurar que tenga sentido Theme/UML es parte de una aproximación para análisis orientado a aspectos y diseño llamado Theme. Theme tiene dos partes: Theme/Doc es un conjunto de heurísticas y herramientas para la visualización y documentación de análisis de requerimientos de software con el propósito de encontrar el Theme a ser diseñado, Theme/UML es la segunda parte, la cual es una aproximación de diseño orientado por aspectos basados en UML.

3.6.1. Proceso THEME/UML. El proceso del Theme se describe6 como un proceso de tres fases, análisis, diseño y composición. En la fase de análisis, se identifican y se caracterizan los Themes. En la fase de diseño los Themes identificados y caracterizados se especifican en modelos técnicos del diseño. Finalmente, en la fase de la composición, se especifica la composición de Themes. El proceso completo se ilustra en la Figura 17.

[6] Chitchyan, R. Rashid,A. Sawyer, P. Garcia, A. Pinto, M. Bakker,J. Tekinerdogan, B. Clarke, S. Jackson, A. Survey of Aspect-Oriented Analysis and Design Approaches. AOSD-EUROPE. 2005

Figura 17: Proceso completo Theme



3.7. MODELO DE INGENIERIA DE REQUISITOS DE AMADOR DURAN

En los últimos años la comunidad de Ingeniería de Software, le ha dado mayor importancia al estudio de modelos y metodologías que fortalezcan a la Ingeniería de Requisitos, como la etapa previa al ciclo de desarrollo de software, integrando de esta manera a la Ingeniería de Requisitos para que haga parte del ciclo de vida.

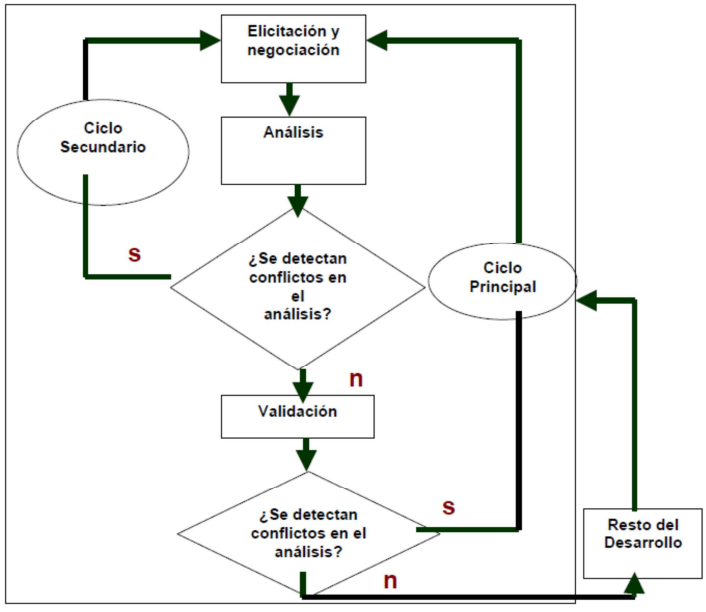
En el documento de ingeniería de requisitos7 de Alberto Restrepo se presenta el modelo propuesto por Amador Durán, el cual será el modelo aplicado en el presente proyecto de investigación para desarrollar la Ingeniería de Requisitos en el caso de estudio tratado.

A continuación se describe el proceso que Durán propone para la realización de la Ingeniería de Requisitos.

3.7.1. Descripción del modelo. La descripción que en esta sección se presenta es la descrita en [6], la cual comprende una explicación de los ciclos que distinguen esta propuesta de los demás modelos y de las actividades que se realizan.

Durán propone tres actividades principales: Elicitación, Análisis y Validación las cuales se relacionan como se muestra en la Figura 18.

Figura 18: Modelo de ingeniería de requisitos de Amador Durán



[7] Restrepo Alberto. Ingeniería de requistos.- Documento basado en tesis doctoral de Amador Durán

Ciclos de Iteración

Como se observa en la Figura 18, en el modelo propuesto se definen tres posibles ciclos de iteración, dos internos al proceso: elicitación/negociación-análisis-validación y elicitación/negociación-análisis, y uno externo, que se da entre la Ingeniería de Requisitos y el resto del desarrollo.

- Ciclo elicitación/negociación–análisis–validación

El ciclo principal elicitación–análisis–validación es un ciclo interno que indica la posibilidad de que durante el proceso de validación de los requisitos por parte de los clientes y usuarios aparezcan conflictos o nuevos requisitos que hasta ese momento estaban ocultos. Por lo tanto, es necesario solucionar dichos conflictos y llegar a un consenso sobre los nuevos requisitos mediante nuevas reuniones de elicitación/negociación, repitiendo a continuación las actividades de análisis y validación.

- Ciclo elicitación/negociación–análisis

El segundo ciclo interno, elicitación/negociación–análisis, es un ciclo que señala la posibilidad de que durante la realización del análisis de los requisitos elicitados se descubran conflictos o deficiencias en dichos requisitos, lo que puede llevar a que sean necesarias nuevas reuniones de elicitación/negociación y el posterior análisis de sus resultados.

- Ciclo ingeniería de requisitos–resto del desarrollo

Este último ciclo, entre la ingeniería de requisitos y el resto del desarrollo, da entender que se puede dar la posibilidad de que durante el resto del desarrollo sea necesario volver a alguna de las actividades de ingeniería de requisitos, posiblemente porque se detecte la necesidad de renegociar algunos requisitos de difícil implementación, porque aparezcan nuevos requisitos durante el desarrollo, etc.

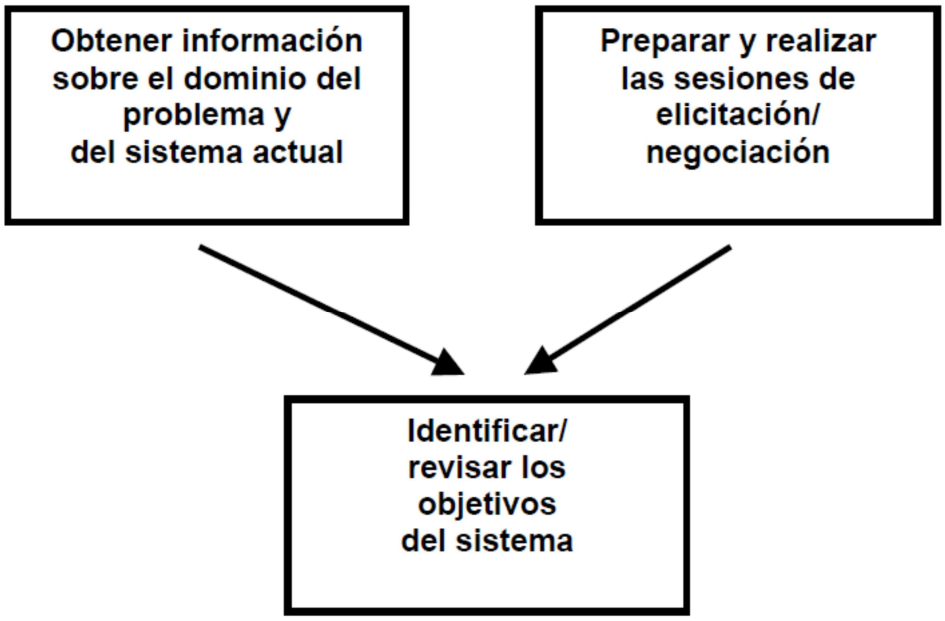
3.7.2 Elicitación/negociación de requisitos. La elicitación de requisitos no se ha considerado parte del ciclo de vida de desarrollo de software hasta hace relativamente pocos años. Se acostumbraba suponer que el cliente proporcionaba los requisitos, de forma que el ciclo de vida comenzaba siempre por el análisis de unos requisitos ya dados, ya que las actividades de elicitación y validación no se consideraban necesarias. Cuando se ha detectado que los problemas en los requisitos son uno de los principales factores de los fracasos de los proyectos

software8, es cuando se le ha comenzado a dar importancia al proceso de obtención de esos requisitos.

- Propuesta metodológica para la elicitación de requisitos

En la tesis doctoral [11], se propone una serie de tareas a realizar y resultados a obtener, tanto internos como externos o entregables. Las tareas que se contemplan dentro de la metodología, se ven en la siguiente gráfica:

Figura 19: Propuesta metodológica de elicitación de Amador Durán



A continuación se hace una descripción de cada una de las tareas:

[8] Contracting for Computer Software Development: Serious Problems Require Management Attention to Avoid Wasting Additional Millions. Report FGMSD–80–4, U. S. Government Account Office, Noviembre 1979. en [Davis 1993], [Christel y Kang 1992] y [Goguen 1994].

- Tarea 1: Obtener información sobre el dominio del problema y del sistema actual

Tiene como objetivo conocer el dominio del problema. Hay tres razones para ello:

El ingeniero de requisitos debe conocer el lenguaje de los clientes y usuarios para lograr una mejor comunicación. Como se expresa9 [32], cada dominio de problema posee un vocabulario propio el cual es necesario conocer.

El ingeniero de requisitos debe evitar la utilización de sus propios esquemas y categorías mentales en el momento de obtener la información, para que no se dificulte la comunicación10. Se debe aprender a pensar en los mismos términos que emplean clientes y usuarios. Se recomienda hacer un estudio del sistema actual, si existe.

Si no se conoce el dominio del problema, se pueden dar malentendidos en las sesiones de elicitación, o se puede dar una pérdida de confianza hacia el equipo de ingeniería de requisitos que lleva a que algunos clientes y usuarios den información aparentemente completa o que se resistan a involucrarse en el proyecto11.

Como resultado de esta tarea se obtienen productos internos basados en la búsqueda de información sobre el dominio del problema y la situación del sistema actual:

- Modelos del sistema actual.
- Documentación de la organización.
- Resultados de entrevistas.
- Resultados de cuestionarios exploratorios.
- Documentación de desarrollos previos sobre el mismo dominio del problema.
- Información proveniente de expertos, etc.

---

[9] H. D. Rombach. Software Specifications: A Framework. Curriculum Module SEI–CM–11–2.1, Software Engineering Institute, Carnegie Mellon University, 1990. http://www.sei.cmu.edu , aunque aparece en [Dorfman y Thayer 1990].

J. W. Brackett. Software Requirements. Curriculum Module SEI–CM–19–1.2, Software Engineering Institute, Carnegie Mellon University, 1990. http://www.sei.cmu.edu . [CCT 1990] CCTA. SSADM Version 4.2, 1990.

[10] J. A. Goguen. Requirements Engineering as the Reconciliation of Social and Technical Issues. En Requirements Engineering: Social and Technical Issues, páginas 165–199. Academic Press, 1994. http://www.cse.ucsd.edu/˜goguen .

[11] J. Rumbaugh, M. Blaha, W. Premerlani, F. Eddy, y W. Lorensen. Object-Oriented Modeling and Design. Prentice–Hall, 1991

Como técnicas se puede utilizar la recopilación de documentación, entrevistas, reuniones grupales, cuestionarios, inmersión en el negocio del cliente12, modelado del sistema actual13, etc.

- Tarea 2: Preparar y realizar las sesiones de elicitación/negociación

El principal objetivo de esta tarea es conocer las necesidades de clientes y usuarios y solucionar los conflictos que se han identificado en las actividades de análisis previas. Esta actividad es crucial, dado que es en ella donde se da una mayor interacción personal entre clientes y usuarios y el equipo de ingeniería de requisitos.

Los productos resultantes de esta tarea también son internos y comprenden:

- Notas tomadas durante las sesiones.
- Grabaciones de audio o video.
- Conflictos resueltos durante las sesiones si estos se han dado conjuntamente con las necesidades de cambio en los requisitos del cliente.

Como técnicas, se pueden emplear entrevistas, sesiones JAD, brainstormig y las plantillas para elicitación propuestas por Amador Durán.

- Tarea 3: Identificar/revisar los objetivos del sistema

El objetivo de esta tarea es conocer por qué se acomete el desarrollo, y por lo tanto, qué objetivos se esperan alcanzar mediante el sistema software a desarrollar. Dado que el proceso de elicitación es iterativo, en la primera iteración se realiza una primera identificación de los objetivos, y en iteraciones posteriores es posible que sea necesario revisarlos si se han presentado conflictos que los afecten.

[12] J. A. Goguen y C. Linde. Techniques for Requirements Elicitation. En Proceedings of the First International Symposium on Requirements Engineering, 1993. [Thayer y Dorfman 1997]. http://www.cse.ucsd.edu/˜goguen .

[13] F. J. García. Modelo de Reutilización Soportado por Estructuras Complejas de Reutilización Denominadas Mecanos. Tesis doctoral, Universidad de Salamanca, 2000.

Como puede inferirse, la idea básica es ir obteniendo requisitos como un refinamiento de los objetivos, de forma que la existencia de un requisito esté justificada como una necesidad para alcanzar uno o más objetivos. Esta es una de las relaciones de prerrastrabilidad[14] que se contempla en el modelo propuesto.
Los productos resultantes son los objetivos del sistema expresados mediante plantillas para objetivos, que se describirán más adelante.

Como técnicas, se utilizan el análisis de factores críticos de éxito15 o alguna otra técnica similar para identificación de objetivos.

- Tarea 4: Identificar/revisar los requisitos de información

El objetivo de esta tarea y las siguientes es identificar, o revisar en función de posibles conflictos, los requisitos-C a partir de la información obtenida en las tareas anteriores. No se da una secuencia de realización entre estas tareas, sino que se pueden realizar en forma paralela.

El objetivo de la actual tarea, es identificar o revisar los requisitos de almacenamiento de información que deberá satisfacer el sistema.

Los productos que se obtienen en esta tarea son los requisitos de almacenamiento de información, expresados en las plantillas que Durán propone en su modelo.

- Tarea 5. Identificar/revisar los requisitos funcionales

Su objetivo es identificar o revisar los requisitos funcionales que el sistema debe satisfacer, para lo cual se ha optado por la utilización de casos de uso. Como se lleva a cabo en la elaboración de casos de uso, es preciso identificar y describir a los actores del sistema. Adicionalmente, un aspecto fundamental es determinar el ámbito del sistema16, esto es, identificar aquellos aspectos que son responsabilidad del sistema y los que se gestionarán manualmente o por otro procedimiento. Por esto, la utilización de los casos de uso permitirán especificar en forma clara que queda dentro y fuera del ámbito del sistema.

Los productos resultantes de esta tarea, son los diagramas de casos de uso.

---

[14] C. Rolland, C. Ben Achour, C. Cauvet, J. Ralyté, A. Sutcliffe, N. A. M. Maiden, M. Jarke, P. Haumer, K. Pohl, [Sawyer y Kontoya 1999] P. Sawyer y G. Kontoya. SWEBOK: Software Requirements Engineering Knowledge Area Description. Informe Técnico Versión 0.5, SWEBOK Project, 1999. http://www.swebok.org.

[15] MAP. Metodología de Planificación y Desarrollo de Sistemas de Información. MÉTRICA Versión 2.1. Tecnos/Ministerio para las Administraciones Públicas, 1995.

[16] S. Lilly. How to Avoid Use-Case Pitfalls. Software Delopment, Enero 2000. http://www.sdmagazine.com/breakrm/features/s0001f4.shtml

- Tarea 6: Identificar los requisitos no funcionales

En esta tarea se deben identificar o revisar los requisitos no funcionales del sistema, normalmente relacionados con aspectos técnicos o legales: comunicaciones, interfaces con otros sistemas, fiabilidad, entorno de desarrollo, portabilidad, etc.

Como resultado, se entregan los requisitos no funcionales expresados en la plantilla que se describe más adelante.

3.7.3. Análisis de requisitos. El término análisis de requisitos se ha utilizado durante bastante tiempo para hacer referencia al conjunto global de las actividades relacionadas con los requisitos en el desarrollo de software, al suponerse que los suministraba directamente el cliente, por lo que no era labor del equipo de desarrollo la elicitación de dichos requisitos ni había necesidad de una validación por parte del cliente, ya que era él mismo el que los había producido.

En este capítulo, el análisis de requisitos se considera como una actividad dentro de la fase de ingeniería de requisitos cuyo objetivo principal es descubrir conflictos en los requisitos–C elicitados previamente, profundizando en el conocimiento del problema mediante la construcción de modelos conceptuales (requisitos–D17)que sean más fácilmente entendibles por los desarrolladores y que puedan servir de base en la fase de diseño, avanzando de esta forma en las dimensiones de compleción y formalidad del proceso18.

Las técnicas de modelado conceptual son la principal herramienta del análisis de requisitos. Tradicionalmente se han considerado las técnicas estructuradas y las orientadas a objetos como dos formas incompatibles de modelar sistemas de información, pero se puede mostrar que las diferencias no son tantas como se supone.

---

[17] H. D. Rombach. Software Specifications: A Framework. Curriculum Module SEI–CM–11–2.1, Software Engineering Institute, Carnegie Mellon University, 1990. http://www.sei.cmu.edu , aunque aparece en [Dorfman y Thayer 1990].

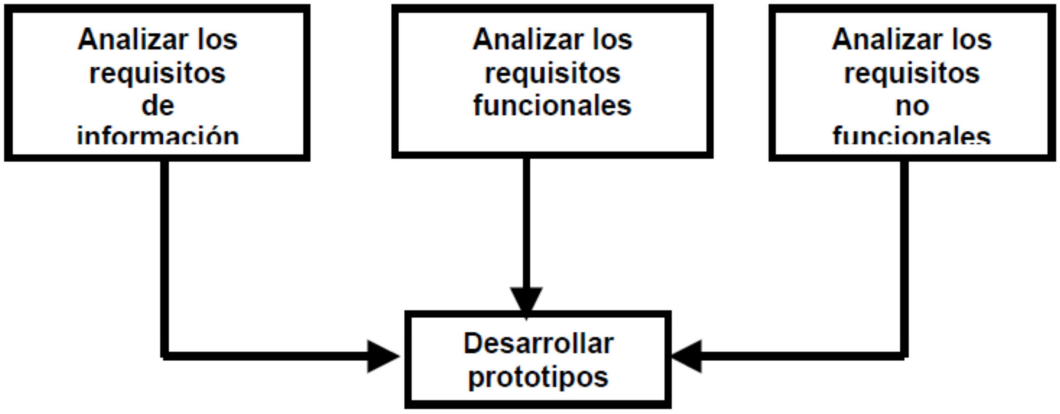
J. W. Brackett. Software Requirements. Curriculum Module SEI–CM–19–1.2, Software Engineering Institute, Carnegie Mellon University, 1990. http://www.sei.cmu.edu .

[18] K. Pohl. The Three Dimensions of Requirements Engineering: A Framework and its Application.Information Systems, 3(19), Junio 1994.
K. Pohl. Requirements Engineering: An Overview. Encyclopedia of Computer Science and Technology, 36, 1997. http://sunsite.informatik.rwth-aachen.de/CREWS /reports96.htm .

- Propuesta metodológica para el análisis de requisitos

La propuesta metodológica para el análisis de requisitos que plantea Amador Durán, como se observa en la figura 20, propone una serie de tareas a realizar y productos a obtener durante la realización de la actividad de análisis de requisitos del modelo de procesos de ingeniería de requisitos descrito la sección 3.8. De este documento .

Figura 20: Propuesta metodológica de análisis de requisitos de Amador Durán



Se contemplan cuatro tareas, donde se propone un orden de realización en el que las tareas de analizar los tres tipos de requisitos–C se realizarían en paralelo. En las siguientes secciones se describen cada una de ellas.

- Tarea 1: Analizar los requisitos de almacenamiento de información

El objetivo principal de esta tarea, al igual que el de la siguiente, es descubrir conflictos en los requisitos–C lo cual permite seguir profundizando en el conocimiento del problema y estableciendo las bases para un futuro diseño del sistema.

La forma habitual de alcanzar estos objetivos es mediante la construcción de modelos abstractos. En esta propuesta se recomienda la utilización de técnicas de modelado orientadas a objetos, aunque también podrían utilizarse técnicas estructuradas, ya que no existen tantas diferencias como se suponen y que pueden considerarse casi equivalentes19.

Los productos resultantes de la realización de esta tarea son el modelo estático del sistema, compuesto por los diagramas de tipos, la descripción textual de los

[19] R. J. Wieringa. A Survey of Structured and Object–Oriented Software Specification Methods and Techniques. ACM Computing Surveys, 30(4), Diciembre 1998.

elementos que lo integran, y los posibles conflictos que se detecten al construir el modelo.

- Tarea 2: Analizar los requisitos funcionales

Esta tarea es similar a la anterior, con la diferencia de que se centra en los requisitos–C funcionales, expresados mediante casos de uso, en lugar de hacerlo en los requisitos de almacenamiento de información.

Para analizar los requisitos funcionales lo habitual es construir modelos funcionales y, si se considera oportuno, modelos dinámicos.

Por lo tanto, los productos resultantes de la realización de esta tarea son los modelos funcional y dinámico y los conflictos que se hayan detectado al construir ambos modelos.

- Tarea 3: Analizar los requisitos no funcionales

Esta tarea tiene también como objetivo descubrir conflictos en los requisitos-C, en este caso en los requisitos no funcionales.

La naturaleza heterogénea de este tipo de requisitos hace que su análisis sea necesario realizarlo mediante una lectura detenida de su contenido, y combinando esta lectura con la experiencia, detectar posibles conflictos como la imposibilidad técnica de la implementación de ciertos requisitos, la necesidad de optar por unas características u otras (por ejemplo flexibilidad frente a eficiencia), etc.

Normalmente, estos requisitos se tendrán en consideración durante el diseño de la arquitectura del sistema20, lo que probablemente provocará iteraciones entre el resto del desarrollo y la fase de ingeniería de requisitos.

Los productos resultantes de esta tarea son, por lo tanto, aquellos conflictos que se hayan detectado al realizar el análisis de los requisitos no funcionales.

3.7.4. Validación de requisitos. Dentro del entorno metodológico propuesto, la validación de requisitos se considera como la actividad de la ingeniería de requisitos en la que clientes y usuarios, junto con la ayuda de los ingenieros de requisitos, revisan los productos obtenidos durante las actividades anteriores para

[20] L. Chung, D. Gross, y E. Yu. Architectural Desing to Meet Stakeholder Requirements. En Proceedings of the First Working IFIP Conference on Software Architecture (WICSA1), San Antonio, (Texas, USA), 1999. A. Ruiz, R. Corchuelo, A. Durán, O. Martín, y J. A. Pérez. Prototipado Arquitectónico de Sistemas Abiertos Distribuidos. En Actas de las V Jornadas de Trabajo Menhir, Granada, 2000.

confirmar que realmente reflejan sus necesidades y que definen el producto deseado.

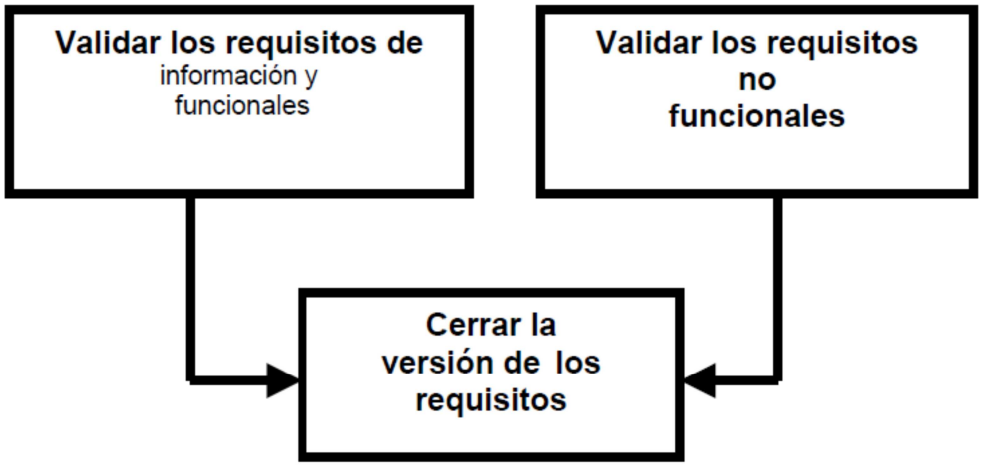
La validación de requisitos es otra de las actividades de la ingeniería de requisitos, que junto con la elicitación, han recibido tradicionalmente poca atención. La razón, al igual que en el caso de la elicitación, es que se ha trabajado bajo el supuesto que los requisitos los proporcionan directamente los clientes, con lo que, implícitamente, se suponen validados.

La ingeniería de requisitos actual supone como una necesidad la participación de los ingenieros de requisitos tanto en las actividades de elicitación, para ayudar a clientes y usuarios a encontrar y describir sus necesidades, como en las actividades de validación, para ayudar a clientes y usuarios a comprobar que los requisitos elicitados y analizados resultantes del proceso describen realmente todas sus necesidades correctamente.

- Propuesta metodológica para la validación de requisitos

La propuesta metodológica para la validación de requisitos, propone una serie de tareas a realizar mostradas en el siguiente gráfico

Figura 21: Propuesta metodológica de validación de requisitos de Amador Durán



En la propuesta se contemplan tres tareas, cuyo orden de realización es orientativo en el que las dos tareas en las que se validan los requisitos–C se realizarían en paralelo. En las siguientes secciones se describen cada una de ellas.

- Tarea 1: Validar los requisitos de almacenamiento de información y funcionales

El objetivo de esta tarea es validar los requisitos de almacenamiento de información y funcionales, es decir, asegurarse de que representan realmente las necesidades de clientes y usuarios.

Se han agrupado los dos tipos de requisitos en una única tarea porque, su validación mediante walkthroughs asistidos con prototipos de interfaz de usuario permite una validación conjunta de forma sencilla guiada por los requisitos funcionales descritos como casos de uso, teniendo siempre en cuenta las relaciones de rastreabilidad establecidas entre los requisitos de almacenamiento de información y los requisitos funcionales.

Los productos resultantes de esta tarea son los requisitos de almacenamiento de información, los requisitos funcionales y el prototipo validados total o parcialmente. En el caso de que se descubran conflictos durante la validación, éstos serían también productos de esta tarea y deberían resolverse en nuevas sesiones de elicitación/negociación.

- Tarea 2: Validar los requisitos no funcionales

El objetivo de esta tarea es validar los requisitos no funcionales. Sus productos, de forma similar a la tarea anterior, serían los requisitos no funcionales validados total o parcialmente, y los posibles conflictos que pudieran aparecer.

A diferencia de la tarea anterior, la única técnica que parece aplicable para realizar esta tarea es una revisión por parte de los clientes y usuarios, ayudados por los ingenieros de requisitos para aclarar las posibles dudas que surjan durante la revisión.

- Tarea 3: Cerrar la versión de los requisitos

Si no han aparecido nuevos conflictos durante el proceso de validación, se debe llegar a un acuerdo entre clientes y desarrolladores para cerrar la versión actual de los requisitos, siempre teniendo en cuenta que representa el conocimiento actual de los mismos y que, probablemente, sufrirá cambios en el futuro21.
El producto de esta actividad es, por lo tanto, una versión cerrada de los requisitos y del prototipo.

[21] J. B. Warmer y A. G. Kleppe. The Object Constraint Language: Precise Modeling with UML. Addison–Wesley, 1999. [Wiegers 1999] K. Wiegers. Customer Rights and Responsabilities. Software Development, Diciembre 1999. Disponible en http://www.sdmagazine.com/breakrm/features/s9912f2. shtml .

3.7.5. Elementos de representación. Amador Durán en su modelo brinda elementos de representación, que resultan de gran utilidad para lograr una definición más clara y precisa de los conceptos trabajados durante el proceso de Ingeniería de Requisitos. A continuación se describen algunos de ellos.

- Plantillas y patrones lingüísticos para elicitación de requisitos

Las plantillas que se presentan a continuación están elaboradas para que se puedan utilizar tanto en las reuniones de elicitación como para registrar y gestionar los requisitos-C. Se incluyen algunas frases estándar habituales en elicitación de requisitos y que se han parametrizado. Para cada una de las plantillas se hace una breve descripción

- Plantilla para los objetivos del sistema

Los objetivos del sistema se consideran como objetivos de alto nivel, de tal forma que los requisitos propiamente dichos son la forma de alcanzar los objetivos (Figura 22).

El significado de los campos es el siguiente:

- Identificador y nombre descriptivo: se debe identificar cada objetivo por un código único y un nombre que lo describa.

- Versión: permite gestionar diferentes versiones. Contiene el número y la fecha de la versión actual.

- Autores, Fuentes: se especifica el nombre y la organización de los autores(normalmente desarrolladores) y de las fuentes(clientes y usuarios), de la versión actual del objetivo. De este modo, la rastreabilidad puede llegar hasta las personas que propusieron la necesidad del requisito.

- Descripción: se tiene un patrón lingüístico que se debe completar con la descripción del objetivo.

- Subobjetivos: se pueden indicar los subobjetivos que dependen del objetivo descrito. Si el sistema es muy complejo, se puede establecer una jerarquía de objetivos antes de identificar los requisitos. Si no se requiere, se deja en blanco.

Figura 22: Plantillas de objetivos del sistema del modelo de Amador Duran

OBJ-<id>	<nombre descriptivo>
Versión	<# de la versión actual>(<fecha de la versión actual>)
Autores	• <autor de la versión actual>(<organización del autor>) • ……
Fuentes	• <fuente de la versión actual>(<organización de la fuente>) • ……
Descripción	El sistema deberá <objetivo a cumplir por el sistema>
Subobjetivos	• OBJ – x <nombre del subobjetivo> • ……
Importancia	<importancia del objetivo>
Urgencia	<urgencia del objetivo>
Estado	<estado del objetivo>
Estabilidad	<estabilidad del objetivo>
Comentarios	<comentarios adicionales sobre el objetivo>

- Importancia: se indica la importancia del cumplimiento del objetivo para los clientes y usuarios. Se puede asignar un valor numérico o alguna expresión enumerada como “vital”, “importante” o “quedaría bien”. Si no se le ha asignado importancia todavía, se puede indicar que está por determinar (PD).

- Urgencia: indica la urgencia en el cumplimiento del objetivo para los clientes y usuarios si hay un desarrollo incremental. Nuevamente, se puede dar un valor numérico o una expresión numerada como inmediatamente, hay presión, puede esperar o PD.

Estado: indica el estado del objetivo desde el punto de vista de su desarrollo. Puede estar en construcción, pendiente de negociación, pendiente de validación o simplemente, validado.

Estabilidad: se da una estimación de la probabilidad de que pueda tener cambios en el futuro. Se puede expresar mediante un valor numérico o una expresión numerada como alta, media, baja o PD.

Comentarios: cualquier información sobre el objetivo que no se pueda dar en los campos anteriores.

- Plantilla para Requisitos Funcionales

La plantilla presentada (Figura 23), describe casos de uso y ayuda a los clientes y usuarios a responder a la pregunta “¿Qué debe hacer el sistema con la información almacenada para alcanzar los objetivos del negocio?”

Los campos allí contenidos tienen el siguiente significado:

- Identificador y nombre descriptivo: cada requisito se debe identificar por un código único y un nombre descriptivo.

- Versión, autores, fuentes: igual significado al de la plantilla de objetivos pero con referencia al requisito.

- Objetivos asociados: debe contener una lista de objetivos asociados a los cuales está asociado el requisito. Permite conocer qué requisitos harán que el sistema a desarrollar alcance los objetivos propuestos y justifican la existencia o propósito del requisito.

- Requisitos asociados: se indican todos aquellos requisitos que están asociados con el requisito que se describe. Esto implica tener una rastreabilidad horizontal.

- Descripción: contiene un patrón lingüístico que debe completarse de forma distinta en función de que el caso de uso sea abstracto o concreto. Si es abstracto, deben indicarse los casos de uso en los que se debe realizar (incluido o extendido). Si es concreto, se debe indicar el evento que provoca su activación.

- Precondición: se expresan en lenguaje natural las condiciones necesarias para que se pueda realizar el caso de uso.

- Secuencia normal: contiene la secuencia normal de interacciones del caso de uso.

- Poscondición: se expresan en lenguaje natural las condiciones que se deben cumplir después de la terminación normal del caso de uso.

- Excepciones: se especifica el comportamiento del sistema en el caso de que se produzca alguna situación excepcional durante la realización de un paso determinado.

- Rendimiento: se puede especificar el tiempo máximo para cada paso.

- Frecuencia esperada: se indica la frecuencia esperada de realización del caso de uso. Se conoce que no es un requisito, pero es una información importante para los desarrolladores.

Figura 23: Plantillas de Requisitos Funcionales del Sistema del Modelo de Amador Duran

RF-<id>	<nombre descriptivo>
Versión	<# de la versión actual>(<fecha de la versión actual>)
Autores	• <autor de la versión actual>(<organización del autor>) • …….
Fuentes	• <fuente de la versión actual>(<organización de la fuente>) • …….
Objetivos asociados	• OBJ-x<nombre del objetivo> • …….
Requisitos asociados	• Rx-y<nombre del requisito> • ……
Descripción	El sistema deberá comportarse tal como se describe en el siguiente caso de uso {durante la realización de los casos de uso <lista de casos de uso>, cuando <evento de activación>}
Precondición	<precondición del caso de uso>
Secuencia normal	Paso Acción p1 {El actor <actor>, El sistema} <acción/es realizada/s por actor/sistema> p2 Se realiza el caso de uso <caso de uso (RF-x)> p3 Si <condición>,{el actor<actor>, el sistema} <acción/es realizadas por actor/sistema> p4 Si <condición>, se realiza el caso de uso <caso de uso (RF-x)> ....
Postcondición	<postcondición del caso de uso>
Excepciones	Paso Acción p1 Si <condición de excepción>, {el actor <actor>, el sistema} <acción/es realizada/s por actor/sistema>, a continuación este caso de uso {continúa, termina} p2 Si <condición de excepción>, se realiza el caso de uso <caso de uso (RF-x)>, a continuación este caso de uso {continúa, termina} ....
Rendimiento	Paso Cota de tiempo - Q m <unidad de tiempo>..... .....
Frecuencia esperada	<# de veces> veces / <unidad de tiempo>

- Plantilla para Requisitos No Funcionales

En la plantilla que se presenta (Figura 24), el único campo específico de esta plantilla es la descripción. Se utiliza un patrón lingüístico que debe completarse con la capacidad que deberá presentar el sistema.

Los demás campos, tienen el mismo significado de la plantilla anterior.

Figura 24: Plantillas de requisitos no funcionales del sistema del modelo

RNF-<id>	<nombre descriptivo>
Versión	<# de la versión actual>(<fecha de la versión actual>)
Autores	• <autor de la versión actual>(<organización del autor>) • ……
Fuentes	• <fuente de la versión actual>(<organización de la fuente>) • ……
Objetivos asociados	• OBJ-x<nombre del objetivo> • ……
Requisitos asociados	• Rx-y<nombre del requisito> • ……
Descripción	El sistema deberá <capacidad del sistema>
Datos específicos	• <datos específicos sobre el concepto relevante> • ……
Intervalo temporal	{ pasado y presente, sólo presente}
Importancia	<importancia del requisito>
Urgencia	<urgencia del requisito>
Estado	<estado del requisito>
Estabilidad	<estabilidad del requisito>
Comentarios	<comentarios adicionales sobre el requisito>

- Matriz de Rastreabilidad

Esta matriz permite visualizar dos aspectos fundamentales. El primero, el dar una visión rápida de las relaciones de dependencias entre objetivos y requisitos, y el segundo, permitir realizar una comprobación rápida de si todos los objetivos tienen un requisito asociado y si todos los requisitos están justificados por un determinado objetivo. Vale la pena mencionar, la dificultad en asociar objetivos a requisitos no funcionales.

Figura 25: Matriz de rastreabilidad del modelo de Amador Duran

	OBJ - 001	OBJ - 002	…	…	OBJ -n
FRQ – 001	X				
FRQ – 002	X				X
…	X	X			
NFR – 001			X		
NFR – 002				X	
…		X			

3.8 ESKEMA: ESQUEMA PARA LA GESTIÓN DE CATÁLOGOS DE PATRONESDE ANÁLISIS UTILIZANDO EL ENFOQUE ASPECTUAL

ESKEMA tiene como objetivo proveer mecanismos que permitan introducir de manera natural en las tareas del desarrollador, la reutilización como práctica sistémica. ESKEMA facilita a las empresas la incorporación de la reutilización en su proceso de desarrollo de software, permitiendo estandarizar las tareas de reutilización (identificación, definición, construcción, almacenamiento, etiquetado, documentación, búsqueda y gestión de activos de software reutilizables) [1].

Para el cumplimiento de este objetivo, ESKEMA provee el catálogo de patrones de análisis representados mediante MORENA, con una estructura que facilita los mecanismos de búsqueda, recuperación e instanciación, además de un componente gestor de experiencias de uso de los patrones de análisis.

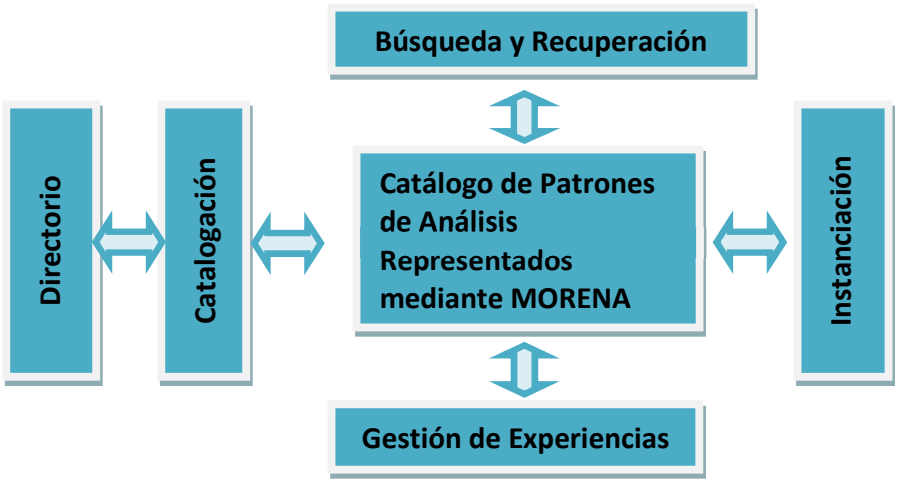
3.8.1 Modelo de ESKEMA. El modelo de ESKEMA se basa en la reutilización de activos de software integrados en un repositorio. En ESKEMA, los activos reutilizables son los patrones de análisis y el repositorio recibe el nombre de catálogo.

Durante la ingeniería de dominio, se crean los patrones de análisis como soluciones a problemas recurrentes en diferentes contextos de un mismo dominio o de varios dominios.

Durante la ingeniería de aplicación se instancian los patrones a las soluciones específicas. ESKEMA integra los mecanismos de gestión del catálogo: catalogación, búsqueda, recuperación e instanciación.

ESKEMA presenta los componentes mostrados en la Figura 26.

Figura 26. Componentes de ESKEMA



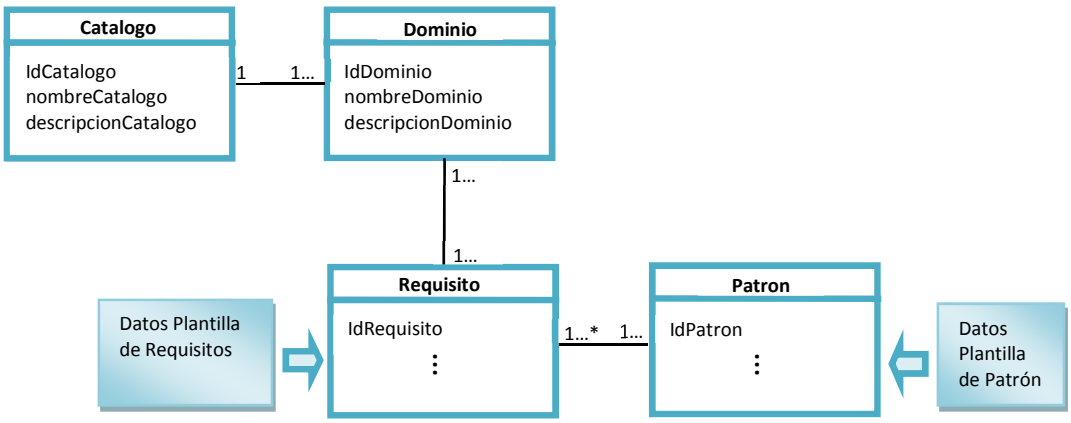
- Catálogo de Patrones de Análisis representados mediante MORENA. El catálogo es el componente que permite la gestión de persistencia de los patrones de análisis. El catálogo provee una estructura que permite ejecutar los mecanismos de gestión de los patrones de análisis (catalogación, búsqueda, recuperación, instanciación y retroalimentación).

El catálogo de patrones de análisis en ESKEMA está divido en dominios. A cada dominio se asocian un conjunto de patrones de análisis y un conjunto de requisitos de dominio. Cada patrón de análisis y cada requisito de dominio pueden asociarse a un dominio o a varios dominios.

La asociación del patrón de análisis con los requisitos de dominio que resuelve y los dominios donde se aplica, hacen parte de la definición de los patrones de análisis en ESKEMA.

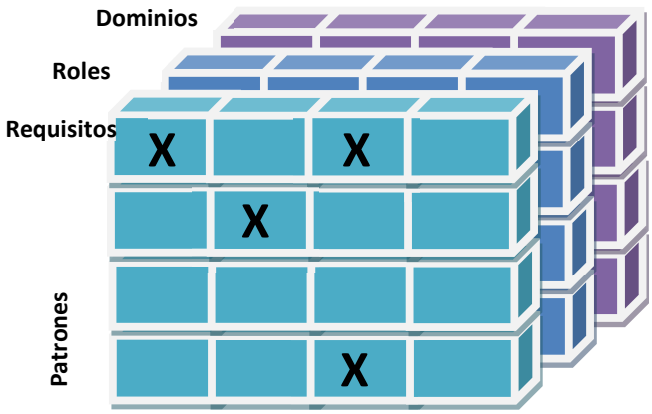
El catálogo de patrones de análisis de ESKEMA hereda los métodos y la arquitectura de la herramienta CASE a la cual se integra.

Figura 27. Estructura del catálogo de patrones de análisis



- Directorio del Catálogo de Patrones de Análisis. El directorio es el componente de ESKEMA que define el sistema de indexación, es decir, los criterios de organización lógica de los patrones de análisis en el catálogo. Este componente se fundamenta en la estructura del catálogo y en los elementos de definición del patrón de análisis. El directorio es consumido por los mecanismos de búsqueda y recuperación. Unificar los índices del directorio con los criterios de búsqueda y recuperación permite optimizar la gestión del catálogo de patrones de análisis.

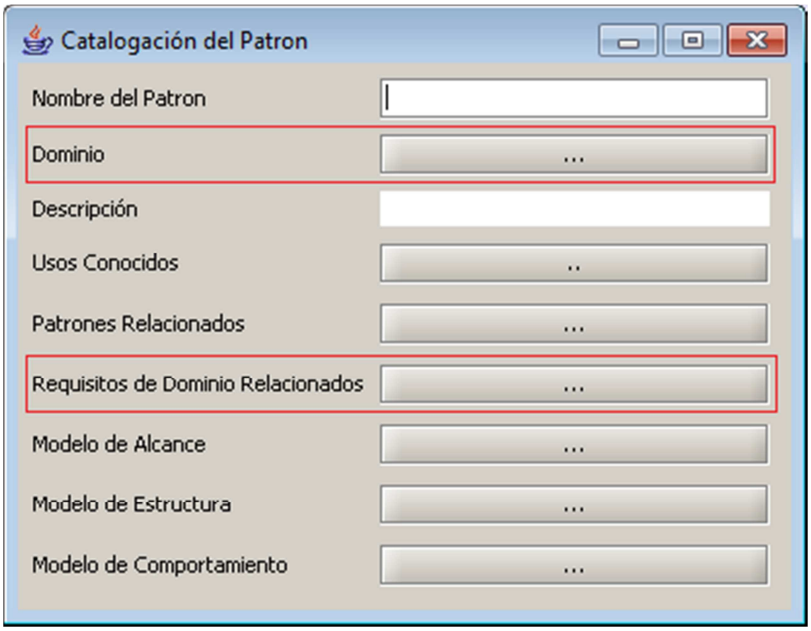
Figura 28. Directorio del catálogo de patrones de análisis



- Mecanismo de Catalogación. El mecanismo de catalogación es la estrategia que provee ESKEMA para integrar un patrón de análisis al catálogo, asignando los valores de indexación definidos en el directorio.

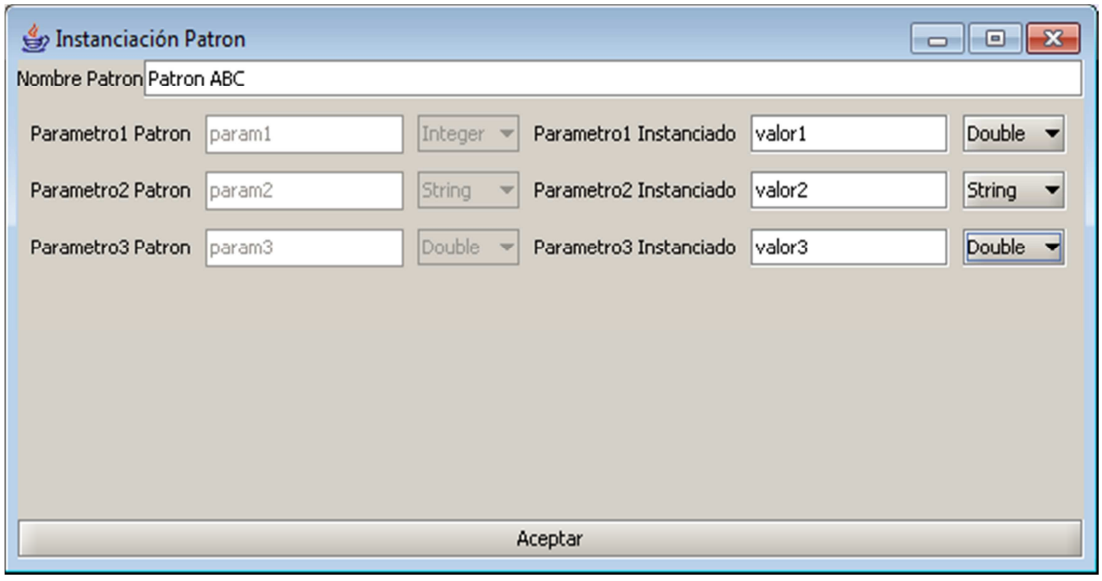
La catalogación se desarrolla durante la definición del patrón indicando el requisito de dominio o los requisitos de dominio que resuelve, el dominio o los dominios en los cuales se aplica y los roles participante. La catalogación de un patrón de análisis puede ser actualizada cuando el desarrollador indica una nueva asociación del patrón con otros requisitos de dominio o la participación de otros roles. En cualquier caso el catálogo se actualiza y ESKEMA provee la vista requerida para el uso de los patrones de análisis en las nuevas condiciones.

Figura 29: Mecanismo de catalogación



- Mecanismo de Instanciación. En ESKEMA el mecanismo de instanciación se desarrolla en dos momentos. Durante el momento uno, el patrón se instancia asignando valores a los parámetros del signature del Template. El momento uno de instanciación termina cuando el desarrollador confirma la operación y los valores asignados a los parámetros del template se integran a los modelos. En este punto se elimina el elemento signature del template y este se convierte en un paquete que contiene los modelos que ahora son la solución del problema específico. Lo modelos resultantes del momento uno pueden ser trabajados para terminar de configurar la solución específica. Esta última tarea hace referencia al momento dos de instanciación.

Figura 30: Mecanismo de instanciación



3.8.2 Patrones de análisis en ESKEMA.

- Descripción del Patrón de Análisis. Existen varios formatos para la descripción de patrones de software. ESKEMA adopta algunos de los elementos del formato de descripción de patrones conocido como formato del GoF, descrito en Gamma [8]. Los elementos que describen un patrón de análisis son: nombre, dominio, descripción, solución, usos conocidos y patrones relacionados.

- Representación del patrón de análisis. los modelos de alcance, estructural y de comportamiento que definen el patrón de análisis son representados por diagrama de colaboración, diagrama de clases y diagrama de secuencia respectivamente. los modelos se encapsulan en un constructor de la especificación uml 2.0 llamado template [9].

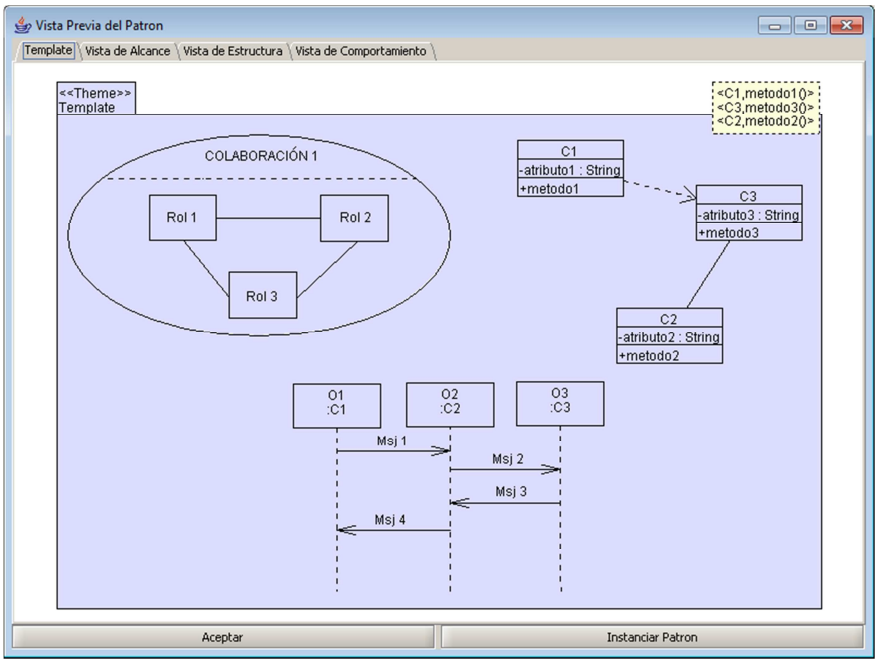
- Representación integrada. el patrón de análisis es representado por un template. el template es un constructor descrito en la superestructura de uml 2.0, como un elemento de modelado que es parametrizado por otros elementos del modelo y pueden ser de tipo clasificador, paquete u operación. para especificar su parametrización el template posee una firma o signature. un template puede ser utilizado para generar otro modelo de elementos utilizando relaciones templatebinding. en esta relación los parámetros del signature se especifican como parámetros formales que serán sustituidos por parámetros actuales [9].

Theme/UML es una aproximación de desarrollo de software orientado por aspectos que utiliza el Template para representar el Theme Aspectual. En Theme/UML, el Template es tipo paquete, pues se utiliza como contenedor de los

diagramas que representan el Theme aspectual. Los parámetros del signature reciben los valores para instanciar los modelos del theme aspectual. La relación Template Binding es el canal por el cual los valores llegan al theme aspectual22.

Se retoma el concepto de patrón de análisis de la aproximación Theme/UML. En ESKEMA, el Template es de tipo paquete pues se utiliza como contenedor de los diagramas que representan los modelos que definen el patrón de análisis. El Signature permite instanciar el patrón. Los parámetros del Signature reciben los valores del problema específico que instancian las variables del patrón de análisis en cada uno de los modelos. La relación TemplateBinding es el canal por medio del cual se envían al patrón de análisis los valores que lo instancian y que son recibidos en los parámetros de la Signature.

Figura 31: Representación integrada del patrón de análisis



[22] Baniassad, E., Clarke, S.: Aspect-Oriented Analysis and Design: The Theme Approach. Addison-Wesley (2005)

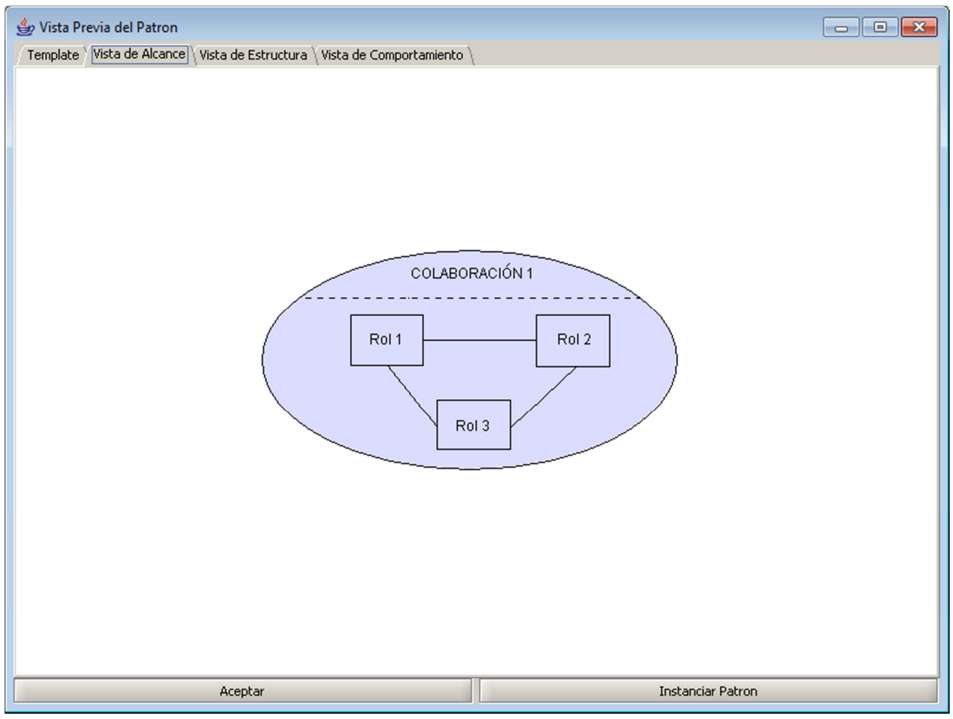
- Modelo de Alcance

El modelo de alcance permite definir como el patrón de análisis interactúa con su contexto y en especial con los usuarios de acuerdo con los roles que asumen.

- Representación del Modelo de Alcance

En ESKEMA, el modelo de alcance es representado por una colaboración.

Figura 32: Representación del modelo de alcance



La colaboración representa de manera adecuada como los roles asociados al patrón de análisis se relacionan entre sí. Las relaciones entre roles, permite generar el comportamiento del patrón de análisis en el contexto.

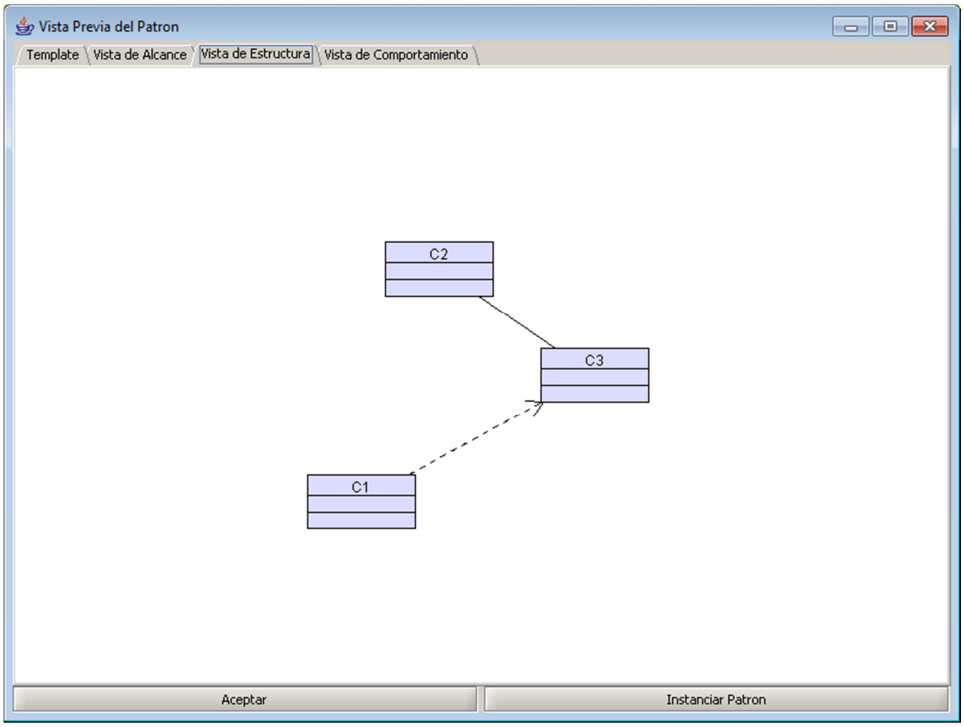
- Modelo estructural

El modelo estructural permite representar como se compone y relaciona la estructura del patrón de análisis.

- Representación del modelo estructural

En ESKEMA, el modelo estructural es representado por un diagrama de clases.

Figura 33: Representación del modelo estructural



El diagrama de clases permite representar la estructura conceptual del patrón de análisis indicando clases, atributos y relaciones. Los elementos del diagrama de clases en el patrón de análisis, reciben nombres genéricos que serán instanciados en la acción instanciar el patrón.

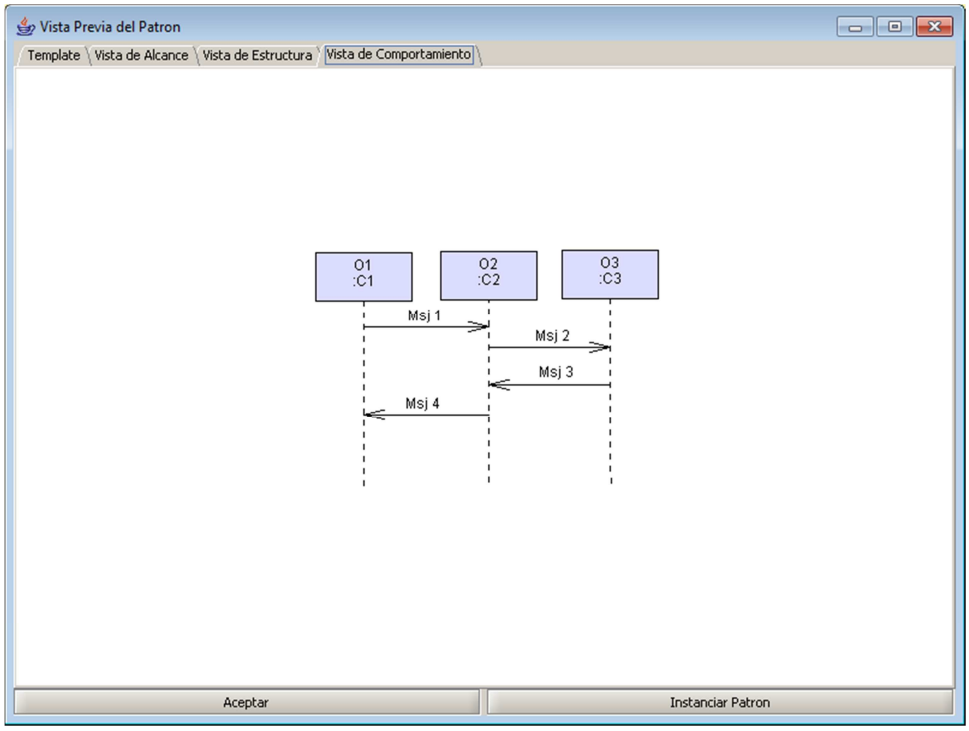
- Modelo de comportamiento

El modelo de comportamiento define la parte dinámica del patrón de análisis. En ESKEMA, describe los aspectos del patrón de análisis que cambian con el tiempo.

- Representación del modelo de comportamiento

En ESKEMA, el modelo de comportamiento se representa por un diagrama de secuencia.

Figura 34: Representación del modelo de comportamiento

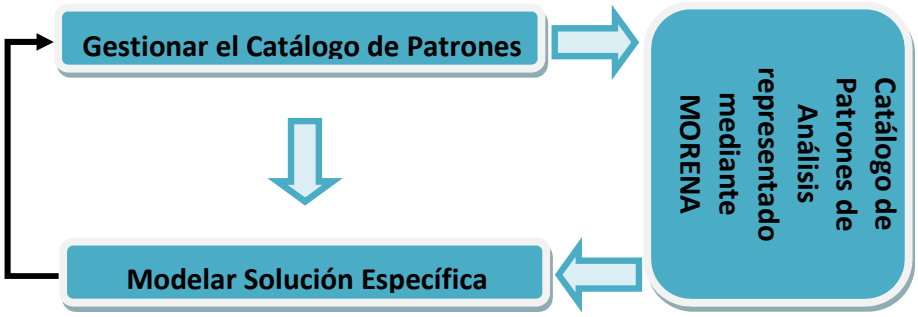


El diagrama de secuencia describe la interacción de los objetos del patrón de análisis a través del tiempo. Contiene detalles de implementación de la solución propuesta por el patrón, incluyendo los objetos y los mensajes pasados entre los objetos. Los elementos del diagrama de secuencia en el patrón de análisis, reciben nombres genéricos que serán instanciados durante el modelado de la solución específica.

3.8.9 Proceso de ESKEMA. En ESKEMA se define un proceso de desarrollo que dispone de actividades lógicamente organizadas para gestionar un desarrollo basado en patrones de análisis.

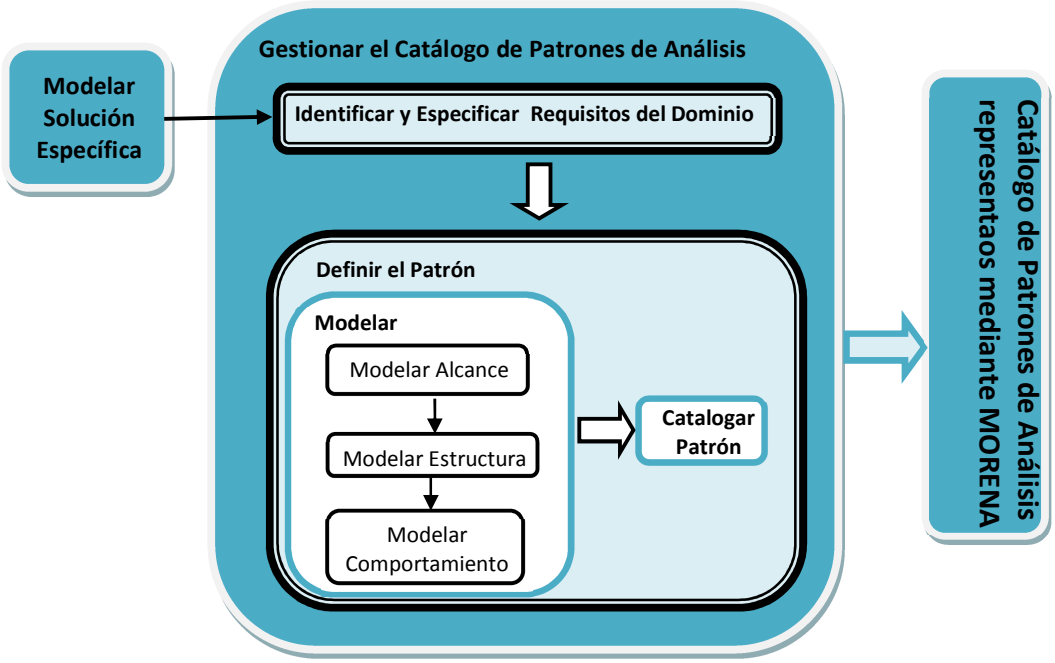
Se distinguen dos grandes actividades: El análisis del dominio para definir los patrones que representan comportamiento genérico y el modelado de solución específica a partir de estos patrones, como se observa en la Figura 35.

Figura 35. Vista general del proceso de desarrollo de ESKEMA



- Gestión del catálogo de patrones de análisis. El objetivo de esta actividad es definir el proceso que guiará la construcción del catálogo a partir de la identificación y especificación de requisitos del dominio y la retroalimentación del catálogo a partir de requisitos de una solución específica generalizada.

Figura 36. Gestionar el catálogo de patrones de análisis



- Identificar y especificar requisitos del dominio. Esta acción busca definir los requisitos que especifican el dominio y que son insumo para la definición del patrón de análisis. Para la construcción inicial del catálogo, los requisitos se

identifican a partir del análisis del dominio. Para la retroalimentación del catálogo, los requisitos se identifican a partir del análisis de una situación específica para la cual no se encontró solución en el catálogo, en este caso, el requisito debe ser generalizado para que se defina como requisito de dominio y sea insumo para la definición de un nuevo patrón.

Esta acción puede ser desarrollada aplicando aproximaciones de ingeniería de dominio como FODA, ODM, DARE o la de Martin Fowler [3]. La ingeniería de dominios reconoce dos procesos distintos: la ingeniería de dominio y la ingeniería de aplicación.

La ingeniería de dominio se centra en el desarrollo de elementos reutilizables que formarán la familia de productos, mientras que la ingeniería de aplicación se orienta hacia la construcción de productos individuales, pertenecientes al dominio, y que satisfacen un conjunto de requisitos y restricciones expresados por un usuario específico, reutilizando, adaptando e integrando los elementos reutilizables existentes y producidos en la ingeniería de dominio. En este punto ESKEMA identifica la ingeniería de dominios como parte del proceso.

- Definir el patrón. El patrón de análisis se define con base en los requisitos. Esta acción incluye modelar y catalogar el patrón.

Modelar el patrón es una tarea que busca generar los modelos que definen al patrón, desde tres vistas que se representan en UML: el modelado del alcance, representado por una colaboración [10]; el modelado estructural, representado por un diagrama de clases [8] y el modelado de comportamiento representado por un diagrama de secuencia [8].

Catalogar el patrón es la actividad que busca integrar el nuevo patrón al catálogo a partir de la instanciación de los elementos de descripción del patrón que son los criterios de organización del catálogo. La catalogación adecuada del patrón facilita su posterior búsqueda, recuperación e instanciación [1].

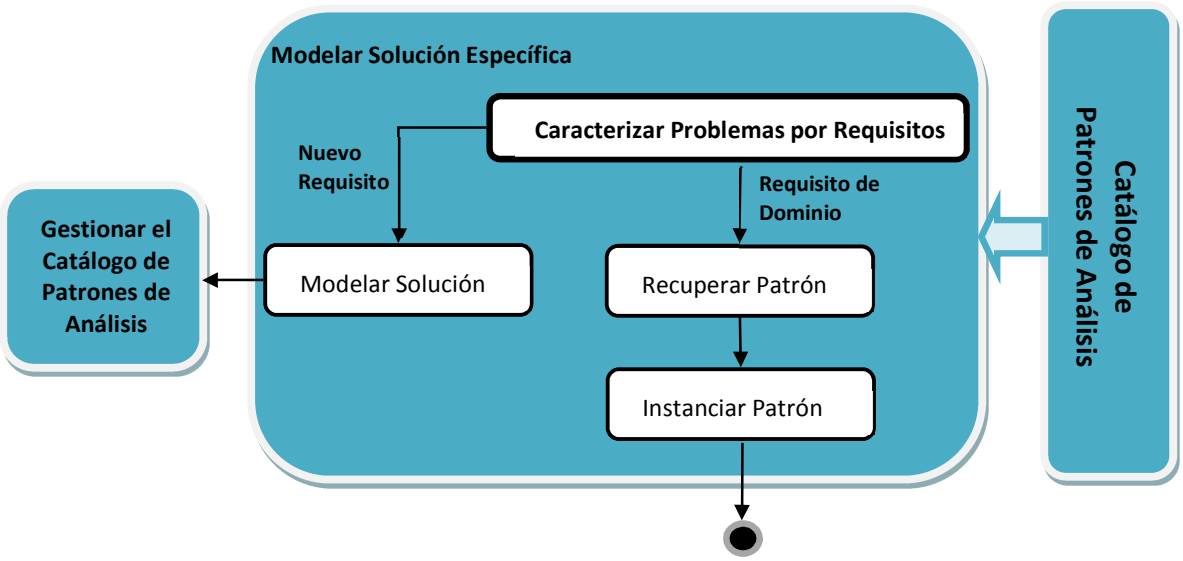
Actualmente existen varios modelos de catalogación que permiten clasificar y catalogar a los patrones, a efectos de facilitar su recuperación y posterior utilización. Así mismo, casi todas las herramientas de modelado incluyen funcionalidades para utilizar patrones en el modelado que permiten navegar por un catálogo de patrones y generar artefactos de software.

Esta actividad es un ciclo que termina una vez se han catalogado los patrones de análisis a partir de los requisitos que han sido identificados y definidos ya sea como producto del análisis del dominio, o de la generalización de una solución específica. Como resultado de este trabajo, se especifican formalmente los requisitos, se modelan los patrones de análisis y se catalogan.

- Modelar la Solución Específica

El objetivo de esta actividad es apoyar el modelado de una solución específica a partir de la correlación entre requisitos del dominio y requisitos del problema, para determinar si el catálogo de patrones provee una solución genérica que podrá ser instanciada, o si por el contrario la solución específica debe ser modelada desde cero. En este último caso, requisito y modelo se evalúan para determinar si generan un nuevo patrón que retroalimenta el catálogo.

Figura 37. Modelar la solución especifica



- Caracterizar problemas por requisitos.

Esta acción busca identificar los requisitos que especifican el problema, aquellos que son comunes al dominio y aquellos que son particulares al problema.

El catálogo provee un conjunto de patrones de análisis cuya descripción integra los requisitos de dominio que resuelve. El desarrollador puede caracterizar el problema utilizando los requisitos del dominio y posteriormente recuperar e instanciar el patrón correspondiente. Es posible que el problema se defina también por requisitos que no provee el catálogo, en tal caso, el requisito debe ser definido en el contexto del problema.

- Recuperar el patrón.

Esta acción busca extraer del catálogo el patrón que resuelve un requisito de dominio y colocarlo a disposición del desarrollador para reutilizarlo.

El catálogo de patrones de análisis provee el mecanismo que permite recorrer la estructura del catálogo, seleccionar elementos que definen el patrón, encapsularlos en un objeto y colocarlo a disposición del desarrollador.

- Instanciar el patrón.

Esta acción busca especializar el patrón de análisis convirtiéndolo en la solución del problema específico.

Debido a que los Patrones de análisis han sido creados para ser utilizados en múltiples contextos, estos deben ser especializados para solucionar un problema específico. Una vez el patrón se encuentra en el área de trabajo del proyecto, es de caja blanca, es decir, el analista no tiene más restricciones para su modificación que las impuestas por la herramienta CASE. El patrón de análisis se representa por medio del Template de la especificación UML 2.0 que encapsula los modelos que definen el patrón.

- Modelar la solución.

Esta acción busca generar los modelos que responden a requisitos para los cuales el catálogo no provee un patrón solución. El requisito del problema y el modelo de la solución son candidatos para retroalimentar el catálogo.

El modelado de la solución específica se desarrolla utilizando las funcionalidades de la herramienta CASE. Este modelado se limita a dar respuesta al problema particular y no integra las vistas de alcance, estructura y comportamiento que si provee un patrón de análisis. En la actividad gestionar el catálogo se determina si el requisito del problema y el modelo de la solución se generalizan a nivel del dominio, en tal caso, se define el nuevo patrón que retroalimenta el catálogo.

Esta actividad es un ciclo que termina una vez se han modelado todos los requisitos que caracterizan el problema y han sido resueltos ya sea a partir de los patrones de análisis que provee el catálogo o de modelado de soluciones específicas. Como resultado de este trabajo, se modela el problema específico y se identifican requisitos y modelos candidatos para retroalimentar el catálogo.

3.9 TRIPODE: CATALOGO DE PATRONES ASPECTUALES DE ANÁLISIS PARA EL DOMINIO POS BASADO EN MORENA

TRIPODE es un catálogo de patrones aspectuales de análisis para el dominio POS basado en MORENA.

El objetivo principal de TRIPODE23 es validar el modelo de representación en el que se basa, para lo cual define unos patrones de análisis que tienen un enfoque aspectual a través del uso de templates parametrizables y que dan solución a requisitos genéricos del dominio POS.

En TRIPODE inicialmente se identifican unos requisitos de dominio POS, luego estos requisitos son descritos en una plantilla de requisitos basada en la plantilla de requisitos de Amador Durán, finalmente se hace la definición de unos patrones de análisis siguiendo el modelo de representación MORENA. Estos elementos son presentados a continuación.
Requisitos del dominio POS

A partir de los sistemas POS existentes en el mercado y sus funcionalidades, se identifican los siguientes requisitos para el dominio POS24:

[23] Cisneros Iván, Erazo Gloria, Peña Germán - Tripode: Un Catálogo De Patrones Aspectuales De Análisis Para El Dominio Pos Basado En Morena, Universidad De Nariño, 2010.

[24] Araújo, Tania. Bolaños, Yurani, Informe Técnico del Análisis de Requerimientos de un Sistema POS, Grupo de Investigación Galeras LIS, Universidad de Nariño, 2008.

3.9.1 Requisitos funcionales

Tabla 2: Requisitos funcionales del dominio POS

Requisito	Descripción
Manejo de Descuentos y promociones	Realizar la aplicación y el registro de los descuentos y las promociones que se apliquen en un centro de venta.
Registrar Venta por Cliente	Realizar el registro de los productos que compra cada cliente por venta, para futuros usos en promociones, descuentos o toma de decisiones.
Manejo Inventario	Realizar el envío de información para actualización de inventario.
Integración Sistemas de pago con tarjeta	Realizar la integración de los diferentes sistemas con los sistemas externos que permiten el pago por medio de tarjeta débito o crédito.
Conversión a moneda extranjera	Calcular la equivalencia del costo de un producto comercializado en moneda extranjera.
Realizar los movimientos de caja	Llevar un control sobre los flujos de entrada y salida de caja, registrando la forma de éstos flujos (cheques, efectivo).
Generar reportes de ventas	Generar un reporte (impreso, archivo, pantalla) de los movimientos básicos de ventas.
Integrar la información	Integración de la información de los diferentes puntos de venta de la empresa con la administración central en tiempo real, en una misma base de datos replicada.
Consulta de productos	Realizar la búsqueda de productos por sus diferentes características, desplegando toda la información relacionada al producto, de esta forma se puede ver la disponibilidad de productos o búsqueda específica de un producto.
Realizar arqueo de caja	Permitir un proceso en el que se totalice las ventas llevadas hasta ese momento y realizar un comparativo con el efectivo.
Reporte de la información totalizada	Realizar un reporte que contenga el total vendido, el total de costos, el total de impuestos y productos que se venden.
Realizar devoluciones	Realizar la respectiva devolución de un producto que ya fue vendido; enviando actualización a inventario, anulando factura y retornando pago.
Gestionar Factura	Crear y hacer persistencia de la factura, la cual deberá contener la información que se considere pertinente. Al finalizar el proceso el sistema deberá emitir la factura siguiendo los parámetros de ley.
Validar Forma de Pago	Validar las diferentes formas de pago y realizar las operaciones correspondientes.
Gestionar acciones	Permitir programar e implementar acciones, las cuales podrán ser completadas, canceladas o suspendidas por un periodo de tiempo.

- Requisitos no funcionales

Tabla 3: Requisitos no funcionales del dominio POS

Requisito	Descripción
Soportar varios idiomas	Funcionamiento multi-idioma, de tal manera que puedan añadirse nuevos idiomas fácilmente.
Interfaz grafica del usuario	El sistema debe ser gráfico, con uso mínimo del teclado para las funcionalidades que requieren atención al cliente.
Interfaz de fácil manejo	La interfaz debe ser de fácil manejo, para que así el sistema no requiera personal muy especializado para su uso.
Manejar un corto tiempo de respuesta	El sistema deberá entregar la información requerida y realizar las operaciones internas de manera ágil para ofrecer un buen servicio.
Soportar alto volumen de transacciones	Manejar un sistema gestor de base de datos que permita el soporte de un alto número de transacciones brindando integración y coherencia en el manejo de la información.
Manejo de información con altos niveles de seguridad	Brindar acceso a la información según el nivel de autoridad definido en la empresa (por medio de módulos de autorización incluidos en el sistema).
Realizar copias de seguridad de bases de datos	Realizar una copia de seguridad de los datos de manera periódica establecida por el usuario.

- Requisitos de información

Tabla 4: Requisitos de información del dominio POS

Requisito	Descripción
Almacenar los periodos especiales de venta	Almacenamiento de la información relacionada con los periodos de venta que tienen asociadas promociones o descuentos (Día de la madre, día del padre, navidad, etc.).
Almacenar los impuestos de venta	Almacenamiento de la información relacionada con los impuestos a las ventas de cada producto.
Almacenar información de ventas	Almacenamiento de información de productos vendidos, cantidades, días, fechas y horas.
Almacenar información de Cliente y productos comprados	Almacenamiento de información de la relación Cliente y productos comprados.
Transmitir Datos	Permitir la transmisión de datos entre los diferentes módulos del sistema.

Reglas de negocio

Tabla 5: Reglas de negocio del dominio POS

Requisito	Descripción
Calculo de impuestos en la venta	Calcular el valor que tiene los impuestos asociados a los productos registrados en la venta.
Calculo del "regreso/cambio" del pago de la venta	Calcular el valor del "regreso/cambio" generado por el pago de la venta.
Descuentos	Definir cuáles son los porcentajes de descuento que se van a dar en cada promoción o evento que así lo amerite.

3.9.2. Plantilla de descripción de requisitos. TRIPODE propone una plantilla para la especificación de los requisitos (Tabla 6), esta plantilla se basa en la propuesta de Amador Duran [11], la cual sufre ciertas modificaciones en sus campos. La descripción de los requisitos de dominio dentro de esta plantilla no se presenta en esta sección porque más adelante en el proceso de aplicación de este catálogo se presenta cada uno de los requisitos utilizados, descritos en esta plantilla

Tabla 6: Plantilla de requisitos TRIPODE

FRQ - <id>	<nombre descriptivo>
Versión	<no. de versión actual>
Autores	<autor de la versión actual>
Fuentes	<fuente de la versión actual>
Descripción	El sistema deberá <capacidad del sistema>
Dominios asociados	<Dominios que se encuentra el requisito>
Patrón solución	<Patrón que da solución al requisito>
Estado	<estado del requisito>
Estabilidad	<estabilidad del requisito>
Comentarios	<comentarios adicionales sobre el requisito>

El significado de los campos específicos de la plantilla es el siguiente:

Identificador y nombre descriptivo: cada requisito se debe identificar por un código único y un nombre descriptivo. Con objeto de conseguir una rápida identificación, los identificadores de los requisitos comienzan con FRQ.

- Versión: para poder gestionar distintas versiones, este campo contiene el número y la fecha de la versión actual del requisito.

- Autores, Fuentes: estos campos contienen el nombre y la organización de los autores (normalmente desarrolladores) y de las fuentes (clientes o usuarios), de la versión actual del requisito, de forma que la rastreabilidad pueda llegar hasta las personas que propusieron la necesidad del requisito

- Descripción: este campo contiene un patrón–L que debe completarse con la capacidad o funcionalidad que debe presentar el sistema a desarrollar.

- Dominios asociados: en el cual se identifican los distintos dominios en los que se encuentra el requisito.

- Patrón solución: en el cual se identifica el patrón que propone una solución al requisito.

- Estado: este campo indica el estado del requisito desde el punto de vista de su desarrollo. El requisito puede estar en construcción si se está elaborando, pendiente de negociación si tiene algún conflicto asociado pendiente de solución, pendiente de verificación si no tiene ningún conflicto pendiente y está a la espera de verificación o, pendiente de validación si ya ha sido verificado y está a la espera de validación o por último, puede estar validado si ya ha sido validado por clientes y usuarios.

- Estabilidad: este campo indica la estabilidad del requisito, es decir una estimación de la probabilidad de que pueda sufrir cambios en el futuro. Esta estabilidad puede indicarse mediante un valor numérico o mediante una expresión enumerada como alta, media o baja o PD en el caso de que aún no se haya determinado.

- Comentarios: cualquier otra información sobre el requisito que no encaje en los campos anteriores puede recogerse en este apartado.

3.9.3. Patrones de análisis propuestos por TRIPODE para el dominio POS

TRIPODE, define unos Patrones de análisis, los cuales dan solución a algunos de los requisitos del dominio POS.

- Los patrones definidos son:
- Patrón Factura.
- Patrón Transacción.
- Patrón Acción.

Posteriormente se presenta cada uno de los patrones con sus respectivas características y especificaciones, representados por cada uno de los modelos propuestos por MORENA (Modelos de Alcance, Estructural y de Comportamiento)

- Patrón Factura

El patrón de análisis Factura describe una propuesta de solución para la recolección, procesamiento y almacenamiento de la información pertinente a una operación de venta.

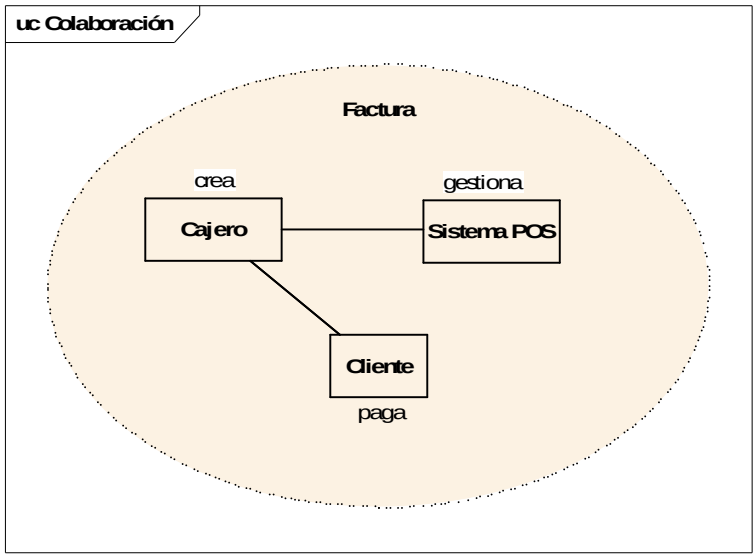
Usos conocidos:

- Supermercados y autoservicios - facturación de productos
- Clubes - facturación de productos y servicios
- Tiendas y negocios de venta minorista - facturación de productos
- Gastronomía (restaurantes, bares y negocios de comidas rápidas) - facturación de productos
- Modelo de Alcance

Tabla 7: Asignación de responsabilidades patrón factura

RESPONSABILIDAD	CLASE RESPONSABLE	ACTOR RESPONSABLE
Vincular ítem	Ítem	Cajero
Actualizar total factura	Factura	Sistema POS
Registrar pago	Pago	Cajero Cliente
Imprimir factura	Factura	Sistema POS

Figura 38: Colaboración patrón factura



Actores

- Cajero: representa el funcionario que opera la caja donde se desarrolla la operación de venta.

- Cliente: representa la persona u organización que adquiere los productos y/o servicios ofrecidos por la empresa.

- Sistema POS: representa el software que gestiona el servicio de venta de productos y/o servicios en la empresa.

Roles

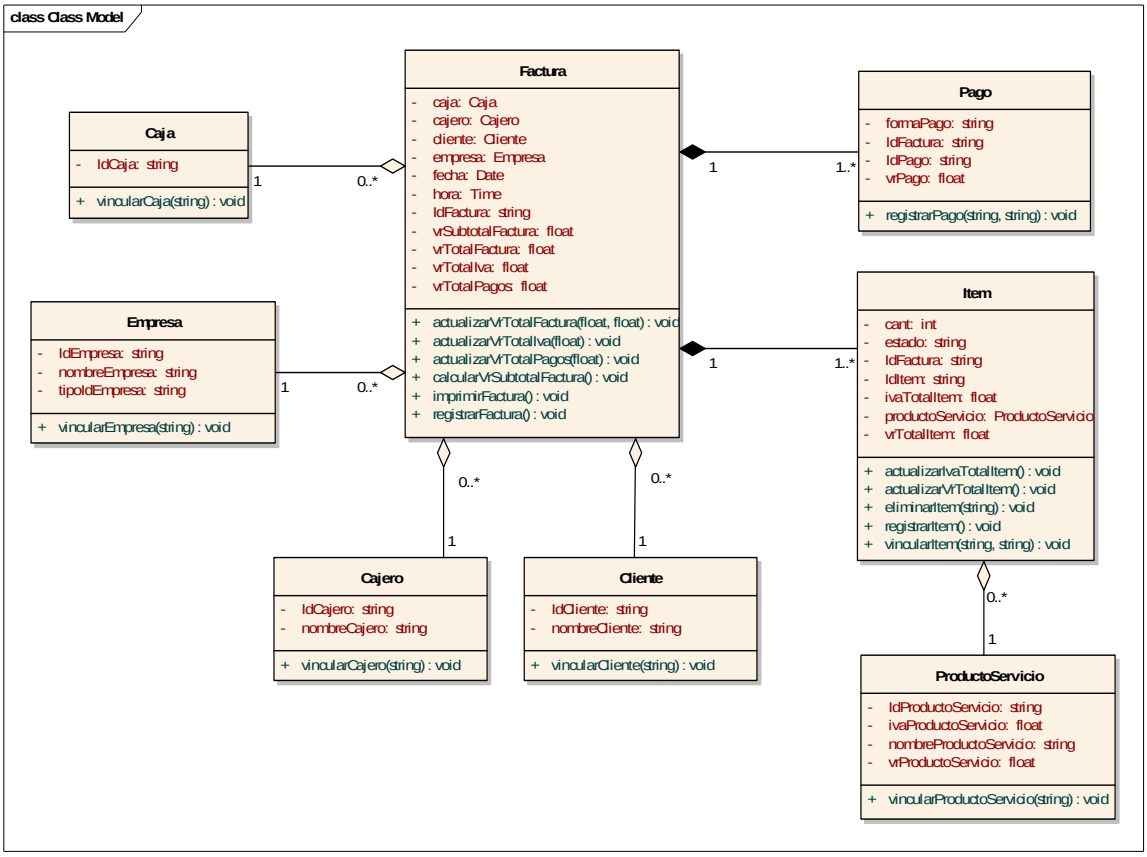
- Crear: representa la función de recolectar la información de la operación de venta y configurar la factura.

- Gestionar: representa la función de recibir, procesar, almacenar e imprimir la información referente a la operación de venta.

- Pagar: representa la función de realizar un desembolso por la adquisición de productos y/o servicios.

- Modelo Estructural

Figura 39: Diagrama de clases del patrón factura



Clase Factura: describe conceptualmente los documentos que contienen la información de una operación de venta de productos y/o servicios.

Atributos

- Caja: representa el punto de venta donde se desarrolla la operación.

- Cajero: representa el funcionario que opera la caja donde se desarrolla la operación.

- Cliente: representa la persona u organización que adquiere los productos y/o servicios ofrecidos por la empresa.

- Empresa: representa la persona u organización que ofrece los productos y/o servicios.

- fecha: representa la fecha de creación de la factura.

- hora: representa la hora de creación de la factura.

- IdFactura: representa el código de identificación de la factura.

- vrSubtotalFactura: representa la diferencia entre el valor total de la factura y el valor total del IVA de la factura.

- vrTotalFactura: representa la sumatoria de los valores totales de los ítems vendidos al cliente.

- vrTotalIva: representa la sumatoria del IVA total de los ítems vendidos al cliente.

- vrTotalPagos: representa la sumatoria de pagos realizados por el cliente en la operación de venta.

Métodos

- actualizarVrTotalFactura: representa la operación que permite acumular el valor total de los ítems vendidos al cliente.

- actualizarVrTotalIva: representa la operación que permite acumular el valor total del IVA de los ítems vendidos al cliente.

- actualizarVrTotalPagos: representa la operación que permite acumular el valor de los pagos realizados por el cliente en la operación.

- calcularVrSubtotalFactura: representa la operación que permite restar el valor total del IVA de la factura al valor total de la factura.

- imprimirFactura: representa la operación que permite imprimir la factura.

- registrarFactura: representa la operación que permite hacer persistencia de la factura.

Clase Caja: describe conceptualmente los puntos de venta donde se desarrollan las operaciones de venta.

Atributos

- IdCaja: representa el código de identificación de la caja.

Métodos

- vincularCaja: representa la operación que permite relacionar la caja con una factura.

Clase Empresa: describe conceptualmente las personas u organizaciones que ofrecen los productos y/o servicios.

Atributos

- IdEmpresa: representa el código de identificación de la empresa.

- nombreEmpresa: representa el nombre de la empresa.

- tipoIdEmpresa: representa el tipo de identificación de la empresa. Por ejemplo: nit, cedula del representante, etc.

Métodos

- vincularEmpresa: representa la operación que permite relacionar la empresa con una factura.

- Clase Cajero: describe conceptualmente los funcionarios que operan las cajas donde se desarrollan las operaciones de venta.

Atributos

- IdCajero: representa el código de identificación del cajero.

- nombreCajero: representa el nombre del cajero.

Métodos

- vincularCajero: representa la operación que permite relacionar un cajero con una factura.

- Clase Cliente: describe conceptualmente las personas u organizaciones que adquieren los productos y/o servicios ofrecidos por las empresas.

Atributos

- IdCliente: representa el código de identificación del cliente.

- nombreCliente: representa el nombre del cliente.

Métodos

- vincularCliente: representa la operación que permite relacionar un cliente con una factura.

- Clase Pago: describe conceptualmente los desembolsos que realizan los clientes por la adquisición de los productos y/o servicios ofrecidos por las empresas.

Atributos

- formaPago: representa la manera en la que se realiza el pago. Por ejemplo: efectivo, tarjeta crédito, tarjeta débito, bono, etc.

- IdPago: representa el código de identificación del pago.

- IdFactura: representa el código de identificación de la factura a la cual está vinculado el pago.

- vrPago: representa el valor del pago.

Métodos

- registrarPago: representa la operación que permite vincular el pago a una factura y almacenar su información.

- Clase Item: describe conceptualmente uno o un conjunto del mismo producto o servicio.

Atributos

- cant: representa la cantidad de productos o servicios por ítem.

- estado: representa un estado que determina si el ítem ha sido adicionado o eliminado de la factura.

- IdFactura: representa el código de identificación de la factura a la cual está vinculado el ítem.

- IdItem: representa el código de identificación del ítem.

- ivaTotalItem: representa la sumatoria del IVA de los productos o servicios del ítem.
- productoServicio: representa el producto o servicio vinculado al ítem.

- vrTotalItem: representa la sumatoria de los valores de los productos o servicios del ítem.

Métodos

- actualizarIvaTotalItem: representa la operación que permite acumular el IVA de los productos o servicios del ítem.

- actualizarVrTotalItem: representa lo operación que permite acumular el valor de los productos o servicios del ítem.

- eliminarItem: representa lo operación que permite cambiar el estado del ítem a eliminado y borrarlo de la factura.

- registrarItem: representa la operación que permite almacenar la información del ítem.

- vincularItem: representa la operación que permite relacionar el ítem con una factura.

- Clase ProductoServicio: describe conceptualmente los productos o servicios ofrecidos por las empresas.

Atributos

- IdProductoServicio: representa el código de identificación del producto o servicio.

- ivaProductoServicio: representa el IVA del producto o servicio.

- nombreProductoServicio: representa el nombre del producto o servicio.

- vrProductoServicio: representa el valor del producto o servicio.

Métodos

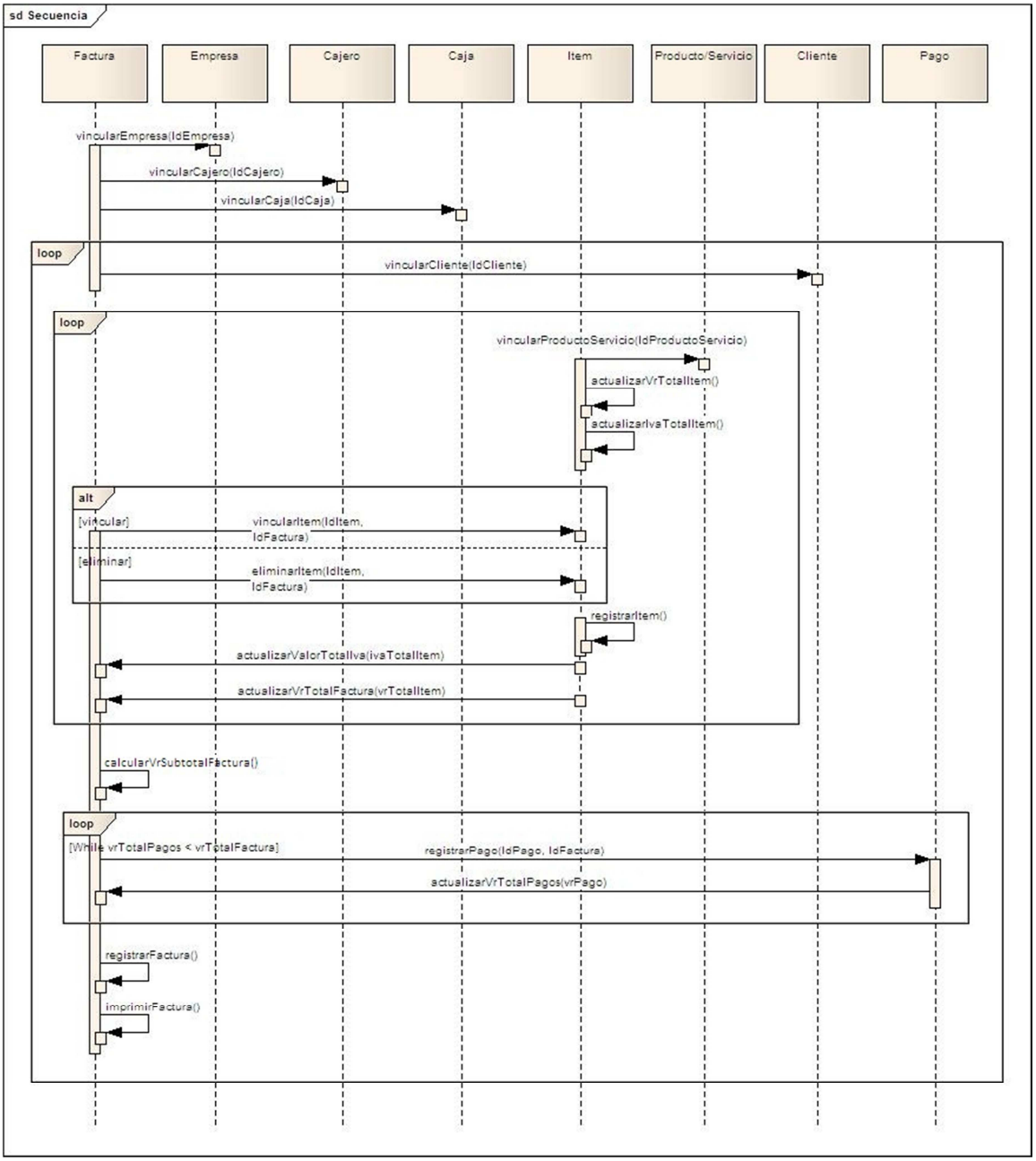
- vincularProductoServicio: representa la operación que permite relacionar un producto o servicio con un ítem.

Cardinalidad

- Una factura tiene una caja, una caja puede tener cero o más facturas.
- Una factura tiene una empresa, una empresa puede tener cero o más facturas.
- Una factura tiene un cajero, un cajero puede tener cero o más facturas.
- Una factura tiene un cliente, un cliente puede tener cero o más facturas.
- Una factura tiene uno o más pagos, un pago tiene una factura.
- Una factura tiene uno o más ítems, un ítem tiene una factura.
- Un ítem tiene un producto o servicio, un producto o servicio puede tener cero o más ítems.

- Modelo de Comportamiento

Figura 40: Diagrama de secuencia del patrón factura



Objetos

- Factura: representa el documento concreto que contiene la información de una operación de venta de productos y/o servicios.

- Empresa: representa la organización o persona concreta que ofrece productos y/o servicios.

- Cajero: representa el funcionario concreto que opera la caja donde se desarrollan las operaciones de venta.

- Caja: representa el punto de venta concreto donde se desarrollan las operaciones de venta.

- Ítem: representa concretamente uno o un conjunto del mismo producto o servicio.
- ProductoServicio: representa el producto o servicio concreto ofrecido por la empresa.

- Cliente: representa la persona u organización concreta que adquiere los productos y/o servicios ofrecidos por la empresa.

- Pago: representa el desembolso concreto que realiza el cliente por la adquisición de los productos y/o servicios ofrecidos por la empresa.

Mensajes

- actualizarIvaTotalItem: representa el proceso de acumular el IVA de los productos o servicios del ítem. Este proceso se ejecuta por cada unidad de producto o servicio vendido.

- actualizarVrTotalFactura: representa el proceso de acumulación del valor total de los ítems adquiridos en la operación de venta. Este proceso se ejecuta por cada ítem.

- actualizarVrTotalItem: representa el proceso de acumular el valor de los productos o servicios del ítem. Este proceso se ejecuta por cada unidad de producto o servicio vendido.

- actualizarVrTotalIva: representa el proceso de acumular el valor total del IVA de los ítems adquiridos en la operación de venta. Este proceso se ejecuta por cada ítem.

- actualizarVrTotalPagos: representa el proceso de acumular el valor de los pagos realizados por el cliente en la operación de venta.

- calcularVrSubtotalFactura: representa el proceso de restar el valor total del IVA de la factura al valor total de la factura. Este proceso se ejecuta una sola vez por cada venta.

- eliminarItem: representa el proceso de cambiar el estado del ítem ha eliminado y borrarlo de la factura.

- imprimirFactura: representa el proceso de imprimir la factura una vez terminada la operación de venta.

- registrarFactura: representa el proceso de hacer persistencia de la factura.

- registrarItem: representa el proceso de almacenar la información del ítem vendido.
- registrarPago: representa el proceso de vincular el pago a la factura y almacenar su información.

- vincularCaja: representa el proceso de relacionar la caja con la factura. Este proceso se ejecuta una sola vez, al inicio de la jornada laboral.

- vincularCajero: representa el proceso de relacionar el cajero con la factura. Este proceso se ejecuta una sola vez, al inicio de la jornada laboral.

- vincularCliente: representa el proceso de relacionar el cliente con la factura en la operación de venta.

- vincularEmpresa: representa el proceso de relacionar la empresa con la factura. Este proceso se ejecuta una sola vez, al inicio de la jornada laboral.

- vincularItem: representa el proceso de relacionar el ítem vendido con la factura. Este proceso se ejecuta por cada ítem.

- vincularProductoServicio: representa el proceso de relacionar el producto o servicio con el ítem. Este proceso se ejecuta por cada producto o servicio.

Ciclos

Los procedimientos de vinculación de caja, cajero y empresa se realizan únicamente al inicio de la jornada, por esto no son incluidos en los ciclos de facturación.

loop venta: representa el ciclo mayor, en el cual se realizan los procedimientos de:

- Vinculación de productos y/o servicios a los ítems.
- Actualización del valor total de los ítems.
- Actualización de IVA total de los ítems.
- Vinculación o eliminación de ítems en la factura.
- Registro de ítems.
- Actualización del valor total de la factura.
- Actualización del valor total del IVA de la factura.
- Cálculo del valor subtotal de la factura.
- Registro de los pagos.
- Actualización del valor total de los pagos.
- Registro e impresión de la factura.

loop items: representa el ciclo donde se vinculan los productos y/o servicios a los ítems y se actualizan sus correspondientes valores, se vinculan o eliminan los ítems de la factura y se actualizan el valor del IVA y los valores totales de factura.

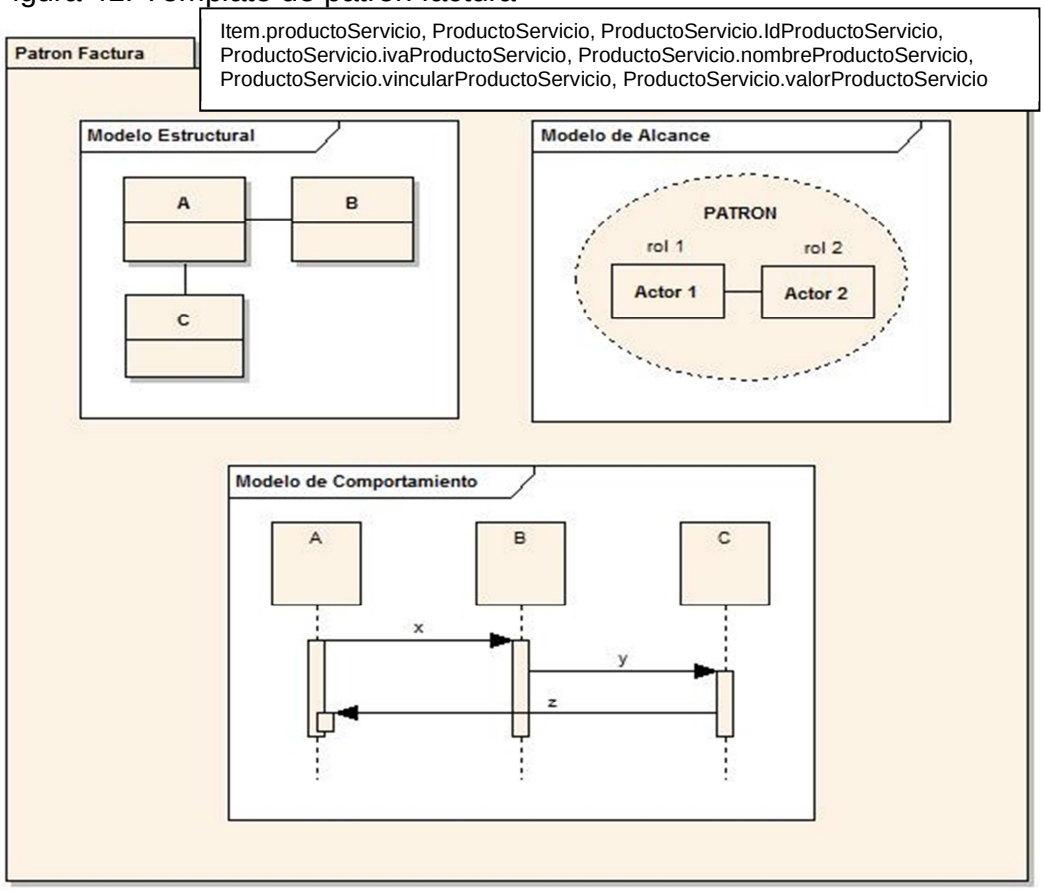
loop pagos: representa el ciclo en el cual se registran los pagos y se actualiza el valor total de pagos en la factura. Este ciclo se ejecuta mientras que el valor total de pagos sea menor que el valor total de la factura.

- Alternativas

alt vincular/eliminar: representa el punto de flujo donde se selecciona una opción de operación sobre un ítem de la factura. Las opciones son: eliminar o vincular.

- Template de Patrón Factura

Figura 41: Template de patrón factura



- Patrón transacción

El patrón de análisis Transacción describe una propuesta de solución para mantener un registro de cambios positivos o negativos en un depósito, realizados a través de unas entradas. Estas entradas son generadas a través de una o más transacciones.

- Usos conocidos:

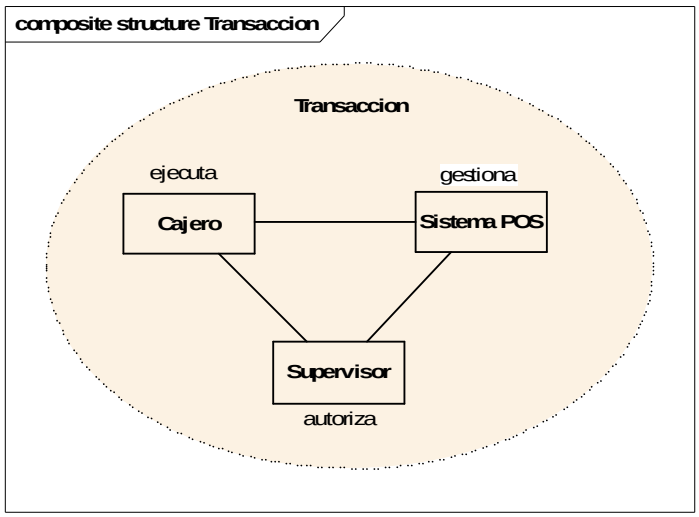
- Sistemas bancarios – manejo de cuentas de clientes
- Inventarios y depósitos – gestión de entradas y salidas
- Aerolíneas – manejo de cuentas de clientes

- Modelo de Alcance

Tabla 8: Asignación de responsabilidades patrón transacción

RESPONSABILIDAD	CLASE RESPONSABLE	ACTOR RESPONSABLE
Generar entrada	Transacción	Cajero/Supervisor
Acumular entrada	Entrada	Sistema POS
Validar forma de transacción	Transacción	Cajero
Actualizar nivel	Depósito	Sistema POS

Figura 42: Colaboración patrón transacción



Actores

- Cajero: representa el funcionario que opera la caja donde se realiza la transacción.

- Sistema POS: representa el software que gestiona los procesos que implican una transacción.

- Supervisor: representa el funcionario encargado de controlar y permitir los procesos de alto nivel requerido en un punto de venta.

Roles

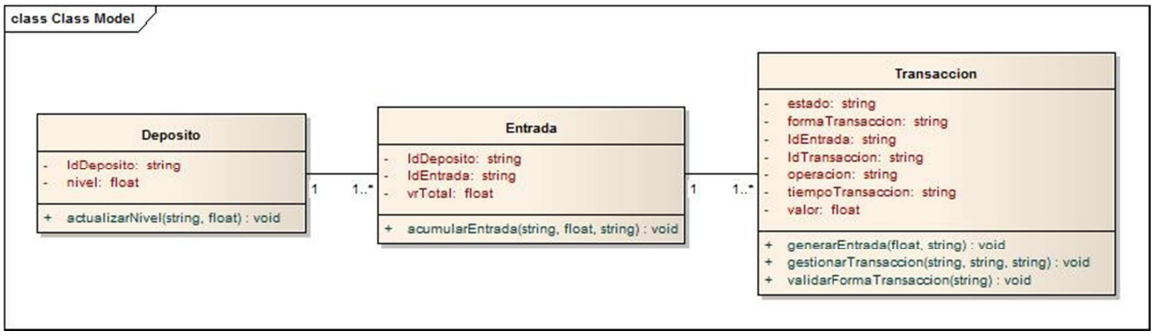
- Autorizar: representa la función de conceder autoridad, facultad o derecho para realizar una acción que requiera de un alto compromiso para la empresa durante una transacción.

- Ejecutar: representa la función de llevar a cabo una transacción.

- Gestionar: representa la función de recibir, procesar, validar, almacenar la información referente a la operación de transacción.

Modelo Estructural

Figura 43: Diagrama de clases del patrón transacción



Clase transacción: describe conceptualmente la acción que genera la entrada que será acumulada para afectar un depósito.

Atributos

Estado: representa un estado que determina que acciones se deben llevar a cabo durante la transacción.

Forma transacción: representa la manera en la que se realiza el proceso que genera la entrada, por ejemplo: si la transacción es un pago éste puede ser en efectivo, tarjeta débito o crédito, bonos, etc.

Id Entrada: representa el código de identificación de cada entrada generada por una transacción.

Id transacción: representa el código de identificación de la transacción.

Operación: representa la forma en que la entrada afectara al depósito. Si será positiva o negativamente.

Tiempo transacción: representa el periodo de tiempo durante el cual es realizada la transacción.

Valor: representa la cantidad de elementos que se generan en una transacción. El tipo de elementos dependerá del tipo de transacción, por ejemplo: si se habla de una transacción económica dicha cantidad será de dinero, si se refiere a un depósito de una bodega la cantidad será de ítems.

Métodos

- Generar Entrada: representa la operación que permite crear la entrada con su respectivo valor y operación a realizar.

- Gestionar Transacción: representa la operación que permite llevar un registro de las transacciones realizadas con un identificador, el tiempo de transacción y el estado de la misma.

- Validar forma transacción: representa la operación que permite al sistema reconocer el tipo de transacción a realizar.

Clase Entrada: describe conceptualmente el punto donde se acumularan todas las posibles entradas que afecten a un depósito.

Atributos

- Id Depósito: representa el código de identificación del depósito que será afectado por las entradas.

- Id Entrada: representa el código de identificación de cada entrada generada por una transacción.

- Vr total: representa la cantidad de elementos que afectaran al depósito.

Métodos

- Acumular Entrada: representa la operación que permite aumentar o disminuir el valor total de la entrada que afectará finalmente al depósito.

- Clase Depósito: describe conceptualmente el lugar donde se almacenan las entradas.

Atributos

- Id Depósito: representa el código de identificación del depósito que será afectado por las entradas.

- Nivel: representa la cantidad de elementos existentes en el depósito.

Métodos

- Actualizar Nivel: representa la operación que permite aumentar o disminuir el nivel del depósito.

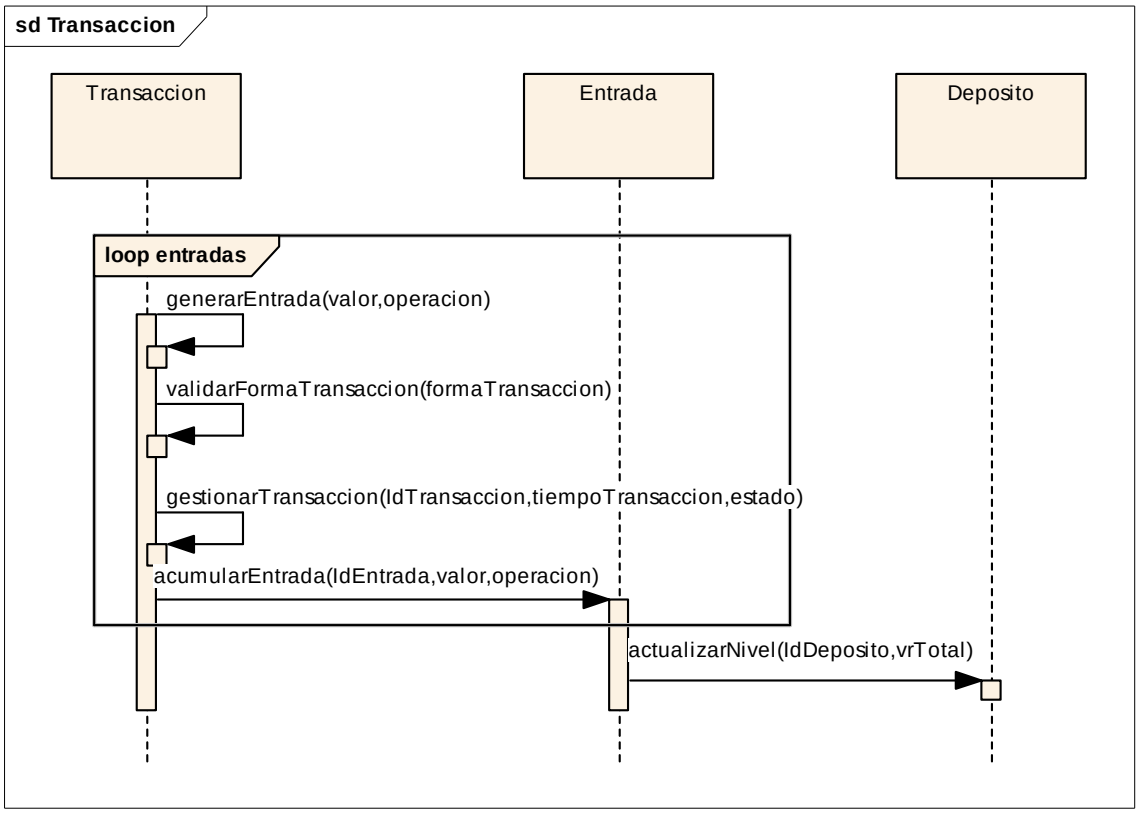
Cardinalidad

- Una transacción puede generar una entrada, una entrada puede ser generada por una o más transacciones.

- Una entrada puede afectar a un solo depósito, un depósito puede ser afectado por una o más entradas.

Modelo de Comportamiento

Figura 44: Diagrama de secuencia del patrón transacción



Objetos

- transacción: representa el proceso concreto que genera la entrada que será acumulada para afectar un depósito.

- Entrada: representa el punto concreto donde se acumularan todas las posibles entradas que afecten a un depósito.

- Deposito: representa el lugar concreto donde se almacenan las entradas.
- Mensajes

- Generar entrada: representa el proceso de crear la entrada con su respectivo valor y operación a realizar. Este proceso se ejecuta una vez por cada transacción.

- Validar forma transacción: representa el proceso de reconocer el tipo de transacción a realizar. Este proceso se ejecuta una vez por cada transacción.

- Gestionar Transacción: representa el proceso de llevar un registro de las transacciones realizadas con un identificador, el tiempo de transacción y el estado de la misma. Este proceso se ejecuta una vez por cada transacción.

- Acumular Entrada: representa el proceso de aumentar o disminuir el valor total de la entrada que afectará finalmente al depósito. Este proceso se ejecuta una vez por cada transacción.

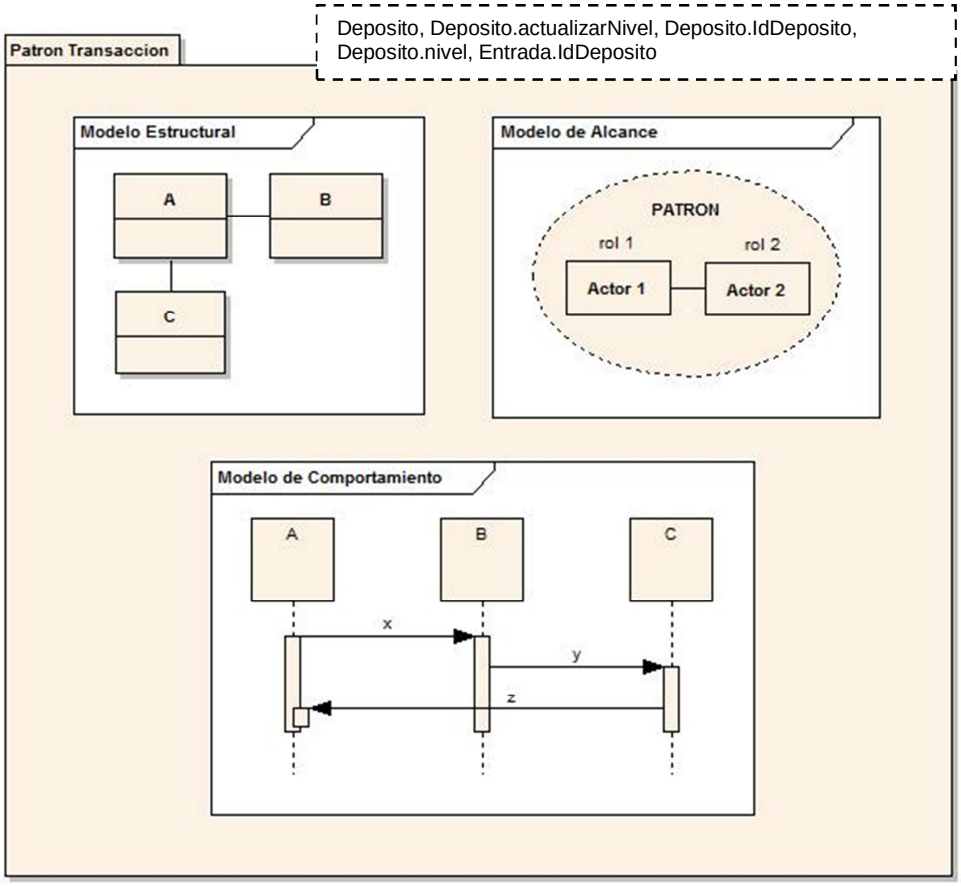
- Actualizar Nivel: representa el proceso de aumentar o disminuir el nivel del depósito. Este proceso se ejecuta una vez ya estén acumuladas las entradas.

Ciclos

- loop entradas: representa el ciclo donde se realizan las transacciones que generan una entrada, se validan y registran las transacciones y se acumulan las entradas que afectaran finalmente al depósito.

- Template del patrón transaccion

Figura 45: Template del Patrón transacción



Patrón Acción

El patrón de análisis Acción describe una propuesta de solución para el registro de acciones llevadas a cabo dentro del dominio. Los posibles estados de una acción se dividen en dos subtipos principales: acción propuesta y acción implementada, que representan la intención y lo que realmente sucede. El final de una acción está igualmente dividido en acción completada y acción abandonada. Una acción abandonada representa una cancelación definitiva de la acción, y posponer temporalmente la acción representa una suspensión.

- Usos conocidos:

Supermercados – programación de envío de información a otros sistemas.

Aerolíneas – programación de vuelos.

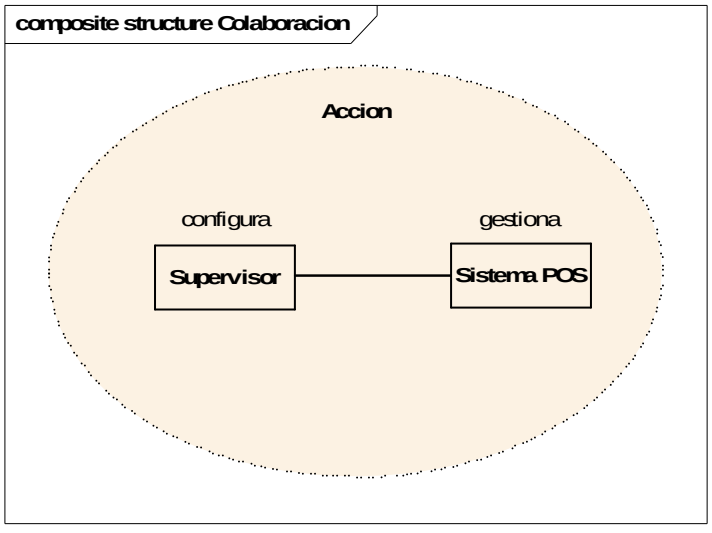
Restaurantes – reservaciones.

- Modelo de Alcance

Tabla 9: Asignación de responsabilidades patrón acción

RESPONSABILIDAD	CLASE RESPONSABLE	ACTOR RESPONSABLE
Programar acción	AccionPropuesta	Sistema POS/ Supervisor
Iniciar acción	AccionImplementada	Sistema POS/ Supervisor
Completar acción	AccionCompletada	Sistema POS/ Supervisor
Cancelar acción	AccionAbandonada	Sistema POS/ Supervisor
Posponer acción	Suspension	Sistema POS/ Supervisor
Retomar acción	Suspension	Sistema POS/ Supervisor

Figura 46: Colaboración patrón acción



Actores

- Supervisor: representa el funcionario encargado de la configuración de las acciones que se llevan a cabo en el dominio POS.

- Sistema POS: representa el software que gestiona los procedimientos para llevar a cabo acciones dentro del dominio POS.

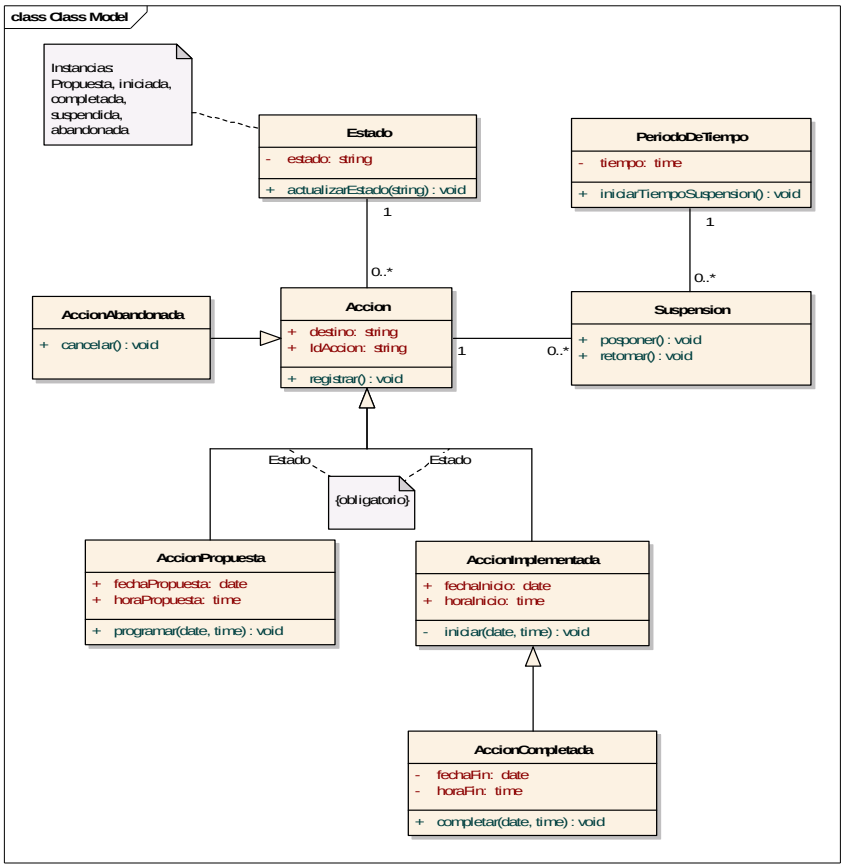
Roles

- Configurar: representa la función de programar e iniciar la implementación de acciones, las cuales podrán ser completadas, canceladas o suspendidas.

- Gestionar: representa la función de recibir, procesar, administrar y almacenar la información referente a las actividades que se llevan a cabo en el dominio POS.

Modelo Estructural

Figura 47: Diagrama de clases del patrón acción



Clase acción: describe conceptualmente las actividades que se llevan a cabo en el dominio.

Atributos

- Destino: representa el lugar donde se va a llevar a cabo la acción.

- Id Acción: representa el código de identificación de la acción.

Métodos

- Registrar: representa la operación que permite almacenar la información de la acción.

Clase acción Propuesta: describe conceptualmente las actividades que se programan dentro del dominio.

Atributos

- Fecha Propuesta: representa la fecha para la cual se programa la acción.

- Hora Propuesta: representa la hora para la cual se programa la acción.

Métodos:

Programar: representa la operación que permite programar la acción.

Clase acción implementada: describe conceptualmente las actividades que se realizan dentro del dominio.

Atributos

- Fecha Inicio: representa la fecha en que se inicia la realización de la acción.

- Hora Inicio: representa la hora en que se inicia la realización de la acción.

Métodos

- Iniciar: representa la operación que permite comenzar la implementación de la acción.

- Clase acción completada: describe conceptualmente las actividades que se han terminado de realizar.

Atributos

- Fecha Fin: representa la fecha en que se completó la acción.

- Hora Fin: representa la hora en que se completó la acción.

Métodos

- Completar: representa la operación que permite finalizar la acción. Completar

- Clase acción abandonada: describe conceptualmente las actividades que se cancelan completa y definitivamente.

Métodos

- Cancelar: representa la operación que permite anular por completo y definitivamente la acción.

- Clase Estado: describe conceptualmente las etapas en la que se encuentran las acciones.

Atributos

- Estado: representa la etapa en la que se encuentra la acción. Por ejemplo: propuesta, iniciada, completada, suspendida o abandonada.

Métodos

- Actualizar estado: representa la operación que permite cambiar del estado anterior al estado actual de la acción.

- Clase Suspensión: describe conceptualmente la cesación temporal de las acciones.

Métodos:

- Posponer: representa la operación que permite detener de manera temporal la acción.

- Retomar: representa la operación que permite continuar la acción que fue suspendida.

- Clase Periodo de tiempo: describe conceptualmente el intervalo de tiempo durante el cual se suspende una acción.

Atributos

- Tiempo: representa la duración de la suspensión.

Métodos

- Iniciar tiempo suspensión: representa la operación que permite comenzar el tiempo de suspensión de la acción.

Cardinalidad

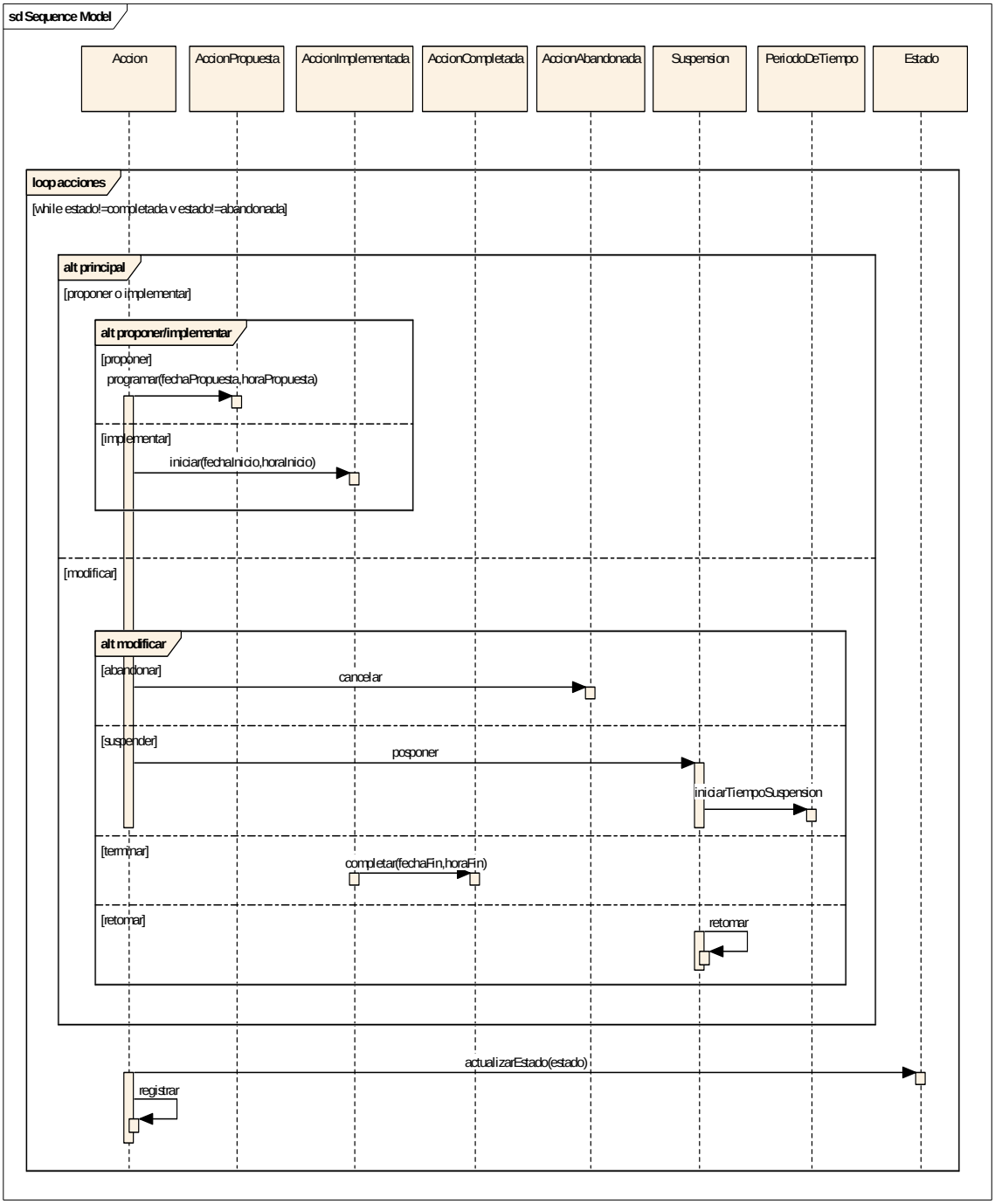
- Una acción puede ser suspendida cero o más veces, cero o más suspensiones pertenecen a una acción.

- Cero o más acciones deben tener un estado, un estado hace parte de cero o más acciones.

- Cero o más suspensiones deben tener un periodo de tiempo, un periodo de tiempo pertenece a cero o más suspensiones.

- Modelo de Comportamiento

Figura 48: Diagrama de secuencia patrón acción



Objetos

- Acción: representa la actividad concreta que se lleva a cabo en el dominio.

- Acción propuesta: representa la actividad concreta que se programa dentro del dominio.

- Acción implementada: representa la actividad concreta que se realiza dentro del dominio.

- Acción completada: representa la actividad concreta que se ha terminado de realizar.

- Acción abandonada: representa la actividad concreta que se cancela completa y definitivamente.

- Suspensión: representa el cese temporal concreto de la acción.

- Periodo de tiempo: representa el intervalo de tiempo concreto durante el cual se suspende la acción.

- Estado: representa la etapa concreta en la que se encuentra la acción.

Mensajes

- Programar: representa el proceso de programación de la acción.

- Iniciar: representa el proceso de implementación de la acción. Este proceso puede ejecutarse habiendo o no programado la acción.

- Cancelar: representa el proceso de anulación completa y definitiva de la acción. Este proceso puede ejecutarse si la acción ha sido programada o implementada anteriormente.

- Posponer: representa el proceso de detención temporal de la acción. Este proceso puede ejecutarse si la acción ha sido programada o implementada anteriormente.

- Iniciar tiempo suspensión: representa el proceso de inicialización del tiempo de suspensión de la acción. Este proceso se ejecuta una vez se suspenda la acción.
- Completar: representa el proceso de finalización de la acción. Este proceso puede ejecutarse si la acción se ha implementado anteriormente.

- Retomar: representa el proceso de continuación de la acción que fue suspendida.

- Actualizar estado: representa el proceso de cambio de estado de la acción. Este proceso se ejecuta cada vez que la acción cambie de estado.

- Registrar: representa el proceso de almacenamiento de la información de la acción.

Ciclos

loop acciones: representa el ciclo en el cual se realizan todos los procedimientos de programación, implementación, finalización, cancelación y suspensión de la acción. Este ciclo se ejecuta mientras que el estado de la acción sea diferente de completado o abandonado.

Alternativas

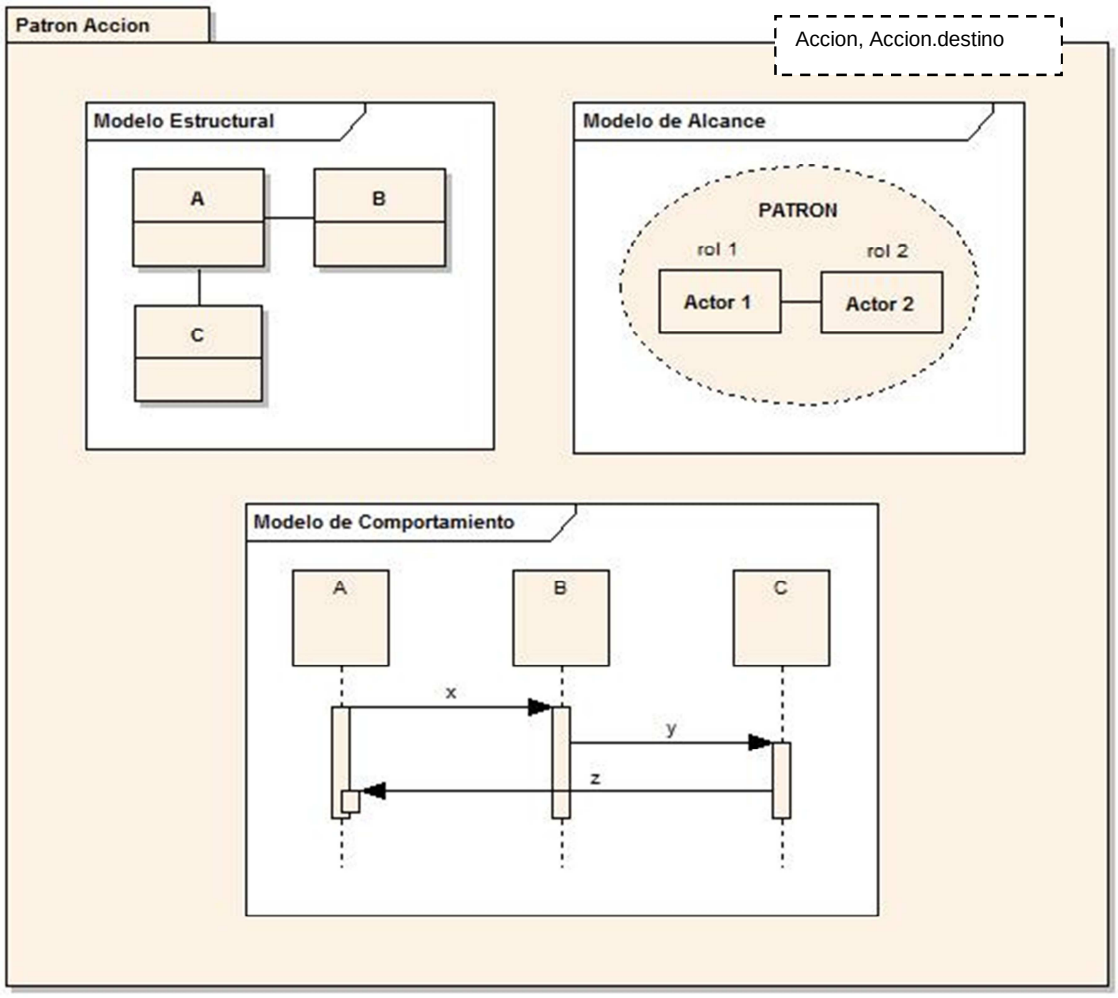
- Alt principal: representa el punto de flujo donde se selecciona una opción de operación sobre una acción. Las opciones son: proponer o implementar, ó modificar la acción que se propuso o implementó con anterioridad.

- Alt proponer/implementar: representa el punto de flujo donde se selecciona una opción de operación sobre una acción. Las opciones son: proponer o implementar.

- Alt modificar: representa el punto de flujo donde se selecciona una opción de operación sobre una acción ya propuesta o implementada. Las opciones son: abandonar, suspender, terminar o retomar.

- Template del patrón acción

Figura 49: Template del patrón acción



Todos los modelos de cada uno de los patrones están contenidos en un Template, el Template es representado por un Signature, el cual toma importancia debido a que representa el enfoque aspectual, este mecanismo permite instanciar el patrón por medio de sus variables convirtiendo el patrón en un problema específico realizándose de esta forma la instanciación del patrón para modelar una solución específica.

3.10 HERRAMIENTAS CASE

Las herramientas CASE (Computer Aided Software Engineering, Ingeniería de Software Asistida por Computadora) son herramientas informáticas destinadas a aumentar la productividad en el desarrollo de software reduciendo el costo de las mismas en términos de tiempo y de dinero.

Estas herramientas nos pueden ayudar en todas las etapas del ciclo de vida de desarrollo del software, en tareas como el proceso de realizar un diseño del proyecto, cálculo de costes, implementación de parte del código automáticamente con el diseño dado, compilación automática, documentación o detección de errores entre otras.

Para realizar el modelado del Sistema POS del caso de estudio del presente proyecto investigativo, se hace necesario realizar un estudio de las herramientas CASE identificando qué herramientas existentes en el mercado actualmente, soportan el modelado de los diagramas de clases, diagramas de secuencia y las colaboraciones, además es importante verificar si existen herramientas CASE que ofrezcan el uso de templates parametrizables. Esto debido al modelo de representación en el que se basan los patrones de análisis de TRIPODE.

En las Tablas 10 y 11 se encuentra consignada la información resultante de este análisis

Tabla 10. Cuadro comparativo herramientas CASE Parte I

HERRAMIENTA CASE	VERSION	TIPO DE SOFTWARE	UML	TEMPLATE	D. DE CLASES	D. DE SECUENCIA	COLABORACIONES	OBSERVACIONES
Altova UModel	v2010 Release 3	licenciado	2.0	NO	SI	SI	SI	Integración con Visual Studio 2010 Apoyo a la versión de C # 4.0 Soporte para los diagramas de estado de máquina de protocolo Opción adicional para la generación de diagramas de secuencia de código existente
Enterprise Architect	8.0	licenciado	2.3	NO	SI	SI	SI	Flexible, completa y poderosa para plataformas Windows. Orientada a objetos Desarrollo de sistemas, administración de proyectos y análisis de negocios. Maneja totalmente el ciclo de vida y el análisis y diseño UML Lenguajes de desarrollo ActionScript, C, C++, C# y VB.NET, Java, Visual Basic 6, Python, PHP, XSD, WSDL y más.
UModel	2010	licenciado	2.3	NO	SI	SI	SI	Genera código en Java, C #, VB.NET lenguas existentes ingenieros inversa Java, C #, VB.NET o código Java, C #, VB.NET Apoyos archivos binarios las últimas versiones: Java 6.0, C # 3.0 y Visual Basic 9.0 Integración con Visual Studio y Eclipse modelo de intercambio con herramientas heredadas a través de XMI.
StarUML	5.0	libre	2.0	SI	SI	NO	SI	Un proyecto de código abierto para desarrollar rápido, flexible, extensible. Está diseñado para apoyo de la MDA. Es principalmente escrito en Delphi. Sin embargo, StarUML es un proyecto multi-idioma.

Tabla 11. Cuadro comparativo herramientas CASE Parte II

HERRAMIENTA CASE	VERSION	TIPO DE SOFTWARE	UML	TEMPLATE	D. DE CLASES	DI. DE SECUENCIA	COLABORACIONES	OBSERVACIONES
MagicDraw UML	16.9	licenciado	2.3	NO	SI	SI	SI	Desarrollo de código para diversos lenguajes de programación (Java, C++ y C#, entre otros)
								La herramienta cuenta con capacidad para trabajar en equipo.
								Es compatible con las siguientes IDEs:
								Sun Java Studio 8.
								Borland CaliberRM 6.0, 6.5 herramienta de requisitos.
								Oracle Workshop 8.1.2.
								E2E Bridge 4.0
								IntelliJ IDEA 4.X or later.
								NetBeans 6.X or later.
								Eclipse 3.1 o superior (versión Java).
								IBM Rational Application Developer
								Borland JBuilder 8.0, 9.0, X, 2005, 2006, 2007
IBM Rational Rose Enterprise	-	licenciado	2.0	NO	SI	SI	SI	Da soporte a Unified Modeling Language (UML), pero no son compatibles con las mismas tecnologías de implementación.
								Permite generar código a partir de modelos Ada, ANSI C++, C++, CORBA, Java/J2EE, Visual C++ y Visual Basic.
								Soporte a modelos de análisis, ANSI C++, Rose J y Visual C++ según el documento "Design Patterns: Elements of Reusable Object-Oriented Software".
								Soporte para Enterprise Java Beans 2.0.
								Funciones de análisis de calidad de código.
								Complemento de modelado Web que incluye funciones de visualización, modelado y herramientas para desarrollar aplicaciones Web.

4. DEFINICION DEL CASO DE ESTUDIO: SUPERMERCADOS YUCATA

4.1 INTRODUCCION

Este capítulo presenta el proceso de definición del caso de estudio que permitirá validar el funcionamiento del modelo propuesto por ESKEMA para la utilización de un catálogo de patrones de análisis dentro de un dominio específico. Inicialmente se hace el estudio del dominio POS, pues es éste el dominio para el cual se han definido los patrones en el catálogo TRIPODE, luego se presenta un estudio sobre los tipos de POS y los sistemas POS existentes, que permitirá determinar que el tipo de POS idóneo es un Supermercado. Finalmente se hace la presentación del caso de estudio Supermercados YUCATA y el documento del sistema deseado presentado por los clientes del caso de estudio.

4.2 DESCRIPCIÓN DEL DOMINIO POS

Uno de los sectores económicos más importantes a nivel mundial es el sector comercio, en cual se llevan a cabo procesos sistematizados y correctos. Dentro del sector comercio las organizaciones que realizan tareas de fabricación, distribución y mercadeo de productos, basan sus objetivos en la venta de los mismos, por esto, este proceso toma importancia.

Dentro del proceso de la venta, el POS como punto interno de la organización donde se desarrolla la venta de productos y/o servicios es de gran importancia, siendo éste el punto donde el cliente realiza la transacción final para obtener el producto.

4.3 DEFINICIÒN DEL PROCESO DE VENTA

En toda empresa existe un objetivo fundamental hacia el que están enfocados prácticamente todos los esfuerzos de la organización y a cuyo servicio se ponen todos los sistemas que dispone la empresa. Dicho objetivo no es otro que la venta de los productos y/o servicios. La venta activa toda una serie de procesos administrativos como impuestos pagos de bonificaciones por venta, inventarios, compras, etc.

El proceso de venta es toda actividad que incluye un proceso personal o impersonal mediante el cual, el vendedor identifica las necesidades y/o deseos del comprador, genera el impulso hacia el intercambio y satisface las necesidades

y/o deseos del comprador (con un producto, servicio u otro) para lograr el beneficio de ambas partes25.

Dentro del proceso de la venta hay un punto fundamental que es llamado POS, en este punto se desarrollan todas las transacciones activadas por la venta.

4.4 DESCRIPCION DEL DOMINIO DE APLICACIÓN POS

4.4.1 Definición de dominio de aplicación POS. El POS hace referencia al lugar dentro de una empresa o negocio en donde se efectúa el proceso de salida y cobro de los productos o servicios que ofrece. En otras palabras es el lugar donde se culmina la operación de venta, es decir, donde el Cliente, paga el importe por un producto o servicio que está adquiriendo y se registra dicha transacción, dándole al cliente un recibo, comprobante o factura que ampara el pago dado por dicho producto o servicio. Son puntos de especial cuidado, porque en éstos se encuentran los soportes de ingreso de cualquier actividad económica lucrativa.

En la comunicación en el punto de venta26 se afirma que el POS adquiere una dimensión superior al simple hecho de ser el lugar físico del intercambio comercial, el establecimiento por sí mismo es capaz de generar y transmitir emociones, sensaciones, sentimientos y experiencia, crear ambientes socioculturales, agudizar los sentidos con el objetivo último de favorecer la compra y el hábito de consumo. El POS en su globalidad actúa como un poderoso medio o canal de comunicación que transmite informaciones de forma ininterrumpida y es capaz de influir en su público. El cliente no sólo compra en un establecimiento, sino también se informa, se educa, se entretiene y, sobre todo, recibe toda clase de estímulos.

El POS es el lugar donde convergen los intereses de todos los actores implicados en la trama comercial, es decir la industria, los distribuidores y los compradores/consumidores finales.

Cada uno de los actores comerciales tiene unos intereses particulares: el fabricante pretende vender más productos de su marca, el distribuidor busca la mayor rentabilidad posible y el cliente busca una mejor información, una mayor calidad, un mejor precio, mayores servicios, etc. Por tanto se considera viable tomar el POS como dominio ya que se convierte en el centro convergente de los fabricantes con sus productos, los comerciantes con su gestión y los clientes con sus necesidades y deseos de compra.

---

[25] American Marketing Asociation. MarketingPower.com, sección Dictionary of Marketing Terms. Internet (www.marketingpower.com <http://www.marketingpower.com/>)

[26] Martínez, I.J. (2005): La comunicación en el punto de venta: estrategias de comunicación en el comercio real y on-line. ESIC, Madrid.

4.4.2 Tipos de POS. Es posible realizar una clasificación de los POS existentes en el sector comercial de acuerdo a las funcionalidades requeridas, al tamaño de la organización y a la actividad comercial desempeñada por la empresa. Los POS pueden clasificarse en cuatro categorías:

Supermercados y autoservicios: en este grupo se reúnen las necesidades de hipermercados, supermercados y autoservicios.

Tiendas y negocios de venta minorista: se especializa en el sector de pequeños negocios, brindándoles un manejo eficiente de su actividad comercial. En esta categoría se pueden incluir negocios tales como droguerías, disco tiendas, librerías, etc.

Gastronomía: orientado a negocios que trabajan con la preparación y venta de alimentos tales como restaurantes, bares y negocios de comidas rápidas.

Clubes: en esta categoría se manejan los negocios de carácter social, cultural o deportivo en su mayoría de tipo privado. Los servicios y productos que estos negocios ofrecen son: restaurante, bar, gimnasio, hospedaje, membrecías, eventos, entre otros.

4.4.3 Sistemas POS. Los negocios cuya actividad económica es la venta de gran número de productos y variedades de manera directa a sus clientes, requieren de un mecanismo que les permita mantener un control permanente sobre las ventas, el inventario y la interacción con sus proveedores y clientes; este mecanismo es llamado Sistema POS.

El Sistema POS es un sistema compuesto por software y hardware, creado con el fin de automatizar y agilizar los procesos relacionados con ventas y las demás actividades que puedan depender de ella. Se puede afirmar que el objetivo de la implantación de este tipo de sistemas dentro de una empresa, es mejorar la funcionalidad de los procesos y poder tener un control sobre información clave de manera consistente y actualizada. Además el sistema POS debe permitir una interacción sistema-usuario a través de una interfaz agradable y de fácil uso.

En L. Londoño27 presenta algunos ejemplos de sistemas POS que actualmente se encuentran en el mercado de software, tales como: Aloha POS, el cual satisface los requisitos particulares de un restaurante o de un bar; RETAIL, que es una herramienta de software para venta al detal; Tina POS, aplicación útil para

[27] Londoño, Luis Fernando. Definición de las Necesidades para la Construcción de un Sistema de Puntos De Venta, Universidad EAFIT, 2008.

comercios detallistas; Checkout POS y Checkout Restaurant, que integran el punto de venta y la administración en una misma base de datos replicada.

- Funcionalidades de los sistemas POS

Las funcionalidades que el sistema brinde deben acoplarse a las necesidades de cada empresa. Estas funcionalidades se identificaron a partir de los sistemas POS mencionados anteriormente.

- Supermercados y autoservicios. Dentro de las funcionalidades que un sistema POS debe brindar a este tipo de Puntos de Venta, encontramos:

Facturación:

Manejo de multiprecios (divisas)

Manejo de descuentos por valor o porcentaje, por productos en promoción, por grupos de productos, por horas, por fechas, etc.

- Cotizaciones.
- Control de domicilio.
- Cambios de productos.
- Impresión de Copias.
- Anulación de Facturas.
- Devolución de Clientes.
- Control de mercancía separada.
- Manejo de pedidos de clientes.
- Control de gastos de caja.
- Creación y configuración de medio para venta (Domicilios, etc.).
- Creación y configuración de medio de pago (Tarjetas de Crédito, Débito, etc.)
- Manejo de bonos.

Inventario:

- Seguimiento rápido al movimiento de productos.
- Control de mercancía.
- Control de mercancía en consignación.
- Control de mercancía separada.
- Conversión de desempaque.
- Actualización de precios.
- Control de vencimiento de productos

Cartera:

- Control de pagos y abonos
- Consulta de saldos

Estadísticas:

- Estadísticas de las funciones propias de cada negocio.
- Generación de Reportes de las funciones propias de cada negocio

Presupuesto:

- Proyecciones y control de logro de metas.

Contabilidad:

- Manejo de impuestos
- Cuadres diarios
- Cuentas por Pagar
- Cuentas por Cobrar
- Generación de informe fiscal (DIAN)

Empleados:

- Manejo de comisiones.

Clientes:

- Registrar venta por cliente
- Registrar información básica
- Manejo de programa de fidelización

Proveedores:

- Registrar información

Comunicaciones:

- Manejo de puntos de cadena
- Manejo de sucursales
- Manejo de dependencias

Información:

- Soporte de alto volumen de transacciones
- Administración de información con seguridad y confiabilidad
- Asignación de Perfiles
- Auditoría
- Copias de seguridad.
- Consultas básicas de productos y clientes

Tiendas y negocios de ventas minoristas. Entre las funcionalidades requeridas por este tipo de POS se tiene:

Facturación:

- Control de vencimiento de productos
- Manejo de Remisiones
- Generar facturación
- Creación y configuración de tipos de medio para venta
- Control de domicilio
- Control de mercancía separada
- Manejo de bonos.
- Manejo de descuentos por valor o porcentaje, por productos en promoción, por grupos de productos, por horas, por fechas, etc.

Inventario:

- Agrupación de Productos
- Control de mercancía

Cartera:

- Control de pagos y abonos
- Consulta de saldos

Estadísticas:

- Generación de Reportes de las funciones propias de cada negocio

Contabilidad:

- Arqueo de Caja
- Cuentas por Pagar

Clientes:

- Registrar información básica

Proveedores:

- Registrar información

Información:

- Consultas básicas de productos y clientes

Gastronomía. Entre sus requisitos están:

Facturación:

- Manejo de multiprecios (divisas)
- Control de domicilio
- Emisión rápida de facturas
- Apertura, adición y cierre de mesas
- Creación y configuración de de medio para venta (Domicilios, etc.)
- Creación y configuración de medio de pago (Tarjetas de Crédito, Débito, etc.)

Inventario:

- Seguimiento rápido al movimiento de productos
- Control de mercancía
- Conversión de desempaque.
- Actualización de precios
- Control de vencimiento de productos

Estadísticas:

- Estadísticas de las funciones propias de cada negocio
- Generación de Reportes de las funciones propias de cada negocio

Producción:

- Control de costos en productos terminados de producción y proceso.
- Administración eficiente de inventarios de materia prima y producto terminado.
- Manejo de órdenes de producción

Presupuesto:

- Proyecciones y control de logro de metas.

Contabilidad:

- Cuentas por Pagar
- Arqueo de Caja

Clientes:

- Registrar información básica
- Manejo de programa de fidelización

Proveedores:

- Registrar información
- Información:
- Consultas básicas de productos y clientes

Auditoría

- Clubes. Para dar un adecuado manejo a los servicios y productos que estos negocios ofrecen, tales como restaurante, bar, gimnasio, hospedaje, membrecías, eventos, entre otros; se requiere de las siguientes características:

Facturación:

- Manejo de multiprecios (divisas)
- Control de domicilio
- Emisión rápida de facturas
- Apertura, adición y cierre de mesas
- Facturación de membrecías, eventos y hospedaje.
- Creación y configuración de medio para venta (Domicilios, etc.)
- Creación y configuración de medio de pago (Tarjetas de Crédito, Débito, etc.)

Inventario:

- Seguimiento rápido al movimiento de productos
- Control de mercancía
- Conversión de desempaque.
- Actualización de precios
- Control de vencimiento de productos

Cartera:

- Control de pagos y abonos
- Consulta de saldos

Estadísticas

- Estadísticas de las funciones propias de cada negocio.
- Generación de Reportes de las funciones propias de cada negocio.

Producción

- Control de costos en productos terminados de producción y proceso
- Administración eficiente de inventarios de materia prima y producto terminado
- Manejo de órdenes de producción

Presupuesto

- Proyecciones y control de logro de metas.

Contabilidad:

- Cuentas por Pagar
- Arqueo de Caja
- Generación de informe fiscal (DIAN)
- Manejo de impuestos

Empleados

- Manejo de comisiones.

Clientes

- Registrar información básica
- Registrar venta por cliente
- Manejo de programa de fidelización
- Controlar posibles prospectos de miembros y beneficiarios

Proveedores:

- Registrar información

Comunicaciones:

- Manejo de dependencias

Información:

- Soporte de alto volumen de transacciones
- Administración de información con seguridad y confiabilidad
- Asignación de Perfiles
- Auditoría
- Copias de seguridad.
- Consultas básicas de productos y clientes

4.5 ANALISIS PARA LA DEFINICION DEL CASO DE ESTUDIO

Para determinar el caso de estudio POS adecuado, para aplicar el catálogo de patrones aspectuales de análisis basado en MORENA, se realiza un análisis basado en el grado de uso de las diferentes funcionalidades de cada interfaz que cada una de las categorías de POS expuestas anteriormente utilizan.

Se ha realizado la organización e integración de la totalidad de las funcionalidades encontradas hasta el momento en los tipos de POS expuestos, de acuerdo a la interfaz a la que pertenezcan:

Facturación:

- Generar facturación
- Manejo de multiprecios (divisas)
- Manejo de descuentos por valor o porcentaje, por productos en promoción, por grupos de productos, por horas, por fechas, etc.
- Cotizaciones.
- Control de domicilio
- Cambios de productos
- Impresión de Copias
- Anulación de Facturas
- Devolución de Clientes
- Control de mercancía separada
- Manejo de pedidos de clientes

- Control de gastos de caja
- Creación y configuración de tipos de medio para venta
- Manejo de bonos.
- Control de vencimiento de productos
- Manejo de Remisiones
- Emisión rápida de facturas
- Apertura, adición y cierre de mesas
- Facturación de membresías, eventos y hospedaje.

Inventario:

- Seguimiento rápido al movimiento de productos
- Control de mercancía
- Control de mercancía en consignación
- Control de mercancía separada
- Conversión de desempaque.
- Actualización de precios
- Control de vencimiento de productos
- Agrupación de Productos

Cartera

- Control de pagos y abonos
- Consulta de saldos

Estadísticas

- Estadísticas de las funciones propias de cada negocio
- Generación de Reportes de las funciones propias de cada negocio

Producción

- Control de costos en productos terminados de producción y proceso
- Administración eficiente de inventarios de materia prima y producto terminado
- Manejo de órdenes de producción

Presupuesto

- Proyecciones y control de logro de metas.

Contabilidad:

- Manejo de impuestos
- Arqueo de Caja
- Generación de informe fiscal (DIAN)
- Cuentas por Pagar
- Cuentas por Cobrar

Empleados

- Manejo de comisiones.

Clientes

- Registrar venta por cliente
- Registrar información básica
- Manejo de programa de fidelización
- Controlar posibles prospectos de miembros y beneficiarios

Proveedores

- Registrar información

Comunicaciones:

- Manejo de puntos de cadena
- Manejo de sucursales
- Manejo de dependecias

Información:

- Soporte de alto volúmen de transacciones
- Administración de información con seguridad y confiabilidad
- Asignación de Perfiles
- Auditoría
- Copias de seguridad.
- Consultas básicas de productos y clientes

En la Tabla 12 se muestra el cruce entre las diferentes categorías de POS y las interfaces anteriormente descritas:

Tabla 12: Matriz Interfaz- Tipo POS de acuerdo al grado de uso de las funcionalidades.

ALTO: El POS utiliza de manera intensa la gran mayoría de las funcionalidades de la interfaz
MEDIO: El POS utiliza de manera moderada las funcionalidades de la interfaz
BAJO: El POS hace uso simple de un pequeño número de funcionalidades de la interfaz
--: No Aplica

TIPO POS / **INTERFAZ**	SUPERMERCADOS Y AUTOSERVICIOS	TIENDAS Y NEGOCIOS DE VENTA MINORISTA	GASTRONOMIA	CLUBES
Facturación	Alto	Bajo	Medio	Alto
Inventario	Alto	Alto	Alto	Alto
Cartera	Bajo	Medio	--	Alto
Estadísticas	Alto	Bajo	Medio	Medio
Producción	--	--	Alto	Medio
Presupuesto	Alto	--	Medio	Alto
Contabilidad	Alto	Medio	Alto	Alto
Empleados	Medio	--	--	Alto
Clientes	Alto	Medio	Medio	Alto
Proveedores	Alto	Bajo	Medio	Medio
Comunicaciones	Alto	--	--	Medio
Información	Alto	Bajo	Medio	Alto

Al realizar un análisis de los grados de utilización de las interfaces de cada uno de los tipos de POS, se determina que el tipo de POS Supermercados y Autoservicios tiene las características necesarias para poder determinar un caso de estudio adecuado para la validación del uso del catálogo, pues como se muestra en la Tabla 11, los Supermercados y Autoservicios hacen uso de las funcionalidades de las interfaces en su mayoría en un grado alto, además comercialmente el uso de los Sistemas POS son muy comunes en este tipo de mercado.

En el caso de estudio escogido se incluirán características que incluyan funcionalidades de interfaces cuyo grado de utilización sea alto y medio, entre las que están:

- Facturación
- Inventario
- Contabilidad
- Clientes
- Comunicaciones
- Información

4.6 DESCRIPCIÓN DEL CASO DE ESTUDIO SUPERMERCADOS YUCATA

Para el planteamiento del caso de estudio que permite evaluar la funcionalidad de TRIPODE, se realizó el estudio del dominio POS descrito en el capítulo anterior.

A continuación se presenta el brochure del caso de estudio “Supermercados YUCATA”, el cual se definió con base en la información recolectada de las características de los supermercados de nivel medio y en las funcionalidades que este tipo de negocio necesita encontrar en un sistema POS.

4.6.1 Supermercados Yucatá

Nuestra Empresa

Actualmente SUPERMERCADOS YUCATÁ cuenta con una sede principal ubicada en el centro de nuestra ciudad, y dos sucursales ubicadas en el sector del parque Bolívar y en el centro comercial Valle de Atriz, brindando fácil acceso a productos de consumo masivo a todas las familias pastusas, con la mejor calidad y los mejores precios.

Las instalaciones de estos puntos de atención oscilan en un área aproximada de 500 m2 a 800 m2, ofreciendo instalaciones cómodas para nuestros clientes, y organizando nuestros productos en secciones que brinden mayor facilidad de orientación.

La sede principal cuenta con 10 puntos de pago, y las dos sucursales cuentan con 5 puntos de pago cada una, procurando ofrecer a nuestros clientes un servicio ágil y eficiente.

El horario de atención comienza a las 8 am y termina a las 9 pm, en jornada continua, de esta manera nuestro servicio está disponible en un horario que se ajusta a cada uno de los clientes.

Política de Servicio

En SUPERMERCADOS YUCATÁ, los trabajadores no sólo tienen que ser expertos en el oficio que realizan, sino que es importante que lo ejecuten con la mayor cortesía y amabilidad, lo cual es parte de su misión.

Los clientes satisfechos son un patrimonio cuyo valor y lealtad son incalculables y crecen con el tiempo. Son los inspiradores de nuestra organización y nuestro trabajo; por lo tanto, deben recibir un trato excelente, para que consideren a SUPERMERCADOS YUCATÁ una organización amable, servidora, ágil y eficiente, que cumple con sus expectativas de servicio. La satisfacción de las necesidades del cliente es la mejor recompensa a nuestro esfuerzo, dedicación y capacitación.

Misión. Calidad en el servicio. Somos una de las compañías líderes en San Juan de Pasto dedicada a la comercialización de productos de consumo masivo de óptima calidad, a través de sucursales que se encuentran ubicadas en puntos estratégicos de la ciudad, orientadas a satisfacer las necesidades y deseos de la comunidad, ofreciendo un buen servicio y los mejores precios, con el respaldo de un talento humano comprometido e integralmente capacitado y con la confianza de nuestros proveedores, procurando una adecuada rentabilidad.

Visión. Llegar más lejos. Gracias a la confianza y a la positiva respuesta de nuestros clientes, SUPERMERCADOS YUCATÁ visiona expandirse por todo el territorio nariñense para llegar a cada uno de los hogares que buscan productos de calidad y precios bajos, contando para esto con la más alta tecnología.

¡Porque Somos Nariñenses!

Principios

Lealtad: Es el respeto a la convicciones, ideales y creencias.

Honestidad: Es actuar con integridad, con transparencia, con base en principios y valores.

Respeto: Es reconocer y aceptar nuestras diferencias

Valores. Trabajo en Equipo: Es crear un compromiso compartido con ideas, valores y metas comunes que nos permita reflejar unidad y equilibrio corporativo.
Creatividad e Innovación: Es la capacidad de generar ideas completamente nuevas e imaginativas que sirvan para obtener resultados beneficios.

Aprendizaje: Es la disposición para asimilar y poner en práctica conceptos nuevos.
Actitud de Servicio: Es la disposición de satisfacer las verdaderas necesidades del cliente, respondiendo a sus requerimientos en forma afectuosa y oportuna.

Nuestros Productos

SUPERMERCADOS YUCATÁ fomenta la cultura del autoservicio, permitiendo que los clientes adquieran los productos a su elección, sin necesidad de estar preguntando los precios; ya que el cliente sólo tiene que tomar el producto que necesita, acercarse a la caja registradora y cancelar el valor del producto requerido.

Para mayor facilidad de ubicación, los productos se encuentran clasificados en las siguientes secciones:

- Perfumería y Cosméticos
- Víveres
- Frutas y Vegetales
- Charcutería
- Lácteos
- Carnicería
- Panadería
- Área de juguetes
- Artículos para el hogar
- Electrodomésticos
- Artículos de limpieza
- Aseo Personal
- Librería y papelería
- Licores
- Variedades

Programas de Lealtad.

Para motivar a nuestros clientes a seguir utilizando los servicios que Supermercados YUCATA ofrece y premiar su fidelidad, manejamos tres programas de fidelización básicos, los cuales son:

Tarjeta de Puntos: El cliente puede adquirirla de manera gratuita, acercándose a la sección de Atención al Cliente dándonos sus datos básicos. Por cada mil pesos en compras realizadas el cliente gana un punto. Si no lleva la tarjeta de puntos, puede acumular sus puntos, dando su número de identificación. Posteriormente podrá hacer cambio de los puntos acumulados por los productos que se estén ofreciendo en el momento.

Mercado Especial: Este servicio está pensado principalmente en los negocios de comida, o familias que desean armar su propio mercado fijo para que Supermercados YUCATA se lo lleve a la puerta de su casa o negocio, en un tiempo comprendido entre uno o dos meses según se haya acordado con el cliente. Ofreciendo de esta manera el servicio de domicilio y un descuento del 10% en el total de su mercado.

Lunes Frutero: Todos los lunes, nuestros clientes podrán adquirir frutas con un 10% y hasta 15% de descuento.

Además durante todo el año, los clientes de SUPERMERCADOS YUCATÁ podrán disfrutar y aprovechar muchas otras promociones.

4.6.2. Descripción del sistema deseado. Se desea adecuar un sistema que controle el punto de pago, a través del cual se pueda ofrecer al cliente las siguientes formas de pago: tarjeta de crédito y débito con las respectivas validaciones de la entidad financiera correspondiente, efectivo, bonos, sistema de separado y pago con moneda extranjera. Se necesita que el sistema controle todos los posibles cambios durante la transacción, como cambios y devolución de productos. Si el cliente desea conocer el precio de algún producto lo puede hacer por medio de la caja registradora.

Cuando los clientes van a utilizar el sistema de separado, la empresa descuenta la mercancía separada de inventario, el saldo que queda pendiente en contabilidad se lo maneja como cuentas por cobrar. Entonces se necesita seguir controlando este aspecto por medio del sistema. Por otro lado, cuando los clientes compran los bonos que ofrecemos, el valor de cada bono se aumenta a la parte de contabilidad que corresponde a cuentas por pagar. Los bonos están exentos del IVA.

Si el cajero realiza un retiro o ingreso de dinero de la caja, se debe registrar esta situación.

Se necesita manejar sistemas de puntos por venta para los clientes, identificación de productos en promoción, descuento de porcentaje por valor de compra y control del domicilio para nuestro servicio “Mercado Especial”.

Cuando se registra una venta, se debe tener en cuenta esta información para hacer la actualización de inventario, cartera para el control de mercancía separada, contabilidad, realizar reportes de clientes para realizar estrategias de mercadeo, reportes de proveedores para saber quién ofrece el mejor servicio, informe de metas de ventas mensual.

Cuando se presentan ventas de cantidades significativas y si en la sucursal donde se va a realizar la venta, no hay existencia suficiente, a través del punto de venta

se podrá observar qué sucursal dispone de las existencias faltantes para realizar la venta.

La sede principal de nuestra empresa debe tener acceso a la información sobre ventas de las demás sucursales para poder realizar informes comparativos.

El punto en cada venta va a tener información actualizada conforme se realice algún cambio en cualquier otra dependencia, como por ejemplo la actualización de precios.
En el momento en que el cliente realice la compra, de los productos que tengan fecha de vencimiento, el sistema debe determinar si esta fecha se encuentra dentro del límite.

Al inicio de cada jornada de trabajo, para que el cajero pueda abrir caja, el sistema debe pedirle una contraseña. Al finalizar la jornada laboral, terminado el cierre de caja, el sistema genera una copia de seguridad de las transacciones realizadas durante el día en cada punto de venta.

Dado el gran número de transacciones diarias se necesita que el sistema brinde un alto grado de agilidad, y no presente problemas cuando haya gran afluencia de compradores, además el sistema debe ser de fácil uso para todas las personas que lo manejen.

Para que los cajeros puedan trabajar con mayor agilidad y brindar un mejor servicio a los clientes, nos gustaría que conserven el lector de código de barras, que a su vez sirve como balanza para las frutas y verduras y también que manejen el sistema a través de las pantallas táctiles, necesitamos que en las facturas además de encontrarse la información que es requerida por ley, debe tener datos como el punto de venta en que fue atendido el cliente además el nombre del cajero que se encontraba realizando la venta, hora, fecha, y si es cliente fiel debe salir en la factura nombre, cedula, puntos por la compra realizada y los puntos acumulados hasta esa fecha.

El sistema debe tener la autonomía de deshacer las últimas operaciones no llevadas a cabo con éxito para de esta forma conservar la integridad de los datos y no llenar de datos inútiles o erróneos en el sistema, debe realizar copias de seguridad dado los posibles escenarios de riesgo a una falla eléctrica o de software. Estas se deben realizar en el menor tiempo posible para que el sistema se pueda recuperar de tal forma que se puedan mitigar pérdidas significativas para la empresa. Además el sistema debe contar con un respaldo eléctrico, a posibles cortos, bajas de energía y posible ausencia de ella.

5. INGENIERIA DE REQUISITOS APLICADA AL CASO DE ESTUDIO SUPERMERCADOS YUCATA

5.1 INTRODUCCIÓN

En este capítulo se presenta la aplicación de la Ingeniería de Requisitos de acuerdo al modelo propuesto por Amador Durán presentado en la sección 8 del capítulo 3 de este documento; inicialmente se presenta el proceso de negociación de requisitos o elicitación de requisitos, luego se realiza el análisis de los requisitos definidos en la etapa de negociación y finalmente se presenta el proceso de validación de los requisitos del caso de estudio que definirán el comportamiento del Sistema POS para Supermercados YUCATA.

5.2 ELICITACION / NEGOCIACION DE REQUISITOS

Esta es la primera actividad propuesta por Durán para el proceso de Ingeniería de Requisitos. Para el desarrollo de este proceso de negociación se plantean tres tareas. A continuación se describe el desarrollo de cada una de estas tareas en el caso de estudio Supermercados YUCATA.

5.2.1 Obtener información sobre el dominio del problema y del sistema actual. En el caso de Supermercados YUCATA, no se cuenta hasta el momento con un aplicativo software, por lo tanto no es posible realizar un análisis del sistema actual. En cuanto al estudio del dominio del problema, en el capítulo anterior se hizo una descripción de sus conceptos básicos, tipos de POS, sistemas POS existentes, su clasificación y las funcionalidades que un sistema POS brinda.

5.2.2 Preparar y realizar las sesiones de Elicitación /Negociación. Con base en el documento entregado del Sistema deseado, se realizaron entrevistas y reuniones tipo JAD, a través de las cuales se obtuvieron los requisitos refinados presentados más adelante. Cabe aclarar que para cumplir los objetivos del proyecto de investigación, se hace necesario únicamente trabajar con los requisitos funcionales, pues son éstos los que se van a modelar a través de los patrones de análisis. Sin embargo, de manera adicional, en el proceso de negociación se incluyeron los requisitos no funcionales.

A continuación se presentan las propuestas iniciales de la descripción técnica de tres requisitos funcionales y su proceso de negociación/elicitación con el cliente hasta obtener los requisitos finales presentados al final de esta sección.

FRQ - 001. Durante la transacción, el sistema debe identificar y registrar los productos implicados.

S. YUCATA: No está claro a través de qué medios se va a poder hacer el registro de los productos. Además no se especifica en donde se va a ir guardando la información de la venta. En la reunión del 9 de Dic de 2009 se había hablado del manejo de una factura temporal en memoria.

G. LIS: Se corrige. Se aclara que la lectura de los productos se hará a través del código de barras y que esta información se irá registrando en la factura temporal.

FRQ - 001. Durante la transacción, el sistema debe identificar y registrar en la factura temporal los productos implicados a través del lector de código de barras.

S. YUCATA: Qué pasa si el lector de código no funciona? Se tiene que garantizar poder registrar todos los productos que el cliente vaya a llevar.
G. LIS: Se aclara cómo se subsana la situación en la que el lector de código de barras no funciona.

FRQ - 001. Durante la transacción, el sistema debe identificar y registrar en la factura temporal los productos implicados a través del lector de código de barras.
Si el lector de código de barras no hace la lectura del código del producto, el sistema debe permitir el ingreso del código de manera manual.

S. YUCATA: APROBADO. Diciembre 14 de 2009

FRQ - 002. El sistema identificará como formas de pago válidas:
Pago en efectivo
Pago con moneda extranjera
Pago con bonos, caso en el cual el sistema valida la vigencia del bono
Pago con tarjeta crédito, con validación a la entidad financiera correspondiente
Pago con tarjeta débito, con validación a la entidad financiera correspondiente
Pago mixto, en el cual el cliente podrá utilizar cualquier combinación de las anteriores formas de pago

S. YUCATA: No se describe explícitamente qué operaciones realiza el sistema en cada una de las formas de pago.

G. LIS: Se amplía la descripción de las operaciones realizadas por el sistema para cada forma de pago.

FRQ - 002. El sistema deberá validar la forma de pago y realizar las operaciones correspondientes:

Pago en efectivo, en este caso el sistema registra en la factura temporal el valor cancelado y hace el cálculo del monto de regreso al cliente.

Pago con moneda extranjera, en el cual el sistema cuenta con una actualización diaria de la TRM, el sistema calcula el monto de pago a registrar de acuerdo a la moneda escogida como forma de pago y registra en la factura temporal el valor cancelado.

Pago con bonos, caso en el cual el sistema valida la vigencia del bono y calcula si el valor del bono cubre de forma parcial o total el monto a cancelar. Si el valor del bono es menor indica el valor faltante. Si el valor del bono es mayor que el valor a cancelar, el cliente perderá el valor del excedente.

Pago con tarjeta crédito, el sistema establece la comunicación con la entidad financiera y hace la consulta si existe un saldo suficiente para la operación, si existe entonces el sistema genera el recibo correspondiente por dicha operación.
Pago con tarjeta débito, el sistema establece la comunicación con la entidad financiera y hace la consulta si existe un saldo suficiente para la operación, si existe entonces el sistema genera el recibo correspondiente por dicha operación.
Pago mixto, en el cual el cliente podrá utilizar cualquier combinación de las anteriores formas de pago.

Para todas las formas de pago el sistema valida que el valor registrado sea mayor o igual al valor de la venta. Si el valor cancelado es menor indica el valor faltante. Para los pagos en efectivo, si el valor cancelado es mayor que el valor a cancelar, el sistema hace el cálculo y genera el valor de regreso al cliente. Para las otras formas de pago no se hará retorno del excedente al cliente.

S. YUCATA: AROBADO. Diciembre 15 de 2009

R.0.3. El sistema debe realizar el control de posibles cambios durante la transacción, tales como:
Devoluciones de 1 ó más productos
Cancelación de la transacción

S. YUCATA: Es necesario especificar cómo el sistema va a controlar cada cambio de acuerdo al momento en el que ocurra durante la venta. Por ejemplo: Si una devolución se realiza antes de que se haya cancelado el pago, el sistema deberá realizar operaciones diferentes a las que debe realizar si la devolución se hace una vez que se imprima la factura.

G. LIS: Se amplía la descripción del requisito definiendo tres tiempos en los que puede ocurrir el cambio, así: Tiempo 1: El cambio se realiza después de ser identificado y enlistado en la factura temporal, antes de efectuar el pago. Tiempo 2: El cambio se realiza después del pago del total de la factura. Tiempo 3: El

cambio se realiza después de ejecutar la venta (imprimir la factura), en este tiempo se hace, a través de la interfaces, el envío de información a los otros sistemas involucrados.

FRQ - 003. El sistema deberá realizar el control de posibles devoluciones durante la venta, de la siguiente manera:

Devoluciones de 1 o más productos:

Si la devolución del (los) producto(s) se realiza después de ser identificado y enlistado en la factura temporal, se procederá a borrarlo(s) de la lista y a actualizar el total de la factura.

Si la devolución del (los) producto(s) se realiza después del pago del total de la factura entonces se actualiza en la factura temporal la lista de productos y el valor total de la venta.

Si el pago se hizo en efectivo con moneda nacional y/o extranjera se reversa el registro de pago por el valor correspondiente al (los) producto(s) devuelto(s) y se realiza la devolución del dinero. Para las otras formas de pago se emitirá un bono con el valor correspondiente, del cual se hace registro en la factura como un producto más de compra.

Si la devolución del (los) producto(s) se realiza después de ejecutar la venta, el sistema solicita el número de unidades por tipo de producto y el código de barra del (los) producto(s) devuelto(s) y envía la información a los sistemas involucrados en la operación inicial para reversar los registros efectuados correspondientes al (los) producto(s) devuelto(s). El sistema emitirá un bono con el valor correspondiente, esto aplica para todas las formas de pago.

Cancelación de la transacción:

Si la cancelación de la transacción se realiza antes de efectuar el pago, el sistema deberá eliminar la factura temporal.

Si la cancelación de la transacción se realiza después de efectuar el pago: si el pago se realiza en efectivo con moneda nacional o extranjera el sistema elimina la factura temporal y se retorna al cliente el valor cancelado. Si se registraron pagos con otro medio, se deberá eliminar los productos de la factura temporal y registrar un producto tipo Bono por el valor del pago realizado. Se imprime el bono.

Si la cancelación de la transacción se realiza después de ejecutar la venta, el sistema hace el registro de la anulación de la factura por concepto de cancelación, envía la información a los sistemas involucrados en la operación inicial para

reversar la transacción y emite un bono por valor igual al total de la venta, esto aplica para todas las formas de pago.

S. YUCATA: Se aprueba el manejo de los tiempos y las operaciones descritas. Falta incluir la actividad Adición de Productos.

G. LIS: La adición de productos es la actividad natural dentro del proceso de venta, no se debe tomar esta acción como un cambio. Además si la adición se hace en el tiempo 3 se consideraría como una nueva venta. Por lo tanto esta actividad no es tenida en cuenta en este requisito.

S. YUCATA: APROBADO. DICIEMBRE 19 DE 2009

5.2.3 Identificar/revisar los objetivos del sistema. Para la identificación de los objetivos del sistema se recurre a la plantilla para objetivos propuesta en este modelo, pues este elemento de representación ayuda a tener una definición más clara y detallada de los objetivos y sub-objetivos del sistema

Tabla 13: Objetivo 001 ejecución de venta

OBJ – 001	Ejecución de Venta
Versión	1.1
Autores	Supermercado YUCATA – Tania Araújo, Yurani Bolaños
Fuente	Supermercado YUCATA
Descripción	El sistema debe permitir registrar y ejecutar las ventas.
SubObjetivos	<OBJ – 1.1> El sistema debe realizar la identificación y registro de los productos involucrados en la venta. <OBJ – 1.2> El sistema debe permitir registrar el pago realizado por el cliente. <OBJ – 1.3> El sistema debe permitir registrar y ejecutar la venta.
Importancia	Vital
Urgencia	Alta
Estado	Validado
Estabilidad	Alta
Comentarios	-

Tabla 14: Objetivo 002 comodidad y calidad en el servicio

OBJ – 002	Comodidad y Calidad en el Servicio
Versión	1.0
Autores	Supermercado YUCATA – Tania Araújo, Yurani Bolaños
Fuente	Supermercado YUCATA
Descripción	El sistema debe permitir realizar las operaciones involucradas en el proceso de venta, de manera sencilla y rápida.
SubObjetivos	<OBJ – 2.1> El sistema debe permitir al cajero realizar el ingreso de la información requerida para el registro y ejecución de la venta, de manera ágil y sencilla a través de una interfaz amigable. <OBJ – 2.2> El sistema debe permitir controlar posibles cambios que el cliente puede realizar durante su proceso de compra de manera ágil y sencilla. <OBJ – 2.3> El sistema debe optimizar los tiempos empleados en el ingreso de información al sistema.
Importancia	Alta
Urgencia	Media
Estado	Validado
Estabilidad	Media
Comentarios	-

Tabla 15: Objetivo 003 coherencias de información

OBJ – 003	Coherencia de Información
Versión	1.0
Autores	Supermercado YUCATA – Tania Araújo, Yurani Bolaños
Fuente	Supermercado YUCATA
Descripción	El sistema debe permitir obtener información confiable, coherente y actualizada del Sistema POS para los procesos diferentes procesos involucrados en el proceso de venta.
SubObjetivos	-
Importancia	Vital
Urgencia	Media
Estado	Validado
Estabilidad	Media
Comentarios	-

Tabla 16: Objetivo 004 seguridades de información

OBJ - 004	Seguridad de Información
Versión	1.0
Autores	Supermercado YUCATA – Tania Araújo, Yurani Bolaños
Fuente	Supermercado YUCATA
Descripción	El sistema debe permitir restringir el acceso al sistema POS de acuerdo al rol del usuario.
SubObjetivos	-
Importancia	Alta
Urgencia	Media
Estado	Validado
Estabilidad	Media
Comentarios	-

Tabla 17: Objetivo 005 intercambio de información con sistemas externos

OBJ - 005	Intercambio de Información con Sistemas Externos
Versión	1.0
Autores	Supermercado YUCATA – Tania Araújo, Yurani Bolaños
Fuente	Supermercado YUCATA
Descripción	El sistema debe permitir intercambiar información con sistemas externos que se pueden ver involucrados en el proceso de venta.
SubObjetivos	<OBJ – 5.1> El sistema debe permitir intercambiar entre la sede principal y sucursales de Supermercados YUCATA <OBJ – 5.2> El sistema debe permitir establecer comunicación con los sistemas de las Entidades Bancarias Asociadas. <OBJ – 5.3> El sistema debe permitir enviar información a los sistemas de contabilidad, inventario y mercadeo.
Importancia	Alta
Urgencia	Media
Estado	Validado
Estabilidad	Media
Comentarios	-

5.2.4 Producto del proceso de negociación: descripción de requisitos. Después del proceso de negociación donde se trató cada uno de los requisitos, el resultado obtenido se presenta a continuación, donde se describen los requisitos finales aprobados por el Cliente Supermercados YUCATA.

Requisitos Funcionales

FRQ - 001. Durante la transacción, el sistema debe identificar y registrar en la factura temporal los productos implicados a través del lector de código de barras. Si el lector de código de barras no hace la lectura del código del producto, el sistema debe permitir el ingreso del código de manera manual.

FRQ - 002. El sistema deberá validar la forma de pago y realizar las operaciones correspondientes:

Pago en efectivo, en este caso el sistema registra en la factura temporal el valor cancelado y hace el cálculo del monto de regreso al cliente.

Pago con moneda extranjera, en el cual el sistema cuenta con una actualización diaria de la TRM, el sistema calcula el monto a cancelar de acuerdo a la moneda escogida como forma de pago, registra en la factura temporal el valor cancelado y genera el valor de regreso al cliente.

Pago con bonos, caso en el cual el sistema valida la vigencia del bono y calcula si el valor del bono cubre de forma parcial o total el monto a cancelar. Si el valor del bono es menor indica el valor faltante. Si el valor del bono es mayor que el valor a cancelar, el cliente perderá el valor del excedente.

Pago con tarjeta crédito, el sistema establece la comunicación con la entidad financiera y hace la consulta si existe un saldo suficiente para la operación, si existe entonces el sistema genera el recibo correspondiente por dicha operación.
Pago con tarjeta débito, el sistema establece la comunicación con la entidad financiera y hace la consulta si existe un saldo suficiente para la operación, si existe entonces el sistema genera el recibo correspondiente por dicha operación.
Pago mixto, en el cual el cliente podrá utilizar cualquier combinación de las anteriores formas de pago

FRQ - 003. El sistema deberá realizar el control de posibles devoluciones durante la venta, de la siguiente manera:

Devoluciones de 1 o más productos:

Si la devolución del (los) producto(s) se realiza después de ser identificado y enlistado en la factura temporal, se procederá a borrarlo(s) de la lista y a actualizar el total de la factura.

Si la devolución del (los) producto(s) se realiza después del pago del total de la factura entonces se actualiza en la factura temporal la lista de productos y el valor total de la venta.

Si el pago se hizo en efectivo con moneda nacional y/o extranjera se reversa el registro de pago por el valor correspondiente al (los) producto(s) devuelto(s) y se realiza la devolución del dinero. Para las otras formas de pago se emitirá un bono con el valor correspondiente, del cual se hace registro en la factura como un producto más de compra.

Si la devolución del (los) producto(s) se realiza después de ejecutar la venta (imprimir la factura), el sistema solicita el número de unidades por tipo de producto y el código de barra del (los) producto(s) devuelto(s) y envía la información a los sistemas involucrados en la operación inicial para reversar los registros efectuados correspondientes al (los) producto(s) devuelto(s). El sistema emitirá un bono con el valor correspondiente, esto aplica para todas las formas de pago.

Cancelación de la transacción:

Si la cancelación de la transacción se realiza antes de efectuar el pago, el sistema deberá eliminar la factura temporal.

Si la cancelación de la transacción se realiza después de efectuar el pago: si el pago se realiza en efectivo con moneda nacional o extranjera el sistema elimina la factura temporal y se retorna al cliente el valor cancelado. Si se registraron pagos con otro medio, se deberá eliminar los productos de la factura temporal y registrar un producto tipo Bono por el valor del pago realizado. Se imprime el bono.

Si la cancelación de la transacción se realiza después de ejecutar la venta, el sistema hace el registro de la anulación de la factura por concepto de cancelación, envía la información a los sistemas involucrados en la operación inicial para reversar la transacción y emite un bono por valor igual al total de la venta, esto aplica para todas las formas de pago.

FRQ – 004. Una vez efectuado el pago, el sistema deberá imprimir y hacer persistencia de la factura, la cual contiene la siguiente información:

Cabecera: Compuesta por el código de la factura, el NIT de la empresa YUCATA, identificación y nombre del cajero, número de caja, fecha y hora de atención.

Productos: Código y nombre del producto, cantidad, si el producto es una fruta o verdura se indicará el peso en gramos, valor unitario o valor por libra para el caso de frutas o verduras, valor total que corresponde al valor unitario por la cantidad de unidades compradas, para el caso de frutas y verduras se indicará el valor correspondiente al peso del producto en gramos por el valor por libra.

Subtotal: Sumatoria de los valores totales listados en la parte de Productos
IVA: Se discrimina el valor cobrado por concepto IVA
Total: Sumatoria del Subtotal más valor IVA.

Pago: Se indica la forma y valor de pago, valor de cambio, total artículos comprados.

Cliente [Opcional]: Si el cliente ha hecho registro de sus datos personales en el punto de atención al cliente y pertenece al programa de puntos se mostrará el número de identificación y el nombre del cliente y la cantidad de puntos por la compra y puntos acumulados.

Información Adicional [Opcional]: Si existen promociones de temporada, la información correspondiente a dichas promociones se podrán mostrar en esta sección.

Si el pago corresponde a un abono a mercancía separada, la factura mostrará el estado actualizado del producto separado.

FRQ - 005. Una vez finalizada la transacción, el sistema debe ejecutar la venta. El sistema enviará la información que corresponda a cada interfaz, así:

GESTIÓN INTERFAZ INVENTARIO

Envío de información de la cantidad de productos para actualizar el inventario.
Envío de información de la cantidad de productos a restablecer en los casos de devolución y cancelación de venta.

GESTIÓN INTERFAZ CONTABILIDAD

Envío de información de cartera para actualizar los abonos realizados por los clientes, en el caso del manejo del sistema de separado.

Envío de información referente a entradas y salidas eventuales para la generación de informes de contabilidad.

Envío de información de pagos con bonos, para que sean registrados en cuentas por pagar.

GESTIÓN INTERFAZ MERCADEO

Envío de información de clientes y sus respectivas compras para poder generar estrategias de mercadeo y programas de fidelización.

Envío de información de ventas para la generación de reportes de proveedores y metas de venta general.

FRQ – 006. GESTION DE PROMOCIONES Y SERVICIOS

Identificar productos en promoción para aplicar el descuento correspondiente
El sistema debe identificar en una transacción el producto “Mercado Especial”
Actualizar puntos por venta de los clientes que utilicen este servicio.

El sistema debe permitir configurar las promociones, programando su fecha y hora de inicio y finalización, productos a los que se aplica el descuento y el porcentaje de descuento a aplicar.

FRQ – 007. GESTION DE INTERCOMUNICACIÓN

El sistema podrá comunicarse con otras sucursales

Si la mercancía existente en un punto no satisface una venta significativa, el sistema visualiza el punto de venta de otra sucursal que puede cubrir la mercancía faltante.

La sede principal tendrá acceso a la información referente a las ventas de las demás sucursales.

El sistema podrá comunicarse con las entidades bancarias aliadas para realizar la validación para las situaciones de pago con tarjeta de crédito y débito.

FRQ - 008. El punto de venta tendrá acceso a la información coherente, actualizada y confiable para cada transacción o consulta:

Precios actualizados de productos, cantidades e información básica.

Aviso de fecha de vencimiento de los productos.

Cambios de impuesto.

Información básica del cliente, nombre cedula, teléfono, celular.

Información del cajero asignado al punto de venta.

Requisitos No Funcionales

NFR – 001. GESTION DE SEGURIDAD FISICA Y LOGICA DE INFORMACION

El sistema realizará un rollback de los procesos no realizados con éxito después de cualquier falla física o lógica que pueda vulnerar la integridad de los datos.

El sistema creará un backup interno en períodos de 5 horas, esta copia de seguridad contendrá todas las transacciones realizadas en el POS en ese espacio de tiempo, esta copia de respaldo se hace para preservar la información ante cualquier falla física o lógica.

El sistema creará un backup externo en medio magnético el cual se conservará en un lugar distinto a la empresa, este contendrá las transacciones del POS de una semana, esta copia de respaldo se hace para preservar la información ante cualquier falla física o lógica.

Cada punto de venta contendrá una UPS, como respaldo eléctrico para las posibles fallas en el suministro eléctrico.

NFR – 002. GESTION DE SEGURIDAD Y CONTROL

El cajero debe ingresar su contraseña e id asignado para la apertura de caja.

El sistema se registraría el retiro o ingreso excepcional de dinero a la caja.

NFR – 003. GESTION DE CALIDAD

El sistema debe tener una interfaz amigable a los usuarios y de fácil manejo.

El sistema debe brindar el soporte necesario para funcionar de manera ágil y eficiente en situaciones de gran afluencia de compradores

NFR – 004. REQUISITOS FÍSICOS

Las ventas se registraran por medio del lector de código de barras que a su vez servirá de balanza para las ventas de productos facturables de acuerdo al peso.

El manejo del sistema se realizará por medio de pantallas táctiles.

El punto de venta debe contar con el uso de lectores de tarjetas crédito y débito.

5.3 ANÁLISIS DE REQUISITOS:

En esta actividad se analizan nuevamente cada uno de los requisitos, buscando garantizar que si existe algún conflicto por resolver en cuanto a la definición de los requisitos sea aclarado y concensuado con el cliente en esta fase del proceso de ingeniería de requisitos.

Para este caso de estudio, el análisis no arrojó nuevos conflictos y persisten las descripciones de los requisitos definidos en la fase de elicitación. Para un mejor análisis de cada uno de los requisitos se hizo uso de la plantilla de descripción de

requisitos propuesta por TRIPODE, la cual se basa en la plantilla de requisitos propuesta por Amador Durán, que permite analizar cómo cada requisito se relaciona con los objetivos del sistema, y ponderar su importancia, estado y estabilidad.

A continuación se muestran los requisitos descritos en la plantilla de requisitos de TRIPODE.

5.3.1. Análisis de requisitos funcionales

Tabla 18: Requisito funcional 001 – identificación y registro de los productos

FRQ-001	**Identificación y Registro de los Productos**
Versión	1.0
Autores	Tania Araujo, Yurani Bolaños
Fuentes	Cajero y Supervisor de Caja Supermercado YUCATA.
Objetivos asociados	OBJ – 001
Requisitos asociados	FRQ-002 Formas de Pago Validas FRQ-004 Gestionar Factura FRQ-008 Mantener disponible información válida.
Descripción	El sistema debe identificar y registrar en la factura temporal los productos implicados. A través del lector de código de barras. Si el lector de código de barras no hace la lectura del código del producto, el sistema debe permitir el ingreso del código de manera manual.
Dominios asociados	POS Comercio
Aspectual	-
Importancia	Vital
Estado	Validado
Estabilidad	Media
Comentarios	Ninguno

Tabla 19: Requisito funcional 002 – formas de pago válidas

FRQ-002	Formas de Pago Válidas
Versión	1.3
Autores	Tania Araujo, Yurani Bolaños
Fuentes	Supervisor de Caja Supermercado YUCATA
Objetivos asociados	OBJ - 001 OBJ – 005
Requisitos asociados	FRQ-001 Identificación y registro de productos FRQ-004 Gestionar Factura FRQ-008 Mantener disponible información válida.
Descripción	El sistema deberá validar la forma de pago y realizar las operaciones correspondientes: Pago en efectivo, en este caso el sistema registra en la factura temporal el valor cancelado y hace el cálculo del monto de regreso al cliente. Pago con moneda extranjera, en el cual el sistema cuenta con una actualización diaria de la TRM, el sistema calcula el monto a cancelar de acuerdo a la moneda escogida como forma de pago, registra en la factura temporal el valor cancelado y genera el valor de regreso al cliente. Pago con bonos, caso en el cual el sistema valida la vigencia del bono y calcula si el valor del bono cubre de forma parcial o total el monto a cancelar. Si el valor del bono es menor indica el valor faltante. Si el valor del bono es mayor que el valor a cancelar, el cliente perderá el valor del excedente. Pago con tarjeta crédito, el sistema establece la comunicación con la entidad financiera y hace la consulta si existe un saldo suficiente para la operación, si existe entonces el sistema genera el recibo correspondiente por dicha operación. Pago con tarjeta débito, el sistema establece la comunicación con la entidad financiera y hace la consulta si existe un saldo suficiente para la operación, si existe entonces el sistema genera el recibo correspondiente por dicha operación. Pago mixto, en el cual el cliente podrá utilizar cualquier combinación de las anteriores formas de pago.
Dominios asociados	POS Comercio
Aspectual	-
Importancia	Vital
Estado	Validado
Estabilidad	Media
Comentarios	Ninguno

Tabla 20: Requisito funcional 003 – control de devoluciones

FRQ-003	Control de Devoluciones
Versión	1.3
Autores	Yurani Bolaños, Tania Araújo
Fuentes	Supervisor de Caja Supermercado YUCATA
Objetivos asociados	OBJ - 001 OBJ – 002
Requisitos asociados	FRQ-001 Identificación y registro de productos FRQ-008 Mantener disponible información válida.
Descripción	El sistema deberá realizar el control de posibles devoluciones durante la venta, de la siguiente manera: Devoluciones de 1 o más productos: Si la devolución del (los) producto(s) se realiza después de ser identificado y enlistado en la factura temporal, se procederá a borrarlo(s) de la lista y a actualizar el total de la factura. Si la devolución del (los) producto(s) se realiza después del pago del total de la factura entonces se actualiza en la factura temporal la lista de productos y el valor total de la venta. Si el pago se hizo en efectivo con moneda nacional y/o extranjera se reversa el registro de pago por el valor correspondiente al (los) producto(s) devuelto(s) y se realiza la devolución del dinero. Para las otras formas de pago se emitirá un bono con el valor correspondiente, del cual se hace registro en la factura como un producto más de compra. Si la devolución del (los) producto(s) se realiza después de ejecutar la venta (imprimir la factura), el sistema solicita el número de unidades por tipo de producto y el código de barra del (los) producto(s) devuelto(s) y envía la información a los sistemas involucrados en la operación inicial para reversar los registros efectuados correspondientes al (los) producto(s) devuelto(s). El sistema emitirá un bono con el valor correspondiente, esto aplica para todas las formas de pago. Cancelación de la transacción: Si la cancelación de la transacción se realiza antes de efectuar el pago, el sistema deberá eliminar la factura temporal. Si la cancelación de la transacción se realiza después de efectuar el pago: si el pago se realiza en efectivo con moneda nacional o extranjera el sistema elimina la factura temporal y se retorna al cliente el valor cancelado. Si se registraron pagos con otro medio, se deberá eliminar los productos de la factura temporal y registrar un producto tipo Bono por el valor del pago realizado. Se imprime el bono. Si la cancelación de la transacción se realiza después de ejecutar la venta, el sistema hace el registro de la anulación de la factura por concepto de cancelación, envía la información a los sistemas involucrados en la operación inicial para reversar la transacción y emite un bono por valor igual al total de la venta, esto aplica para todas las formas de pago.
Dominios asociados	POS Comercio
Aspectual	-
Importancia	Vital
Estado	Validado
Estabilidad	Alta
Comentarios	Ninguno

Tabla 21: Requisito funcional 004 – gestionar factura

FRQ-004	Gestionar Factura
Versión	1.1
Autores	Tania Araújo, Yurani Bolaños
Fuentes	Supervisor de Caja Supermercado YUCATA
Objetivos asociados	OBJ – 001
Requisitos asociados	FRQ-001 Identificación y registro de productos FRQ-002 Identificación de formas válidas de pago FRQ-005 Gestionar Interfaces FRQ-008 Mantener disponible información válida.
Descripción	Una vez efectuado el pago, el sistema deberá hacer persistencia de la factura, la cual contiene la siguiente información: Cabecera: Compuesta por el código de la factura, el NIT de la empresa YUCATA, identificación y nombre del cajero, número de caja, fecha y hora de atención. Productos: Nombre del producto, cantidad, si el producto es una fruta o verdura se indicará el peso en gramos, valor unitario o valor por libra para el caso de frutas o verduras, valor total que corresponde al valor unitario por la cantidad de unidades compradas, para el caso de frutas y verduras se indicará el valor correspondiente al peso del producto en gramos por el valor por libra. Subtotal: Sumatoria de los valores totales listados en la parte de Productos IVA: Se discrimina el valor cobrado por concepto IVA Total: Sumatoria del Subtotal más valor IVA. Pago: Se indica la forma y valor de pago, valor de cambio, total artículos comprados. Cliente [Opcional]: Si el cliente ha hecho registro de sus datos personales en el punto de atención al cliente y pertenece al programa de puntos se mostrará el número de identificación y el nombre del cliente y la cantidad de puntos por la compra y puntos acumulados. Información Adicional [Opcional]: Si existen promociones de temporada, la información correspondiente a dichas promociones se podrán mostrar en esta sección.
Dominios asociados	POS Comercio
Aspectual	-
Importancia	Vital
Estado	Validado
Estabilidad	Media
Comentarios	Ninguno

Tabla 22: Requisito funcional 005 – gestionar interfaces

FRQ – 005	Gestionar Interfaces
Versión	1.1
Autores	Tania Araújo, Yurani Bolaños
Fuentes	Supervisor de Caja Supermercado YUCATA
Objetivos asociados	OBJ – 003 OBJ – 005
Requisitos asociados	FRQ – 004 Gestionar Factura FRQ-008 Mantener disponible información válida
Descripción	El sistema debe permitir enviar la información que corresponda a cada interfaz, así: GESTIÓN INTERFAZ INVENTARIO Envío de información de la cantidad de productos para actualizar el inventario. Envío de información de la cantidad de productos a restablecer en los casos de devolución y cancelación de venta. GESTIÓN INTERFAZ CONTABILIDAD Envío de información de cartera para actualizar los abonos realizados por los clientes, en el caso del manejo del sistema de separado. Envío de información referente a entradas y salidas eventuales para la generación de informes de contabilidad Envío de información de pagos con bonos, para que sean registrados en cuentas por pagar. GESTIÓN INTERFAZ MERCADEO Envío de información de clientes y sus respectivas compras para poder generar estrategias de mercadeo y programas de fidelización Envío de información de ventas para la generación de reportes de proveedores y metas de venta general
Dominios asociados	General
Aspectual	-
Importancia	Vital
Estado	Validado
Estabilidad	Media
Comentarios	Ninguno

Tabla 23: Requisito funcional 006 – gestión de promociones y servicios

FRQ-006	Gestión de Promociones y Servicios
Versión	1.0
Autores	Tania Araujo, Yurani Bolaños
Fuentes	Cajeros del Supermercado YUCATA.
Objetivos asociados	OBJ – 001 OBJ – 003
Requisitos asociados	FRQ-001 Identificación y registro de los productos FRQ-004 Gestionar Factura. FRQ-008 Mantener disponible información válida.
Descripción	Identificar productos en promoción para aplicar el descuento correspondiente El sistema debe identificar en una transacción el producto "Mercado Especial" Actualizar puntos por venta de los clientes que utilicen este servicio El sistema debe permitir configurar las promociones, programando su fecha y hora de inicio y finalización, productos a los que se aplica el descuento y el porcentaje de descuento a aplicar.
Dominios asociados	POS Comercio
Aspectual	-
Importancia	Vital
Estado	Validado
Estabilidad	Media
Comentarios	Ninguno

Tabla 24: Requisito funcional 007 – gestión de intercomunicación

FRQ-007	Gestión de Intercomunicación
Versión	1.0
Autores	Tania Araujo, Yurani Bolaños
Fuentes	Cajero, y Supervisor de Caja del Supermercado YUCATA.
Objetivos asociados	OBJ – 005
Requisitos asociados	FRQ-006 Gestión De Promociones Y Servicios FRQ-008 Mantener disponible información válida
Descripción	El sistema podrá comunicarse con otras sucursales Si la mercancía existente en un punto no satisface una venta significativa, el sistema visualiza el punto de venta de otra sucursal que puede cubrir la mercancía faltante La sede principal tendrá acceso a la información referente a las ventas de las demás sucursales El sistema podrá comunicarse con las entidades bancarias aliadas para realizar la validación para las situaciones de pago con tarjeta de crédito y debito.
Dominios asociados	POS Comercio
Aspectual	-
Importancia	Vital
Estado	Validado
Estabilidad	Media
Comentarios	Ninguno

Tabla 25: Requisito funcional 008 – mantener disponible información válida

FRQ-008	Mantener Disponible Información Válida
Versión	1.0
Autores	Tania Araujo, Yurani Bolaños
Fuentes	Cajero, y Supervisor de Caja del Supermercado YUCATA.
Objetivos asociados	OBJ – 003 OBJ – 005
Requisitos asociados	FRQ-001 Identificación y Registro de los productos FRQ-002 Forma de Pago Válidas. FRQ-003 Control de Devoluciones. FRQ-004 Gestionar Factura. FRQ-005 Gestionar Interfaces. FRQ-006 Gestión De Promociones Y Servicios FRQ-007 Gestión De Intercomunicación
Descripción	El punto de venta tendrá acceso a la información coherente, actualizada y confiable para cada transacción o consulta: Precios actualizados de productos, cantidades e información básica Aviso de fecha de vencimiento de los productos Cambios de impuesto Información básica del cliente, nombre cedula, teléfono, celular Información del cajero asignado al punto de venta
Dominios asociados	POS Comercio
Aspectual	-
Importancia	Vital
Estado	Validado
Estabilidad	Media
Comentarios	Ninguno

5.3.2. Análisis de requisitos no funcionales

Tabla 26: Requisito no funcional 001 – gestión de respaldo de información

NFR-001	Gestión de Respaldo de Información
Versión	1.1
Autores	Tania Araújo, Yurani Bolaños
Fuentes	Supervisor de Caja Supermercado YUCATA
Objetivos asociados	OBJ – 003
Requisitos asociados	FRQ-008 Mantener disponible información válida
Descripción	El sistema debe permitir programar la creación de: Un backup interno en períodos de 5 horas, esta copia de seguridad contendrá todas las transacciones realizadas en el POS en ese espacio de tiempo. Un backup externo (medio magnético) el cual se conservará en un lugar distinto a la empresa, este contendrá las transacciones del POS de una semana. Estas copias de respaldo se hacen para preservar la información ante cualquier falla física o lógica.
Dominios asociados	General
Aspectual	-
Importancia	Vital
Estado	Validado
Estabilidad	Media
Comentarios	Ninguno

Tabla 27: Requisito no funcional 002 – gestión de seguridad y control

NFR-002	GESTION DE SEGURIDAD Y CONTROL
Versión	1.1
Autores	Tania Araújo, Yurani Bolaños
Fuentes	Cajero y Supervisor de Caja Supermercado YUCATA
Objetivos asociados	OBJ – 004
Requisitos asociados	FRQ-008 Mantener disponible información válida
Descripción	El cajero debe ingresar su contraseña e id asignado para la apertura de caja. El sistema se registraría el retiro o ingreso excepcional de dinero a la caja.
Dominios asociados	General
Aspectual	-
Importancia	Vital
Estado	Validado
Estabilidad	Media
Comentarios	Ninguno

Tabla 28: Requisito no funcional 003 – gestión de calidad

NFR-003	GESTION DE CALIDAD
Versión	1.1
Autores	Tania Araújo, Yurani Bolaños
Fuentes	Cajero y Supervisor de Caja Supermercado YUCATA
Objetivos asociados	OBJ – 002
Requisitos asociados	FRQ-008 Mantener disponible información válida
Descripción	El sistema debe tener una interfaz amigable a los usuarios y de fácil manejo. El sistema debe brindar el soporte necesario para funcionar de manera ágil y eficiente en situaciones de gran afluencia de compradores.
Dominios asociados	General
Aspectual	-
Importancia	Vital
Estado	Validado
Estabilidad	Media
Comentarios	Ninguno

Tabla 29: Requisito no funcional 004 – requisitos físicos

NFR-004	REQUISITOS FÍSICOS
Versión	1.1
Autores	Tania Araújo, Yurani Bolaños
Fuentes	Cajero y Supervisor de Caja Supermercado YUCATA
Objetivos asociados	OBJ – 001 OBJ – 003
Requisitos asociados	FRQ-008 Mantener disponible información válida
Descripción	Las ventas se registraran por medio del lector de código de barras que a su vez servirá de balanza para las ventas de productos facturables de acuerdo al peso El manejo del sistema se realizará por medio de pantallas táctiles El punto de venta debe contar con el uso de lectores de tarjetas crédito y débito.
Dominios asociados	General
Aspectual	-
Importancia	Vital
Estado	Validado
Estabilidad	Media
Comentarios	Ninguno

5.4 VALIDACION DE REQUISITOS

El propósito de esta actividad es realizar una nueva revisión conjunta entre los desarrolladores y los clientes, para poder identificar nuevos conflictos y obtener como resultado una versión definitiva de los requisitos. Para el caso de Supermercados YUCATA, el proceso de validación no causó cambios en la definición de requisitos, por lo que se respeta la versión presentada en la fase de Análisis de Requisitos, en las que se puede verificar que el estado de cada requisito es Validado. Adicionalmente se construye la Matriz de Rastreabilidad de los Requisitos, mostrada a continuación

5.4.1 Matriz de rastreabilidad. A través de esta matriz se puede observar la relación de cada requisito con los objetivos del sistema y además se puede verificar que cada objetivo del sistema tiene asociado al menos un requisito, garantizando el cumplimiento de los objetivos del sistema en desarrollo.

Tabla 30: Matriz de rastreabilidad

	OBJ – 001	OBJ - 002	OBJ - 003	OBJ - 004	OBJ - 005
FRQ – 001	X				
FRQ – 002	X				X
FRQ – 003	X	X			
FRQ – 004	X				
FRQ – 005			X		X
FRQ – 006	X		X		
FRQ – 007					X
FRQ – 008			X		X
NFR – 001			X		
NFR – 002				X	
NFR – 003		X			
NFR – 004	X		X		

6 MODELADO DE UN SISTEMA DE PUNTO DE VENTA HACIENDO USO DE UN CATÁLOGO DE PATRONES DE ANÁLISIS BASADO EN MORENA

INTRODUCCIÓN

Este proyecto de investigación tiene como objetivo “Aplicar el catálogo de patrones de análisis basado en el Modelo de Representación de Patrones de Análisis - MORENA, en el análisis de un sistema POS”.

Para el logro de este objetivo se seguirá el metodología de Ingeniería de Aplicación propuesto en ESKEMA haciendo uso del catálogo de patrones de análisis TRIPODE.

Para el propósito de esta investigación se hará el modelado del sistema haciendo uso de requisitos principales del caso de estudio Supermercados YUCATA.

Para el modelado se utiliza la herramienta CASE Enterprise Architect que es una de las herramientas más robustas del mercado, además como se mostró en el capítulo 3 sección 11 de este documento, la única herramienta que soporta templates parametrizables es STARUML, esta herramienta CASE es de código abierto, sin embargo no provee el soporte para los diagramas de secuencia, que permite representar el modelo de comportamiento, esencial en el modelado de las soluciones específicas con los patrones de análisis de TRIPODE.

La parametrización de los templates se hace de manera simulada describiendo como se utilizan los parámetros de cada uno de los patrones utilizados.

Después del modelado del sistema POS, se realiza el proceso de retroalimentación del catálogo con un requisito del caso de estudio para el cual no existe un patrón dentro del catálogo TRIPODE que lo solucione. Finalmente se muestra un cuadro de resultados de cada uno de los procesos llevados a cabo en el proceso investigativo.

6.2 PROCESO DE USO DEL CATÁLOGO

6.2.1 Solución del requisito formas de pago válidas

- Caracterizar problemas por requisitos

Tabla 31. Plantilla de requisito específico formas de pago válidas

FRQ-002	Formas de Pago Válidas
Versión	1.3
Autores	Tania Araujo, Yurani Bolaños
Fuentes	Supervisor de Caja Supermercado YUCATA
Objetivos asociados	Llevar a cabo la venta
Requisitos asociados	FRQ-001 Identificación y registro de productos FRQ-004 Gestionar Factura
Descripción	El sistema deberá validar la forma de pago y realizar las operaciones correspondientes: Pago en efectivo, en este caso el sistema registra en la factura temporal el valor cancelado y hace el cálculo del monto de regreso al cliente. Pago con moneda extranjera, en el cual el sistema cuenta con una actualización diaria de la TRM, el sistema calcula el monto a cancelar de acuerdo a la moneda escogida como forma de pago, registra en la factura temporal el valor cancelado y genera el valor de regreso al cliente. Pago con bonos, caso en el cual el sistema valida la vigencia del bono y calcula si el valor del bono cubre de forma parcial o total el monto a cancelar. Si el valor del bono es menor indica el valor faltante. Si el valor del bono es mayor que el valor a cancelar, el cliente perderá el valor del excedente. Pago con tarjeta crédito, el sistema establece la comunicación con la entidad financiera y hace la consulta si existe un saldo suficiente para la operación, si existe entonces el sistema genera el recibo correspondiente por dicha operación. Pago con tarjeta débito, el sistema establece la comunicación con la entidad financiera y hace la consulta si existe un saldo suficiente para la operación, si existe entonces el sistema genera el recibo correspondiente por dicha operación. Pago mixto, en el cual el cliente podrá utilizar cualquier combinación de las anteriores formas de pago.
Dominios asociados	POS Comercio
Aspectual	-
Importancia	Vital
Estado	Validado
Estabilidad	Media
Comentarios	Ninguno

Para realizar la asociación entre requisitos de dominio y requisitos específicos, se inicia con la búsqueda de los requisitos de dominio en el catálogo TRIPODE, analizando la descripción del requisito para ver si es posible realizar la asociación con el problema específico.

Para este caso, se obtiene como resultado que el requisito Formas de Pago Válidas del caso de estudio Supermercados YUCATA (Ver Tabla 31), se relaciona con el requisito de dominio POS Validar Forma de Pago.

Tabla 32. Requisito de dominio validar formas de pago

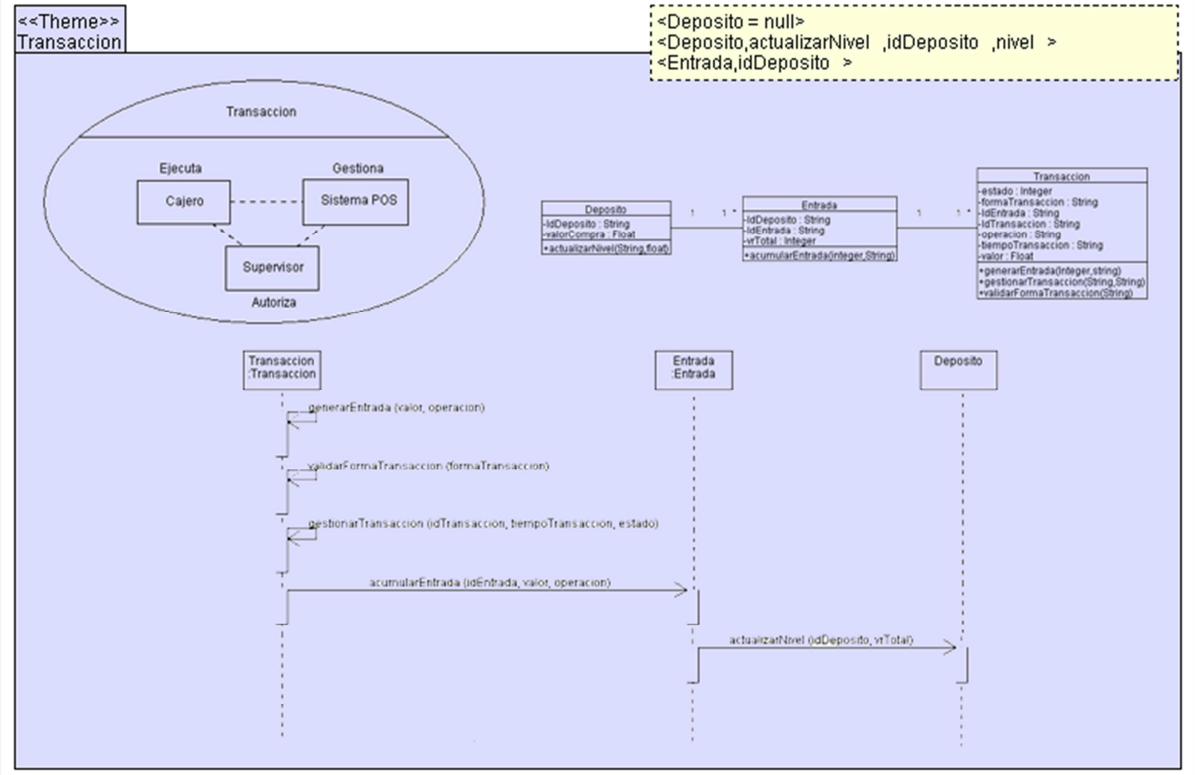
FRQ-002	**Validar Forma de Pago**
Versión	1.0
Autores	Iván Cisneros, Gloria Erazo, Germán Peña
Fuentes	Supervisor de Caja
Descripción	El sistema deberá validar la forma de pago y realizar las operaciones correspondientes, por ejemplo: Pago en efectivo. Pago con moneda extranjera. Pago con bonos. Pago con tarjeta crédito. Pago con tarjeta débito. Pago mixto.
Dominios asociados	POS Comercio
Patrón solución	Transacción
Estado	Pendiente de verificación
Estabilidad	Alta
Comentarios	Ninguno

- Recuperar Patrón

Después de realizar la caracterización de requisitos, se continúa con la recuperación del patrón que puede dar solución al requisito Formas de Pago Válidas, para lo cual se realiza la búsqueda dentro del catálogo, de los patrones asociados al requisito de dominio relacionado con el requisito específico.

Como resultado de ésta búsqueda, el patrón, sugerido en el catálogo, para dar solución al requisito de dominio POS Validar Forma de Pago, es el patrón Transaccion (figura 50). Por lo tanto se hace la recuperación de este patrón para dar inicio a la solución del requisito específico Formas de Pago Válidas.

Figura 50: Template patrón transacción



- Instanciar Patrón

Momento 1

En este primer momento, se hace el análisis de cada uno de los atributos y métodos de las clases Transacción, Entrada y Depósito, viendo cómo a través de estos elementos se resuelven las peticiones del requisito de Formas de Pago Válidas.

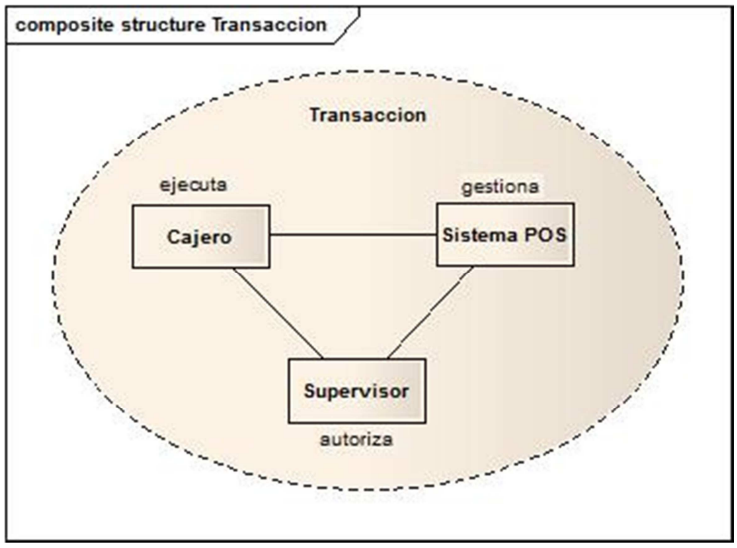
De acuerdo a los parámetros que el Signature del patrón plantea, se realizan los siguientes cambios:

- El nombre de la clase Depósito por Pago
- El atributo Id Deposito por Id Pago
- El atributo nivel por valor Pago
- El método actualizar Nivel por actualizar Pago
- En la clase Entrada se modifica el atributo Id Deposito por Id Pago manteniendo la coherencia con el cambio realizado al atributo Id Deposito de la clase Deposito.

Estos cambios se hacen con el fin de dar un nombre que sea más representativo, haciendo referencia al pago realizado por el cliente en el punto de venta. En este momento de instanciación no se realiza ninguna otra modificación.

- Modelo de alcance

Figura 51: Colaboración frq002 formas de pago válidas: momento 1



Actores

- Cajero: representa el funcionario que opera la caja donde el cliente ejecuta el pago.

- Sistema POS: representa el software que gestiona los procesos que implican la validación del pago realizado por el cliente.

- Supervisor: representa el funcionario encargado de controlar y permitir los procesos de alto nivel requerido en un punto de venta para validar el pago realizado por el cliente.

Roles

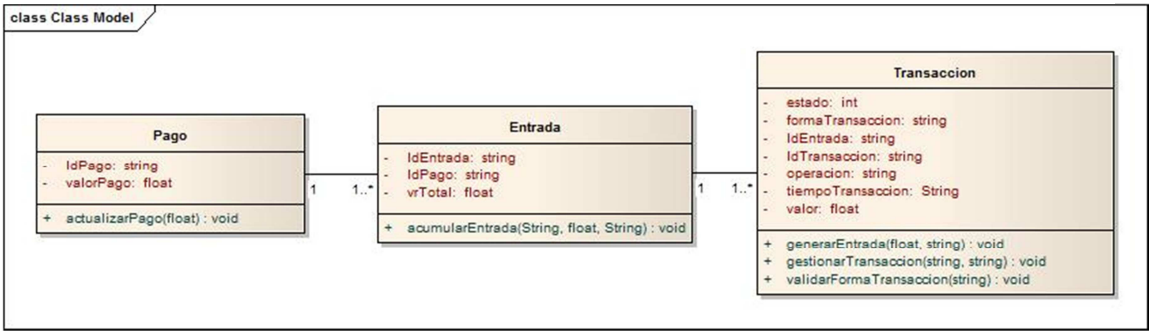
- Autorizar: representa la función de conceder autoridad, facultad o derecho para realizar una acción que requiera de un alto compromiso para la empresa durante la validación del pago realizado por el cliente.

- Ejecutar: representa la acción de validar el pago realizado por el cliente.

- Gestionar: representa la función de recibir, procesar, validar, almacenar la información referente a la validación del pago realizado por el cliente.

- Modelo Estructural

Figura 52: Diagrama de Clases FRQ002 Formas de Pago Válidas: Momento 1



Clase Transacción: Describe conceptualmente cada uno de los pagos que el cliente puede realizar para cancelar el valor de su compra, los elementos son utilizados así:

Atributos:

Estado: Atributo que representa la vigencia de la transacción.

Forma Transacción: Atributo que representa la forma de pago (efectivo, tarjeta de crédito, tarjeta débito, bono, moneda extranjera).

Id Entrada: Atributo que representa el código de identificación de la entrada generada por los pagos realizados.

Id transacción: Atributo que representa el identificador de la transacción actual.
Operación: Atributo que representa si el valor ingresado se sumará o se restará en el acumulador.

Tiempo Transacción: este atributo no cumple ninguna función para el requisito tratado, por lo tanto se lo reevaluará en el momento 2 de la instanciación.

Valor: Atributo que representa el valor del pago actual.

Métodos:

Generar Entrada: representa el proceso encargado de enviar a la clase Entrada el valor y la operación del pago actual para su posterior sumatoria.

Gestionar Transacción: representa el proceso que actúa como interfaz para poder realizar las operaciones necesarias relacionadas con la confirmación exitosa del pago.

Validar Forma transacción: representa el proceso que valida que el pago que realiza el cliente se efectúe a través de un mecanismo que sea permitido por el sistema.

Clase Entrada: Describe conceptualmente la acumulación parcial de cada uno de los pagos realizados por el cliente.

Atributos:

Id Pago: representa el código de identificación del acumulador del pago que será afectado por las entradas.

IdEntrada: representa el código de identificación de la entrada actual, en la que se lleva la sumatoria de los montos de pagos ingresados a través de los diferentes medios válidos.

Vr Total: representa el acumulador de los montos pagados.

Métodos:

Acumular Entrada: representa el proceso encargado de realizar la sumatoria de cada uno de los pagos efectuados por el cliente, para cancelar el valor total de la compra.

Clase Pago: Describe conceptualmente la sumatoria total de los pagos realizados por el cliente para cancelar el valor de la compra.

Atributos:

Id Pago: Atributo que representa el código de identificación del pago total.

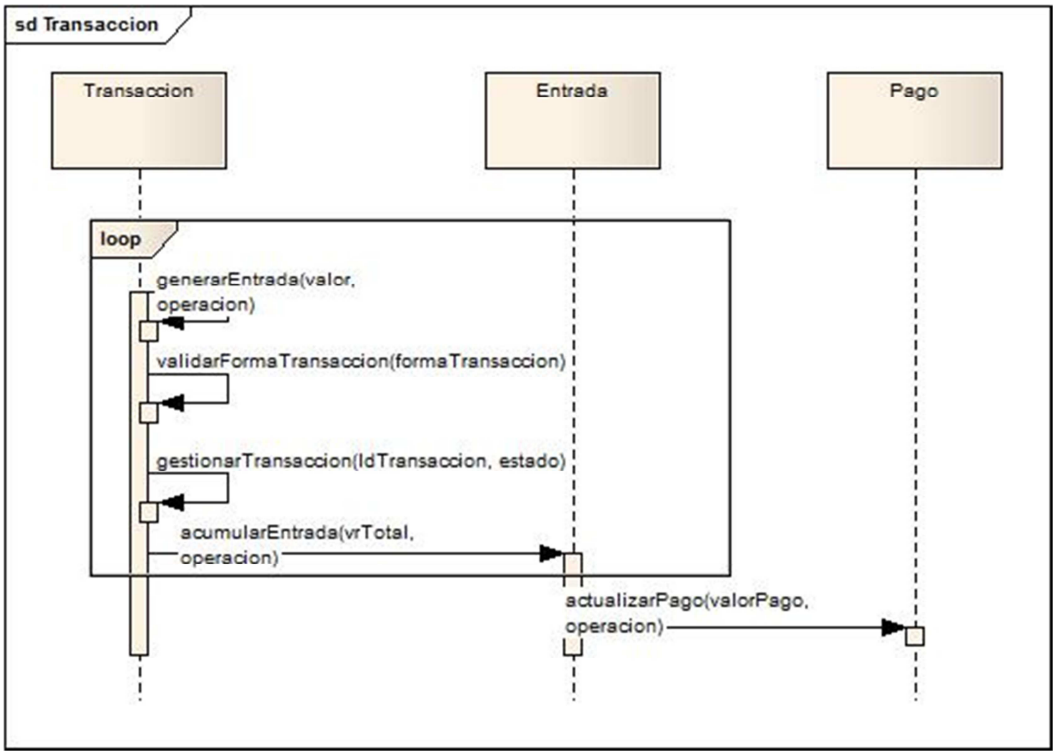
Valor pago: Atributo que representa el valor total cancelado por el cliente a través de una o varias formas de pagos válidos.

Métodos

Actualizar Pago: representa el proceso que permite llevar la sumatoria de los pagos parciales realizados por el cliente a través de una o varias formas de pago válidas.

- Modelo de Comportamiento

Figura 53: Diagrama de secuencia frq002 formas de pago válidas: momento 1



Objetos:

Transacción: representa cada uno de los pagos realizados para cancelar el valor de una compra, utilizando un medio válido en el sistema.

Entrada: representa el acumulador de cada uno de los pagos parciales.

Pago: representa el pago total realizado por el cliente a través de una o varias formas de pago válidas.

Mensajes:

Generar Entrada: representa el proceso de crear el pago con su respectivo valor y operación a realizar. Este proceso se ejecuta una vez por cada transacción.

Validar forma transacción: representa el proceso con el que se verifica que el medio utilizado para realizar el pago sea uno de los medios permitidos de acuerdo a las reglas del negocio. Este proceso se ejecuta una vez por cada transacción.

Gestionar transacción: representa el proceso que actúa como interfaz para poder realizar las operaciones necesarias relacionadas con la confirmación exitosa del pago. Este proceso se ejecuta una vez por cada transacción.

Acumular Entrada: representa el proceso con el cual se aumenta o disminuye el valor de los pagos realizados por el cliente, el cual afectará finalmente al pago total. Este proceso se ejecuta una vez por cada transacción.

Actualizar Pago: representa el proceso de actualizar el valor total del pago realizado por parte del cliente a través de las diferentes formas de pago. Este proceso se ejecuta una vez ya estén acumuladas todas las entradas.

Ciclos

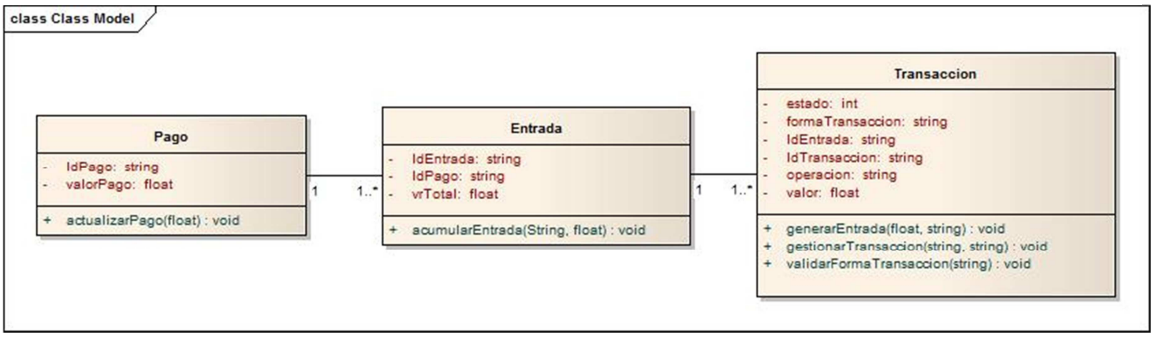
Entradas: representa el ciclo donde se realizan los pagos parciales que generan una entrada, se validan y registran estos pagos parciales y se acumula la entrada q finalmente afecta al valor del pago total realizado por el cliente. Estas actividades vuelven a ejecutarse cada vez que el cliente realice un pago parcial a través de cualquier medio de pago válido para el sistema.

Momento 2
En este momento de instanciación, se hace necesario eliminar de la clase Transaccion el atributo tiempoTransaccion, debido a que dentro del requisito actual, este atributo no cumple ninguna funcionalidad.

Las otras clases del modelo estructural no sufren ninguna modificación, así como tampoco se realiza ningún cambio dentro de los modelos de comportamiento y alcance.

Modelo Estructural

Figura 54: Diagrama de clases frq002 formas de pago válidas: momento 2



6.2.2. Solución del requisito control de devoluciones

- Caracterizar problemas por requisitos

Tabla 33. Plantilla de requisito específico control de devoluciones

FRQ-003	Control de Devoluciones
Versión	1.3
Autores	Yurani Bolaños, Tania Araújo
Fuentes	Supervisor de Caja Supermercado YUCATA
Objetivos asociados	OBJ - 001 OBJ – 002
Requisitos asociados	FRQ-001 Identificación y registro de productos FRQ-008 Mantener disponible información válida.
Descripción	El sistema deberá realizar el control de posibles devoluciones durante la venta, de la siguiente manera: Devoluciones de 1 o más productos: Si la devolución del (los) producto(s) se realiza después de ser identificado y enlistado en la factura temporal, se procederá a borrarlo(s) de la lista y a actualizar el total de la factura. Si la devolución del (los) producto(s) se realiza después del pago del total de la factura entonces se actualiza en la factura temporal la lista de productos y el valor total de la venta. Si el pago se hizo en efectivo con moneda nacional y/o extranjera se reversa el registro de pago por el valor correspondiente al (los) producto(s) devuelto(s) y se realiza la devolución del dinero. Para las otras formas de pago se emitirá un bono con el valor correspondiente, del cual se hace registro en la factura como un producto más de compra. Si la devolución del (los) producto(s) se realiza después de ejecutar la venta (imprimir la factura), el sistema solicita el número de unidades por tipo de producto y el código de barra del (los) producto(s) devuelto(s) y envía la información a los sistemas involucrados en la operación inicial para reversar los registros efectuados correspondientes al (los) producto(s) devuelto(s). El sistema emitirá un bono con el valor correspondiente, esto aplica para todas las formas de pago. Cancelación de la transacción: Si la cancelación de la transacción se realiza antes de efectuar el pago, el sistema deberá eliminar la factura temporal. Si la cancelación de la transacción se realiza después de efectuar el pago: si el pago se realiza en efectivo con moneda nacional o extranjera el sistema elimina la factura temporal y se retorna al cliente el valor cancelado. Si se registraron pagos con otro medio, se deberá eliminar los productos de la factura temporal y registrar un producto tipo Bono por el valor del pago realizado. Se imprime el bono. Si la cancelación de la transacción se realiza después de ejecutar la venta, el sistema hace el registro de la anulación de la factura por concepto de cancelación, envía la información a los sistemas involucrados en la operación inicial para reversar la transacción y emite un bono por valor igual al total de la venta, esto aplica para todas las formas de pago.
Dominios asociados	POS Comercio
Aspectual	-
Importancia	Vital
Estado	Validado
Estabilidad	Alta
Comentarios	Ninguno

Al realizar la búsqueda de los requisitos de dominio en el catálogo TRIPODE, se encuentra que la descripción del requisito de dominio POS Control de devoluciones (Tabla 3), está estrechamente asociada con el requisito específico Control de Devoluciones (Ver Tabla 33).

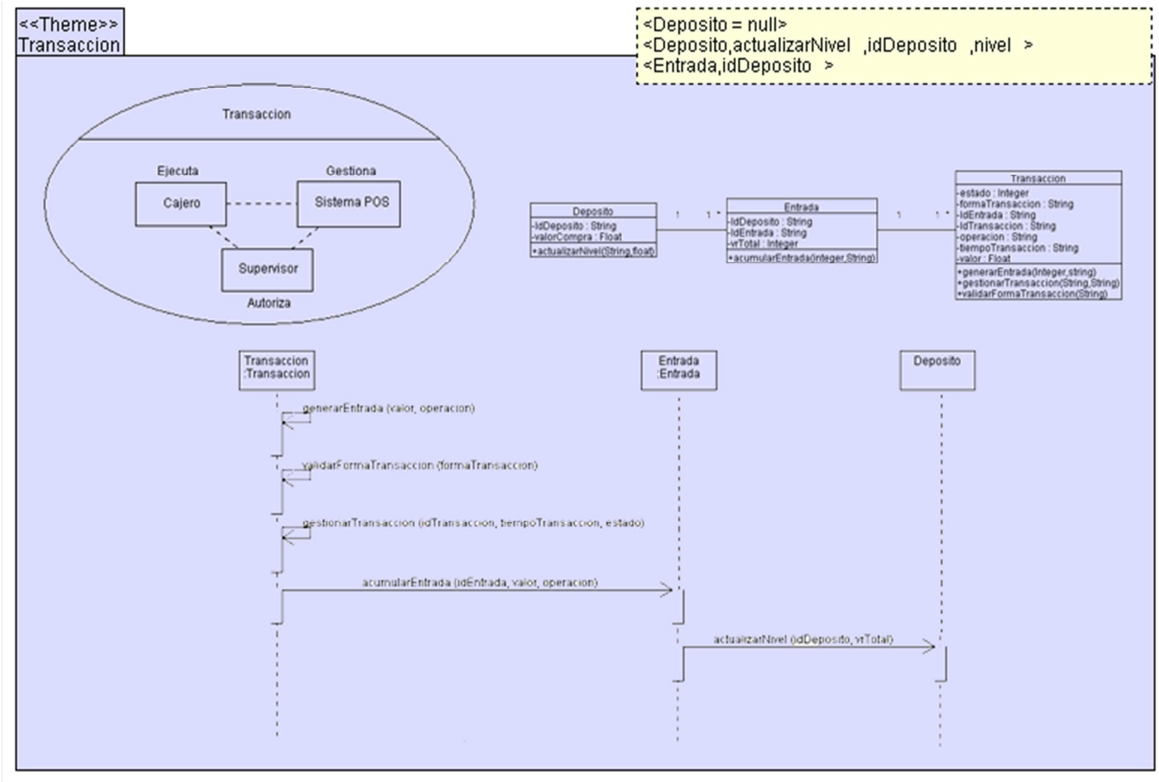
Tabla 34. Plantilla de requisito de dominio control de devoluciones

FRQ-003	Control de devoluciones
Versión	1.0
Autores	Iván Cisneros, Gloria Erazo, Germán Peña
Fuentes	Supervisor de Caja
Descripción	El sistema deberá realizar el control de posibles devoluciones: Devoluciones de 1 o más productos: Si la devolución del (los) producto(s) se realiza después de ser identificado y enlistado en la factura temporal, se procederá a borrarlo(s) de la lista y a actualizar el total de la factura. Si la devolución del (los) producto(s) se realiza después del pago del total de la factura entonces se borra el producto de la lista, se actualiza el total de la factura y se reversa la transacción de pago por el valor correspondiente al (los) producto(s) devuelto(s) de acuerdo a las políticas de la organización. Si la devolución del (los) producto(s) se realiza después de ejecutar la venta, el sistema envía la información a los sistemas involucrados en la operación inicial para reversar los registros generados correspondientes al (los) producto(s) devuelto(s). Cancelación de la venta: Si la cancelación de la venta se realiza antes de efectuar el pago, el sistema deberá eliminar la factura temporal. Si la cancelación de la venta se realiza después de efectuar el pago o ejecutar la venta, el sistema envía la información a los sistemas involucrados en la operación inicial para reversar los registros generados correspondientes a la venta y se reversa la transacción de pago por el valor correspondiente a la venta de acuerdo a las políticas de la organización.
Dominios asociados	POS Comercio
Patrón solución	Transaccion
Estado	Pendiente de verificación
Estabilidad	Media
Comentarios	Ninguno

- Recuperar Patrón

Para recuperar el patrón que puede dar solución al requisito Control de Devoluciones, primero se realiza la búsqueda de los patrones asociados al requisito de dominio relacionado con el requisito específico. Como resultado de ésta búsqueda, el patrón, sugerido en el catálogo, para dar solución al requisito de dominio POS Control de Devoluciones, es el patrón Transaccion (Ilustración 55), por lo tanto se hace la recuperación de este patrón para dar inicio a la solución del requisito específico Control de Devoluciones.

Figura 55: Template patrón transacción



- Instanciar Patrón

Momento 1

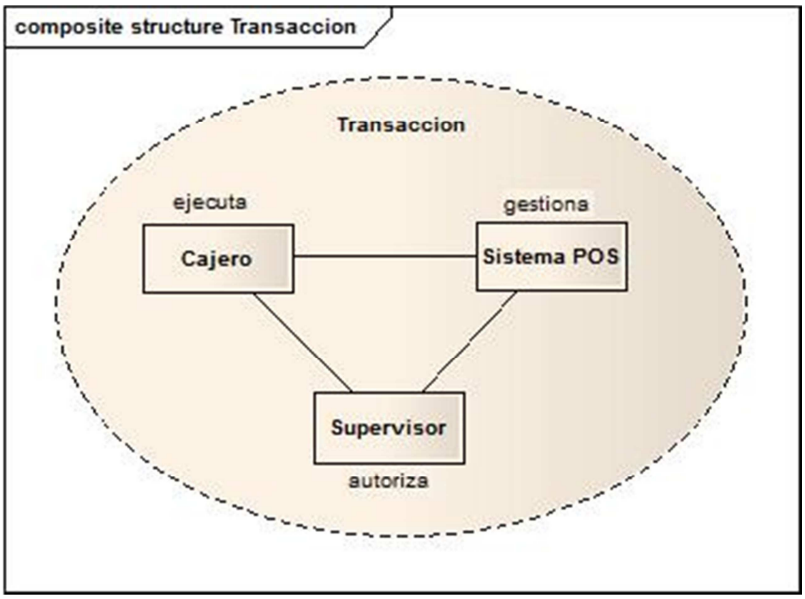
Al igual que para el anterior requisito, en este primer momento, se hace el análisis de cada uno de los atributos y métodos de las clases Transacción, Entrada y Depósito, viendo cómo a través de estos elementos se resuelven las peticiones del requisito Control de Devoluciones.

De acuerdo a los parámetros que el Signature del patrón plantea, se realizan los siguientes cambios:

- El nombre de la clase Depósito por Compra
- El atributo Id Deposito por Id Compra
- El atributo nivel por valor Compra
- El método actualizar Nivel por actualizar Valor
- En la clase Entrada se modifica el atributo Id Deposito por Id Compra manteniendo la coherencia con el cambio realizado al atributo Id Deposito de la clase Depósito.

Estos cambios se hacen con el fin de dar un nombre que sea más representativo, haciendo referencia al valor de la compra realizada por el cliente en el punto de venta. En este momento de instanciación no se realizan ninguna otra modificación. Modelo de alcance

Figura 56: Colaboración requisito específico frq003 control de devoluciones



Actores

- Cajero: representa el funcionario que opera la caja donde se realiza la devolución o cancelación del producto.

- Sistema POS: representa el software que gestiona los procesos que implican la devolución o cancelación del producto.

- Supervisor: representa el funcionario encargado de controlar y permitir los procesos de alto nivel relacionados con devoluciones o cancelaciones.

Roles

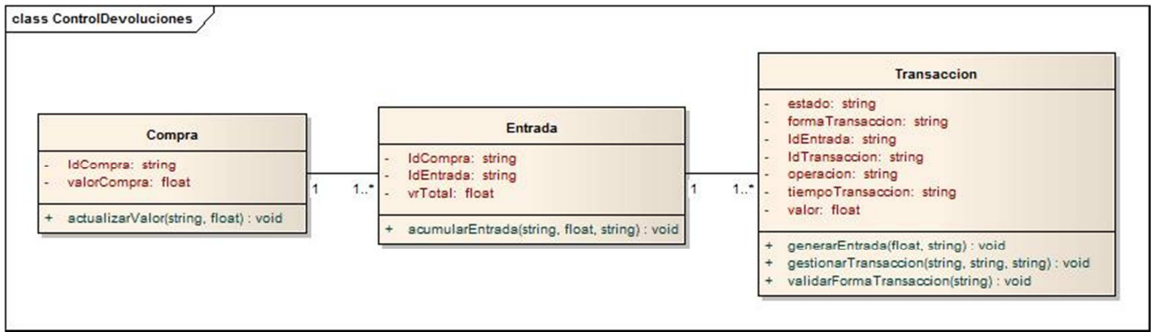
- Autorizar: representa la función de conceder autoridad, facultad o derecho para realizar una acción que requiera de un alto compromiso para la empresa durante una devolución o cancelación.

- Ejecutar: representa las diferentes acción de realizar dentro del sistema POS una devolución o cancelación.

- Gestionar: representa la función de recibir, procesar, validar, almacenar la información referente a la operación de devolución o cancelación.

- Modelo Estructural

Figura 57: Diagrama de clases requisito específico frq 003 control de devoluciones



Clase Transacción: Describe conceptualmente cada una de las devoluciones o cancelaciones de productos el cliente puede realizar durante la compra, los elementos son utilizados así:

Atributos:

Estado: representa si la devolución o cancelación se hizo o no efectiva.

Forma Transacción: representa el tipo de transacción que se realiza, en este caso el tipo va ser una devolución o una cancelación.

Id Entrada: representa el código de identificación del valor generado por concepto de todas las devoluciones o cancelaciones que ser realizaron.
Id Transaccion: representa el código de identificación del valor generado por cada devolución o cancelación.

Operación: representa la forma en que el valor generado por la devolución o cancelación afectara el valor de la compra.

Tiempo Transacción: representa el momento en el que se realiza la devolución o cancelación. por ejemplo si se realiza cuando se enlistan los productos, después del pago o después de ejecutar la venta.

Valor: representa la cantidad monetaria que se debe descontar al hacer la devolución o cancelación del producto.

Métodos:

Generar Entrada: representa la operación de resta del producto devuelto o cancelado, llevando como parámetros el valor del producto y la operación que en este caso siempre va ser resta.

Gestionar Transacción: representa al proceso que actúa como interfaz para poder actualizar en la factura los productos devueltos o cancelados, este lleva como parámetros el identificador de la devolución o cancelación, el tiempo en el que se realiza, y su respectivo estado.

Validar Forma transacción: representa a la operación que permite reconocer si el tipo de transacción que se va a realizar, en este caso una devolución o una cancelación, sea una transacción válida para el sistema.

Clase Entrada: Describe conceptualmente la acumulación parcial de cada una de las devoluciones o cancelaciones realizadas por el cliente.

Atributos:

Id Compra: representa el código de identificación del valor de la compra que será afectado por el valor de las entradas.

Id Entrada: representa el código de identificación de cada entrada generada por una devolución o cancelación.

Vr Total: representa el valor monetario de todos los productos que fueron devueltos o cancelados que se descontarán al valor total de la compra.

Métodos:

Acumular Entrada: representa la operación de acumular el valor de cada devolución o cancelación para que después el valor total sea descontado al valor total de la compra

Clase Compra: Describe conceptualmente el valor acumulado de la compra y cómo este se ve afectado en el momento en el que se haga efectiva la entrada generada por las devoluciones o cancelaciones realizadas por el cliente.

Atributos:

Id Compra: representa el código de identificación del valor total de la compra.
valorCompra: representa la sumatoria del valor de los productos adquiridos por el cliente en la compra antes y después de hacer efectiva la entrada generada por conceptos de devolución o cancelación.

Métodos:

Actualizar Valor: representa la operación de restar al total de la compra, el valor obtenido por los descuentos o cancelaciones que se llevaron a cabo.

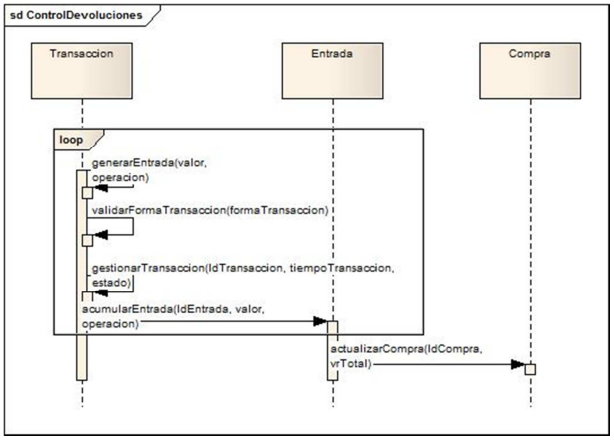
Cardinalidad

Una devolución ó cancelación puede generar una entrada, una entrada puede ser generada por una o más devoluciones o transacciones.

Una entrada puede afectar a una compra, una compra puede ser afectada por una o más entradas.

Modelo de Comportamiento

Figura 58. Diagrama de secuencia requisito específico frq003 control de devoluciones



Objetos

Transaccion: representa cada una de las devoluciones o cancelaciones de productos efectuadas por el cliente y que afectaránel valor total de la compra.

Entrada: representa el valor acumulado de todos los valores de las devoluciones o cancelaciones de los productos que van a afectar el valor total de la compra.

Compra: representa el lugar concreto donde se almacenan el valor total de la compra y en donde entra a descontar el valor de las entradas.

Mensajes

Generar Entrada: representa el proceso de crear la entrada con su respectivo valor y operación a realizar. Este proceso se ejecuta una vez por cada devolución o cancelación.

Validar Forma transacción: representa el proceso de reconocer la validez del tipo de transacción a realizar. Este proceso se ejecuta una vez por cada transacción, el tipo de transacción puede ser una devolución o una cancelación.

Gestionar Transacción: representa el proceso de llevar un registro de las devoluciones o cancelaciones realizadas con un identificador, el tiempo y el estado. Este proceso se ejecuta una vez por cada transacción.

Acumular Entrada: representa el proceso de aumentar el valor total de la entrada que afectará finalmente a la compra. Este proceso se ejecuta una vez por cada transacción.

Actualizar Compra: representa el proceso de disminuir del valor de la compra el valor acumulado de la entrada. Este proceso se ejecuta una vez ya estén acumuladas las entradas.

Ciclos

Entradas: representa el ciclo donde se realiza la devolución o cancelación del producto, este valor genera una entrada, posteriormente se realiza una validación y un registro, después se acumulan los valores en las entradas para descontarle al valor total de la compra.

Momento 2

Los modelos de la solución al requisito Control de Devoluciones, no sufren ninguna modificación adicional a las realizadas en el momento uno de instanciación, debido a que con estas modificaciones ya se encuentra la solución completa al requisito tratado.

Solución del Requisito Gestionar Factura

Caracterizar problemas por requisitos

Tabla 35. Requisito específico gestionar factura

FRQ-004	Gestionar Factura
Versión	1.1
Autores	Tania Araújo, Yurani Bolaños
Fuentes	Supervisor de Caja Supermercado YUCATA
Objetivos asociados	Registrar las ventas realizadas en el Supermercado
Requisitos asociados	FRQ-001 Identificación y registro de productos FRQ-002 Identificación de formas válidas de pago FRQ-005 Gestionar Interfaces
Descripción	Una vez efectuado el pago, el sistema deberá hacer persistencia de la factura, la cual contiene la siguiente información: Cabecera: Compuesta por el código de la factura, el NIT de la empresa YUCATA, identificación y nombre del cajero, número de caja, fecha y hora de atención. Productos: Nombre del producto, cantidad, si el producto es una fruta o verdura se indicará el peso en gramos, valor unitario o valor por libra para el caso de frutas o verduras, valor total que corresponde al valor unitario por la cantidad de unidades compradas, para el caso de frutas y verduras se indicará el valor correspondiente al peso del producto en gramos por el valor por libra. Subtotal: Sumatoria de los valores totales listados en la parte de Productos IVA: Se discrimina el valor cobrado por concepto IVA Total: Sumatoria del Subtotal más valor IVA. Pago: Se indica la forma y valor de pago, valor de cambio, total artículos comprados. Cliente [Opcional]: Si el cliente ha hecho registro de sus datos personales en el punto de atención al cliente y pertenece al programa de puntos se mostrará el número de identificación y el nombre del cliente y la cantidad de puntos por la compra y puntos acumulados. Información Adicional [Opcional]: Si existen promociones de temporada, la información correspondiente a dichas promociones se podrán mostrar en esta sección.
Dominios asociados	POS Comercio
Aspectual	-
Importancia	Vital
Estado	Validado
Estabilidad	Media
Comentarios	Ninguno

Al realizar la búsqueda de los requisitos de dominio en el catálogo TRIPODE, se obtiene la descripción del requisito de dominio POS Gestionar Factura (Tabla 3), el cual está estrechamente asociado con el requisito específico Gestionar Factura (Ver Tabla 35).

Tabla 36. Plantilla de requisito de dominio gestionar factura

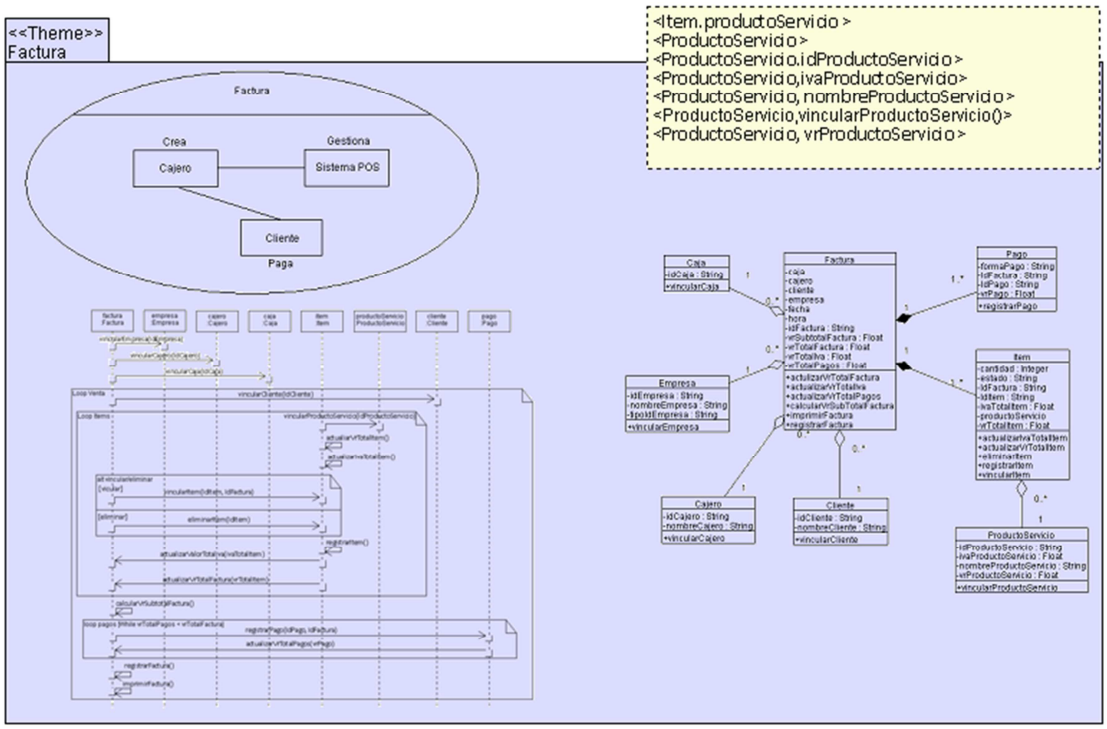
FRQ-011	Gestionar Factura
Versión	1.1
Autores	Gloria Erazo, Iván Cisneros, Germán Peña
Fuentes	Supervisor de Caja Supermercado LINAR
Objetivos	Registrar las ventas realizadas en el Supermercado

asociados	
Requisitos asociados	FRQ-001 Identificación y registro de productos FRQ-002 Identificación de formas válidas de pago FRQ-004 Ejecutar la venta
Descripción	Una vez efectuado el pago, el sistema deberá hacer persistencia e impresión de la factura, la cual contiene la siguiente información: Cabecera: Compuesta por el código de la factura, el NIT de la empresa LINAR, identificación y nombre del cajero, número de caja, fecha y hora de atención. Productos: Nombre del producto, cantidad, valor unitario, valor total que corresponde al valor unitario por la cantidad de unidades compradas. Subtotal: Sumatoria de los valores totales listados en la parte de Productos IVA: Se discrimina el valor cobrado por concepto IVA Total: Sumatoria del Subtotal más valor IVA. Pago: Se indica la forma y valor de pago. Cliente: Si el cliente ha hecho registro de sus datos personales en el punto de atención al cliente y pertenece al programa de puntos se mostrará el número de identificación y el nombre del cliente y la cantidad de puntos por la compra y puntos acumulados.
Dominios asociados	POS Comercio
Importancia	Vital
Estado	Validado
Estabilidad	Media
Comentarios	Ninguno

- Recuperar Patrón

Para recuperar el patrón que puede dar solución al requisito Gestionar Factura, primero se realiza la búsqueda de los patrones asociados al requisito de dominio relacionado con el requisito específico. Como resultado de ésta búsqueda, el patrón, sugerido en el catálogo, para dar solución al requisito de dominio POS Gestionar Factura, es el patrón Factura (Ilustración 59), por lo tanto se hace la recuperación de este patrón para dar inicio a la solución del requisito específico Gestionar Factura.

Figura 59: Template patrón factura



- Instanciar Patrón

Momento 1

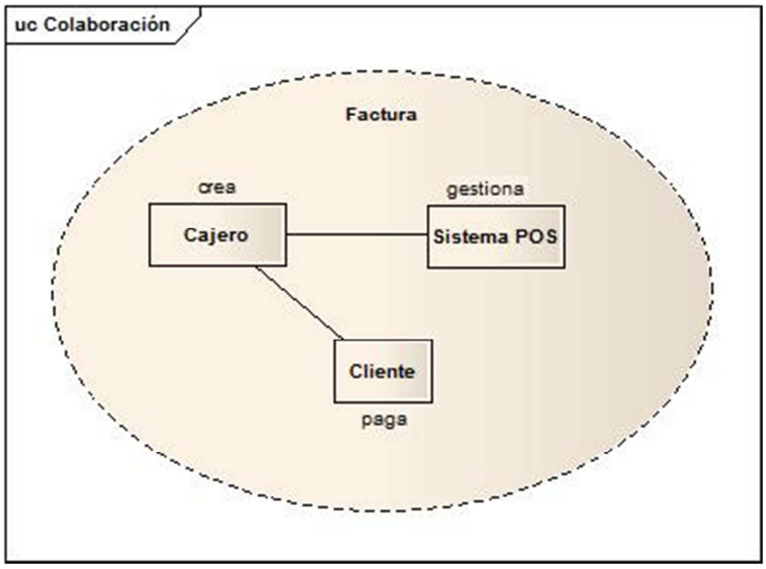
En este primer momento, se realiza el análisis de cada uno de los atributos y métodos de las clases propuestos por el patrón Factura.

Los valores de los parámetros que el Signature del patrón permite modificar son los de la clase ProductoServicio sus atributos y funciones, el atributo ProductoServicio de la clase Item.

En el caso de estudio POS, el supermercado se dedica a la venta de productos y no de servicios por lo cual se modifica la palabra Producto Servicio a Producto, para establecer la relación. Las demás clases del patrón Factura no sufren ninguna modificación en este momento de instanciación.

Modelo de Alcance

Figura 60: Colaboración requisito específico frq004 gestionar factura: momento 1



Actores:

Cajero: representa el funcionario que opera la caja donde se desarrolla la operación de venta.

Cliente: representa la persona u organización que adquiere los productos ofrecidos por Supermercados YUCATA.

Sistema POS: representa el software que gestiona el servicio de venta de productos en Supermercados YUCATA.

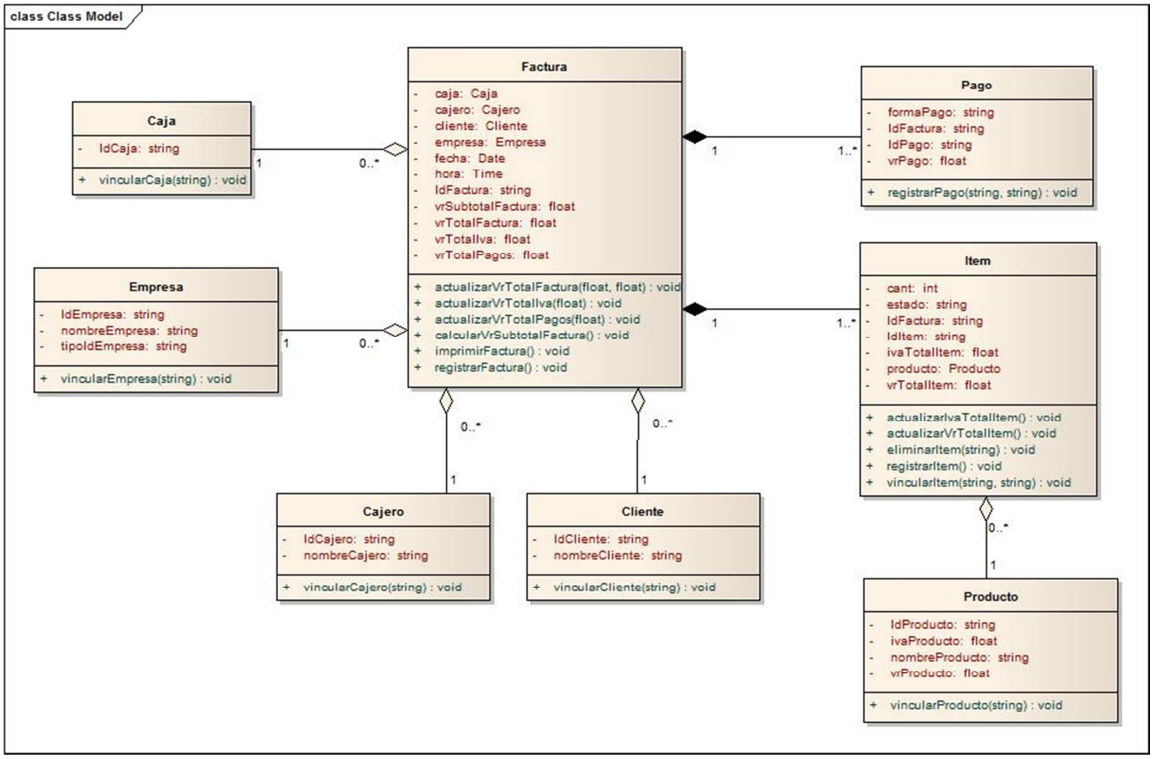
Roles:

Crear: representa la función de recolectar la información de la operación de venta.
Gestionar: representa la función de recibir, procesar, almacenar e imprimir en la factura, la información referente a la operación de venta.

Pagar: representa la función de realizar un desembolso por la adquisición de productos.

Modelo Estructural

Figura 61: Diagrama de clases requisito específico frq004 gestionar factura: momento 1



Clase Factura: Describe conceptualmente el documento que contiene la información de una operación de venta de productos.

Atributos:

Caja: representa el punto de venta donde se realiza la transacción.

Cajero: representa al funcionario que opera en la caja donde se desarrolla la transacción.

Cliente: representa a la persona que adquiere los productos del supermercado YUCATA.

Empresa: representa a la empresa supermercados YUCATA, en este caso hará referencia al NIT de la empresa.

Fecha: representa la información correspondiente a la fecha de creación de la factura que es la misma en la que se realiza la compra.

Hora: representa la información correspondiente a la hora de la creación de la factura que es la misma en la que se realiza la compra.

Id Factura: representa el Identificador de la factura, corresponde a un código cuyo valor es único.

Vr Subtotal Factura: representa el valor de la diferencia entre el valor total de la factura y el valor total del IVA de la factura.

Vr Total Factura: representa el valor de la suma total de los valores correspondientes a todos los productos adquiridos por el cliente.

Vr Total IVA: representa la suma total del IVA aplicado a cada uno de los productos comprados por el cliente.

Vr Total Pagos: representa la suma total de los pagos realizados por el cliente en la transacción.

Métodos

Actualizar Vr Total Factura: representa la operación de acumular los valores de cada uno de los productos.

Actualizar Vr Total Iva: representa la operación de acumular el valor total del IVA de cada uno de los items, de acuerdo al valor del porcentaje de IVA aplicado.

Actualizar VrT otal Pagos: representa la operación de acumular el valor total de los pagos realizados por el cliente en la transacción.

Calcular Vr Subtotal Factura: representa la operación de acumular el valor de cada uno de los productos incluidos en la compra sin tener en cuenta el valor de IVA es aplicado a cada uno de ellos.

Imprimir Factura: representa la operación de imprimir la factura.

Registrar Factura: representa la operación de hacer persistencia de la factura en la base de datos.

Clase Caja: describe conceptualmente los puntos de venta donde se desarrollan las operaciones de venta.

Atributos:

Id Caja: representa el código de identificación de la caja, este código es único.

Métodos:

Vincular Caja: representa la relación que se establece entre el punto de venta donde se realiza la transacción con la factura.

Clase Empresa: describe conceptualmente a la empresa Supermercados YUCATA

Atributos:

Id Empresa: representa el código de identificación de la empresa YUCATA, en este caso el NIT de la empresa.

nombreEmpresa: representa el nombre de la empresa, en este caso es Supermercados YUCATA.

Tipo Id Empresa: representa el tipo de identificación de la empresa YUCATA, en este caso el tipo de identificación es el Número de Identificación Tributaria (NIT).
Métodos:

Vincular Empresa: representa la relación entre la empresa Supermercados YUCATA con la factura.

Clase Cajero: describe conceptualmente los funcionarios que operan las cajas donde se desarrollan las operaciones de venta.

Atributos:

Id Cajero: representa el código de identificación del cajero.
Nombre Cajero: representa el nombre del cajero

Métodos:
Vincular
Cajero: representa la relación entre el cajero con la factura.

Clase Cliente: describen conceptualmente las personas u organizaciones que adquieren los productos ofrecidos por la empresa Supermercados YUCATA.

Atributos:

Id Cliente: representa el código de identificación del cliente, donde este código es único.
Nombre Cliente: representa el nombre del cliente.
Métodos:

Vincular Cliente: representa la relación de un cliente con la factura.

Clase Pago: describe conceptualmente los desembolsos que realizan los clientes por la adquisición de los productos ofrecidos por la empresa Supermercados YUCATA.

Atributos:
Forma Pago: representa el tipo de pago. Para el caso de Supermercados YUCATA el tipo de pago puede ser efectivo, tarjeta débito, tarjeta crédito o pago con bonos.

Id Pago: representa el código de identificación del pago, donde este código es único.

Id Factura: representa el código de identificación de la factura la cual está vinculada al pago, este código es único.

Vr Pago: representa el valor del pago.

Métodos:

Registrar Pago: representa la relación del pago con la factura y almacenar la información.

Clase Item: describe conceptualmente uno o un conjunto del mismo producto.

Atributos:

cant: representa el número de unidades que el cliente adquiere en la compra, por producto.

Estado: representa si el producto ha sido adicionado o eliminado de la factura.

Id Factura: representa el código de identificación de la factura a la cual está vinculado el item.

Id Item: representa el código de identificación del producto.

Iva Total Item: representa la suma total del IVA de cada una de las unidades adquiridas en la venta por producto.

Producto: representa el producto vinculado al ítem.
Vr Total Item: representa la sumatoria del valor de cada una de las unidades adquiridas en la venta por producto.

Métodos:

Actualizar Iva TotalItem: representa la operación de acumular el IVA de cada una de las unidades adquiridas en la venta por producto.

Actualizar Vr Total Item: representa la operación de acumular el valor de cada una de las unidades adquiridas en la venta por producto.

Eliminar Item: representa la operación de cambiar el estado del ítem a eliminado y borrarlo de la factura.

Registrar Item: representa la operación para almacenar la información del ítem.
Vincular Item: representa la operación para relacionar el ítem con una factura.

Clase Producto: describe conceptualmente los productos ofrecidos por la empresa Supermercados YUCATA.

Atributos:

Id Producto: representa el identificador del producto este se señala con un código.

Iva Producto: representa el valor del IVA del producto.

Nombre Producto: representa el nombre del producto.

Vr Producto: representa el valor del producto.

Métodos:

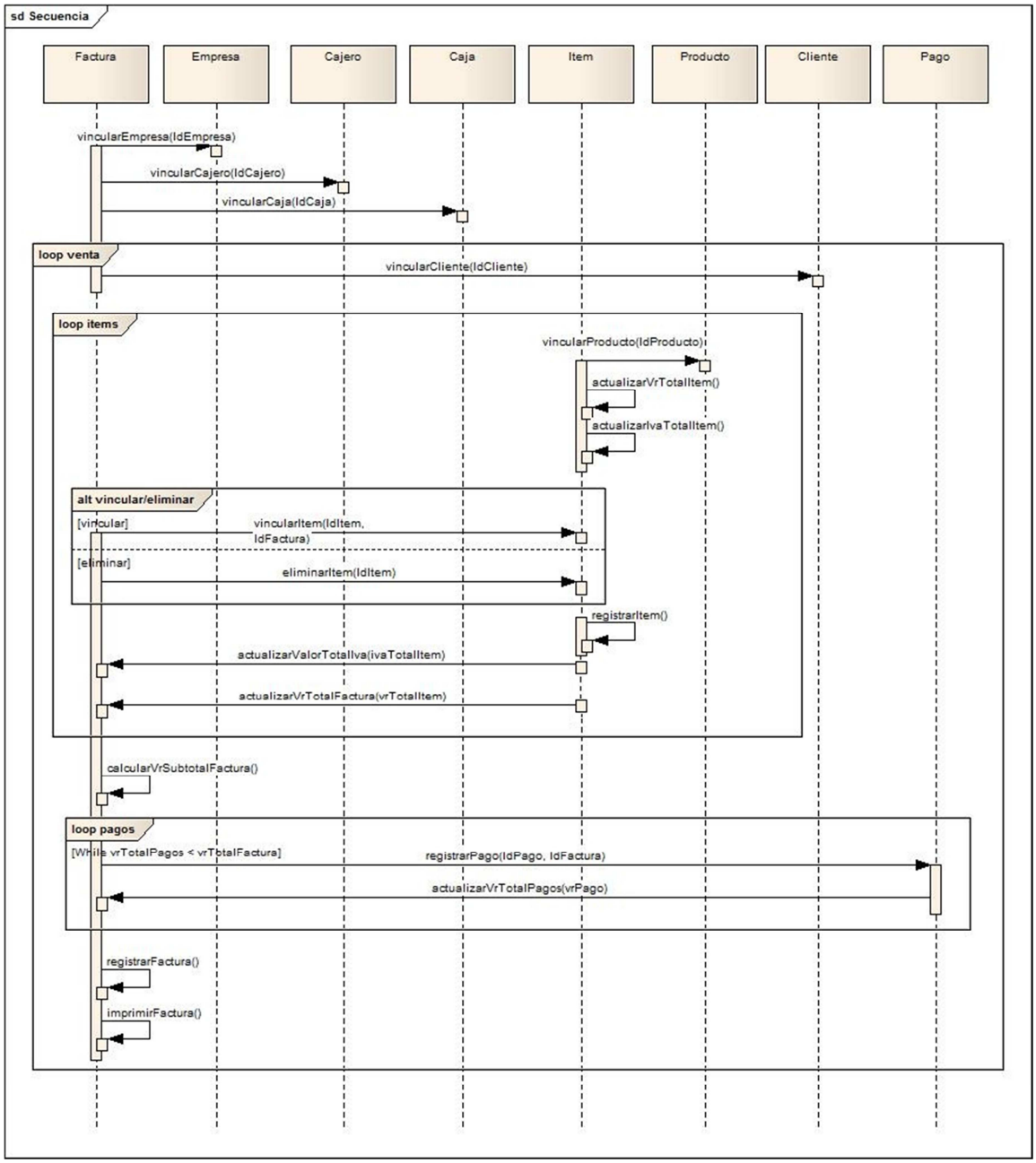
Vincular Producto: representa la operación para relacionar un producto con un ítem.

Cardinalidad:

- Una factura tiene una caja, una caja puede tener cero o más facturas.
- Una factura tiene una empresa, una empresa puede tener cero o más facturas.
- Una factura tiene un cajero, un cajero puede tener cero o más facturas.
- Una factura tiene un cliente, un cliente puede tener cero o más facturas.
- Una factura tiene uno o más pagos, un pago tiene una factura.
- Una factura tiene uno o más ítems, un ítem tiene una factura.
- Un ítem tiene un producto, un producto puede tener cero o más ítems.

Modelo de Comportamiento

Figura 62: Diagrama de secuencia requisito específico frq004 gestionar factura: momento 1



Objetos

Factura: representa el documento que contiene la información de una transacción.
Empresa: representa la organización en este caso es Supermercados YUCATA que ofrece los productos para la venta.

Cajero: representa el funcionario que opera en la caja donde se desarrollan las transacciones.

Caja: representa el punto de venta donde se desarrollan las transacciones.

Ítem: representa uno o varios productos del mismo tipo.

Producto: representa el objeto que la empresa tiene para el cliente a cambio de dinero.

Cliente: representa la persona u organización que adquiere los productos ofrecidos por la empresa.

Pago: representa el valor cancelado por el cliente a través de una forma válida de pago, a cambio del producto que ofrece la empresa al cliente.

Mensajes

Actualizar Iva Total Item: representa el proceso de acumular el IVA de los productos del ítem. Se ejecuta por cada unidad de producto vendido.

Actualizar Vr Total Factura: representa el proceso de acumulación del valor total de los ítems adquiridos en la venta. Se ejecuta por cada ítem.

Actualizar Vr Total Item: representa el proceso de acumular el valor de los productos del ítem. Se ejecuta por cada unidad de producto vendido.

Actualizar Vr Total Iva: representa el proceso de acumular el valor total del IVA de los ítems adquiridos en la operación de venta. Se ejecuta por cada ítem.

Actualizar Vr Total Pagos: representa proceso de acumular el valor de los pagos realizados por el cliente en la operación de venta.

Calcular Vr Subtotal Factura: representa el proceso de restar el valor total del IVA de la factura al valor total de la factura. Este proceso se ejecuta una sola vez por cada venta.

Eliminar Item: representa el proceso de cambiar el estado del ítem a eliminado y borrarlo de la factura.

Imprimir Factura: representa el proceso de imprimir la factura una vez terminada la operación de venta.

Registrar Factura: representa el proceso de hacer persistencia de la factura.

Registrar Item: representa el proceso de almacenar la información del ítem vendido o devuelto.

Registrar Pago: representa el proceso de vincular el pago a la factura y almacenar su información.

Vincular Caja: representa el proceso de relacionar la caja con la factura. Este proceso se ejecuta una sola vez, al inicio de la jornada laboral.

Vincular Cajero: representa el proceso de relacionar el cajero con la factura. Este proceso se ejecuta una sola vez, al inicio de la jornada laboral.

Vincular Cliente: representa el proceso de relacionar el cliente con la factura en la operación de venta.

Vincular Empresa: representa el proceso de relacionar la empresa con la factura. Este proceso se ejecuta una sola vez, al inicio de la jornada laboral.

Vincular Item: representa el proceso de relacionar el ítem vendido con la factura. Este proceso se ejecuta por cada ítem.

Vincular Producto: representa el proceso de relacionar el producto con el ítem. Este proceso se ejecuta por cada producto.

Ciclos

Los procedimientos de vinculación de caja, cajero y empresa se realizan únicamente al inicio de la jornada, por esto no son incluidos en los ciclos de facturación.

- Venta: representa el ciclo principal, en el cual se realizan los procedimientos de:
- Vinculación de productos a los ítems.
- Actualización del valor total de los ítems.
- Actualización de IVA total de los ítems.
- Vinculación o eliminación de ítems en la factura.
- Registro de ítems.
- Actualización del valor total de la factura.
- Actualización del valor total del IVA de la factura.
- Cálculo del valor subtotal de la factura.

- Registro de los pagos.
- Actualización del valor total de los pagos.
- Registro e impresión de la factura.

Items: representa el ciclo donde se vinculan los productos a los ítems y se actualizan sus correspondientes valores, se vinculan o eliminan los ítems de la factura y se actualizan el valor del IVA y los valores totales de factura.

Pagos: representa el ciclo en el cual se registran los pagos y se actualiza el valor total de pagos en la factura. Este ciclo se ejecuta mientras que el valor total de pagos sea menor que el valor total de la factura.
Alternativas

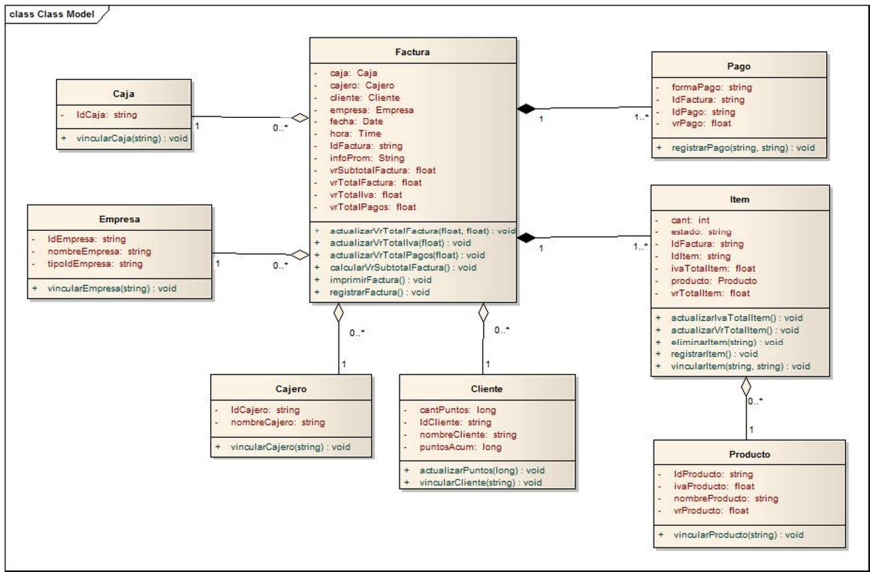
Vincular/Eliminar: representa el punto de flujo donde se selecciona una opción de operación sobre un ítem de la factura. Las opciones son: eliminar o vincular.

Momento 2

En este momento de instanciación, se hace necesario adicionar algunos atributos y métodos con los cuales se pueda cubrir el requisito de Gestionar Factura que tiene Supermercados YUCATA, el cual propone el manejo de puntos de cliente fiel y un anuncio de promociones en la factura que se entrega al cliente. A continuación se detallan los cambios realizados en cada uno de los diagramas del modelado

- Modelo Estructural

Figura 63: Diagrama de clases requisito específico FRQ004 gestionar factura: momento 2



- Clase Cliente

Atributos: Se han adicionado los siguientes elementos:

Cant Puntos: representa el número de puntos obtenidos por la compra actual.

Puntos Acum: representa el acumulado de puntos obtenidos por el cliente a lo largo de su historial de compras en el supermercado.

Métodos: Se ha adicionado el siguiente método:

actualizar Puntos: representa la sumatoria de los puntos obtenidos por el cliente en cada compra.

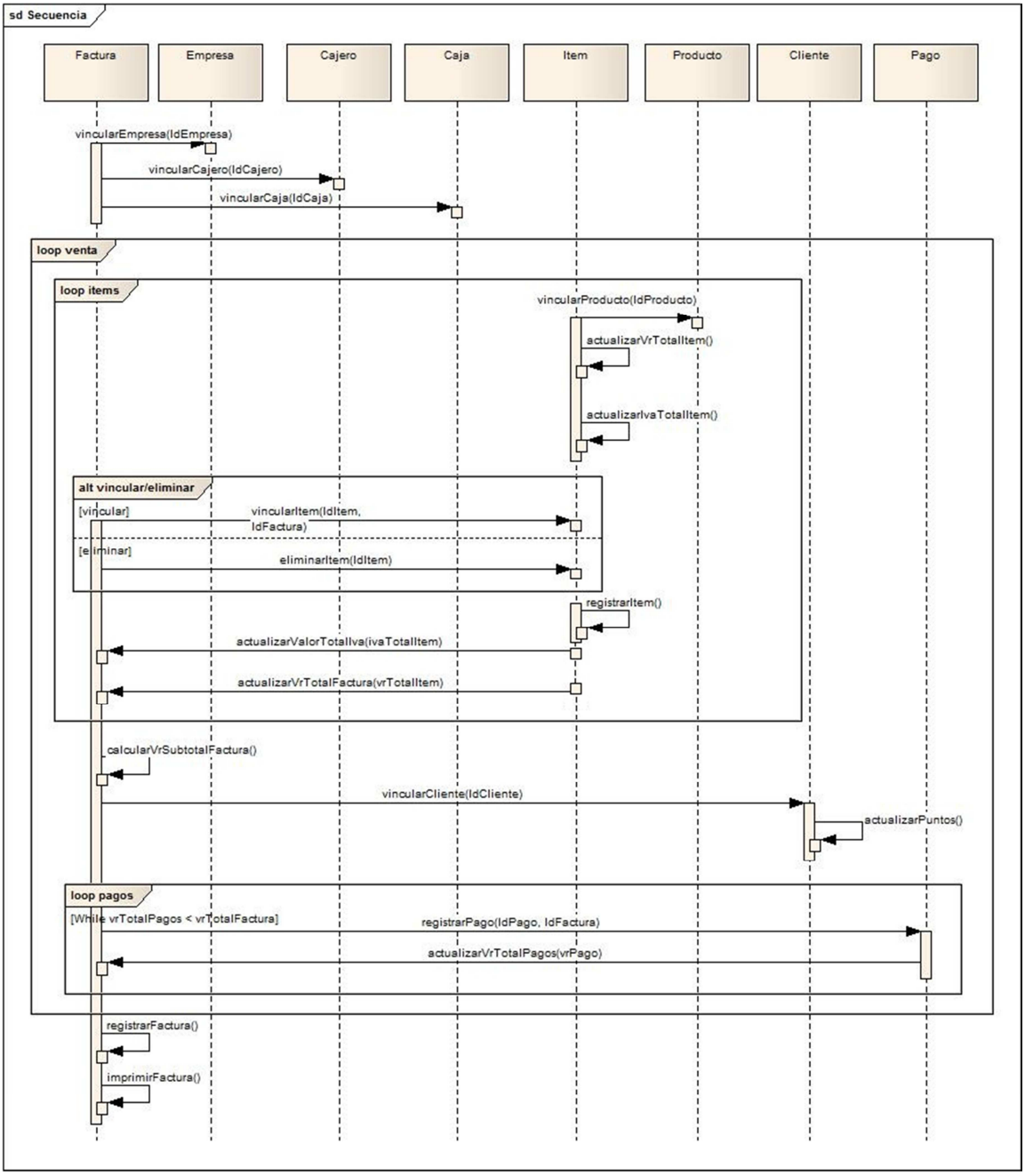
- Clase Factura:

Atributos: Se adiciona el siguiente atributo:

Info Prom: este atributo contiene la información de todas las promociones vigentes en forma de un texto en la factura.

- Modelo de Comportamiento

Figura 64: Diagrama de secuencia requisito específico FRQ004 gestionar factura: momento 2



Mensajes

Actualizar Puntos: se adiciona este método para poder representar el proceso que acumula los puntos obtenidos por el cliente en cada venta.

Solución del Requisito Gestión de Respaldo de Información

Caracterizar problemas por requisitos

Tabla 37. Requisito específico gestión de respaldo de información

NFR-001	Gestión de Respaldo de Información
Versión	1.1
Autores	Tania Araújo, Yurani Bolaños
Fuentes	Supervisor de Caja Supermercado YUCATA
Objetivos asociados	Mantener información coherente y confiable
Requisitos asociados	FRQ-008 Mantener disponible información válida
Descripción	El sistema debe permitir programar la creación de: Un backup interno en períodos de 5 horas, esta copia de seguridad contendrá todas las transacciones realizadas en el POS en ese espacio de tiempo. Un backup externo (medio magnético) el cual se conservará en un lugar distinto a la empresa, este contendrá las transacciones del POS de una semana. Estas copias de respaldo se hacen para preservar la información ante cualquier falla física o lógica.
Dominios asociados	General
Aspectual	-
Importancia	Vital
Estado	Validado
Estabilidad	Media
Comentarios	Ninguno

Al realizar la búsqueda de los requisitos de dominio en el catálogo TRIPODE, se encuentra la descripción del requisito de dominio POS Gestionar Acciones (Tabla 3), el cual se puede asociar con el requisito específico Gestión de Respaldo de Información (Ver Tabla 38).

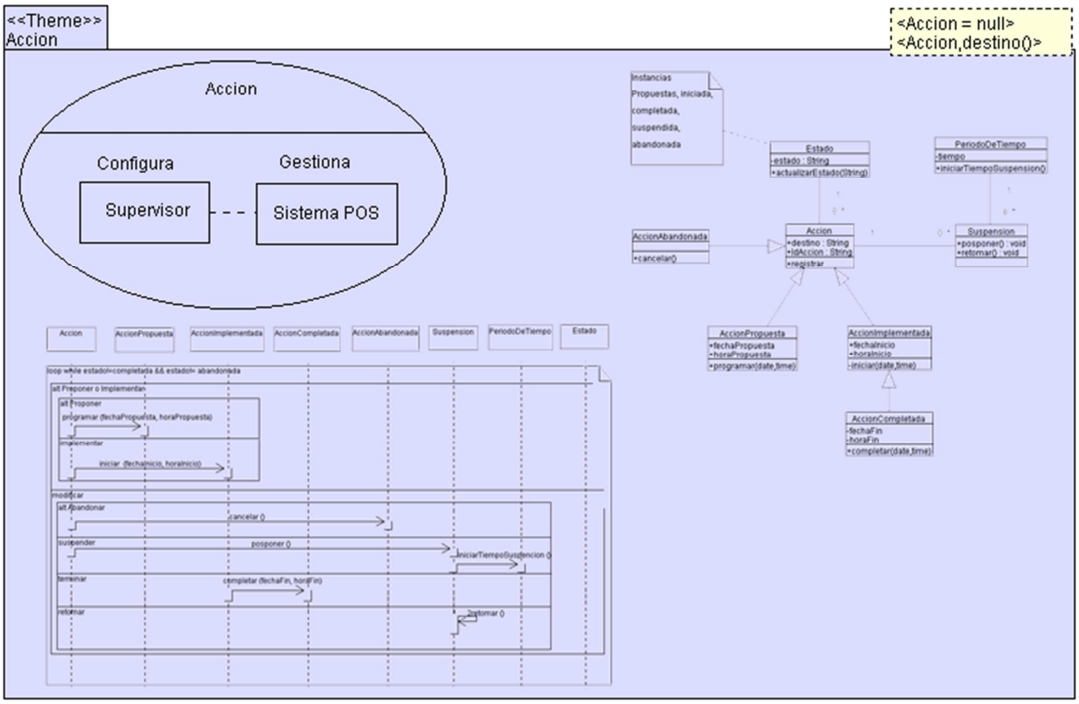
Tabla 38. Plantilla de requisito de dominio gestionar acciones

FRQ-004	Gestionar Acciones
Versión	1.0
Autores	Iván Cisneros, Gloria Erazo, Germán Peña
Fuentes	Supervisor de Caja
Descripción	El sistema POS deberá permitir programar e implementar acciones, las cuales podrán ser completadas, canceladas o suspendidas por un periodo de tiempo.
Dominios asociados	POS Comercio
Patrón solución	Accion
Estado	Pendiente de verificación
Estabilidad	Media
Comentarios	Ninguno

- Recuperar Patrón

Para recuperar el patrón que puede dar solución al requisito Gestión de Respaldo de Información, primero se realiza la búsqueda de los patrones asociados al requisito de dominio relacionado con el requisito específico. Como resultado de ésta búsqueda, el patrón, sugerido en el catálogo, para dar solución al requisito de dominio POS Gestionar Acciones, es el patrón Accion (Ilustración 65), por lo tanto se hace la recuperación de este patrón para dar inicio a la solución del requisito específico Gestionar Respaldo de Información.

Figura 65: Template patrón acción



Instanciar Patrón

Momento 1

En este primer momento, se hace el análisis de cada uno de los atributos y métodos de las clases Accion, AccionPropuesta, AccionImplementada, AccionCompletada, AccionAbandonada, Estado, Suspension y PeriodoDeTiempo, viendo cómo a través de estos elementos se resuelven las peticiones del requisito Gestionar Respaldo de Información.

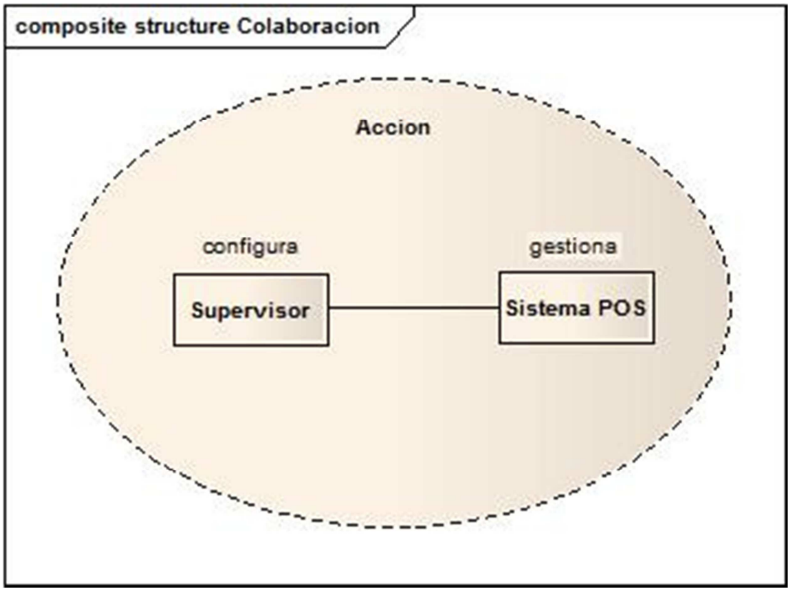
De acuerdo a los parámetros que el Signature del patrón plantea, se realizan los siguientes cambios:

Clase Accion por Backup

El atributo IdAccion de la Clase Accion por IdBackup

Modelo de alcance

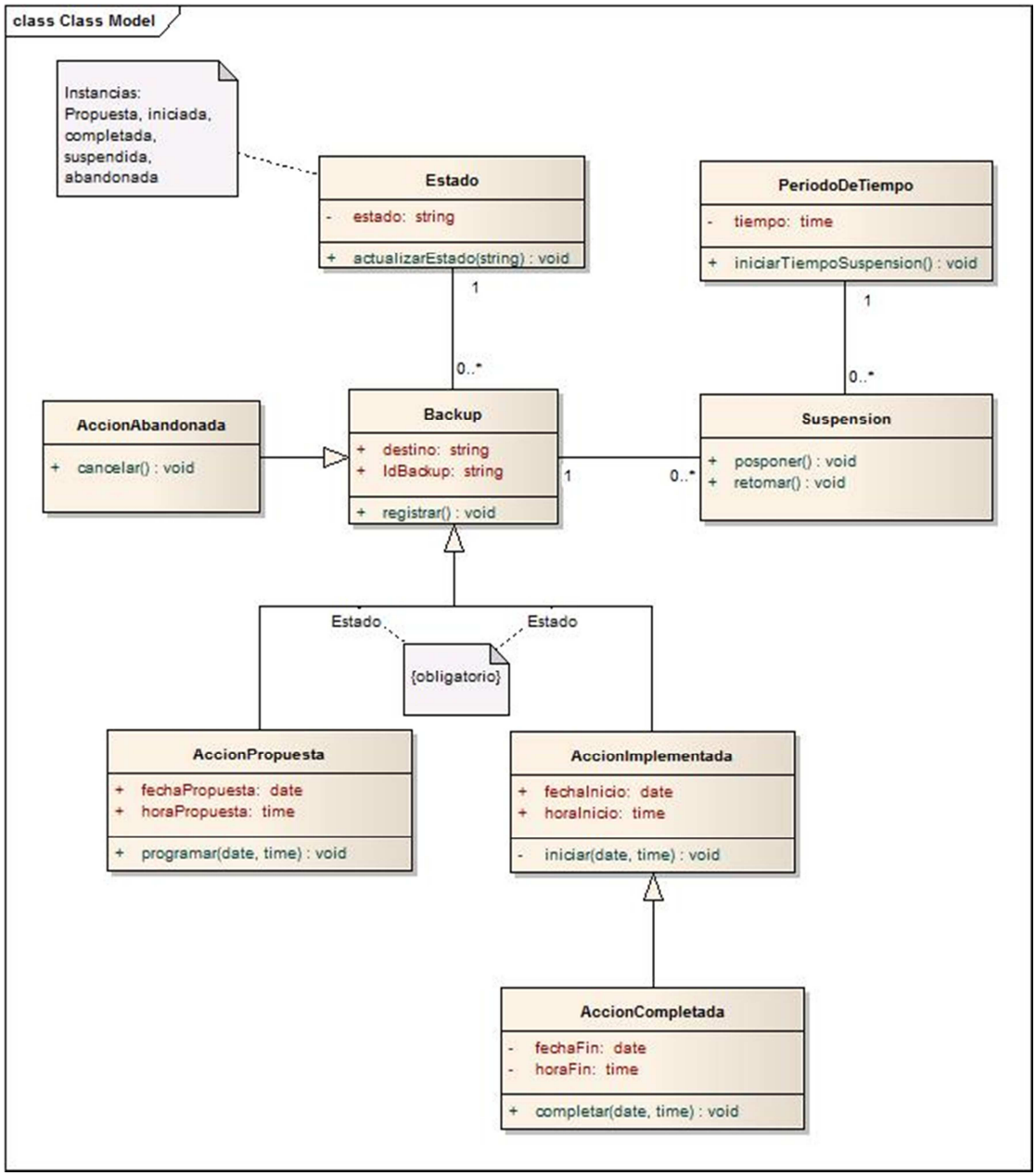
Figura 66: Colaboración requisito específico NFR001 gestionar respaldo de información: momento 1



Los cambios realizados al patrón durante el momento uno de la instanciación, no afectan el modelo de alcance debido a que no se adicionan nuevos actores, ni nuevos roles.

Modelo estructural

Figura 67: Diagrama de clases requisito específico NFR001 gestionar respaldo de información: momento 1



Clase BackUp: describe conceptualmente la copia de seguridad de las operaciones efectuadas en el POS.

Atributos

Destino: representa el lugar donde se va a realizar la copia de seguridad, puede ser interna (disco duro) o externa (medio magnético).

Id Backup: representa el código de identificación de la copia de seguridad.

Métodos:

Registrar: representa la operación que permite almacenar la información de la copia de seguridad que se va a realizar: fecha, hora, número de caja.

Clase acción Propuesta: describe conceptualmente la programación de la copia de seguridad.

Atributos:

Fecha Propuesta: representa la fecha para la cual se programa la copia de seguridad.

Hora Propuesta: representa la hora para la cual se programa la copia de seguridad.

Métodos:

Programar: representa la operación que permite programar la copia de seguridad.

Clase acción Implementada: describe conceptualmente la puesta en marcha de la creación de la copia de seguridad.

Atributos:

Fecha Inicio: representa la fecha en que se inicia la realización de la copia de seguridad.

Hora Inicio: representa la hora en que se inicia la realización de la copia de seguridad.

Métodos:

Iniciar: representa la operación que permite iniciar la creación de la copia de seguridad

Clase acción Completada: describe conceptualmente finalización de la creación de la copia de seguridad.

Atributos:

Fecha Fin: representa la fecha en que se acabó de realizar la copia de seguridad.
Hora Fin: representa la hora en que se acabó de realizar la copia de seguridad.

Métodos:
Completar: representa la operación que permite finalizar la copia de seguridad.

Clase accionaban donada: describe conceptualmente cancelación de la copia de seguridad.

Métodos:

Cancelar: representa la operación que permite anular la realización de la copia de seguridad.

Clase Estado: describe conceptualmente la etapa en la que se encuentran la copia de seguridad.

Atributos

Estado: representa la etapa en la que se encuentra la copia de seguridad: propuesta, iniciada, completada, suspendida o abandonada.

Métodos

Actualizar Estado: representa la operación que permite cambiar del estado anterior al estado actual de la copia de seguridad.

Clase Suspensión: describe conceptualmente la cesación temporal de la copia de seguridad.

Métodos:

Posponer: representa la operación que permite detener de manera temporal la realización de la copia de seguridad.

Retomar: representa la operación que permite continuar la realización de la copia de seguridad suspendida anteriormente.

Clase Periodo de tiempo: describe conceptualmente el intervalo de tiempo durante el cual se suspende la copia de seguridad.

Atributos:

Tiempo: representa la duración de la suspensión.

Métodos:

Iniciar tiempo suspensión: representa la operación que permite comenzar el tiempo de suspensión de la copia de seguridad.

Cardinalidad

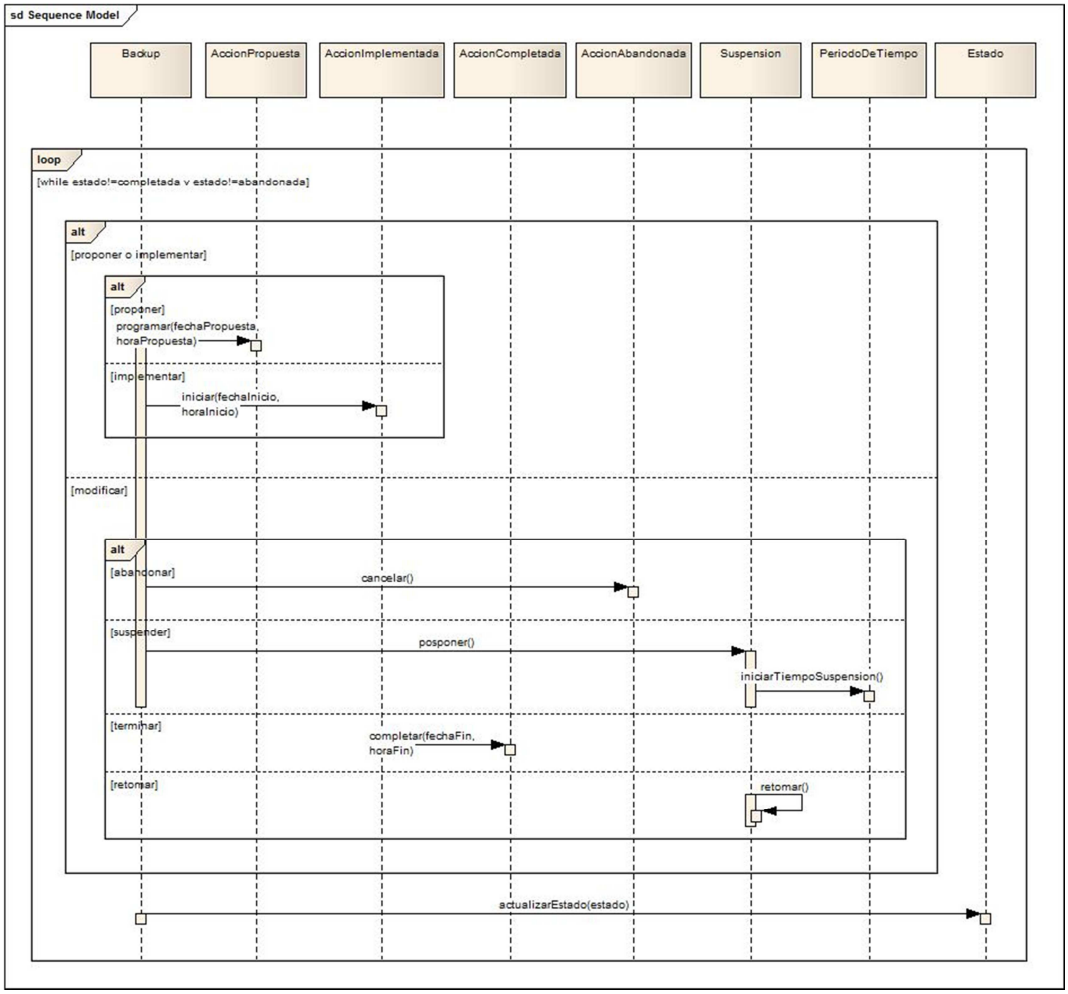
Una copia de seguridad puede ser suspendida cero o más veces, cero o más suspensiones pertenecen a una copia de seguridad.

Cero o más copias de seguridad deben tener un estado, un estado hace parte de cero o más copias de seguridad

Cero o más suspensiones deben tener un periodo de tiempo, un periodo de tiempo pertenece a cero o más suspensiones.

- Modelo de Comportamiento

Figura 68: Diagrama de secuencia requisito específico NFR001 gestionar respaldo de información: momento1



Objetos

Backup: representa la realización de la copia de seguridad de las transacciones realizadas en el POS de Supermercados YUCATA.

Acción Propuesta: representa la realización de la copia de seguridad que se programa en el POS de Supermercados YUCATA.

Acción Implementada: representa la realización de la copia de seguridad que se realiza en el POS de Supermercado YUCATA.

Acción Completada: representa la finalización de la creación de la copia de seguridad.

Acción Abandonada: representa la cancelación definitiva de la creación de la copia de seguridad.

Suspensión: representa el cese temporal de la creación de la copia de seguridad.
PeriodoDeTiempo: representa el intervalo de tiempo concreto durante el cual se suspende la creación de la copia de seguridad.

Estado: representa la etapa concreta en la que se encuentra la copia de seguridad.

Mensajes

Programar: representa el proceso de programación de la copia de seguridad.

Iniciar: representa el proceso de comienzo de creación de la copia de seguridad.

Este proceso puede ejecutarse habiendo o no programado la copia de seguridad.
Cancelar: representa el proceso de anulación de la creación de la copia de seguridad. Este proceso puede ejecutarse si la creación de la copia de seguridad ha sido programada o implementada anteriormente.

Posponer: representa el proceso de detención temporal de la copia de seguridad. Este proceso puede ejecutarse si la creación de la copia de seguridad ha sido programada o implementada anteriormente.

Iniciar tiempo suspensión: representa el proceso de inicialización del tiempo de suspensión de la copia de la seguridad. Este proceso se ejecuta una vez se suspenda la copia de seguridad.

Completar: representa el proceso de finalización de la copia de seguridad. Este proceso puede ejecutarse si la creación de la copia de seguridad se ha implementado anteriormente.

Retomar: representa el proceso de continuación de la copia de seguridad que fue suspendida.

Actualizar estado: representa el proceso de cambio de estado de la copia de seguridad. Este proceso se ejecuta cada vez que la copia de seguridad cambie de estado.

Registrar: representa el proceso de almacenamiento de la información de la copia de seguridad.

Ciclos

Acciones: representa el ciclo en el cual se realizan todos los procedimientos de programación, implementación, finalización, cancelación y suspensión de la acción. Este ciclo se ejecuta mientras que el estado de la copia de seguridad sea diferente de completado o abandonado.

Alternativas

Principal: representa el punto de flujo donde se selecciona una opción de operación sobre la copia de seguridad. Las opciones son: proponer o implementar, ó modificar la creación de la copia de seguridad que se propuso o implementó con anterioridad.

Proponer/Implementar: representa el punto de flujo donde se selecciona una opción de operación sobre la copia de seguridad. Las opciones son: proponer o implementar.

Modificar: representa el punto de flujo donde se selecciona una opción de operación sobre una copia de seguridad ya propuesta o implementada. Las opciones son: abandonar, suspender, terminar o retomar.

Momento 2

Los modelos de la solución al requisito Gestión de Respaldo de Información, no sufren ninguna modificación adicional a las realizadas en el momento uno de instanciación, debido a que con estas modificaciones ya se encuentra la solución completa al requisito tratado.

6.3 RETROALIMENTACIÓN DEL CATÁLOGO: PATRÓN ESTRATEGIA

El proceso de retroalimentación del catálogo de patrones de análisis, puede tener lugar cuando se presenta la situación en la que un requisito del caso de estudio tratado no se puede solucionar con los patrones existentes en el catálogo. En este caso, el analista debe realizar la solución específica, convirtiéndose en candidata a nuevo patrón. El siguiente paso es generalizar el requisito específico a un requisito de dominio, que para este caso es el dominio POS. Finalmente los diagramas de clase, de secuencia y la colaboración también deben generalizarse obteniendo de esta manera los modelos del nuevo patrón, de esta generalización también se obtienen los parámetros que conformarán el Signature del patrón.

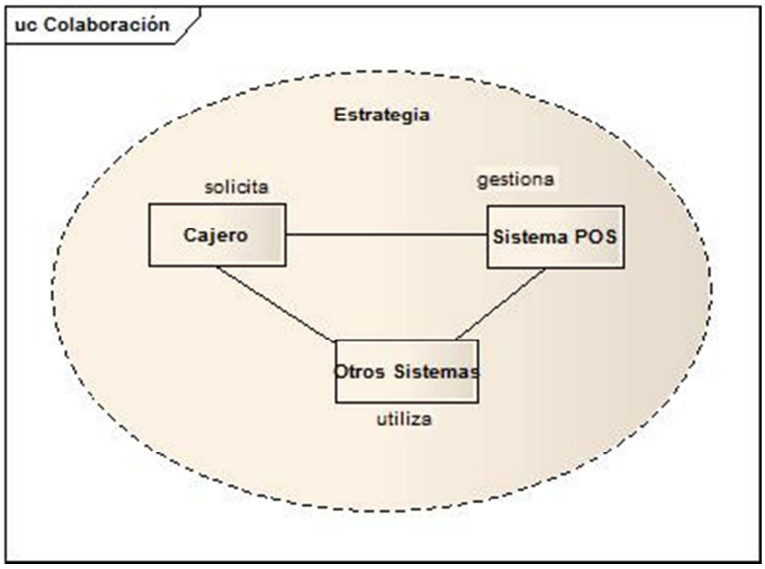
6.3.1 Caracterizar problemas por requisitos. En el modelado de la solución a los requisitos de Supermercados YUCATA, se presenta la situación descrita en el apartado anterior, con el requisito Gestionar Interfaces, el cual se describe en la plantilla que se muestra en la Tabla 39.

Tabla 39. Requisito específico gestionar interfaces - supermercados YUCATA

FRQ – 005	**Gestionar Interfaces**
Versión	1.1
Autores	Tania Araújo, Yurani Bolaños
Fuentes	Supervisor de Caja Supermercado YUCATA
Objetivos asociados	Mantener información coherente y confiable
Requisitos asociados	FRQ – 004 Gestionar Factura FRQ-008 Mantener disponible información válida
Descripción	El sistema debe permitir enviar la información que corresponda a cada interfaz, así: GESTIÓN INTERFAZ INVENTARIO Envío de información de la cantidad de productos para actualizar el inventario. Envío de información de la cantidad de productos a restablecer en los casos de devolución y cancelación de venta. GESTIÓN INTERFAZ CONTABILIDAD Envío de información de cartera para actualizar los abonos realizados por los clientes, en el caso del manejo del sistema de separado. Envío de información referente a entradas y salidas eventuales para la generación de informes de contabilidad Envío de información de pagos con bonos, para que sean registrados en cuentas por pagar. GESTIÓN INTERFAZ MERCADEO Envío de información de clientes y sus respectivas compras para poder generar estrategias de mercadeo y programas de fidelización Envío de información de ventas para la generación de reportes de proveedores y metas de venta general
Dominios asociados	General
Aspectual	-
Importancia	Vital
Estado	Validado
Estabilidad	Media
Comentarios	Ninguno

6.3.2 Modelar la solución específica

Figura 69: Colaboración requisito específico FRQ005 gestión de interfaces



Modelo de Alcance

Actores:

Cajero: representa el funcionario que opera la caja donde se realiza la solicitud de envío de información a sistemas externos.

Sistema POS: representa el software que gestiona los procesos que implican el envío de información a sistemas externos.

Otros Sistemas: representa a los sistemas externos que van a recibir la información producto del procesamiento de la solicitud realizada, en este caso se refiere al sistema contable, al sistema de mercadeo y al sistema de inventario.

Roles:

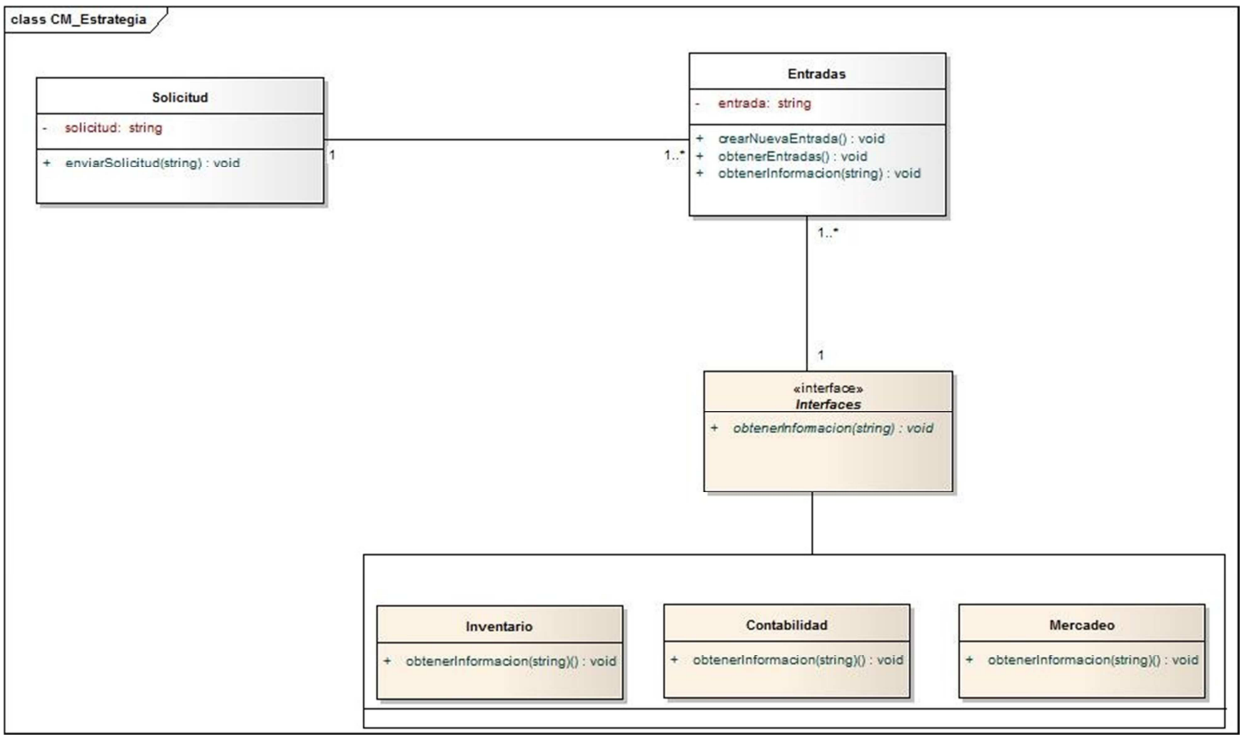
Solicitar: representa la función de realizar al sistema POS la petición de envío de información a los sistemas externos.

Utilizar: representa la función de recibir una información y ejecutar determinadas acciones con esta información.

Gestionar: representa la función de recibir, procesar, validar, almacenar la información referente a la petición realizada.

- Modelo Estructural

Figura 70: Diagrama de clases requisito específico FRQ005 gestionar interfaces



Clase Solicitud: Describe conceptualmente la acción por medio de la cual se realiza una petición para dar inicio al envío de información a las interfaces que comunican con sistemas externos relacionados con el sistema POS.

Atributos:

Solicitud: representa la petición realizada, en este caso envío de información a interfaces.

Métodos

Enviar Solicitud: representa la operación con la que se envía al sistema la petición de envío de información a las interfaces y se da inicio a este proceso. Este proceso se inicia automáticamente al ejecutarse la venta.

Clase Entradas: Describe conceptualmente cada una de las interfaces a las cuales se debe realizar el envío de la información.

Atributos:

Entrada: representa al nombre de la interfaz a la cual se debe realizar el envío de la información.

Métodos:

Obtener Entradas: representa la operación a través de la cual se obtienen todas las interfaces a las cuales se les debe enviar información.

Obtener Información: representa la operación a través de la cual se extrae la información particular a cada una de las interfaces a las se le hace el envío de información.

Crear Nueva Entrada: representa la operación a través de la cual se envía a la interfaz correspondiente la información requerida en el formato adecuado.

Clase Interfaces [abstract]: Esta clase funciona como una clase abstracta encargada de invocar a cada clase singleton (Inventario, Contabilidad, Mercadeo) de acuerdo a la interfaz a la cual se le deba realizar el envío de la información.

Métodos:

Obtener Información: es la representación del método que será invocado de acuerdo a cada interfaz que esté siendo tratada.

Clase Singleton Inventario: Con esta clase se representa el procesamiento particular de información de acuerdo a la interfaz de inventario.

Métodos:

Obtener Información: es la representación del método que obtiene la información que debe ser enviada a la interfaz de inventario.

Clase Singleton Contabilidad: Con esta clase se representa el procesamiento particular de información de acuerdo a la interfaz de contabilidad.

Métodos:

Obtener Información: es la representación del método que obtiene la información que debe ser enviada a la interfaz de contabilidad.

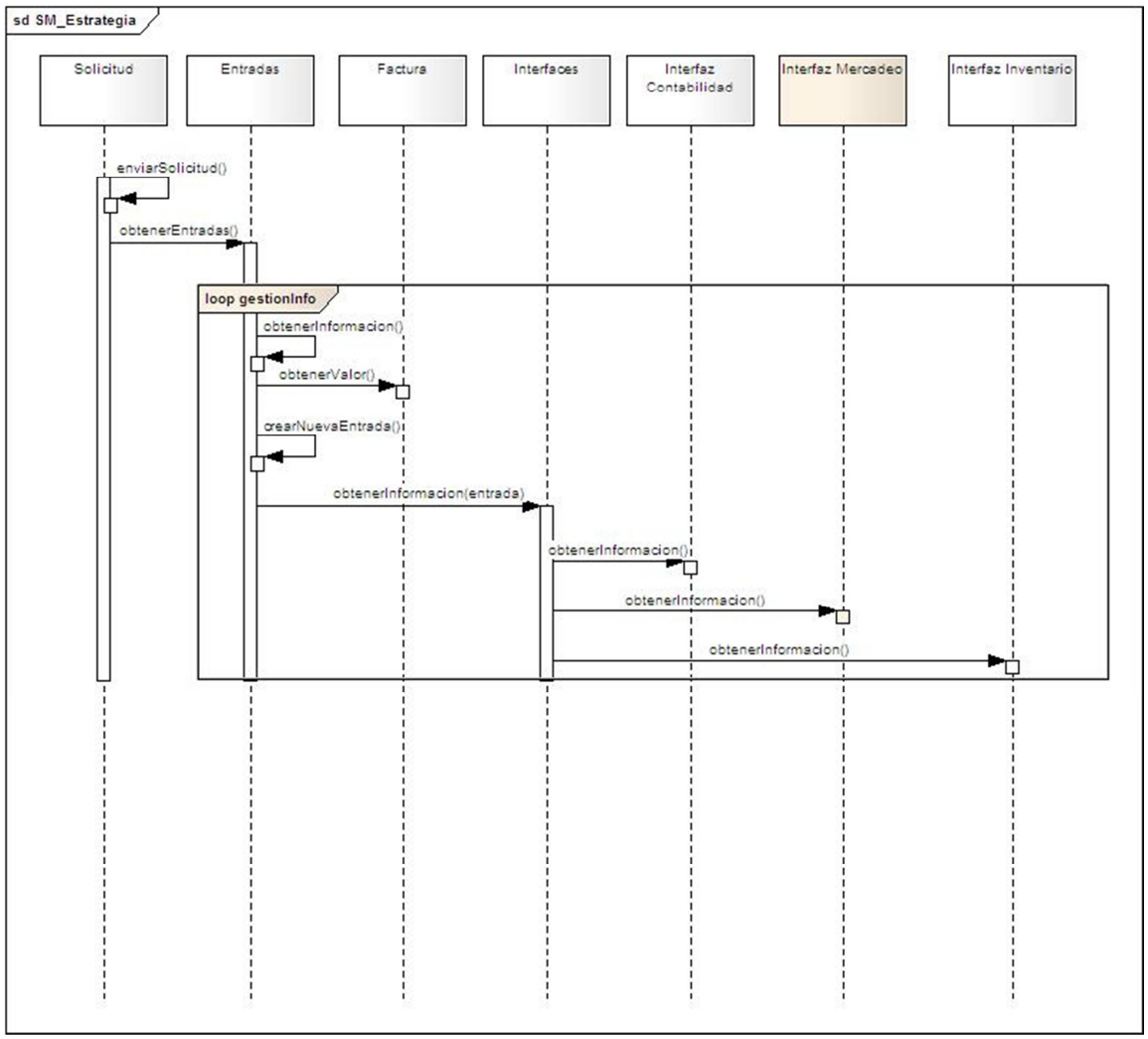
Clase Singleton Mercadeo: Con esta clase se representa el procesamiento particular de información de acuerdo a la interfaz de mercadeo.

Métodos:

Obtener Información: es la representación del método que obtiene la información que debe ser enviada a la interfaz de mercadeo.

Modelo de Comportamiento

Figura 71: Diagrama de secuencia requisito específico FRQ005 gestionar interfaces



Objetos

Solicitud: representa la petición para que se haga envío de la información que corresponda, a los sistemas externos relacionados con el sistema POS, en este

caso, interfaz de mercadeo, interfaz contable e interfaz de inventario. Esta petición se dispara en el momento en que una venta es ejecutada.

Entradas: representa cada una de las interfaces a las que se les debe enviar información, es decir inventario, mercadeo y contabilidad.

Interfaces: representa el elemento a través del cual se puede acceder a los objetos Inventario, Mercadeo y Contabilidad.

Factura: representa el objeto a través del cual se puede adquirir la información necesaria sobre la venta y que es requerida para mantener la coherencia de información entre el Sistema POS y los sistemas de Inventario, Mercadeo y Contabilidad.

Interfaz Inventario: representa la interfaz a través de la cual se le hace envío al sistema de inventario de la información que éste sistema requiera y que se genera cada vez que se ejecuta una venta, para poder tener una sistema con información confiable y coherente.

Interfaz Mercadeo: representa la interfaz a través de la cual se le hace envío al sistema de mercadeo de la información que éste sistema requiera y que se genera cada vez que se ejecuta una venta, para poder tener un sistema con información confiable y coherente.

Interfaz Contabilidad: representa la interfaz a través de la cual se le hace envío al sistema de contabilidad de la información que éste sistema requiera y que se genera cada vez que se ejecuta una venta, para poder tener un sistema con información confiable y coherente.

Mensajes

Enviar Solicitud: representa el proceso de realizar la solicitud al sistema POS, para que se envíe la información correspondiente a cada venta realizada, a las interfaces: Mercadeo, Contabilidad, Inventario.

Obtener Entradas: representa el proceso de obtener las interfaces a las que se les deberá hacer el envío de información relacionada con cada venta ejecutada.

Obtener Información: representa el proceso de realizar las operaciones que son particulares para cada una de las interfaces tratadas.

Obtener Valor: representa el proceso a través del cual se puede obtener información requerida para el procesamiento particular de la entrada actual.

Crear Nueva Entrada: representa el proceso a través del cual se envía a la interfaz correspondiente, la información extraída en la consulta realizada.

Gestionar el Catálogo de Patrones de Análisis

Generalizar el Requisito Específico a Requisito de Dominio.

Para esto se comienza eliminando de la plantilla del requisito Gestionar Interfaces los campos: Objetivos asociados y Requisitos asociados, acondicionando esta plantilla a la plantilla propuesta en el capítulo anterior para los requisitos de dominio. La descripción del requisito se debe hacer lo más general posible, de tal manera que sea aplicable en diferentes tipos de sistemas POS.

El resultado de este proceso de generalización del requisito Gestionar Interfaces, se muestra en Tabla 40.

Tabla 40. Plantilla de requisito de dominio FRQ007 gestionar envío de información

FRQ-007	**Gestionar Envío de Información**
Versión	1.0
Autores	Tania Araújo. Yurani Bolaños
Fuentes	Supervisor de Caja
Descripción	El sistema deberá permitir enviar información a los sistemas externos relacionados con el sistema POS, a través de interfaces de comunicación, por ejemplo: envío de información a sistema contable, envío de información para manejo de inventario.
Dominios asociados	POS Comercio
Patrón solución	Estrategia
Estado	Validado
Estabilidad	Media
Comentarios	Ninguno

Tabla 41. Tarjeta CRC de asignación de responsabilidades para el patrón estrategia

RESPONSABILIDAD	CLASE RESPONSABLE	ACTOR RESPONSABLE
Ingreso de información	Solicitud	Cajero/Sistemas POS
Envío de información	Entradas	Sistema POS
Filtro de información	Método	Sistema POS
Envío de información adicional	ObjetoX	Sistema POS
Utiliza la información	MétodoY	Otros Sistemas

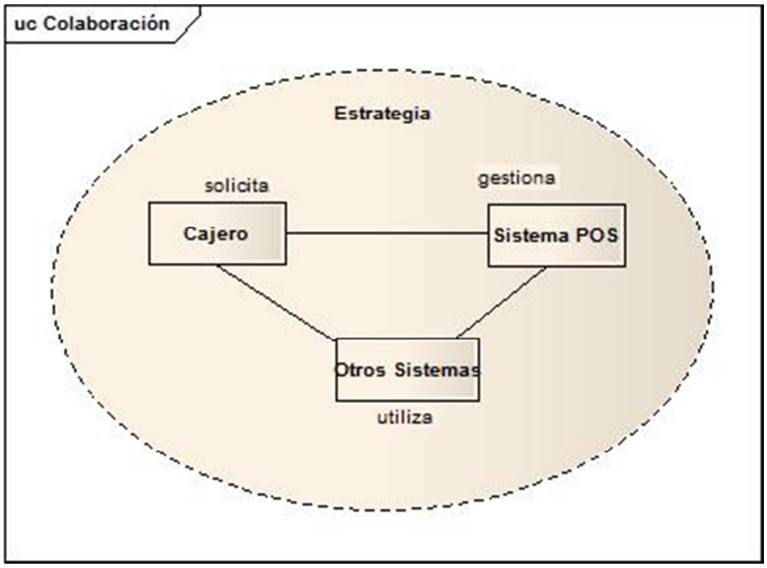
- Generalizar los Modelos

Modelo de Alcance

Para la definición del modelo de alcance utilizamos la tarjeta CRC de asignación de responsabilidades, mostrada en la Tabla 41.

El diagrama de colaboraciones resultante se muestra en la Figura 72.

Figura 72: Colaboración patrón estrategia



La descripción de los elementos de este diagrama se presenta a continuación:

Actores

Cajero: representa el funcionario que opera la caja donde se realiza la solicitud.
Sistema POS: representa el software que gestiona los procesos que implican una solicitud.

Otros Sistemas: representa las diferentes aplicaciones software que pueden hacer uso de la información resultante del procesamiento de las entradas correspondientes a la solicitud realizada.

Roles

Solicitar: representa la función de realizar la petición al sistema POS
Utilizar: representa la función de recibir una información y ejecutar determinadas acciones con esta información.
Gestionar: representa la función de recibir, procesar, validar, almacenar la información referente a la petición realizada.

Modelo Estructural

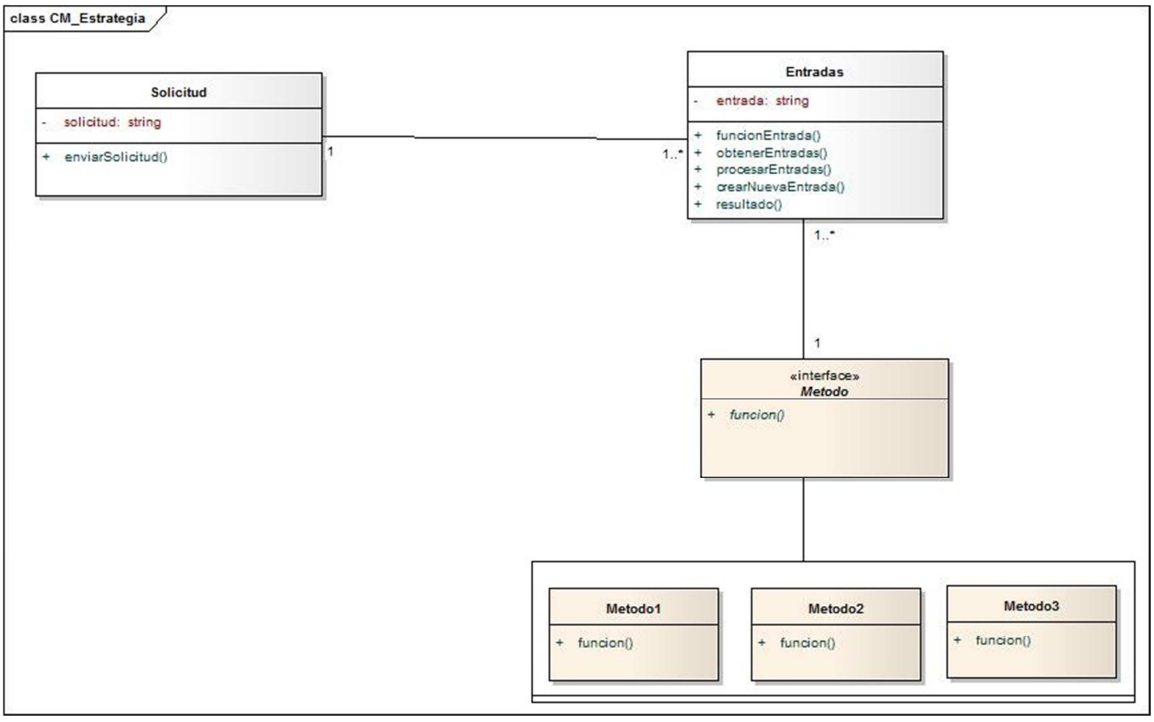
En este modelo se realizan las siguientes modificaciones para realizar la generalización:

- Clase Singleton Interfaz Inventario por Clase Singleton Método1
- Clase Singleton Interfaz Contabilidad por Clase Singleton Método2
- Clase Singleton Interfaz Mercadeo por Clase Singleton Método3
- Clase Abstracta Interfaces por Clase Abstracta Método
- Inventario.obtenerInformacion por Metodo1.funcion
- Contabilidad.obtenerInformacion por Metodo2.funcion
- Mercadeo.obtenerInformacion por Metodo3.funcion
- Interfaces.obtenerInformacion por Metodo.funcion
- Entradas.obtenerInformacion por Entradas.funcion
- Se adiciona el método procesarEntrada() en la clase Entradas. Esto con el fin de poder dar lugar a un procesamiento general para todas las entradas iniciales antes de utilizar la clase abstracta Metodo.
- Se adiciona el método resultado() en la clase Entradas, para poder representar el retorno de un posible resultado después de haber enviado

El objetivo de estos cambios es dejar el patrón lo más general posible, de tal manera que no se limite al envío de información, sino también que se pueda utilizar para situaciones en donde se dependa de ciertas variables de entrada para tomar una decisión, como por el ejemplo, manejo de promociones, para que de

acuerdo al producto que haya ingresado en la transacción de compra se aplique determinado porcentaje, o en la conversión de divisas.

Figura 73: Diagrama de clases patrón estrategia



Los elementos que conforman este diagrama de clases se describen a continuación:

Clase Solicitud: describe conceptualmente la acción por medio de la cual se realiza una petición para que se dé inicio a una o más acciones.

Atributos

Solicitud: representa la petición realizada.

Métodos

Enviar Solicitud: representa la operación con la que se envía al sistema la petición y se da inicio a un proceso determinado.

Clase Entradas: describe conceptualmente las diferentes entradas que se deben procesar de acuerdo a la solicitud realizada.

Atributos

Entrada: representa a cada uno de los parámetros que deben ser tratados de acuerdo a la solicitud realizada.

Métodos

Obtener Entradas: representa la operación a través de la cual se obtienen todas las entradas que deben ser tratadas de acuerdo a la solicitud realizada.

Procesar Entrada: representa la operación que permite realizar una serie de acciones que se pueden aplicar de manera general a cada una de las entradas que se hayan obtenido de la solicitud realizada.

Función Entrada: representa la operación a través de la cual se pueden llevar a cabo acciones particulares dependiendo de la entrada que esté siendo tratada.

Crear Nueva Entrada: representa la operación a través de la cual se envía al sistema o módulo externo, la información requerida en el formato adecuado.

Clase Método [abstract]: esta clase actúa como una interfaz a través de la cual se puede acceder a un conjunto de clases singleton (Metodo1, Metodo2, Metodo3), en donde cada una de ellas ha sido particularizada para dar respuesta a solicitudes específicas.

Métodos:

Función: este método está definido de manera particular en cada una de las clases singleton que lo contiene (Metodo1, Metodo2, Metodo3). Sus acciones son previamente definidas para que puedan dar respuesta a las necesidades de cada una de las entradas que están siendo tratadas.

Cardinalidad:

Un objeto de la clase Solicitud puede estar asociada a uno o más objetos de la clase Entradas.
Uno o más objetos de la clase Entrada puede estar asociada a un objeto de la clase Metodo.

Modelo de Comportamiento

Las modificaciones realizadas en este modelo son consecuencia de los cambios realizados en el modelo estructural.

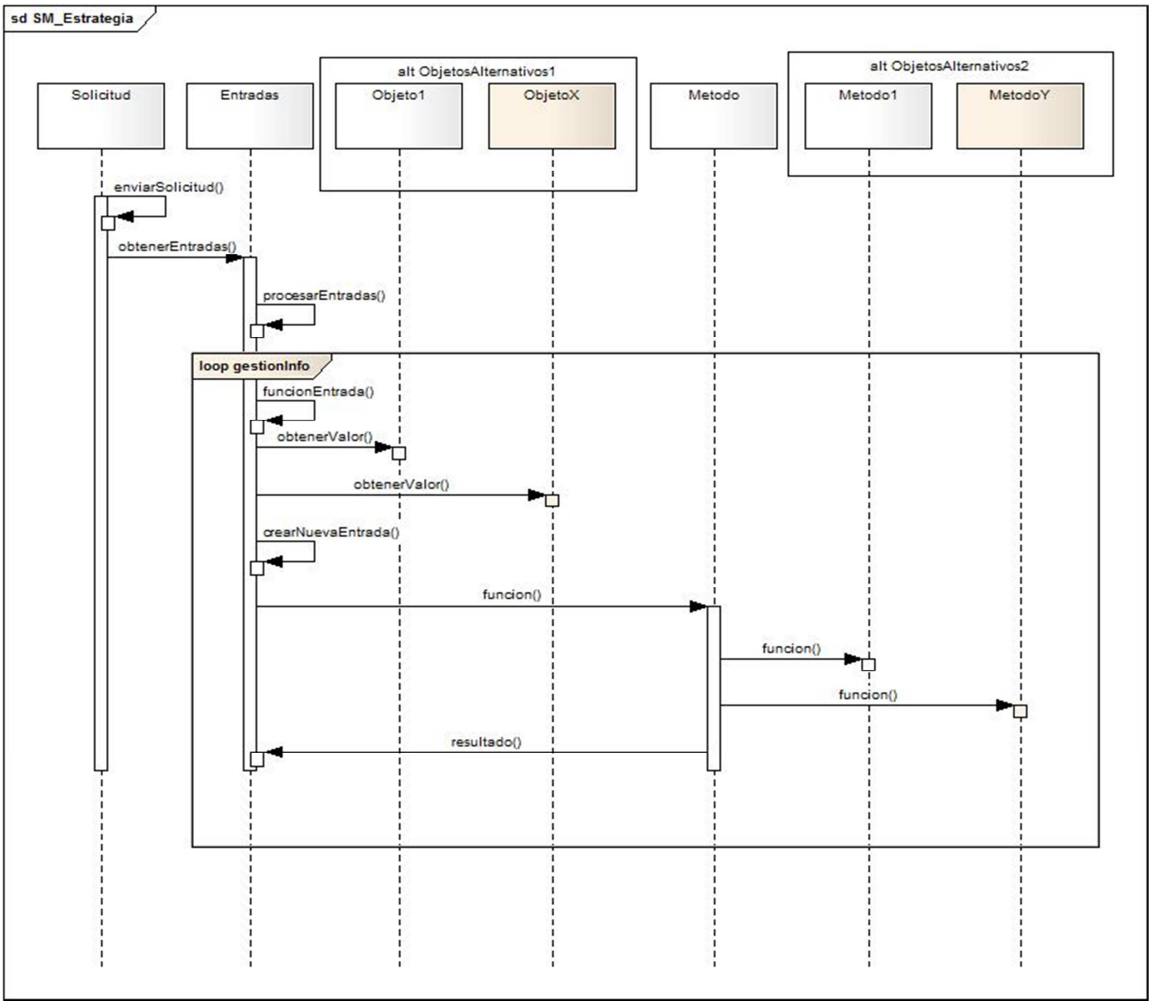
Se elimina el objeto Factura y se reemplaza por los objetos alternativos denominados Objeto1 y Objeto x, representando que puede existir un número variable de este tipo de objetos.

Se eliminan los objetos Interfaz Inventario, Interfaz Contabilidad, Interfaz Mercadeo y se reemplazan por los objetos alternativos Metodo1 y Método, representando que puede existir un número variable de este tipo de objetos.

Se adiciona el mensaje procesar Entrada () en el objeto Entradas.

Se adiciona el mensaje resultado (), en el objeto Entradas.

Figura 74: Diagrama de secuencia patrón estrategia



Los elementos que conforman este diagrama de secuencia se describen a continuación:

Objetos

Solicitud: representa la petición que se realiza al sistema POS para dar inicio a un determinado proceso.

Entradas: representa cada una de las entradas que se deben procesar de acuerdo a la solicitud hecha.

Método: representa el elemento a través del cual se puede dar un tratamiento particular a cada una de las entradas.

Objeto1 – ObjetoX: representan cualquier objeto a través del cual se pueda adquirir información que sea necesaria para procesar la entrada actual.

Metodo1 – MetodoY: representan los objetos que adquieren el comportamiento particular de acuerdo a la entrada que esté siendo tratada.

Mensajes

Enviar Solicitud: representa el proceso de enviar la solicitud al sistema POS.

Obtener Entradas: representa el proceso de obtener las entradas correspondientes a la solicitud realizada.

Procesar Entrada: representa el proceso de realizar operaciones que se pueden aplicar de manejar general a las entradas correspondientes a la solicitud realizada.

Función Entrada: representa el proceso de realizar las operaciones que son particulares de cada entrada de la solicitud realizada.

Obtener Valor: representa el proceso a través del cual se puede obtener información requerida para el procesamiento particular de la entrada actual.

Resultado: representa el proceso a través del cual se obtiene la información resultante del procesamiento particular de cada una de las entradas de la solicitud realizada.

Crear Nueva Entrada: representa el proceso a través del cual se envía al sistema o módulo externo, la información requerida en el formato adecuado.

Ciclos

Gestionar Info: representa el ciclo en el cual se hace el procesamiento que le corresponde a la entrada actual. Este ciclo abarca los procedimientos función Entrada, obtener Valor, crear Nueva Entrada y resultado.

Alternativas

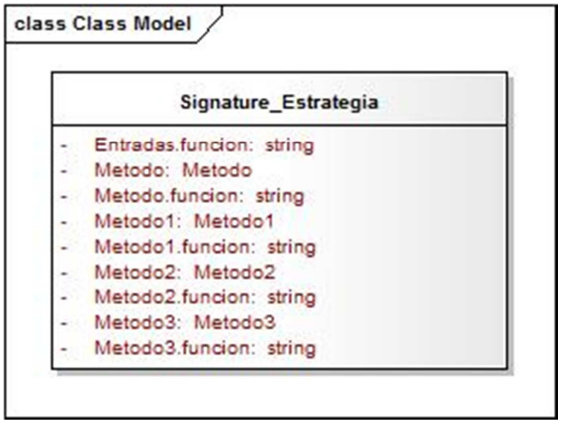
ObjetosAlternativos1 y ObjetosAlternativos2: representan el dinamismo a través del cual, dependiendo de la entrada que esté siendo tratada se requiera o no obtener algún dato extra proveniente de un objeto externo representado por Objeto1 y ObjetoX, para seguir con el procesamiento de la misma; así como también representa la posibilidad de aplicar un proceso particular a cada entrada, acción que es representada en el diagrama con los objetos Metodo1 y MetodoY

Signature

Después de realizar un análisis de cada uno de los elementos que componen el patrón Estrategia, se decide dejar como parámetros los siguientes elementos:

- Nombre de la Clase Método
- Metodo.funcion
- Entradas.funcionEntrada
- Nombre de la Clase Metodo1
- Metodo1.funcion
- Nombre de la Clase Metodo2
- Metodo2.funcion
- Nombre de la Clase Metodo3
- Metodo3.funcion

Figura 75: Signature patrón estrategia



- Catalogación del Patrón

Finalmente se presenta la descripción del patrón resultante de este proceso de retroalimentación, siguiendo la plantilla de patrones del catálogo de patrones propuesta en el capítulo anterior.

Nombre: Estrategia

Dominio: POS, Comercio, Cuentas Bancarias, Sistemas Tributarios

Descripción: El patrón de análisis Estrategia describe una propuesta de solución para que se realice una acción de acuerdo a una solicitud hecha, la cual puede o no generar entradas que se reciben como parámetros que se procesarán de manera individual. Una vez se hayan recibido las entradas necesarias se inicia la de toma de decisión, solo una de las posibles acciones a realizarse puede ejecutarse por cada entrada procesada, es decir estas acciones son excluyentes entre sí. Las entradas recibidas varían de acuerdo a la solicitud que haya sido realizada, al tratar cada una de estas entradas es posible que se haga necesario obtener nuevos parámetros que van a depender de la entrada que esté siendo tratada, así como también del procesamiento de cada entrada puede generarse o no un resultado particular. Finalmente, del procesamiento realizado a cada entrada se puede originar información que servirá como entrada a otro subsistema.

- Usos conocidos:

Sistemas bancarios – manejo de cuentas de clientes
Inventarios y depósitos – gestión de entradas y salidas
Aerolíneas – manejo de cuentas de clientes

Patrones relacionados:

Account [3]
Summary account [3]
Memo account [3]
Posting rules [3]

Requisitos de dominio

Manejo de descuentos y promociones
Integración sistemas de pago con tarjeta
Conversión a moneda extranjera
Validar forma de pago

7. RESULTADOS DEL PROCESO DE APLICACIÓN DE TRIPODE

En la Tabla 42 se muestran los resultados de la validación del uso de TRIPODE para modelar el Sistema POS de Supermercados YUCATA, de acuerdo al modelo de Ingeniería de Aplicación propuesto por ESKEMA.

Los resultados están categorizados de acuerdo a cada uno de los procesos llevados a cabo para el modelado de la solución específica.

Tabla 42. Resultados validación uso eskema y proceso de aplicación de TRIPODE

Actividad	Elemento	Objetivo de elemento	Observaciones
Modelar Solución Especifica	Caracterizar problemas por requisitos	Identificar los requisitos que especifican el problema, aquellos que son comunes al dominio y aquellos que son particulares al problema.	Para la identificación de los requisitos comunes al dominio, se hace uso de la plantilla que nos ofrece el requisito del dominio POS, la cual facilita la asociación con el requisito del caso de estudio por que esta descripción parte de un concepto general.
	Recuperar Patrón	Busca extraer del catálogo el patrón que resuelve un requisito de dominio y poder reutilizarlo con el requisito del caso de estudio.	Para la recuperación y posterior uso del patrón resulta útil la etapa de caracterización debido a que permite identificar de manera ágil y hacer uso del patrón que posiblemente dará solución al requisito del caso de estudio.
	Instanciar Patrón	Buscar la forma de especializar el patrón de análisis convirtiéndolo en la solución del problema específico, adaptándolo para que cumpla con el requisito del caso de estudio	La principal ventaja que ofrece este proceso es que si en el momento 1 de instanciación no se logra satisfacer las exigencias del requisito, ESKEMA ofrece la posibilidad de modificar el patrón en el momento 2, a conveniencia del usuario para poder obtener la solución, esto siempre y cuando no se altere la esencia del patrón. En este proceso resulta útil tener las tres vistas que nos ofrece el patrón, teniendo como resultado unas representaciones claras del

			alcance del patrón.
	Modelar patrón	Busca generar los modelos que responden a requisitos para los cuales el catálogo no provee un patrón solución. El requisito del problema y el modelo de la solución son candidatos para retroalimentar el catálogo	Este elemento logra una retroalimentación del catálogo de patrones debido a que permite identificar, modelar y posteriormente generalizar un requisito que después de un análisis se logra concluir que es candidato para realizar un nuevo patrón.
Representación	Diagrama de Clases	Representar la estructura conceptual del patrón de análisis indicando clases, atributos y relaciones.	Este elemento cumple con su objetivo ya que permite representar de una manera clara y adecuada una solución conceptual al requisito del caso de estudio.
	Diagrama de Secuencia	Describir la interacción y comunicación de los objetos del patrón de análisis a través del tiempo.	Permite describir y visualizar de una manera apropiada, como a través del tiempo interactúan los objetos para cumplir con lo descrito en el requisito.
	Colaboración	Representar de manera adecuada como los roles asociados al patrón de análisis se relacionan entre sí.	Permite representar cómo los actores asociados al patrón solución se relacionan entre sí y con el contexto, a través del rol que desempeñan. Además para la actividad Modelar la solución del proceso de retroalimentación, resulta muy útil el uso de la tarjeta CRC porque facilita la creación de la colaboración.
	Template	Servir de contenedor de los diagramas que representan los modelos que definen el patrón de análisis.	Este elemento cumple con su objetivo ya que permite contener los modelos que definen al patrón y muestra sus elementos parametrizables, dando al usuario una idea general de la funcionalidad del patrón.

CONCLUSIONES

La aplicación del catálogo TRIPODE para el modelado del sistema POS del caso de estudio Supermercados YUCATA, permite poner en práctica la reusabilidad de activos software para dar solución a requisitos funcionales del sistema, lo cual se ve reflejado en la disminución de tiempo y costo en la fase de análisis del desarrollo del aplicativo software deseado. Además permite demostrar la funcionalidad práctica tanto de MORENA como del proceso de Ingeniería de Aplicación propuesto por ESKEMA e identificar posibles mejoras de estos modelos.

El caso de estudio Supermercados YUCATA sirvió como el escenario adecuado para hacer uso de TRIPODE, pues al ser éste un Tipo de POS, está dentro del dominio para el cual el catálogo está diseñado haciendo posible probar que TRIPODE ofrece la posibilidad de aplicar el concepto de reutilización de artefactos software.

La aplicación de una metodología formal para la definición de los requisitos del sistema deseado facilita el tratamiento de los conflictos existentes entre las necesidades de los clientes y la percepción de estas necesidades por parte de los desarrolladores, y permite además, hacer un acotamiento del sistema que se ve reflejado en la definición de un sistema que cubra las verdaderas necesidades del cliente.

En cuanto a la metodología utilizada en este proyecto para la Ingeniería de Requisitos, que corresponde al modelo de Amador Durán se puede afirmar que es una propuesta importante para la comunidad de Ingeniería de Software, pues no solo presenta un paradigma formal para la definición de requisitos, sino que también propone elementos de representación con los cuales se puede dar un formato estándar a conceptos básicos dentro de este proceso. Además es un modelo iterativo, que no solo se puede aplicar en la fase inicial del proceso de desarrollo de software, sino que puede utilizarse en cualquier fase de este proceso garantizando una precisa acotación de las funcionalidades del sistema.

La aplicación del Modelo de Amador Durán en el caso de estudio Supermercados YUCATA fue de gran utilidad en el proceso de definición de los requisitos. El producto principal de este proceso, que es la definición de una versión final de los requisitos, fue la entrada clave para dar un inicio exitoso a la primera actividad del proceso propuesto por ESKEMA para la Ingeniería de Aplicación, en la fase de Análisis del Sistema POS y que corresponde a la Caracterización de los Requisitos.

El modelado del Sistema POS utilizando TRIPODE en la fase de análisis, para el caso de estudio Supermercados YUCATA, permitió realizar una abstracción clara de los elementos necesarios para poder diseñar un sistema que tenga el comportamiento deseado y que cumpla con sus objetivos, sin embargo no se pudo realizar el modelado de todo el sistema, pues TRIPODE cuenta con un número limitado de patrones.

El modelado del sistema POS fue el proceso principal para la validación no sólo de TRIPODE, sino también de la metodología para la aplicación y gestión del catálogo propuesta por ESKEMA. Las bondades ofrecidas tanto por el catálogo como por ESKEMA, no se pudieron aprovechar en su totalidad debido a que estos elementos no se encuentran integrados en una herramienta CASE.

Una fortaleza identificada al modelar el Sistema POS utilizando TRIPODE, es que con este catálogo se pueden manejar tres vistas importantes en el análisis del sistema a través de los tres modelos en los que están definidos cada uno de los patrones, y a su vez permite analizar el comportamiento general de cada una de las funcionalidades modeladas, al integrar estos tres modelos en el template.

La utilización de TRIPODE en el modelado del sistema POS de Supermercados YUCATA, permitió aplicar el concepto de reuso de activos software, pues los patrones de análisis además de estar diseñados con un enfoque aspectual, están definidos en templates parametrizables; estas dos características permitieron particularizar el patrón a funcionalidades distintas.

El modelo presentado por ESKEMA es un modelo apto, eficiente y funcionalmente adecuado para la gestión de catálogos de patrones de análisis, en la etapa de aplicación y uso de este tipo de artefactos software. Además propone un modelo de retroalimentación del catálogo bastante útil que permite aprovechar y generalizar las soluciones propuestas para un caso particular, en el que el catálogo no brinda una solución.

RECOMENDACIONES

Realizar el Modelado de la Solución Específica para el Sistema POS de Supermercados YUCATA, de acuerdo al proceso de Ingeniera de Aplicación propuesto por ESKEMA, se detecta una necesidad de que en ESKEMA se describa y precise qué cambios son permitidos realizar a los patrones en el momento 2 de la instanciación, de tal forma que se garantice que con los cambios efectuados por el analista no se pierda la esencia conceptual del patrón.

Definir el Modelo de Alcance a través de Diagramas de Casos de Uso, pues aunque las Colaboraciones, cumplen con el propósito de este modelo, los Casos de Uso también ofrecen la posibilidad de representar cómo el patrón de análisis interactúa con su contexto y con los usuarios de acuerdo a los roles que asumen.

TRABAJOS FUTUROS

Como complemento a la presente investigación se propone ahondar en la búsqueda y análisis de herramientas CASE disponibles, para determinar qué herramienta CASE brinda los soportes necesarios para la implementación de templates parametrizables.

Una vez detectada la herramienta CASE que soporte el uso de templates parametrizables, se propone la implementación del Catálogo de Patrones de Análisis TRIPODE, junto con la implementación del modelo de gestión para el catálogo propuesto por ESKEMA.

Se propone robustecer TRIPODE, incrementando el número de patrones de análisis y ampliando su dominio de alcance al dominio Comercio, que incluye al dominio POS para el cual el catálogo está definido actualmente.

BIBLIOGRAFIA

[1] Barón, A. ESKEMA – Esquema para la Gestión de Catálogos de Patrones de Análisis utilizando el enfoque aspectual. Maestría en Ingeniería Informática, Universidad Eafit, Medellín Colombia. 2010.

[2] Software Engineering Institute. Internet: (www.sei.cmu.edu/domain-engineering/domain_anal.html <http://www.sei.cmu.edu/domain-engineering/domain_anal.html>)

[3] Fowler, Martin. Analysis Patterns, Reusable objects models, Addison Wesley, 1997.

[4] Gamma, E., R. Helm R Johnson, and J. Vlissides. Design Patterns: Elements of Reusable Objects-Oriented Software Reading, MA: Addison Weley, 1995.

[5] Coplien J.O. “Agenerative Development-process Pattern Language”. In Pattern Language of program design. J.O. Coplien and D.C. Schmidt, ED. Reading, MA:Addison-Weley, 1995.

[6] Isla, J. L. “Modelado conceptual de sistemas cooperativos en base a patrones en amenities”. Universidad de Granada, 2007

[7] Hay, D. Datta Model Patterns: Conventions of Thought. New York: Dorset House, 1996

[8] Booch, G. Rumbaugh, J. Jacobson, I. El Lenguaje Unificado de modelado, Pearson Rentice Hall, 2006.

[9] Object Management Group, Unified Modeling Language: Superstructure, Versión 2.1.1, 2007

[10] Ortín, Maria José; Garcia Molina, José. “Modelado basado en roles con UML”.Departamento de Informática, Lenguajes y Sistemas. Facultad de Informática - Universidad de Murcia.

[11] Durán, Amador. Metodología para la Elicitación de Requisitos de Sistemas Software. Universidad de Sevilla, 2000