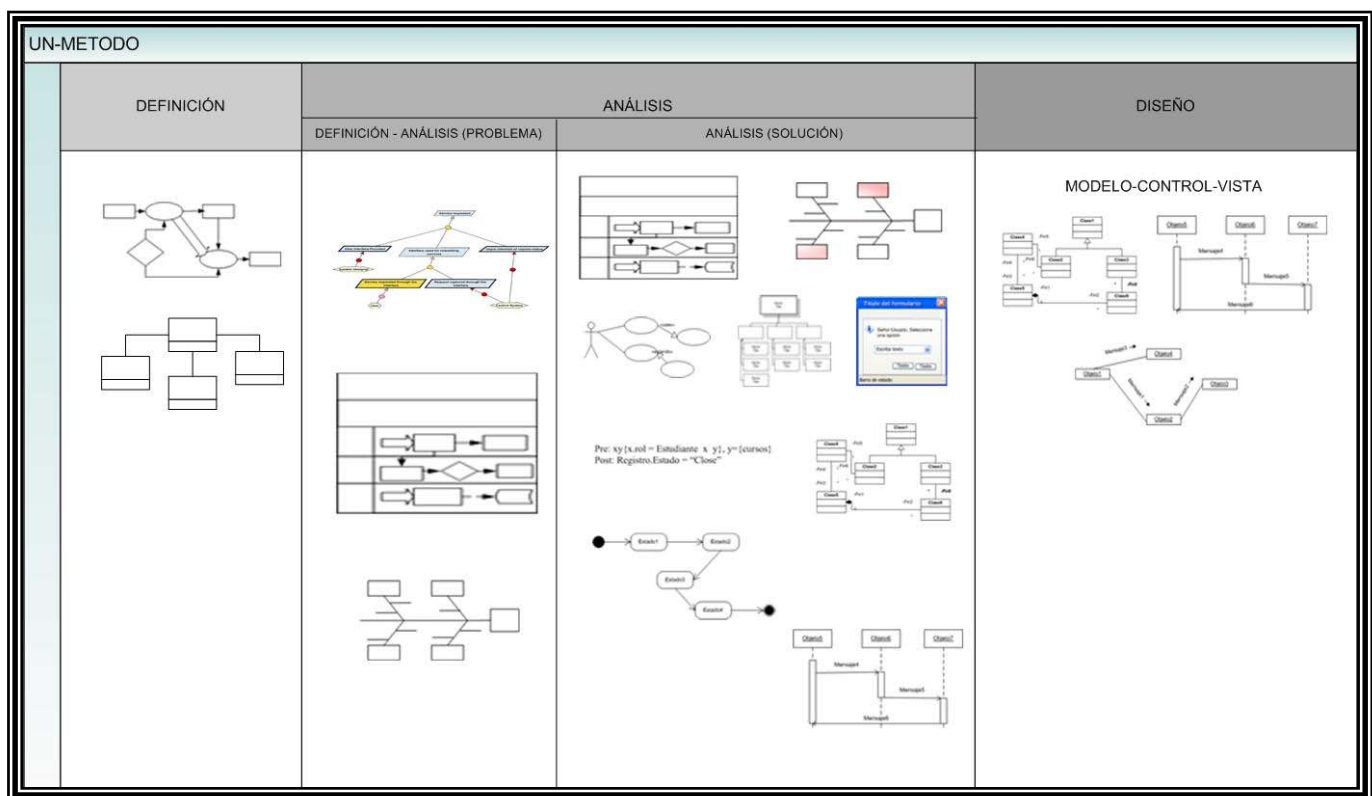


# UN MÉTODO para la Elicitación de Requisitos de Software



FERNANDO ARANGO I.  
CARLOS MARIO ZAPATA J.



# **UN-MÉTODO para la Elicitación de Requisitos de Software**

**FERNANDO ARANGO I.**

**CARLOS MARIO ZAPATA J.**

**Colaboradores:  
Sandra Milena Villegas  
Juan David Bolívar  
Juan Fernando Taborda**

**ISBN: 958-33-9717-2**

# **UN-MÉTODO PARA LA ELICITACIÓN DE REQUISITOS DE SOFTWARE**

Fernando Arango I.  
Carlos Mario Zapata J.

Colaboradores:  
Sandra Milena Villegas  
Juan David Bolívar  
Juan Fernando Taborda

Editor-Autor: Carlos Mario Zapata J.

## **DERECHOS RESERVADOS**

Queda prohibida la reproducción o transmisión total o parcial del texto de la presente obra bajo cualesquiera formas, electrónica o mecánica, incluyendo fotocopiado, almacenamiento en un sistema de recuperación de información, o grabado sin el consentimiento previo y por escrito del editor.

Datos para Catalogación Bibliográfica:  
Arango, Fernando y Zapata, Carlos Mario  
*UN-MÉTODO para la Elicitación de Requisitos de Software*

ISBN: 958-33-9717-2

Esta obra se terminó de imprimir en Agosto de 2006  
en la Escuela de Sistemas de la Universidad Nacional de Colombia  
Medellín, Colombia.

Impreso en Colombia  
*Printed in Colombia*

## DEDICATORIA

A Gloria, Vero y Susi

*Fernando*

A Vicky, la Perseverancia, Sebas, la Nobleza y Pipe, la Alegria.  
Esas son las materias primas de las que se componen mis sueños  
y las realidades que los materializan.

*Carlos M.*

## **AGRADECIMIENTOS**

Las ideas plasmadas en este libro han sido secundadas por muchas personas, comenzando por nuestros colaboradores directos Sandra, Juan David y Juan Fernando, pero también por otros estudiantes como Luz Marcela Ruiz, David Cardona y Fernán Alonso Villa, egresados como María Clara Gómez y Andrés Muñetón, y muchas generaciones de estudiantes de la Línea de Profundización en Ingeniería de Software de la Escuela de Sistemas y de nuestros diferentes proyectos de Investigación.

Deseamos expresar también nuestro sentido de gratitud a la Escuela de Sistemas de la Facultad de Minas, Universidad Nacional de Colombia sede Medellín, a la DIME (Dirección de Investigación de la Sede Medellín), a la Dinain (Dirección Nacional de Investigación) y a Conciencias, por el apoyo que hemos recibido en los diferentes proyectos de Investigación que han nutrido nuestras ideas y nos han dado la oportunidad de ensayarlas.

## TABLA DE CONTENIDO

TEMA	PÁGINA
Dedicatoria	iii
Agradecimientos	iv
Tabla de Contenido	v
Tabla de Figuras	vii
Índice de Tablas	viii
Preliminares	ix
 1. LOS MÉTODOS DE DESARROLLO DE SOFTWARE.	 1
1.1. INTRODUCCION	1
1.2. ALGUNOS MÉTODOS DE DESARROLLO DE SOFTWARE	2
1.2.1. Custom Development Method (CDM)	2
1.2.2. Racional Unified Process (RUP)	3
1.2.3. Feature Driven Development (FDD)	4
1.2.4. Extreme Programming	5
1.2.5. Aspectos Positivos y Negativos de los Métodos de Desarrollo de Software	6
1.2.5.1. Aspectos Positivos	6
1.2.5.2. Aspectos Negativos	7
1.3. UN-MÉTODO: EL MÉTODO DE ELICITACIÓN DE REQUISITOS PARA EL DESARROLLO DE SOFTWARE	8
1.4. ENUNCIADO DEL EJEMPLO	11
 2. ENTREGABLE 1: CONTEXTO DEL SOFTWARE	 13
2.1. INTRODUCCION	13
2.2. ENTREGABLE	13
2.2.1. Actores	14
2.2.1.1. Estructura de la Organización	14
2.2.1.1.1. Organigrama	14
2.2.1.1.2. Responsabilidades Generales de las Áreas	14
2.2.1.2. Área del Problema	15
2.2.1.2.1. Objetivos y Responsabilidades propias del Área	15
2.2.1.2.2. Organigrama del Área	15
2.2.1.2.3. Responsabilidades de las diferentes Componentes del Área	16
2.2.1.2.4. Actores y sus Roles	16
2.2.2. Esquema Preconceptual	17
2.2.3. Modelo del Dominio	18
2.3. ASPECTOS DE CONSISTENCIA Y REFINAMIENTO EN UN-MÉTODO	22
 3. ENTREGABLE 2: ANÁLISIS DEL PROBLEMA	 23
3.1. INTRODUCCIÓN	23
3.2. ENTREGABLE	23
3.2.1. Introducción	23
3.2.2. Procesos del Área	24
3.2.2.1. Tabla Explicativa de los Procesos	28
3.2.2.2. Diccionario de Datos	33
3.2.3. Objetivos del Área	35
3.2.4. Problemas y sus Causas	40

## TABLA DE CONTENIDO

TEMA	PÁGINA
3.3. ASPECTOS DE CONSISTENCIA Y REFINAMIENTO EN UN-MÉTODO	41
4. ENTREGABLE 3: PROPUESTAS DE SOLUCIÓN	45
4.1. INTRODUCCIÓN	45
4.2. ENTREGABLE	46
4.2.1. Introducción	46
4.2.2. Nuevo Organigrama	48
4.2.3. Cambios en los Actores	49
4.2.4. Nuevo Diagrama de Procesos	49
4.2.5. Casos de Uso	49
4.2.6. Carta de Navegación de Interfaces	64
4.2.7. Valoración de la Propuesta de Solución	65
4.2.8. Costeo de la Propuesta de Solución	68
4.2.9. Factores Críticos de Éxito de la Propuesta de Solución	74
4.3. ASPECTOS DE CONSISTENCIA Y REFINAMIENTO EN UN-MÉTODO	75
4.4. UN USO ADICIONAL DEL DIAGRAMA DE PROCESOS EN EL MARCO DE UN-MÉTODO	77
5. ENTREGABLE 4: ESQUEMA CONCEPTUAL	83
5.1. INTRODUCCIÓN	83
5.2. ENTREGABLE	83
5.2.1. Introducción	83
5.2.2. Consultas y Transacciones	84
5.2.2.1. Consultas	84
5.2.2.2. Transacciones	88
5.2.3. Diagrama de Clases	91
5.2.4. Derivaciones y Restricciones	93
5.2.4.1. Derivaciones	93
5.2.4.2. Restricciones	93
5.2.5. Eventos y Operaciones	94
5.2.5.1. Eventos y Operaciones	94
5.2.5.2. Máquina de Estados	94
5.2.5.3. Diagrama de Interacción	95
5.3. ASPECTOS DE CONSISTENCIA Y REFINAMIENTO EN UN-MÉTODO	95
6. LA ELICITACIÓN DE REQUISITOS EMPLEANDO UN-MÉTODO	97
REFERENCIAS	99



## TABLA DE FIGURAS

DESCRIPCIÓN	PÁGINA
Figura 1.1. Fases y Procesos de CDM	3
Figura 1.2. Arquitectura de Rational Unified Process (RUP)	4
Figura 1.3. Esquema de las Fases de un Proyecto realizado mediante FDD	5
Figura 1.4. Esquema de un Desarrollo de Software usando XP	6
Figura 1.5. Esquema General de los Artefactos de UN-MÉTODO	10
Figura 2.1. Organigrama de RAPIZZA Ltda.	14
Figura 2.2. Simbología básica del Esquema Preconceptual.	17
Figura 2.3. Esquema Preconceptual de RAPIZZA Ltda.	19
Figura 2.4. Esquema de un Modelo del Dominio	20
Figura 2.5. Modelo del Dominio de RAPIZZA Ltda..	21
Figura 3.1. Ejemplo de Diagrama de Procesos con sus diferentes Elementos	26
Figura 3.2. Diagrama de Procesos de la Situación Actual de RAPIZZA Ltda.	27
Figura 3.3. Simbología básica del Diagrama de Objetivos de KAOS	36
Figura 3.4. Diagrama de Objetivos correspondiente a RAPIZZA Ltda.	38
Figura 3.5. Detalle del Objetivo “Satisfacer las necesidades del cliente En relación con los productos de la pizzería” Correspondiente a la Figura 3.4.	39
Figura 3.6. Principales Componentes del Diagrama Causa-Efecto	40
Figura 3.7. Diagrama Causa-Efecto correspondiente a RAPIZZA Ltda.	42
Figura 3.8. Regla de Consistencia entre la Tabla Explicativa de Procesos y los Diagramas de Procesos y Objetivos	43
Figura 3.9. Regla de Consistencia entre la Tabla Explicativa de Procesos y el Diagrama Causa-Efecto	43
Figura 4.1. Diagrama de Procesos para el Sistema Informático para el Manejo de Pedidos de RAPIZZA Ltda.	50
Figura 4.2. Representación Gráfica de un Caso de Uso	51
Figura 4.3. Representación de Inclusión entre dos Casos de Uso	51
Figura 4.4. Representación de Extensión entre dos Casos de Uso	51
Figura 4.5. Carta de Navegación de Interfaces para el Sistema Informático para el Manejo de Pedidos de RAPIZZA Ltda.	65
Figura 4.6. Asignación de Porcentajes de Importancia en el Diagrama Causa-Efecto	66
Figura 4.7. Diagrama de Procesos utilizado en UNC-Diagramador Para obtener automáticamente el código Java® de RAPIZZA Ltda.	78
Figura 4.8. Menú Principal del Software de RAPIZZA Ltda. generado por UNC-Diagramador	79
Figura 4.9. Interfaz para el Registro de Personas de RAPIZZA Ltda. generada por UNC-Diagramador	79
Figura 4.10. Interfaz de Registro de Pagos de RAPIZZA Ltda. generada por UNC-Diagramador	80
Figura 4.11. Interfaz de Elaboración de Cuentas de Cobro de RAPIZZA Ltda. generada por UNC-Diagramador	80
Figura 4.12. Interfaz de Registro de Pedidos de RAPIZZA Ltda. Generada por UNC-Diagramador	81
Figura 4.13. Interfaz de Generación de Encabezado de Pedidos de RAPIZZA Ltda. generada por UNC-Diagramador	81
Figura 4.14. Interfaz para la Elaboración de Detalles de Pedido de RAPIZZA Ltda. generada por UNC-Diagramador	82

## TABLA DE FIGURAS

DESCRIPCIÓN	PÁGINA
Figura 5.1. Consultas correspondientes al Caso de Uso Registrar Personas	84
Figura 5.2. Consultas correspondientes al Caso de Uso Registrar Pedidos	85
Figura 5.3. Consultas correspondientes al Caso de Uso Registrar Pedidos en su Interacción Agregar Detalle Pedido	86
Figura 5.4. Consultas correspondientes al Caso de Uso Registrar Pago	87
Figura 5.5. Consultas correspondientes al Caso de Uso Imprimir Despachos	87
Figura 5.6. Simbología básica del Diagrama de Clases	91
Figura 5.7. Diagrama de Clases correspondiente a RAPIZZA Ltda.	92
Figura 5.8. Diagramas de Máquina de Estados para los Objetos “Pedido” y “Despacho”, correspondientes al Diagrama de Clases de RAPIZZA Ltda.	95
Figura 5.9. Diagrama de Comunicación correspondiente a RAPIZZA Ltda.	96

## INDICE DE TABLAS

DESCRIPCIÓN	PÁGINA
Tabla 3.1. Definiciones de la Tabla Explicativa de los Procesos	28
Tabla 3.2. Reglas del Negocio	29
Tabla 3.2. Tabla Explicativa de los Procesos de RAPIZZA Ltda.	30-32
Tabla 3.4. Reglas del Negocio de RAPIZZA Ltda.	32-33
Tabla 3.5. Diccionario de Datos	34
Tabla 3.6. Diccionario de Datos correspondiente a RAPIZZA Ltda.	34-35
Tabla 4.1. Especificación Textual de un Caso de Uso	52
Tabla 4.2. Descripción Textual de las Interacciones de un Caso de Uso	52
Tabla 4.3. Caso de Uso Registrar Personas	53-55
Tabla 4.4. Caso de Uso Registrar Pedidos	56-60
Tabla 4.5. Caso de Uso Registrar Pagos	60-62
Tabla 4.6. Caso de Uso Imprimir Despachos	63-64
Tabla 4.7. Porcentaje atendido del Diagrama Causa-Efecto mediante los Casos de Uso de la Solución	67-68
Tabla 4.8. Clasificación de los Actores según su Tipo de Interacción	69
Tabla 4.9. Clasificación de los Casos de Uso según la Cantidad de Transacciones y Escenarios	70
Tabla 4.10. Bases para el Cálculo del Factor de Complejidad Técnica	70-71
Tabla 4.11. Elementos para el Cálculo del Factor de Entorno	71-72
Tabla 4.12. Elementos para el Cálculo del Factor de Complejidad Técnica	73
Tabla 4.13. Elementos para el Cálculo del Complejidad del Entorno	74

## PRELIMINARES

Hace 30 años la imagen de un conjunto de bestias prehistóricas hundiéndose en un pozo de alquitrán se convirtió en una de las imágenes más representativas de las industrias de software de ese entonces. El creador de la metáfora, Frederick P. Brooks Jr., creía firmemente que la salida de ese pozo para las industrias de software sería un enfoque disciplinado y metodológico que les permitiese brindar soluciones cada vez más mantenibles, evitándoles hundirse cada vez más en el alquitrán. En la actualidad, esa metáfora sigue siendo vigente y, pese a que existen muy buenos métodos de desarrollo de software, su uso aún no es generalizado; ello origina, como en los setentas, síntomas de las dificultades que atraviesa el medio, como las demoras en los proyectos, los sobrecostos en su realización y los problemas de mantenibilidad que aún siguen siendo evidentes.

Conscientes de las necesidades de esta área, en el grupo de Investigación en Ingeniería de Software de la Escuela de Sistemas de la Universidad Nacional de Colombia se han venido realizando investigaciones y desarrollos del denominado UN-MÉTODO, del cual en este libro se presentan los resultados pertinentes al proceso de Elicitación de Requisitos. UN-MÉTODO no pretende ser un reemplazo de los métodos de desarrollo de software vigentes y exitosos en la actualidad, sino más bien una alternativa que permita generar consciencia en las generaciones futuras de Ingenieros de Sistemas sobre la necesidad de involucrar procesos metodológicos en el desarrollo de software.

Este libro se ha desarrollado en seis capítulos, de los cuales el primero se enfoca en los métodos de desarrollo existentes y el último compendia la visión general de UN-MÉTODO. Los capítulos 2 a 5 se ocupan de los entregables que se deben elaborar para realizar el proceso completo de Elicitación de requisitos de una pieza de software, lo cual implica la captura de los requisitos del interesado y su procesamiento hasta convertirlos en especificaciones formales y semiformales de la solución. Algunos de los artefactos que se emplean en los capítulos 2 a 5 son comunes a muchos métodos de desarrollo y se pueden considerar estándares de modelamiento; por ello, las descripciones de los diferentes artefactos que se presentan son muy someras y se deberían complementar (para los lectores con poca experiencia en modelamiento en Ingeniería de Software) con lecturas de las referencias bibliográficas que se anotan. Confiamos en que esta obra será de interés general para todos aquellos que, de una u otra manera, se encuentren inmersos en el mundo del Análisis y Diseño de herramientas de software.



# Capítulo 1

## Los Métodos de Desarrollo de Software

### 1.1. INTRODUCCIÓN

El Desarrollo de Software ha sido desde sus inicios una actividad caótica y poco ingenieril; en efecto, si se observa con atención la historia de su realización, se pueden encontrar más problemas que aciertos. Es común que los proyectos de software se atrasen y que los presupuestos destinados a ellos se rebasen casi sin límite; es común que los esfuerzos en la gestión de esos proyectos caigan en terrenos estériles y que en ocasiones parezca que esos proyectos van a la deriva.

Sin embargo, con el nacimiento, hace ya más de tres décadas, de la Ingeniería de Software como disciplina, se han propuesto varios elementos que han procurado suministrarle al Desarrollo de Software una imagen más organizada y a sus productos una mantenibilidad y trazabilidad que se creían utópicas y lejanas en los días iniciales de este arte.

Desde las propuestas de diferentes tipos de diagramas para representar los problemas del mundo del software, hasta su reunión en métodos de diferentes características, pasando por los diferentes paradigmas de representación, se ha procurado la planificación del desarrollo y el modelamiento de los problemas de manera sistemática y con etapas bien definidas, con el fin de combatir el caos característico de los proyectos de Desarrollo de Software. En efecto, se conocen muchos ejemplos de los diferentes tipos de elementos, a saber:

- Diagramas: Flujo de Datos, Entidad-Relación, Procesos, Causa-Efecto, Clases, Objetos, Transición de Estados, entre otros.
- Métodos: Custom Development Method (CDM), Rational Unified Process (RUP), Feature Driven Development (FDD), Extreme Programming (XP), entre otros.
- Paradigmas: Estructurado, Funcional, Relacional, Objetual, Objeto-Relacional.

En este libro se aborda un nuevo método de Desarrollo de Software, que busca incorporar una serie de diagramas característicos de diferentes paradigmas, buscando subsanar algunas de las deficiencias que exhiben los métodos tradicionales de Desarrollo de Software. Para ello, se discuten en este primer capítulo algunos de los métodos más reconocidos, se abordan sus carencias y se

resaltan los aspectos positivos que dan lugar a su uso. Finalmente, se presenta UN-MÉTODO, como una alternativa que incorpora un conjunto de diagramas y artefactos para realizar la elicitación de requisitos en el desarrollo de software.

## **1.2. ALGUNOS MÉTODOS DE DESARROLLO DE SOFTWARE**

### **1.2.1. Custom Development Method (CDM)**

Oracle® Corporation ha desarrollado productos que a nivel empresarial han alcanzado altos estándares de calidad y se han caracterizado por su versatilidad y capacidad para el manejo de grandes soluciones de software. Dentro de este cúmulo de productos, han incorporado una línea orientada al diseño de soluciones de software, con herramientas CASE propias que, como el Designer 2000, han procurado apoyar y automatizar la gestión de los analistas de software en la materialización de tales soluciones.

Paralelamente con el desarrollo de sus herramientas de diseño, Oracle® creó un método de desarrollo basado en sus diagramas y herramientas, que permite hacer un seguimiento intensivo de las diferentes fases del desarrollo, las cuales identifican como Definición, Análisis, Diseño, Construcción, Transición y Producción. Para ello, realizan un conjunto de tareas que se agrupan en procesos. Cada proceso hace parte de cada una de las fases del desarrollo y se reporta mediante un documento denominado “entregable”.

El Método de Desarrollo Adaptable (CDM) [ORACLE, 2000] es un método intensivo en documentación. Cada uno de los entregables que se realiza posee estándares ya definidos, los cuales se recomiendan especialmente para productos de software de gran tamaño, especialmente orientados hacia el manejo de grandes cantidades de datos. El proceso completo del método CDM se puede apreciar en la Figura 1.1., en la cual se consignan las fases en las columnas y los diferentes procesos en las filas. Si bien se afirma que la cantidad de documentación será variable dependiendo del tamaño y complejidad de la pieza de software que se desea desarrollar, de todos modos la documentación es bastante grande y requiere validaciones permanentes por parte de los interesados, con el fin de garantizar que los requisitos se están cumpliendo adecuadamente hasta llegar a la pieza de software.

Ahora, se afirma que el método CDM toma como punto de partida los procesos de la organización, pero se deja de lado un conocimiento necesario de la organización misma, representado por los objetivos generales de la organización y sus diferentes áreas y los problemas que se revelan en cada paso de los procesos de la misma. Ello ocurre porque CDM es un método más orientado hacia la solución que hacia los problemas y necesidades explícitas e implícitas de los interesados en el proceso de desarrollo.

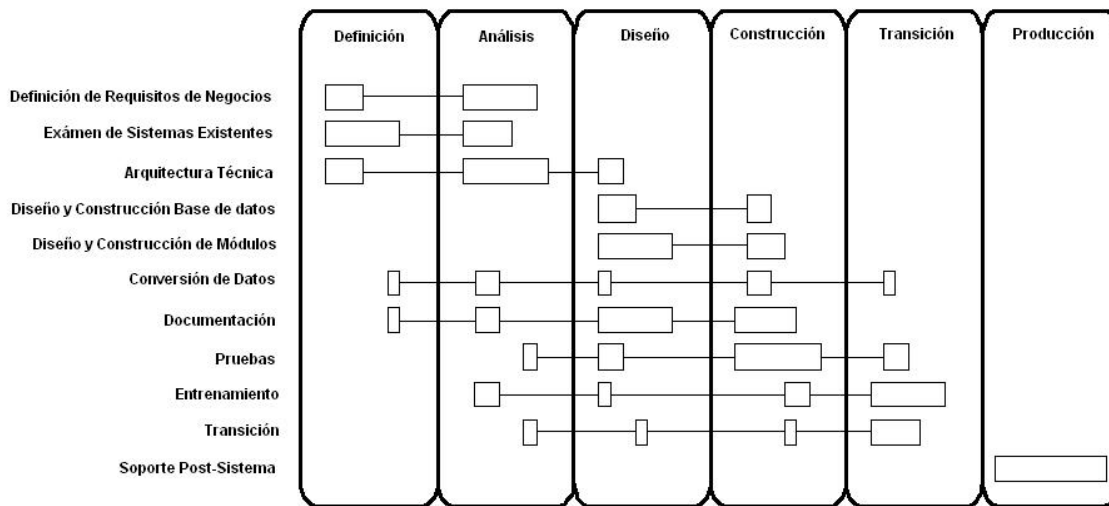


Figura 1.1. Fases y procesos de CDM

### 1.2.2. Rational Unified Process (RUP)

El surgimiento, a mediados de la década de los noventa, del Unified Modeling Language (Lenguaje Unificado de Modelamiento, UML) [OMG, 2006] como el principal estándar de modelamiento para el desarrollo de software, marcó el inicio también del método RUP [KRUCHTEN, 1999]. De los mismos creadores del UML, RUP tomó ventaja de la necesidad de estandarización de la creciente industria del software y, sobre la base de un grupo de diagramas basados en UML, definió un conjunto de artefactos aplicables a todo el ciclo de desarrollo del software. Al igual que CDM, RUP es un método ampliamente documental, en el cual se busca un refinamiento permanente de las diferentes representaciones de la solución informática a un problema hasta su codificación y entrega; ahora, a diferencia de CDM, RUP se basa en el estándar de desarrollo dictado por el OMG en la actualidad, que es UML. Además, es un proceso iterativo e incremental, en el cual paulatinamente se van obteniendo más elementos que conducen al desarrollo completo de la pieza de software.

El punto de partida del proceso de RUP es la elicitación de requisitos del software, la cual afirman se puede hacer completa mediante casos de uso. A partir de allí, el proceso completo se basa en un refinamiento sucesivo de los casos de uso acompañado de otros artefactos que se van construyendo en la medida en que son necesarios.

En la Figura 1.2. se puede apreciar la Arquitectura de RUP. Allí se presentan dos ejes para el proceso seguido por RUP: el eje horizontal, que representa el tiempo de desarrollo, y el eje de contenido, que involucra las áreas temáticas que se van desarrollando. En el eje de tiempo se consignan las diferentes fases del desarrollo

(Incepción, Elaboración, Construcción y Transición), además de las diferentes iteraciones que se realizan a lo largo del tiempo. En el eje vertical se consignan las diferentes actividades que se realizan en la Ingeniería del Software para lograr los resultados (actividades y roles básicamente).

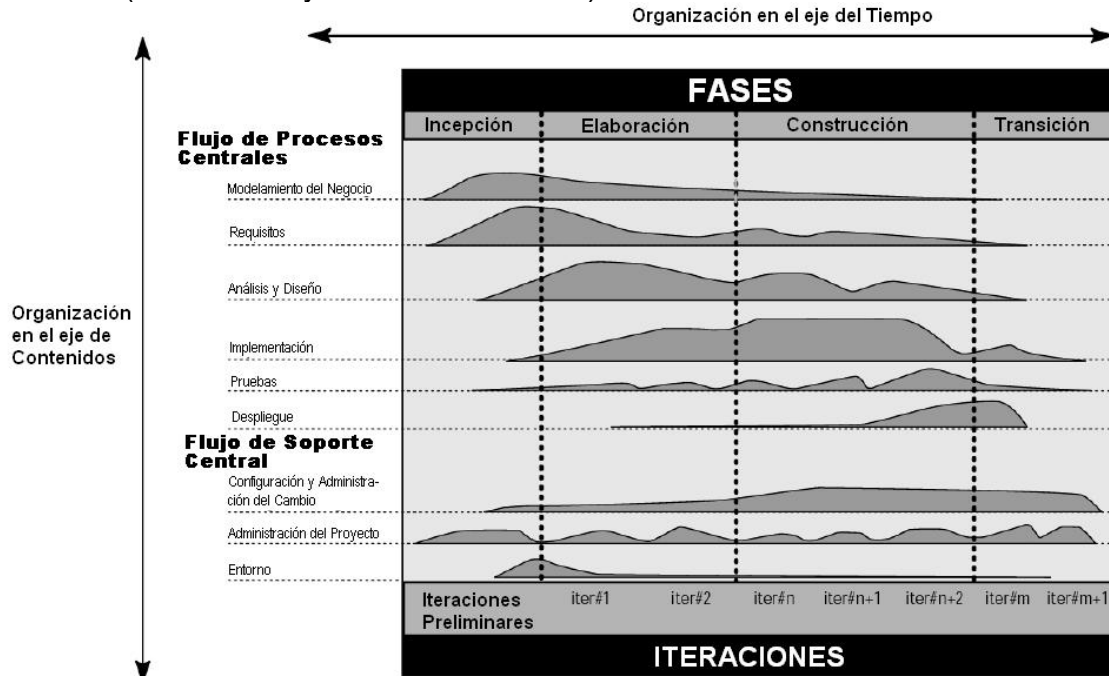


Figura 1.2. Arquitectura de Rational Unified Process (RUP)

Entre las críticas que se pueden hacer al método RUP se encuentra el hecho de que su punto de partida y el artefacto que guía el proceso hasta su terminación está constituido por los Casos de Uso, que en el modelamiento de piezas de software se suelen asociar con la solución, la cual debería ser obtenida a partir de un examen minucioso de las dificultades que atraviesa la organización. Sin embargo, como puntos a favor tiene la estandarización que se logra al emplear artefactos de UML y el proceso iterativo que se realiza al interior de cada una de las fases, que simula condiciones muy reales, pues en la medida en que se avanza en las fases del desarrollo se van obteniendo cada vez más detalles de la aplicación.

### 1.2.3. Feature Driven Development (FDD)

Este es un método de desarrollo basado en las características o funcionalidades del software (de allí su nombre “desarrollo basado en funcionalidades”) y cuya documentación no es tan intensiva como la que se podría esperar en métodos como CDM o RUP. Además, es una técnica grupal en la que se procura que existan analistas expertos y novatos en cada uno de los grupos que va a desarrollar la aplicación, con el fin de promover la experiencia en su utilización.



Un elemento importante de FDD [COAD ET AL., 1999] lo constituye el conjunto de métricas que define durante el proceso, las cuales tienen utilidad para el seguimiento de los interesados, la gestión de los jefes de proyecto, la estimación de proyectos futuros con características similares y la evaluación del estado del proyecto en cualquier instante.

En la Figura 1.3. se presenta un esquema de las fases típicas de un proyecto realizado mediante FDD. Es también un proceso iterativo en el cual las tres primeras iteraciones consumen gran parte del tiempo inicial pues en ellas se realiza un proceso de refinamiento; a medida que avanza el proyecto, las dos fases finales son las que más tiempo consumen.

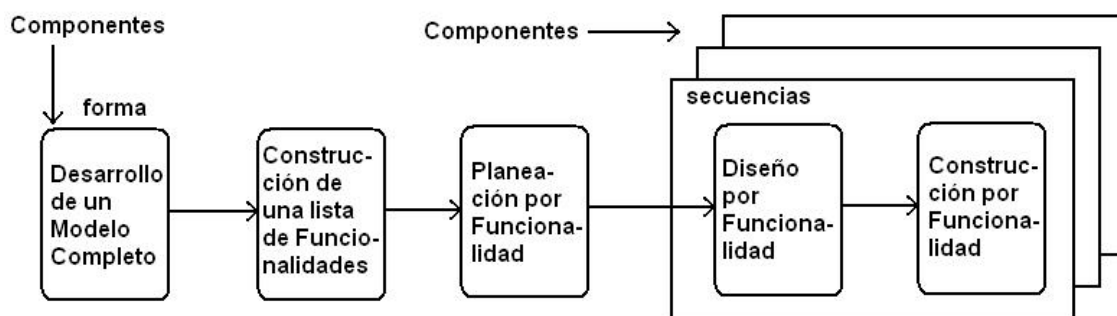


Figura 1.3. Esquema de las fases de un proyecto realizado mediante FDD.

FDD puede considerarse un método intermedio entre los métodos denominados “monumentales”, como CDM y RUP que adquieren esa denominación por la gran cantidad de documentación que requieren, y los métodos “ágiles”, de los cuales XP, que se verá con más detalle en la próxima sección, es un ejemplo.

#### 1.2.4. Extreme Programming (XP)

En oposición a los métodos altamente documentales, como los presentados en las secciones previas, han venido surgiendo propuestas de métodos que buscan llegar muy rápidamente al objetivo de desarrollo de software, sin realizar un análisis documental intensivo, sino más bien basados en una interacción entre los interesados y los programadores. Entre estos métodos, llamados “ágiles” por la razón anotada, se encuentra la Programación Extrema (Extreme Programming o XP) [BECK, 2000].

En este método, un equipo de programadores se comunica con un grupo de interesados para construir las “Historias de Usuario”, que constituyen la base inicial del método. A partir de allí, se inicia un proceso iterativo que busca la construcción del software de manera prototípica, realizando un acercamiento continuo con los interesados y fomentando las labores de comunicación entre los

miembros del equipo de desarrollo. Se diferencian tres roles principales en el equipo de desarrollo:

- Cliente (o interesado, como se ha venido denominando este rol a lo largo de este capítulo): Define lo que se hace y lo que se aplaza de la pieza de software y entrega las Historias de Usuario.
- Programador: analiza, diseña, prueba, programa e integra el sistema. Es quien realiza las abstracciones y refinamientos al sistema para que finalmente se entregue una pieza de software.
- Gerente: quien sirve de nexo entre clientes y programadores para que realicen su función de la forma más rápida posible. Es más un facilitador que un ejecutor, pues su función principal es garantizar que los procesos se ejecuten con facilidad.

Un esquema del desarrollo de una pieza de software mediante XP se ejemplifica en la Figura 1.4.

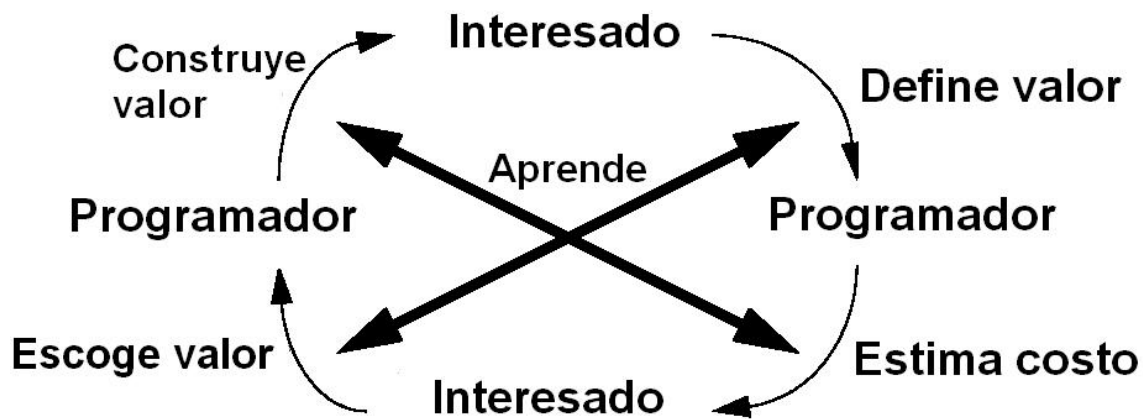


Figura 1.4. Esquema de un desarrollo de software usando XP.

Este es nuevamente un proceso iterativo, pero en este caso las iteraciones son muy cortas y completamente enfocadas a la clarificación de la pieza de software.

### 1.2.5. Aspectos Positivos y Negativos de los Métodos de Desarrollo de Software

#### 1.2.5.1. Aspectos Positivos

- Los métodos monumentales poseen una base documental abundante, que permite la verificación y validación constante de la pieza de software.
- Todos los métodos anotados son iterativos y la construcción ya sea de la documentación (en el caso de CDM, RUP y FDD) o de la pieza definitiva de

software (en el caso de XP) obedece a un conjunto de iteraciones incrementales, en las cuales se va obteniendo cada vez más información en relación con la solución que se está implementando.

- La mayoría de los métodos utilizan diagramas estándar para representar la solución informática. Esto posibilita la comunicación entre analistas de diferentes formaciones y conocimientos.
- El desarrollo por fases permite la definición de “hitos” de desarrollo de software, que son puntos de revisión del proceso para evaluar avances en el mismo.

#### **1.2.5.2. Aspectos Negativos**

- La totalidad de los métodos descritos en las secciones previas se enfocan en el modelamiento de la solución, y dejan de lado asuntos tan importantes como los objetivos de alto nivel de la organización o los problemas explícitos o implícitos que dan origen a una pieza de software.
- Los métodos monumentales son bastante intensivos en documentación, en tanto que los métodos ágiles son muy pobres en documentación. Si bien los dos tipos de métodos se recomiendan para diferentes proyectos de desarrollo de software, no existe un consenso sobre la utilidad de unos u otros métodos en proyectos “promedio”, no tan grandes como los que se realizan con CDM o RUP ni tan pequeños como los que se realizan con XP.
- En general, se utilizan modelos semiformales para el modelamiento de la pieza de software. La falta de formalismo, en especial en fases cercanas a la implementación del software, genera diferencias de interpretación que pueden ser ambiguas.
- En los métodos monumentales, el acercamiento al modelamiento de la pieza de software se da desde fases muy avanzadas del desarrollo; por ejemplo, RUP comienza ese acercamiento a partir de los casos de uso y CDM lo hace a partir del diagrama de procesos. En los métodos ágiles el acercamiento al modelamiento sufre un vacío porque se parte de modelos muy verbalizados (en el sentido que se aproximan más a historias que se cuentan en un lenguaje cercano al natural) y rápidamente se pasa a modelos que se relacionan mucho más con la solución.
- La transición es abrupta en casi todos los métodos y no existe un conjunto de normativas que permita “hilar” adecuadamente los requisitos desde las primeras expresiones de los interesados hasta el desarrollo de la pieza de software. En estos métodos se carece de mecanismos para garantizar la consistencia del proceso de elicitación de requisitos, la cual se debe garantizar a partir de reglas empíricas de consistencia entre los diferentes diagramas. UML, por ejemplo, tiene muy bien definidas a nivel teórico las reglas internas que deben cumplir los modelos (su sintaxis y semántica), pero existe poca formalización en las reglas que se deben hacer respetar cuando en dos diagramas se expresan los mismos datos.

### **1.3. UN-MÉTODO: EL MÉTODO DE ELICITACIÓN DE REQUISITOS PARA EL DESARROLLO DE SOFTWARE**

Como una respuesta a los aspectos negativos anotados, y tratando de conservar los más representativos aspectos positivos, el grupo de investigación UN-INFO de la Escuela de Sistemas perteneciente a la Facultad de Minas de la Universidad Nacional de Colombia se dio a la tarea de elaborar un método de desarrollo de software que permitiera, en primer término, una adecuada elicitación de requisitos, conservando la trazabilidad necesaria para transmitir los requisitos a la solución informática, cuando ésta se terminase. El resultado se denominó UN-MÉTODO y se ha constituido desde entonces en el método de elicitación de requisitos usado en la Línea de Profundización en Ingeniería del Software, además de múltiples Trabajos de Grado, Programas Especiales de Trabajo Académico y Prácticas Profesionales de los estudiantes adscritos a la Escuela de Sistemas.

Entre las características más destacables de UN-MÉTODO se cuentan las siguientes:

- Se realiza una base documental necesaria para definir la trazabilidad de la pieza de software, desde su concepción hasta la solución definida. Sin embargo, esta documentación no es tan abundante como la que se requiere en CDM o RUP.
- Si bien no es un método iterativo, como sí lo eran los métodos analizados en la sección anterior, se puede considerar que UN-MÉTODO es reiterativo, puesto que los diagramas y artefactos que se desarrollan en las fases iniciales se van abordando y modificando recurrentemente a medida que se va adquiriendo un conocimiento mayor de la organización, el problema y su solución. Esto adicionalmente permite que se puedan definir “hitos” de revisión asociados con estos grados de conocimiento.
- UN-MÉTODO no se aparta del estándar de modelamiento sugerido en la actualidad, e incluye un conjunto de diagramas pertenecientes a UML, además de otros diagramas pertenecientes a otros métodos (por ejemplo el diagrama de procesos de CDM se trabaja con ciertas modificaciones), algunos diagramas adicionales pertenecientes a diferentes teorías (como el diagrama de objetivos y el diagrama causa-efecto) y un diagrama particular desarrollado por el grupo UN-INFO (el esquema preconceptual).
- A diferencia de los métodos analizados en la sección anterior, UN-MÉTODO no parte de la solución, sino que se enfoca en primero conocer la organización, sus funciones y razón de ser, sus problemas y objetivos. Una vez se ha recabado toda esta información, se procede a definir un conjunto de soluciones posibles, entre las cuales se seleccionará usando los artefactos de valoración y costeo definidos por UN-MÉTODO. Es decir, la solución es un punto casi terminal de UN-MÉTODO, en lugar del punto de partida que es para los métodos analizados.

- Se conserva la figura de “entregables” que se sugería en CDM y RUP. En este caso dichos entregables, más que referirse a fases del desarrollo, se refieren a instantes de estudio de la organización, el problema, la solución y el esquema conceptual de la solución. Haciendo el equivalente con CDM, UN-MÉTODO se podría situar entre las fases de Definición, Análisis y el inicio de Diseño.
- Si bien en UN-MÉTODO se continúan empleando modelos semiformales para la representación de los diferentes elementos, se emplea también un artefacto final en conjunción con el diagrama de clases UML, para representar las pre y postcondiciones, restricciones y derivaciones y las operaciones asociadas con el diagrama de clases. Este artefacto es la lógica de predicados de primer orden, con la cual se realiza un análisis de los principales cuadros de diálogo definidos, los casos de uso y el diagrama de clases.
- En UN-MÉTODO se parte de artefactos muy cercanos al lenguaje del interesado (muy aproximado al “lenguaje natural”) y se va realizando el proceso hasta alcanzar modelos muy cercanos a la máquina. Esto posibilita, al igual que en XP, la retroalimentación constante del proceso tomando en consideración a los interesados, y se retoman luego los artefactos propios del modelador en cualquiera de sus instancias (analista o diseñador).
- En UN-MÉTODO se procura realizar una transición “suave” entre los diagramas, tratando de representar los elementos de la organización, el problema y la solución de manera consistente a lo largo del proceso de desarrollo. Para ello, se define un conjunto de reglas de consistencia que se deben cumplir entre los diferentes artefactos para garantizar la continuidad en el proceso.

El principal aporte en la realización de UN-MÉTODO radica en la concepción del método mismo pues, con excepción de artefactos tales como el esquema preconceptual, la tabla explicativa de los procesos y el método de valoración de las piezas de software, surgidos de la experiencia e investigación del grupo UN-INFO, los artefactos que se involucran en UN-MÉTODO pertenecen ya a métodos o teorías reconocidas para el desarrollo de software y la solución de problemas en algunas áreas.

En UN-MÉTODO se pueden diferenciar cuatro grandes hitos, que se esquematizan en la Figura 1.5., y que se describen brevemente a continuación:

- Contexto del Software: En esta fase se caracteriza la organización, con sus funciones y relaciones de poder, los actores y sus responsabilidades y el vocabulario del área.

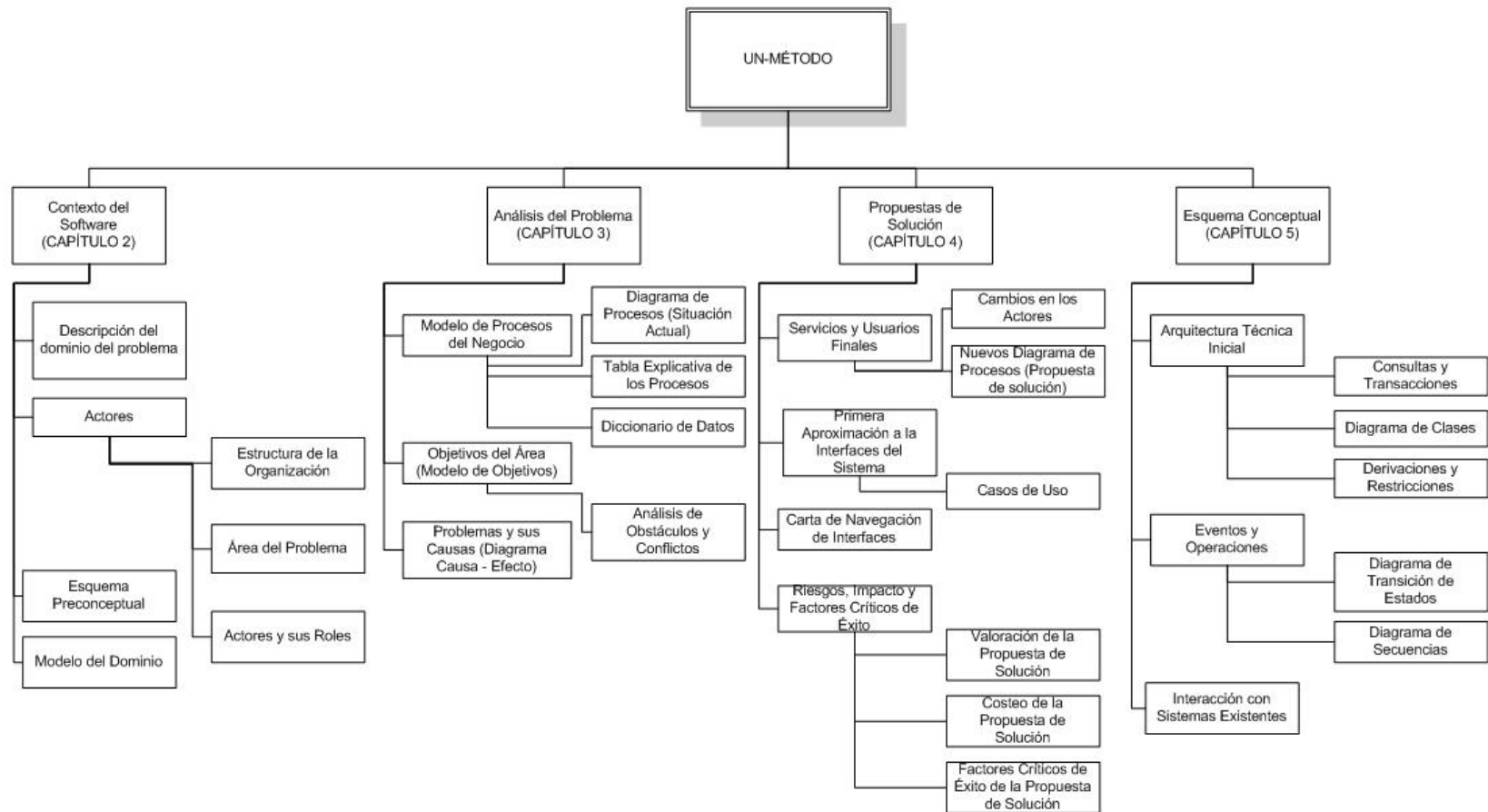


Figura 1.5. Esquema General de los artefactos de UN-MÉTODO

- **Análisis del Problema:** En esta fase se representan los procesos de la organización, los objetivos asociados con esos procesos y los principales problemas y sus causas que la organización es consciente que tiene y aquellos que sólo se podrán detectar luego de un análisis intensivo de la organización, puesto que la organización misma explícitamente no los conoce aunque puede haber experimentado sus consecuencias.
- **Propuestas de Solución:** En esta fase se identifica un conjunto de soluciones, se determina cuáles son los efectos de cada solución sobre la forma de proceder en el área, se determinan las características de cada solución, se valora cada solución para estimar el aporte que realiza cada una de ellas a la supresión o modificación de las causas de los problemas de la organización y se costea cada solución con una técnica de estimación del esfuerzo. Finalmente, se selecciona una solución para pasar a la siguiente fase del método.
- **Esquema Conceptual:** En esta fase, la solución seleccionada en la anterior se complementa y refina con la ayuda de un lenguaje formal, que en este caso es la lógica de predicados de primer orden. Igualmente, se determina la estructura de la solución y se modela el comportamiento que pueden exhibir algunos objetos y algunas transacciones de la solución.

#### **1.4. ENUNCIADO DEL EJEMPLO**

En los capítulos 2 a 5 se expondrá UN-MÉTODO con todos los artefactos que lo integran, ejemplificando cada parte del proceso con un caso de estudio al cual se aplicará el método completo. El caso de estudio se refiere al manejo de una pizzería que tiene problemas en el manejo de la información correspondiente a los pedidos de los clientes, los despachos y los pagos. El enunciado correspondiente al ejemplo, que servirá de base para la elaboración de los diferentes artefactos con que cuenta UN-MÉTODO, es el siguiente:

RAPIZZA Ltda. lleva ya cerca de un año de actividad y hasta ahora no se ha conseguido ganar tanto dinero como se había esperado. Esta pizzería ofrece a los clientes diversos tipos y tamaños de pizza, además de la posibilidad de ordenar aditivos. Su estructura no es muy grande pues cuenta con un despachador, tres chef y siete repartidores que realizan las diferentes funciones necesarias para la satisfacción de las necesidades de los clientes en relación con los productos de la pizzería. Así, el despachador se debe encargar de tomar los pedidos telefónicos y garantizar el envío de los productos, los chef preparan las pizzas de los pedidos y los repartidores las entregan. La pizzería tiene una zona de cobertura determinada y con el fin de competir frente a otras pizzerías existentes en la zona se ha estado ofreciendo al cliente una promoción que consiste en entregarle el pedido totalmente gratis si éste tarda en ser entregado más de 30 minutos. Esta promoción ha atraído una buena cantidad de clientes al negocio, pero también se ha convertido en una de las principales fuentes de pérdida de dinero, puesto que muy a menudo los repartidores no consiguen llevar todas las pizzas a tiempo. A

inconformidad de los repartidores, se ha impuesto en el reglamento que cada pedido entregado tarde, deberá ser pagado por el repartidor responsable, con el fin de evitar tanta pérdida de dinero y conseguir que los repartidores se esfuercen más en su trabajo.

Cuando un cliente de RAPIZZA Ltda. tiene necesidad de una pizza realiza una llamada a la pizzería en la cual es atendido por el despachador, quien toma el pedido de los productos requeridos; cada producto posee un código y su valor unitario se encuentra en una lista de precios. Del cliente, el despachador solicita la cédula, el nombre, teléfono y dirección. Dependiendo de la dirección, el cliente se ubica en una zona de cobertura. Además, del pedido el despachador registra el número, la fecha, la hora de salida y la hora de entrega, además del detalle de los productos a enviar, incluyendo la cantidad y una observación (si la hay).

Luego, el pedido pasa al chef, quien prepara los productos allí incluidos. Una vez los productos se han preparado y en cuanto uno de los repartidores se encuentre disponible, el despachador asigna un repartidor y realiza el despacho; el repartidor toma los pedidos preparados y se traslada a la dirección de destino del pedido para realizar la entrega.

Cuando llega el repartidor, el cliente recibe las pizzas y realiza el pago, siempre y cuando el tiempo de entrega sea inferior a 30 minutos; en caso contrario, únicamente firma el pedido sin realizar el pago. Una vez entregados todos los pedidos que lleva el repartidor, éste regresa a la pizzería y entrega el dinero y los pedidos firmados al despachador. El despachador, entonces, guarda el dinero en la caja y elabora cuentas de cobro al repartidor por cada pedido firmado que el cliente no haya pagado. Tanto los pagos como las cuentas de cobro cancelan los pedidos.

Uno de los principales problemas de la pizzería es la recuperación del dinero de las ventas por parte del despachador, ya que se requiere asegurar el envío del producto al cliente y la entrega a tiempo de los pedidos a cargo de los repartidores. Esto es sumamente importante para RAPIZZA Ltda., ya que de ello depende el éxito de las ventas y la satisfacción al cliente, medio por el cual se espera conseguir reconocimiento y un incremento en la clientela.



# Capítulo 2

## Entregable 1: Contexto del Software

### 2.1. INTRODUCCIÓN

El desarrollo de una pieza de software suele comenzar con una serie de reuniones entre Analistas e Interesados, en las cuales se define una terminología común (perteneciente al dominio en que se desempeña la organización que requiere ese desarrollo). Esa terminología incluye, entre otros aspectos, la forma de proceder de los diferentes actores de la organización, sus relaciones de poder, sus objetivos de corto y largo plazo. Además, se busca que ese vocabulario tenga una representación; en la mayoría de los métodos analizados en el capítulo anterior dicha representación se basa en modelos verbales o en documentos de la organización, pero en UN-MÉTODO se reúne en dos diagramas: el Esquema Preconceptual y el Modelo del Dominio; ambos diagramas se nutren de la información textual recolectada y se constituyen en mecanismos de comunicación con los interesados, pues su lectura es cercana a una forma muy controlada de lenguaje natural y, por ello, es posible su validación por parte de los Interesados, especialmente en estas fases iniciales del desarrollo, cuando el conocimiento del Analista en relación con la organización aún es incipiente.

El contexto del software identifica:

- El marco organizacional que determina las funciones del área de aplicación.
- Los objetivos propios del área de la aplicación.
- La organización propia del área.
- Los actores e interesados en las funciones del área.
- Los roles e intereses de los actores e interesados frente a las funciones que les competen.
- El vocabulario propio del área.

### 2.2. ENTREGABLE

Durante esta primera fase, en el entregable se trata de delimitar el contexto que rodea la pieza de software que se piensa construir, con el fin de generar un acercamiento entre los Analistas (que generalmente tienen pocos conocimientos en relación con el área) y los Interesados; para ello, el entregable posee tres grandes divisiones: los Actores, el Esquema Preconceptual y el Modelo del Dominio.

## 2.2.1. Actores

### 2.2.1.1. Estructura de la Organización

#### 2.2.1.1.1. Organigrama

En Administración, los organigramas juegan un papel fundamental en la representación de las relaciones de poder entre los diferentes actores. Este aspecto estructural permite igualmente delimitar el alcance de los problemas de la organización que se espera sean resueltos mediante la pieza de software. Aunque existen muchos tipos de organigrama, con múltiples paradigmas, en el contexto de UN-MÉTODO se opta por una estructura de tipo jerárquico.

En la Figura 2.1 se puede apreciar el organigrama de RAPIZZA Ltda.



Figura 2.1. Organigrama de RAPIZZA Ltda.

#### 2.2.1.1.2. Responsabilidades Generales de las Áreas

En esta sección se deben presentar los objetivos generales de la organización y la manera como contribuyen a él las diferentes secciones del organigrama.

En el caso de RAPIZZA LTDA., las responsabilidades generales son las siguientes:

- RAPIZZA LTDA. es una entidad con ánimo de lucro, por lo cual busca ganar dinero para su propietario.
- La actividad de la pizzería debe ser suficiente para asegurar que se mantenga la venta del producto y que se conserven los precios a un nivel

competitivo. Para ello, es necesario que la compra se incentive mediante promociones y publicidad, pero también ofreciendo un servicio de calidad que permita el retorno de los clientes a realizar nuevos pedidos.

- Se debe garantizar el acceso de los clientes a los diferentes productos de la pizzería, conservando las políticas necesarias de envío de productos y realizar estudios de sus necesidades con el fin de ofrecer nuevos productos.
- También, se debe garantizar el recaudo del dinero, ya sea por pago directo del cliente, o por recuperación de pagos mediante las cuentas de cobro a los repartidores.

#### **2.2.1.2. Área del Problema**

##### **2.2.1.2.1. Objetivos y Responsabilidades propias del Área**

En teoría, las organizaciones siempre tendrán problemas por solucionar. Sin embargo, cuando se trata de soluciones de tipo informático, es necesario acotar los problemas que se va a tratar de solucionar, puesto que no es posible solucionar todos los problemas simultáneamente. Los interesados siempre tienen una primera aproximación al área crítica de su organización, y es esa “premonición” lo que motiva el análisis del problema para determinar un conjunto de soluciones; sin embargo, esa primera aproximación se debe validar con el análisis, puesto que los problemas de un área podrían tener su origen en otra área y, por ende, la solución se desplazaría hacia el área problemática. En esta sección se deben presentar los objetivos propios del área del problema.

En el problema de RAPIZZA LTDA., el área que se supone tiene los mayores problemas es el área de despachos y es allí donde se buscará la realización de la solución. En esta área, existen algunas responsabilidades asociadas:

- Garantizar la recuperación del dinero de las ventas, asegurando el pago del cliente o, en su defecto, cumpliendo las políticas de cobro a los repartidores.
- Es importante asegurar el registro de la información de los pedidos y la información del despacho.
- En esta área se debe buscar, en lo posible, asegurar la entrega a tiempo de los productos (con el fin de evitar las cuentas de cobro a los repartidores), tanto como asegurar el cobro de la entrega.

##### **2.2.1.2.2. Organigrama del Área**

Una vez se ha identificado el área a la cual se espera solucionar sus problemas (o posiblemente sólo una parte de ellos) con una herramienta informática, se debe detallar el área con la adición de nuevos organigramas que deben representar la

estructura de la organización en el área del problema. Por lo general, la complejidad de las organizaciones hace necesario este nuevo nivel de detalle, con el fin de determinar la estructura a nivel micro del área, diferente a la comprensión amplia que se logra con el organigrama de nivel general.

En el caso de RAPIZZA LTDA., la sencillez de su estructura organizacional es tal que no se requiere un detalle adicional del área problemática, más que el mostrado en la Figura 2.1., en la cual el área se referirá al despachador y los repartidores (lo que se podría conocer como “área de despachos”).

#### **2.2.1.2.3. Responsabilidades de las diferentes Componentes del Área**

Cada componente del área tendrá un grado de responsabilidad en la satisfacción de los objetivos del área. En esta sección se deben presentar los objetivos generales de las partes de la organización del área del problema.

En RAPIZZA LTDA., las responsabilidades por esos objetivos están delimitadas así:

- Garantizar la recuperación del dinero de las ventas, Asegurar el pago del cliente, asegurar la entrega a tiempo de los productos (con el fin de evitar las cuentas de cobro a los repartidores): Despachador y Repartidores.
- Cumplir las políticas de cobro a los repartidores, Asegurar el registro de la información de los pedidos y la información del despacho: Despachador.

#### **2.2.1.2.4. Actores y sus Roles**

En esta sección se deben describir los actores (tanto internos como externos) y sus responsabilidades en el área del problema, verbalizando sus actividades principales. No todos los actores estarán presentes en el organigrama y por ello se debe velar por la inclusión de la mayor cantidad de interesados que pueda cumplir una función en el desarrollo de los procesos del área.

En el caso de RAPIZZA LTDA., los actores y sus funciones son los siguientes:

- Cliente: Llama a la pizzería para realizar los pedidos, recibe los productos del pedido, realiza el pago del pedido y firma el pedido.
- Despachador: Registra y despacha el pedido, asigna el repartidor para un determinado conjunto de pedidos, registra la hora de entrega de los pedidos (cuando el repartidor regresa), registra el pago que realiza el cliente y elabora las cuentas de cobro que permiten cancelar los pedidos.
- Chef: Prepara los productos de los pedidos.
- Repartidores: Entregan los pedidos a los clientes y pagan las cuentas de cobro cuando no se cumple a tiempo con la entrega.

### 2.2.2. Esquema Preconceptual

Es una representación intermedia entre las especificaciones textuales en lenguaje natural (lo que se ha denominado repetidamente en este libro como “modelos verbales”) y los diferentes esquemas conceptuales que permiten el modelamiento de una pieza de software [ZAPATA *et al.*, 2006]. Una revisión cuidadosa de los Esquemas Preconceptuales permite inferir que son representaciones del modelo verbal del problema, al cual se le han resuelto por algún medio las ambigüedades propias del lenguaje natural y que se ha normalizado con el uso de ciertos verbos clave, con el fin de acercar este modelo del discurso a los esquemas conceptuales.

Si bien el Esquema Preconceptual es un primer intento de modelamiento del dominio del problema, es posible realizar una validación inicial de su contenido con los interesados, puesto que requiere poco entrenamiento para su lectura (se podría decir incluso que es fácil “leer” el modelo inambigüo y normalizado en un lenguaje controlado, cercano al natural), con el fin de incluir faltantes y corregir anomalías de la representación.

La simbología básica del Esquema Preconceptual se muestra en la Figura 2.2.

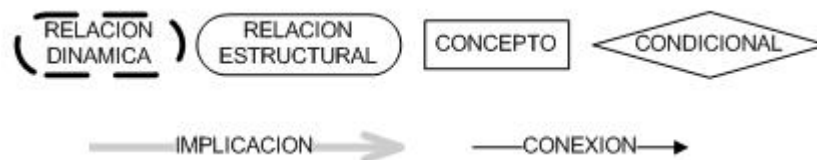


Figura 2.2. Simbología básica del Esquema Preconceptual.

Los diferentes elementos tienen los siguientes usos:

- Los conceptos únicamente admiten sustantivos sencillos o uniones de dos sustantivos con la preposición “de” (por ejemplo “hora de llegada”; el uso de este tipo de elementos, denominados “sintagmas nominales” se deja a criterio del modelador). Los conceptos únicamente pueden aparecer una vez en el Esquema Preconceptual (a menos, claro está, que el diagrama por su cantidad de elementos impida una adecuada visualización del concepto, en cuyo caso se puede repetir pero el significado será el mismo para ambos conceptos, actuando como uno solo); ello implica que, a medida que se generan representaciones de un concepto, se van sumando relaciones al mismo.
- Se permite únicamente la presencia de verbos en las relaciones. Puede ser un verbo simple o acompañado de un auxiliar. Existen dos verbos especiales que se deberían tomar en consideración al realizar el diagrama: el verbo “es” y el verbo “tener”, que se manejan en las denominadas “Relaciones Estructurales”; en tanto sea posible, expresiones equivalentes

se deben llevar a este tipo de verbos. Por ejemplo, “incluye”, “pertenece a”, etc. deberían ser llevados a relaciones del tipo “tiene”. Los verbos que expresen actividades se incluyen dentro de las denominadas “Relaciones Dinámicas”, como por ejemplo “realiza”, “registra”, “prepara”, etc.

- Los condicionales, representados por el rombo, pueden incluir expresiones que contengan operadores y conceptos. En general, se refieren a restricciones o reglas del negocio que se deben cumplir tomando como base los conceptos del mundo.
- Las implicaciones, representadas por flechas un poco más gruesas, tienen dos usos diferentes:
  - Unen pares de relaciones dinámicas para representar conexiones entre relaciones. En este sentido, convierten las relaciones dinámicas en precondiciones de una determinada relación dinámica.
  - Unen condicionales con relaciones dinámicas, también para expresar precondiciones.
- Las conexiones (flechas delgadas), pueden tener diferentes fines:
  - Unir conceptos con relaciones, con el fin de que la acción denotada por la relación sea ejecutada por el concepto.
  - Unir relaciones con conceptos, para que la acción expresada por la relación se transfiera a un concepto.
  - Se pueden colocar preposiciones en los enlaces posteriores a las relaciones cuando el interesado lo considere pertinente para aclarar el sentido de la acción descrita por la relación.

En la Figura 2.3. se presenta el Esquema Preconceptual para RAPIZZA LTDA., con base en la información textual que se ha venido presentando en las diferentes secciones.

### **2.2.3. Modelo del Dominio**

La terminología del área representada en el Esquema Preconceptual se comienza a acercar a la conceptualización de una solución a los problemas de la organización cuando se realiza el agrupamiento de los conceptos del mundo en lo que se denomina un Modelo del dominio, el cual muestra a los modeladores las clases conceptuales significativas del dominio del problema y sus relaciones. Es importante resaltar que en este modelo se representan clases conceptuales del mundo real y no componentes software como bases de datos o clases de C++ o java, que se incorporan al modelo de clases en la fase de diseño para soportar la funcionalidad requerida por el sistema.

Un modelo de dominio es una representación visual de las clases conceptuales u objetos del mundo real en un dominio de interés. En este tipo de modelos se representan las clases conceptuales del dominio, sus atributos y las relaciones

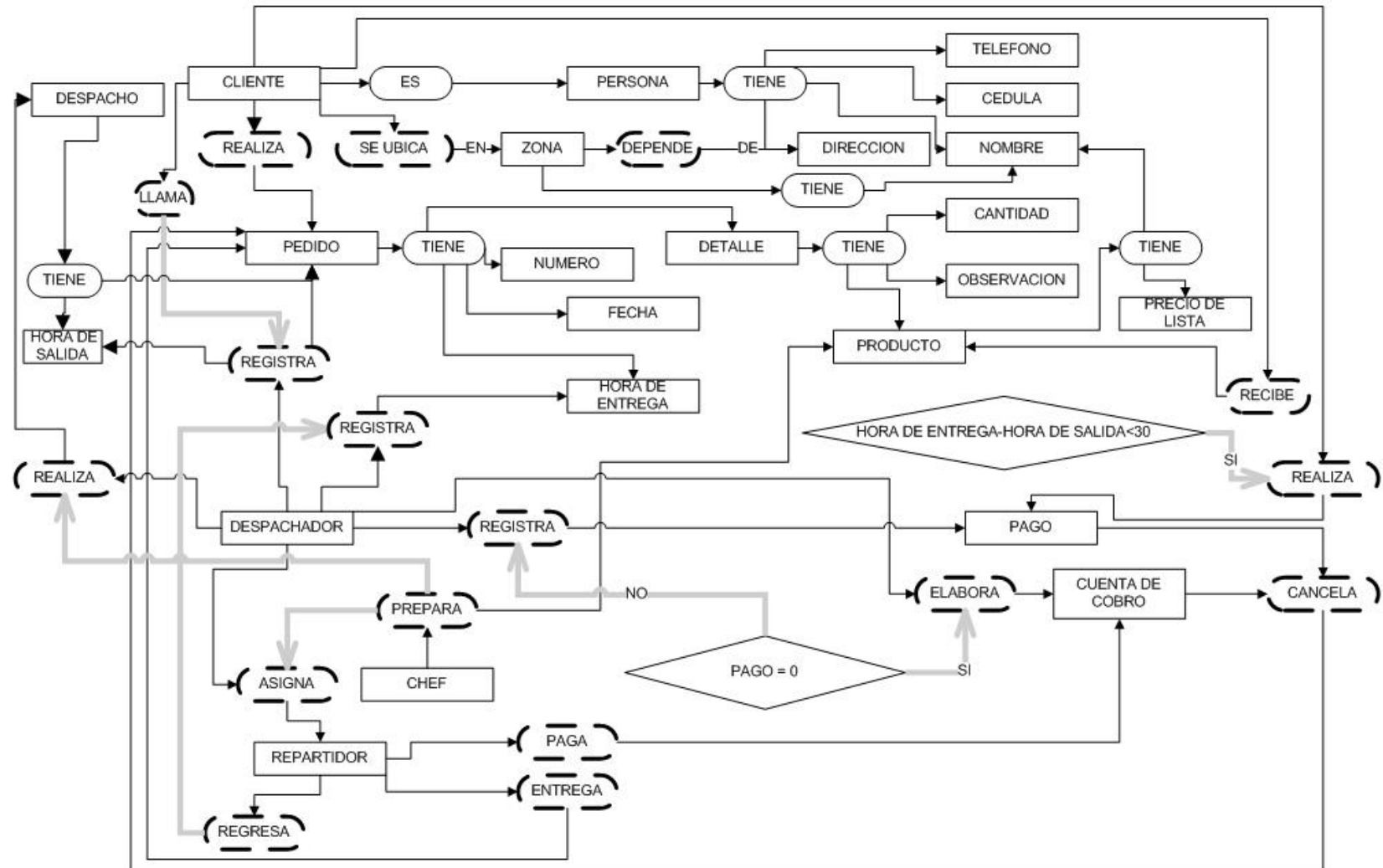


Figura 2.3. Esquema Preconceptual de RAPIZZA LTDA.

entre dichas clases. Es decir, al construir un modelo del dominio lo que se hace es crear un diccionario visual del dominio del problema, ya que se muestran conjuntamente los objetos importantes del entorno del problema junto con sus cualidades relevantes y la manera como todos se relacionan entre sí. Es importante realizar de manera paralela un diccionario de términos, el cual describe verbalmente el significado en el marco del área de aplicación de todos los términos y rótulos (de datos, funciones, reglas, restricciones etc.) que se suelen usar a lo largo de todo el proceso de desarrollo; si bien esta actividad se seguirá completando en los entregables futuros, debido a que día a día el analista deberá conocer más detalles de la organización, el proceso que se realiza en la organización y sus objetivos, la solución y el esquema conceptual de la solución, este primer entregable es un buen punto de partida para aclarar con la mayor cantidad posible de interesados los términos del área.

En un modelo del dominio, las “clases conceptuales” podrían definirse como ideas, cosas u objetos y se representan por medio de cajas rectangulares estereotipadas con el nombre, y contienen una serie de valores de datos lógicos o “atributos” importantes para dicha clase conceptual; las “asociaciones” se representa como una línea entre dos clases con un nombre de asociación; en cada extremo de la asociación se encuentra la “multiplicidad”, que indica cuántas instancias de una clase se pueden relacionar con las instancias de otra. En la Figura 2.4. se puede apreciar la simbología básica del modelo del dominio.

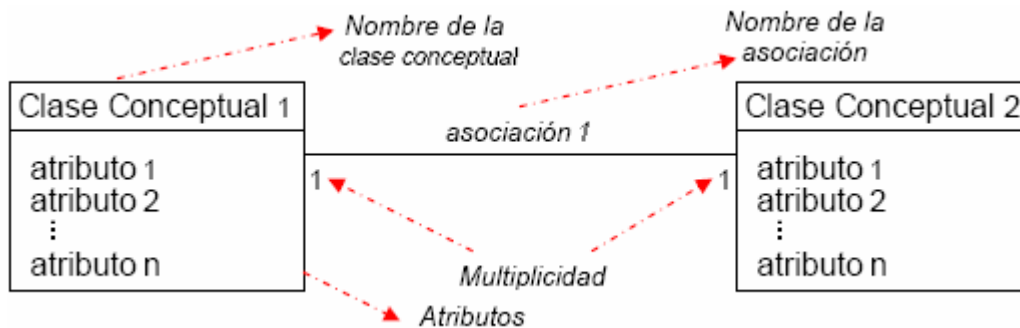


Figura 2.4. Esquema de un modelo del dominio

Tanto los Esquemas Preconceptuales como los Modelos del Dominio pueden considerarse como “Ontologías” del dominio, en el sentido expresado por [GUARINO, 1998] de proveer sistemas de categorías que cuenten para una cierta visión del mundo, o como formas de comunicación entre los interesados en la realización de una pieza de software. En la Figura 2.5. se puede apreciar el Modelo del dominio de RAPIZZA Ltda., en el cual existen similitudes apreciables con el Esquema Preconceptual, las cuales se discuten en la sección siguiente.



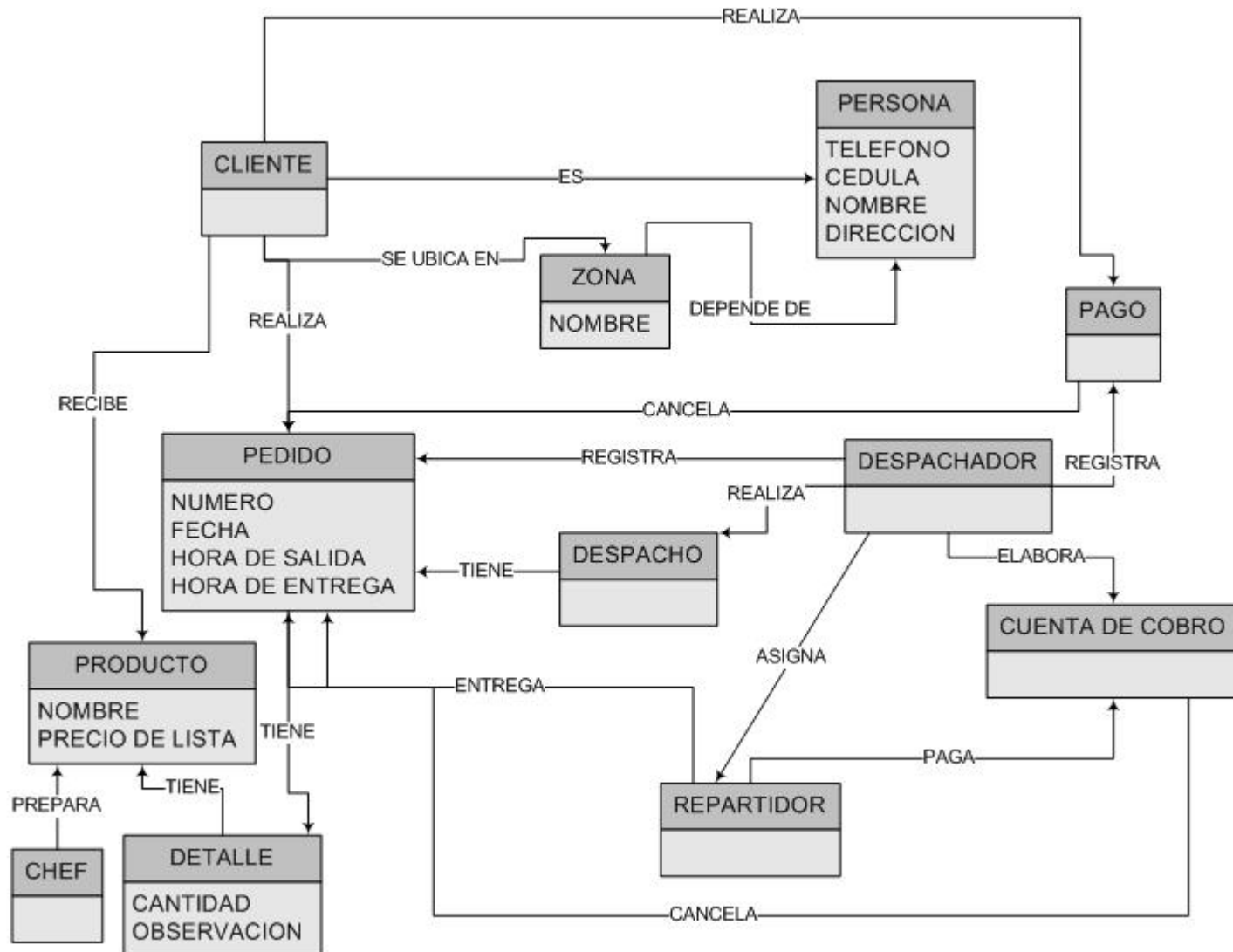


Figura 2.5. Modelo del Dominio de RAPIZZA Ltda..

### **2.3. ASPECTOS DE CONSISTENCIA Y REFINAMIENTO EN UN-MÉTODO**

En el desarrollo de una pieza de software se hace fundamental mantener la coherencia a lo largo del proceso, de forma que la solución informática que finalmente se implemente y se entregue a los interesados sea exactamente la que necesitan, la que permite solucionar sus problemas y la que llena sus expectativas. Es por ello que en UN-MÉTODO los aspectos de Consistencia y Refinamiento adquieren capital importancia.

En [ZOWGHI Y GERVASI, 2002] se reconoce la importancia de la consistencia en el proceso de Ingeniería de Requisitos, entendiendo por consistencia la relación que debe existir en los elementos de los diferentes artefactos que componen una descripción del dominio, y que en el caso de UN-MÉTODO consistirán en un conjunto de diagramas y elementos que se referirán siempre al mismo conjunto de conceptos y relaciones. El refinamiento, tal y como es definido por [D'SOUZA Y WILLS, 1998] se puede entender como la adición de detalles a medida que se va teniendo mayor conocimiento de ellos en el proceso de desarrollo de software, pero también podría entenderse como la transformación que van sufriendo los diferentes conceptos para ser representados en diferentes artefactos, a medida que se baja el nivel de abstracción del dominio del problema, o a medida que se va avanzando en las fases del desarrollo de la pieza de software.

Estos dos conceptos se traducen en un conjunto de reglas que paulatinamente se irán mencionando en los diferentes capítulos, y que se constituyen en sugerencias para verificar la coherencia que deben mantener los diferentes artefactos que hacen parte de UN-MÉTODO.

En relación con los artefactos relativos al Contexto del Software, se tienen las siguientes reglas:

1. Los Actores del área del problema deben estar representados como conceptos del Esquema Preconceptual o como clases conceptuales del Modelo del Dominio.
2. El modelo verbal del dominio del problema suministra pistas en relación con los elementos que harán parte de los Esquemas Preconceptuales y los Modelos del Dominio. Para una mayor ampliación de este tema, consultar en [ZAPATA Y GÓMEZ, 2006] el capítulo 3, en el cual se describe la forma de identificar diferentes elementos que podrán ser de utilidad en el proceso de desarrollo de software a partir del modelo verbal.
3. Los conceptos del Esquema Preconceptual deben estar contenidos en el modelo del dominio como clases conceptuales o atributos.
4. Las relaciones del Esquema Preconceptual deben corresponder a relaciones de asociación en el Modelo del Dominio.

# Capítulo 3

## Entregable 2: Análisis del Problema

### 3.1. INTRODUCCIÓN

Una vez se ha establecido el primer nexo con la Organización y se tiene un buen conocimiento de ella y de la terminología del área, se continúa con el proceso de realización de reuniones periódicas con los interesados. En este caso, se trata de establecer otros elementos que permitirán realizar un diagnóstico de la organización, que suministre datos sobre la razón de ser del software y los motivos que apoyan su aparición. A partir de este momento, la labor del Ingeniero del Software se parece mucho a la labor que realizan los médicos cuando quieren establecer los males que aquejan a sus pacientes. Los resultados de estas reuniones se traducirán en varios tipos de diagramas y artefactos que tratarán de describir:

- Los procesos propios del área: qué hacen, quién los lleva a cabo, qué información involucran, cuánto duran, cuántas veces se llevan a cabo, dónde se llevan a cabo, qué regulaciones deben tener en cuenta, qué problemas tienen.
- En qué grado los procesos del área dan cuenta de los objetivos del área.
- Qué problemas deben ser resueltos (para dar satisfacción a los objetivos), y cuáles son sus causas.

### 3.2. ENTREGABLE

#### 3.2.1. Introducción

En esta sección se resume el enfoque que se piensa usar en el área del problema para dar satisfacción a sus objetivos. No siempre las soluciones a los problemas son de tipo informático; hay ocasiones en las que una solución informática puede complicar sustancialmente el proceso o hacerlo poco viable, y en estos casos la visión de la Ingeniería de Sistemas debe suministrar pistas en relación con cuál sería la mejora de los procesos que puede conducir a facilitar el desempeño de las funciones de los interesados, ayudándoles a alcanzar los objetivos de la organización y contribuyéndoles en la solución de sus problemas. No se trata en esta sección de repetir el modelo verbal o tratar de expresar verbalmente los diagramas que se harán en la sección.

En el caso de RAPIZZA Ltda. La información correspondiente a esta sección es:

El propietario de RAPIZZA Ltda., preocupado porque aún no se consigue el nivel de utilidades que se esperaría en este tipo de negocio, ha implementado una serie de estrategias que hasta el momento no han entregado resultados concretos a la pizzería. Él piensa que, si se lograra mejorar la eficiencia en la entrega de las pizzas, su política de cobro a los repartidores de los pedidos entregados fuera de tiempo podría surtir mejores resultados, pues considera que hay factores ajenos a los repartidores que están afectando las entregas y cuya solución podría tomar en consideración la construcción de una herramienta informática.

### **3.2.2. Procesos del Área**

El diagrama de procesos muestra las actividades de la organización y la forma como éstas se llevan a cabo. A través de este diagrama se puede apreciar la manera como actualmente la organización satisface sus objetivos. Dicho modelo proviene del análisis organizacional, y de él [HARRINGTON, 1991] presenta algunas versiones preliminares; el CDM incluye dentro de sus artefactos una versión del diagrama de procesos [ANDERSON Y WENDELKEN, 1996].

El modelo de procesos muestra el flujo de información; es decir la manera cómo se producen entradas y salidas de datos entre un proceso y otro. En un modelo de procesos se incluyen los siguientes elementos:

*Rol:* puede ser una persona, dependencia o grupo de personas, que realiza algún tipo de acción que está involucrada dentro del conjunto de pasos realizados para satisfacer los objetivos de la organización. Dentro de la simbología del modelo, por cada rol se debe trazar un carril horizontal, en el que se especificarán las acciones de las que es responsable dicho rol. Una persona dentro de la organización puede desempeñar muchos roles; sin embargo, en el diagrama de procesos se detallan uno a uno con carriles independientes.

*Proceso o Acción:* es una actividad que puede ser considerada como básica dentro de la organización y se representa a través de un rectángulo rotulado con su respectivo nombre. Un proceso es considerado una actividad de trabajo discreta, pues tiene principio y fin (que son representados a través de los dos lados opuestos del rectángulo).

*Evento:* define los hechos externos o condiciones internas al sistema (distintas al inicio o finalización de un proceso interno) que determinan el inicio de un conjunto de procesos asociados por relaciones de precedencia. Un evento se representa por una flecha gruesa rotulada con el nombre del evento. Si un evento causa el inicio de uno o más procesos se denomina disparador; si el evento es el efecto de la culminación de uno o más procesos se denomina resultado. En general, los eventos disparadores coinciden con el surgimiento de alguna necesidad o la llegada de una fecha o situación particular (por ejemplo, el fin de un mes o la

llegada de un pedido); los eventos de resultado se suelen expresar en términos de verbos en participio pasado y pueden enlazarse con eventos disparadores (el evento de resultado “orden diligenciada” puede enlazarse con el evento “llegada de una orden”).

*Condiciones:* corresponden a verificaciones de hechos particulares que sea necesario hacer dentro de un proceso ya que, dependiendo de si se cumplen o no, se debe seguir una secuencia de pasos específica. Las condiciones se representan a través de rombos de cuyas puntas deben salir flujos rotulados con los nombres de las alternativas de cada condición (por ejemplo “sí” o “no”).

*Datos o Almacenamientos:* representan información generada o requerida en los procesos. Para su representación se utiliza el símbolo de los datos almacenados de los diagramas de flujo convencionales, rotulados con el nombre de los datos.

*Características Almacenables:* Este elemento se usa en UN-MÉTODO para detallar los datos que se van a almacenar (por ejemplo, para un almacén “orden” probablemente se almacenen el número, la fecha y el producto solicitado). Para su representación se emplea el símbolo del display de los diagramas de flujo convencionales y se incluyen en una tabla denominada diccionario de datos (véase Tabla 3.5.), donde se realiza una descripción más detallada de su uso.

*Flujos:* también llamados intercambios, pueden ser de datos o de cosas físicas y los flujos temporales que muestran una secuencia de control donde haya necesidad de un orden particular entre los pasos. En el diagrama de procesos los flujos se representan por medio de flechas unidireccionales, que suelen ser continuas si se trata de flujos que indican secuencia entre los procesos, y punteadas si se trata de flujos desde o hacia los almacenamientos.

*Fin de proceso:* Indica la finalización de una rama del proceso, pues allí termina la secuencia del proceso. Se representa con una circunferencia con un círculo relleno en el centro.

*Etiqueta de continuidad:* posibilita la unión de elementos del diagrama de procesos que se encuentren lejanos; es una forma de referenciación que se representa con un número encerrado en un círculo después del elemento de partida y antes del elemento de llegada.

En la Figura 3.1. se puede apreciar un ejemplo del diagrama de procesos en el cual se señalan algunos de sus principales elementos. En la Figura 3.2. se muestra el diagrama de procesos correspondiente a RAPIZZA Ltda.

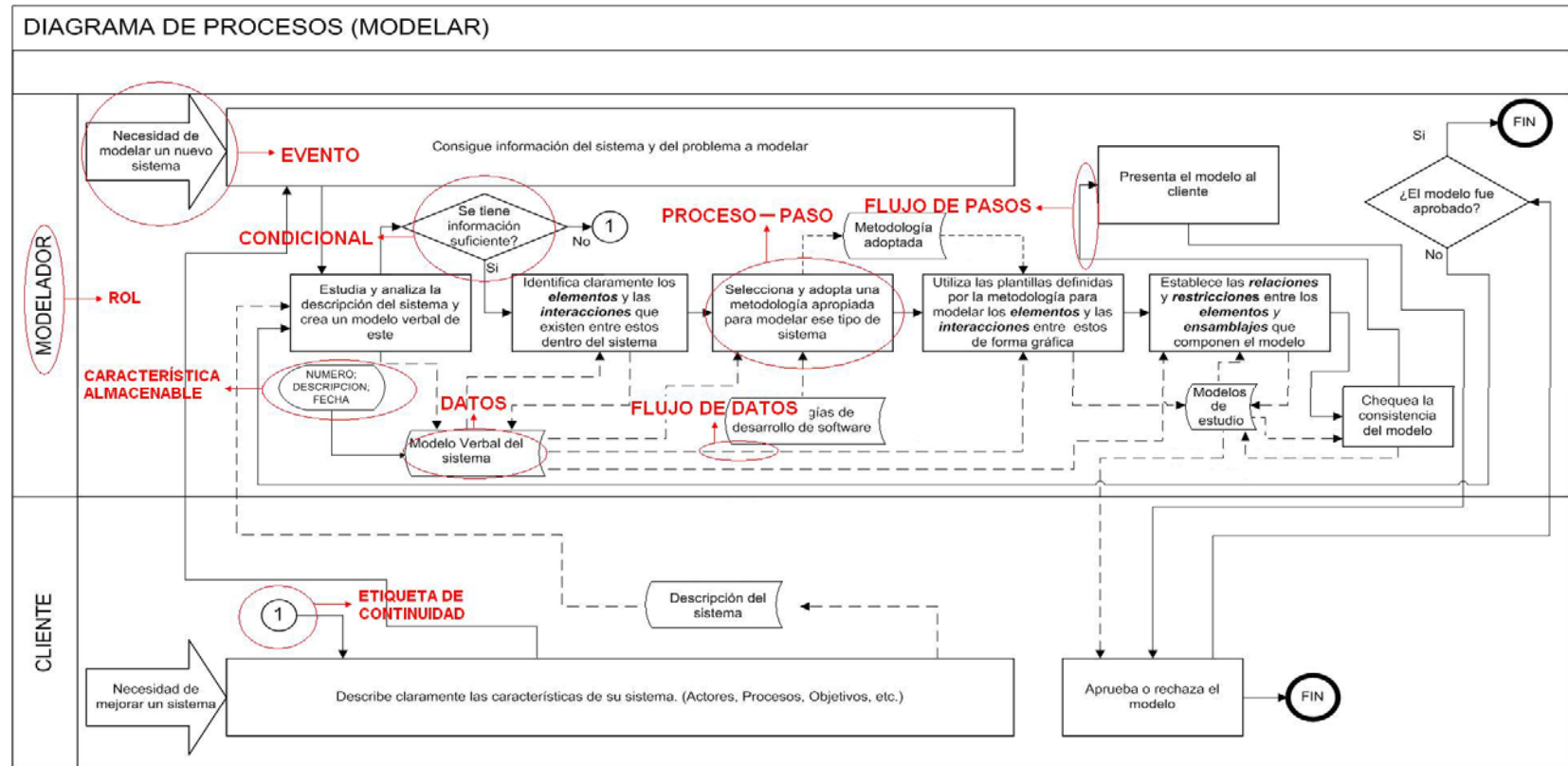


Figura 3.1. Ejemplo de Diagrama de Procesos con sus diferentes elementos.

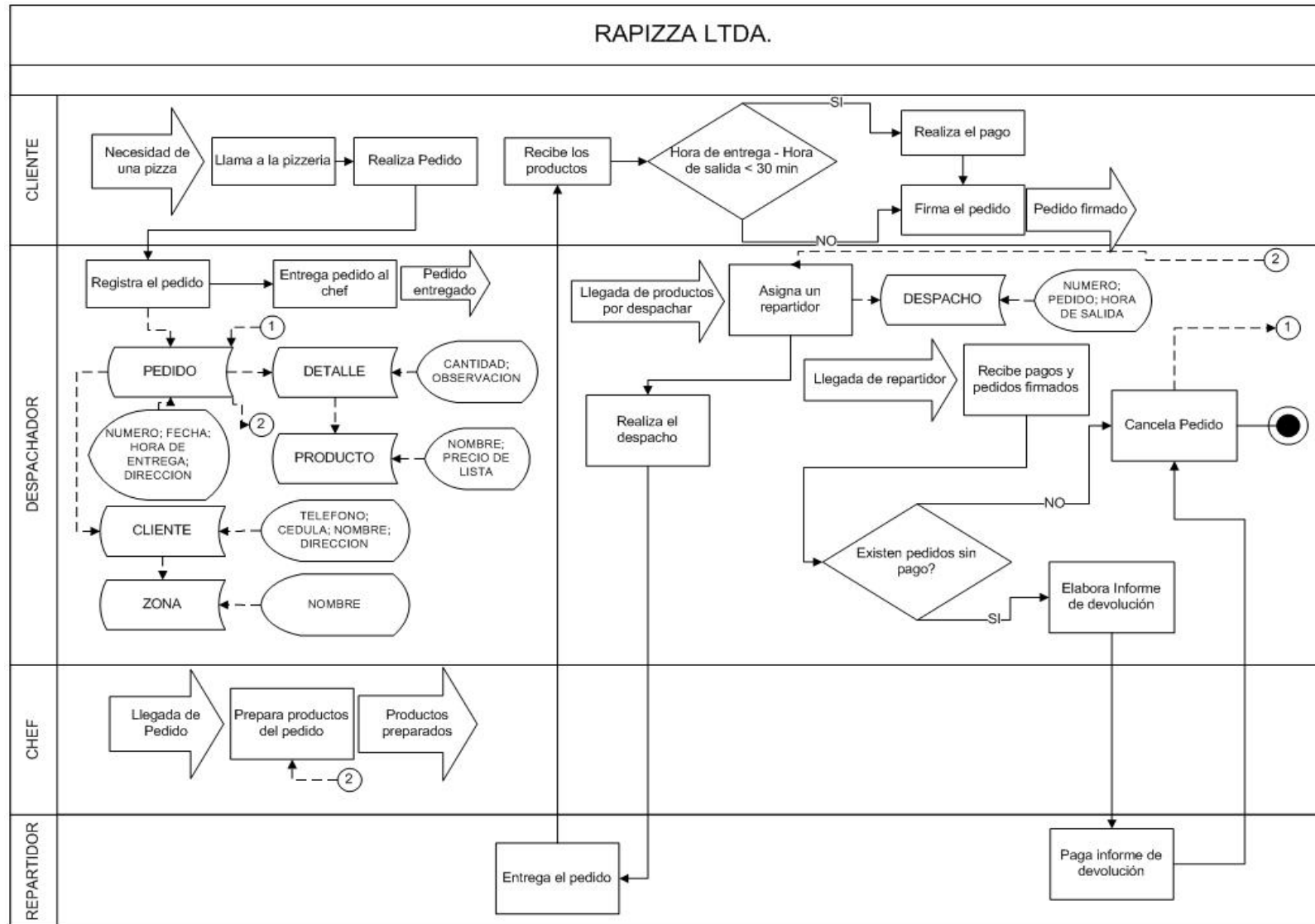


Figura 3.2. Diagrama de Procesos de la Situación Actual de RAPIZZA LTDA.

El diagrama de procesos muestra de manera esquemática el flujo de los procesos de la organización. Sin embargo, es necesario enlazar esos procesos con los objetivos de alto nivel de la organización, con el fin de determinar como contribuye cada proceso al logro de esos objetivos; además, se requiere identificar cuáles son los problemas que se pueden presentar en cada proceso, pues esta es una herramienta que permitirá intuir la necesidad de la solución informática. Finalmente, los procesos se deben ligar con las restricciones impuestas por la organización misma o el entorno que la rodea, haciendo explícitas las reglas del negocio que están relacionadas con un determinado proceso. En UN-MÉTODO, las relaciones de los procesos con estos elementos se materializan en la tabla explicativa de los procesos. Adicionalmente, los almacenes y actores del diagrama de procesos se deben detallar en un diccionario de datos.

### 3.2.2.1. Tabla Explicativa de los Procesos

La Tabla 3.1. se debe diligenciar para cada uno de los procesos incluidos en el diagrama de procesos.

<b>Nombre</b>	<b>Objetivo</b>	<b>Duración / Frecuencia</b>	<b>Cómo / Dónde</b>	<b>Problemas</b>	<b>Reglas del Negocio aplicadas</b>
<b>&lt;Nombre con el que se hará referencia al proceso&gt;</b>	<i>&lt;QUE lleva a cabo el proceso. Razón de ser del proceso. NOTA: No repita la descripción de la información requerida o producida por el proceso. Además, cada objetivo debe llevar un código correspondiente a un objetivo del diagrama de objetivos.&gt;</i>	<i>&lt;CUANTO TIEMPO/ CUANTAS VECES: <ul style="list-style-type: none"><li>Duraciones actual, deseable y máxima del proceso.</li><li>Número de veces que se lleva a cabo.</li></ul>&gt;</i>	<i>&lt; Resumen de la forma como se lleva a cabo el proceso (enfoque). Lugar donde se lleva a cabo NOTA: No repita una descomposición del proceso en sus subprocesos&gt;</i>	<i>&lt;Dificultades actuales en la ejecución del proceso en cuanto a su capacidad de lograr los resultados esperados con la calidad esperada (incluido el costo) y en el tiempo esperado  NOTA: Cada problema debe llevar un código correspondiente al diagrama Causa - Efecto&gt;</i>	<i>&lt;Coloque la referencia de las reglas del negocio aplicadas si es del caso NOTA: Cada regla debe llevar un código correspondiente a la tabla de reglas del negocio.&gt;</i>

Tabla 3.1. Definiciones de la Tabla Explicativa de los Procesos.

La columna “objetivo” establece la relación con el diagrama de objetivos (véase sección 3.2.3.) y la columna “problemas” establece la relación con el diagrama Causa – Efecto (véase sección 3.2.4.). La columna “Reglas del Negocio aplicadas” se debe relacionar con un artefacto complementario, que es también una tabla de



Reglas del Negocio (véase Tabla 3.2.); en esta tabla se deben incluir elementos como los siguientes:

- Las restricciones propias del negocio: por ejemplo “los profesores pueden cambiar de categoría después de cuatro años de servicio.
- Las fórmulas para la realización de cálculos: por ejemplo “la nota definitiva es el promedio ponderado de las notas parciales multiplicadas por sus pesos respectivos.
- Las limitaciones de acceso a la información: por ejemplo “únicamente el Director de la Escuela puede autorizar los viajes a congresos”.
- Las políticas de la organización: por ejemplo “cuando un estudiante pierde la misma materia por tercera vez, pierde la calidad de estudiante”.

En ocasiones, el espacio de la tabla es insuficiente porque es necesario citar de manera textual un documento de la organización. En este caso, se puede hacer la anotación en la tabla, referenciando el texto completo que deberá consignarse en los anexos del entregable.

En la elaboración de la Tabla de Reglas del Negocio, se deberían tomar en consideración las siguientes recomendaciones generales:

- Se deben distinguir las reglas del negocio de la descripción de los procesos. Las reglas del negocio son las restricciones que regulan el proceso y no los pasos del mismo (v.g. las fórmulas para efectuar los cálculos y resultados obligatorios y en general las políticas de la organización que deben ser satisfechas; estas reglas no deberían ser violadas por el proceso mismo).
- Se debe omitir la repetición de los elementos que se representan en el diagrama de procesos, en particular: Responsables (actores), Secuencias de procesos, Eventos e Información requerida o producida por el proceso. Si bien estos elementos La naturaleza de las restricciones es diferente a estos elementos; sin embargo, se debe notar que estos elementos pueden hacer parte de las reglas del negocio, si bien dichas reglas no deben repetir las funciones específicas de esos elementos.

Código	Nombre	Descripción	Fórmula	Fuente	Reglas del Negocio relacionadas
<Lo que identifica la regla. Se sugiere RN####>	<Coloque un nombre corto que sintetice lo que hace la regla>	<Un resumen de lo que debe realizar la regla del negocio>	<En caso de tratarse de un cálculo numérico, colocar aquí la formulación correspondiente>	<Persona, unidad organizacional o documento que suministró la información correspondiente>	<Códigos de las reglas que sean relativas a la regla que se describe>

Tabla 3.2. Reglas del Negocio.

En la Tabla 3.3. se ejemplifica la tabla explicativa de los procesos en el problema de RAPIZZA LTDA.

<b>Nombre</b>	<b>Objetivo</b>	<b>Duración / Frecuencia</b>	<b>Cómo / Dónde</b>	<b>Problemas</b>	<b>Reglas del Negocio aplicadas</b>
<b>Llama a la pizzería / realiza el pedido</b>	O10 Satisfacer las necesidades del Cliente en relación con los productos de la pizzería R4 Asegurar la recepción de pedidos telefónicos.	En promedio, una llamada dura de 3 a 5 minutos. Se atienden unas 50 llamadas diarias.	La comunicación se realiza vía telefónica, a partir de viviendas ubicadas en las zonas de cobertura.	C18 La Pizzería no responde la llamada. C19 El número telefónico está errado. C15 El Cliente no conoce el número telefónico de la pizzería.	RN001 Zonas de cobertura. RN002 Horario de atención.
<b>Registra el pedido</b>	O17 Facilitar al Cliente el acceso al producto R3 Conocer las necesidades del Cliente E6 Ofrecer los productos más solicitados E4 Ofrecer otros productos alternativos.	Se realiza simultáneamente con la llamada del cliente. No todas las llamadas incluyen pedidos, por lo cual se debe seleccionar cuáles registrar.	El conmutador central se encuentra en RAPIZZA Ltda. y es el Despachador el único responsable.	C14 No se entiende claramente qué es lo que quiere el Cliente. C16 El Cliente llama a una hora no adecuada (cierre de la pizzería, etc.). C5 El cliente vive o se encuentra fuera de la zona de cobertura. SP2 No se da abasto con la atención de los pedidos.	
<b>Entrega Pedido al Chef</b>	O15 Asegurar la producción del pedido solicitado por el Cliente.	Es un proceso inmediato. Se realiza sólo para las llamadas que se materializan en pedidos.	Se realiza a través de una ventana de comunicación con el chef ubicada en la pizzería.	C7 El chef está muy ocupado. C22 Los pedidos a veces se pierden.	RN003 Formato de entrega de pedidos.

Tabla 3.3. Tabla explicativa de los procesos de RAPIZZA Ltda. (continúa)

<b>Código</b>	<b>Nombre</b>	<b>Descripción</b>	<b>Fórmula</b>	<b>Fuente</b>	<b>Reglas del Negocio relacionadas</b>
<b>Prepara los productos del Pedido</b>	O10 Satisfacer las necesidades del Cliente en relación con la pizzería.	Usualmente dura entre 5 y 10 minutos. Se realiza para todos los pedidos.	Se realiza en la cocina de la pizzería, con base en el recetario oficial de la pizzería.	C10 No hay suficientes ingredientes para elaborar los productos solicitados. C8 La cantidad de productos solicitados excede las capacidades de producción del chef.	RN004 Recetario Oficial de la pizzería. RN005 Solicitud de compra de ingredientes.
<b>Asigna un repartidor / Realiza el Despacho</b>	E7 Garantizar el envío del producto al Cliente. R2 Garantizar rutas de entrega organizadas.	Se realiza para todos los pedidos, con base en la disponibilidad de repartidores. Se suele demorar menos de 2 minutos.	Se realiza en la oficina de despachos. El repartidor firma el pedido como aceptación de la asignación.	C11 No hay repartidores disponibles.	RN006 Criterios de asignación de repartidores.
<b>Entrega el pedido</b>	O10 Satisfacer las necesidades del cliente en relación con los productos de la pizzería	Dependiendo de la zona, el traslado puede tardar de 10 a 15 minutos. Se realiza para todos los pedidos.	Se realiza en la dirección entregada por el cliente. El repartidor se encarga de contactar al cliente.	C13 El producto entregado no corresponde al producto solicitado por el Cliente.	
<b>Recibe los productos</b>	O10 Satisfacer las necesidades del cliente en relación con los productos de la pizzería	Por lo general, menos de un minuto, mientras el cliente constata el pedido.	Se realiza en la dirección entregada por el cliente.	C13 El producto entregado no corresponde al producto solicitado por el Cliente.	RN007 Políticas de pago de productos.
<b>Realiza el pago</b>	O11 Asegurar el pago del Cliente.	Por lo general, menos de un minuto, siempre y cuando el repartidor haya llegado dentro del plazo.	Se realiza en la dirección entregada por el cliente.		RN007 Políticas de pago de productos.
<b>Firma el pedido</b>	R1 Asegurar registro de los pedidos.	Por lo general, menos de un minuto.	Se realiza en la dirección entregada por el cliente.	C21 El repartidor pierde el comprobante de pedido, inhabilitando al cliente para suministrar su firma.	RN007 Políticas de pago de productos.

Tabla 3.3. Tabla explicativa de los procesos de RAPIZZA Ltda. (Continuación)

<b>Código</b>	<b>Nombre</b>	<b>Descripción</b>	<b>Fórmula</b>	<b>Fuente</b>	<b>Reglas del Negocio relacionadas</b>
<b>Recibe pagos y pedidos firmados</b>	<i>O3 Garantizar la recuperación del dinero de las ventas.</i>	<i>Por lo general, menos de un minuto. Se debe realizar para todos los pedidos asignados a un repartidor.</i>	<i>Se realiza en la oficina de despachos. Todos los pedidos deben estar firmados.</i>	<i>C20 Se presenta pérdida de algunos pedidos firmados.</i>	<i>RN008 Políticas de cobro de devoluciones.</i>
<b>Elabora Cuenta de cobro / Paga Cuenta de Cobro</b>	<i>O7 Cumplir con las políticas de cobro a repartidores.</i>	<i>Entre tres y cinco minutos por pedido no pagado. Se realiza para todos aquellos pedidos que no tengan pago del cliente.</i>	<i>Se realiza en la oficina de despachos. El repartidor debe firmar la Cuenta de Cobro como aceptación del retraso. Se fija un plazo de pago.</i>	<i>C23 El despachador pierde el informe del repartidor, impidiéndole conocer con certeza la cantidad a cobrar.</i>	<i>RN008 Políticas de cobro de devoluciones.</i>
<b>Cancela pedido</b>	<i>R1 Asegurar registro de los pedidos.</i>	<i>Menos de un minuto por pedido. Se realiza para todos los pedidos y se debe diferenciar los pagados por el cliente de los que tienen Cuenta de Cobro.</i>	<i>Se realiza en la oficina de despachos. El despachador consigna en un listado cuáles pedidos se pagaron y cuáles generaron Cuenta de Cobro.</i>	<i>C20 Se presenta pérdida de algunos pedidos firmados.</i>	

Tabla 3.3. Tabla explicativa de los procesos de RAPIZZA Ltda. (Final)

En la Tabla 3.4. se incluyen las reglas del negocio correspondientes al mismo problema.

<b>Código</b>	<b>Nombre</b>	<b>Descripción</b>	<b>Fórmula</b>	<b>Fuente</b>	<b>Reglas del Negocio relacionadas</b>
<b>RN001</b>	<i>Zonas de Cobertura</i>	<i>La pizzería sólo atiende los barrios del centro – occidente de la ciudad.</i>		<i>Propietario</i>	
<b>RN002</b>	<i>Horario de atención.</i>	<i>Se presta servicio en el horario de lunes a viernes de 10:00 a.m. a 10:00 p.m.</i>		<i>Propietario</i>	

Tabla 3.4. Reglas del Negocio de RAPIZZA Ltda. (continúa)

Código	Nombre	Descripción	Fórmula	Fuente	Reglas del Negocio relacionadas
<b>RN003</b>	<i>Formato de entrega de pedidos.</i>	<i>El despachador debe llenar un formato en el que se especifiquen claramente los pedidos.</i>		<i>Manual de Funciones del despachador.</i>	
<b>RN004</b>	<i>Recetario Oficial de la Pizzería.</i>	<i>Los productos solicitados deberán estar incluidos en un recetario y ser realizados de conformidad con este documento.</i>		<i>Manual de funciones del Chef.</i>	<b>RN005</b>
<b>RN005</b>	<i>Solicitud de compra de ingredientes.</i>	<i>Los ingredientes sólo se pueden solicitar los diez primeros días de cada mes, con las proyecciones adecuadas.</i>		<i>Manual de funciones del Chef.</i>	<b>RN004</b>
<b>RN006</b>	<i>Criterios de asignación de repartidores.</i>	<i>Los repartidores se asignan por disponibilidad, zona e historial de entregas.</i>		<i>Manual de funciones del Despachador.</i>	
<b>RN007</b>	<i>Políticas de pago de productos.</i>	<i>RAPIZZA Ltda. Exime del pago de pedidos a sus clientes si el tiempo transcurrido entre la salida del pedido y su entrega es mayor a 30 minutos.</i>	<i>Hora de Entrega – Hora de Salida &lt;=30 minutos</i>	<i>Propietario</i>	<b>RN008</b>
<b>RN008</b>	<i>Políticas de cobro de devoluciones.</i>	<i>Las Cuentas de Cobro deben ser elaboradas para los pedidos no pagados por los Clientes y se deben cobrar a los repartidores responsables del no pago.</i>		<i>Propietario.</i>	<b>RN007</b>

Tabla 3.4. Reglas del Negocio de RAPIZZA Ltda. (final)

### 3.2.2.2. Diccionario de Datos

El diagrama de procesos incluye elementos como los almacenes y los actores, cuyo comportamiento se debería detallar para servir posteriormente para especificar su comportamiento. En UN-MÉTODO, cada diagrama de procesos se debe acompañar con la tabla de estructuración de los datos que se muestra en la Tabla 3.5. Se debe incluir una entrada en la tabla por cada dato o grupo de datos rotulado en el proceso; en general, los datos corresponderán a los almacenes o los actores del diagrama de procesos. Cabe anotar que dentro de “componentes”, en el caso de los almacenamientos del diagrama de procesos, se deben incluir las “características almacenables” que se unen a los almacenes.

No se debe confundir el Diccionario de Datos con el Diccionario de Términos que se plantea en la sección 2.2.3., puesto que la información en ambos casos es

diferente, aunque complementaria. El Diccionario de Términos contiene las definiciones de diferentes términos que se usan en el desarrollo del problema, en tanto que el Diccionario de Datos examina algunos de esos términos que pueden ser considerados como “Datos” dentro del dominio del problema para analizar su estructura con miras a una representación posterior de los mismos.

<b>Nombre</b>	<b>Alias</b>	<b>Cómo/Dónde</b>	<b>Descripción</b>	<b>Componentes</b>
<i>&lt;Nombre del dato&gt;</i>	<i>&lt;otros nombres con los que se puede conocer el dato&gt;.</i>	<i>&lt;Establecer si el dato es una entidad externa o un almacenamiento y si es de ingreso o salida de datos&gt;</i>	<i>&lt;Colocar brevemente los procesos en los cuales se involucra el dato&gt;.</i>	<i>&lt;Coloque aquí las características principales que se deban almacenar de los diferentes datos&gt;</i>

Tabla 3.5. Diccionario de Datos

En la Tabla 3.6. se puede apreciar el Diccionario de Datos correspondiente a RAPIZZA Ltda.

<b>Nombre</b>	<b>Alias</b>	<b>Cómo/Dónde</b>	<b>Descripción</b>	<b>Componentes</b>
<b>Pedido</b>		<i>Almacén que se emplea como entrada y salida de datos en casi todos los procesos del dominio.</i>	<i>Registra el pedido, entrega el pedido al chef, prepara los productos del pedido, despacha y entrega el pedido, realiza el pago, firma el pedido, cancela el pedido.</i>	<i>Número, Fecha, Hora de entrega.</i>
<b>Despacho</b>		<i>Almacén que se emplea para agrupar los pedidos que se asignan a un repartidor</i>	<i>Realiza el despacho</i>	<i>Número, pedidos, Hora de Salida</i>
<b>Detalle</b>		<i>Almacén que se emplea para especificar los productos del pedido.</i>	<i>Realiza el pedido, registra el pedido, entrega pedido al chef, prepara productos del pedido, recibe los productos.</i>	<i>Cantidad, Observación.</i>
<b>Producto</b>	<i>Pizzas</i>	<i>Almacén que se emplea como salida de datos hacia el detalle de pedido.</i>	<i>Realiza el pedido, registra el pedido, entrega pedido al chef, prepara productos del pedido, recibe los productos.</i>	<i>Nombre, precio de lista.</i>
<b>Cliente</b>	<i>Comprador</i>	<i>Actúa de manera dual como almacén y como entidad externa.</i>	<i>Realiza pedido, registra pedido, recibe los productos, realiza el pago, firma el pedido.</i>	<i>Teléfono, Cédula, Nombre, dirección.</i>
<b>Zona</b>	<i>Ubicación</i>	<i>Almacén que se emplea para determinar la viabilidad del despacho al cliente.</i>	<i>Registra el pedido, asigna un repartidor, despacha el pedido, entrega el pedido</i>	<i>Nombre</i>

Tabla 3.6. Diccionario de Datos correspondiente a RAPIZZA Ltda. (continúa)

Nombre	Alias	Cómo/Dónde	Descripción	Componentes
<b>Despachador</b>		<i>Entidad externa.</i>	<i>Registra el pedido, entrega pedido al chef, asigna un repartidor, despacha el pedido, recibe pagos y pedidos firmados, elabora cuentas de cobro, cancela pedido.</i>	
<b>Chef</b>		<i>Entidad Externa.</i>	<i>Prepara productos del pedido.</i>	
<b>Repartidor</b>		<i>Entidad Externa.</i>	<i>Entrega el pedido, paga cuentas de cobro.</i>	

Tabla 3.6. Diccionario de Datos correspondiente a RAPIZZA Ltda. (final)

### 3.2.3. Objetivos del Área

Para modelar la estructura de los objetivos del área, en UN-MÉTODO se emplea un diagrama que hace parte de la metodología KAOS (Knowledge Acquisition autOmated Specification o Especificación Automatizada de la Adquisición del Conocimiento), la cual fue presentada por [DARDENNE ET AL., 1993]. En este diagrama, se parte de los objetivos de más alto nivel de la organización, los cuales se van subrogando paulatinamente en otros objetivos hasta alcanzar el los requisitos que la pieza de software deberá cumplir para satisfacer los objetivos organizacionales. El proceso para el trazado del diagrama de objetivos de KAOS requiere que se definan los objetivos secundarios que subrogan los objetivos generales, para luego presentar los objetivos más elementales que los subrogan, y así sucesivamente hasta llegar a objetivos que se consideren elementales o atómicos (que no vale la pena descomponer) o hasta que se alcancen expectativas, requisitos o propiedades del dominio, como se definen más adelante.

La simbología básica del diagrama de objetivos se presenta en la Figura 3.3., donde se puede apreciar adicionalmente la explicación del “por qué” de un determinado objetivo (moviéndose hacia abajo en el diagrama) y la explicación del “cómo” de un grupo de objetivos (moviéndose hacia arriba en el diagrama).

Los elementos que se incluyen en la Figura 3.3. son los siguientes:

**Objetivo:** Fin o intento que se espera lograr con un proceso o actividad o incluso con la misión de una organización.

**Requisito:** Un objetivo a nivel de la solución informática que no será negociable dentro del proceso de desarrollo.

**Expectativa:** Un objetivo a nivel de la solución informática que no se espera que se incorpore a la misma, pero que podría ser deseable siempre y cuando las condiciones del proyecto lo permitiesen.

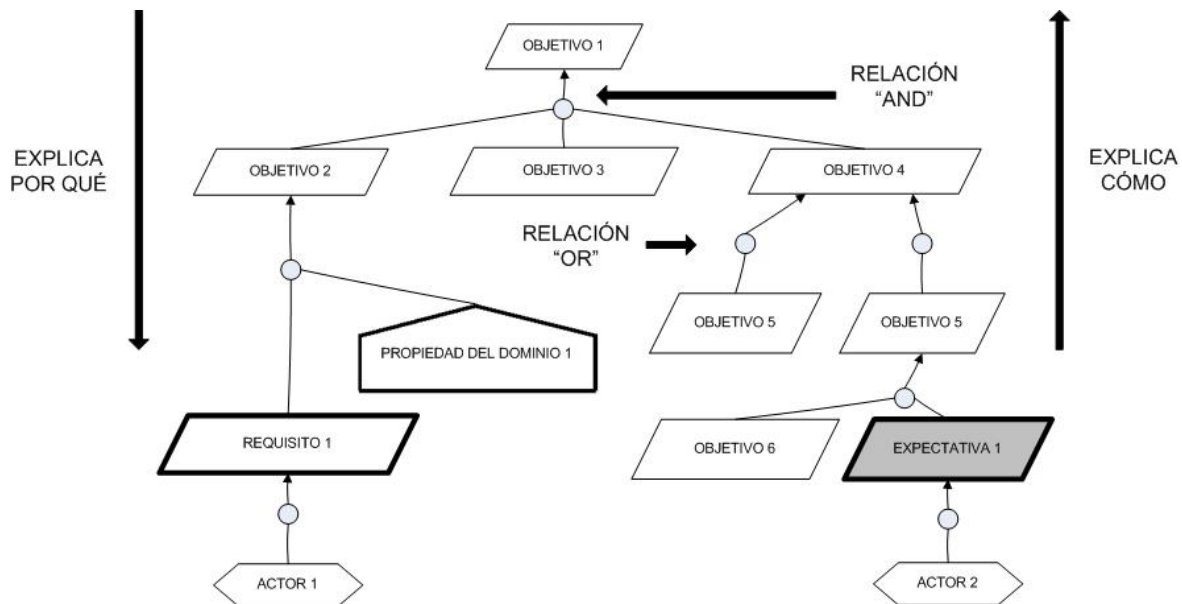


Figura 3.3. Simbología básica del Diagrama de Objetivos de KAOS.

*Actor:* Correspondiente al “rol” del diagrama de procesos. Es el responsable por un requisito o una expectativa.

*Propiedad del dominio:* Es una propiedad que se requiere para que los objetivos se alcancen a un determinado nivel.

*Conectores:* Flechas que vinculan objetivos, expectativas, requisitos, propiedades del dominio y actores. Pueden pertenecer a relaciones tipo “AND”, cuando se requiere el cumplimiento simultáneo de dos o más objetivos unidos mediante el conector para que el objetivo que subrogan se cumpla, o a relaciones tipo “OR” cuando es suficiente con el cumplimiento de uno de los objetivos unidos mediante el conector.

Con el Diagrama de Objetivos se busca señalar cuáles de los objetivos subrogados son satisfechos actualmente por medio de los procesos del área. Se debe notar que la incapacidad de dar satisfacción a los objetivos generales está asociada a la incapacidad de dar satisfacción a los objetivos subrogados.

Los diferentes elementos del Diagrama de Objetivos se deben codificar para permitir su referenciación en otros diagramas y artefactos de UN-MÉTODO. Se sugiere que la codificación sea así: O# (objetivos), R# (requisitos), E# (expectativas), P# (propiedad del dominio), A# (actor).



En KAOS se presentan dos criterios que permiten evaluar la completitud de los diagramas, a saber:

- Un Modelo de Objetivos se dice completo respecto de sus relaciones de refinamiento “sí y solo si” cada objetivo hoja es una expectativa, una propiedad del dominio o un requisito.
- Un Modelo de Objetivos es completo respecto de sus relaciones de responsabilidad “sí y solo si” cada requisito está bajo la responsabilidad de uno y sólo un agente (ya sea explícita o implícitamente si el requisito refina otro que se ha colocado bajo la responsabilidad del mismo agente).

El Diagrama de Objetivos se realiza no sólo para el área problemática, sino para la organización que la rodea; se realiza de esta manera porque se requiere que la pieza de software del área que la requiere se contextualice y se pueda determinar su influencia en la organización, evaluando la viabilidad de cumplimiento de los objetivos subrogados pertenecientes a esa área, pero determinando adicionalmente su influencia en los objetivos de alto nivel de la organización. Es por ello que, si se fuera a aplicar rigurosamente el primer criterio anotado, cualquier pieza de software que se buscara desarrollar debería poseer completo el diagrama de objetivos de la organización, el cual rara vez se tiene en las diferentes organizaciones. En la práctica, en UN-MÉTODO basta con describir las ramas correspondientes a la solución misma, dejando explícito dentro del diagrama que habrá ramas del árbol de objetivos que no se desarrollarán por no ser relevantes para el problema que se pretende resolver. Sin embargo, es necesario que, en las ramas que se desarrollen del diagrama de objetivos, el primer criterio de completitud sí es aplicable, lo cual implica que esas ramas deberán terminar en un requisito, propiedad del dominio y/o expectativa. El segundo criterio de completitud expuesto también se deberá cumplir para aquellas ramas que se desarrollen del Diagrama de Objetivos.

Si bien KAOS posee otros diagramas para la elicitación de requisitos del software, en UN-MÉTODO únicamente se emplea el Diagrama de Objetivos de esta metodología (KAOS). Los demás programas como el de responsabilidades y el de actividades, emplean también una notación propia y particular del esquema KAOS, que se aleja un poco del estándar definido para el modelamiento de las piezas de software, que es el UML. En UN-MÉTODO sólo se emplea el de objetivos, puesto que los demás diagramas están atendidos de alguna manera en el estándar de UML.

En las Figuras 3.4. y 3.5. se puede apreciar el Diagrama de Objetivos correspondiente a RAPIZZA Ltda. Nótese que las ramas que no se van a desarrollar de dicho diagrama se marcan de manera explícita y que en las ramas que se desarrollan se pueden verificar los dos criterios de completitud.

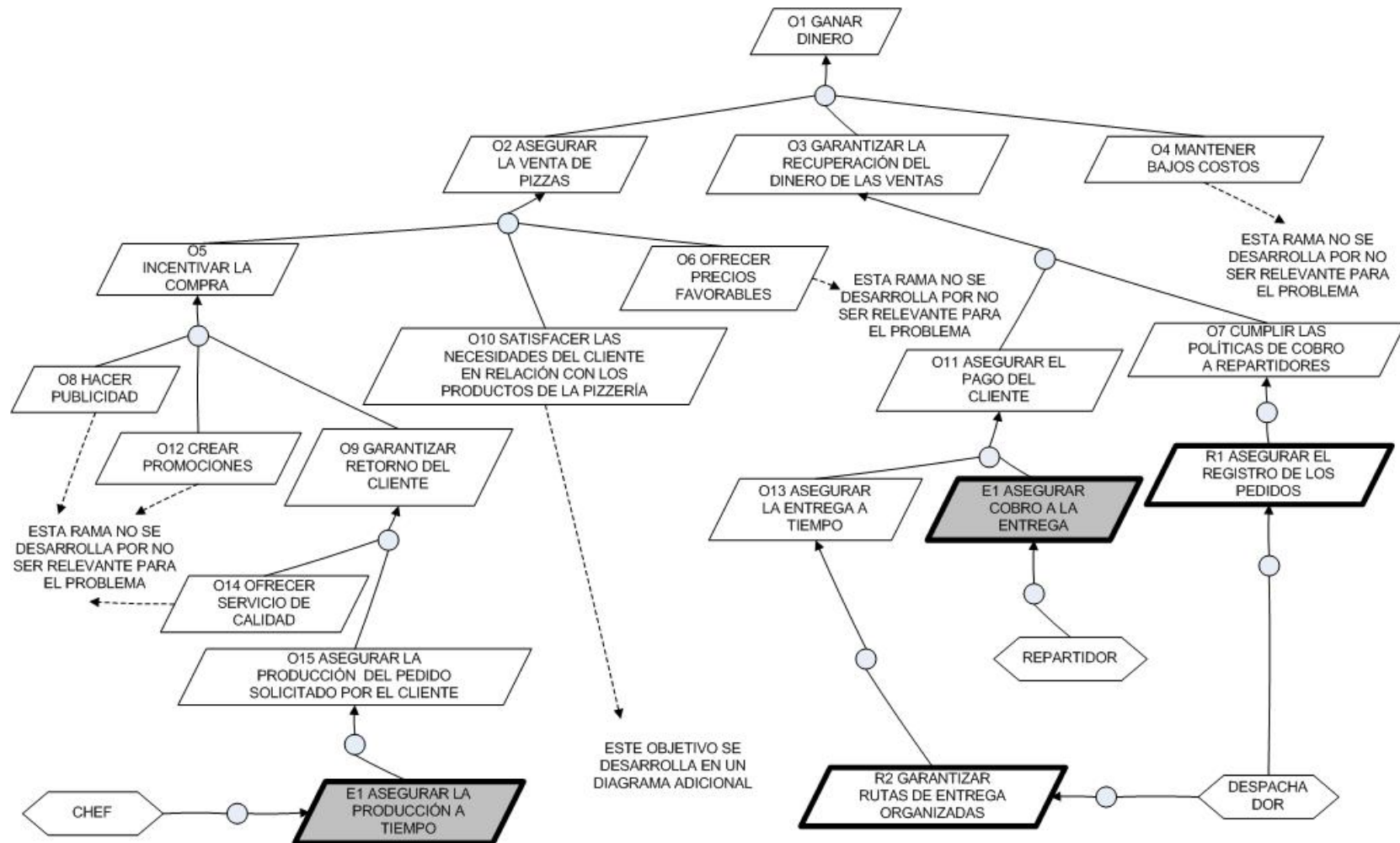


Figura 3.4. Diagrama de Objetivos correspondiente a RAPIZZA LTDA.

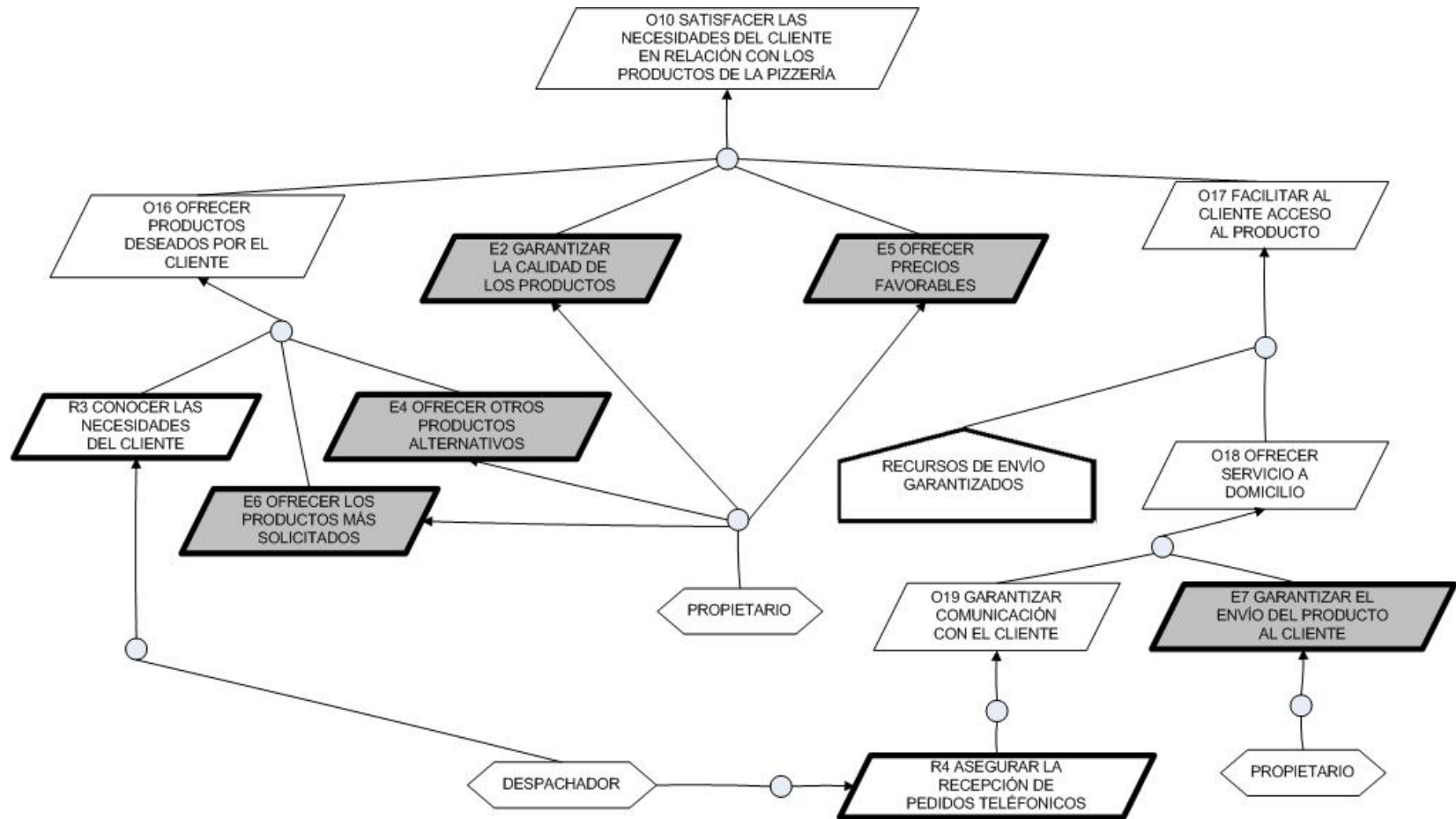


Figura 3.5. Detalle del objetivo “Satisfacer las necesidades del cliente en relación con los productos de la pizzería”, correspondiente a la Figura 3.4.

### 3.2.4. Problemas y sus Causas

Los procesos que se realizan en la organización tienen no sólo objetivos asociados, que se pueden estructurar como se describió en la sección anterior, sino que en esos procesos se pueden encontrar problemas asociados, los cuales también se pueden estructurar, usando para ello una técnica que proviene de la teoría organizacional y la Calidad Total: el diagrama Causa – Efecto [ISHIKAWA, 1986]

Este diagrama le permite al analista estructurar y jerarquizar los problemas que identifica durante las entrevistas que realiza con el interesado para, de esta forma, tomar decisiones respecto a cuál deberá ser el área en la que se enfoque su trabajo. En otras palabras, a través del análisis que aquí se origina, el analista podrá decidir qué problemas deberá atacar en su totalidad, y cuáles podrá omitir o atacar parcialmente; claro que no debe alejarse del objetivo general que justifica la creación del software. Este diagrama también es conocido como espina de pescado por la similitud de su apariencia física con la de un esqueleto de un pez o como diagrama de Ishikawa en honor a su creador [ISHIKAWA, 1986].

Para la elaboración del diagrama causa-efecto, cuyas principales componentes se pueden apreciar en la Figura 3.6., se puede proceder de dos formas:

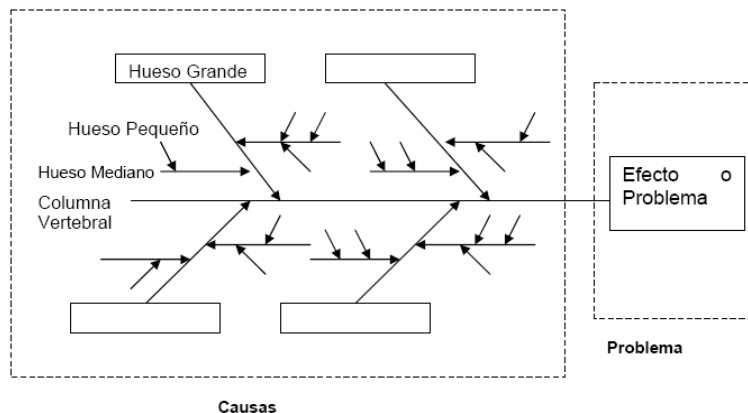


Figura 3.6. Principales componentes del Diagrama Causa-Efecto.

- Listar todos los problemas que son identificados (un tipo de lluvia de ideas), para luego intentar jerarquizarlos y estructurarlos identificando cuáles son problemas principales y cuáles son las causas de estos problemas; este procedimiento se realiza reiteradas veces hasta que se logren ubicar todos los problemas encontrados, o hasta que las causas que se tengan sean consideradas atómicas.

- Identificar los problemas principales y ubicarlos como “huesos primarios” y, posteriormente, comenzar a identificar causas secundarias que se ubicarán en “huesos pequeños” que se desprenderán todos de las ramas principales.

En cualquiera de los casos, en un diagrama causa-efecto se debe identificar un problema principal que logre encerrar la problemática del área en la que se concentrará el trabajo del analista, el cual deberá coincidir con un objetivo no satisfecho o parcialmente satisfecho. Es recomendable prestar especial cuidado a este aspecto, ya que de la correcta identificación de dicho problema depende si se obtiene o no un diagrama correcto y completo.

Nótese que los problemas más generales son la incapacidad del área de dar satisfacción a sus objetivos generales, y que esto es debido a la incapacidad de dar cuenta de los objetivos subrogados. Se puede presentar esta incapacidad por dos causas fundamentales: debido a la inexistencia de procesos que tengan estos objetivos o a los problemas de los procesos existentes.

El diagrama Causa-Efecto correspondiente a RAPIZZA Ltda. se muestra en la Figura 3.7.

### **3.3. ASPECTOS DE CONSISTENCIA Y REFINAMIENTO EN UN-MÉTODO**

El Análisis del Problema aún guarda una estrecha relación con el Contexto del Software que se estudió en el capítulo anterior. En especial, el diagrama de procesos tiene una serie de características que deben estar contenidas en el Esquema Preconceptual y, consecuentemente, en el Modelo del Dominio. Además, el desarrollo de los diferentes diagramas de UN-MÉTODO presenta algunos aspectos de consistencia que se deben preservar en su realización.

El listado de reglas de consistencia para el Análisis del Problema es el siguiente:

1. Los verbos incluidos en los procesos del Diagrama de Procesos deben corresponder a relaciones dinámicas del Esquema Preconceptual.
2. Los roles, almacenes y características almacenables del Diagrama de Procesos deben estar incluidos como conceptos del Esquema Preconceptual. Las relaciones “es” y “tiene” del Esquema Preconceptual suelen incluirse implícitamente en el manejo de los almacenes y las características almacenables de los mismos.
3. La Tabla Explicativa de los procesos debe ligar los procesos del Diagrama de Procesos con los objetivos del Diagrama de Objetivos, por lo cual para cada proceso debe existir mínimo un objetivo asociado; igualmente, si existen problemas asociados en los procesos de dicha tabla, esos problemas deben estar contenidos en el Diagrama Causa-Efecto. La ausencia de problemas en un proceso en particular se permite en la Tabla Explicativa de los Procesos, puesto que permite inferir que el proceso en

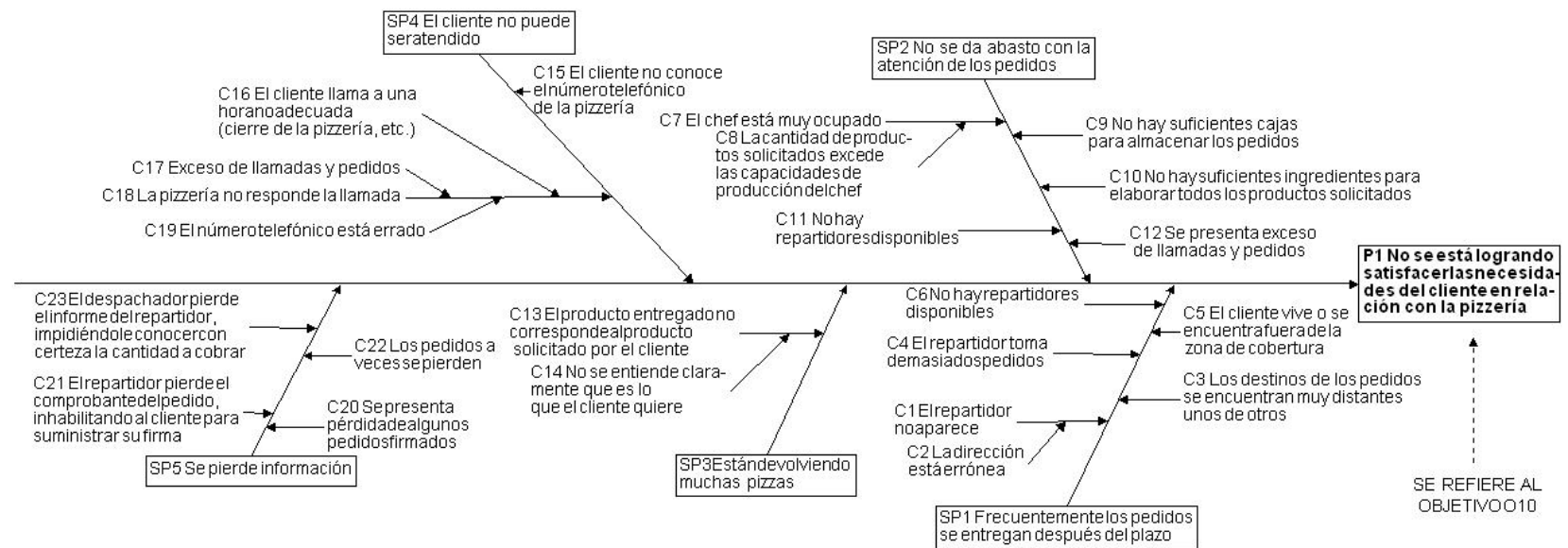


Figura 3.7. Diagrama Causa-Efecto correspondiente a RAPIZZA LTDA.

cuestión no presenta problemas que estén afectando el desempeño de la organización o que estén causando insatisfacción en los objetivos del proceso. Estas reglas de consistencia se expresan en las Figuras 3.8. y 3.9., tomadas de [ZAPATA et al., 2006b].

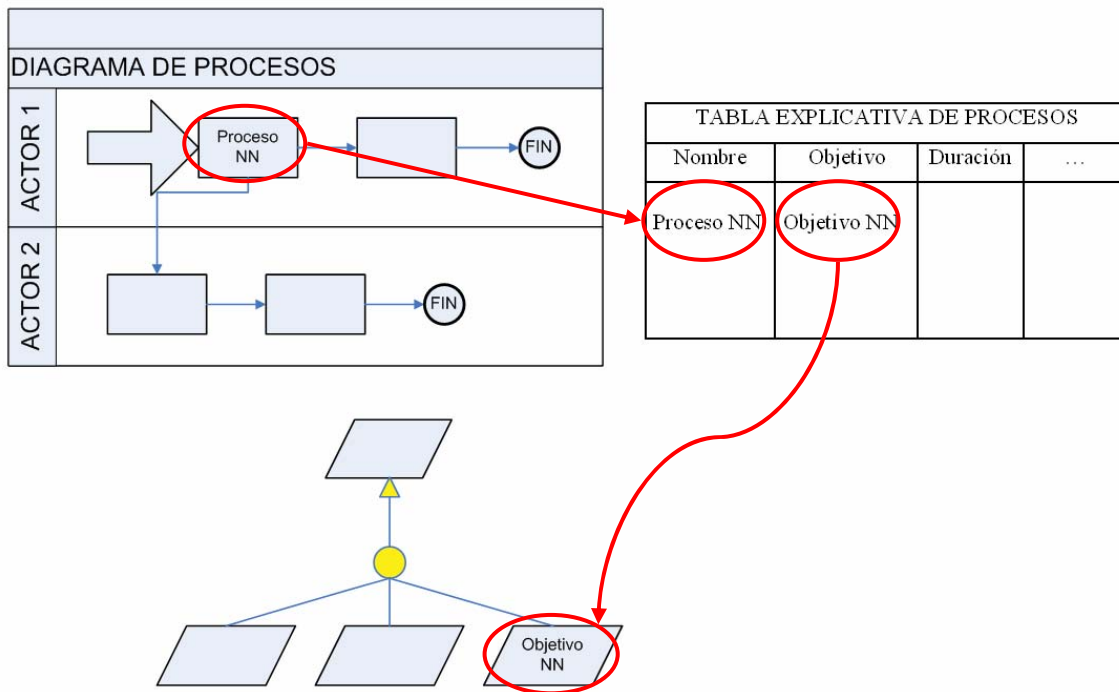


Figura 3.8. Regla de consistencia entre la Tabla Explicativa de Procesos y los Diagramas de Procesos y Objetivos.

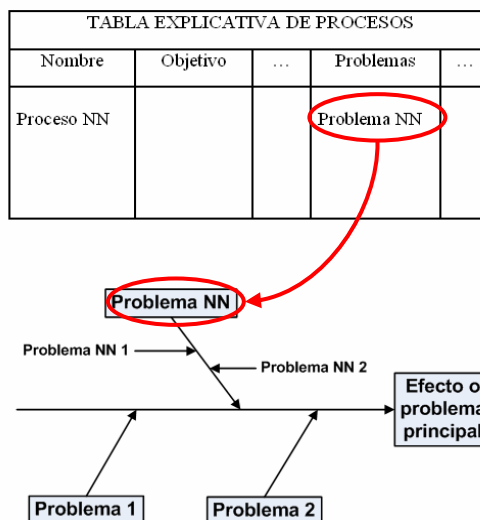


Figura 3.9. Regla de consistencia entre la Tabla Explicativa de Procesos y el Diagrama Causa-Efecto.

4. Las reglas de completitud del Diagrama de Objetivos, enunciadas en la sección 3.2.3., se deben cumplir únicamente para aquellas ramas que contribuyan a la comprensión del dominio del problema. Se debe hacer explícito el hecho de que las ramas restantes no se desarrollarán, como se hizo en la Figura 3.4.
5. El problema principal de un Diagrama Causa-Efecto se debe referir a la falta de satisfacción de un objetivo del Diagrama de Objetivos.
6. Algunos errores comunes en la elaboración del Diagrama de Procesos son los siguientes:
7. Colocar verbos de procesos en los eventos disparadores y/o de resultados. Por ejemplo “diligenciar el formato” es un verbo, pero “llegada del fin de mes” puede ser el nombre de un evento que anteceda al proceso “diligenciar el formato”. Los eventos de resultado siempre se deben manejar con un verbo en participio pasado, como en “formato diligenciado”.
8. Generar “puntos muertos” en las secuencias de los procesos. A los procesos entran flujos de procesos y eventos disparadores y salen flujos de procesos o eventos de resultados. Un proceso sin entradas o sin salidas (así sea hacia un fin de proceso) no tiene sentido porque genera un “punto muerto” en el cual el proceso se estanca y no fluye su información. Se debe tener presente que los flujos desde y hacia los almacenamientos no reemplazan los flujos desde y hacia los procesos, por lo cual, al momento de evaluar la continuidad de un Diagrama de Procesos, se recomienda ignorar los flujos desde y hacia los almacenes.
9. Intercambiar la convención de las líneas de flujo de datos y procesos. Las líneas de flujo de procesos son continuas y las de flujo de datos desde y hacia los almacenes son punteadas.
10. En los condicionales siempre debe haber un flujo de entrada y mínimo dos flujos de salida.
11. Los eventos terminales no sirven como eventos disparadores simultáneamente.
12. Algunos errores comunes en la elaboración del Diagrama Causa – Efecto son:
  - Falta legibilidad en los diagramas. La redacción debe ser tal que simplemente agregando la palabra "porque..." se puede leer el diagrama hacia atrás. (Por ejemplo: A porque B porque C).
  - Muchas veces los nombres de los elementos (problemas, subproblemas o causas) no permiten definir una causa o un problema. Por ejemplo "registro de pedido" no parece decir cuál es el problema, en tanto que “no siempre se registra la información del pedido” podría ser una causa de problemas en la organización.



# Capítulo 4

## Entregable 3: Propuestas de Solución

### 4.1. INTRODUCCIÓN

Una vez se identifican los diferentes conceptos del dominio y cuando el problema se haya analizado de manera satisfactoria para el analista y el interesado, mediando para ello los diagramas y artefactos descritos en el capítulo anterior, se procede ahora a determinar cuáles son las características más relevantes que podrían llegar a integrar la solución. Al respecto, se debe hacer claridad en el hecho de que las soluciones no siempre serán informáticas; es posible que, después del Análisis del Problema se encuentre que una pieza de software no contribuiría sustancialmente a la solución de los problemas de la organización, porque se puede requerir previamente un cambio organizacional sustancial, por ejemplo en las funciones de los diferentes actores involucrados en los problemas. Además, en ocasiones, y especialmente en organizaciones pequeñas, es posible que una pieza de software en lugar de solucionar los problemas existentes introduzca problemas nuevos, que hagan que la solución de tipo informático no sea viable; por ejemplo, si RAPIZZA LTDA. fuera tan pequeña que el nivel de pedidos pudiese ser manejado eficientemente por el despachador sin necesidad de una aplicación informática, pero la causa C7 (el chef está muy ocupado) fuera completamente determinante en el problema, es probable que únicamente con contratar otro chef se pudiera solucionar gran parte del problema principal (No se está logrando satisfacer las necesidades del cliente en relación con la pizzería).

Con un examen minucioso del Análisis del Problema enunciado en el capítulo anterior se puede determinar un conjunto de soluciones posibles, desde las más sencillas hasta las más complejas, en las cuales se atiendan las causas de los problemas en mayor o menor grado. Es de notar que las soluciones en informática siempre son infinitas, en el sentido de que existirá una gama de soluciones con sutiles o grandes modificaciones entre una y otra, que atiendan los problemas de la organización; de esa gama infinita habrá que seleccionar un subconjunto de soluciones que puedan ser realizables para la organización y sobre las cuales se pueda tomar una decisión razonable desde el punto de vista económico pero también con base en criterios de usabilidad y solución de las causas de los problemas de la organización.

La decisión sobre cuál solución implementar correrá a cargo del Cliente, pero los elementos que le permitan tomar la decisión más adecuada deberán ser provistos

por el Analista, siguiendo los parámetros definidos en este capítulo. UN-MÉTODO provee un conjunto de artefactos que permiten caracterizar la solución, pero que, a la vez, categorizan las soluciones desde el punto de vista de costos y valoración.

En este capítulo se discuten las definiciones que deben se deben hacer para cada solución propuesta, las cuales contemplan:

- La nueva estructura organizacional del área.
- Los nuevos procesos propios del área: qué hacen, quién los lleva a cabo, qué información involucran, cuánto duran, cuántas veces se llevan a cabo, dónde se llevan a cabo, qué regulaciones se deben tener en cuenta, qué problemas tienen.
- La manera como participa el sistema informático en los nuevos procesos del área.
- Efecto de la nueva forma de proceder sobre los problemas y objetivos del área.
- Costo aproximado de la nueva forma de proceder.
- Factores críticos del éxito de la nueva propuesta: elementos básicos de un proyecto de desarrollo e implantación.

## **4.2. ENTREGABLE**

### **4.2.1. Introducción**

En esta sección se debe describir verbalmente el enfoque de la solución propuesta, así:

- Características generales de la solución.
- Causas de problemas que se pretenden aliviar.
- Cambios a la estructura organizacional indicando los cambios en las responsabilidades y funciones de las partes de la organización.
- Aparición de nuevos actores e interesados indicando sus roles e intereses; se debe reportar también la supresión de actores e interesados existentes.
- Breve descripción de los nuevos procesos (luego se colocará un diagrama más detallado) incluyendo sus necesidades de datos.
- Breve descripción de la forma como participa el sistema informático en estos nuevos procesos (luego se colocará una descripción más detallada por medio de los casos de uso)

Para RAPIZZA Ltda., a manera de ejemplo, sólo se presentará una solución. Sin embargo, se debe hacer énfasis en que podrían ser infinitas soluciones. Las características de la solución planteada son:

- Características generales de la solución: RAPIZZA Ltda. es una organización pequeña y, aún una solución limitada, podría contribuir notablemente en la mejora de sus procesos. Es por ello que la solución no cambiará radicalmente la forma de proceder sino que apoyará con el computador la gestión del despachador, cuyos procesos presentan los problemas que más están influyendo en la gestión de la organización. Se respetarán, eso sí, los cuatro requisitos generales identificados del diagrama de objetivos (Garantizar rutas de entrega organizadas, Asegurar el registro de los pedidos, Asegurar la recepción de los pedidos telefónicos y Conocer las necesidades del cliente).
- Causas de problemas que se pretenden aliviar: Las causas que pueden ser susceptibles de actuación son las siguientes:
  - No hay repartidores disponibles, el repartidor toma demasiados pedidos y los destinos de los pedidos se encuentran muy distantes unos de otros: Para estas causas se pretende dotar al Despachador de un sistema informático con la capacidad de organizar las rutas de los repartidores, de forma tal que el despachador no pierda tiempo en la asignación de los repartidores, la cual se realizará de forma automática en el sistema; con ello se pretende optimizar la ruta de los diferentes repartidores y minimizar el tiempo invertido en la elaboración manual de las rutas, lo cual permitirá aliviar la carga del despachador y le liberará tiempo para otras funciones.
  - El cliente vive o se encuentra fuera de la zona de cobertura: En este caso, y también para agilizar la atención del cliente por parte del despachador, el sistema informático será quien informe si es posible atender un pedido o no, estableciendo de manera automática la pertenencia a las zonas permitidas para la entrega.
  - Se pierde información: en general, el registro de toda la información (pedidos, clientes, rutas, zonas), minimizará la posibilidad de pérdida de documentos físicos.
  - No se entiende claramente lo que el cliente quiere: A este respecto, una de las innovaciones de esta solución es el envío del catálogo detallado de productos por e-mail. Con esta parte de la solución se presentará mayor familiaridad de los clientes con los productos de la compañía y se pueden agilizar de esta manera los pedidos.
- Cambios a la estructura organizacional: En cuanto a las funciones, no cambian decididamente las responsabilidades de los roles involucrados; el rol que más cambios sufre es el despachador, que se apoyará en la herramienta informática. La única función nueva que surge es la del envío del catálogo visual por e-mail mediante el sistema informático, la asignación de repartidor usando un cálculo interno del sistema y el cálculo de la zona

para determinar rápidamente si se puede o no atender el pedido, para evitar retrasos en las entregas por causa de malas interpretaciones en los límites de las zonas.

- Aparición de nuevos actores e interesados y supresión de actores e interesados existentes: Dado que la solución es bastante simple en este caso, no se presentarán supresiones de cargo o adiciones de nuevo personal.
- Breve descripción de los nuevos procesos: La realización del cálculo de la zona requiere que la dirección se desagregue en dos nomencladores correspondientes a las calles o carreras en las que se ubica el cliente y otro adicional para el número de la casa. El envío del catálogo digital requiere que se elabore el catálogo, para disponer de él cuando se necesite una claridad mayor por parte del cliente. El proceso de asignación de repartidor utilizará los mismos datos de nomencladores para categorizar las rutas por cercanía. Por lo demás, se requiere la alimentación del sistema informático con los datos de personas, pedidos, pagos y cuentas de cobro.
- Breve descripción de la forma como participa el sistema informático en estos nuevos procesos: Se facilitará el envío de los catálogos mediante un mensaje genérico, pero personalizado con el nombre y el e-mail del cliente; este envío se realizará sólo con oprimir el botón correspondiente en la interfaz de registro de pedido. La asignación del repartidor requiere la parametrización de un algoritmo de optimización de rutas.

#### **4.2.2. Nuevo Organigrama**

La aparición de un sistema informático puede dar lugar a cambios grandes o pequeños sobre la estructura de la organización. Algunos cargos se pueden suprimir o fusionar y pueden aparecer otros cargos como un efecto directo de la solución que se plantea a los problemas de la organización. Cualquier cambio de esta índole que se genere deberá ser reportado con un nuevo organigrama, que muestre de forma clara cómo cambiará la organización estructuralmente cuando entre en funcionamiento el software.

En el caso de RAPIZZA LTDA., la solución descrita en el numeral anterior no representa cambios organizacionales y, por ende, el organigrama de la Figura 2.1. aún continúa teniendo validez. Si se hubiese propuesto una solución mucho más automática, como la elaboración de pedidos por parte del cliente directamente en una página Web o en el conmutador de la pizzería con una serie de opciones numéricas del teclado, seguramente sí se hubiera alterado el organigrama.

### **4.2.3. Cambios en los Actores**

En esta sección se deben describir los actores nuevos (si los hay) y las nuevas responsabilidades de viejos actores en el área del problema, verbalizando sus actividades principales. Esto se debe hacer sólo si es aplicable al caso.

En el caso de RAPIZZA LTDA. el único cambio apreciable será el apoyo que le dará el sistema informático a la realización de las siguientes funciones correspondientes al despachador: registro de personas, registro de pedidos, registro de detalles, registro de pagos y elaboración de cuentas de cobro. Además, para el despachador surgen dos nuevas funciones, también apoyadas en la solución informática: envío de catálogos por e-mail y cálculo de zonas para la determinación de la aceptación de un pedido.

### **4.2.4. Nuevo Diagrama de Procesos**

La aparición de la solución informática engendra una nueva forma de proceder al interior de la organización. Pueden surgir nuevos procesos o ser modificados los existentes; pueden aparecer o desaparecer roles y las diferentes funciones y responsabilidades al interior de la organización serán apoyadas por la solución informática.

En relación con el Diagrama de Procesos, además de los cambios en roles y procesos, aparece la notación empleada para representar los procesos, decisiones y almacenes que son apoyados por la solución informática; en este caso, las formas siguen siendo iguales al caso del Diagrama de Procesos anterior a la aparición de la solución informática, sólo que su reborde será mucho más grueso cuando se trata de estos elementos representados en la solución. Los almacenes que toman esta característica tienden a convertirse en almacenes electrónicos, en lugar de los almacenes físicos de documentos.

En la Figura 4.1. se puede apreciar el Diagrama de Procesos correspondiente a la Solución Informática: un Sistema para el manejo de los pedidos de RAPIZZA Ltda.

### **4.2.5. Casos de Uso**

Un Caso de Uso es una simplificación del diagrama de procesos que se centra en señalar las interacciones de los actores con el sistema para indicar el papel que éste (el sistema) cumplirá en la nueva forma de proceder. En ese sentido, un caso de uso es una secuencia de interacciones (o procesos que requieren interacción con el sistema) y que tiene un propósito específico. Esto es lo mismo que un diagrama de procesos donde no se muestran los procesos en los que no hay interacción con el sistema (que, para el caso de la Figura 4.1., tienen el reborde normal). Se debe, en consecuencia, juntar en un caso de uso las interacciones asociadas a un propósito específico, tratando de separar cada propósito en un

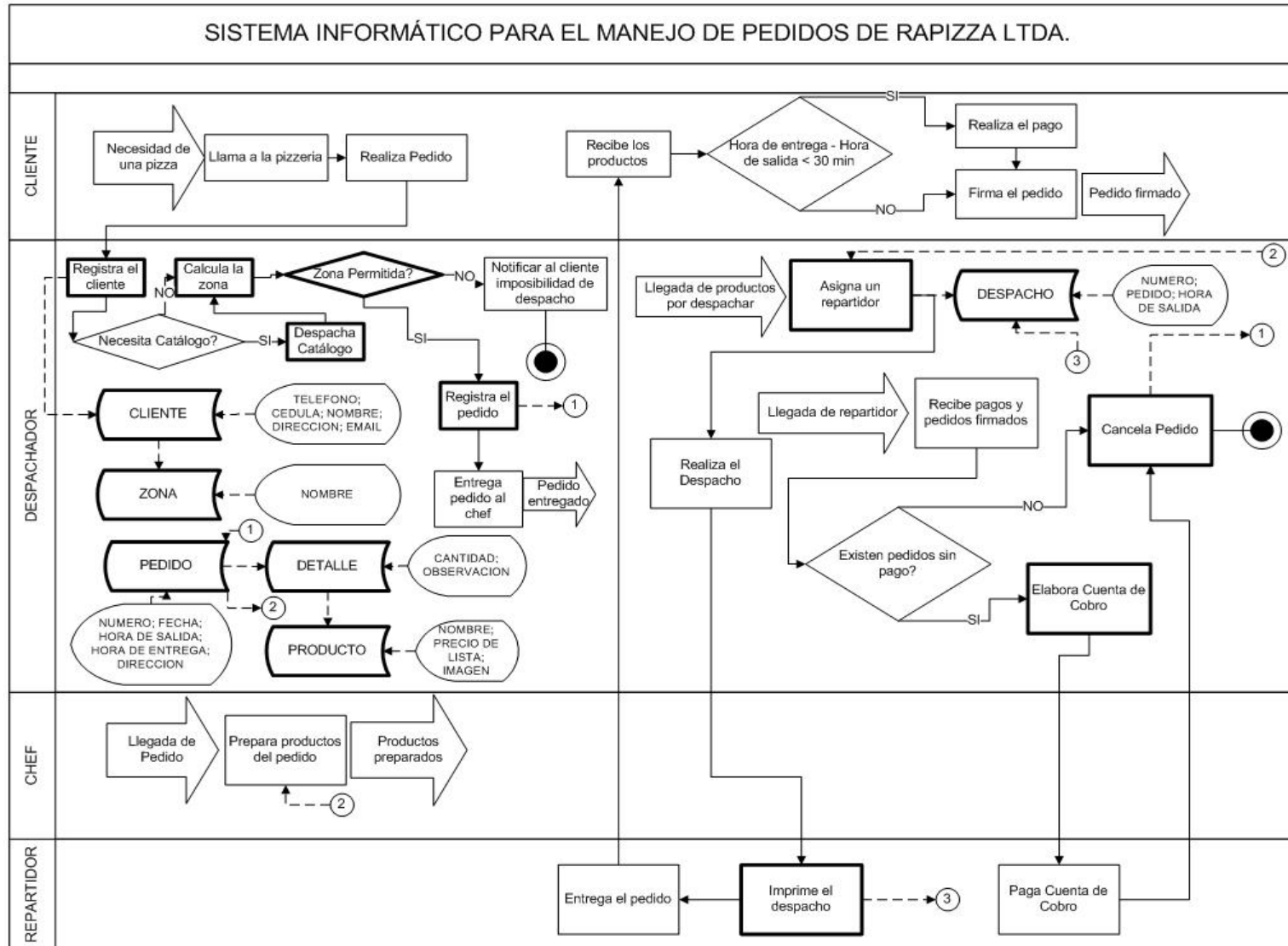


Figura 4.1. Diagrama de Procesos para el sistema informático para el manejo de pedidos de RAPIZZA LTDA.

caso de uso diferente. Esto puede conducir, por ejemplo en un sistema de consulta, a casos de uso con una sola interacción básica.

Los diagramas de casos de uso son diagramas de UML para modelar aspectos funcionales del sistema. Estos diagramas permiten reconocer claramente los límites del sistema y las relaciones del sistema con su entorno [OMG, 2006]. Para una mejor descripción consultar a Fowler [Fowler, 2004] o el estándar de UML2.0 presentado por el OMG [OMG, 2006]. En la Figura 4.2. se muestra la representación gráfica de un caso de uso; en ella, se observa un icono que simboliza el rol del actor que se relaciona con el caso de uso. En cuanto a los casos de uso, se simbolizan a través de óvalos en los que se inscribe el nombre del caso de uso y se muestran las respectivas flechas que muestran las relaciones entre los casos de uso.

Los diagramas de casos de uso comúnmente contienen: los actores, que representan los roles que un usuario puede desempeñar en el sistema; y los casos de uso que son operaciones o tareas específicas que se realizan tras una orden de algún agente externo, sea desde la petición de algún actor o desde la invocación de otro caso de uso. Entre los casos de uso se pueden establecer relaciones de inclusión (indica que el caso de uso base u origen incluye el comportamiento descrito por el caso de uso destino) como en la Figura 4.3, y extensión (muestra que el caso de uso origen extiende o amplía el comportamiento del caso de uso destino) como en la Figura 4.4.



Figura 4.2. Representación gráfica de un caso de uso



Figura 4.3. Representación de inclusión entre dos casos de uso



Figura 4.4. Representación de extensión entre dos casos de uso

Para describir los casos de uso y cada una de sus interacciones, en el contexto de UN-MÉTODO se usan las dos plantillas que se ejemplifican en las Tablas 4.1. y 4.2.

<b>Caso de Uso</b>	<Nombre del caso de uso>.		
<b>Versión</b>	<Número correspondiente>	<b>Fecha</b>	<de elaboración>
<b>Autor</b>	<Quién lo está realizando>		
<b>Fuente</b>	<Persona, unidad organizacional o documento de origen de la información>		
<b>Propósito</b>	<Qué?: Resultados del caso de uso>		
<b>Objetivo</b>	<Para qué?: Propósito del caso de uso. En particular si alivia los problemas detectados, listar que causas suprime>		
<b>Resumen</b>	<Cómo?: Resumen de la manera como se lleva a cabo>		
<b>Actores</b>	<Quiénes intervienen en la realización del caso de uso>		
<b>Precondición</b>	<Qué se requiere en principio para la activación del caso de uso>		
<b>Secuencia de interacciones</b>	<Describir en el gráfico la secuencia normal de las interacciones de los usuarios con el sistema y las posibles secuencias alternas.>		
<b>Demora</b>	<Qué tan rápido?: Tiempo que tarda en realizarse completamente un caso de uso. Indique los tiempos máximos esperados, los óptimos, y los promedios>		
<b>Frecuencia</b>	<Cuanto?: Número de veces que se lleva a cabo el caso de uso y el número de veces que se espera proceder en la forma normal y en las formas alternas>		
<b>Tipo</b>	< <b>Primario</b> si genera la información que da solución a los problemas, implicando un beneficio directo para la organización.>  < <b>Secundario</b> si es necesario para ingresar la información que los primarios necesitan y constituye un costo para la organización.>		
<b>Postcondiciones</b>	<Resultados o cambios en los datos por la aplicación del caso de uso>		
<b>Gráfico</b>	<Colocar aquí el diagrama de muñecos y óvalos>		

Tabla 4.1. Especificación Textual de un Caso de Uso.

<b>Interacción</b>	<Nombre de la interacción (nombre de la bola en el diagrama anterior) >.
<b>Actores</b>	<Quién lleva a cabo la interacción (nombre del actor en el diagrama anterior) >.
<b>Resumen</b>	<Cómo?: Resumen de la manera como se lleva a cabo>
<b>Secuencia de interacciones</b>	<Describir en detalle la interacción del usuario con el sistema señalando paso a paso lo que el usuario ingresa al sistema y los que el sistema le responde. Deben indicarse las excepciones y caminos alternos>
<b>Pantalla</b>	Colocar aquí la forma de diálogo asociada con la interacción.

Tabla 4.2. Descripción Textual de las interacciones de un Caso de Uso.



La primera describe el caso de uso como un todo, mientras que la segunda describe en detalle cada interacción. En la descripción del caso de uso como un todo se usa la primera plantilla y se coloca el dibujo de muñecos y óvalos; en la descripción de las interacciones se usa la segunda plantilla y se coloca el dibujo de la forma de diálogo. La forma de diálogo no tiene que ser la definitiva, que puede requerir un proceso de definición posterior en la fase de diseño; debe, sin embargo, identificar claramente los datos que el usuario provee u obtiene. Si una forma no está aún diseñada, esto se debe indicar en el campo correspondiente al dibujo.

Si se presenta un Caso de Uso que se describa mediante una única interacción, ambas tablas se pueden unir. Para RAPIZZA Ltda. se identifican tres casos de uso: registrar personas, registrar pedidos y registrar pagos. Por ser casos de uso muy elementales, se pueden fundir la especificación textual del caso de uso y la descripción textual de su interacción, dando por resultado las tablas 4.3. a 4.6.

<b>Caso de Uso</b>	<i>Registrar Personas</i>		
<b>Versión</b>	<i>1.0</i>	<b>Fecha</b>	<i>14/03/2006</i>
<b>Autor</b>	<i>Carlos Mario Zapata J.</i>		
<b>Fuente</b>	<i>Propietario de RAPIZZA Ltda.</i>		
<b>Propósito</b>	<i>Se realiza concretamente para disponer de clientes para los pedidos, repartidores para su asignación a los pedidos e información básica del despachador y los chef.</i>		
<b>Objetivo</b>	<i>Este caso de uso se asocia con el objetivo O17 Facilitar al Cliente el acceso al producto y contribuye en la solución general del problema SP5 Se pierde información e indirectamente de las causas C12 y C17 Se presenta exceso de llamadas y pedidos.</i>		
<b>Resumen</b>	<i>Este caso de uso permite el registro de la información básica de los clientes, despachadores, repartidores y chefs. En cualquiera de los casos la información recopilada es siempre la misma y se incluye un proceso para el cálculo de la zona de cobertura.</i>		
<b>Actores</b>	<i>Despachador</i>		
<b>Precondición</b>	<i>No debe existir el cliente, despachador, repartidor o chef que se vaya a registrar. Las zonas deben estar definidas.</i>		
<b>Secuencia normal de interacciones</b>	<b>DESPACHADOR</b>	<b>SISTEMA</b>	
<b>1</b>	<i>Selecciona la opción "Registro" del Sistema, y selecciona "Registro de Personas"</i>	<i>Presenta la interfaz que se muestra en la interacción.</i>	
<b>2</b>	<i>Da click en la lista de tipos de Persona que se pueden registrar.</i>	<i>Despliega una lista que contiene Cliente y Repartidor.</i>	

Tabla 4.3. Caso de Uso Registrar Personas (Continúa).

<b>Secuencia normal de interacciones</b>	<b>DESPACHADOR</b>	<b>SISTEMA</b>
<b>3</b>	<i>Selecciona el tipo de persona que va a registrar.</i>	
<b>4</b>	<i>Ingresa Cédula, Nombre y Teléfono de la persona y da click en la primera lista de opciones de dirección.</i>	<i>Despliega una lista que contiene CLL (calle), CRA (Carrera), TR (Transversal), CIRC (Circular) y AV (Avenida).</i>
<b>5</b>	<i>Selecciona la primera opción de dirección y anota el número correspondiente.</i>	
<b>6</b>	<i>Da click en la segunda lista de opciones de dirección.</i>	<i>Despliega una lista que contiene CLL (calle), CRA (Carrera), TR (Transversal), CIRC (Circular) y AV (Avenida).</i>
<b>7</b>	<i>Selecciona la segunda opción de dirección y anota el número correspondiente.</i>	<i>Realiza el cálculo de la zona y lo despliega automáticamente en el campo “zona”, lo cual permitirá establecer si se puede atender una llamada desde su domicilio.</i>
<b>8</b>	<i>Digita el número de la placa de la puerta y el e-mail de la persona. Luego, oprime el botón aceptar.</i>	<i>Cierra la interacción y realiza el registro de la información.</i>
<b>Secuencia alternativa de interacciones</b>	<b>DESPACHADOR</b>	<b>SISTEMA</b>
<b>2-8</b>	<i>En cualquier momento, puede presionar el botón cancelar.</i>	<i>Cierra la interacción sin hacer el registro.</i>
<b>Demora</b>	<i>Si se realiza de manera óptima, no debe tardar más de 1 minuto; en promedio será de 2 minutos y se puede esperar un máximo de 5 minutos.</i>	
<b>Frecuencia</b>	<i>Cuando el software se ponga en funcionamiento, se espera que se invoque este caso de uso unas 50 veces diarias, las cuales irán disminuyendo en la medida en que se vayan registrando los clientes.</i>	
<b>Tipo</b>	<i>Primario</i>	
<b>Postcondiciones</b>	<i>Se registra una persona con sus datos.</i>	

Tabla 4.3. Caso de Uso Registrar Personas (Continuación).

<b>Gráfico</b>	<pre> graph LR     Despachador((Despachador)) --- RegistrarPersonas((Registrar Personas))     RegistrarPersonas -.-&gt; &lt;&lt;extend&gt;&gt;  RegistrarClientes((Registrar Clientes))     RegistrarPersonas -.-&gt; &lt;&lt;extend&gt;&gt;  RegistrarRepartidores((Registrar Repartidores))     RegistrarPersonas -.-&gt; &lt;&lt;extend&gt;&gt;  RegistrarDespachadores((Registrar Despachadores))     RegistrarPersonas -.-&gt; &lt;&lt;extend&gt;&gt;  RegistrarChefs((Registrar Chefs))     RegistrarPersonas -.-&gt; &lt;&lt;include&gt;&gt;  CalcularZona((Calcular Zona))   </pre> <p>The diagram shows a central use case 'Registrar Personas' connected to an actor 'Despachador'. It has four 'extend' relationships with 'Registrar Clientes', 'Registrar Repartidores', 'Registrar Despachadores', and 'Registrar Chefs'. It also has an 'include' relationship with 'Calcular Zona'.</p>
<b>Pantalla</b>	<p>The screenshot shows a web form titled 'Registro de Personas'. It contains the following fields and controls:</p> <ul style="list-style-type: none"> <li><b>Tipo:</b> A dropdown menu with 'CLIENTE' selected.</li> <li><b>Cédula:</b> A text input field.</li> <li><b>Nombre:</b> A text input field.</li> <li><b>Teléfono:</b> A text input field.</li> <li><b>Dirección:</b> A section with two dropdown menus labeled 'CLL' and 'CRA', each followed by a text input field, and a 'No.' label with a text input field.</li> <li><b>Zona:</b> A text input field.</li> <li><b>E-mail:</b> A text input field.</li> <li><b>Buttons:</b> 'Aceptar' and 'Cancelar' buttons at the bottom.</li> </ul>

Tabla 4.3. Caso de Uso Registrar Personas (Final).

<b>Caso de Uso</b>	<i>Registrar Pedidos</i>		
<b>Versión</b>	<i>1.0</i>	<b>Fecha</b>	<i>14/03/2006</i>
<b>Autor</b>	<i>Carlos Mario Zapata J.</i>		
<b>Fuente</b>	<i>Propietario de RAPIZZA Ltda.</i>		
<b>Propósito</b>	<i>Se hace necesario que se registre y conserve la información pertinente al pedido.</i>		
<b>Objetivo</b>	<i>Este caso de uso se asocia con el objetivo O17 Facilitar al Cliente el acceso al producto, con los requisitos R1 Asegurar el registro de los pedidos, R2 Garantizar rutas de entrega organizadas y R3 Conocer las necesidades del Cliente y contribuye en la solución general del problema SP5 Se pierde información e indirectamente de las causas C6 y C11 No hay repartidores disponibles, C4 El repartidor toma demasiados pedidos, C3 Los destinos de los pedidos se encuentran muy distantes unos de otros, C5 El cliente vive o se encuentra fuera de la zona de cobertura y C13 El producto entregado no corresponde al producto solicitado por el cliente..</i>		
<b>Resumen</b>	<i>En este caso de uso se realiza el registro completo de la información correspondiente al pedido, incluyendo el cálculo de la zona a la cual se realizará el despacho y la asignación de los repartidores.</i>		
<b>Actores</b>	<i>Despachador</i>		
<b>Precondición</b>	<i>Ya se debe haber registrado el cliente y los productos disponibles en la pizzería. Además, se debe disponer del catálogo detallado de productos para un potencial envío por e-mail.</i>		
<b>Secuencia normal de interacciones</b>	<b>DESPACHADOR</b>	<b>SISTEMA</b>	
<b>1</b>	<i>Selecciona la opción “Registro” del Sistema, y selecciona “Registro de Pedidos”</i>	<i>Presenta la interfaz de “Registro de pedidos” que se muestra en la interacción, calcula el consecutivo del pedido y lo coloca en el campo correspondiente en la interfaz y coloca la fecha del sistema en el campo Fecha.</i>	
<b>2</b>	<i>Da click en la lista de Clientes disponibles.</i>	<i>Despliega una lista que contiene el teléfono, el nombre del cliente y su identificación.</i>	
<b>3</b>	<i>Selecciona el cliente.</i>	<i>Muestra de manera automática el nombre y el teléfono del cliente en los campos correspondientes.</i>	
<b>4</b>	<i>Da click en la primera lista de opciones de dirección.</i>	<i>Despliega una lista que contiene CLL (calle), CRA (Carrera), TR (Transversal), CIRC (Circular) y AV (Avenida).</i>	

Tabla 4.4. Caso de Uso Registrar Pedidos (Continúa).

<b>Secuencia normal de interacciones</b>	<b>DESPACHADOR</b>	<b>SISTEMA</b>
<b>5</b>	<i>Selecciona la primera opción de dirección y anota el número correspondiente.</i>	
<b>6</b>	<i>Da click en la segunda lista de opciones de dirección.</i>	<i>Despliega una lista que contiene CLL (calle), CRA (Carrera), TR (Transversal), CIRC (Circular) y AV (Avenida).</i>
<b>7</b>	<i>Selecciona la segunda opción de dirección y anota el número correspondiente.</i>	<i>Realiza el cálculo de la zona y lo despliega automáticamente en el campo “zona”, lo cual permitirá establecer si se puede atender una llamada desde su domicilio.</i>
<b>8</b>	<i>Digita el número de la placa de la puerta.</i>	
<b>9</b>	<i>Da click en “Agregar detalle pedido”</i>	<i>Presenta la Interfaz “Agregar detalle pedido” que se muestra en la interacción, presentando el número del pedido en el campo “Pedido”.</i>
<b>10</b>	<i>Da click en la lista de opciones de producto.</i>	<i>Despliega una lista que contiene todos los productos registrados en el sistema.</i>
<b>11</b>	<i>Selecciona el producto.</i>	<i>Llena automáticamente los campos “Precio de Lista” e “Imagen” en el producto seleccionado.</i>
<b>12</b>	<i>Ingresa la cantidad y la observación y da click en “Aceptar”.</i>	<i>Presenta nuevamente la interfaz “Registro de Pedidos” trayendo la información de producto, cantidad y observación en la lista inferior.</i>
<b>13</b>	<i>Da click en “Asignar Repartidor”</i>	<i>Calcula automáticamente las rutas de los repartidores y realiza la asignación, desplegando el nombre del repartidor en el campo correspondiente.</i>
<b>14</b>	<i>Da click en el botón “Aceptar”.</i>	<i>Cierra la interacción y realiza el registro de la información. Ingresa, además, la hora de salida del pedido (la hora en que da click en Aceptar) y despliega un cuadro de diálogo con el valor del pedido.</i>

Tabla 4.4. Caso de Uso Registrar Pedidos (Continuación).

Secuencia alternativa de interacciones	DESPACHADOR	SISTEMA
4-8 o 13	Da click en “Enviar Catálogo Digital”.	Busca el e-mail del cliente y envía un correo digital con el catálogo digital desarrollado para tal fin.
7		Se despliega un cuadro de diálogo con el mensaje “Esa zona no está dentro de la cobertura de Rapizza Ltda”.
7	El Despachador da click en “Aceptar” y cambia a una nueva dirección realizando nuevamente los pasos 4-8.	
13	Repite los pasos 9-12 tantas veces como productos tenga la orden.	
2-14	En cualquier momento, puede presionar el botón cancelar.	Cierra la interacción sin hacer el registro.
Demora	Si se realiza de manera óptima, no debe tardar más de 3 minutos; en promedio será de 5 minutos y se puede esperar un máximo de 10 minutos.	
Frecuencia	Cada vez que se realice un pedido telefónico. Se espera un promedio de 100 pedidos diarios.	
Tipo	Primario	
Postcondiciones	Se registra el pedido con todos los detalles de pedido.	
Gráfico	<pre> graph LR     Actor(Despachador) --- UC1((Registrar Pedidos))     UC1 -.-&gt; &lt;&lt;include&gt;&gt;  UC2((Agregar Detalle Pedido))     UC1 -.-&gt; &lt;&lt;include&gt;&gt;  UC3((Asignar Repartidor))     UC1 -.-&gt; &lt;&lt;include&gt;&gt;  UC4((Calcular Zona))     UC3 -.-&gt; &lt;&lt;extend&gt;&gt;  UC5((Enviar Catálogo Digital)) </pre> <p>The diagram shows a stick figure actor labeled 'Despachador' connected to a central use case 'Registrar Pedidos'. From 'Registrar Pedidos', three dashed arrows labeled '&lt;&lt;include&gt;&gt;' point to 'Agregar Detalle Pedido', 'Asignar Repartidor', and 'Calcular Zona'. A fourth dashed arrow labeled '&lt;&lt;extend&gt;&gt;' points from 'Asignar Repartidor' to 'Enviar Catálogo Digital'.</p>	

Tabla 4.4. Caso de Uso Registrar Pedidos (Continuación).

<b>Pantalla</b>	
-----------------	--

Tabla 4.4. Caso de Uso Registrar Pedidos (Continuación).

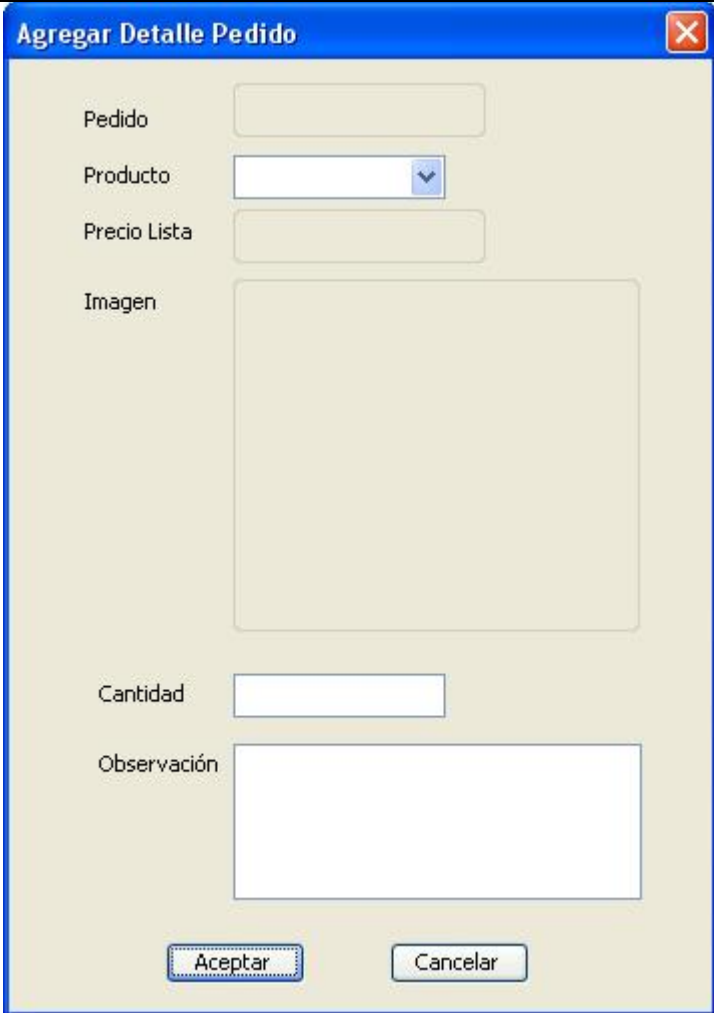
<b>Pantalla</b>	
-----------------	---

Tabla 4.4. Caso de Uso Registrar Pedidos (Final).

<b>Caso de Uso</b>	<i>Registrar Pagos</i>		
<b>Versión</b>	1.0	<b>Fecha</b>	14/03/2006
<b>Autor</b>	<i>Carlos Mario Zapata J.</i>		
<b>Fuente</b>	<i>Propietario de RAPIZZA Ltda..</i>		
<b>Propósito</b>	<i>Es necesario para conservar la información de los pagos que realizan los clientes y las cuentas de cobro con destino a los repartidores.</i>		
<b>Objetivo</b>	<i>Este caso de uso se asocia con el objetivo O3 Garantizar la recuperación del dinero de las ventas, el O7 Cumplir con las políticas de cobro a repartidores, con el requisito R1 Asegurar registro de los pedidos y contribuye en la solución general del problema SP5 Se pierde información.</i>		

Tabla 4.5. Caso de Uso Registrar Pagos (Continúa).



<b>Resumen</b>	<i>En este caso de uso se registran los pagos y se generan las cuentas de cobro a los repartidores, que sustituyen el pago de los clientes.</i>	
<b>Actores</b>	<i>Despachador</i>	
<b>Precondición</b>	<i>Debe existir el pedido que se va a cancelar, asignado a un repartidor.</i>	
<b>Secuencia normal de interacciones</b>	<b>DESPACHADOR</b>	<b>SISTEMA</b>
<b>1</b>	<i>Selecciona la opción “Registro” del Sistema, y selecciona “Registro de Pagos”</i>	<i>Presenta la interfaz de “Registro de Pago” que se muestra en la interacción.</i>
<b>2</b>	<i>Da click en la lista de Pedidos disponibles.</i>	<i>Despliega una lista que contiene los pedidos que no han sido cancelados o que no tienen cuenta de cobro.</i>
<b>3</b>	<i>Selecciona uno de los pedidos de la lista.</i>	
<b>4</b>	<i>Da click en “Cancelar Pedido”.</i>	<i>Cambia el estado del pedido a “Cancelado” y lo saca de la lista disponible de pedidos para pago.</i>
<b>5</b>	<i>Da click en el botón “Aceptar”.</i>	<i>Cierra la interacción y realiza el registro de la información.</i>
<b>Secuencia alternativa de interacciones</b>	<b>DESPACHADOR</b>	<b>SISTEMA</b>
<b>3</b>	<i>Da click en “Elaborar Cuenta de Cobro”.</i>	<i>Presenta la interfaz de “Elaboración Cuentas de Cobro” que se muestra en la interacción. Coloca automáticamente el número del pedido en el campo correspondiente; además, ubica por defecto la lista de repartidores en el repartidor que hizo la entrega.</i>
<b>3</b>	<i>Ingresa un número de la cuenta de cobro y da click en el botón “Aceptar”</i>	<i>Cierra la interacción, ingresa la información y regresa a la interacción anterior.</i>
<b>2-5</b>	<i>En cualquier momento, puede presionar el botón cancelar.</i>	<i>Cierra la interacción sin hacer el registro.</i>
<b>Demora</b>	<i>Si se realiza de manera óptima, no debe tardar más de 1 minuto; en promedio será de 2 minutos y se puede esperar un máximo de 5 minutos.</i>	
<b>Frecuencia</b>	<i>Cada vez que llegue un repartidor con los pedidos firmados y el dinero correspondiente. Se calcula unas 20 veces diarias.</i>	
<b>Tipo</b>	<i>Primario</i>	

Tabla 4.5. Caso de Uso Registrar Pagos (Continuación).

<b>Postcondiciones</b>	<i>Se cancela el pedido o se asigna una cuenta de cobro que posteriormente paga el repartidor.</i>
<b>Gráfico</b>	<pre> graph LR     Actor[Despachador] --- UC1((Registrar Pagos))     UC2((Cancelar Pedido)) -.-&gt; &lt;&lt;extend&gt;&gt;  UC1     UC3((Elaborar Cuenta de Cobro)) -.-&gt; &lt;&lt;extend&gt;&gt;  UC1         </pre> <p>The diagram shows a stick figure actor labeled 'Despachador' connected to a use case 'Registrar Pagos'. Two other use cases, 'Cancelar Pedido' and 'Elaborar Cuenta de Cobro', are connected to 'Registrar Pagos' with dashed arrows labeled '&lt;&lt;extend&gt;&gt;'.</p>
<b>Pantalla</b>	<p>The 'Pantalla' section contains two screenshots of software windows. The top window, titled 'Registro de Pago', has a dropdown menu for 'Pedido', buttons for 'Cancelar Pedido', 'Elaborar Cuenta de Cobro', 'Aceptar', and 'Cancelar'. The bottom window, titled 'Elaboración Cuentas de Cobro', has input fields for 'Pedido', 'Repartidor' (with a dropdown arrow), and 'Cuenta de Cobro', along with 'Aceptar' and 'Cancelar' buttons.</p>

Tabla 4.5. Caso de Uso Registrar Pagos (Final).

Caso de Uso	Imprimir Despacho		
Versión	1.0	Fecha	14/03/2006
Autor	Carlos Mario Zapata J.		
Fuente	Grupo de Desarrollo		
Propósito	Los repartidores deben enterarse de la distribución de los pedidos por despacho, con el fin de planear su ruta de entrega.		
Objetivo	Este caso de uso se asocia con el objetivo O13 Asegurar la entrega a tiempo, con el requisito R2 Garantizar rutas de entrega organizadas y con la expectativa E7 Garantizar el envío del producto al cliente y contribuye en la solución de la causa C4 El repartidor toma demasiados pedidos.		
Resumen	En este caso de uso los repartidores imprimen los despachos que se han conformado con la asignación de los pedidos a los repartidores..		
Actores	Repartidor		
Precondición	Debe existir el despacho que se va a imprimir.		
Secuencia normal de interacciones	REPARTIDOR	SISTEMA	
1	Selecciona la opción “Impresión” del Sistema, y selecciona “Impresión de despachos”	Presenta la interfaz de “Impresión de despachos” que se muestra en la interacción y despliega una lista que contiene las cédulas de los repartidores.	
2	Selecciona la cédula del repartidor.	Presenta una lista de los números de despacho asignados al repartidor, comenzando por el último.	
3	Selecciona uno de los despachos de la lista.	El sistema presenta de forma preliminar un informe con la información de los pedidos incluidos en el despacho.	
4	Da click en “Imprimir”.	Imprime el despacho y cierra la interacción	
Secuencia alternativa de interacciones	DESPACHADOR	SISTEMA	
2-4	En cualquier momento, puede presionar el botón cancelar.	Cierra la interacción sin realizar la impresión.	
Demora	Si se realiza de manera óptima, no debe tardar más de 1 minuto; en promedio será de 2 minutos y se puede esperar un máximo de 5 minutos.		
Frecuencia	Cada vez que llegue un repartidor a la pizzería después de realizar las entregas de los pedidos de un despacho..		
Tipo	Secundario.		

Tabla 4.6. Caso de Uso Imprimir Despachos (Continúa).

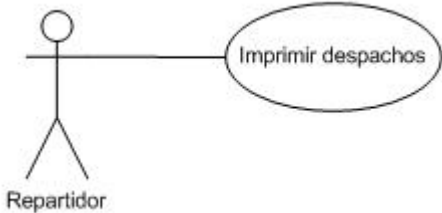

<b>Postcondiciones</b>	<i>Se imprime un despacho.</i>
<b>Gráfico</b>	 <pre> graph LR     Repartidor((Repartidor)) --- ImprimirDespachos([Imprimir despachos]) </pre>
<b>Pantalla</b>	

Tabla 4.6. Caso de Uso Imprimir Despachos (Final).

#### 4.2.6. Carta de Navegación de Interfaces

Las interfaces gráficas de usuario, que se muestran en las interacciones de los casos de uso, tienen una relación estrecha y colaboran para realizar las transacciones que se definen en los casos de uso. En esta sección se grafican esas relaciones, que por lo general tienen la forma de diagrama jerárquico, en un esquema denominado Carta de Navegación de Interfaces. En ese diagrama se incluyen no sólo las interfaces gráficas de usuario (las que tienen pantalla en el

software), sino también aquellas funciones que son invocadas en desarrollo del caso de uso para una determinada interacción.

En la Figura 4.5 se detalla la Carta de Navegación de Interfaces.



Figura 4.5. Carta de Navegación de Interfaces para el Sistema Informático para el Manejo de Pedidos de RAPIZZA Ltda.

#### 4.2.7. Valoración de la Propuesta de Solución

El software surge como una respuesta a las necesidades del interesado, que se suelen traducir en soluciones a las causas de los diferentes problemas de la organización; esas soluciones deben tomar en consideración los objetivos de la organización y los requisitos que manifiesten los diferentes interesados en relación con el software. Por ello, la pieza de software tendrá un impacto sobre la organización que se puede calcular desde el punto de vista de las causas de los problemas que suprime total o parcialmente.

En la sección 4.1. se había afirmado que las soluciones informáticas a un problema son infinitas y de esas soluciones se debían seleccionar algunas para presentar a los interesados. Para que la valoración cumpla el efecto requerido,

debería realizarse para las diferentes soluciones que se plantearon; esto permite realizar una comparación para determinar cuáles soluciones pueden ser más completas desde el punto de vista de los problemas que soluciona a la organización misma.

En [ZAPATA Y ARANGO, 2004] se plantea un método para la valoración de la pieza de software desde el punto de vista de las causas que se atienden mediante los casos de uso de la solución informática. Para ello, utilizan como insumo un diagrama Causa-Efecto con pesos ponderados para cada una de las causas y subproblemas del dominio, según se puede apreciar en la Figura 4.6. En esa forma de asignación de pesos porcentuales a las causas se debe cumplir lo siguiente:

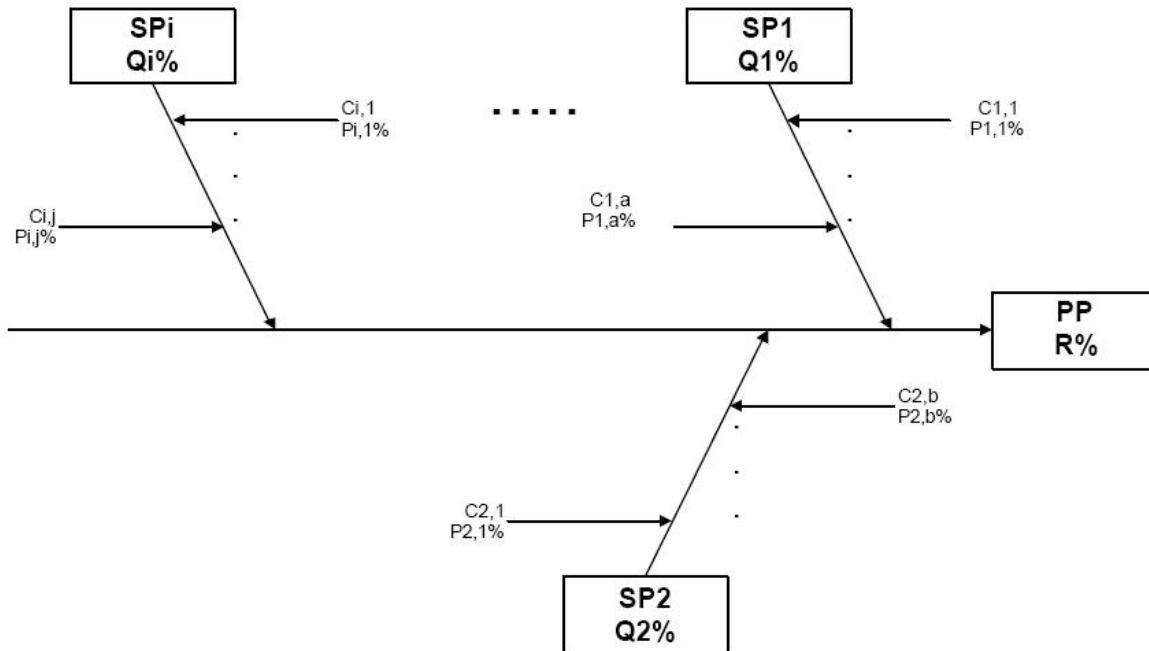


Figura 4.6. Asignación de porcentajes de importancia en el diagrama Causa-Efecto.

$$\sum_{i=1}^n Q_i * (\sum_{j=1}^m P_{i,j}) = 100\% \quad \text{y} \quad \forall i \sum_{j=1}^m P_{i,j} = 100\%$$

En las ecuaciones anteriores  $Q_i$  es el porcentaje de relevancia del problema  $SP_i$  sobre el problema principal,  $P_{i,j}$  es el porcentaje de relevancia de la causa  $C_{i,j}$  en el problema  $SP_i$ .

El método de valoración consiste en definir un porcentaje de cumplimiento  $A_{i,j}$  de una causa determinada por parte de una función de un caso de uso. En este caso,

si  $\tilde{n}$  es el conjunto de funciones asociadas con un caso de uso y ACU el porcentaje de atención del caso de uso sobre el problema, se debe cumplir que:

$$\%ACU = \sum_{\kappa=1}^{\tilde{n}} \left( \sum_{i=1}^n Q_i * \left( \sum_{j=1}^m P_{ij} * A_{ij} \right) \right) \quad y \quad \forall i,j \sum_{\kappa=1}^{\tilde{n}} A_{ij} \leq 100\%$$

En el caso de RAPIZZA LTDA. la asignación de porcentajes es la que se muestra en la Figura 4.7. La aplicación del método de valoración supone la elaboración de una tabla que incluya las funciones de los casos de uso listando los porcentajes de cumplimiento y ponderando sobre los porcentajes del diagrama Causa – Efecto. La Tabla 4.7. entrega el resultado de atención de los problemas del área mediante los casos de uso definidos para la solución.

Caso de Uso	Causa	Justificación	P <sub>i,j</sub>	Q <sub>i,j</sub>	A <sub>i,j</sub>	ACU
Registrar Personas	SP5	En especial el registro de los clientes contribuirá a evitar la pérdida de su información.	100%	10%	10%	0.01
Registrar Personas	C17	Se atenderán más rápido los clientes en la medida en que ya se tengan registros previos de muchos de ellos.	48%	15%	100%	0.072
Registrar Personas	C12	Se atenderán más rápido los clientes en la medida en que ya se tengan registros previos de muchos de ellos.	30%	25%	100%	0.075
Registrar Pedidos	SP5	El registro de los pedidos en el nuevo sistema evitará las pérdidas de la información en las diferentes etapas de la venta.	100%	10%	60%	0.06
Registrar Pedidos	C13	El catálogo digital que se enviará al cliente solucionará muchos de los problemas de desconocimiento de los productos.	100%	20%	100%	0.20
Registrar Pedidos	C11	La asignación automática de repartidores deberá lograr que las rutas sean más balanceadas y organizadas.	25%	25%	100%	0.0625
Registrar Pedidos	C6	La asignación automática de repartidores deberá lograr que las rutas sean más balanceadas y organizadas.	30%	30%	100%	0.09
Registrar Pedidos	C5	El cálculo automático de la zona con base en la dirección de despacho evitará que se pierda esfuerzo en una zona que no se puede alcanzar en el tiempo pactado.	25%	30%	100%	0.075
Registrar Pedidos	C4	El sistema asignará automáticamente el repartidor y por ello la decisión ya no será del repartidor.	20%	30%	50%	0.03

Tabla 4.7. Porcentaje atendido del Diagrama Causa – Efecto mediante los Casos de Uso de la solución (Continúa).

Caso de Uso	Causa	Justificación	$P_{i,j}$	$Q_{i,j}$	$A_{i,j}$	ACU
Registrar Pedidos	C3	Las rutas serán tomadas en consideración por el sistema para realizar la asignación de repartidores.	15%	30%	100%	0.045
Registrar Pagos	SP5	El registro de los pedidos permitirá establecer cuáles pedidos se encuentran registrados y no fueron pagados, para poder elaborar las cuentas de cobro.	100%	10%	30%	0.03
Imprimir Pagos	C4	El repartidor se enterará de los pedidos que debe atender en un despacho por la información del sistema.	20%	30%	50%	0.03
		TOTAL				77.95%

Tabla 4.7. Porcentaje atendido del Diagrama Causa – Efecto mediante los Casos de Uso de la solución (Final).

Según el resultado de la Tabla 4.7., se logra atender casi el 78% de los problemas identificados para la organización. Los problemas restantes parecerían requerir soluciones no ligadas con la informática, pero que podrían estudiarse desde el punto de vista de la Ingeniería de Sistemas; por ejemplo, la causa C8 podría requerir un estudio para determinar si efectivamente se deberían contratar otros chef para atender la demanda existente en la pizzería. Otras causas como la C15 pueden requerir incluso de otras soluciones, tales como campañas publicitarias agresivas que generen posición de marca en los clientes, lo suficiente como para que recuerden el número telefónico de RAPIZZA Ltda.

Como se había establecido antes, sería necesario realizar la valoración sobre el conjunto de soluciones seleccionado para presentarle al interesado, con el fin de mirar su impacto sobre la organización. En el caso de RAPIZZA Ltda., sólo se ha trabajado una solución hasta el momento y esa es la que se valoró en la Tabla 4.7.

#### 4.2.8. Costeo de la Propuesta de Solución

Los interesados no tendrán suficiente con el impacto de la nueva pieza de software sobre la organización, sino que también necesitarán saber qué esfuerzo requiere el desarrollo de esa pieza de software. Entre la gama de soluciones posibles, es necesario realizar el costeo de cada solución, con el fin de definir las diferencias económicas que existen entre ellas. El costeo y la valoración se pueden emplear para seleccionar las soluciones más probables en su realización, que son aquellas de alto impacto sobre la organización y poco esfuerzo requerido para su elaboración; la decisión final correrá a cargo del interesado, pues él podrá definir qué recursos económicos posee y cuál es el nivel de impacto en los problemas de la organización que requiere solucionar.



En UN-MÉTODO la estimación del esfuerzo se realiza con Puntos de Casos de Uso (UCP – Use Case Points), un trabajo de [KARNER, 1993]. Esta medida reconoce la importancia de la funcionalidad del software en relación con el esfuerzo que supone su desarrollo. Así, emplea los casos de uso como técnica de estimación para calcular el esfuerzo en el desarrollo de software. Es de notar que ésta es una técnica empírica cuyos coeficientes han sido calculados para otras latitudes; por ello, el costo que finalmente arrojan las diferentes propuestas de solución será un estimativo y no deberá utilizarse como una cotización precisa para el cobro a los interesados. Un trabajo futuro de UN-MÉTODO podría ser la calibración de estos coeficientes para el entorno colombiano.

Los pasos que se realizan para el cálculo de los Puntos de Casos de Uso son:

1. Se determina el peso sin ajustar de los actores (UAW – Unadjusted Actor weight): este peso corresponde al factor de complejidad, el cual se selecciona con un test de litmus (una prueba en la cual un factor simple es decisivo), el cual toma en consideración las características de los actores en relación con la aplicación. En la Tabla 4.8. se puede consultar la clasificación de los actores.

Clasificación	Test de litmus para reconocer clasificaciones	Valor/Factor
Actores simples	Son aquellos que se comunican con el sistema a través de APIs.	1
Actores Promedio	Se reconocen si se rigen por las siguientes propiedades: <ul style="list-style-type: none"> <li>• Actores que están interactuando el sistema a través de algún protocolo (http, Ftp o probablemente algún protocolo definido por el interesado).</li> </ul>	2
Actores Complejos	Aquellos que interactúan normalmente a través de Interfaces Gráficas de Usuario (Graphic User Interface o GUI).	3

Tabla 4.8. Clasificación de los Actores según su tipo de interacción.

2. Se determina el peso sin ajustar de los casos de uso (UUCW – Unadjusted Use Case Weight): Se realiza un conteo de los escenarios y transacciones para determinar este factor. En la Tabla 4.9. se muestran las características que se emplean en el cálculo de este factor.
3. Se determina el total de Puntos no ajustados de Casos de Uso (UUCP – Unadjusted Use Case Point): Se emplea para ello la siguiente fórmula:

$$\text{Total UUCP} = \text{Total UAW} + \text{Total UUCW}$$

Tipo de Caso de Uso	Test de litmus para decidir la Clasificación	Valor/Factor
Simple	Mayor o igual a tres transacciones	5
Promedio	Entre 4 y 7 transacciones	10
Complejo	Mayor de 7 transacciones	15

Tabla 4.9. Clasificación de los Casos de Uso según la cantidad de transacciones y escenarios.

4. Se calcula un factor técnico y ambiental: En este paso se toma en consideración la complejidad técnica, asignando un puntaje entre 0 y 5 a diferentes factores técnicos que dependen de la complejidad, y que se consignan en la Tabla 4.10.

Cod	Factor Técnico	Peso	Descripción
t1	Sistema Distribuido	2	La arquitectura es centralizada o distribuida?
t2	Tiempo de Respuesta	1	El tiempo de respuesta es un criterio importante? El interesado necesita que el sistema corra rápido?
t3	Eficiencia del usuario final	1	Cómo es la eficiencia del usuario final?
t4	Procesamiento interno complejo	1	El proceso de negocios es muy complejo? Por ejemplo: cuentas complicadas, cierres, seguimiento de inventario, cálculo difícil de impuestos, etc.
t5	Código reutilizable	1	Se intenta mantener la reusabilidad alta? Esto incrementará la complejidad del diseño.
t6	Facilidad de instalación	0.5	El interesado está buscando una facilidad de instalación? Por defecto, se colocan muchos instaladores que crean los paquetes. Sin embargo, el interesado está buscando una instalación que probablemente dependa de módulos inteligentes. Uno de los interesados tiene como requisito una instalación personalizada. Si el requisito es tal que cuando haya una nueva versión deba auto-instalarse. Estos factores contarán cuando se asigne valor a este factor.
t7	Uso fácil	0.5	Es la amigabilidad un factor importante para el usuario?
t8	Portabilidad	2	El interesado está buscando implementación en diferentes plataformas?
t9	Fácil de cambiar	1	El interesado está buscando personalización en el futuro? Esto también incrementa la complejidad del diseño de la arquitectura y, en consecuencia, este factor.
t10	Concurrente	1	El interesado está buscando muchos usuarios simultáneos con apoyo automático? Este valor incrementa la complejidad de la arquitectura y, en consecuencia, este factor.

Tabla 4.10. Bases para el cálculo del factor de complejidad técnica (Continúa).

Cod	Factor Técnico	Peso	Descripción
t11	Objetivos de seguridad	1	El interesado quiere alta seguridad o encriptación de la información?
t12	Acceso directo a terceras partes	1	El proyecto depende del uso de controles por terceras partes? La comprensión de los controles de terceras partes y el estudio de los pros y contras requiere mucho esfuerzo. Así, este factor se debe asignar en consecuencia.
t13	Facilidades de entrenamiento de usuarios.	1	Será tan complejo el software desde la perspectiva del usuario que requiera entrenamiento? Este factor debería variar en consecuencia.

Tabla 4.10. Bases para el cálculo del factor de complejidad técnica (Final).

- Se suman los resultados de los diferentes factores para obtener el valor denominado Tfactor.
- Se calcula el factor de Complejidad Técnica (TCF – Technical Complexity Factor): Esto se realiza con la fórmula siguiente:

$$TCF = 0.6 + (0.01 * Tfactor)$$

- Se calcula el factor ambiental (Environmental Factor): El costo estimado se ve también afectado por factores del entorno, como el personal entrenado, la motivación de los programadores, etc., los cuales se listan en la Tabla 4.11.

Cod	Factor Ambiental	Peso	Descripción
e1	Familiaridad con el proyecto	1.5	Toda la gente que trabaja en el proyecto se familiarizó con el dominio y factores técnicos del proyecto? Sino, probablemente se gastará mucho tiempo en la explicación de los Know-How.
e2	Experiencia en la aplicación.	0.5	Qué tanta experiencia en la aplicación tiene el equipo de desarrollo?
e3	Experiencia orientada a objetos	1	Los documentos de los casos de uso son entradas para el diseño orientado a objetos. Es importante que el equipo humano del proyecto tenga conocimientos básicos de orientación a objetos.
e4	Capacidad del analista líder.	0.5	Cómo está conduciendo el proyecto el analista? Tiene suficiente conocimiento del dominio?
e5	Motivación	1	Están los programadores motivados de trabajar en el proyecto? La inestabilidad en el proyecto puede conducir a la gente a dejar el código fuente a medio camino. Este factor se puede hacer acorde a la realidad de la industria del software; por ejemplo, si el mercado del software es bueno se puede colocar el máximo puntaje. A mejor mercado hay más trabajos y más programadores aparecerán.

Tabla 4.11. Elementos para el cálculo del Factor de Entorno (Continúa).

Cod	Factor Ambiental	Peso	Descripción
e6	Requisitos estables	2	Tiene claridad el interesado en lo que quiere? Las expectativas del interesado son el factor más importante en la estabilidad de los requisitos. Si la naturaleza del interesado es altamente cambiante, coloque este valor al máximo.
e7	Personal de tiempo parcial	-1	Existe personal de tiempo parcial en el proyecto como consultores, etc.?
e8	Lenguaje de programación difícil	-1	Qué tan complejo es el lenguaje? Vb6, c++, c, etc.

Tabla 4.11. Elementos para el cálculo del Factor de Entorno (Final).

8. Se calcula la suma de los factores anteriores y se guarda en Efactor.
9. Se calcula el factor ambiental empleando la siguiente ecuación:

$$EF = 1.4 + (-0.03 * Efactor)$$

10. Se calculan los puntos de casos de uso ajustados (AUCP – Adjusted Use Case Points), con la siguiente ecuación:

$$AUCP = UUCP * TCF * EF$$

11. Se multiplica por el factor de Hombre/hora, para obtener el esfuerzo:

$$ESFUERZO = AUCP * \text{Hombre/hora/AUCP}$$

[KARNER, 1993] propuso un factor de 20 horas/hombre por punto de caso de uso ajustado para realizar la estimación, aunque otros autores han propuesto un rango entre 15 y 30 para ese valor.

La aplicación del método para el cálculo de los puntos de casos de uso al software de despachos de RAPIZZA Ltda. es como sigue:

1. El único actor reconocido para interactuar con el sistema es el Despachador y es un actor complejo porque sólo interactúa a través de interfaces gráficas de usuario; por lo tanto UAW = 3.
2. Se identificaron únicamente tres transacciones, dadas por los casos de uso “registrar personas”, “registrar pedidos” y “registrar pagos”. Tomando en consideración que la extensión “enviar catálogo digital” es independiente del caso de uso “registrar pedidos” y que el caso de uso “registrar pagos” realmente se divide en “cancelar pedido” y “elaborar cuenta de cobro”, el total de escenarios es de cinco. Por ello, UUCW = 10.
3.  $UUCP = UAW + UUCW = 3 + 10 = 13$ .

4. La Tabla 4.12. muestra los elementos para el cálculo del factor de complejidad técnica de RAPIZZA Ltda.

Cod	Factor Técnico	Peso	Valor	Subtotal	Explicación
t1	Sistema Distribuido	2	0	0	Se elige una arquitectura centralizada en el equipo del despachador y sin conexiones a red.
t2	Tiempo de Respuesta	1	3	3	El sistema debería correr rápido para que se puedan atender los pedidos más rápidamente. Sin embargo, no es un software de precisión.
t3	Eficiencia del usuario final	1	1	1	Se presume que el despachador es bueno en el uso de sistemas.
t4	Procesamiento interno complejo	1	1	1	El proceso no es de alta complejidad. Las transacciones son muy claras y se realizan sólo operaciones algebraicas.
t5	Código reutilizable	1	0	0	No se piensa que sea un código reutilizable.
t6	Facilidad de instalación	0.5	0	0	La instalación deberá ser supremamente simple, por lo cual no será un factor que afecte la complejidad.
t7	Uso fácil	0.5	5	2.5	El sistema deberá ser muy amigable, con el fin de que las transacciones se puedan realizar fácilmente.
t8	Portabilidad	2	0	0	No se busca implementación en otras plataformas.
t9	Fácil de cambiar	1	2	2	El sistema debería ser paramétrico, especialmente en la adición de zonas cuando la pizzería crezca.
t10	Concurrente	1	0	0	No se requiere concurrencia. Sólo existirá un usuario del sistema.
t11	Objetivos de seguridad	1	1	1	La seguridad estará dada únicamente por la contraseña de acceso.
t12	Acceso directo a terceras partes	1	0	0	El único control del sistema correrá a cargo del dueño de la pizzería. No se requieren auditorías de oficinas especializadas.
t13	Facilidades de entrenamiento de usuarios.	1	1	1	Poca complejidad de parte del software. El entrenamiento deberá ser muy sencillo.

Tabla 4.12. Elementos para el cálculo del factor de complejidad técnica.

5. De la tabla anterior,  $T_{factor} = 11.5$   
 6.  $TCF = 0.6 + (0.01 * T_{factor}) = 0.6 + 0.01 * 11.5 = 0.715$

7. La Tabla 4.13. muestra los valores de los elementos empleados en el cálculo del factor de complejidad del entorno.

Cod	Factor Ambiental	Peso	Valor	Subtotal	Descripción
e1	Familiaridad con el proyecto	1.5	5	7.5	Es un proyecto supremamente simple, en el cual la familiaridad se logra rápidamente.
e2	Experiencia en la aplicación.	0.5	5	2.5	Se cree que un sistema en Visual Basic sea suficiente, con lo cual no se requiere mucha destreza en la aplicación.
e3	Experiencia orientada a objetos	1	5	5	Se está usando UN-MÉTODO para la elaboración de la documentación, por lo cual la orientación a objetos se garantiza.
e4	Capacidad del analista líder.	0.5	5	2.5	Tampoco afecta sustancialmente, por el hecho de ser una aplicación bastante simple.
e5	Motivación	1	5	5	Sí existe suficiente motivación respecto del proyecto.
e6	Requisitos estables	2	1	2	No existe mucha claridad del propietario de RAPIZZA Ltda. en relación con las necesidades que tiene.
e7	Personal de tiempo parcial	-1	0	0	No existe personal de tiempo parcial.
e8	Lenguaje de programación difícil	-1	0	0	Se seleccionó Visual Basic, que es de fácil manejo incluso para programadores inexpertos.

Tabla 4.13. Elementos para el cálculo de Complejidad del Entorno.

8. Efactor = 24.5

9.  $EF = 1.4 + (-0.03 * Efactor) = 1.4 + (-0.03 * 24.5) = 0.665$

10.  $AUCP = UUCP * TCF * EF = 13 * 0.715 * 0.665 = 6.18$

11.  $ESFUERZO = 6.18 * 20 = 123.6$  Horas/hombre

En nuestro medio, el valor promedio de Horas/hombre es \$35.000, por lo cual se presume que el costo total estará cercano a \$ 4'326.000.

#### 4.2.9. Factores Críticos de Éxito de la Propuesta de Solución

El software raras veces se origina en un sentir colectivo de una organización, sino que más bien se trata de iniciativas particulares que detectan algún problema específico dentro de la organización y tratan de resolverlo. Por ello, a lo largo del proceso de desarrollo de una pieza de software, se pueden presentar inconvenientes que podrían eventualmente hacer fracasar el proyecto, en especial

actitudes negativas de algunas personas de la organización, resistencia al cambio, falta de entrenamiento en nuevas tecnologías, etc.

Además, por sus características especiales, el software suele acarrear riesgos de tipo tecnológico, que también eventualmente podrían afectar el desarrollo del proyecto.

Esos aspectos, tanto humanos como tecnológicos que podrían impedir el éxito futuro del sistema o su desarrollo se suelen denominar Factores Críticos de Éxito y se deben identificar en la propuesta de solución que más factiblemente se implemente. Con ello se busca alertar al grupo de interesados para que tome acciones tendientes a minimizar el impacto de esos factores dentro de la organización, posibilitando una suave transición de la organización hacia la nueva pieza de software.

Para el caso de RAPIZZA Ltda., los factores críticos de éxito que se pueden identificar son:

- Resistencia al cambio por parte de los repartidores: si los pedidos se registran de manera electrónica y se puede consultar de manera segura su contenido, muchas de las excusas de los repartidores para evitar el pago de las cuentas de cobro perderían validez.
- Resistencia al cambio por parte del despachador: El paso del registro en papel al registro electrónico puede causar inconvenientes en las personas. En efecto, se puede controlar mejor la información que se encuentra en formato digital, simplemente por el hecho de que se puede calcular información consolidada mucho más rápidamente que sumando cantidades en papel. El despachador de RAPIZZA Ltda. puede tener inconvenientes con este tipo de registros porque puede sentir un mayor control sobre su gestión.
- Carencia de recursos económicos para la compra de licencias de la plataforma de desarrollo. En general, este puede ser un factor crítico de éxito para organizaciones pequeñas, que por lo general no tienen un plan de inversiones en tecnología, como puede ser el caso de RAPIZZA Ltda.
- Carencia de recursos económicos para la realización del estudio que conduzca a disponer de un catálogo digital adecuado.

#### **4.3. ASPECTOS DE CONSISTENCIA Y REFINAMIENTO EN UN-MÉTODO**

El análisis del problema entrega como resultado una serie de propuestas de solución, en las cuales la terminología debe seguir siendo la misma (lo cual implica los aspectos de consistencia con el Contexto del Software), pero en las cuales pueden existir diferencias apreciables en la forma de proceder. Además, se

introducen nuevos elementos que deben guardar consistencia también con lo que se ha manejado hasta ahora del contexto y del problema.

Las reglas de consistencia definidas para las Propuestas de Solución son las siguientes:

1. El diagrama causa – efecto con el cual se valoran las propuestas de solución debe ser el mismo que se identificó en el Análisis del Problema, únicamente con la adición de los porcentajes de relevancia de las causas y subproblemas. En relación con estos porcentajes, en la actualidad UN-MÉTODO no posee un método estándar de asignación de los mismos, por lo cual se debe recurrir a la experiencia de los interesados para poder realizar la ponderación de la importancia de las causas sobre el problema principal. Si en las propuestas de solución se identifican nuevas causas o subproblemas, las modificaciones pertinentes se deberían hacer sobre el diagrama causa-efecto en los dos entregables.
2. En el diagrama de procesos de la solución, los procesos con reborde grueso (aquellos que describen la funcionalidad de la nueva aplicación informática) deberán aparecer representadas en los casos de uso y el esquema de navegación de interfaces. Además, los almacenes con reborde grueso y sus características almacenables deberían estar representadas en las interacciones que acompañan los casos de uso (las interfaces gráficas de usuario).
3. Los casos de uso, incluyendo sus relaciones <<extend>> e <<include>>, deberían estar representados en las interfaces gráficas de usuario. Cabe anotar que los botones que llevan a funciones adicionales, las listas desplegables y los cálculos automáticos al interior de las interfaces son candidatos a relaciones especiales, que se deberían chequear en relación con el diagrama.
4. En las tablas de descripción de los casos de uso, los objetivos se relacionan con algunos de los siguientes elementos:
  - a. Objetivos, expectativas y requisitos del diagrama de objetivos.
  - b. Subproblemas y causas del diagrama causa – efecto.
5. Los actores listados en las tablas de descripción de los casos de uso deben corresponder a roles incluidos en el diagrama de procesos de la solución.
6. Las secuencias normales y alternativas de las tablas de descripción de los casos de uso se deben referir a elementos existentes en las interfaces gráficas de usuario que se asocian con ellas.
7. Las funciones con o sin interfaz que se describen mediante los casos de uso deben estar referenciadas en la carta de navegación de interfaces.
8. En la valoración de las Propuestas de Solución, la sumatoria de los porcentajes de cumplimiento para una causa particular a través de los diferentes casos de uso no podrá ser superior al 100%. Además, las causas y subproblemas que se incluyen en la tabla de valoración deben estar definidas en el diagrama causa – efecto y los porcentajes de relevancia



correspondientes deberán ser los mismos que se incluyeron en el diagrama causa-efecto.

#### **4.4. UN USO ADICIONAL DEL DIAGRAMA DE PROCESOS EN EL MARCO DE UN-MÉTODO**

El grupo de investigación UN-INFO, paralelamente con el desarrollo de UN-MÉTODO, viene realizando otros trabajos que emplean especialmente herramientas de soporte a UN-MÉTODO. Uno de esos trabajos, denominado UNC-PROTOTIPADOR, emplea el diagrama de procesos de la solución para elaborar de manera automática un prototipo compilable y ejecutable en lenguaje JAVA®. De esta manera, se podría establecer una conexión entre el analista y el interesado que permitiera muy rápidamente la validación de las diferentes interfaces gráficas de usuario, buscando su completitud. Por ser una técnica de desarrollo automático de prototipos, no se espera que la aplicación resultante de UNC-PROTOTIPADOR tenga mucha cercanía con la solución definitiva, a pesar de que el código generado podría ser modificado en un editor de JAVA®, sino que se realiza teniendo en mente la validación de las interfaces por parte del interesado de manera rápida.

En la Figura 4.7. se puede apreciar el diagrama de procesos de RAPIZZA Ltda. que sirvió de base para la producción del prototipo. Nótese las diferencias con el diagrama de procesos de la solución expresado en la Figura 4.1., las cuales se deben específicamente a que el diagrama de la Figura 4.7. se podría encontrar en un nivel de abstracción más bajo o que estaría más cercano a la implementación. Finalmente, en las Figuras 4.8. a 4.14. se pueden apreciar las interfaces gráficas de usuario generadas por UN-PROTOTIPADOR para el diagrama de procesos de la Figura 4.7.

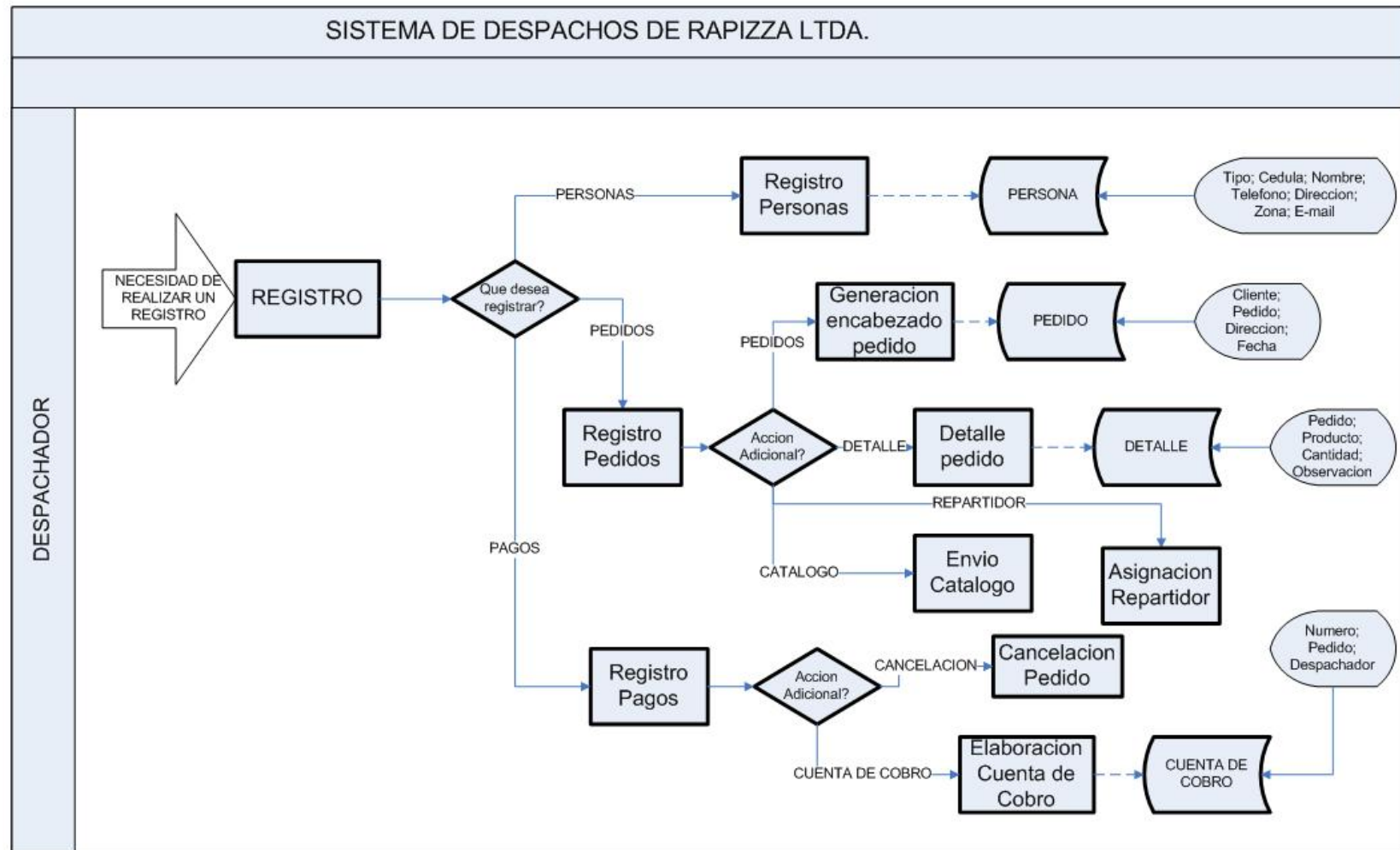


Figura 4.7. Diagrama de procesos utilizado en UNC-Diagramador para obtener automáticamente el código Java® de Rapizza Ltda.

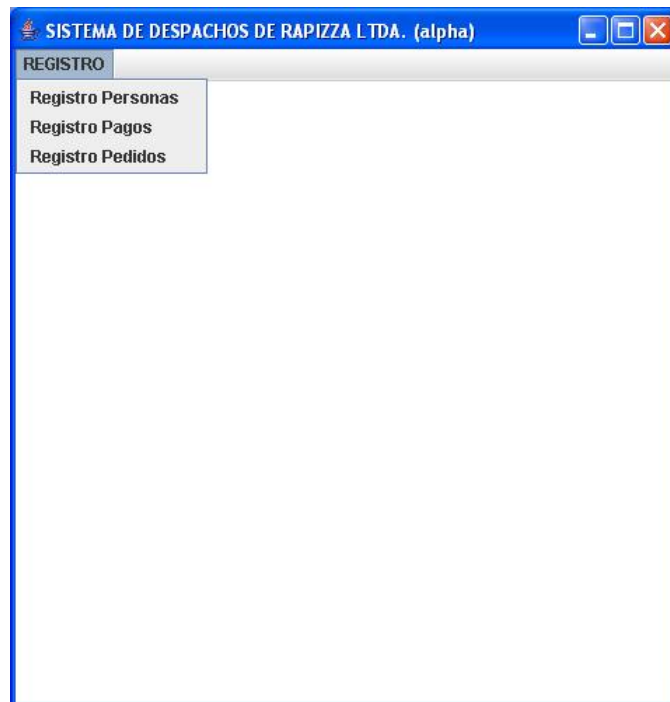


Figura 4.8. Menú Principal del Software de Rapizza Ltda. generado por UNC-Diagramador.

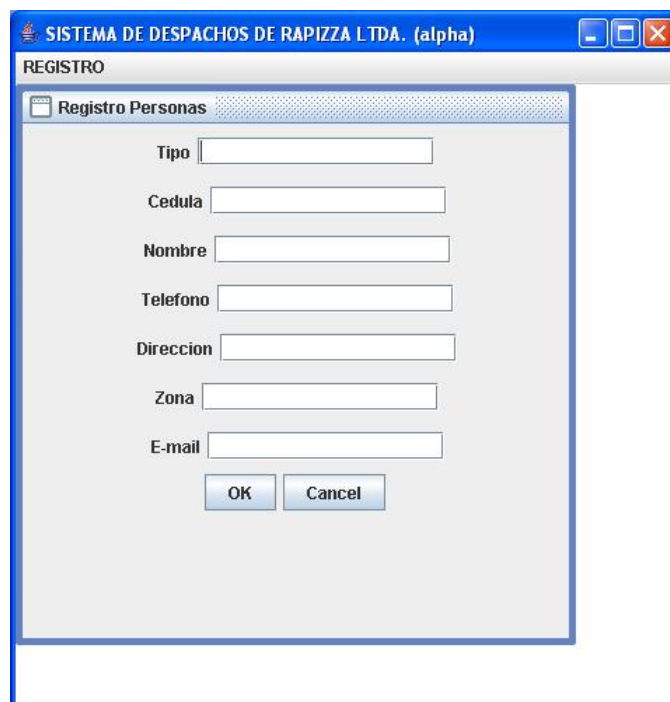


Figura 4.9. Interfaz para el Registro de personas de Rapizza Ltda. generada por UNC-Diagramador.

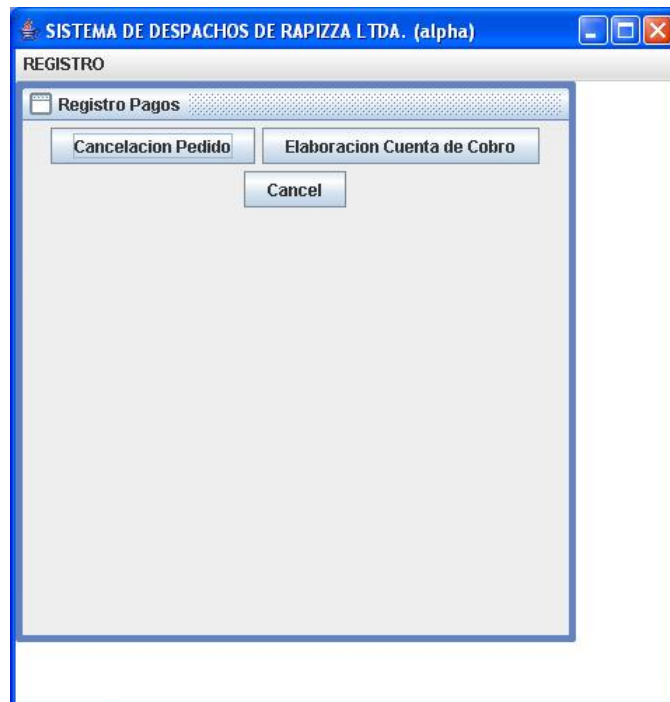


Figura 4.10. Interfaz de Registro de Pagos de Rapizza Ltda. generada por UNC-Diagramador.

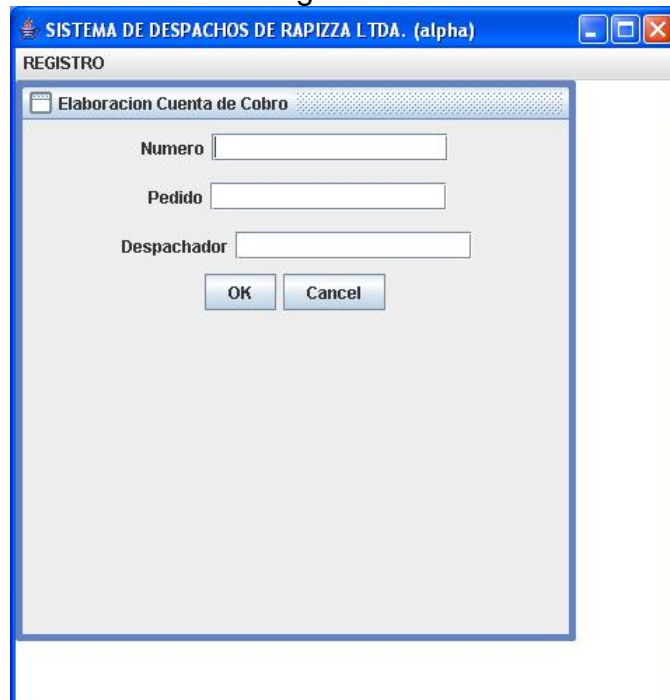


Figura 4.11. Interfaz de Elaboración de Cuentas de Cobro de Rapizza Ltda. generada por UNC-Diagramador.

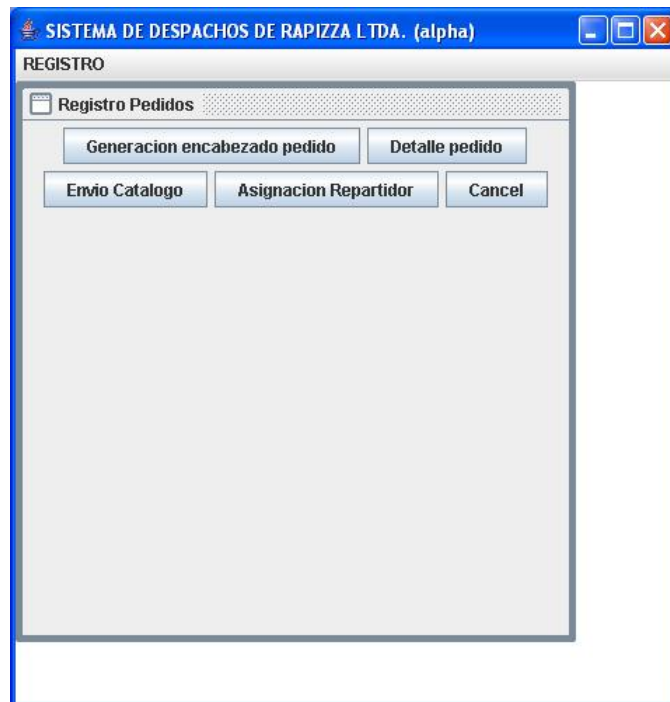


Figura 4.12. Interfaz de Registro de Pedidos de Rapizza Ltda. generada por UNC-Diagramador.

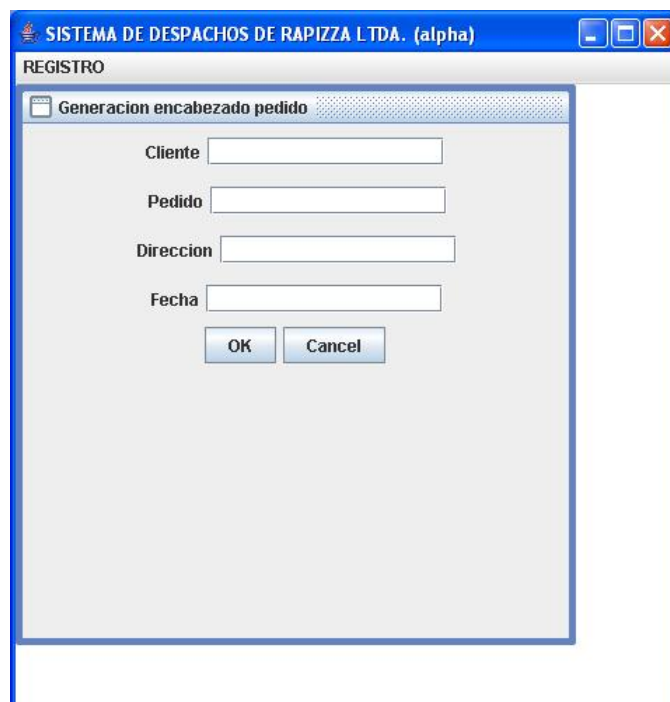


Figura 4.13. Interfaz de Generación de encabezado de pedidos de Rapizza Ltda. generada por UNC-Diagramador.

The image shows a screenshot of a software application window. The title bar at the top reads "SISTEMA DE DESPACHOS DE RAPIZZA LTDA. (alpha)". Below the title bar, the window contains a form titled "REGISTRO". Inside the "REGISTRO" form, there is a sub-section titled "Detalle pedido". This section contains four text input fields: "Pedido", "Producto", "Cantidad", and "Observacion". Below these fields are two buttons: "OK" and "Cancel". The window has a standard Windows-style border with minimize, maximize, and close buttons in the top right corner.

Figura 4.14. Interfaz para la elaboración de detalles de pedido de Rapizza Ltda. generada por UNC-Diagramador.

# Capítulo 5

## Entregable 4: Esquema Conceptual

### 5.1. INTRODUCCIÓN

En el capítulo anterior, con base en la valoración y el costeo de las diferentes piezas de software, los interesados eligen sólo una solución, que es la que seguirá el proceso de análisis y diseño detallado, del cual el Esquema Conceptual que se discute en este capítulo es el paso inicial. En esta fase los diagramas de UML [OMG, 2006], tales como clases, actividades y transición de estados juegan un papel importante en la especificación de la estructura y el comportamiento; estos diagramas, en UN-MÉTODO, se complementan con la expresividad y tratamiento formal de la lógica de predicados, con el fin de definir de la manera más completa posible la especificación de la solución, lo cual redundará en mayor acoplamiento entre los requisitos de los interesados y el código fuente de la aplicación que soluciona sus problemas.

Con el Esquema Conceptual se pretende especificar:

- Los objetos del sistema.
- Los aspectos estructurales de los objetos del sistema que garantizan el comportamiento propuesto.
- Los aspectos dinámicos de los objetos del sistema que garantizan el comportamiento propuesto.
- Las restricciones sobre los objetos del sistema que son necesarias para garantizar el comportamiento propuesto.

### 5.2. ENTREGABLE

#### 5.2.1. Introducción

En esta sección se procura verbalizar el enfoque con el que elaborará la especificación del sistema. Este determinará el grado de detalle con el que se lleva a cabo la especificación. Por ejemplo:

- La especificación se orienta a definir la arquitectura general del sistema (en particular de los datos). Esta será posteriormente redefinida y refinada en la etapa de diseño por el mismo grupo de desarrollo (quien definirá el modelo relacional, el grado de normalización, los “triggers”, las formas de

diálogo, las consultas, las funciones, las capas y la lógica asociada, las estrategias para garantizar las restricciones etc..). En este caso el detalle de la especificación puede ser mínimo ya que el grupo de desarrollo tomará directamente de las descripciones anteriores los elementos en cada etapa de la construcción.

- La especificación se orienta a crear un modelo del sistema susceptible de ser traducido de forma reglada y semiautomática a la plataforma de implantación. En este caso la especificación debe contener todos los elementos de las descripciones anteriores.

En este ejercicio se asumirá que se deben especificar formalmente en el Esquema Conceptual todos los elementos de las descripciones anteriores.

### 5.2.2. Consultas y Transacciones

Identifique y especifique las consultas y transacciones que implican los casos de uso propuestos en el capítulo anterior. Para ello se emplean las interfaces gráficas de usuario contenidas en los casos de uso y se describen las consultas relevantes en una lógica de predicados.

#### 5.2.2.1. Consultas

Figura 5.1. Consultas correspondientes al Caso de Uso Registrar Personas.

(1): {'CLIENTE', 'REPARTIDOR'}

(2): {z.Calcular() /  $z \in \text{Zona}$ }



The screenshot shows a software window titled "Registro de Pedidos". It contains the following elements with numbered callouts:

- 3**: Points to the "Pedido" text label.
- 4**: Points to the "Fecha:" text label.
- 5**: Points to the "Enviar Catálogo Digital" button.
- 6**: Points to the "Teléfono" text label.
- 7**: Points to the "Dirección" section, which includes dropdowns for "CLL" and "CRA", and a "No." text label.
- 8**: Points to the "Agregar Detalle Pedido" button.
- 9**: Points to the "Repartidor" text label.

Other visible elements include a "Nombre" text label, a "Zona" text label, a large empty text area, an "Asignar Repartidor" button, and "Aceptar" and "Cancelar" buttons at the bottom.

Figura 5.2. Consultas correspondientes al Caso de Uso Registrar pedidos.

- (3):  $|\{ p \in \text{Pedido} \}| + 1$
- (4):  $\{ p.\text{Telefono} / p \in \text{Persona} \wedge p.\text{Tipo} = \text{'CLIENTE'} \}$
- (5):  $\{ c.\text{Nombre} / c \in \text{Cliente} \wedge c.\text{Telefono} = \text{RegistrolePedidos.Cliente} \}$
- (6):  $\{ c.\text{Telefono} / c \in \text{Cliente} \wedge c.\text{Telefono} = \text{RegistrolePedidos.Cliente} \}$
- (7):  $\{ c.\text{Direccion} / c \in \text{Cliente} \wedge c.\text{Telefono} = \text{RegistrolePedidos.Cliente} \}$
- (8):  $\{ z.\text{Calcular}() / z \in \text{Zona} \}$
- (9):  $\{ r.\text{Asignar}() / r \in \text{Repartidor} \}$

The image shows a Windows-style dialog box titled "Agregar Detalle Pedido". It contains several input fields and buttons. Numbered annotations point to specific elements:

- 10**: Points to the "Pedido" text label.
- 11**: Points to the "Producto" dropdown menu.
- 12**: Points to the "Precio Lista" text input field.
- 13**: Points to the "Imagen" image placeholder area.

Other visible elements include "Cantidad" and "Observación" text input fields, and "Aceptar" and "Cancelar" buttons at the bottom.

Figura 5.3. Consultas correspondientes al Caso de Uso Registrar Pedido en su interacción Agregar Detalle Pedido.

(10):  $|\{ p \in \text{Pedido} \}| + 1$

(11):  $\{ p.\text{Nombre} / p \in \text{Producto} \}$

(12):  $\{ p.\text{PreciodeLista} / p \in \text{Producto} \wedge p.\text{Nombre} = \text{AgregarDetallePedido.Producto} \}$

(13):  $\{ p.\text{Imagen} / p \in \text{Producto} \wedge p.\text{Nombre} = \text{AgregarDetallePedido.Producto} \}$

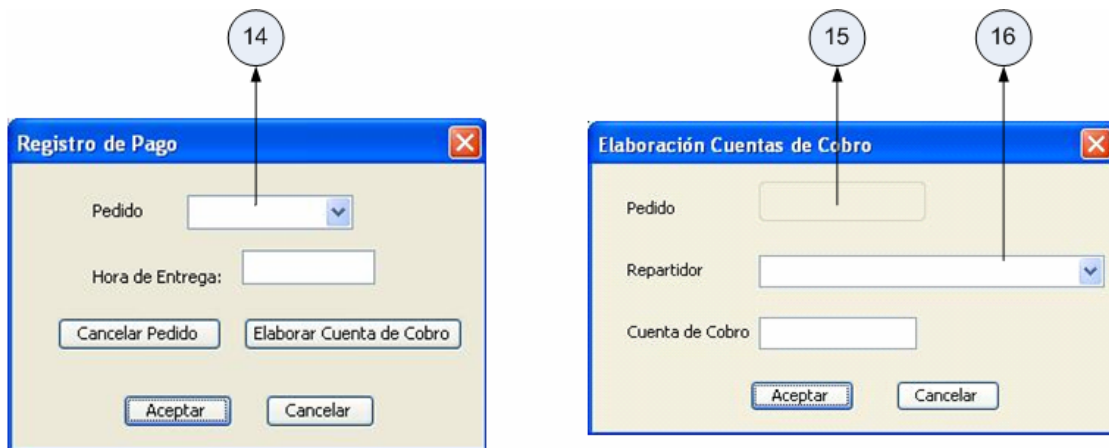


Figura 5.4. Consultas correspondientes al Caso de Uso Registrar Pago.

- (14):  $\{ p.\text{Numero} / p \in \text{Pedido} \wedge p.\text{Estado} = \text{'REGISTRADO'} \wedge \exists d \in \text{Despacho} (d.\text{Estado} = \text{'CERRADO'} \wedge p.\text{Numero} = d.\text{Pedido}.\text{Numero}) \}$
- (15):  $\{ p.\text{Numero} / p \in \text{Pedido} \wedge p.\text{Numero} = \text{RegistrodePago}.\text{Pedido} \}$
- (16):  $\{ r.\text{Nombre} / r \in \text{Repartidor} \}$



Figura 5.5. Consultas correspondientes al Caso de Uso Imprimir Despachos.

(17): { r.Nombre / r ∈ Repartidor }

#### 5.2.2.2. Transacciones

En los casos de uso se pretende la realización de un conjunto de operaciones que se deben ejecutar completamente para que efectivamente se considere que el caso de uso se completó. Este conjunto de operaciones que se deben realizar completamente reciben el nombre de Transacción. En UN-MÉTODO las transacciones, al igual que las consultas, se expresan en lógica de predicados.

- **REGISTRAR PERSONAS:**

Registrar(Tipo,Cedula,Nombre,Telefono,Direccion(Opc1,Dir1,Opc2,Dir2,No),E-mail)

**Sea:**

$P = \{p \in \text{Persona} / p.\text{Telefono} = \text{Registrar}().\text{Telefono}\}$

**Pre:**

$|P| = 0$

**Post:**

$|P| = 1$

P.Tipo = Registrar().Tipo

P.Cedula = Registrar().Telefono

P.Nombre = Registrar().Nombre

P.Telefono = Registrar().Telefono

P.Opcion1 = Registrar().Direccion.Opc1

P.Opcion2 = Registrar().Direccion.Opc2

P.Direccion1 = Registrar().Direccion.Dir1

P.Direccion2 = Registrar().Direccion.Dir1

P.Numero = Registrar().Direccion.No

P.Email = Registrar().E-mail

- **REGISTRAR PEDIDOS:**

Registrar(Fecha,Cliente,Direccion(Opc1,Dir1,Opc2,Dir2,No),  
(\*Detalle(Producto,Cantidad,Observacion)),Repartidor)

**Sea:**

$P = \{p \in \text{Pedido} / p.\text{Numero} = |\text{Pedido}| + 1\}$   
 $C = \{c \in \text{Cliente} / c.\text{Telefono} = \text{Registrar}().\text{Cliente}\}$   
 $D = \{d \in \text{Despacho} / d.\text{Repartidor.Nombre} =$   
 $\quad \text{Registrar}().\text{Repartidor} \wedge d.\text{Estado} = \text{'ABIERTO'}\}$   
 $N = |\text{Pedido}|$

**Pre:**

$|\text{P}| = 0$   
 $|\text{C}| = 1$   
 $\forall s \in \text{Registrar}().\text{Detalle} \exists pr \in \text{Producto} (pr.\text{Nombre} = s.\text{Producto})$

**Post:**

$|\text{P}| = 1$   
 $\text{P.Numero} = N + 1$   
 $\text{P.Fecha} = \text{Registrar}().\text{Fecha}$   
 $\text{P.HoradeAceptacion} = \text{Hora Actual del Sistema}$   
 $\text{P.Opcion1} = \text{Registrar}().\text{Direccion.Opc1}$   
 $\text{P.Opcion2} = \text{Registrar}().\text{Direccion.Opc2}$   
 $\text{P.Direccion1} = \text{Registrar}().\text{Direccion.Dir1}$   
 $\text{P.Direccion2} = \text{Registrar}().\text{Direccion.Dir1}$   
 $\text{P.Nro} = \text{Registrar}().\text{Direccion.No}$   
 $\text{P.Cliente.Telefono} = \text{Registrar}().\text{Cliente}$   
 $\forall s \in \text{Registrar}().\text{Detalle} \exists t \in \text{Detalle}$   
 $\quad (t.\text{Producto.Nombre} = s.\text{Producto} \wedge t.\text{Cantidad} = s.\text{Cantidad} \wedge$   
 $\quad \quad \quad t.\text{Observación} = s.\text{Observación})$   
 $\text{P.Detalle}[i] = \text{Registrar}().\text{Detalle}[i], \quad \forall i = 1, \dots, |\{\text{RegistrarPedidos}().\text{Detalle}\}|$   
 $|\text{D}| = 1$   
 $\text{P.Despacho} = \text{D}$

- **REGISTRAR PAGOS:**

Registrar(Pedido, Hora de Entrega)

➤ **Cancelar Pedido:**

Cancelar()

**Sea:**

$P = \{p \in \text{Pedido} / p.\text{Numero} = \text{Registrar}().\text{Pedido} \wedge$

$p.Estado \neq 'CANCELADO' \wedge p.Despacho.Estado = 'CERRADO'$   
 $S = \{s \in \text{Pago} / s.Pedido.Numero = \text{Registrar}().Pedido\}$

**Pre:**

$|P| = 1$   
 $|S| = 0$   
 $P./HoraLimite > \text{Registrar}().Hora \text{ de Entrega}$

**Post:**

$|S| = 1$   
 $P.Hora \text{ de Entrega} = \text{Registrar}().Hora \text{ de Entrega}$   
 $P.Estado = 'CANCELADO'$

➤ **Elaborar Cuenta de Cobro:**

$\text{Elaborar}(\text{Repartidor}, \text{Cuenta de Cobro})$

**Sea:**

$P = \{p \in \text{Pedido} / p.Numero = \text{Registrar}().Pedido \wedge$   
 $p.Estado \neq 'CANCELADO' \wedge p.Despacho.Estado = 'CERRADO'\}$   
 $C = \{c \in \text{Cuenta de Cobro} / c.Pedido.Numero =$   
 $\text{Registrar}().Pedido\}$

**Pre:**

$|P| = 1$   
 $|C| = 0$   
 $P./HoraLimite < \text{Registrar}().Hora \text{ de Entrega}$

**Post:**

$|C| = 1$   
 $P.Hora \text{ de Entrega} = \text{Registrar}().Hora \text{ de Entrega}$   
 $P.Estado = 'PENDIENTE'$   
 $C.Numero = \text{Elaborar}().Cuenta \text{ de Cobro}$   
 $C.Repartidor.Nombre = \text{Elaborar}().Repartidor$

• **IMPRIMIR DESPACHOS:**

$\text{Imprimir}(\text{Repartidor})$

**Sea:**

$$D = \{d \in \text{Despacho} / d.\text{Repartidor.Nombre} = \text{ImprimirDespachos}().\text{Repartidor} \wedge d.\text{Estado} = \text{'ABIERTO'}\}$$
**Pre:**

$$|D| = 1$$
**Post:**

D.Estado = 'CERRADO'  
D.Hora de Salida = Hora Actual del Sistema

**5.2.3. Diagrama de Clases**

En el capítulo 2 se definió el Modelo del Dominio como un primer acercamiento a las clases conceptuales del mundo con algunas de sus propiedades básicas. Una vez se han definido las consultas y transacciones que están presentes en la solución, se puede proceder a la complementación del Modelo del Dominio y su traducción al denominado Diagrama de Clases [OMG, 2006]. La simbología básica del Diagrama de Clases se puede apreciar en la Figura 5.6. y una mayor descripción y ejemplificación del mismo diagrama se puede consultar en [FOWLER, 2004].

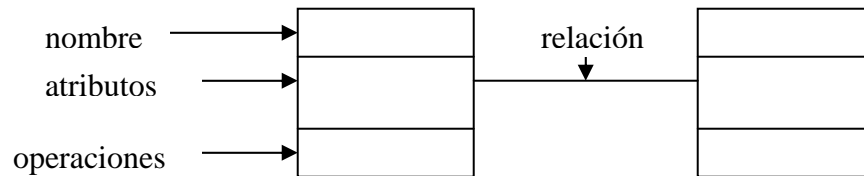


Figura 5.6. Simbología básica del Diagrama de Clases.

Las clases en este diagrama se representan como rectángulos con tres compartimientos correspondientes al nombre, los atributos y las operaciones respectivamente. Las relaciones pueden ser de varios tipos (para el ejemplo sólo se usarán tres: generalización, terminada en un triángulo en uno de los extremos, agregación, terminada en un rombo en uno de los extremos, y asociación, sin ningún elemento en los extremos).

En la Figura 5.7. se presenta el Diagrama de Clases de Rapizza Ltda.

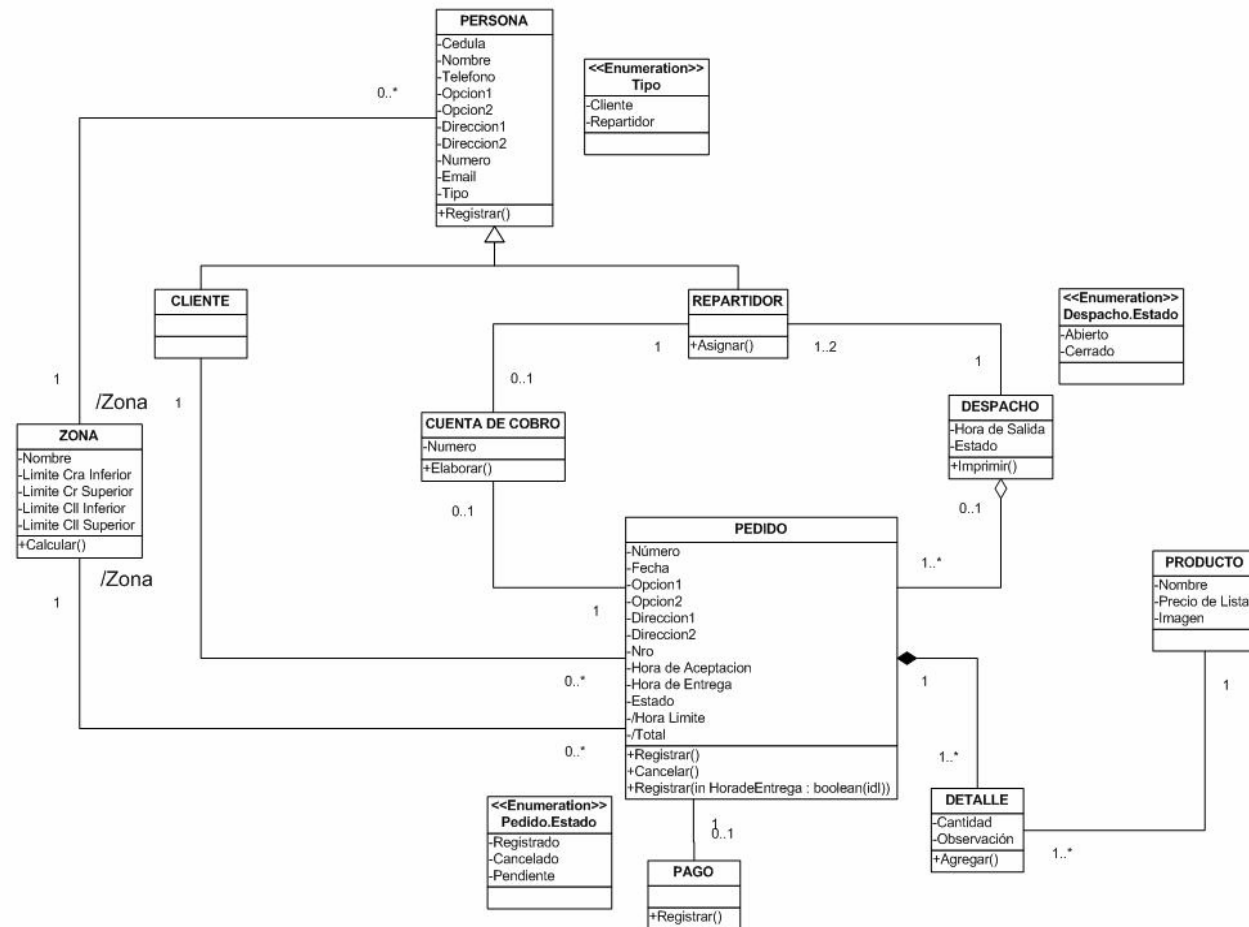


Figura 5.7. Diagrama de Clases correspondiente a Rapizza Ltda..



Paralelamente con el Diagrama de Clases, en UN-MÉTODO se deben identificar los atributos básicos y derivados (aquellos que se obtienen a partir de otros mediante algún tipo de fórmula), las vías de acceso a los objetos (atributos clave), y las relaciones básicas y derivadas que sean relevantes al comportamiento definido en los casos de uso. Además, se deben identificar las principales restricciones que hacen parte de la solución y especificar las derivaciones y restricciones en lógica de predicados.

## 5.2.4. Derivaciones y Restricciones

### 5.2.4.1. Derivaciones

A partir de los atributos y relaciones básicas, se presenta seguidamente la especificación en lógica de predicados de los atributos y relaciones derivadas.

- **Hora Limite:**

$$\text{Pedido.HoraLimite} = \text{Pedido.Hora de Aceptacion} + 30$$

- **Total:**

$$\text{Pedido.Total} = \sum_{i \in \text{Pedido.Detalle}} (\text{i.Producto.Precio de Lista} * \text{i.Cantidad})$$

- **Zona:**

Ver el evento Calcular Zona en Eventos y Operaciones (Numeral 5.2.5.1)

### 5.2.4.2. Restricciones

Una de las restricciones que se aplican a los estados de los objetos de Rapizza Ltda. se muestra seguidamente:

- Un pedido puede tener asociado un pago o una cuenta de cobro pero no las dos simultáneamente:

$$\forall p \in \text{Pedido} ( |p.\text{Pago}| = 1 \Rightarrow |p.\text{CuentadeCobro}| = 0 \\ \wedge |p.\text{CuentadeCobro}| = 1 \Rightarrow |p.\text{Pago}| = 0 )$$

### 5.2.5. Eventos y Operaciones

Como paso final del proceso de Elicitación de Requisitos de UN-MÉTODO, se deben especificar los eventos y operaciones que se realizan sobre los objetos del sistema, para lo cual nuevamente se emplea la lógica de predicados. Se utilizan adicionalmente diagramas de comunicación y de Máquina de Estados para complementar la especificación en lógica de predicados.

#### 5.2.5.1. Eventos y Operaciones

- **Calcular Zona:**

/Zona = Calcular(Opc1,Opc2,Dir1,Dir2)

**Sea:**

$Z = \{z \in \text{Zona}\}$

**Pre:**

$|Z| > 0$

**Post:**

$(\text{Opc1} = \text{'Cra'} \wedge \text{Opc2} = \text{'CII'} \wedge Z[i].\text{Limite Cra Inferior} \leq \text{Dir1} \wedge Z[i].\text{Limite Cra Superior} \geq \text{Dir1} \wedge Z[i].\text{Limite CII Inferior} \leq \text{Dir2} \wedge Z[i].\text{Limite CII Superior} \geq \text{Dir2}) \vee (\text{Opc1} = \text{'CII'} \wedge \text{Opc2} = \text{'Cra'} \wedge Z[i].\text{Limite CII Inferior} \leq \text{Dir1} \wedge Z[i].\text{Limite CII Superior} \geq \text{Dir1} \wedge Z[i].\text{Limite Cra Inferior} \leq \text{Dir2} \wedge Z[i].\text{Limite Cra Superior} \geq \text{Dir2}) \Rightarrow (/Zona = Z[i].\text{Nombre}),$   
 $\forall i = 1, \dots, |Z|$

$\neg ((\text{Opc1} = \text{'Cra'} \wedge \text{Opc2} = \text{'CII'} \wedge Z[i].\text{Limite Cra Inferior} \leq \text{Dir1} \wedge Z[i].\text{Limite Cra Superior} \geq \text{Dir1} \wedge Z[i].\text{Limite CII Inferior} \leq \text{Dir2} \wedge Z[i].\text{Limite CII Superior} \geq \text{Dir2}) \vee (\text{Opc1} = \text{'CII'} \wedge \text{Opc2} = \text{'Cra'} \wedge Z[i].\text{Limite CII Inferior} \leq \text{Dir1} \wedge Z[i].\text{Limite CII Superior} \geq \text{Dir1} \wedge Z[i].\text{Limite Cra Inferior} \leq \text{Dir2} \wedge Z[i].\text{Limite Cra Superior} \geq \text{Dir2})) \Rightarrow (/Zona = \text{'Fuera de la Zona de Cobertura'}),$   
 $\forall i = 1, \dots, |Z|$

#### 5.2.5.2. Máquina de Estados

Con el diagrama de Máquina de Estados Use una máquina de estados para especificar las secuencias de eventos permisibles sobre los objetos del sistema. En la Figura 5.8. se presentan los diagramas de Máquina de Estados para los objetos “pedido” y “despacho”. En estos diagramas el círculo negro es el estado inicial, el círculo negro dentro de un círculo blanco es el estado final, los rectángulos con los bordes redondeados son estados y las flechas son

transiciones. Una descripción más completa del diagrama de Máquina de Estados se puede consultar en [FOWLER, 2004].



Figura 5.8. Diagramas de Máquina de Estados para los objetos “Pedido” y “Despacho”, correspondientes al Diagrama de Clases de Rapizza Ltda..

### 5.2.5.3. Diagrama de Interacción

En UN-MÉTODO se usan los diagramas de interacción entre objetos para especificar las secuencias de eventos obligatorias luego de la ocurrencia de una transacción sobre el sistema. En la Figura 5.9. se puede apreciar el Diagrama de Comunicación correspondiente a Rapizza Ltda. En dicho diagrama, los rectángulos son los objetos, que pertenecen a una determinada clase, las líneas son las comunicaciones entre los objetos y las flechas corresponden a los mensajes que se envían entre los diferentes objetos, y que viajan por las comunicaciones. Una descripción más completa del diagrama nuevamente se puede consultar en [FOWLER, 2004].

## 5.3. ASPECTOS DE CONSISTENCIA Y REFINAMIENTO EN UN-MÉTODO

El Esquema Conceptual es el resultado final del proceso de Elicitación de Requisitos empleando UN-MÉTODO, pues contiene la especificación de la solución a los problemas de la organización, que se vino estudiando a partir de los entregables descritos en los capítulos 2, 3 y 4. Por pertenecer varios de los diagramas que se emplean en este entregable a UML, las reglas de consistencia entre esos diagramas son válidas también para UN-MÉTODO; sin embargo, como esas reglas no han sido claramente explicitadas en la especificación de UML, se hará un recuento de las mismas.



Figura 5.9. Diagrama de Comunicación correspondiente a Rapizza Ltda.

Adicionalmente, entre la especificación en Lógica de Predicados y la especificación en UML, debe existir completa consistencia, lo cual se refleja en otro conjunto de reglas de consistencia de UN-MÉTODO.

Las reglas de Consistencia del Esquema Conceptual de UN-MÉTODO son las siguientes:

1. Cada Diagrama de Máquina de Estados deberá referirse a una y sólo una clase del Diagrama de Clases.
2. Los estados correspondientes a un objeto en el Diagrama de Máquina de Estados deberán ser el resultado de una operación (o conjunto de operaciones) incluidas en el Diagrama de Clases.
3. Las clases de los objetos del Diagrama de Comunicación deberán estar contenidas en el Diagrama de Clases.
4. Los mensajes que se intercambian los objetos del Diagrama de Comunicación deben corresponder a operaciones de la clase del objeto de llegada en el Diagrama de Clases.
5. Los diferentes elementos (clases, atributos y operaciones) empleados en la especificación de consultas, transacciones, derivaciones, restricciones, eventos y operaciones, en lógica de predicados, deberán estar presentes en el Diagrama de Clases.

# Capítulo 6

## La Elicitación de Requisitos empleando UN-MÉTODO

En los capítulos 2 a 5 se realizó una exhaustiva y ejemplificada descripción de UN-MÉTODO. Desde el punto de vista del fin que persigue, UN-MÉTODO no es muy diferente de otros métodos que posibilitan la Elicitación de Requisitos; la real importancia de todos estos métodos radica en la elaboración de unos “planos” rigurosos de la aplicación que se va a construir y que deberá solucionar los principales problemas del dominio de los interesados. Se habla de “planos” haciendo el símil con otras disciplinas como la Arquitectura, la Ingeniería Civil o la Ingeniería Mecánica, en las cuales se planea meticulosamente cada detalle de una construcción, una obra civil o una máquina, antes de emprender su solución en el mundo real. Un edificio, por ejemplo, se analiza desde diferentes puntos de vista, y se pueden encontrar planos de su estructura, sus instalaciones eléctricas e hidrosanitarias, su distribución arquitectónica, etc. De manera similar, para una pieza de software se elaboran diagramas de Casos de Uso, Clases, Comunicación, Máquina de Estados y otros muchos que permiten especificar su comportamiento antes de construirlo. A diferencia de las diferentes creaciones tangibles de otras ciencias ingenieriles, el software como creación tan sólo hace unos pocos años se elabora siguiendo un diseño metodológico y es esto precisamente lo que procuran UN-MÉTODO y los demás métodos de desarrollo que se estudiaron en el capítulo 1.

La diferencia de UN-MÉTODO con los otros métodos de desarrollo analizados radica en el enfoque que se emplea para definir la solución. Mientras los demás métodos tratan de construir con el Interesado una solución de tipo informático a sus problemas, sin tratar de validar si esos problemas son efectivamente problemas reales del Interesado, en UN-MÉTODO el interés inicial es la determinación y validación de los problemas del interesado, comprendiendo su entorno, sus objetivos y procedimientos, así como la raíz de sus problemas, para poder definir un conjunto de soluciones que se puedan especificar y presentar al Interesado para su análisis y la selección de una de ellas. Así pues, mientras en RUP se procuran especificar los casos de uso y en XP se trata de construir las historias de usuario, modelos claramente enfocados a la solución, y que constituyen los pasos iniciales de esos métodos, en UN-MÉTODO se tratan de construir Esquemas Preconceptuales y Modelos del dominio, modelos claramente enfocados en el problema y en la terminología del área. Se trata en UN-MÉTODO

de identificar, seleccionar y especificar la solución que de mejor manera atiende los problemas de la organización, permitiéndole además acercarse a sus objetivos.

Otro aspecto que diferencia UN-MÉTODO de los otros métodos de desarrollo de software presentados radica en el énfasis que se hace en la consistencia a lo largo del proceso, lo cual permite mantener la coherencia de los diferentes artefactos que emplea, convirtiéndolos en vistas interconectadas del mismo modelo. Con esto se busca garantizar la trazabilidad en el proceso, de forma tal que los elementos del discurso del interesado que efectivamente sobrevivan a lo largo de las diferentes fases del ciclo de vida del software, se reflejen en los artefactos correspondientes a cada una de esas fases. Si bien en los otros métodos es posible construir las reglas de consistencia que vinculan los diferentes artefactos que emplean, dichas reglas no se encuentran explícitamente enunciadas en los diferentes métodos, correspondiendo al usuario del método definir aquellas reglas que considere pertinentes de uso.

Sin embargo, sería pretencioso afirmar que con UN-MÉTODO se pueden desplazar métodos de tanta aceptación en el mercado del software como RUP o CDM u otros métodos de creciente aceptación como XP o FDD. UN-MÉTODO simplemente es diferente a los otros métodos, los cuales pueden tener ámbitos de aplicación disímiles dependiendo de las características del problema. La idea detrás de UN-MÉTODO es el fomento de los métodos de desarrollo de software, con el fin de garantizar que la pieza de software antes de su construcción sea cuidadosamente documentada; los otros métodos, en la práctica, son adaptados por sus usuarios, haciendo más énfasis en algunos artefactos más que en otros, lo que los hace métodos igualmente versátiles para la Elicitación de los requisitos del software. No importa, en realidad, cuál es el método de desarrollo que se emplea; lo que realmente importa es que se use un método de desarrollo, pues de esta manera se puede garantizar que por lo menos la solución estará comprendida antes de iniciar la fase de construcción. Además, por la amplitud de los demás métodos, siempre será posible incorporar algunos otros artefactos que se consideren de valor para la comprensión y el análisis de las soluciones a los problemas de la organización.

## REFERENCIAS

[ANDERSON Y WENDELKEN, 1996] Anderson, C. y Wendelken, D. *The Oracle® Designer/2000 Handbook*. Addison-Wesley, 1996

[BECK, 2000] Beck, K. *Extreme Programming Explained: Embrace Change*. Reading, Addison-Wesley, 2000.

[COAD ET AL., 1999] Coad, P., LeFebvre, E., et al. *Java Modeling in Color with UML*, Prentice Hall, 1999.

[D'SOUZA Y WILLS, 1998] D'Souza, D. y Wills, A. *Objects, Components and Frameworks With UML: The Catalysis Approach*. Addison-Wesley, 1998

[DARDENNE ET AL., 1993] Dardenne, A., van Lamsweerde, A. y Fickas, S. *Goal-Directed Requirements Acquisition*. En: *Science of Computer Programming*, Vol. 20, 1993. Pp. 3-50.

[FOWLER, 2004] Fowler, M. *UML Distilled: A brief guide to the Standard Object Modeling Language*. Addison – Wesley, tercera edición, 2004.

[GUARINO, 1998] Guarino, N. *Formal Ontology and information systems*. En: *Proceedings of the FOIS'98 – Formal Ontology in Information Systems*, Trento, 1998.

[HARRINGTON, 1991] Harrington, J. *Business Process Improvement: The Breakthrough Strategy for Total Quality, Productivity and Competitiveness*. McGraw-Hill, 1991.

[ISHIKAWA, 1986] Ishikawa, K. *Guide to quality control*. Asian Productivity Organization, Tokio. 225 p. 1986.

[KARNER, 1993] Karner, G. *Metrics for Objectory*. Diploma thesis, University of Linköping, Sweden. No. LiTH-IDA-Ex-9344:21, 1993

[KRUCHTEN, 1999] Kruchten, Ph. *Rational Unified Process—An Introduction*. Addison-Wesley, 1999.

[OMG, 2006] Object Management Group. *OMG Unified Modeling Language Specification*. Object Management Group. Available: <http://www.omg.org/UML/>. [Citado 24 de Febrero de 2006].

[ORACLE, 2000] Oracle® Corporation. *Oracle Method<sup>SM</sup> CDM Quick Tour*. Oracle Corporation, Redwood City, 2000.

[ZAPATA *et al.*, 2006] Zapata, C. M., Arango, F. y Gelbukh, A. *Pre-conceptual Schema: a UML Isomorphism for Automatically Obtaining of UML Conceptual Schemas*. En: Research in Computing Sciences, Vol. 19, 2006. pp. 3-14.

[ZAPATA *et al.*, 2006b] Zapata, C. M., Villegas, S. M. y Arango, F. *Reglas de consistencia entre modelos de requisitos de UN-Método*. En: Revista Universidad Eafit, Vol. 42, No. 141, 2006. pp. 40-59.

[ZAPATA Y ARANGO, 2004] Zapata, C. M. y Arango, F. *Alineación entre Metas Organizacionales y Elicitación de Requisitos del Software*. En: Revista Dyna, Año 71, No. 143, 2004. pp. 101-110.

[ZAPATA Y GÓMEZ, 2006] Zapata, C. M. y Gómez, M. C. *Ingeniería del Software: Una Disciplina de Modelamiento*. Carlos Mario Zapata (Autor-Editor), Medellín, 2006.

[ZOWGHI Y GERVASI, 2002] Zowghi, D. y Gervasi, V. The Three Cs of Requirements: Consistency, Completeness and Correctness. En: Proceedings of 8th International Workshop on Requirements Engineering: Foundation for Software Quality, (REFSQ'02), Essen, 2002. pp. 155-164.



¿Sería usted capaz de construir un edificio de 40 pisos sin tener los planos necesarios para ello? Una labor semejante han acometido los desarrolladores de software desde los inicios de la ciencia de la computación. El desarrollo de software se volvió Ingeniería desde el momento en que se comenzaron a utilizar métodos de desarrollo que posibilitaron la construcción de los “planos” del software: los modelos y diagramas.

En este libro se presenta UN-MÉTODO para la Elicitación de Requisitos de software, el método que se ha venido articulando en la Escuela de Sistemas de la Universidad Nacional de Colombia a través de diversos proyectos de investigación y con la participación de varias generaciones de estudiantes de la línea de profundización en Ingeniería de Software. UN-MÉTODO combina algunos de los más reconocidos estándares de modelamiento con diagramas y artefactos provenientes de otras áreas del conocimiento, para especificar una solución que se adapte a los objetivos, problemas y necesidades de la organización.

Se espera que UN-MÉTODO pueda ser empleado por Analistas y Desarrolladores como una forma alternativa a los métodos convencionales de desarrollo de software.

FERNANDO ARANGO ISAZA es Doctor en Informática de la Universidad Politécnica de Valencia (España) y actualmente se desempeña como Profesor Asociado de la Escuela de Sistemas de la Universidad Nacional de Colombia

CARLOS MARIO ZAPATA JARAMILLO es Magíster en Sistemas y Candidato a Doctor en Ingeniería de la Universidad Nacional de Colombia. En la actualidad se desempeña como Profesor Asistente de la Escuela de Sistemas de la Universidad Nacional de Colombia