Centro de Investigación Científica y de Educación Superior de Ensenada

DESARROLLO DE UNA ARQUITECTURA DE COORDINACION DE PROCESOS ORGANIZACIONALES EN INTERNET

TESIS
MAESTRIA EN CIENCIAS

FRANCISCO GARCIA CARRILLO

Ensenada, *Baja California*, Mexico. Diciembre de 2001.

TESIS DEFENDIDA POR

Francisco García Carrillo

Y APROBADA POR EL SIGUIENTE COMITÉ

Dra. Ana Isabel Martínez García

Director del Comité

M.C. Josefina Rodríguez Jacobo

Miembro del Comité

Dr. David Hilario Covarrubias Rosales

Miembro del Comité

Dr. Fernando Rojas Iñíguez

Miembro del Comité

Dr. Guillermo Licea Sandoval

Miembro del Comité

M.C. José Luis Briseño Cervantes

Jefe del Departamento de
Ciencias de la Computación

Dr. Luis Alberto Delgado Argote

Director de Estudios de Posgrado

5 de Diciembre del 2001

CENTRO DE INVESTIGACIÓN CIENTÍFICA Y DE EDUCACIÓN SUPERIOR DE ENSENADA

DIVISIÓN DE FÍSICA APLICADA

DEPARTAMENTO DE CIENCIAS
DE LA COMPUTACIÓN

Desarrollo de una Arquitectura de Coordinación de Procesos Organizacionales en Internet.

TESIS

Que para cubrir parcialmente los requisitos necesarios para obtener el grado de
MAESTRO EN CIENCIAS presenta:

FRANCISCO GARCÍA CARRILLO

Ensenada, Baja California, México, Diciembre del 2001.

RESUMEN de la tesis de **FRANCISCO GARCÍA CARRILLO**, presentada como requisito parcial, para la obtención del grado de **MAESTRO EN CIENCIAS** en **CIENCIAS DE LA COMPUTACIÓN**. Ensenada, Baja California, México. Diciembre del 2001.

Desarrollo de una Arquitectura de Coordinación de Procesos Organizacionales en Internet.

Resumen aprobado por:

Dr. Ana Isabel Martínez García
Directora de Tesis

Hoy en día la necesidad de las organizaciones por producir más en menor tiempo y reducir los costos asociados, ha hecho que éstas se preocupen por mejorar sus procesos y aprovechar los recursos, tanto humanos como materiales, de manera más adecuada. Para entender y visualizar esos procesos es importante modelarlos a través de alguna técnica diagramática que muestre los elementos involucrados en forma sistemática, como son: las personas, los objetivos, las tareas, las responsabilidades, entre otros.

Una vez que los procesos han sido analizados y mejorados, una manera de brindar apoyo a la ejecución de los mismos, es a través del uso de Tecnología de Información (TI) para facilitar la realización de las actividades. Este soporte puede ser proporcionado por medio de sistemas que coordinen las actividades que deben realizar las personas involucradas en los procesos, las interacciones o comunicación que éstas llevan a cabo y las entidades de información que están siendo manipuladas.

En este sentido, se propone el desarrollo de una arquitectura de coordinación, que facilite la transformación del modelo estático de un proceso organizacional realizado con diagramas rol-actividad (RAD), a un sistema de coordinación que lleve a cabo la ejecución del proceso en tiempo real (modelo dinámico). Esta arquitectura hace uso de la tecnología de Internet (particularmente el *Web*), permitiendo la coordinación de personas físicamente distribuidas, haciendo más eficiente su participación en los procesos.

Palabras clave: proceso, modelo, RAD, roles, arquitectura, sistemas de coordinación, *enactment*.

Abstract of the thesis of **FRANCISCO GARCÍA CARRILLO**, presented as partial requirement, to obtain the master title of **MASTER IN SCIENCIES** in **Computer Science**. Ensenada, Baja California, Mexico. December of 2001.

"DEVELOPMENT OF A COORDINATION TECHNOLOGY (ENACTMENT) FOR ORGANIZATIONAL PROCESSES IN INTERNET"

Abstract approved by:

Dra. Ana Isabel Martínez García
Thesis Director

Nowadays, the need of organizations to increase their production in less time and reduce costs, has made them to worry for improving their processes and to use their human and material resources in a better manner. To be able to understand, analyze and improve processes, it is important to model them using diagrammatic techniques that capture their main elements such as: people, objectives, tasks and responsibilities, in a systematic form.

Once the process has been analyze and improved, one way to provide support to the processes execution is through the use of Information Technology (IT) to facilitate the developments of their activities. This support can be provided with coordination systems to coordinate the activities carried out by the process' agents, its interactions or communication with others, and the information entities that are being manipulated.

According with the facts established before, we propose a coordination architecture to facilitate the transformation of a static process model, represented with Role Activity Diagrams (RADs), into a coordination system. Where the last, will enable real time process execution or enactment (dynamic model) driven by the agents who represent the roles interacting with the support of a computer software environment. The architecture uses the Internet Technology (the Web in particular), to enable the coordination of people physically distributed and to make their participation in the process more efficient.

Keywords: process, model, RAD, roles, architecture, coordination systems, enactment.

Dedicatoria

¡A mi madre Toñita, hermanos

Gina y Toño, y mi sobrino Edy.

Los quiero mucho!

Agradecimientos.

Un especial agradecimiento a mi directora de tesis, Dra. Ana Isabel Martínez García, por su invaluable apoyo y guía en la realización de este trabajo, además de su agradable amistad. Gracias.

A los miembros del comité de tesis, Dr. David Hilario Covarrubias Rosales, Dr. Guillermo Licea Sandoval, M.C. Josefina Rodríguez Jacobo y Dr. Fernando Rojas Iñiguez, por la aportación realizada a este trabajo con sus valiosos comentarios.

Al Dr. Fernando Rojas Iñiguez por sus comentarios y sugerencias realizadas en el transcurso de este trabajo.

A los profesores del Departamento de Ciencias de la Computación que contribuyeron en mi formación.

A mis compañeros y amigos de generación Sonia Sosa, Rafael Villa, José Antonio Cárdenas, Aglay González, Brenda Flores, Yolanda Ávila, Yuridia Hernández, Jorge Ibarra, Adrián Munguía, gracias por compartir conmigo su amistad durante todo este tiempo.

A los miembros de "La Mora Mecánica" gracias por su amistad y los momentos que pasamos juntos en los torneos deportivos.

A Elitania por bridarme su apoyo, confianza y motivación necesaria para terminar este trabajo. Te quiero mucho mi K!

Al Consejo Nacional de Ciencia y Tecnología.

Contenido

Página

Lista de Figuras

Figura **Página**

Figura **Página**

Figura **Página**

Lista de Tablas

Capítulo I. Introducción

En la actualidad la eficiencia y rapidez que se requiere para el desempeño de las actividades dentro de una organización son cada vez más importantes. Por lo tanto, las organizaciones están buscando formas de mejorar y optimizar la manera en que las llevan a cabo analizando sus procesos. El análisis, rediseño y soporte que se realiza sobre las actividades de los procesos se apoya en la utilización de Tecnología de Información (TI) para facilitar su ejecución.

Una manera de brindar ese soporte es a través del desarrollo de sistemas que coordinen las actividades realizadas por las personas involucradas en los procesos, además de las interacciones y el intercambio de información que éstas tienen. En este sentido, se propone el desarrollo de una arquitectura de coordinación que permita la transición del modelo de un proceso organizacional (modelo estático) a un sistema que lleve a cabo la ejecución del mismo (modelo dinámico o activo).

I.1 Antecedentes

Cuando se analiza un proceso organizacional y la forma en que se le da soporte, es importante realizar un estudio que ayude a entender y visualizar sus objetivos de una mejor manera. Para este propósito se utiliza la ingeniería y el modelado de procesos que facilitan la comunicación y el entendimiento de los procesos, permiten evaluarlos, proporcionan el soporte para su mejora, y dan soporte al manejo de los mismos, entre otros aspectos (Curtis, B. *et al.*, 1992).

Los procesos están formados por *roles* que se comunican entre sí (interacciones) y ejecutan diferentes actividades para alcanzar un objetivo en común. Estos roles por su parte, son un conjunto de elementos del proceso (actividades) asignados a *agentes* que pueden ser personas, máquinas o sistemas, y que ejecutan parte de las actividades. De aquí que podemos definir un proceso como un conjunto de roles que interactúan entre sí realizando actividades (repetibles) parcialmente ordenadas con el fin de lograr un objetivo común (Curtis, B. *et al.*, 1992).

Para el estudio de procesos organizacionales es necesario seguir una metodología que permita tener un control del análisis que se va a realizar. Una metodología estándar como la PADM (*Process Analysis and Design Methodology*) (Wastell, D. *et al.*, 1994) para el estudio de procesos consiste en la captura de información, el modelado del proceso que proporciona una idea global de las actividades que se realizan, la evaluación y el análisis del modelo, el rediseño y el soporte al proceso mediante el uso de TI para llevarlo a su estado de madurez con la creación de un sistema que represente la funcionalidad del proceso.

En la fase de modelado existen muchas técnicas para la representación de procesos, entre éstas sobresalen los Diagramas de Flujo, Diagramas de Definición Integrada (IDEF0 por sus siglas en inglés) y los Diagramas Rol-Actividad (RAD, Rol-Activity Diagram) (Miers, D., 1996). Los RADs son diagramas que capturan las tareas, objetivos y personas involucradas en un proceso a roles, actividades e interacciones (Rojas Iñiguez y Martínez García, 1998) capturando muchos de los aspectos de las perspectivas propuestas por Curtis:

funcional (elementos del proceso), de comportamiento (¿cuándo y cómo se ejecuta una actividad?), organizacional (¿dónde y quién la ejecuta?) y algunos elementos de la parte informacional (reportes, formatos, etc.) representando un modelo completo.

Por otro lado, la TI proporciona técnicas y herramientas que ayudan a las organizaciones a ejecutar sus actividades o procesos de una forma más eficiente. Algunos ejemplos son herramientas CASE, técnicas de programación orientada a objetos, modelado de procesos, entre otros. Hasta hace poco, la TI se había enfocado en el problema de crear herramientas que apoyaran la participación de personas con sus ideas en las organizaciones. Actualmente se está viendo un gran crecimiento en la utilización de TI orientada hacia *coordinar* la participación individual dentro de las organizaciones (Greenwood, R.M. *et al.*, 1998).

Con el crecimiento de las redes de computadoras se ha hecho posible la explotación de TI para dar soporte a la comunicación entre varios individuos físicamente distribuidos, abriendo la posibilidad al uso de diferentes tecnologías para llevar a cabo la coordinación de grupos de trabajo realizando distintos procesos. Dentro de estas tecnologías tenemos aquellas utilizadas en el *enactment* de procesos, es decir, las que capturan la representación de un modelo en un lenguaje de programación o herramienta, para llevar a cabo la ejecución simultánea del programa y el proceso real, interactuando de alguna manera (Fernström, C. y Lennart, F., 1991). Estos sistemas utilizados en el *enactment* de procesos se conocen como sistemas de coordinación, y son aquellos donde los usuarios pueden distribuir sus materiales de trabajo y responsabilidades, establecer acuerdos de cooperación, y terminar sus tareas de manera guiada para asegurar que las personas adecuadas realizan

las actividades correspondientes (Roberts, C., 1993), todo esto basado en una arquitectura de coordinación. En donde esta última, es aquella que soporta el trabajo en grupo a través de la utilización de equipo de cómputo, redes de computadoras y sistemas, proporcionando procesos coordinados de una mejor manera.

De aquí que el alcance de la TI no se limita a procesos locales dentro de una organización, sino que se puede dar soporte a procesos que requieren la interacción y coordinación entre dos o más organizaciones. Esto es posible gracias a *"Internet"*, que es una red de computadoras distribuidas por todo el mundo, permitiendo manipular una gran cantidad de información accesible a aquellas personas que la soliciten con sólo tener una conexión a esta red por medio de una dirección del Protocolo de Internet (IP).

Los sistemas de coordinación representan ciertas ventajas para las organizaciones. Entre éstas se encuentran la ejecución más eficiente del proceso, proporciona otro tipo de sistemas o aplicaciones para facilitar la realización de las actividades, la coordinación entre los miembros de una organización está claramente definida, el tiempo en que se lleva a cabo el proceso se reduce, en ocasiones se minimizan los costos inherentes a la ejecución del proceso, entre otras ventajas. Estos sistemas facilitan y mejoran la manera de alcanzar los objetivos principales de una organización.

Actualmente se está desarrollando una herramienta de apoyo a la Ingeniería de Procesos Asistida por Computadora (CAPETool por sus siglas en inglés) (Flores Ríos, 2001). Ésta se propone como una herramienta integral de soporte a todas las fases de la Ingeniería de

Procesos. Donde una de las fases es el soporte a procesos utilizando TI, en particular por medio del *enactment* de procesos con la ejecución de sistemas de coordinación.

I.2 Planteamiento del problema

Cuando un proceso organizacional es analizado para mejorar su eficiencia, es importante realizar su modelo utilizando alguna técnica diagramática. El modelado de procesos tiene principalmente cinco usos básicos que van desde el entendimiento del proceso hasta el soporte para la ejecución automatizada. Uno de sus usos es facilitar el entendimiento y la comunicación humana de los procesos proporcionando una representación común del modelo a las personas. En un segundo aspecto, se encuentra el soporte a la mejora del proceso identificando áreas problemáticas y estableciendo responsabilidades más acertadas a las personas involucradas, haciendo que el proceso esté mejor definido. Una vez que tenemos este nivel de efectividad del proceso, podemos pensar en el soporte a la administración del mismo para monitorear, administrar y coordinar a las personas, además de establecer métricas de medición del proceso. En este punto estamos hablando de un proceso cuyo grado de madurez es mayor y en el cual se presenta el cuarto de los usos que es el soporte a una dirección automatizada de la ejecución del proceso apoyado en su modelo como una guía para facilitar su ejecución. Finalmente, llegamos a la última fase de madurez del proceso en donde ya estamos hablando del soporte a la ejecución automatizada del mismo, apoyando el trabajo cooperativo entre las personas, equipo y sistemas. A está última fase de soporte es donde precisamente está enfocado este trabajo. En particular la construcción de una arquitectura de coordinación que facilite la evolución de un modelo estático (diagramático) a un modelo activo (que muestre la dinámica de los elementos de un proceso) generando sistemas de coordinación para el *enactment* de procesos.

La Figura 1 muestra los cinco usos del modelado mencionados anteriormente.

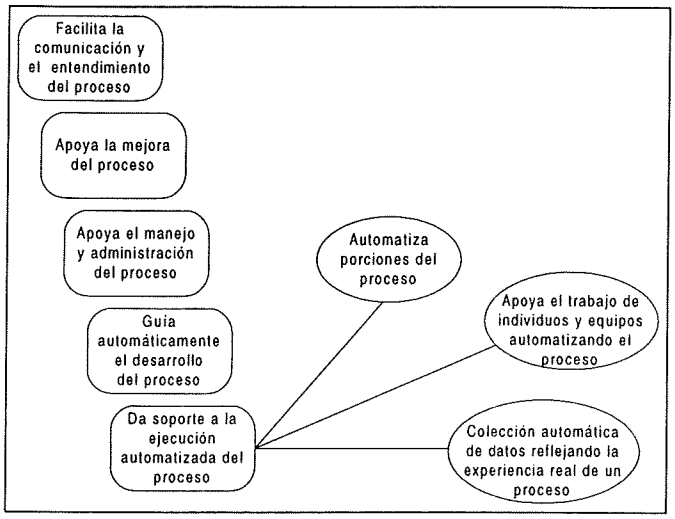

Figura 1. Cinco usos básicos del modelado de procesos.

Con base en lo anterior, la arquitectura que se propone permitirá crear sistemas para dar soporte a procesos a través de diferentes plataformas permitiendo la interacción de los roles en un proceso organizacional. Debido a que los RADs representan claramente los elementos de un proceso, esta técnica diagramática será utilizada como base en la generación de sistemas de coordinación por medio de la arquitectura. Por otro lado, algunos requerimientos importantes que deben ser considerados en el desarrollo de la arquitectura para producir el *enactment* de un proceso son: la persistencia de los objetos generados, la comunicación entre los diferentes roles (capa de coordinación), un manejador o agente de estados que controle el estado actual para cada rol en cualquier momento, además de las actividades y roles de un proceso.

En la Figura 2, se muestra un sistema global de ingeniería de procesos en donde se encuentra inmerso el *enactment* de procesos que se lleva a cabo a través de una arquitectura de coordinación. El rectángulo representa un proceso organizacional que es analizado con

el propósito de establecer mejoras mediante el rediseño o la construcción de un nuevo modelo.

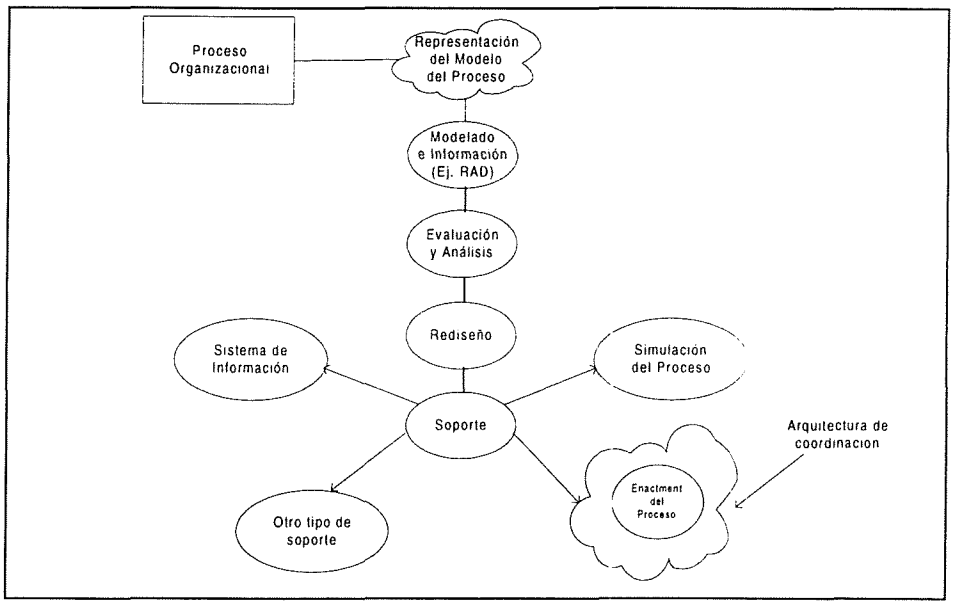

Figura 2. Sistema global de un proceso organizacional.

Una vez que se tiene el modelo en el último grado de madurez mencionado anteriormente, se pueden utilizar sistemas de información u otro tipo de TI para darle soporte, realizar simulaciones antes del rediseño del proceso o con los nuevos elementos, así como la realización de sistemas de coordinación que brinden soporte automatizado a la ejecución del proceso, siendo éste uno de los objetivos que deseamos lograr con este trabajo.

Otro aspecto importante a considerar es el hecho de poder reutilizar la representación del modelo estático de un proceso facilitando la generación de diferentes sistemas de soporte, de aquí que se proponga establecer un modelo base reusable que además de contener las características estáticas del proceso en RADs (utilizando tal vez un lenguaje de marcado como XML) (Flores Ríos, 2001), pueda ser enriquecido con atributos de comportamiento e

información necesarios en la generación de sistemas de coordinación utilizando una arquitectura genérica. XML (Extensible Markup Language) (Rusty, E. y Scott, W., 2001) es un lenguaje que permite definir un tipo de documento específico con reglas estructurales bien claras y que podría seguir la estructura de un RAD de manera formal dando la pauta para la generación del modelo base (textual) de un proceso con elementos como actividades, roles, interacciones, condicionales, entre otros.

Considerando lo anterior, la arquitectura de coordinación que se propone deberá aceptar como entrada de información el modelo base de un proceso, generando un sistema de coordinación que implemente la interacción entre los roles y sus actividades para poder llevar a cabo el *enactment* del proceso permitiendo alcanzar los objetivos para los que fue diseñado.

Basado en la idea de un modelo base reusable, la Figura 3 presenta los elementos genéricos que deberá contener una arquitectura de coordinación que permita a las usuarios interactuar dentro de un proceso organizacional.

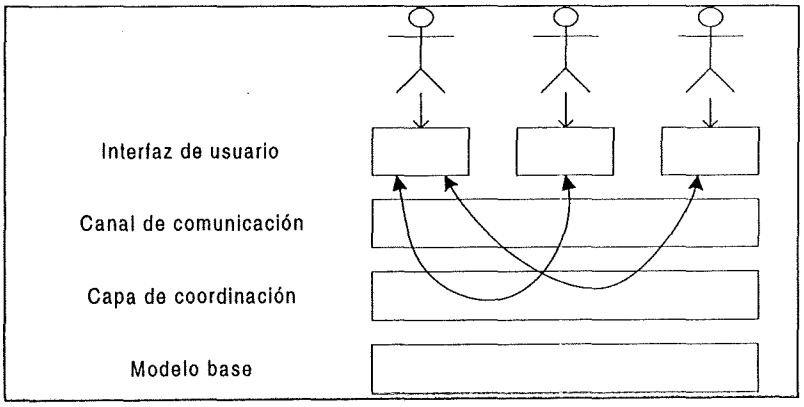

Figura 3. Interacción de los usuarios con la arquitectura de coordinación.

Los cuadros superiores representan las interfaces a través de las cuales los usuarios pueden ejecutar sus actividades. El rectángulo que sigue hacia abajo (canal de comunicación) permite llevar la información desde los usuarios hacia la capa de coordinación y viceversa. La capa de coordinación informa a los usuarios las actividades que deben ejecutar en un momento determinado y maneja las interacciones que éstos tienen de acuerdo a los roles que representan dentro del proceso en que participan. Por otro lado, esta capa extrae la información de un proceso tomando los datos de su modelo base mostrado en la capa inferior.

A medida que los roles van realizando sus actividades pasan por diferentes estados. Éstos no se ven reflejados explícitamente en el diagrama RAD de un proceso, sin embargo, es importante identificarlos para poder realizar la coordinación de un proceso. Una manera de representar esta información sería utilizando una tabla de estados como la que se muestra en la Tabla I.

Tabla I. Tabla de estados.

No	**Estado**	**Interacción**	**Acción**	**.....**
1	Llamando	Rol_3	Enviar inf.	.....
2	Esperando	Rol_1	Recibir inf.	.....
.....	.....	.....		.....

Donde por ejemplo, la primer columna enumera los diferentes estados de cada rol, la segunda columna tiene los estados de los roles, además de otras columnas con datos como con quién interactúa un rol, la acción que va a realizar, etc.

I.3 Caso de estudio

Es este trabajo se realiza un caso de estudio con el propósito de identificar los requerimientos que demanda un proceso organizacional real cuando se desea utilizar TI (como sistemas de coordinación) para darle soporte. Además, otra de las razones es observar la problemática que se presenta cuando las organizaciones no definen claramente las responsabilidades de los roles involucrados en los procesos, ni tampoco los requisitos que se deben cumplir para pasar de una actividad a otra. Esto puede provocar problemas significativos que no permitan utilizar sistemas de coordinación para mejorar la ejecución de los procesos. En este mismo sentido, identificar cuales son las partes de un proceso que pueden ser apoyadas mediante TI y cuales no, y las razones o características que se presentan que no permiten brindar ese soporte, por ejemplo, que las responsabilidades no estén bien definidas.

Finalmente, considerando lo anterior, se prueba la funcionalidad de la arquitectura de coordinación para apoyar la ejecución automatizada de una parte del proceso del caso de estudio. En el apéndice A se presenta una parte del caso de estudio mencionado.

I.4 Objetivo general

El objetivo principal es desarrollar el prototipo de una arquitectura de coordinación para la generación de sistemas que permitan la representación de procesos y su coordinación basados en modelos RAD, facilitando el *enactment* de un proceso.

I.5 Objetivos específicos

- Entender el problema de coordinación de procesos organizacionales distribuidos a través de la construcción de una herramienta de soporte basada en los RADs.
- Definir los requerimientos básicos de la herramienta de soporte.
- Definir los elementos de una representación abstracta formal del modelo, específicos para la coordinación de procesos.
- Diseñar la herramienta de soporte.
- Implementar las diferentes fases de la herramienta.
- Desarrollar un caso de estudio real para probar la funcionalidad de la herramienta.

I.6 Contenido de la tesis

El contenido de la tesis está distribuido en siete capítulos y dos apéndices que se explican brevemente a continuación.

En el capítulo II se introducen los conceptos y elementos necesarios para el desarrollo de este trabajo en el área de Ingeniería de Procesos. Se definen conceptos como proceso, roles, interacción, agentes, modelado, entre otros. El capítulo III presenta un análisis comparativo de las características que contienen algunas arquitecturas de coordinación. Posteriormente el capítulo IV describe el desarrollo de un prototipo basado en el modelo del "Juego de la Cerveza" que consiste de tres roles: el *dueño de un bar*, los *clientes* y una *compañía distribuidora* de cerveza, con la finalidad de identificar los elementos importantes para el desarrollo de la arquitectura de coordinación que nos permita generar sistemas para el *enactment* de un proceso. El capítulo V describe el diseño de la arquitectura de coordinación propuesta. El capítulo VI presenta las pruebas realizadas y los resultados

obtenidos. Finalmente el capítulo VII se establecen las conclusiones de la tesis y el trabajo a futuro.

Los apéndices A y B contienen información referente al caso de estudio de la Unidad de Medicina Familiar No. 32 y parte del diseño de la herramienta, respectivamente.

Capítulo II. Ingeniería y modelado de procesos

II.1 Introducción

Poco después de la revolución tecnológica en los años 70´s y 80´s, las empresas se apoyaron en el uso de software genérico para darle soporte a sus procesos adaptándolo a sus necesidades. En esta época, este tipo de software ya no satisfacía los requerimientos de las organizaciones de tal manera que empezaron a demandar sistemas hechos a la medida. Esta necesidad dio la pauta al estudio del proceso de desarrollo de software y con esto al modelado del proceso mismo, surgiendo así el modelado de procesos como un conjunto de técnicas para la mejora del proceso de desarrollo de sistemas. Estas técnicas ayudan a las organizaciones a entender sus procesos, rediseñar nuevas formas de realizarlos, representar en forma gráfica las actividades de los mismos, aplicar reingeniería, aplicar técnicas para la calidad total y eventualmente proporcionar soporte a los procesos utilizando sistemas de cómputo (Ould, M., 1993). A través del modelado se pueden representar las distintas actividades que tienen lugar durante la ejecución de un proceso. Cuando se modelan procesos, por lo general aparecen deficiencias por diferentes razones: la falta de una buena administración, la falta de herramientas o equipo necesarios, la existencia de demasiadas personas realizando una misma actividad o por el contrario la carencia de éstas, la falta de una coordinación adecuada entre los roles, el no delimitar responsabilidades de las personas que participan en el proceso, entre otras.

II.2 Modelado de procesos

El modelado de procesos se define como la representación abstracta de un proceso actual o propuesto, que representa elementos selectos considerados importantes para el estudio de un proceso organizacional (Rojas Iñiguez, 1998). Para este modelo se puede crear un sistema de coordinación que sea ejecutado por un humano o una máquina.

En esta tesis se define un proceso como un conjunto de roles que colaboran y llevan a cabo actividades (elementos de un proceso) parcialmente ordenadas con la finalidad de alcanzar algunas metas. Donde los roles son un conjunto de elementos del proceso asignados a un agente como una unidad de responsabilidad funcional. Los agentes a su vez, ejecutan los elementos del proceso y están representados por una persona, un grupo de personas, un sistema o una máquina. Durante la ejecución (*enactment*) de un proceso o un elemento del mismo, también se crean o modifican sus productos (artefactos).

Por su parte, la Ingeniería de Procesos es una colección de técnicas para el análisis, diseño y evolución de los procesos basado en el uso del modelado de procesos, además de seguir una metodología bien definida para este fin (Kawalek, P., 1997).

El modelado de procesos abarca un amplio intervalo de usos. Cinco de los más básicos se presentan en la Tabla II y cubren desde el entendimiento hasta la ejecución automatizada (Curtis, B. *et al.*, 1992):

Tabla II. Usos del modelado de procesos.

Facilita el entendimiento y la comunicación humana
• Representa el proceso de manera entendible para los humanos. • Habilita la comunicación y los acuerdos sobre los procesos. • Formaliza el proceso para que las personas trabajen en equipo eficientemente. • Proporciona suficiente información que permita a un individuo o equipo ejecutar el proceso.
Soporte a la mejora del proceso
• Identifica los componentes necesarios del desarrollo de software o el mantenimiento del proceso. • Reutiliza procesos bien definidos y efectivos en proyectos futuros. • Compara alternativas de procesos. • Estima el impacto de los cambios potenciales a un proceso sin ponerlos en práctica. • Apoya en la selección e incorporación de tecnología (por ejemplo, herramientas) dentro de un proceso. • Facilita el aprendizaje organizacional considerando los procesos efectivos.
Soporte a la administración del proceso
• Desarrollar el proceso de un proyecto específico para colocar los atributos de un proyecto particular. Tal como su producto ó ambiente organizacional. • Soporte al desarrollo de planes para el proyecto. • Monitorea, administra y coordina el proceso. • Proporciona una base para la medición del proceso.
Dirección automatizada en la ejecución de procesos
• Define un ambiente efectivo de desarrollo de software. • Proporcionar una guía, sugerencias y material de referencia para facilitar al personal la ejecución del proceso.
Soporte a la ejecución automatizada
• Automatiza partes del proceso. • Soporte al trabajo cooperativo entre individuos y equipos, automatizando el proceso. • Obtener automáticamente los datos de medición que reflejan la experiencia actual con un proceso. • Reforzar las reglas para asegurar la integridad del proceso.

La Tabla II presenta sólo algunos de los usos que se le puede dar al modelado de procesos. También se puede extraer información interesante para los miembros de una organización a partir de los modelos como: *¿qué se va a hacer?*, *¿quién lo va a hacer?*, *¿cuándo y dónde será hecho?* y ¿de *quién depende que esto se haga?*. En la Figura 4 se presentan cuatro

perspectivas relacionadas con las preguntas anteriores, estas son la funcional, organizacional, de comportamiento e informacional.

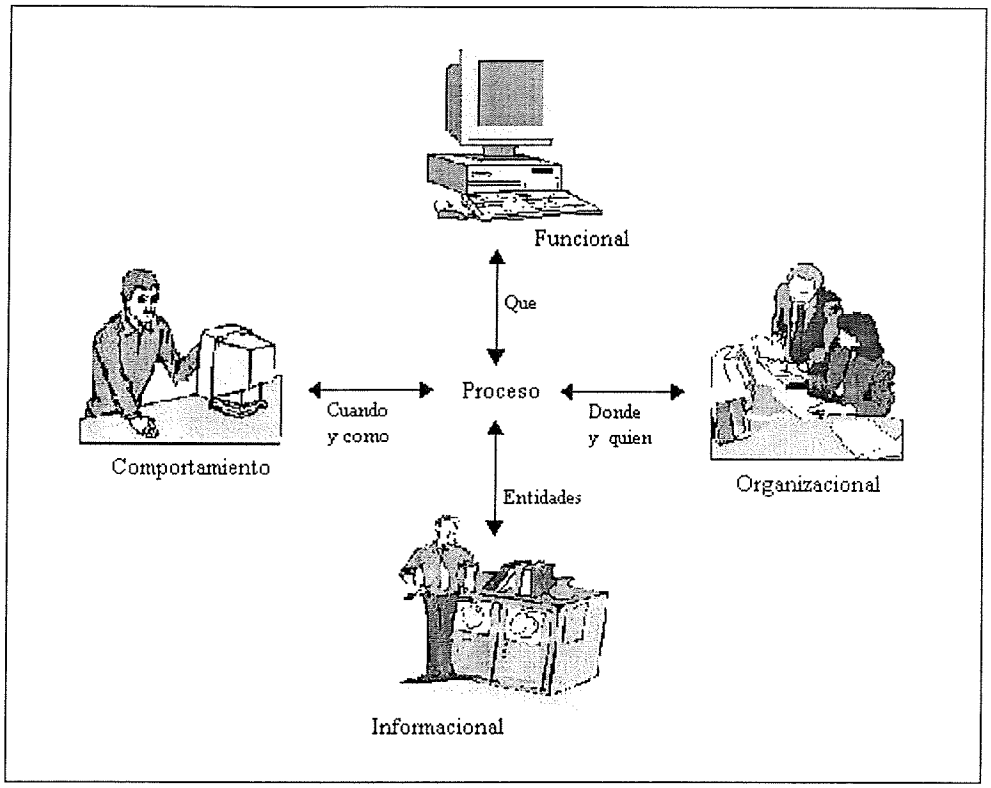

Figura 4. Perspectivas propuestas por Curtis para el modelo de un proceso.

Donde la perspectiva *funcional* representa qué elementos del proceso están siendo ejecutados y qué entidades de información son relevantes para estos elementos (ej. solicitar cita médica); la *organizacional* representa dónde y por quién son ejecutados los elementos (ej. asistente médica); de *comportamiento* especifica cuándo son ejecutados los elementos del proceso (ej. cuando el solicitante entrega su tarjeta de citas); por último, la *informacional* representa las entidades de información producidas o manipuladas por el proceso. Estas entidades incluyen datos, artefactos, productos y objetos (ej. cita médica o tarjeta de citas, etc.).

Existen algunas técnicas para el modelado de procesos que se han desarrollado en las últimas dos décadas en las cuáles podemos identificar el soporte a una o más de las perspectivas descritas. En la siguiente sección se introducen algunas de sus características.

II.3 Técnicas de modelado de procesos

La captura de un proceso es un paso importante en el análisis del mismo, en particular es muy útil para identificar las responsabilidades de todos los agentes involucrados. El modelo de un proceso se representa y visualiza mucho mejor utilizando diagramas (Rojas Iñiguez, 1998). Algunas de las técnicas diagramáticas que han sido desarrolladas o adaptadas para este propósito son las siguientes:

- Diagrama de flujo de datos (Sommerville, I., 1996): es una manera útil e intuitiva de describir un sistema. Son entendibles normalmente sin capacitación previa. Muestran el flujo de procesamiento desde que se introducen los datos al sistema hasta cuando salen del mismo. Las perspectivas que cubre son la funcional, comportamiento e informacional.
- Diagramas IDEF0 (KBS, 2000): esta técnica describe el proceso como una serie de actividades (cajas) definidas en términos de sus entradas, salidas, controles y mecanismos (flechas). Es buena para aspectos funcionales, aspectos de comportamiento y de información.
- Diagramas de flujo de trabajo (Yu, L., 1990): muestra el proceso como una serie de elipses conectadas. En cada elipse se representan los elementos de comunicación (interacción) entre las personas. Las perspectivas que presenta son la funcional, comportamiento y resaltando más la parte informacional.

- Diagramas Rol Actividad (RAD) (Ould, M., 1996): es una de las técnicas más completas, fácil de entender y de utilizar (Miers, D., 199). Es una técnica diagramática pura basada en el modelo de un rol (Warboys, B., 1998). La representación del proceso se realiza desde el punto de vista de roles llevando a cabo actividades e interactuando para completar una tarea. Esta técnica cubre casi todas las perspectivas propuestas por Curtis, excepto por la parte informacional que no es tan fuerte en este sentido.

Para que el modelo de un proceso sea considerado completo debe cumplir con todas las perspectivas descritas en la sección anterior. En la Tabla III se muestra el soporte que brindan las técnicas diagramáticas mencionadas anteriormente, donde se observan las características que proporcionan al modelo de un proceso organizacional (Miers, D., 1996).

Tabla III. Comparación de las técnicas diagramáticas de flujo de datos, RAD, IDEF0 y flujo de trabajo.

	Flujo de Datos	RAD	IDEF0	Flujo de Trabajo
Actividad	■	■	■	■
Orden	■	■	■	■
Decisiones	■	■	■	■
Roles	-	■	-	■
Responsabilidad en las decisiones	-	■	-	■
Interacciones	-	■	-	■
Disparador de eventos	-	■	■	■
Metas	-	■	-	■
Reglas de la organización	?	■	■	?
Fuente y destino de los datos	?	-	■	-
Movimiento de datos	■	?	■	?

Observando la Tabla III, se puede notar que los diagramas de flujo de datos únicamente proporcionan información sobre actividades, orden de las mismas, decisiones y el movimiento de datos (perspectiva funcional e informacional). Sin embargo, no muestran quienes llevan a cabo las actividades, cuales son las interacciones existentes, las responsabilidades, en que momento se realizan las acciones, etc., es decir, las perspectivas de organización y de comportamiento son débiles en estos diagramas. Esto es una restricción muy grande al momento de querer utilizarlos para crear sistemas de coordinación debido a que no es posible asignar responsabilidades a partir de la información que proporcionan.

Los diagramas de flujo de trabajo definen claramente la perspectiva funcional mostrando las actividades que se deben realizar, la organizacional especificando donde y quien realizará las actividades y un poco la informacional. Estos diagramas tienen el inconveniente de no mostrar con claridad las reglas que debe seguir una organización para realizar las actividades y alcanzar los objetivos de un proceso (perspectiva de comportamiento).

Por otra parte, algunas de las ventajas de los diagramas IDEF0 son que el flujo de información está bien definido, los documentos involucrados en cada una de las actividades de un proceso están bien identificados, el momento en que se deben realizar, entre otras. Se puede decir que los puntos fuertes de estos diagramas son la parte informacional, funcional y de comportamiento. La desventaja se presenta en la parte organizacional, al no proporcionar ninguna información de los roles involucrados, sus interacciones, así como

tampoco la meta que se desea alcanzar. Lo anterior representa un problema, como se vio en los diagramas de flujo de datos, ya que no es fácil desarrollar un sistema de coordinación porque se desconocen completamente los roles involucrados y sus respectivas responsabilidades.

Anteriormente se mencionó, que el modelo de un proceso se consideraba completo cuando al menos cumplía con las perspectivas propuestas por Curtis. Los RADs sobresalen de las otras técnicas porque reflejan los aspectos más importantes que se presentan en el *enactment* de un proceso como son: la organización (roles, interacciones, localidades para almacenar datos, etc.), el comportamiento (secuencia de actividades, iteraciones, decisiones, etc.) y la funcionalidad (datos, artefactos, productos). La parte débil es en el aspecto informacional, es decir, no muestra con claridad cuáles entidades de información que se producen ni la estructura que éstas tienen. Sin embargo, esta información a más detalle puede obtenerse por medio de otras técnicas como son los diccionarios de datos, los diagramas de flujo de datos, entre otras, perfilando al RAD como la más apta para ser utilizada en una arquitectura de coordinación.

II.4 Diagramas Rol Actividad (RAD)

El RAD es una técnica de modelado que captura los aspectos más relevantes de un proceso: actividades, la secuencia de actividades, decisiones, roles, responsabilidades, interacciones, metas y reglas (Miers, D., 1996). Estos diagramas representan las tareas, objetivos y personas involucradas en un proceso a manera de roles, actividades e interacciones (Rojas Iñiguez y Martínez García, 1998), considerando muchos de los aspectos de las perspectivas

propuestas por Curtis: funcional, ¿qué actividades están siendo ejecutadas?; organizacional, ¿dónde y quien las ejecuta? y algunos aspectos informacionales dependiendo del modelo, es decir, las unidades de información producidas y manipuladas por un proceso como parte de su información funcional.

En la Figura 5 se muestran algunos elementos de la notación gráfica de los RADs (Warboys, B. *et al.*, 1998) y enseguida se describen a más detalle.

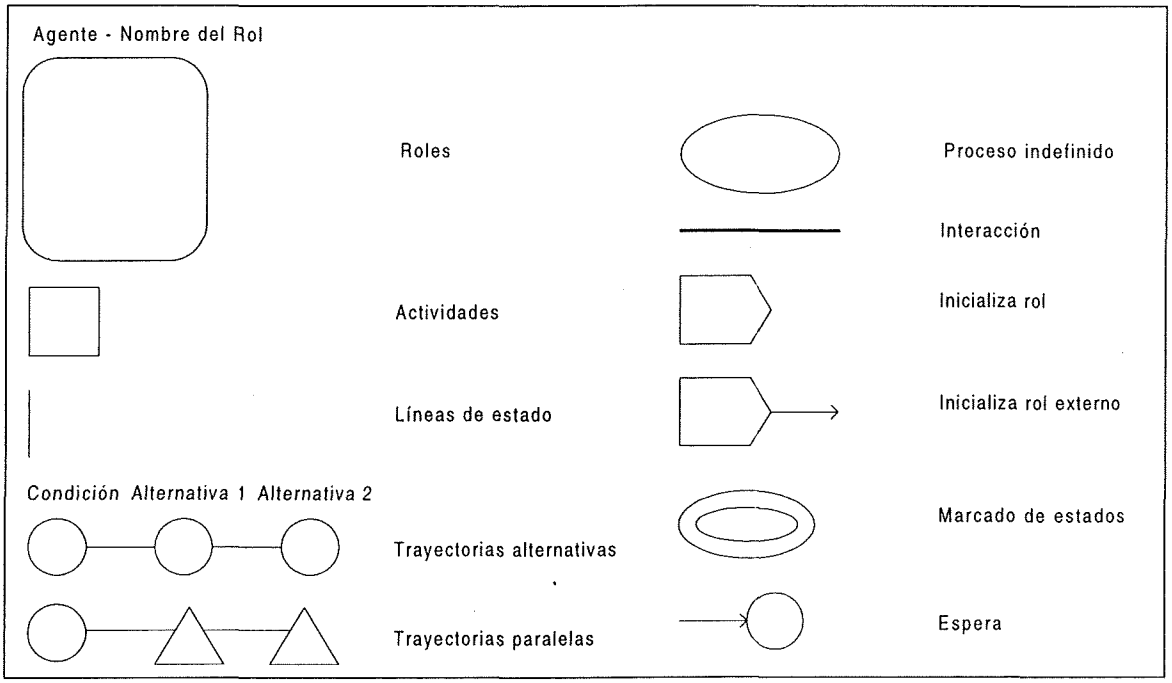

Figura 5. Algunos elementos de la notación gráfica de los RADs.

Un *rol* se representa por un rectángulo grande. En la parte superior izquierda de la caja se encuentra una etiqueta con el nombre del agente y el rol que está desempeñando (agente – rol). Las cajas pequeñas dentro de un rol representan las *actividades* (elementos del proceso) que son los pasos básicos y atómicos del trabajo que realizan las personas. Las

líneas de estado conectan una actividad a la siguiente describiendo cómo están ordenadas las mismas.

Las líneas de interacción se utilizan para representar los elementos de comunicación entre los roles. La notación es una línea que conecta la actividad de un rol con la de otro. Las *decisiones* se representan por medio de opciones que son condiciones bajo las cuales toman lugar diferentes actividades. La ruta de inicio se identifica por un circulo pequeño con una condición, seguida por rutas alternativas o por rutas paralelas. En el caso de las *rutas alternas*, los hilos de actividades se seleccionan de acuerdo a la satisfacción de la condición especificada en el símbolo inicial. Por otro lado, las *rutas paralelas* se ejecutan simultáneamente no importando el orden, pero deben terminarse antes de iniciar la siguiente actividad en el hilo principal.

Para indicar que una actividad o un conjunto de éstas no está definida, es decir, que todavía no se determina la forma en la que ésta se realiza, se utiliza una *elipse*. Otro elemento es el símbolo *esperar*, que se utiliza cuando se introducen requerimientos por eventos o entradas externas necesarios para continuar con la siguiente actividad. Su representación gráfica es una flecha apuntando a un círculo.

El elemento *inicia rol* indica la inicialización de un rol dentro de otro. El elemento *inicializa un rol externo*, inicializa un rol que no está siendo modelado. Por último, el *marcador de estado* indica un estado específico (por ejemplo: inicio, fin).

Estos componentes son los más importantes en la representación gráfica de los RADs para el mapeo de la mayoría de los procesos en las organizaciones (Rojas Iñiguez, 1998).

En la Figura 6 se muestra un ejemplo sencillo de un diagrama RAD para el subproceso *otorgar cita* del caso de estudio realizado en el proceso de atención médica de la Unidad de Medicina Familiar No. 32 del IMSS de Ensenada, B.C.

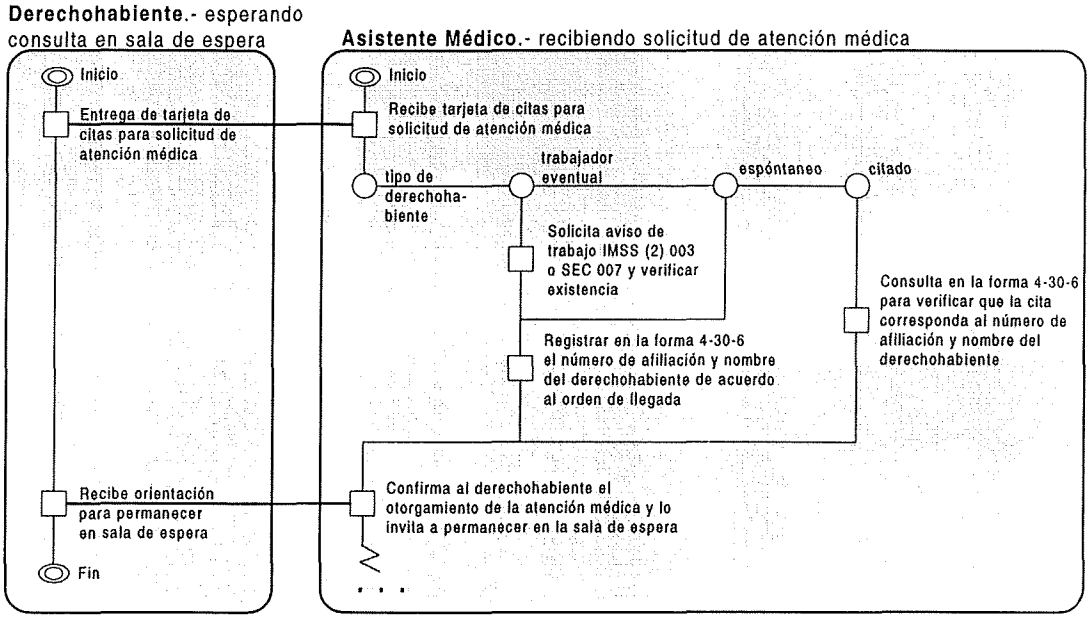

Figura 6. Diagrama RAD que muestra los elementos y responsabilidades del proceso *otorgar cita*.

Éste consta de dos roles: *esperando consulta en sala de espera* y *recibiendo solicitud de atención médica*. Éstos son ejecutados por dos agentes: un "derechohabiente" y una "asistente médica" respectivamente. Se pueden observar en el diagrama las actividades asignadas a ambos agentes y la interacción que tienen cuando el "derechohabiente" entrega su tarjeta de citas a la "asistente médica", ésta la recibe, verifica el tipo de derechohabiente que hace la solicitud y realiza las actividades necesarias para otorgar la consulta informando al "derechohabiente" la hora de la cita y el consultorio asignado para que permanezca en la sala de espera. En las actividades del rol recibiendo solicitud de atención médica se puede observar un elemento condicional cuando se verifica el tipo de derechohabiente.

La captura y modelado de procesos son las primeras etapas de la Ingeniería de Procesos. Una vez que se cuenta con el modelo, éste se valida para posteriormente llevar a cabo el análisis del proceso con el fin de encontrar áreas problemáticas, si existen, y proponer mejoras sobre la base de los resultados encontrados, esto conduciendo a un posible rediseño del modelo del proceso. Una vez modificado se procede a la fase de soporte, por medio de TI se elige la tecnología más apropiada de acuerdo al proceso bajo estudio y las características de la organización.

Hoy en día, existe una gran variedad de TI para el soporte a procesos, entre éstas podemos mencionar: bases de datos, sistemas de información, sistemas para la toma de decisiones, simulación, sistemas basados en conocimiento, sistemas cooperativos, aplicaciones en Internet, sistemas de coordinación, entre otras. En este trabajo nos concentraremos en los sistemas de coordinación para el *enactment* de procesos. En las siguientes secciones se discutirán estos aspectos a más detalle.

II.5 Enactment de procesos

Una de las definiciones del *enactment de procesos* dice que es la ejecución simultánea y sincronizada de un proceso orientado al humano y un modelo ejecutable del mismo, con la finalidad de mejorar el soporte que brinda la computadora al hombre (Fernström, C. y Lennart, F., 1991).

En la ejecución de los procesos, los modelos se utilizan principalmente como anteproyectos de sistemas para el desarrollo de un sistema computarizado dirigido a automatizar parte del flujo de información y el procesamiento. Expresando el modelo de un proceso en términos

de un programa, éste puede ser ejecutado ya sea para emular el comportamiento del sistema modelado o para dar soporte al proceso real. Es decir, el *enactment* de procesos captura su modelo a un lenguaje de programación o herramienta para ejecutarlo posteriormente como un sistema. De aquí que la funcionalidad del sistema proporcione el soporte necesario a los usuarios y su organización (Fernström, C. y Lennart, F., 1991).

El concepto de *enactment* de procesos es muy poderoso, debido a que provee un ambiente para coordinar a todos los usuarios participantes y las herramientas de software involucradas en el proceso a través de diferentes máquinas y bajo diferentes plataformas (Yeomans, B., 1996).

Cuando se termina una actividad, se llevan a cabo automáticamente acciones específicas como notificar a otros miembros del equipo o distribuir los resultados.

Crear el modelo de un proceso cuyo estado actual lo proporciona el sistema, permite al administrador obtener una visión completa del estado de la organización en tiempo real.

II.6 Sistemas de coordinación

El uso creativo de TI (ej. redes de computadoras como medios de propósito general para apoyar el trabajo en grupo) mejora de manera significativa los procesos coordinados. Si utilizamos redes de computadoras para apoyar la coordinación de procesos, los usuarios podrán decidir qué trabajo van a realizar en la red, cuál fuera de ésta y cómo relacionar ambos. El administrador del proceso sería capaz de utilizar sus habilidades organizacionales sin requerir expertos en computadoras para programar sus actividades (Roberts, C., 1993).

De acuerdo con Roberts los *sistemas de coordinación* son aquellos donde los usuarios pueden distribuir sus materiales de trabajo y responsabilidades, establecer acuerdos de cooperación y terminar sus tareas de manera guiada para asegurar que las personas adecuadas realizan las actividades que les corresponden. Por lo tanto, los usuarios de un sistema de coordinación pueden determinar fácilmente el estado del proceso y de todos los subprocesos en los cuales están involucrados. Lo más importante es que los usuarios pueden:

- Determinar donde realizar su trabajo si fuera o dentro de la red.
- Utilizar una metáfora de oficina para dividir el trabajo en actividades de grupo. La metáfora de la oficina permite a los usuarios utilizar sus computadoras teniendo como base el conocimiento actual de su trabajo, independientemente de su experiencia con las computadoras.

A continuación se presentan tres aspectos que se deben considerar para llevar a cabo la coordinación de procesos organizacionales (Roberts, C., 1993):

- *Coordinación: actividades humanas organizadas.* Cada una de las operaciones comerciales de una compañía consiste de procesos organizacionales. Para alcanzar sus metas y objetivos, la compañía debe organizar sus recursos y administrar sus procesos. Pero no es el esfuerzo individual lo que permite a la compañía terminar su trabajo; por el contrario, las metas de la compañía se alcanzan a través de los esfuerzos coordinados de muchos trabajadores organizados.

Las actividades grupales son de una coordinación intensa, esto es, involucran una gran cantidad de esfuerzo del cual una buena parte no se ejecuta conscientemente. Cada individuo gasta energía en la coordinación, toma tiempo, cuesta dinero el espacio físico que se ocupa y los materiales de apoyo que se utilizan.

- *Lado físico de la coordinación.* Existe un espacio identificable para cada rol en un equipo. Las funciones de un rol están restringidas por el ambiente físico. La actividad humana organizada es simplemente la expresión de las personas determinando su propósito y objetivos, organizándose de acuerdo a los recursos disponibles. La coordinación es el denominador común para todas las actividades organizadas.

- *El problema del desarrollo de aplicaciones.* Las computadoras en red pueden utilizarse: (i) como herramientas para el análisis y diseño de patrones de trabajo, (ii) como un conducto para apoyar las interacciones y (iii) el flujo de material entre los miembros del grupo de trabajo. Para tomar ventaja de esto, los usuarios de computadoras deben estar facultados para organizarse así mismos, a sus grupos y sus espacios de trabajo (tanto electrónicos como físicos).

Sin embargo, cuando se realizan aplicaciones para grupos de trabajo puede suceder que las relaciones entre los miembros cambien de tal manera que la aplicación desarrollada ya no sea de utilidad al grupo. Por lo tanto, las aplicaciones de cómputo deben construirse

reconociendo las interdependencias funcionales entre los miembros del grupo: ¿quién es responsable de lograr qué?, ¿qué materiales y actividades están involucradas?, ¿cómo se relaciona e interactúa cada rol con los roles de otros miembros del grupo de trabajo?. Esta dependencia funcional entre los miembros proporciona una guía vital de cómo se debe proceder (Roberts, C., 1993).

El paradigma básico de la tecnología de coordinación es que la computadora realice cálculos o manipulación de datos a partir de la solicitud de un individuo. Sin embargo, estas actividades individuales se ejecutan en el contexto de procesos organizacionales y éstos procesos no son ejecutados por individuos actuando solos sino por individuos trabajando en equipo (Greenwood, R.M. *et al.*, 1998).

Las nuevas tecnologías están emergiendo debido a la interconexión de computadoras y al abaratamiento de los costos que permiten a más personas y organizaciones contar con equipos de cómputo. Por ello, se explota la TI como una manera de conectar a las personas. El correo electrónico, el trabajo cooperativo asistido por computadora (CSCW por sus siglas en inglés) y el flujo de trabajo se enfocan en los beneficios obtenidos al proporcionar TI a grupos en lugar de a individuos.

II.7 Tecnología de información

Al desarrollar TI para apoyar a grupos de trabajo, una parte de la comunicación está implícita en el sistema. Los sistemas de flujo de trabajo son un buen ejemplo. El modelo general para esto es reemplazar la forma como se maneja la información. Por ejemplo, si

tenemos un sobre con documentos que debe enviarse a una serie de personas dentro de otro departamento en una organización. El sobre pasa de oficina en oficina por toda la organización hasta llegar a su destino. El reemplazo sería por ejemplo, utilizar una forma electrónica que fluya de un correo electrónico a otro. El sistema de flujo de trabajo conoce el "procedimiento" que la forma debe seguir y puede redirigirla a su próximo destino.

Las organizaciones a partir de los años 90's comenzaron una ola de cambios radicales para alcanzar mejores resultados en la ejecución de sus procesos. Este concepto se popularizó como "reingeniería", definiéndolo como un nuevo enfoque y un rediseño radical de los procesos de la organización para alcanzar mejoras significativas en su desempeño (Hammer, M. y Champy, J., 1993). Un aspecto central a estos esfuerzos es la TI como uno de los principales facilitadores del cambio en los procesos (Chan, P. y Land, C., 1999). Es decir, la TI con frecuencia es uno de los principales catalizadores que permiten a una compañía alcanzar sus objetivos organizacionales.

La Figura 7 muestra una posible infraestructura de TI a utilizar en las organizaciones, la cual consiste de sistemas de coordinación, interfaces de usuario (computadoras), aplicaciones de apoyo para la ejecución de actividades y medios de almacenamiento como bases de datos.

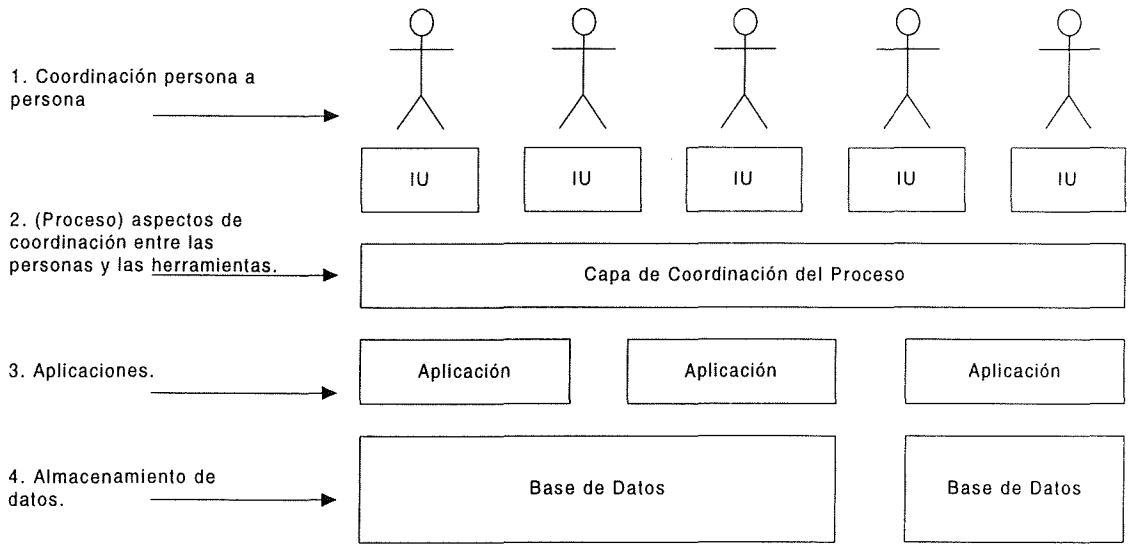

Figura 7. Infraestructura de tecnología de información para las organizaciones (Greenwood, R.M. *et al.*, 1998).

La TI se manifiesta de diferentes maneras: procesa datos, obtiene información, almacena materiales, acumula conocimiento y permite la comunicación. De hecho, juega un rol en la mayoría de las operaciones diarias de los negocios que están alrededor del mundo. Chan (2000) propone y expone tres roles de la TI: iniciador, facilitador y habilitador. Puede asumir cualquiera de los tres dependiendo del ámbito de la organización y de cómo se aplica la tecnología. Como iniciador actúa como un agente de cambio permitiendo a las personas reconocer una solución viable antes de encontrar el problema que puede resolver. Como facilitador puede servir para hacer el trabajo de las personas más fácil. Y como habilitador ha recibido la mayor atención. El habilitador provee procesamiento rápido y capacidades analíticas, acceso paralelo y captura de información. Es decir, trabajar con una productividad mayor.

II.8 Conclusiones

En este capítulo se definen algunos conceptos básicos con el propósito de conocer la terminología en el contexto de la Ingeniería y Modelado de Procesos, y los diferentes aspectos que involucran. Se ha discutido la selección de los diagramas RAD para el modelado de procesos y su utilización como base en la generación de sistemas de coordinación. Finalmente, se explicó la necesidad de contar con este tipo de sistemas para dar soporte a las organizaciones, proporcionando el apoyo necesario para ejecutar un proceso de manera más eficiente.

En el siguiente capítulo se presenta un análisis comparativo entre distintas arquitecturas de coordinación para identificar las características más elementales que debe tener toda arquitectura.

Capítulo III. Arquitecturas de coordinación: Análisis comparativo

III.1 Introducción

En este capítulo se presentan diferentes arquitecturas que facilitan la generación de sistemas para la coordinación de procesos organizacionales. Hoy en día, muchas compañías y universidades han desarrollado sistemas para apoyar la ejecución automatizada de actividades en las organizaciones, cada una desde diferentes perspectivas. A continuación, se presenta una breve descripción de algunas arquitecturas destacando sus características principales y realizando un análisis comparativo entre éstas. Estas características son las más relevantes en una arquitectura que facilitan la creación de sistemas de soporte. Además, se definen los requerimientos que se deben satisfacer en este trabajo.

III.2 Sistema ProcessWeb

El sistema ProcessWeb surge como la necesidad de unir dos aplicaciones diferentes para dar soporte a procesos organizacionales: el *Process Wise Integrator* (PWI) y el *World Wide Web* (WWW). El PWI es un sistema de *enactment* de propósito general (Yeomans, B., 1996). Los sistemas de coordinación (programas) de procesos se definen utilizando un lenguaje de alto nivel llamado lenguaje para la gestión de procesos (PML por sus siglas en ingles). Éste se carga dentro de un administrador para el control de procesos (PCM por sus siglas en inglés). El PCM es el responsable de proporcionar y mantener el contexto para cada rol en el modelo, y calendarizar sus actividades. También coordina las interacciones con los usuarios y las aplicaciones de software. En esencia, el programa de un proceso le

dice al usuario cuáles opciones están disponibles en un momento determinado y coordina el trabajo del equipo al cual pertenece el usuario. La ejecución de una actividad puede provocar que el programa de un proceso envíe actividades a otros miembros del equipo proporcionando de esta manera soporte a procesos colaborativos. En general, el programa de un proceso tiene control de su progreso, no es reversible y solicita información al usuario conforme la va necesitando. En el sistema ProcessWeb se agregó el *Web* al PWI utilizando la interfaz de puerta de enlace común (CGI de sus siglas en inglés). Esto permitió que las actividades realizadas en el WWW invocaran a los programas de procesos del PWI. En este sentido, también los programas pueden enviar un número arbitrario de páginas virtuales HTML de regreso al *Web*. Debido a que estas páginas son virtuales, solo pueden ser accesadas a través del programa del proceso. Claramente, el orden en que estas páginas se pueden leer está controlado por el programa. Los usuarios exploran estas páginas de manera normal en un navegador y seleccionan la actividad a realizar. De esta forma, el usuario elige las acciones y el proceso responde a las mismas. En la Figura 8 se muestra la arquitectura del ProcessWeb, en la que se observa el sistema ProcessWise en el nivel inferior y en los superiores, los componentes que permitirán que un rol pueda ejecutar sus actividades y coordinar sus interacciones.

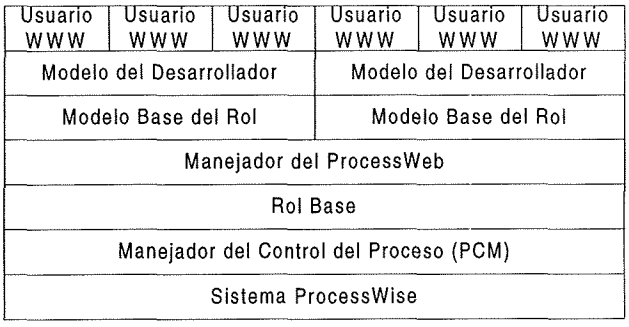

Figura 8. Componentes de la arquitectura del ProcessWeb (Yeomans, B., 1996).

III.3 Sistema de flujo de trabajo(Workflow): Panta Rhei

El sistema Panta Rhei (Groiss, H., y Eder, J., 2001) utiliza el WWW para interactuar con los usuarios, con otros sistemas de flujo de trabajo y con otras aplicaciones. Dentro de una organización, la coordinación entre las tareas se logra mediante el envío de formas que contienen los datos necesarios para ejecutar una tarea. La recepción de un documento inicia o continúa un proceso del lado del receptor y el envío de un documento a otra empresa es el resultado (parcial) de un proceso. El sistema Panta Rhei está integrado completamente a la *Web*, puede enviar y recibir formas, y cada interacción del usuario se hace a través de un navegador de la WWW. Los procesos se definen utilizando el Lenguaje de Definición del Flujo de Trabajo (WDL por sus siglas en inglés).

Por ejemplo, después del encabezado del programa de un proceso se definen algunas de sus propiedades como: el dueño del proceso, la descripción del mismo, el tiempo máximo para ejecutarlo y la tarea a realizar. También se define un contenedor de datos para cada proceso mediante una forma. Después de la palabra clave *"begin"*, se define la estructura del proceso. La ejecución de una tarea se especifica proporcionando el nombre o identificador del usuario seguido por el nombre de la tarea y las formas necesarias para llevarla a cabo.

Durante la ejecución, la máquina de flujo de trabajo interpreta la descripción del proceso, inicia las tareas y asigna los usuarios. Cuando se termina una tarea, el sistema inicia con la siguiente. La tarea por sí misma se ve como una caja negra, donde un usuario o programa puede hacer algunas manipulaciones sobre las formas y obtener un resultado como consecuencia. La arquitectura de este sistema consta de tres partes como se muestra en la Figura 9:

- El *servidor HTTP* como interfaz entre los usuarios y la máquina de flujo de trabajo.
- La *máquina de flujo de trabajo* que es una colección de procedimientos invocados por medio del servidor HTTP.
- El *sistema administrador de la base de datos* (DBMS) que contiene toda la información relevante para la ejecución de los procesos.

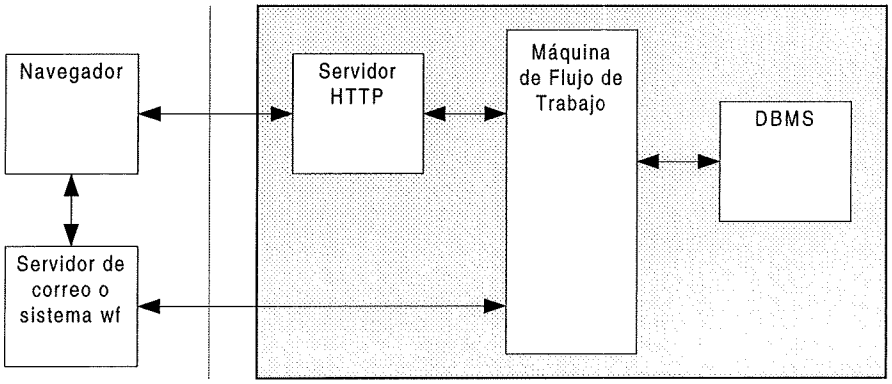

Figura 9. Arquitectura del sistema Panta Rhei (Groiss, H. y Eder, J., 2001).

III.4 ORBWork: Un sistema de enactment para METEOR$_2$

Las técnicas de manejo de flujo de trabajo desarrolladas en el proyecto METEOR$_2$ están orientadas a dar soporte confiable a la coordinación de usuarios y tareas automatizadas en ambientes heterogéneos de computación para empresas del mundo real. Las capacidades claves del sistema de manejo del flujo de trabajo (WFMS por sus siglas en inglés) METEOR$_2$, incluyen un conjunto de herramientas para construir estos flujos (mapas/datos/diseño de tareas), dando soporte al modelado de procesos de alto nivel, a la especificación detallada del flujo y a la generación automática de código para el sistema de *enactment*: WEBWork y ORBWork (Das, S. *et al.*, 2001).

La arquitectura de METEOR$_2$ consta actualmente de un diseñador y dos sistemas de *enactment* del flujo de trabajo (WEBWork basado en Web y ORBWork basado en CORBA) incluyendo los respectivos generadores de código en tiempo de ejecución.
El diseñador de METEOR$_2$ es una interfaz gráfica de usuario utilizada para especificar tanto el mapa completo del flujo de trabajo, los objetos de datos manipulados por éste y la información detallada para invocar las tareas.

La especificación mencionada anteriormente, se almacena en un formato intermedio llamado Lenguaje Intermedio del Flujo de Trabajo (WIL por sus siglas en inglés). WIL es similar en estructura y semántica al Lenguaje de Definición de Procesos de Flujo de Trabajo (WPDL por sus siglas en inglés) y soporta la mayoría de los aspectos de sus lenguajes intermedios: Lenguaje de Especificación del Flujo de Trabajo (WSL ó Workflow Specification Language) y Lenguaje de Especificación de Tareas (TSL ó Task Specification Language). La especificación de WIL incluye todas las dependencias predecesor-sucesor entre las tareas, así como los objetos de datos que fluyen entre éstas.

La funcionalidad del diseñador de METEOR$_2$ permite una generación, casi completa, de código para una aplicación de flujo de trabajo. El generador de código es un módulo que toma el lenguaje WIL como entrada y entrega como salida el código de ejecución de una aplicación. Éste genera código para los administradores de tareas incluyendo componentes calendarizadores, el código de invocación de tareas, las rutinas de acceso a los objetos de datos y el mecanismo de recuperación.

El sistema en tiempo de ejecución consiste de varios gestionadores de tareas y sus tareas asociadas, las interfaces de usuario, el mecanismo de recuperación, el calendarizador (distribuido entre los manejadores de tareas) y los diferentes componentes de monitoreo. En la Figura 10 se muestran los distintos módulos de este sistema y sus interacciones.

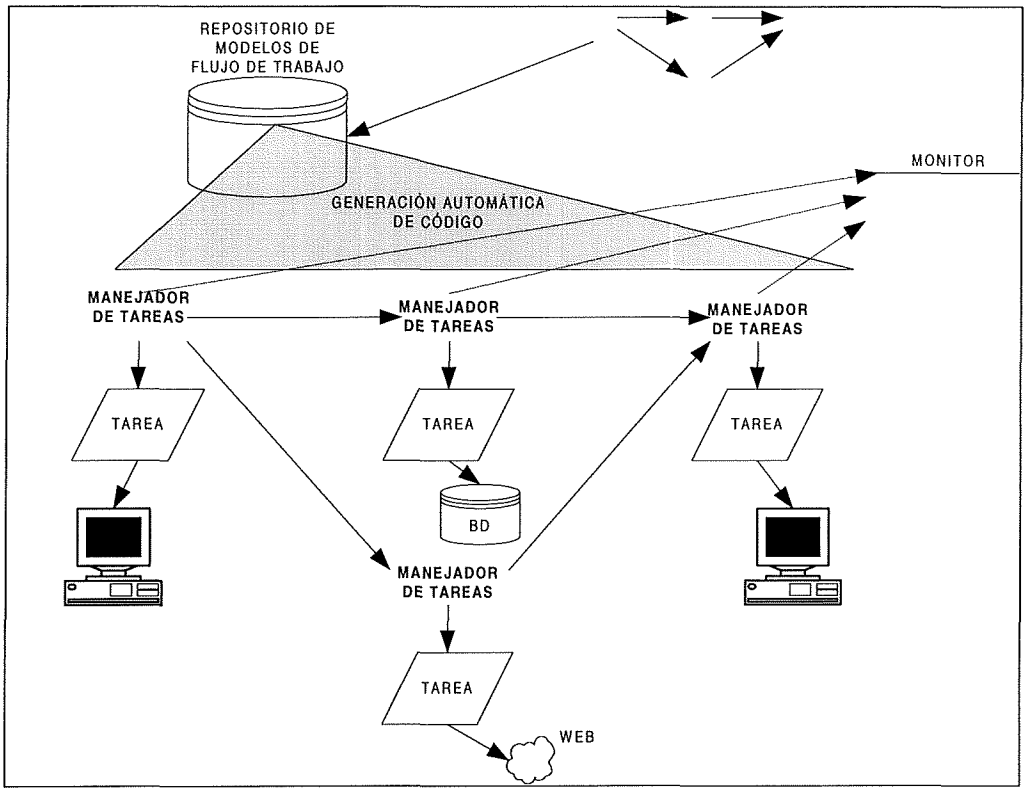

Figura 10. Arquitectura del Sistema METEOR$_2$ (Das, S. *et al.*, 2001).

III.5 Enactment flexible y descentralizado del flujo de trabajo basado en agentes para la coordinación de tareas

Un agente de tareas, aunque es un poco simple, es un agente reactivo poderoso que tiene una arquitectura típica que consta de tres capas: comunicación, decisión y operacional (Joeris, G., 2001). La capa operacional se representa por una clase *Tarea* y contiene todas las operaciones/ transiciones incorporadas, las cuales se pueden categorizar dentro de

transiciones de estados, operaciones de asignación de actores, operaciones para el manejo de entradas y salidas, y operaciones para el cambio del flujo de trabajo.

Para cada operación, el agente de tareas sabe cuándo dispararla, conoce la condición que debe mantenerse para ejecutar la operación y tiene una lista de receptores a los cuales se les pasan los eventos. De esta manera, el conocimiento sobre cómo reaccionar ante los eventos se representa explícitamente en la capa de localización/decisión en términos de reglas específicas evento-condición-acción (ECA).

Esta es la característica básica de una arquitectura de agentes, sin embargo, no son inteligentes ni autónomos, dado que su comportamiento durante la ejecución se deriva del esquema de flujo de trabajo y se comporta exactamente como se definió dentro del mismo. La comunicación entre los agentes de tareas se basa en la primitiva del paso de mensajes. Un evento consiste de su nombre, una referencia al productor del mismo y un conjunto arbitrario de pares (atributo, valor).

Se asume en esta herramienta un diagrama de transición de estados que define el estado fundamental y las transiciones de un agente de tareas. El comportamiento de este agente durante su ejecución se define por medio de las transiciones de estados y la descripción del comportamiento de la transición que determina cuándo se invoca una operación/transición y cuándo es aplicable.

III.6 Proyecto IPSE 2.5

IPSE (Integrated Project Support Enviroment) (Warboys, B., 1989) está interesado en problemas sobre cómo pueden utilizarse los sistemas de cómputo para el desarrollo de software basado en sistemas de información. Este proyecto busca una integración eficiente de los usuarios de IPSE y sus procesos, con un conjunto de herramientas de soporte. Específicamente IPSE 2.5 es un proyecto cuyo objetivo es desarrollar y demostrar un pequeño número de IPSEs, cada uno construido sobre la base de características particulares utilizando una herramienta IPSE genérica.

El propósito de este ambiente de soporte es proporcionar los medios por los cuales los procesos de desarrollo, mantenimiento, soporte y mejora de los sistemas de información sean más eficientes, en términos cualitativos y productivos. En la parte medular del sistema IPSE 2.5 se encuentra una Máquina de Control de Procesos (PCE por sus siglas en inglés). La razón del PCE es que sea un sistema de cómputo que proporcione los ambientes de trabajo para sus usuarios, es decir, las personas involucradas en el proceso de desarrollo. El PCE está consciente del proceso mismo, y por ello, es capaz de proporcionar los ambientes de trabajo apropiados en el momento adecuado. Es necesario proveer los medios por los cuales un PCE de propósito general pueda ser programado para dar soporte a diferentes procesos.

En este sistema se introduce el concepto de Lenguaje de Gestión de Procesos (PML por sus siglas en inglés) que será el medio por el cual se puedan describir y componer los fragmentos de un proceso. Estos fragmentos también proporcionan los medios para acceder

a las herramientas externas de trabajo. Los ambientes de trabajo que serán proporcionados por el PCE a sus usuarios deben incluir todas los elementos que éstos necesitan para realizar sus actividades. El PML debe entonces informar las descripciones de los objetos sobre los cuales los usuarios ejecutan algunas operaciones, las herramientas que los ayuden a realizar esas operaciones y los medios para comunicar los cambios en los ambientes de trabajo de él mismo o de otros. La notación de roles e interacciones es la base de los modelos de procesos. La esencia de dichos modelos es que consisten de un conjunto de roles que interactúan de manera significativa. Este grupo de roles está dirigido a explotar la idea de ejecución concurrente de agentes, todos coordinados para alcanzar una meta común.

III.7 Sistema administrador del flujo de trabajo: CodAlf

El sistema CodAlf (Code Name Alfa) (Schill, A. y Mittasch, C., 1996) está basado en una arquitectura descentralizada. Todos sus componentes son modelados y se implementan como objetos de un sistema distribuido sobre DC++, que es una extensión orientada a objetos del ambiente de computación distribuida (DCE por sus siglas en inglés). Este sistema contempla los siguientes aspectos:

- Arquitectura descentralizada: no depende de un servidor o base de datos centralizada.
- Mapeo dinámico: da soporte al mapeo dinámico de tareas y a los roles asociados en las instancias de ejecución.
- Control: se tiene un control lógico centralizado para la ejecución del flujo de trabajo.
- Extensibilidad: se pueden extender las instancias de ejecución, los tipos de flujo de trabajo y las instancias de estos flujos.

- Heterogéneo: debido a que los flujos de trabajo se ejecutan típicamente en organizaciones heterogéneas, se da soporte a este aspecto.

La fase de diseño del flujo de trabajo se realiza utilizando un editor que tiene una interfaz gráfica de usuario y está basado en la descripción del grafo de ejecución a través de redes de *Petri* incluyendo pruebas sintácticas y semánticas. Estas pruebas abarcan la corrección de la estructura del grafo (grafo dirigido con un nodo de inicio y un nodo final), corrección de los rangos y tipos de atributos, comparación sintáctica de los tipos de datos, entre otros.

El componente del sistema en tiempo de ejecución está instalado en cualquier nodo participante e involucra los servidores de aplicación. Todas las instancias del flujo de trabajo están modeladas como objetos movibles basados en DC++ y se desplazan a través del sistema distribuido durante la ejecución del flujo de trabajo. En la Figura 11 se muestran los componentes del sistema CodAlf.

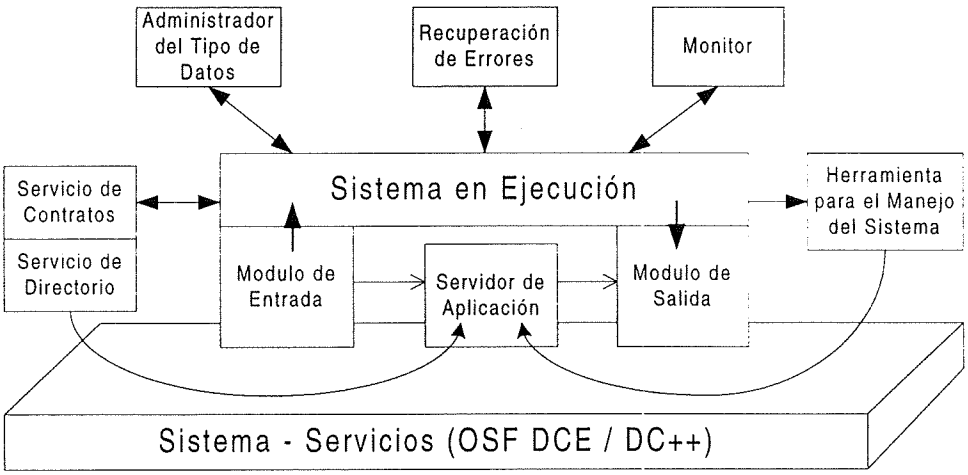

Figura 11. Componentes del Sistema CodAlf.

III.8 Sistema administrador del flujo de trabajo: Ultimus

Ultimus (Ultimus, 2001) es un sistema robusto basado en *Web* para la automatización del flujo de trabajo. Ultimus apoya a las compañías en el desarrollo de aplicaciones sofisticadas para el *Web* en forma rápida, sin nada de programación ni macros. Debido a que este sistema cuenta con "Internet" o "Intranets" como medio de transporte primario, la aplicación puede ser escalable en términos del número de usuarios y la localización de los mismos. Este sistema ofrece los siguientes beneficios:

- Diseño gráfico del proceso: permite a los usuarios diseñar gráficamente los procesos sin la necesidad de programar.
- Comercio electrónico/EDI por medio de XML: proporciona soporte robusto para XML permitiendo a las aplicaciones integrarse a otras sin mucha complicación.
- Ramificación condicional y enrutamiento: permite a los procesos cambiar dependiendo de las entradas de usuario y las condiciones en tiempo real para manejar las excepciones y las condiciones especiales que se presentan en cada organización.
- Seguimiento del proceso y métricas: permite a los usuarios monitorear el estado de los procesos y obtener métricas para su análisis detallado. Este aspecto ayuda a las organizaciones a mejorar y optimizar sus procesos.
- *Flobots*: permiten a un proceso interactuar con otras aplicaciones como bases de datos, procesadores de palabras, entre otras.
- Procesamiento anidado: un proceso puede iniciar a otro, produciendo la automatización de procesos complejos en más simples.

- Simulación: habilita a los diseñadores de procesos para simular completamente el comportamiento de un proceso antes de adoptarse. Esto permite a los usuarios probarlo y hacer las correcciones antes de utilizarlo.
- Conectividad a bases de datos: conectividad del lado del servidor robusta y segura con bases de datos empresariales donde las organizaciones almacenan información importante utilizada por sus procesos.

Hasta el momento se han descrito algunas arquitecturas de coordinación mostrando parte de su funcionalidad, además de los recursos y mecanismos utilizados en su construcción.
En la siguiente sección se presenta el análisis comparativo de estas arquitecturas con respecto a la que se propone en este trabajo, con la finalidad de observar las diferencias entre éstas y los aspectos que involucran para que una herramienta de este tipo pueda brindar soporte a los procesos.

III.9 Análisis comparativo

Una arquitectura de coordinación debe considerar ciertos elementos básicos para apoyar la creación de sistemas que den soporte a las organizaciones y sus procesos por medio de equipo de cómputo, redes de comunicación y sistemas. Estos elementos han sido clasificados en tres categorías dependiendo de que tan importante es para una arquitectura contar con un elemento en particular, es decir, si la falta o presencia de éste afecta de manera negativa o positiva la funcionalidad de la arquitectura. Este grado se ha establecido de acuerdo a que tan fuerte es la necesidad de un elemento para facilitar la generación de sistemas de coordinación funcionales. Se han definido las categorías como sigue:

1) *Nivel 1* (N1), significa que es necesario tener el elemento en la arquitectura.
2) *Nivel 2* (N2), significa que es deseable contar con el elemento, más no es indispensable.
3) *Nivel 3* (N3), significa que es opcional contar con este elemento, es decir, si no se tiene no representa mayor problema.

La Tabla IV muestra una comparación entre las arquitecturas descritas anteriormente. Como se puede observar, todas contemplan los elementos más importantes de un diagrama RAD como son los roles, las actividades o tareas, las metas para el rol, las responsabilidades bien definidas, entre otros. Lo cual nos permite deducir que la selección de este tipo de diagramas para modelar los procesos fue acertada y que el nivel de importancia asignado (N1) a los elementos de un RAD fue el correcto porque representan unidades atómicas e indispensables de un proceso, y por ende, deben ser considerados en una arquitectura para generar sistemas de coordinación.

En las arquitecturas analizadas existe un aspecto importante que hace diferente una de otras. Esto es, el tipo de ambiente en el que se trabaja. Puede considerarse un ambiente centralizado en el que existe un servidor central encargado de recibir y enviar las actividades a los roles de un proceso que está siendo coordinado por un sistema. Este sistema asigna a cada uno de ellos las actividades que deben realizar y los recursos necesarios para llevarlas a cabo. Por otro lado, también puede ser un ambiente distribuido en donde la comunicación e intercambio de información se hace directamente entre cada uno de los agentes participantes en un proceso, es decir, sin un intermediario (servidor central) que se encargue de coordinarlos y comunicarlos. En nuestra arquitectura se

propone el ambiente centralizado utilizando un servidor de *Web* (Apache por ejemplo) por la facilidad que presenta para coordinar y comunicar los roles cuando estos ejecutan sus actividades. Por otra parte, esta tecnología no restringe el ambiente operacional bajo el que se puede encontrar un usuario (rol), ya que la ejecución de sus actividades las hace a través del *Web* utilizando prácticamente cualquier navegador existente en el mercado.

Tabla I. Análisis comparativo de las características básicas de las arquitecturas.

Nivel	Característica	ProcessWeb	Panta Rhei	ORBWork	Flujo de Trabajo con Agentes	IPSE 2.5	CodAlf	Ultimus	Arquitec. Propuesta
1	Persistencia	√	√	√	√	√	√	√	√
1	Comunicación entre Roles	√	√	√	√	√	√	√	√
1	Coordinación entre Roles	√	√	√	√	√	√	√	√
1	Manejo de Estados	√	X	X	√	X	√	X	√
1	Manejo de Información	√	√	√	√	√	√	√	√
1	Controlador de Actividades	√	√	√	√	√	√	√	√
1	Ejecución Asíncrona	√	√	√	√	√	√	√	√
1	Roles	√	√	√	√	√	√	√	√
1	Actividades	√	√	√	√	√	√	√	√
1	Independencia de Plataforma	√	√	√	√	X	√	√	√
1	Orden	√	√	√	√	√	√	√	√
1	Responsabilidad en las Decisiones	√	√	√	√	√	√	√	√
1	Decisiones	√	√	√	√	√	√	√	√
1	Metas	√	√	√	√	√	√	√	√

√ significa que la herramienta cuenta con la característica.

X significa que la herramienta no cuenta con la característica.

Continuación Tabla I. Análisis comparativo de las características básicas de las arquitecturas.

Nivel	Característica	ProcessWeb	Panta Rhei	ORBWork	Flujo de Trabajo con Agentes	IPSE 2.5	CodAlf	Ultimus	Arquitec. Propuesta
2	**Presencia**	√	√	√	√	√	√	√	√
2	**Síncrona**	√	√	√	√	√	√	√	√
2	**Rol Base**	√	X	X	X	√	X	X	√
2	**Tipo de Arquitectura**	Centralizada	Centralizada	Distribuida	Distribuida	Centralizada	Distribuida	Centralizada	Centralizada
2	**Lenguaje de Programación**	PML/HTML	WDL	WIL	Reglas ECA	PML	DC++	X	X
2	**Modelo Reusable**	X	X	X	√	X	X	X	√
2	**Interacción con el Web**	√	√	√	X	X	X	√	√
2	**Integración con otras Aplicaciones**	√	√	√	√	√	√	√	√
2	**Generación Automática del Programa**	X	X	√	√	X	√	√	√
3	**Integración de Herramientas**	√	√	√	X	√	√	√	√

√ significa que la herramienta cuenta con la característica.

X significa que la herramienta no cuenta con la característica.

Por otra parte, algunas de las arquitecturas analizadas utilizan un editor gráfico para representar el flujo de trabajo de un proceso generando a partir de la gráfica, el código de programación respectivo y el sistema de coordinación que dará soporte a un proceso. *Ultimus* es un sistema que no se basa en un lenguaje de programación, lo que hace es definir el flujo de trabajo de un proceso organizacional de manera gráfica. Para ello, utiliza un editor gráfico por medio del cual representa el flujo de actividades y los agentes responsables de llevarlas a cabo. Esto conlleva una ventaja porque facilita la generación de sistemas sin tener que programar código en algún lenguaje. Con nuestra arquitectura no estamos proponiendo un lenguaje de programación para crear los sistemas de coordinación, sino un lenguaje para representar los modelos de procesos involucrando: las actividades, la secuencia de las mismas, los roles, las responsabilidades, interacciones, entre otros elementos.

Otro punto interesante que hace diferente a la arquitectura propuesta con respecto a las otras, es la reusabilidad del modelo base de un proceso que si bien no es una característica necesaria, si es deseable con el propósito de aprovechar la información del modelo para generar otro tipo de sistemas de soporte. De las arquitecturas analizadas solamente la de Flujo de Trabajo con Agentes presenta reusabilidad en su modelo.

En la literatura, se pueden observar diversas formas de interactuar con un sistema. Entre otras el uso de la tecnología WWW por medio del intercambio de información basados en el protocolo HTTP, permitiendo a los usuarios interactuar con el sistema mediante páginas HTML. Por otro lado, a través de interfaces de usuario definidas por los desarrolladores del

sistema, las cuales deben estar instaladas en las computadoras personales para que los usuarios puedan ejecutar sus actividades. Se propone utilizar la tecnología WWW debido a que la mayoría de los usuarios conectados a la red de "Internet" tienen acceso al *Web* de manera distribuida, desde cualquier parte del mundo el usuario puede utilizar el navegador de su preferencia para realizar sus actividades.

Una característica fundamental que debe cumplir una arquitectura es permitir la coordinación de los roles involucrados en un proceso y el manejo de información que siempre es importante para ejecutar una actividad. Como se observa en la Tabla IV estas características tienen un nivel N1.

Las arquitecturas generalmente tienen la facilidad de comunicarse con otras aplicaciones para que los usuarios cuenten con más recursos para ejecutar sus actividades. A esta característica se le otorgó un grado de importancia N2 ya que no es indispensable contar con esta funcionalidad para que la arquitectura pueda proporcionar el soporte necesario para coordinar un proceso, sin embargo, puede facilitar el trabajo de los roles. Por esta razón, se contempla esta posibilidad en la arquitectura propuesta.

El concepto de estados no es muy utilizado para coordinar el comportamiento de los roles en un proceso. La propuesta considera este aspecto para facilitar el flujo de las actividades a cada uno de los roles y la coordinación de los mismos. Del análsis se concluye que se contemplan las necesidades básicas de una arquitectura de coordinación con la finalidad de proporcionar un soporte adecuado a los procesos organizacionales, con el mínimo esfuerzo.

Capítulo IV. Prototipo: Enactment del juego de la cerveza

IV.1 Introducción

Con la finalidad de identificar los requerimientos y necesidades de los sistemas de coordinación se realizó un prototipo basado en el modelo del "Juego de la Cerveza" desarrollado por el Instituto Tecnológico de Massachusetts (MIT) (Forrester, J., 1961). La creación de prototipos rápidos permiten identificar entre otras cosas, los requerimientos tecnológicos que se tienen para realizar un sistema de coordinación. La razón por la que se seleccionó este modelo en particular fue lo interesante del proceso, debido a que aporta requerimientos conceptuales atractivos y presenta aspectos relevantes para la generación de sistemas de coordinación como: la iteración de actividades permitiendo a un rol repetir una o más veces una actividad; puntos de decisión, en donde los roles pueden elegir que tarea realizar dependiendo de las circunstancias; intercambio de información; un rol realiza las actividades de los otros dos en un momento determinado, y por último, en el transcurso del proceso un rol depende de otro para continuar con sus actividades.

IV.2 Definición del proceso

El modelo del "Juego de la Cerveza" captura el problema de un expendio de cerveza que debe satisfacer la demanda de sus clientes cada vez que le solicitan productos y a la vez mantener un nivel de inventario apropiado para no incurrir en gastos de almacenamiento. El expendio tiene que verificar constantemente la cantidad de cerveza existente en el almacén, de tal manera que si es baja, solicita este producto a la compañía distribuidora de cerveza

para satisfacer la demanda de sus clientes, siempre cuidando que la cantidad no sea demasiada por los costos de almacenamiento asociados y a la vez sea suficiente para tener la cantidad necesaria de acuerdo a la demanda de los clientes. La compañía distribuidora recibe las solicitudes de cerveza y realiza la entrega respectiva.

Al analizar el "Juego de la Cerveza" se identificaron tres roles. El Cliente (rol 1) solicita regularmente cierta cantidad de artículos. El Minorista (rol 2) debe atender las demandas de artículos de sus clientes, decrementando el nivel del inventario a medida que satisface las órdenes de productos enviadas por los clientes. El Minorista debe tener la capacidad de satisfacer las demandas de artículos en cualquier momento, por lo tanto, necesita monitorear constantemente el inventario con el propósito de incrementarlo, cuando de acuerdo a su consideración el nivel esté bajo. En este caso, tiene que realizar una orden solicitando el producto al Mayorista (rol 3), quien recibe la orden y la satisface respondiendo cuando serán entregados.

El reto que se tuvo al realizar este prototipo fue desarrollar un sistema que satisfaga los requerimientos de este modelo en los diferentes aspectos de coordinación que presenta. Por ejemplo, como se mencionó, existe un rol Minorista cuyo comportamiento está determinado o condicionado por las actividades de los otros dos roles. Esto, desde el punto de vista de soporte, es un punto muy interesante porque debemos identificar ¿cuál es el rol que inicia la interacción?, ¿cómo atender ésta?, para enviar una respuesta adecuada. El modelo de la cerveza también contiene aspectos de decisión en el Minorista, al tener dos posibles opciones para decidir que actividad se realiza, originando acciones diferentes a partir de la decisión tomada. De la misma manera, se ven involucrados aspectos de intensa

interacción entre los roles para llevar a cabo el objetivo del proceso. Además, existe la presencia de entidades de información de tipo orden y respuesta necesarias para ejecutar las actividades de los roles. Todo esto hace que pensemos conscientemente sobre los requerimientos funcionales y tecnológicos del prototipo para facilitar en primer lugar la ejecución del proceso, y en segundo, identificar los aspectos a considerar para proponer una arquitectura de coordinación funcional.

Cuando se lleva a cabo la ejecución de un proceso dentro de una organización es indispensable la interacción o comunicación entre los agentes o actores participantes. En el caso de este proceso, existen cuatro diferentes interacciones que permiten el intercambio de información entre los roles:

1) El Cliente solicita artículos al Minorista

2) El Minorista entrega la orden de artículos al Cliente.

3) El Minorista solicita productos al Mayorista.

4) El Mayorista satisface la orden de productos hecha por el Minorista.

Como se mencionó en el capítulo II, la representación de un proceso mediante un modelo ayuda a entenderlo mejor y tener una visión más clara de lo que sucede. En la Figura 12 se muestra el RAD del "Juego de la Cerveza" (Rojas Iñiguez, 1998), en el que se puede observar el flujo de actividades que realiza cada rol para cumplir con sus objetivos particulares.

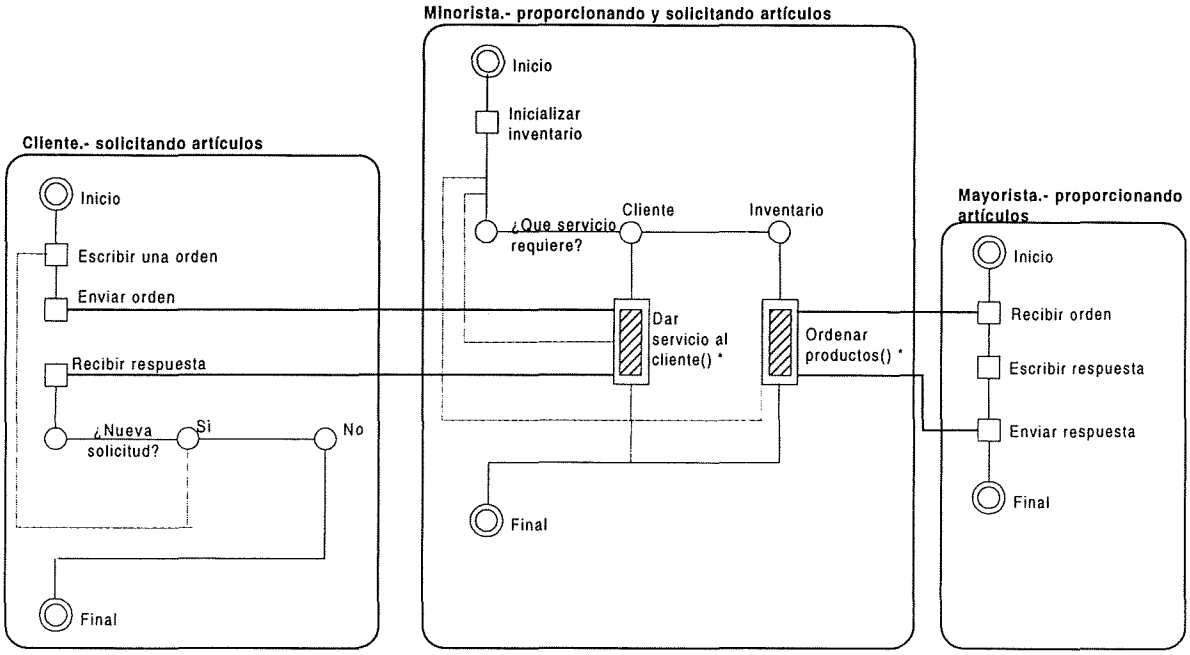

Figura 12. Modelo del proceso en RAD. Diagrama en donde se identifican los tres roles presentes en el proceso y las actividades e interacciones que cada uno tiene que realizar.

Se pueden observar los tres roles del "Juego de la Cerveza" identificando las perspectivas propuestas por Curtis: ¿qué actividad se va a hacer?, ¿quién y donde la va a realizar?, ¿cómo y cuándo se hará? y se visualiza parcialmente la parte informacional, es decir, no es claro que tipo de información contienen las entidades de tipo orden y respuesta.

El rol Cliente representado por el rectángulo izquierdo tiene como característica que puede realizar más de una orden y esto se indica con la línea punteada (iteratividad). El rectángulo del centro representa al rol Minorista, que tiene la posibilidad de elegir entre dos subprocesos: *dar servicio al cliente* u *ordenar productos* (responsabilidad en las decisiones). De la misma manera que el rol anterior, el Minorista puede repetir más de una vez alguno de los dos subprocesos. Por último, en la parte derecha se encuentra el Mayorista cuyo propósito es abastecer de productos al Minorista (meta).

Por otra parte, en la Figura 13a se muestra el subproceso *dar servicio al cliente* en el cual se verifica si existe suficiente producto en el almacén para satisfacer la demanda de los clientes, haciendo la entrega respectiva y decrementando el inventario (secuencialidad de actividades). En la Figura 13b se muestra el subproceso *ordenar productos* que tiene las actividades necesarias para realizar una orden de cerveza y enviarla al Mayorista (interacción).

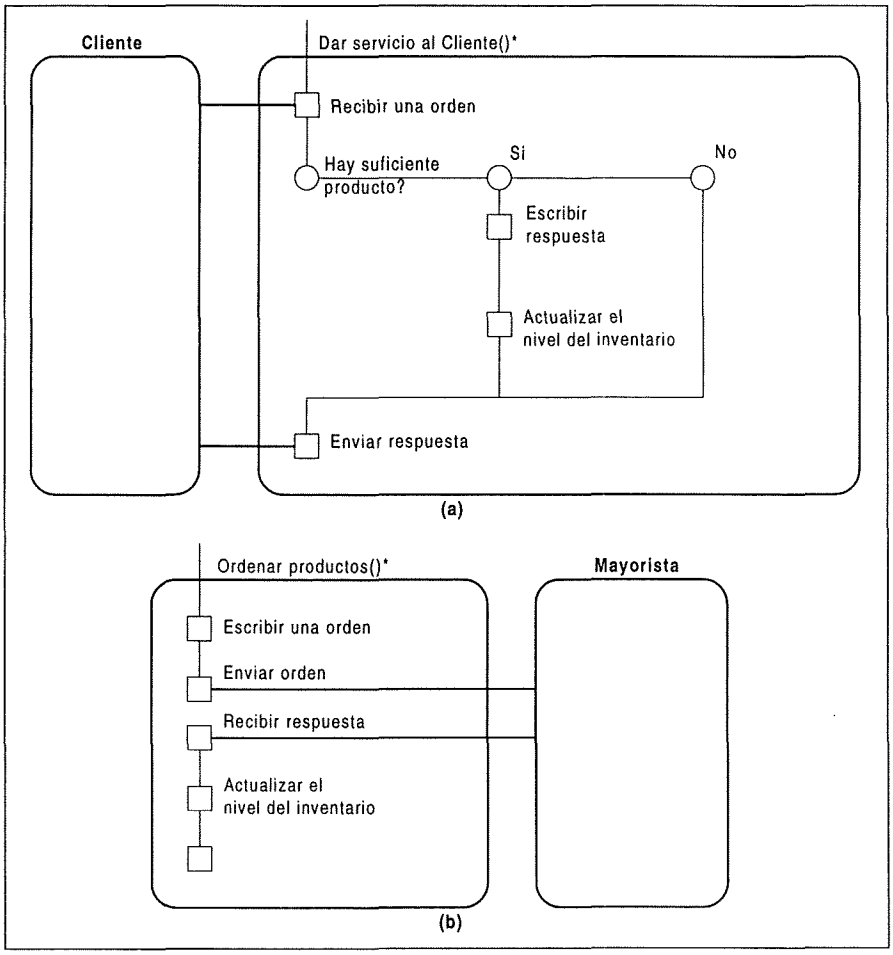

Figura 13. Diagramas RAD del subproceso *dar servicio al Cliente* (a) y el subproceso *ordenar productos* (b) del modelo de la cerveza correspondiente al rol minorista.

Como se mencionó en los capítulos I y III, a medida que los roles ejecutan sus actividades el estado de cada uno de estos va cambiando, indicando la situación en que se encuentran

en un momento determinado. También se mencionaba que el identificar los estados ayuda a identificar el comportamiento de los roles, y por ende, se facilita una mejor representación de la coordinación del proceso.

En el caso particular del rol Cliente su estado inicial es *ordenando*. Una vez que realiza la orden y la envía al Minorista, se cambia al estado *esperando* mientras recibe una respuesta. Cuando ésta llega al Cliente, entonces el estado cambia a *terminando* indicando que los productos ya han sido recibidos, teniendo la posibilidad de realizar más órdenes o pasar al estado *finalizando* en donde termina la interacción con el Minorista. De aquí que el rol Cliente esté definido de acuerdo a cuatro estados (Figura 14):

- *Ordenando*, en este estado el Cliente realiza una orden por cierta cantidad de artículos.
- *Esperando*, el Cliente permanece en el estado esperando mientras recibe una respuesta del Minorista.
- *Terminando*, este estado indica que la orden ha sido recibida y la transacción se ha terminado.
- *Finalizando*, estado en el que no se hacen más transacciones.

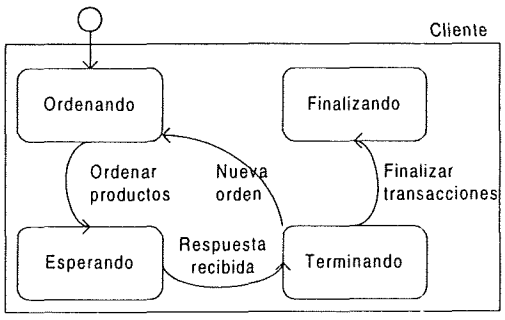

Figura 14. Diagrama de transición de estados del rol Cliente.

Las actividades del rol Minorista, por otro lado, están definidas con los siguientes estados (Figura 15):

- *Inicializando*, este es el primer estado del rol. En primera instancia se da un valor inicial al inventario del Minorista y con esto ya se encuentra listo para la ejecución de sus actividades y la interacción con el Cliente o el Mayorista.
- *Seleccionando*, en este estado el agente que ejecuta las actividades del rol tiene la posibilidad de elegir uno de dos subprocesos:

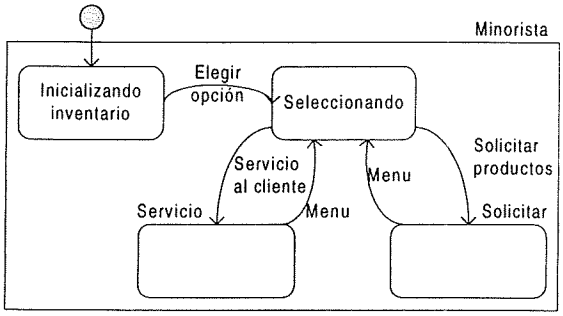

Figura 15. Diagrama global de transición de estados del Minorista. El Minorista primero inicializa el inventario y puede elegir entre dos subprocesos: dar servicio al Cliente o solicitar productos al Mayorista.

Uno de ellos es *solicitar* que tiene como función enviar una orden al Mayorista cuando el nivel del inventario está bajo (Figura 16). Los tres estados de este subproceso son:

- *Ordenando*, en este estado se escribe una orden de productos y se envía al Mayorista.
- *Esperando*, estado en el que se espera la respuesta de la orden enviada.
- *Terminando*, este estado indica que la respuesta ha sido recibida.

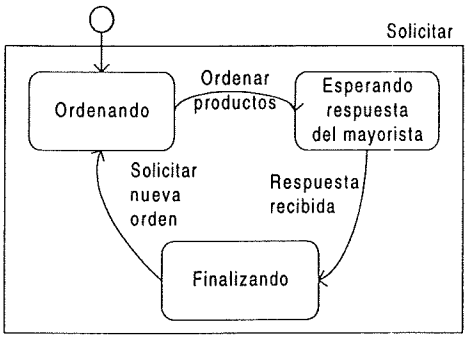

Figura 16. Diagrama de transición de estados del subproceso solicitar. El Minorista en el estado ordenando hace una solicitud de productos al Mayorista.

La segunda opción es el subproceso *Servicio* que permite al Minorista realizar las actividades necesarias para responder a las órdenes de productos enviadas por un Cliente (Figura 17). Los estados que se presentan en este subproceso son dos:

- *Esperando*, el Minorista espera recibir una orden de productos.
- *Contestando*, indica que la respuesta a una orden está siendo enviada al Cliente.

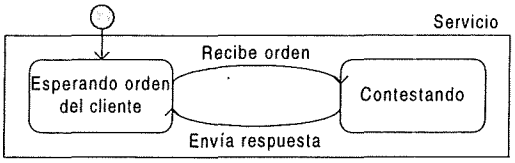

Figura 17. Diagrama de transición de estados del subproceso servicio. Cuando el Minorista elige *dar servicio al cliente*, su estado cambia a *esperando* una orden.

Los estados de las actividades realizadas por el rol Mayorista son dos: Primero se encuentra en el estado *esperando* mientras llega la orden del Minorista. Cuando la recibe, cambia al estado *contestando* para enviar una respuesta notificando la fecha de entrega de los productos.

En la Figura 18 se puede observar el respectivo diagrama de transición de estados.

- *Esperando*, este estado indica que está esperando una orden de productos del Minorista.
- *Contestando*, envía la respuesta a las órdenes de productos recibidas.

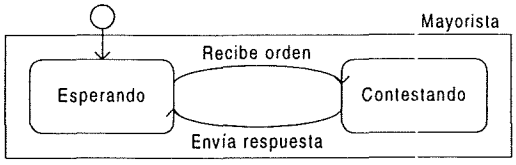

Figura 18. Diagrama de transición de estados del Mayorista. Con estos dos estados realiza las actividades correspondientes a su rol.

Resumiendo el análisis del modelo de la cerveza, son tres los roles que están involucrados en este proceso: el rol Cliente que requiere cierta cantidad de artículos y envía una orden al Minorista. El rol Minorista cuyo propósito principal es tener la capacidad de entregar los artículos sin retrasos al Cliente y tenerlo satisfecho con el servicio, además de verificar periódicamente el nivel del inventario para detectar cuando tiene que solicitar productos al rol Mayorista. Este último rol se encarga de abastecer la cantidad de productos pedidos por el Minorista.

El rol más interesante de este proceso es el del Minorista debido a que presenta tanto el comportamiento del Cliente como del Mayorista, ya que es el encargado de proporcionar un buen servicio a los clientes (rol del Mayorista) manteniendo la cantidad suficiente de artículos en almacén, y solicitar productos (rol del Cliente) para satisfacer sus demandas.

Hasta este momento, se ha identificado los elementos importantes del diagrama RAD para este modelo, además de los diferentes estados por los que atraviesa un rol (diagramas de transición de estados) después de terminar cada una de sus actividades. A continuación se presenta el análisis y diseño que nos permitirán llevar a cabo la implementación del prototipo.

IV.3 Análisis y diseño del prototipo del modelo de la cerveza

El análisis que se presenta a continuación se realizó para identificar los requerimientos necesarios para desarrollar el prototipo, que a su vez nos permitirá establecer los requerimientos para construir una arquitectura que permita generar sistemas de coordinación.

El análisis y diseño del sistema se realizó en el Lenguaje de Modelado Unificado (UML, por sus siglas en inglés) (Rumbaugh, J. et al., 2000). Se realizaron los diagramas de casos de uso, diagramas de secuencia y diagramas de clases ya que éstos nos permiten obtener los requerimientos, comportamiento y estructura básica del sistema, además de complementar la información proporcionada por los RADs y los diagramas de transición de estados. Los casos de uso son una interacción típica entre un usuario y un sistema de cómputo. Entre las propiedades de los casos de uso están: captan alguna función visible para el usuario, pueden ser pequeños o grandes, y logran un objetivo discreto (Fowler, M., 1999). Haciendo una comparación, los casos de uso representan la función del RAD mostrando los roles de un proceso y las actividades que tienen asignadas, e integrando los conceptos del modelado de procesos con el análisis para el sistema de soporte.

Como se mencionó anteriormente, los roles del proceso del "Juego de la Cerveza" son tres y para cada rol se identificaron sus respectivos casos de uso, los cuales se describen a continuación. El **caso de uso *accesar al sistema***, mostrado en la Figura 19 es llevado a cabo por cualquier rol del modelo de la cerveza.

Propósito: permitir el acceso a las personas apropiadas al sistema de soporte para este proceso.

Descripción: el usuario selecciona el rol que desea jugar y proporciona la clave de acceso al sistema para el rol elegido. El sistema recibe esta solicitud y compara si la clave corresponde con el rol. Si esto es verdadero entonces se permite el acceso al sistema y al rol especificado.

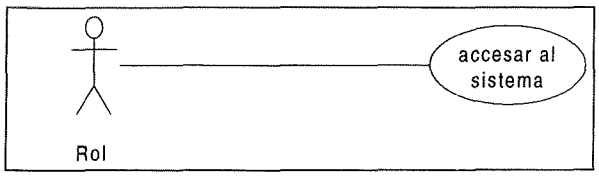

Figura 19. Accesar al sistema del modelo de la cerveza.

El **caso de uso** ***realizar una orden o solicitando artículos*** de la Figura 20 lo ejecuta el actor Cliente.

Propósito: hacer una orden en la que se escriba su nombre y la cantidad de cerveza que está solicitando.

Descripción: el Cliente escribe una orden de cerveza que contiene dos parámetros. Uno es el nombre de quien hace la petición y el otro la cantidad de productos. Cuando se tiene lista la orden se envía al Minorista y el Cliente se queda esperando una respuesta.

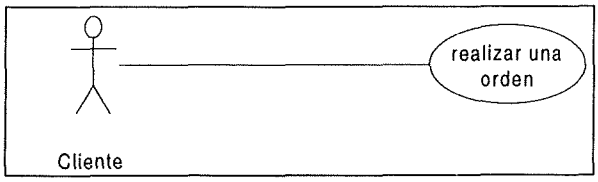

Figura 20. El Cliente envía una orden al Minorista por un número determinado de artículos y espera hasta recibir una respuesta.

La Figura 21 muestra el **caso de uso** ***abastecer productos o proporcionando artículos*** del rol Mayorista.

Propósito: entregar en un tiempo razonable la cantidad de cerveza que solicita el Minorista.

Descripción: al inicio de las actividades del Mayorista se encuentra esperando que el Minorista envíe una orden de cerveza. Cuando ésta llega, toma la orden, lee la información y envía como respuesta los productos que se le han solicitado.

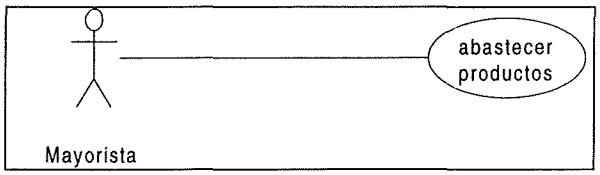

Figura 21. El Mayorista satisface la petición de artículos que solicita el Minorista.

En la Figura 22 se presentan los casos de uso del Minorista y enseguida se describe cada uno de ellos.

Caso de uso: ***inicializar inventario***.

Propósito: inicializar la cantidad de artículos en el inventario.

Descripción: cuando el Minorista inicia sus actividades por primera vez, tiene que especificar la cantidad de cerveza con la que empezará a dar servicio a sus Clientes.

Caso de uso: ***solicitar artículos***.

Propósito: incrementar el nivel del inventario cuando la cerveza se esté terminando.

Descripción: si la cantidad de cerveza en el almacén no es suficiente para satisfacer la demanda de los Clientes, es necesario pedir productos al Mayorista para continuar dando servicio. Para hacer esto, se llena una orden en la cual se indica el nombre de quien hace la solicitud y la cantidad.

Caso de uso: ***proporcionar artículos***.

Propósito: satisfacer la demanda de productos realizada por los Clientes.

Descripción: cuando el Minorista desea dar servicio al Cliente, verifica si han llegado órdenes. Si éstas existen, revisa el inventario y si cuenta con la capacidad suficiente envía a cada Cliente la cantidad de cerveza solicitada.

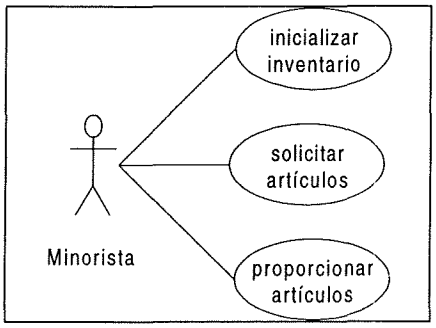

Figura 22. Se presentan los tres casos de uso del rol Minorista: ***inicializar inventario, solicitar artículos y proporcionar artículos.***

Una vez identificados los casos de uso, se realizaron los diagramas de secuencia para cada uno de ellos. Estos diagramas muestran el paso de mensajes en el tiempo de los objetos involucrados en el sistema. Existe igualmente para los diagramas de secuencia una relación conceptual con los roles de un diagrama RAD, debido a que sigue una secuencia de actividades para alcanzar un objetivo específico apoyado en los métodos definidos. Estos diagramas ayudan a cumplir las metas de los roles y proporcionan un mejor soporte informacional al proceso, ya que requieren especificar los campos de información que conforman una actividad o los objetos de tipo entidad.

En la Figura 23 se presenta el diagrama de secuencia para el caso de uso *realizar una orden*. Se puede observar el mensaje *crearOrden()* con el cual se crea un nuevo objeto tipo *Orden* para el Cliente. Como se mencionaba, en este objeto se especifica claramente la parte informacional del proceso con los atributos: nombre y la cantidad de artículos. Una vez que se completa la orden se envía al Minorista.

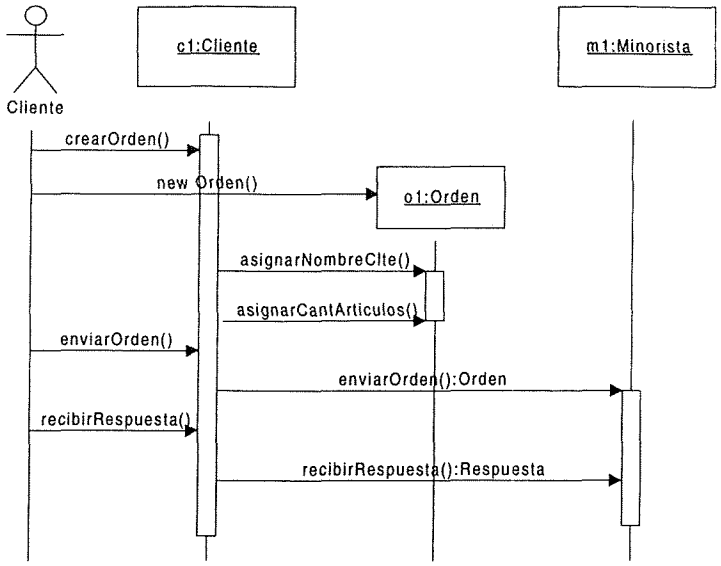

Figura 23. Diagrama de secuencia del caso de uso realizar una orden.

En la Figura 24 se muestra el diagrama de secuencia del caso de uso *abastecer productos* del Mayorista. En este diagrama el objeto *ma1* recibe la orden enviada por el Minorista, obtiene los datos de la misma y crea una respuesta por medio del objeto *rb1* en el que se indica a quién se entregará, la cantidad y la fecha de entrega. Finalmente se envía la respuesta al Minorista.

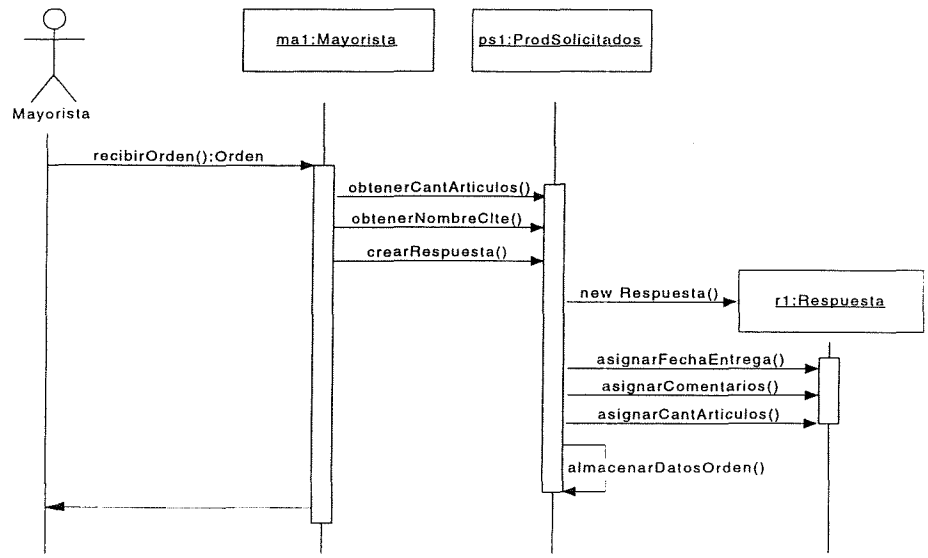

Figura 24. Diagrama de secuencia para el caso de uso abastecer productos.

En la Figura 25 se presenta el diagrama de secuencia para el caso de uso *proporcionar artículos* del Minorista. En este diagrama el objeto *m1* recibe la orden enviada por el Cliente, obtiene los datos de la misma y crea una respuesta por medio del objeto *rb1* en el que se indica a quién se entregará y la cantidad de productos. Finalmente se envía la respuesta al Cliente.

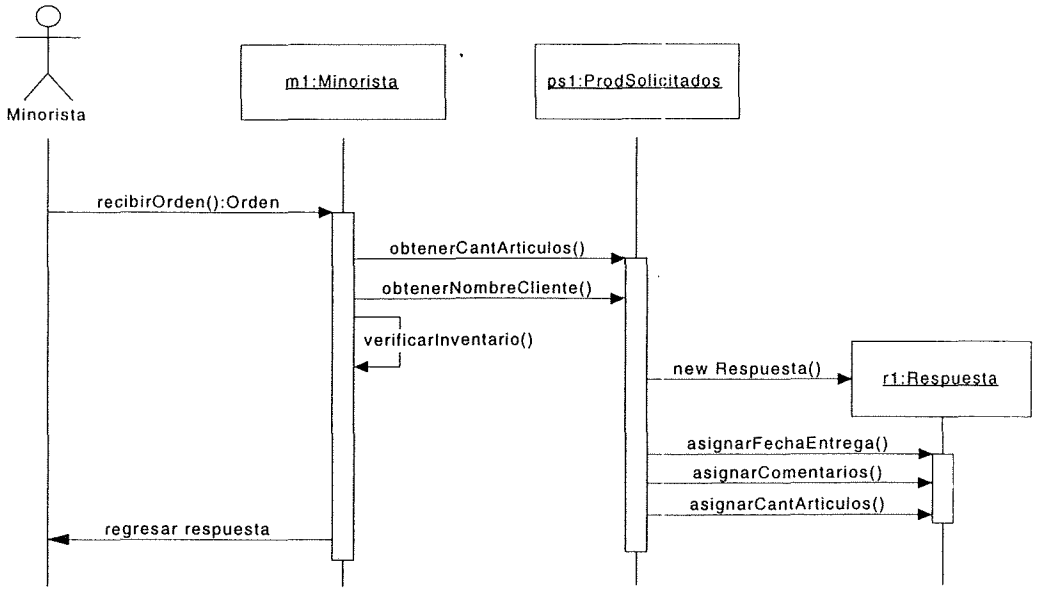

Figura 25. Diagrama de secuencia del caso de uso proporcionar artículos.

El diagrama de secuencia de la Figura 26 representa el caso de uso *solicitar artículos* del Minorista. Primeramente se checa el inventario, esto se realiza por medio del mensaje *verificarInventario()*. Se puede observar el mensaje *crearOrden()* con el cual se crea un nuevo objeto tipo *Orden* para el Minorista, que tiene como atributos el nombre y la cantidad de artículos. Una vez que se completa la orden se envía al Mayorista.

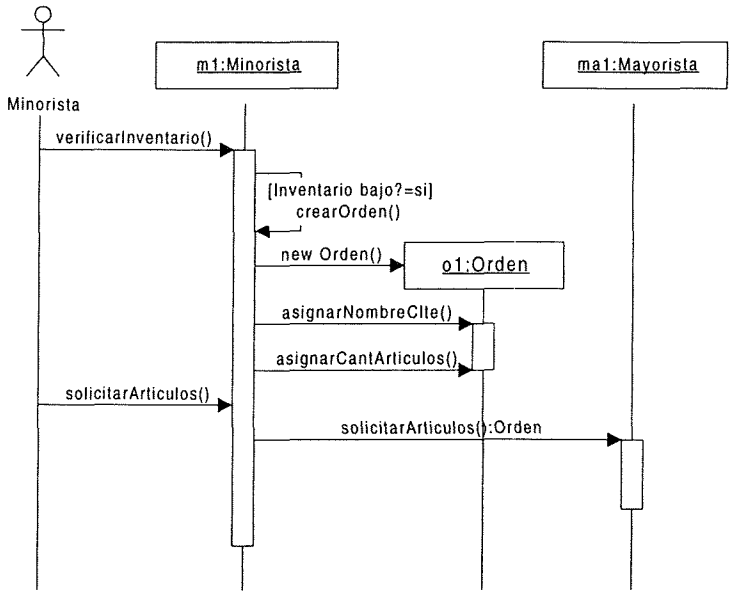

Figura 26. Diagrama de secuencia del caso de uso solicitar artículo.

A partir de los diagramas de secuencia es posible generar el diagrama de clases que nos permita obtener la estructura del sistema para posteriormente realizar su implementación. Los diagramas de clases describen los tipos de objetos en el sistema y las diferentes clases de relaciones estáticas que existen entre ellos.

Por supuesto la arquitectura debe facilitar la persistencia del proceso durante todo su ciclo de ejecución, proporcionando en todo momento a los roles, las actividades que necesitan realizar. Estos puntos han sido identificados y tomados en cuenta para desarrollar un sistema apropiado para el soporte del modelo de la cerveza.

La Figura 27 presenta la arquitectura propuesta inicialmente mediante la cual se implementó el prototipo. Como se puede observar el rectángulo grande representa un rol. Éste debe tener algunas propiedades como las *entidades de información* manipuladas cuando realiza sus actividades y una *tabla de estado* la cual guiará su comportamiento en el proceso. Así mismo, existe un c*oordinador de roles* que se comunica con dos componentes: un *agente de estados* que se encarga de cambiar el estado de un rol después de una actividad y un *manejador de interfaces* que permita la ejecución de las actividades.

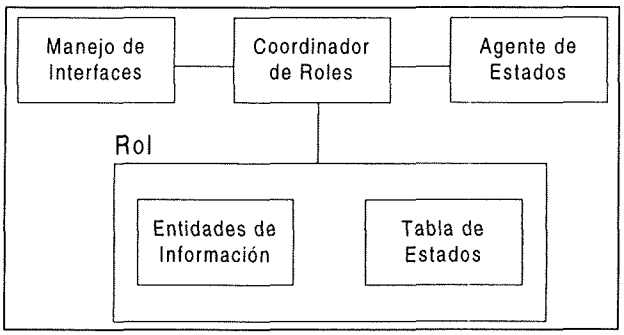

Figura 27. Arquitectura propuesta para la implementación del modelo de la cerveza.

La arquitectura que acabamos de describir se implementó por medio del diagrama de clases de la Figura 28, que está formado por una clase *Rol* a la cual se agregan dos clases que le proporcionan parte de su funcionalidad. La primera de ellas es la clase *Coordinacion* en la cual se definen los métodos que permiten la interacción entre los roles. La segunda es la clase *Estados* que se encarga de guardar los estados de cada uno de los roles del proceso y permite recuperar esa información conforme se vayan ejecutando las actividades de los roles.

Igualmente se encuentran las clases específicas para la implementación del modelo de la cerveza. A continuación se mencionan estas clases y en el apéndice B se muestra el diccionario de clases.

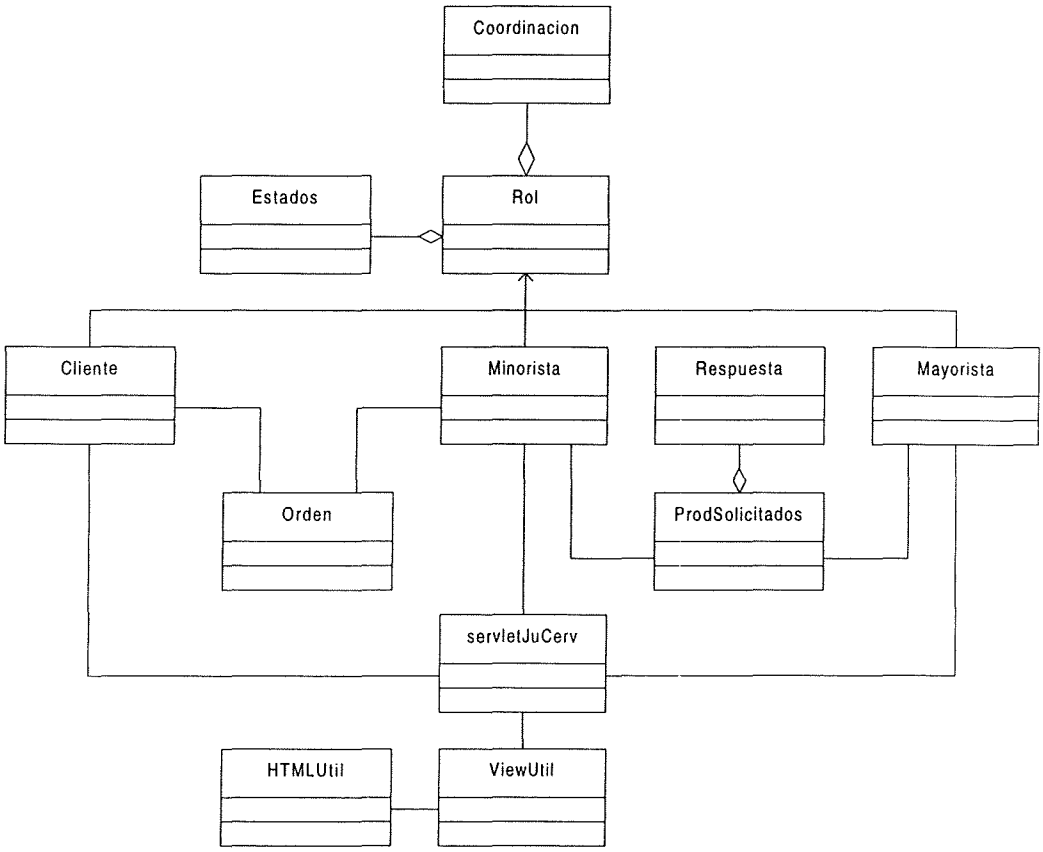

Figura 28. Diagrama de clases del proceso del "Juego de la Cerveza", en donde se tiene una clase principal llamada Rol (que es una agregación de las clases Coordinacion y Estados) de la cual heredan sus características las clases que representan los roles del proceso (Cliente, Minorista y Mayorista).

- *Cliente*, en esta clase se define los atributos y métodos que le dan la funcionalidad requerida al rol con el mismo nombre para realizar y enviar órdenes.
- *Minorista*, esta clase se encarga de definir los atributos y métodos que van a permitir a este rol interactuar con el Cliente y satisfacer sus demandas de productos, así como también, verificar constantemente la cantidad de artículos en almacén con el propósito de enviar una orden al Mayorista cuando el nivel del inventario esté bajo.

- *Mayorista*, esta clase se encarga de responder exclusivamente a las órdenes de productos del Minorista y mientras no existan éstas, espera hasta el arribo de una orden.
- *Orden*, utilizada para que los Clientes o Minoristas puedan enviar solicitudes de artículos a sus respectivos proveedores.
- *ProdSolicitados*, esta clase se encarga de guardar las órdenes de productos recibidas por el Minorista y Mayorista.
- *Respuesta*, la función de esta clase es crear una respuesta a los roles que solicitan productos y enviarla a los mismos.
- *servletJuCerv* (coordinador del proceso), esta clase tiene como propósito coordinar las interacciones y actividades realizadas por los agentes del proceso.
- *ViewUtil,* esta clase permite a los roles crear sus interfaces dependiendo de la actividad que deban realizar.
- *HTMLUtil,* esta clase es utilizada por ViewUtil para formar las interfaces de los roles.

IV.4 Implementación y secuencia de escenarios

En esta sección se presentan los aspectos tecnológicos considerados para construir el sistema de coordinación para el modelo de la cerveza, el cual fue la base para tomar una decisión sobre la tecnología a utilizar en el desarrollo de la arquitectura. Además se muestran las interfaces o escenarios generados para los roles, explicando con un ejemplo la coordinación entre el Cliente, Minorista y Mayorista.

IV.4.1 Aspectos tecnológicos de implementación

El desarrollo de un sistema en general requiere establecer con claridad algunas características como: funcionalidad esperada, el tipo de usuarios a quien está dirigida, la plataforma bajo la cual se ejecutará, entre otras, para seleccionar la tecnología apropiada. Sin embargo, existen algunos riesgos tecnológicos que deben ser considerados como la manera en que funcionan conjuntamente los componentes seleccionados para evitar demoras en la implementación de un sistema y asegurar la integración de la tecnología seleccionada.

Para el sistema del juego de la cerveza se consideraron en principio dos formas de implementación: una centralizada y otra distribuida. El ambiente centralizado involucra la utilización de un servidor de *Web* como Apache para el manejo de las interfaces HTML por medio de un navegador como Netscape Navigator, Microsoft Internet, Mozilla, Galeon, etc. Este servidor tiene soporte a Servlets para el manejo de las peticiones provenientes de los usuarios y las respuestas a las mismas.

El ambiente distribuido permite tener objetos localizados en las máquinas de los usuarios de un sistema enviándose información entre estos objetos. CORBA es una tecnología que facilita el desarrollo de este tipo de aplicaciones proporcionando la transparencia de estos objetos en la red, la independencia de la plataforma y el lenguaje de programación.

El lenguaje de programación utilizado fue Java (Chan, M.C., 1997) debido a su portabilidad, a que es un lenguaje libre y que mucha de su aportación está orientada hacia el

manejo de aplicaciones para "Internet", proporcionando una gran cantidad de clases que ayudan a alcanzar este propósito.

Finalmente, la tecnología seleccionada fue Java en un ambiente centralizado con la ayuda de un servidor Apache con soporte a Servlets (Rossbach, P. y Schreiber, H., 2000), el cual se encargará de recibir las peticiones de los clientes y regresar una respuesta. Esto significa que la información va estar en un solo lugar y que el control del proceso de llevará a cabo de manera local en el servidor. La ventaja de utilizar Java y Servlets es que la API (Application Program Interface) del primero tiene acceso a todas las funciones comúnmente utilizadas en los Servlets, facilitando la integración de ambas tecnologías. Por su parte, los Servlets resuelven los problemas de desempeño de un servidor ejecutando todas las solicitudes como hilos en un sólo proceso, es decir, no se ejecuta un nuevo Servlet para cada cliente, evitando con esto la sobrecarga de trabajo en el servidor y asegurando de esta forma que los usuarios conectados sean atendidos rápidamente.

Una desventaja del protocolo HTTP utilizado por los *servidores de Web* es que no permite mantener una conexión persistente con los clientes (usuarios), sin embargo, se han establecido dos mecanismos para evitar este problema. Uno es el *server push* que mantiene una conexión persistente con el cliente, al enviar únicamente bloques de información. Otro mecanismo es el *client pull* el cual envía una petición al servidor cada *t* segundos para que éste le devuelva información. Este último mecanismo es el que se está utilizando en el sistema porque satisface con la funcionalidad esperada, y no se pierde la comunicación con el servidor.

IV.4.2 Secuencia de escenarios

En esta sección se muestra una secuencia de interfaces para ejemplificar lo que sucede en el proceso a medida que los roles realizan sus actividades. La Figura 29 es la interfaz principal de la implementación del "Juego de la Cerveza" y corresponde al caso de uso *accesar al sistema*. En ésta, se tiene la posibilidad de elegir el rol que el usuario desea representar. Esto se realiza mediante la selección del mismo e introduciendo la clave de acceso correspondiente.

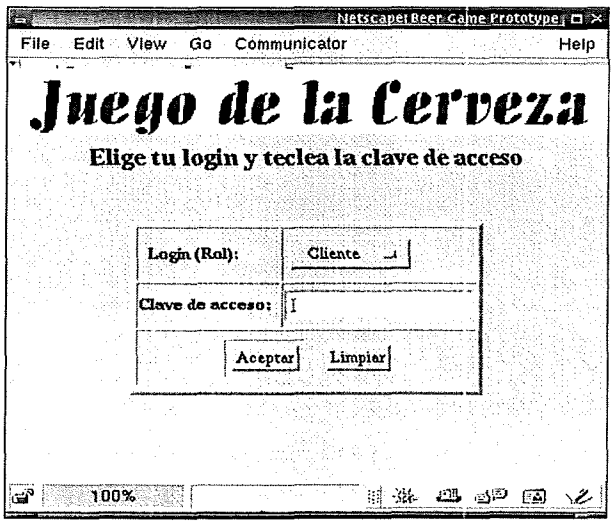

Figura 29. Interfaz principal de acceso al sistema del "Juego de la Cerveza". El usuario tiene que elegir un rol del menú que aparece en el campo *Login* e introducir la clave de acceso para el rol seleccionado.

La Figura 30 presenta un ejemplo en el que un usuario seleccionó el rol Cliente, la interfaz que aparece en la esquina superior izquierda le permite realizar una orden de productos mediante la captura de los datos en los campos *Nombre del cliente* y *Cantidad del pedido* de la interfaz (caso de uso *realizar una orden*). En este caso el cliente "Francisco" hace la solicitud de 20 artículos. Para enviar la orden al Minorista se debe presionar el botón

Aceptar. Si alguno de los dos campos no contiene los datos correctos, el usuario puede presionar el botón *Limpiar* para eliminar el contenido de los mismos.

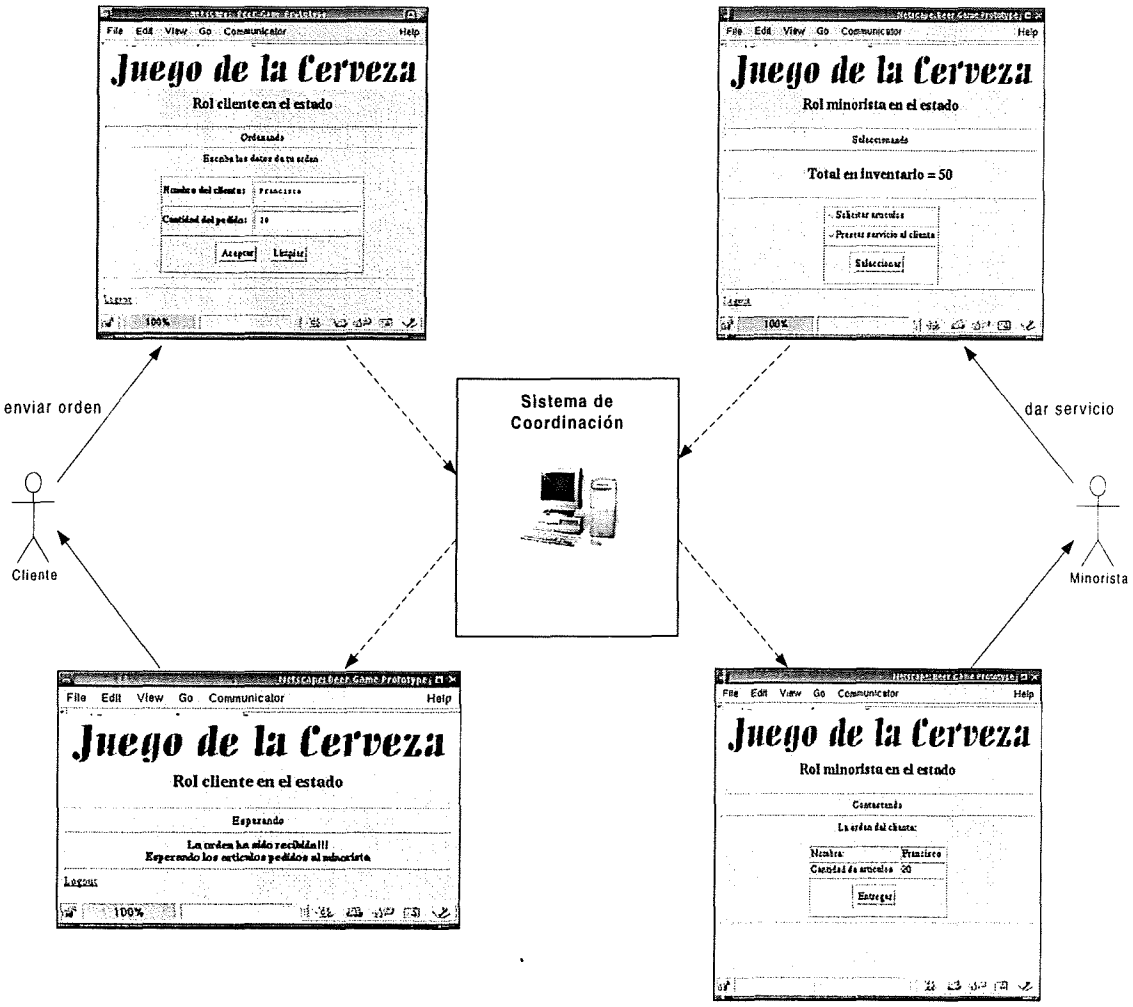

Figura 30. Esta figura muestra la parte dinámica que presenta el sistema de coordinación para que el Minorista responda a las solicitudes de cerveza de los Clientes.

El usuario puede salir del rol presionando la liga *Logout* que se localiza en la parte inferior de la interfaz. Cuando se envía la orden, ésta viaja al sistema de coordinación quien la recibe y se encarga de actualizar su estado a *esperando* asignándole la siguiente actividad y la interfaz correspondiente. El sistema pasa esta orden al rol Minorista para que le dé una respuesta al Cliente. Un usuario que desea conectarse al rol Minorista tiene que elegir esa

opción en la interfaz de acceso de la Figura 29 e introducir la clave correspondiente a este rol. Cuando el Minorista entra por primera vez recibe una interfaz con el estado actual que en este caso es *seleccionando*. Este rol puede realizar dos subprocesos: *prestar servicio al Cliente* (caso de uso *proporcionar artículos*) y/o *solicitar artículos* (caso de uso con el mismo nombre) al Mayorista. Como se observa en la Figura 30, si el Minorista selecciona la opción prestar servicio al Cliente, su estado cambiará a *contestando* y la interfaz mostrará la orden recibida.

Por otra parte, la Figura 31 presenta el subproceso *solicitar artículos*. En este caso, el rol Minorista captura los datos de una orden de productos y la envía al Mayorista cambiando su estado a *esperando* que se puede observar en la interfaz de la parte superior izquierda. El Minorista puede permanecer ahí o regresar al estado *seleccionando* de la interfaz principal de este rol. El rol Mayorista tiene como propósito principal satisfacer la demanda de productos que realiza el Minorista. En la parte superior derecha se muestra la interfaz inicial en el estado *esperando* ya que el rol, en un inicio, está atento esperando recibir una solicitud de productos en cualquier momento. El estado del Mayorista cambia de *esperando* a *contestando* en el momento en que se recibe una orden. Al llegar la solicitud de productos aparece una interfaz se muestra en la parte inferior derecha. En esta aparecen los datos de quien envía la orden. En este ejemplo, aparece la orden de "Pedro Gómez" por una cantidad de 10 artículos. A esta respuesta se le agregan otros dos campos. Uno para indicar la fecha de entrega y el otro para algunos comentarios sobre la orden (caso de uso *abastecer productos*). Al enviar la respuesta el estado del Mayorista cambia nuevamente a *esperando*. La parte inferior izquierda muestra la respuesta a la petición de artículos realizada por el

Minorista "Pedro Gómez" en donde se especifica la cantidad, la fecha de entrega y algunos comentarios si son necesarios. Las entidades de información intercambiadas entre los roles se crean a partir de los campos que se encuentran en las formas HTML.

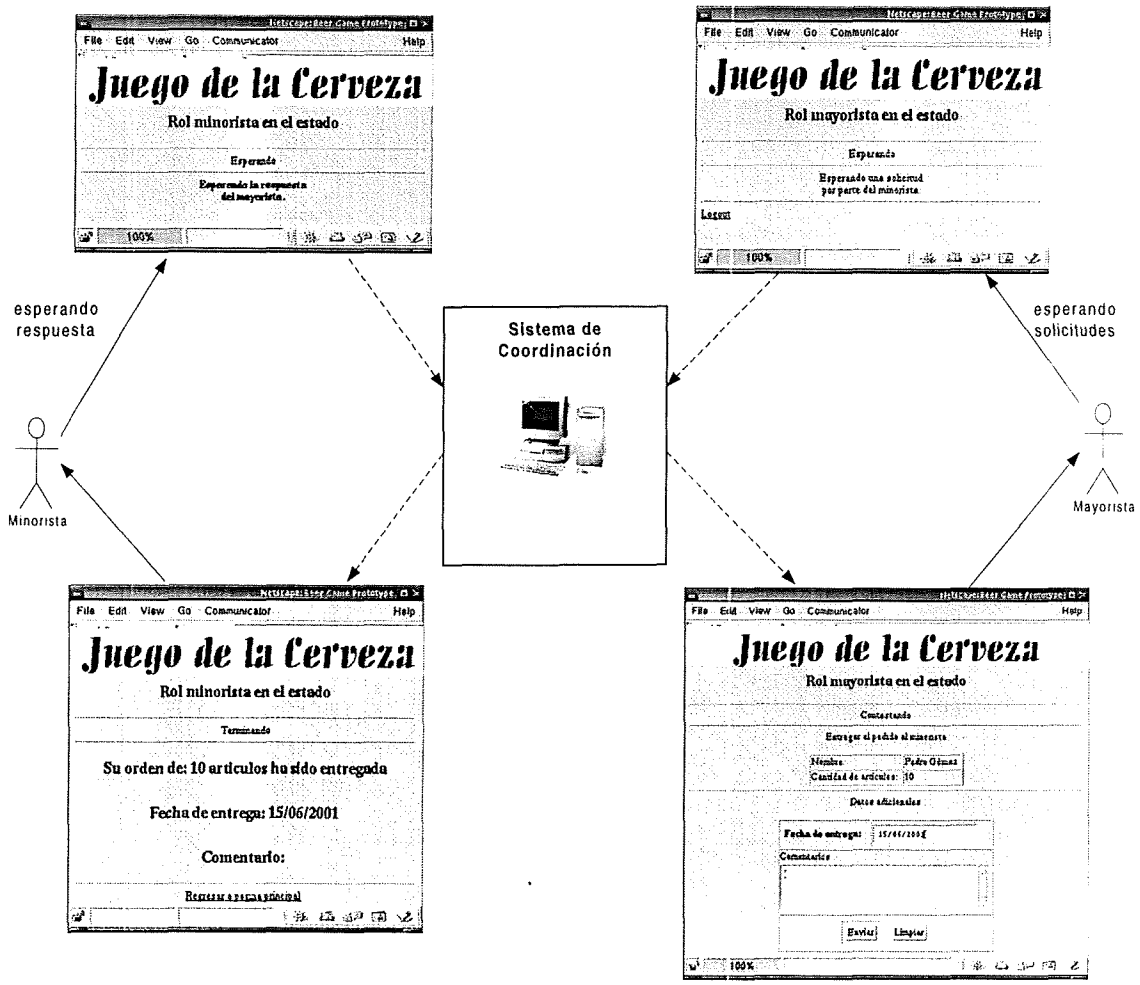

Figura 31. Aspecto dinámico del subproceso *solicitar artículos* del Minorista.

IV.5 Conclusiones

El desarrollo del prototipo presentado en este capítulo, fue muy importante porque permitió identificar los requerimientos a satisfacer con la arquitectura de coordinación para generar sistemas que brinden soporte a procesos.

Estos requerimientos tienen que ver con los aspectos que involucra el *enactment* de un proceso como roles, actividades, coordinación de roles, intercambio de información, persistencia del proceso, entre otros. De tal forma que se analiza la manera de incluirlos en la implementación del prototipo, además de resolver algunos problemas técnicos que no se detectaron en un inicio como la falta de conexión persistente entre los roles de un proceso y el sistema de coordinación del prototipo.

En este prototipo se utilizó un servidor *Web* (Apache) con soporte a *Servlets* y para la implementación el lenguaje de programación Java. De lo anterior se puede concluir que la integración de tecnologías se realizó sin problemas, facilitando el desarrollo del sistema y permitiendo la coordinación del proceso sin dificultades técnicas. Por esta razón, se seleccionó estas tecnologías como las más viables para implementar la arquitectura de coordinación que se propone en este trabajo.

También con este prototipo se definieron los componentes que conforman la arquitectura. Por ejemplo, uno de los detalles técnicos que se habían considerado en un principio era utilizar una arquitectura dentro de la cual se definieran los roles de un proceso y algunas responsabilidades. Unas de estas últimas, era asignar a cada rol las tareas que debía realizar; y la otra, dirigir el comportamiento de los mismos. Una vez que se realizó el prototipo, se observó que recaía bastante responsabilidad en la arquitectura para coordinar a todos los roles involucrados en un proceso. Por lo tanto, en la arquitectura que se propone en el siguiente capítulo, a cada rol se le asocia un agente de actividades y otro de estados para coordinar sus propias actividades. Además, se vio la necesidad de utilizar diagramas de

transición de estados para identificar el comportamiento de los roles y facilitar la asignación de nuevas actividades. Otro aspecto interesante proporcionado por el prototipo fue que ayudó a definir el mecanismo de comunicación entre los roles porque se tuvo que establecer claramente quién iniciaba una interacción en un proceso y como se llevaba a cabo. El mecanismo de navegabilidad o responsabilidad propuesto para este trabajo es el *client pull*, el cual permite a un rol hacer una petición de información al sistema de coordinación para verificar si puede continuar con sus actividades, este último apoyándose en una tabla de estados para asignar las tareas pendientes a los roles.

Capítulo V. Arquitectura de coordinación

V.1 Introducción

En este capítulo se presentan los requerimientos principales que se han identificado para construir una arquitectura que permita mapear el modelo de un proceso a un sistema de coordinación. La definición de los requerimientos es una etapa importante al estudiar los problemas, ya que son la base para entender su dominio y realizar un análisis y diseño precisos que permitan, en nuestro caso, representar los elementos necesarios de una arquitectura para construir un sistema que pueda coordinar un proceso. Durante la etapa de diseño se generan diagramas del sistema, se identifican las entidades de información involucradas, se especifica la manera en que se resolverá un problema, entre otros aspectos. En las siguientes secciones se presentan los aspectos mencionados enfocados en la arquitectura de coordinación.

V.2 Requerimientos específicos de la arquitectura

La finalidad de estos requerimientos es describir de manera precisa la funcionalidad de la arquitectura de coordinación y las restricciones de operación. También se conocen como especificaciones funcionales. A continuación se presenta una lista de los requerimientos esperados.

1. Debe estar bien definido el modelo RAD del proceso al que se desea dar soporte.
2. Es necesario crear un modelo base del diagrama de un proceso, mapeando los elementos representados en el diagrama al modelo. Este modelo debe cumplir con las

características de reusabilidad e independencia de la plataforma para permitir la construcción de diferentes sistemas de soporte a partir del mismo modelo. Los elementos que se deben identificar son los siguientes:

- Los roles que intervienen en el proceso.
- Las interacciones existentes entre los roles.
- Las actividades que realiza cada rol.
- Las acciones que debe llevar a cabo un rol para ejecutar cada una de sus actividades.
- Los estados asociados a cada una de las actividades de los roles.
- La información que necesitan los roles para llevar a cabo sus actividades.

3. Debe estar disponible la información concerniente a cada uno de los roles de un proceso. Para cumplir este punto se necesita hacer lo siguiente:
   - Guardar la información de los estados para cada rol.
   - Almacenar los datos capturados por los agentes a través de interfaces en el sistema de coordinación.
4. Definir un Administrador Global de Procesos (AGP) que se encargue de coordinar las actividades e interacciones de los roles, que como se comentó anteriormente, es una parte importante del *enactment* de procesos.
5. Definir un rol base que contenga los elementos fundamentales para cualquier rol, y a partir del cual se puedan crear los roles involucrados en un proceso. Este rol base permite:
   - Declarar las interacciones de un rol.
   - Declarar las actividades de un rol.

- Declarar un agente de actividades que se encargue de proporcionar al AGP las actividades que debe realizar un rol.
- El agente de actividades también proporciona las entidades de información que necesita un rol para realizar sus actividades.
- Declarar un agente de estados que se encargue de obtener y proporcionar información concerniente al estado de un rol a un agente de actividades.
- El agente de estados también proporciona al agente de actividades las interfaces necesarias para los roles, de acuerdo al estado en que éstos se encuentren.

6. Se tiene que informar al AGP sobre los roles que intervienen en un proceso para que pueda llevar a cabo su coordinación.
7. Es necesario crear las interfaces correspondientes a cada uno de los estados de los roles involucrados en el proceso, ya que por medio de éstas podrán ejecutar sus actividades. Este punto involucra aspectos informacionales por medio de las interfaces, y aspectos de comportamiento empleando el concepto de estados para dirigir las actividades de un rol. Siendo características presentes en todo sistema de coordinación.
8. Un punto importante que debe considerar el sistema de coordinación es mantener la persistencia de los roles y sus estados en un proceso. Persistencia en un sistema significa no perder la secuencia de actividades que están realizando los roles, y como consecuencia, conocer el estado global de un proceso. Para ello,
   - El sistema de coordinación debe almacenar los estados subsecuentes a la realización de cada una de las actividades de un rol para asegurar su persistencia.
   - El sistema de coordinación debe proporcionar el estado que tenía un rol antes de desconectarse para continuar con alguna actividad pendiente.

- De la misma forma, la información correspondiente a un rol debe perdurar en el sistema de coordinación para que sea recuperada en cualquier momento.

9. Utilizar un ambiente cliente-servidor para implementar el esquema de comunicación e intercambio de información a través del sistema de coordinación que se genere con la arquitectura.
10. La arquitectura de coordinación que se desarrolle debe facilitar la creación de un sistema para coordinar procesos, y que estos sistemas puedan ser ejecutados desde cualquier computadora conectada a Internet (usuarios físicamente distribuidos), produciendo el *enactment* de un proceso.

Los requerimientos descritos cubren las características que demanda el *enactment* de procesos mencionados en el capítulo II. Se propone la creación de un administrador para el manejo del proceso que facilite la interacción y la comunicación de los roles que siempre es necesaria cuando se da soporte a la ejecución automatizada de procesos. Los roles por su parte demandan las entidades de información para llevar a cabo sus actividades, las cuales se consideran para la implementación de este administrador. Así como estas características, existen otras que son elementales y por lo tanto se consideran en el diseño de la arquitectura para que ésta permita generar un sistema que apoye de manera eficiente a un proceso. Una vez establecidos los requerimientos básicos para la arquitectura de coordinación, en la siguiente sección presentamos el diseño que permitirá construirla.

V.3 Diseño

En base a los requerimientos establecidos, proponemos la arquitectura de coordinación que se presenta en la Figura 32, la cual está compuesta de un *administrador global de procesos* (AGP) que tendrá la responsabilidad de coordinar las actividades que realizan los roles involucrados en un proceso, además de establecer la comunicación entre ellos. El rectángulo etiquetado como *rol base* es un componente que permite definir las características de cada uno de los roles de un proceso para que puedan ser agregados al AGP. El rectángulo etiquetado como *agente de actividades* informa al AGP las actividades que debe realizar un rol y con quién va a interactuar. También tiene como función recuperar las entidades de información que necesita el rol para llevar a cabo sus actividades. El *agente de estados* que se observa en la Figura 32 tiene como función actualizar el estado de un rol a partir de las actividades que vaya realizando, obteniendo esta información de una tabla de estados. Además el *agente de estados* debe recuperar una interfaz (página HTML) con los elementos específicos necesarios de acuerdo al estado en que se encuentra un rol para que pueda ejecutar sus actividades dentro del proceso.

Para la creación de un sistema de coordinación, la arquitectura a través del AGP leerá el modelo base en XML del proceso, el cual contendrá una representación estructurada y textual de sus elementos como los roles, los agentes, las actividades, las interacciones, entre otros.

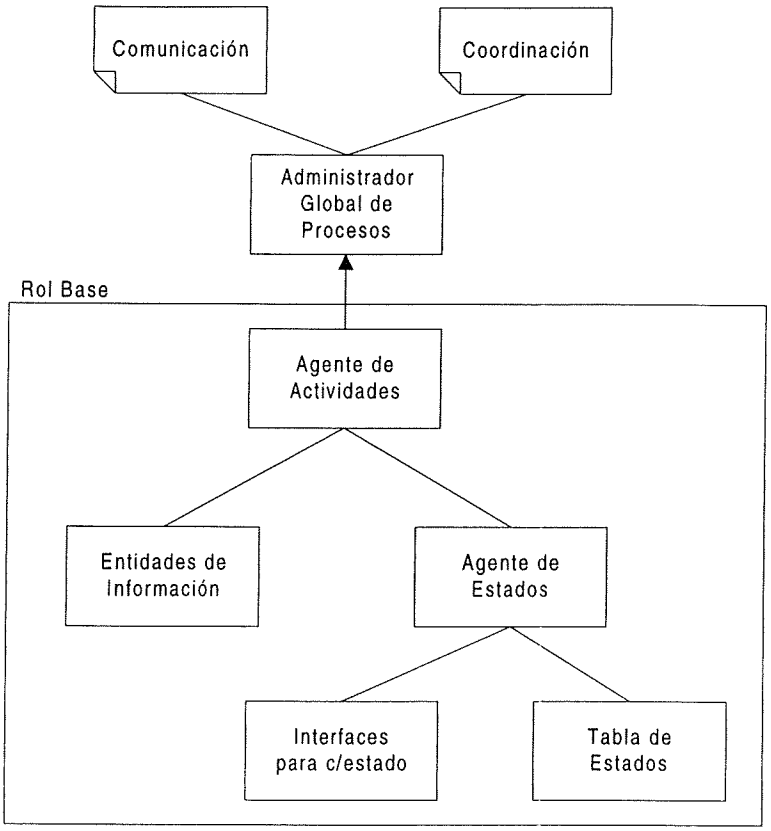

Figura 32. Arquitectura de coordinación.

La Figura 33 muestra la arquitectura de coordinación vista en forma de capas, la cual está constituida en la parte superior por los *usuarios*, que son los agentes que ejecutan un rol; a continuación está el *administrador global de procesos* cuyo propósito es coordinar a los roles involucrados en un proceso; enseguida, se encuentra el *rol base* que permite definir las características comunes a cualquier rol como: con quién interactua, los estados por los que atraviesa durante el proceso, las actividades asignadas, entre otros, permitiendo adecuarlo a un rol específico; en la cuarta capa está el *agente de actividades* y en la quinta capa están las *entidades de información* que son los elementos o datos que necesita un rol para realizar sus actividades, y en el mismo nivel se localiza el *agente de estados*, quien es el encargado de mantener actualizado el estado de cada rol según las actividades que va

realizando; en la capa inferior, se encuentra el protocolo de comunicación por medio del cual se realizará la conexión y el envío de información entre los roles.

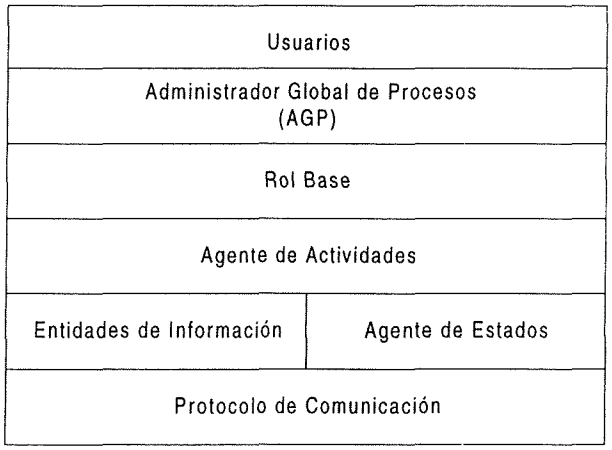

Figura 33. Arquitectura de coordinación vista en forma de capas.

Se han descrito hasta ahora los componentes que conforman la arquitectura de coordinación, por medio de los cuales se podrá construir un sistema que brinde soporte a un proceso. A continuación, se describen los casos de uso, diagramas de secuencia y diagramas de clases con los cuales se construyó la arquitectura mencionada anteriormente.

V.4 Casos de uso

El propósito de los casos de uso es mostrar la interacción que tienen los actores con un sistema o las metas que éstos desean alcanzar. Un actor puede ser tanto una persona como un sistema. El caso de uso de la Figura 34 tiene como actor principal la *arquitectura de coordinación* y se describe a continuación.

Caso de uso: *generar roles.*

Propósito: crear y definir los roles que participan en un proceso y declararlos en el AGP.

Descripción: para crear los roles, primeramente se tiene que leer el modelo base de un proceso e identificar los roles involucrados. Conforme se realiza esta actividad, se verifica mediante un analizador que el modelo esté bien formado y sea válido de acuerdo a las reglas estructurales definidas. Si se cumple con estos dos requisitos, para cada rol se identifican y definen las actividades que deben realizar, las interacciones que tienen con otros roles, los estados por los que pasan en el proceso, las entidades de información que necesitan para realizar sus actividades, entre otros aspectos. Una vez definido un rol se declara en el AGP para que coordine su participación en el proceso.

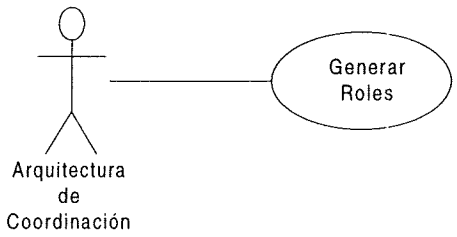

Figura 34. Caso de uso generar roles de la arquitectura de coordinación.

Otro actor identificado es el *Administrador Global de Procesos*. Sus casos de uso se muestran en la Figura 35 y se describen a continuación.

Caso de uso: *establecer comunicación entre roles.*

Propósito: informar a los roles que interactúan entre sí sobre el estado de las actividades que están realizando.

Descripción: cuando un rol termina de ejecutar una actividad que es importante para que otro rol pueda continuar con la suya, el AGP le comunica esta situación al segundo rol para que prosiga con sus actividades. El AGP debe estar informado de las actividades que se están realizando y quién las está ejecutando. También necesita saber qué roles están

interactuando en un momento determinado. Sobre la base de esta información, el AGP puede notificar a un rol que continúe con la siguiente actividad.

Caso de uso: *coordinar un proceso.*

Propósito: controlar las actividades de los roles indicándoles cuando deben actuar dentro de un proceso.

Descripción: cada uno de los roles tiene un agente que informa al AGP las actividades que van a realizar. Otro agente accede a la tabla de estados de su rol para obtener el estado y comunicar al AGP cual es su situación actual.

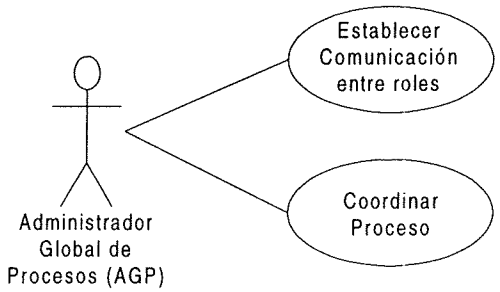

Figura 35. Casos de uso del administrador global de procesos.

En la Figura 36 se presentan los casos de uso del *agente de actividades.* A continuación se describen cada uno de ellos.

Caso de uso: *realizar actividad.*

Propósito: la finalidad de este caso es que un rol ejecute una tarea asignada como parte de su responsabilidad en un proceso.

Descripción: cuando un rol realiza una actividad su agente de actividades recibe esta acción y ejecuta los pasos necesarios para cumplir con los objetivos de esta tarea. Guarda los datos introducidos por el rol en las entidades de información correspondientes y avisa a

otro rol que puede continuar con su siguiente actividad en caso de que existiese una interacción entre dos roles.

Caso de uso: *informar actividades al AGP.*

Propósito: comunicar al AGP las actividades que debe realizar un rol para coordinar su participación en un proceso.

Descripción: cada rol tiene un agente de actividades que informa al AGP la tarea que va a realizar. Cuando un rol termina una actividad, el agente de estados obtiene información sobre el siguiente estado del rol, la actividad que va a realizar y la interfaz correspondiente para ejecutar esta actividad.

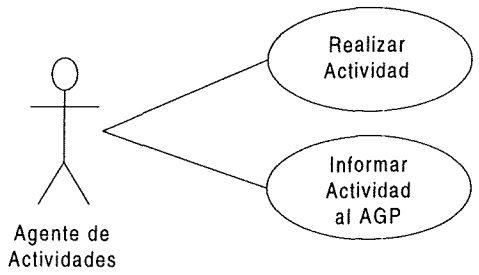

Figura 36. Casos de uso del agente de actividades.

El *agente de estados* mencionado anteriormente, también tiene un caso de uso asociado. En la Figura 37 se muestra este caso y enseguida se hace una descripción del mismo.

Caso de uso: *recuperar interfaces.*

Propósito: asociar al estado actual de un rol una interfaz, obtenerla y mostrarla al rol a través del sistema de coordinación.

Descripción: una vez que se tiene el estado actual de un rol, se busca la interfaz relacionada para enviarla al rol y que éste pueda realizar su actividad. Si la interfaz debe

mostrar cierta información a un rol, se leen las entidades necesarias para desplegarlas en el contenido de la misma.

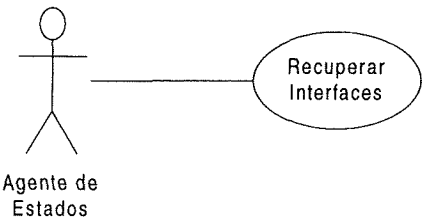

Figura 37. Casos de uso del agente de estados.

Por medio de los casos de uso hemos descrito la serie de actividades que llevan a cabo los elementos de la arquitectura. Enseguida presentamos su comportamiento por medio de los diagramas de secuencia.

V.5 Diagramas de secuencia

En la Figura 38 se presenta el diagrama de secuencia para el caso de uso generar roles. Se puede observar el mensaje *leerDoctoXML()* del objeto *agp1* representado por un rectángulo, que indica que se lea el modelo base de un proceso en XML en donde están definidos los roles, las actividades, los estados, las interacciones, entre otros. El objeto *agp1* lee el documento XML donde está definido un proceso y crea un objeto de tipo *RolBase* para cada uno de los roles encontrados en el modelo. Estos roles al ser creados se definen con las actividades y estados que les corresponden a través de los objetos *AgenteActividades* y *AgenteEstados*. Se puede observar también un objeto tipo *AnalizarProceso* que se encarga por un lado, de verificar que el modelo esté bien formado y sea válido, y por otro, extraer la información del proceso para generar los roles.

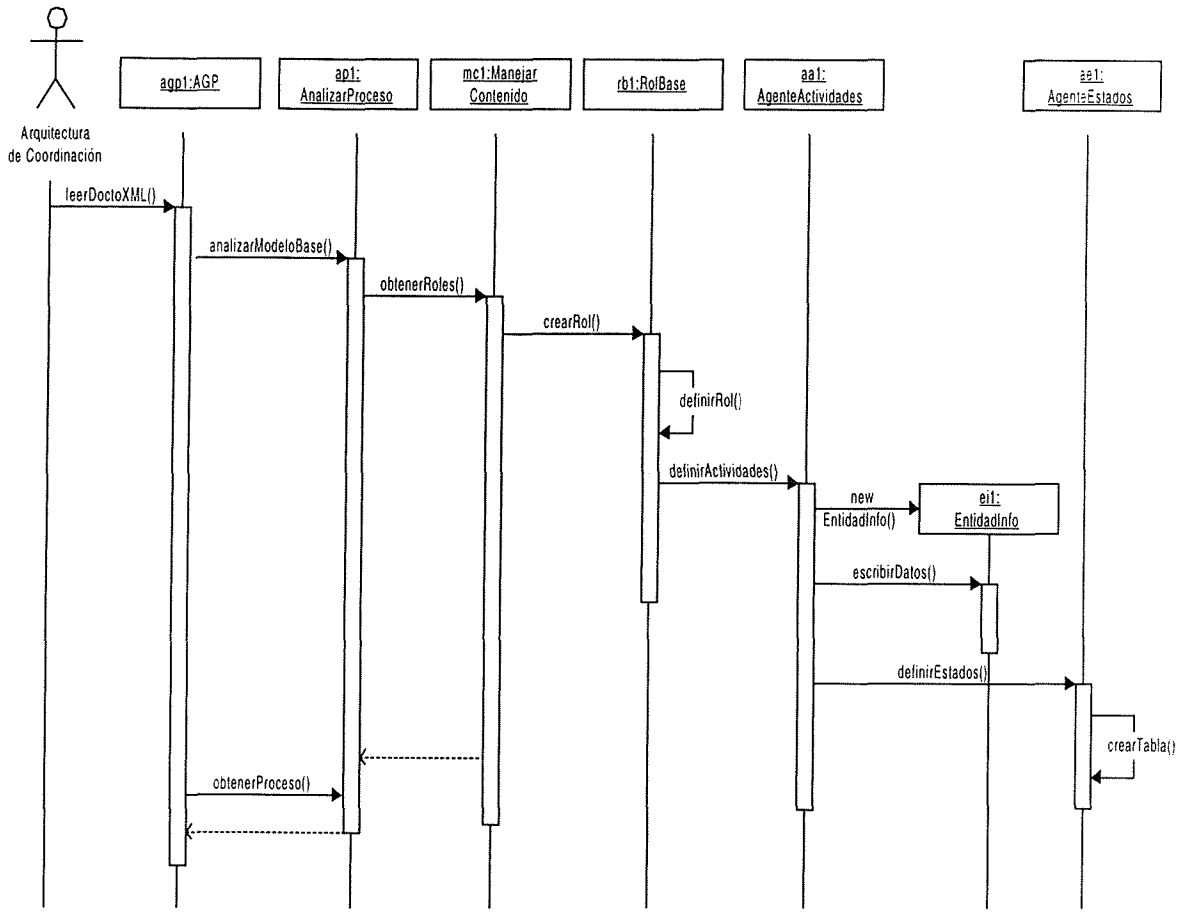

Figura 38. Diagrama de secuencia del caso de uso generar roles.

La Figura 39 muestra el diagrama de secuencia para el caso de uso *establecer comunicación entre roles* del administrador global de procesos. En el diagrama se observa un objeto de tipo *AGPS* que se encarga de recibir las peticiones que hacen cada uno de los roles de un proceso por medio de la *Web*. Al recibir la petición, se comunica con el objeto *agp1* a través del mensaje *comunicarRoles()* para informar a los roles que alguien ya terminó de ejecutar una actividad que tal vez era importante para que otros pudieran continuar con las propias. Si a partir de la actividad realizada por un rol, es necesario almacenar algún dato, éste se guarda en un objeto tipo *EntidadInfo*.

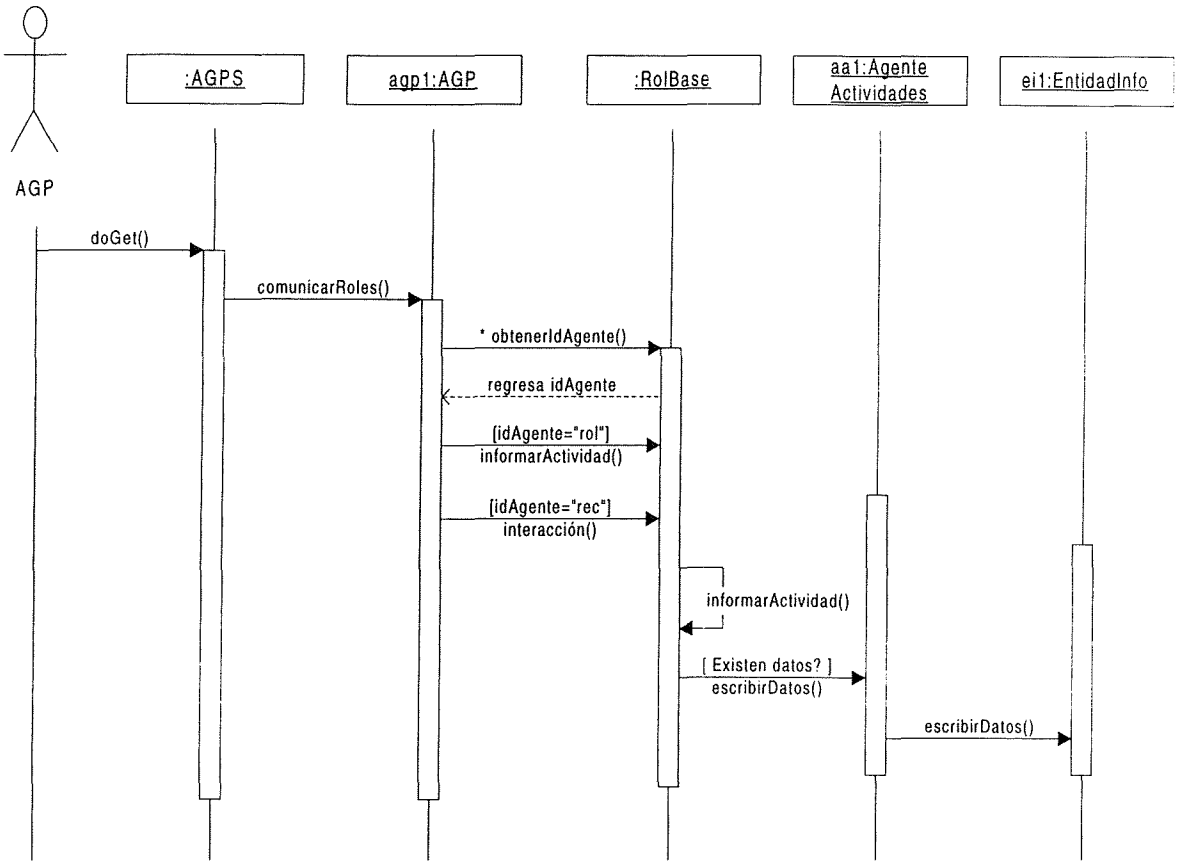

Figura 39. Diagrama de secuencia del caso de uso establecer comunicación entre roles.

El segundo diagrama de secuencia para el administrador global de procesos se muestra en la Figura 40 en la que se puede observar la serie de mensajes entre los objetos para llevar a cabo la *coordinación de un proceso*. Esto se realiza informando a los roles las actividades que deben ejecutar. El objeto *agp1* establece la comunicación entre los roles de un proceso. El *agp1* debe identificar al rol que va a coordinar a través del método *obtenerIdAgente()*. Para ello, se busca el identificador del rol en cuestión y se informa a su objeto *RolBase* la siguiente actividad. Como veíamos en la Figura 39 el *agp1* puede informar a un segundo rol que otro ya terminó su actividad para que éste continúe con la suya. Sin embargo, es en

el diagrama de la Figura 40 donde el *agp1* envía la siguiente actividad del rol, coordinando de esta manera su participación en el proceso.

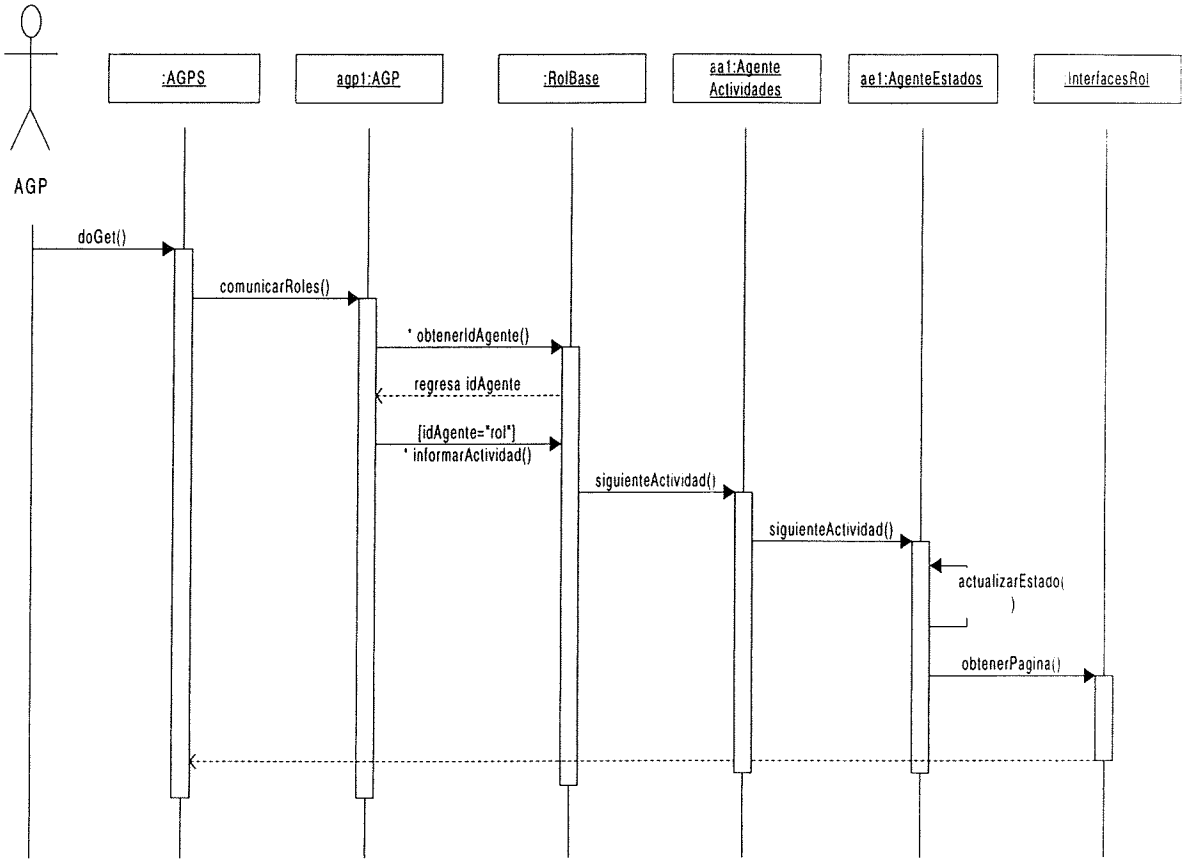

Figura 40. Diagrama de secuencia del caso de uso coordinar proceso.

En el apéndice B se anexan los diagramas de secuencia restantes que conforman la arquitectura, definiendo el comportamiento de los objetos y el paso de mensajes entre estos para llevar a cabo los casos de uso explicados anteriormente.

En la siguiente sección se presenta el diagrama de clases derivado de los diagramas de secuencias mostrados anteriormente, con el que representamos la estructura del sistema.

V.6 Diagrama de clases

La Figura 41 muestra el diagrama de clases de la arquitectura de coordinación. En la parte superior se encuentra la clase *AGPS* que es la interfaz entre el administrador global de procesos y los usuarios que jugarán el papel de un rol de un proceso por medio de su sistema de coordinación.

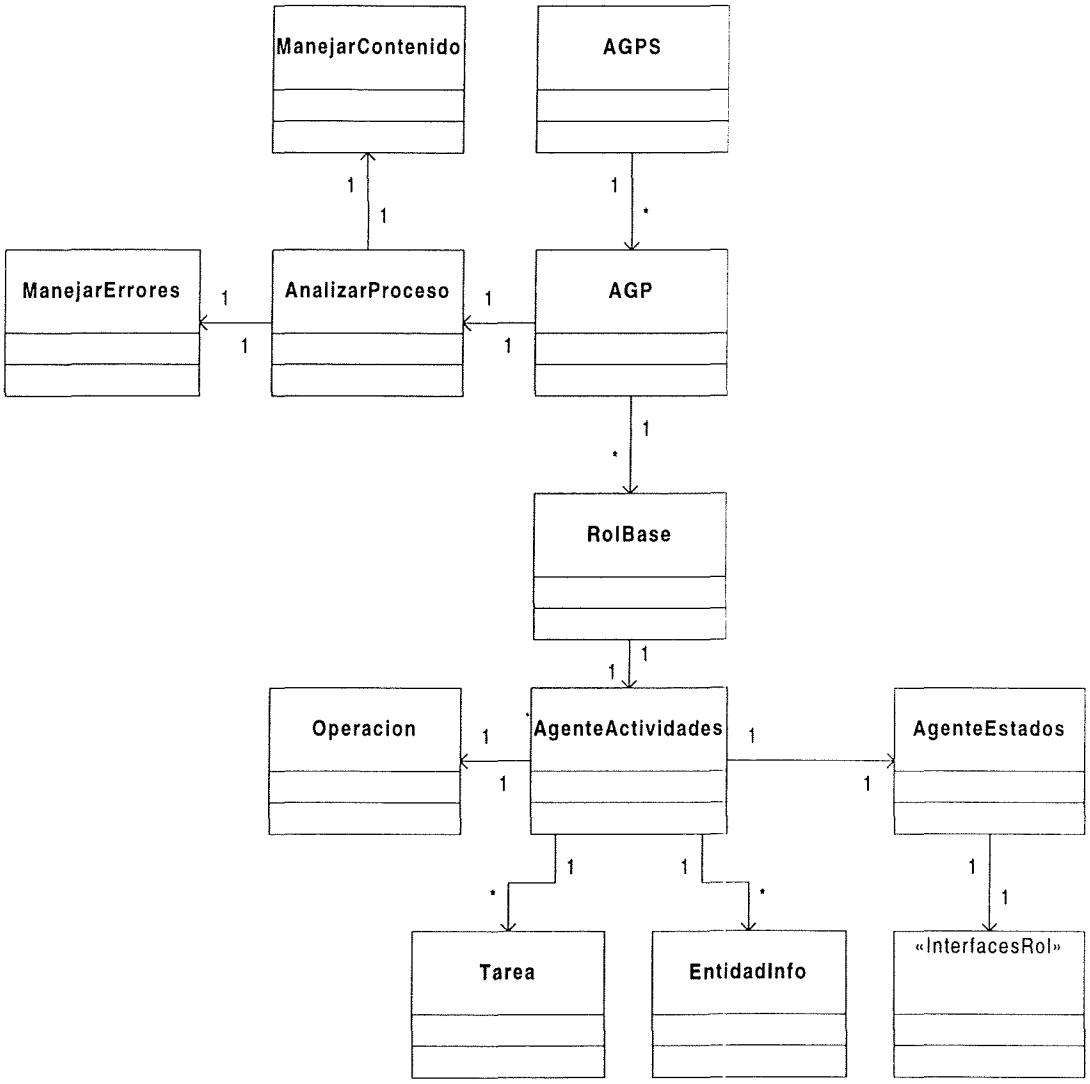

Figura 41. Diagrama de clases de la arquitectura de coordinación.

La clase *AGP* tiene como función coordinar los roles de un proceso organizacional. Esta clase tendrá asociada uno o más roles dependiendo del proceso. La clase *RolBase* permite definir las características específicas para cada uno de los roles. Esta clase está asociada con un agente de actividades que se encarga de comunicar al *AGP* las tareas que debe realizar un rol. De la misma forma existe una clase llamada *AgenteEstados* en donde se definen los estados por los que atraviesa un rol conforme va realizando sus actividades. Existen otras tres clases asociadas a *AgenteActividades* que permiten que ésta cumpla con su cometido. Por un lado, está la clase *Operación*, que contiene algunos métodos para realizar operaciones básicas como sumas, restas, etc., que son necesarias en alguna actividad; la clase *EntidadInfo* permite crear entidades de información que son importantes en el transcurso de la ejecución de un proceso. La clase *Tarea* permite ejecutar cada una de las actividades de un rol para que este alcance su objetivo. Por otra parte, la clase *AgenteEstados* está asociada con una clase llamada *<<InterfacesRol>>* la cual es un estereotipo que permite acceder a las interfaces que necesita un rol para ejecutar sus actividades. Estas interfaces pueden ser páginas *html* ó interfaces de actividades que no estén orientadas al *Web*.

En la Figura 42 se muestra el aspecto dinámico que presenta la arquitectura por medio de sus componentes.

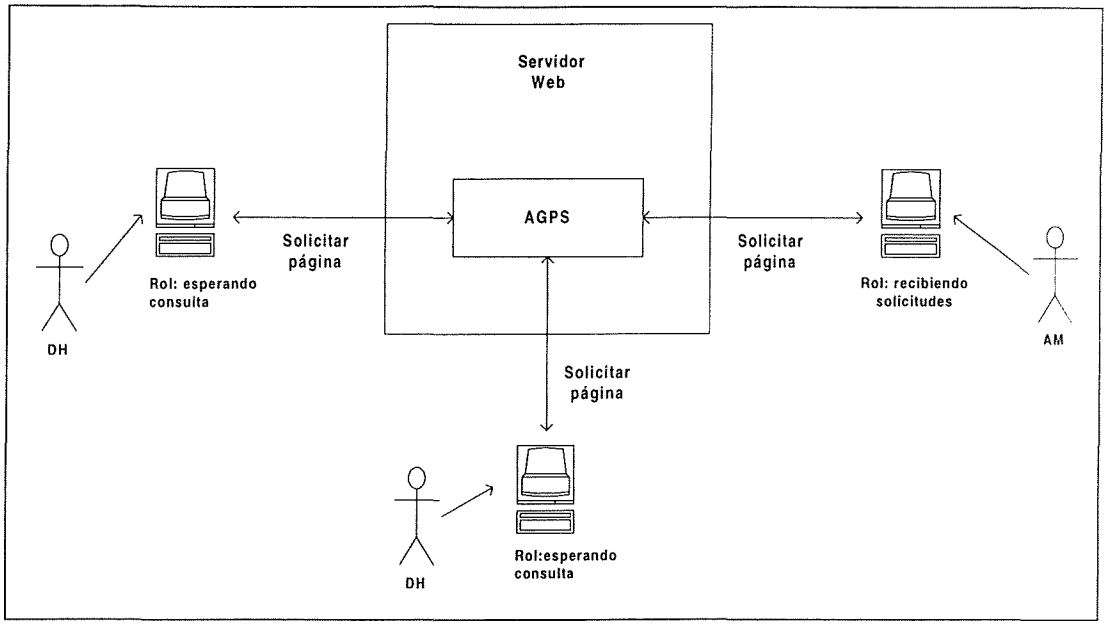

Figura 42. Aspecto dinámico de la arquitectura de coordinación.

Como se puede observar, se cuenta con un servidor *Web* que se encarga de enviar la información a través del sistema utilizando el protocolo de comunicación HTTP. Este servidor tiene soporte a la tecnología de *Servlets* con el propósito de atender las peticiones de los roles que están involucrados en un proceso y darles respuesta. Esta funcionalidad se realiza a través del AGPS, el cual se comunica internamente con el AGP que tiene la responsabilidad de llevar a cabo la coordinación de un proceso y sus respectivos roles.

Hasta el momento se ha definido el diseño que conforma a la arquitectura de coordinación, y en el apéndice B se presenta el diccionario de clases de este diagrama. En la siguiente sección, se establecen una serie de pasos a seguir con la finalidad de utilizar la arquitectura y generar el sistema de coordinación que dé soporte a un proceso.

V.7 Pasos a seguir para utilizar la arquitectura coordinación

El diagrama de actividades de la Figura 43 muestra los pasos que debe realizar el usuario para generar un sistema de coordinación utilizando la arquitectura descrita.

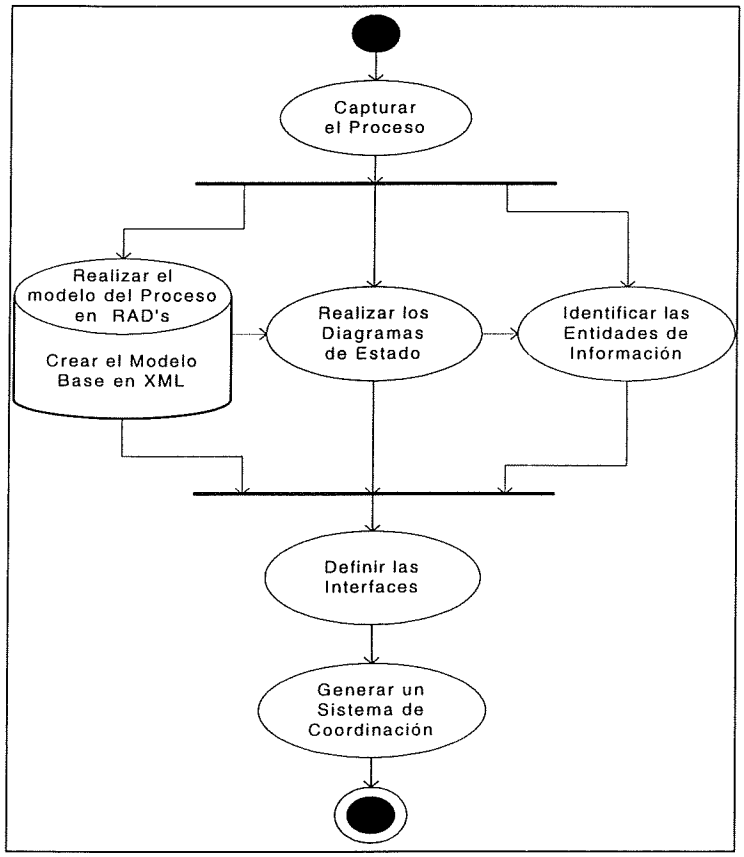

Figura 43. Diagrama de actividades del usuario.

Primeramente se tiene que *capturar el proceso* para identificar el objetivo principal que se desea alcanzar, los roles involucrados, las actividades realizadas por estos roles, como interactúan entre sí, las entidades de información que necesitan, para ser concretos, cómo se lleva a cabo el proceso. Una vez realizado lo anterior, el proceso debe ser documentado de acuerdo a la información obtenida. Después de terminar esta actividad, se puede observar en la Figura 43 una barra de sincronización horizontal de la cual se disparan tres actividades. Esta barra indica que las actividades pueden ejecutarse de manera paralela. La

actividad *realizar el modelo RAD del proceso* captura el proceso en una forma clara, identificando la información de acuerdo a las perspectivas de Curtis (Curtis, B., *et al.*, 1992): funcional, organizacional, comportamiento e informacional (no es cubierta explícitamente por los RADs) consideradas importantes por la información que arrojan sobre un proceso. Embebida en esta actividad se encuentra la tarea de generar el modelo base en XML del proceso, esto es posible gracias a la información que se puede obtener de los diagramas y las entidades de información identificadas para el proceso. En este modelo se define un proceso de la manera más completa posible para generar a partir de él, el sistema de coordinación que dé soporte al mismo. En la actividad *realizar diagramas de estados*, el usuario debe crear los diagramas para identificar el comportamiento que tienen los roles en un proceso ya que los diagramas rol-actividad (RAD) no muestran esta información explícitamente. La última actividad que se dispara de la barra de sincronización es *identificar las entidades de información* que ayudan al usuario a detectar los datos que intercambian los roles al realizar sus actividades.

Hasta este momento se han realizado las tres actividades que se derivan de la captura del proceso. Existe otra barra de sincronización a partir de la cual se lleva a cabo una actividad. Ésta es *definir las interfaces*, es decir, las páginas HTML con las cuales van a interactuar los roles del proceso para ejecutar las tareas que tienen asignadas. Una vez que se tiene esta información, se procede a generar el sistema de coordinación respectivo utilizando la arquitectura. En el apéndice B se muestra una guía más completa de estos pasos.

V.8 Conclusiones

En el presente capítulo se presentó una arquitectura de coordinación que facilita la creación de sistemas de soporte a procesos. Además, se describió el diseño de la misma mostrando las partes que la constituyen y se establecieron una serie de pasos que permiten utilizar la arquitectura para mejorar el desempeño de las organizaciones a través de sus procesos.

Esta arquitectura cubre los aspectos importantes involucrados en el enactment de procesos de tal forma que sea posible apoyar a las organizaciones con sistemas de soporte que contemplen los elementos mínimos requeridos para coordinar a los roles y sus actividades permitiendo alcanzar la meta global del proceso.

Durante el diseño se definió el contenido estructural que tendrá el modelo base de un proceso apoyados en el lenguaje de marcado XML. XML permite crear un tipo de documento llamado DTD (Document Type Definition) mediante el cual se fijan las reglas estructurales a seguir para mapear un diagrama RAD a un modelo textual (base) conteniendo además los elementos dinámicos necesarios para generar un sistema de coordinación. En el apéndice B se presenta el DTD utilizado para crear un modelo.

En el siguiente capítulo, se presentan las pruebas realizadas y los resultados obtenidos.

Capítulo VI. Pruebas y resultados de la arquitectura de coordinación

VI.1 Introducción

Verificación y validación (V y V) es el nombre genérico que se le da al proceso de pruebas para asegurar que el software contiene las especificaciones establecidas y satisface las necesidades de los clientes (Sommerville, I., 1996). La diferencia entre validación y verificación es que la primera evalúa si se está construyendo el producto adecuado, y la segunda, si se está construyendo en forma correcta.

En este capítulo se presentan las pruebas realizadas a la arquitectura de coordinación con la finalidad de identificar si se construyó en forma correcta y detectar los defectos que puedan existir para proponer las posibles soluciones. Estas pruebas se realizan siguiendo la secuencia de pasos que se han definido para generar un sistema de coordinación probando su funcionalidad que se obtiene de las clases implementadas. Así mismo, se presentan los resultados obtenidos de esta evaluación.

VI.2 Pruebas

La fase de pruebas es una parte importante dentro de todo proceso de desarrollo para verificar la funcionalidad de un producto e identificar sus fallas. En esta sección se definen los aspectos a probar de la arquitectura.

Para utilizar la arquitectura se han identificado dos tipos de usuarios: un administrador del sistema y los usuarios del sistema de coordinación (miembros del proceso organizacional al que se da soporte). El administrador se encarga de complementar el modelo base obtenido a partir del RAD de un proceso, agregando los elementos dinámicos (comportamiento de cada uno de los roles) necesarios y las entidades de información involucradas. Los diagramas de transición de estados realizados por el administrador reflejan el dinamismo del comportamiento de los roles en un proceso. También este usuario debe diseñar las interfaces que utilizan los roles para ejecutar sus actividades.

A continuación se mencionan las actividades a realizar para verificar si se están cumpliendo los requerimientos que demandan los casos de uso identificados en el análisis del sistema. Recordando que los casos de uso reflejan las actividades que se realizan a medida que avanza tanto el proceso de creación del sistema de coordinación como la funcionalidad que debe reflejar el mismo, se tiene lo siguiente:

- El administrador proporciona el modelo base de un proceso a la arquitectura de coordinación. Las actividades que realiza la arquitectura con este modelo son:
  - Leer el modelo base del proceso.
  - Generar las instancias necesarias para cada uno de los roles.

- El AGP de la arquitectura establece la comunicación entre los roles participantes en un proceso. Las actividades a realizar son:
  - Identificar los roles que interactúan en el transcurso de sus actividades.
  - Informar a un rol que puede continuar con sus actividades.

- El AGP lleva a cabo la coordinación de un proceso. Las actividades involucradas son:
  - Informar a un rol su siguiente actividad.
  - Accesar la tabla de estados para obtener el nuevo estado del rol.
- El Agente de Actividades facilita la realización de las tareas para un rol específico. Las actividades necesarias para esto son:
  - Actualizar las entidades de información relacionadas con una actividad.
  - Asignar a un rol la siguiente actividad que debe ejecutar.

- El Agente de Actividades informa al AGP la siguiente actividad de un rol. Las actividades involucradas son:
  - Obtener el siguiente estado del rol.
  - Enviar la actividad con su interfaz correspondiente para ejecutarla.

- El Agente de Estados recupera las interfaces necesarias para ejecutar las actividades de los roles. La actividad que debe realizar para este fin es:
  - Obtener la interfaz correspondiente al estado actual de un rol.

- El usuario (miembro de la organización) accesa el sistema de coordinación que da soporte al proceso en el cual está involucrado. Las actividades que debe realizar son:
  - Seleccionar el rol que juega en el proceso.
  - Ejecutar su rol.

- El usuario (miembro de la organización) sale del sistema de coordinación para continuar posteriormente con sus actividades. La acción a ejecutar es:
  - Terminar la sesión del usuario.

VI.3 Relación de la prueba con la funcionalidad del sistema

La Tabla V muestra la relación que existe entre la funcionalidad de la arquitectura de coordinación con respecto a las pruebas que se aplican para comprobar que se cumple satisfactoriamente con la actividad que se desea realizar.

Tabla V. Relación prueba – función.

Id	Prueba	Funcionalidad
A1	Leer el modelo base	Acceder a la ubicación del modelo y leerlo. Analizar la validez del modelo. Mostrar un mensaje de error cuando ocurra un problema.
A2	Generar las instancias necesarias para cada uno de los roles y sus elementos.	Crear objetos tipo RolBase. Construir la tabla de estados de los roles. Definir el agente de actividades. Definir el agente de estados. Asociar las interfaces con los estados de los roles.
A3	Identificar la interacción entre los roles.	Identificar el rol que inicia la interacción. Verificar la tabla de estados.
A4	Informar a un rol que continúe con su siguiente actividad.	Informar a un rol que puede proseguir con sus tareas debido a la acción realizada por otro rol.
A5	Actualizar entidades de información.	Almacenar los nuevos valores de una entidad.
A6	Asignar actividades.	Buscar la siguiente actividad de un rol.
A7	Obtener el siguiente estado del rol.	Leer la tabla de estados.
A8	Entregar al AGP la actividad e interfaz correspondiente a un rol.	Obtener el nuevo estado, la siguiente actividad y la interfaz correspondiente. Informar al AGP esta actividad.
A9	Obtener la interfaz de un rol.	Actualizar el estado. Leer la tabla de estados. Tomar la interfaz correspondiente al estado actual.
U1	Seleccionar un rol del proceso.	Recibe una interfaz con la actividad a realizar de acuerdo al estado del rol.
U2	Ejecutar su rol	El usuario realiza los pasos necesarios para terminar una actividad.
U3	Terminar la sesión del usuario.	Guardar el estado del rol para mantener su persistencia en el proceso. Desconectar al usuario del sistema.

VI.4 Metodología para realizar las pruebas.

Para la aplicación de las pruebas se utiliza uno de los modelos desarrollados para comprobar la funcionalidad de la arquitectura. En este caso se utiliza como ejemplo el subproceso del caso de estudio realizado en la Unidad de Medicina Familiar No. 32 (UMF32), en el área de atención médica (Apéndice A). El objetivo de este subproceso es obtener una cita para ser médicamente atendido. El subproceso involucra dos roles: el Derechohabiente (DH), que tiene como propósito obtener una cita; y la Asistente Médica (AM), quien otorgará la cita y asignará la fecha, hora y consultorio correspondiente. En lo subsecuente, a este subproceso se le llamará proceso "*otorgar cita*". Los pasos a seguir para realizar las pruebas serán los siguientes:

- En primer lugar, se modelará el proceso en RAD generando su modelo base, contemplando todos los elementos necesarios para generar el sistema de coordinación (por ejemplo, la transición de estados).
- En segundo término, se colocará el modelo base en un lugar específico dentro de un servidor de *Web* para que la arquitectura pueda leerlo. Se hace lo mismo con las interfaces asociadas (páginas HTML) a cada uno de los estados.
- Como tercer paso, se utilizará la arquitectura para generar el sistema de coordinación respectivo.
- Enseguida se ejecutarán los roles del proceso a través del sistema que le da soporte y se buscan todos los defectos posibles.
- Por último, se registran los resultados y observaciones realizadas durante la ejecución del sistema.

Las pruebas se enfocan principalmente a que se cumpla con los requerimientos específicos de funcionalidad establecidos en el análisis del sistema (pruebas de caja negra). Las pruebas de caja negra se enfocan en los requerimientos funcionales (Pressman, Roger S., 1997). Esto es, se crea un conjunto de condiciones de entrada que probarán los requerimientos funcionales de un programa.

VI.5 Fase de prueba

En esta sección se mostrará la fase de prueba aplicada al proceso *otorgar cita* de la UMF32, presentando algunas de las interfaces utilizadas para cumplir con este objetivo.

VI.5.1 Definición del modelo base del proceso

La definición de este modelo se realizó utilizando el Lenguaje XML ya que éste permite definir un documento de manera estructurada facilitando la captura del modelo RAD de un proceso. En la Figura 6 del capítulo II se presenta el modelo en RAD del proceso *otorgar cita*. A partir de éste, se generó el modelo base agregando los elementos dinámicos explicados al inicio de este capítulo, tomando esta información de los diagramas de transición de estados utilizados para identificar el comportamiento de los roles, además de definir las entidades de información involucradas en el proceso. En la Figura 44 se presenta la transición de estados del DH, en la cual se observa el estado inicial *entregando* que indica que el DH proporciona la tarjeta de citas a la AM. Cuando realiza esta acción, su estado cambia a *esperando* mientras recibe una respuesta. Cuando ésta llega, el estado se actualiza a *recibiendo*, indicando que ha llegado la respuesta a su solicitud. Finalmente, el rol termina sus actividades (estado *terminando*).

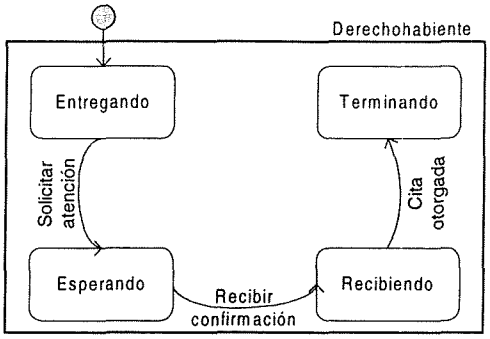

Figura 44. Diagrama de transición de estados del Derechohabiente.

En la Figura 45 se muestra una parte del modelo base realizado para el proceso *otorgar cita*, en donde se definen algunas de las actividades del rol DH.

```
<rol nombre="esperando consulta">
    <agente quien="b">Derechohabiente (dh)</agente>
    <act id="b1">
        <nom>Entregar tarjeta</nom>
        <edoactual>entregando</edoactual>
        <evento>entrega tarjeta de citas </evento>
        <edosigui>esperando</edosigui>
        <interaccion cual="a1" tipo="E">a</interaccion>
    </act>
    <act id="b2">
        <nom>Esperando respuesta</nom>
        <edoactual>esperando</edoactual>
        <evento>informa datos de la cita</evento>
        <edosigui>recibiendo</edosigui>
        <interaccion cual="a5" tipo="R">a</interaccion>
        <cadena>afiliacion</cadena>
    </act>
    .
    .
    .
    </rol>
```

Figura 45. Se muestra parte del modelo base del Derechohabiente.

De la misma manera está definido el rol de la AM para completar el modelo de este proceso quedando listo para ser utilizado por la arquitectura.

Para último, se deben diseñar las interfaces que permitan a los roles ejecutar sus actividades. Estas interfaces serán colocadas en un lugar específico para que puedan ser accesadas por el sistema de coordinación.

VI.5.2 Almacenamiento del modelo en el servidor

El modelo se almacena específicamente dentro del directorio "/servlets" del servidor *Web* para permitir el acceso de la arquitectura a la información del proceso. La arquitectura lee el documento XML que contiene el modelo y verifica que esté exento de errores sintácticos y semánticos para que pueda ser interpretado. De la misma manera, las interfaces de los roles también se almacenan en el servidor ya que serán las que permitan ejecutar las actividades desde el *Web*.

Algunos errores encontrados en esta fase están relacionados con la ubicación incorrecta de los modelos en el servidor, debido a que los nombres de las páginas HTML (interfaces) no son los correctos, ocasionando que el sistema no pueda localizarlas, o que la localización de estas páginas también es incorrecta.

VI.5.3 Construcción del sistema de coordinación

Una vez definido el modelo del proceso y las interfaces necesarias para llevar a cabo las actividades, se agregan estos modelos a una interfaz (Figura 46) que contiene una lista de los procesos existentes y a los cuales se da soporte. Cuando se selecciona uno de los modelos, se envían unos parámetros al servidor, que a su vez los pasa a la arquitectura, que contienen el nombre del proceso seleccionado para crear el sistema de coordinación y la ruta donde se encuentran las interfaces de ese proceso. La arquitectura lee el modelo y

verifica mediante un analizador sintáctico de XML que se encuentre bien formado y sea válido, de acuerdo a lo especificado en el documento DTD (Apéndice B). Si el modelo cumple con estos requisitos, entonces se empieza a crear el sistema de soporte agregando al administrador global de procesos las instancias de los roles involucrados conforme se lee el modelo. Cada rol contendrá los componentes necesarios mencionados en el capítulo V que le permitirán ejecutar las actividades asignadas. Una vez creado el sistema de coordinación, aparece una interfaz que contiene los roles del proceso, los cuales pueden ser seleccionados por un usuario para realizar las actividades de un rol.

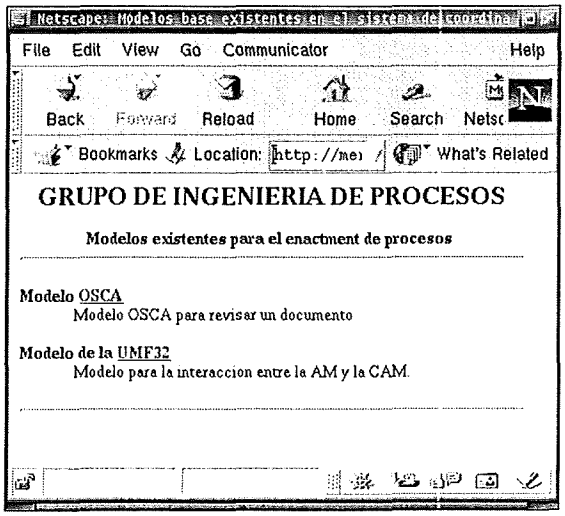

Figura 46. Interfaz principal que contiene modelos base de procesos.

Por ejemplo, al seleccionar la liga UMF32, se envían unos parámetros a la arquitectura de coordinación para indicarle que se construya el sistema que dé soporte a ese proceso. Si el sistema se genera sin problemas, entonces se presenta una interfaz con una lista de los roles participantes en el proceso (en este caso, del proceso *otorgar cita*). La Figura 47 presenta esta interfaz con dos opciones: Derechohabiente y Asistente Médica.

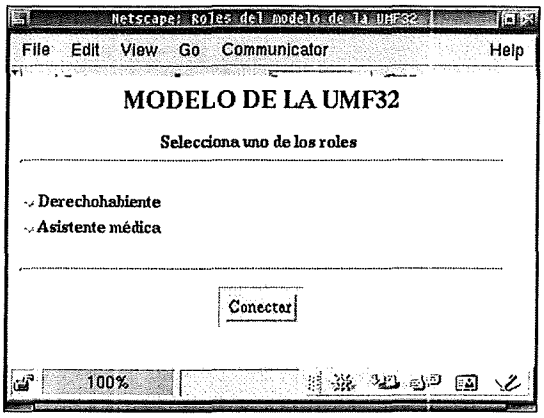

Figura 47. Interfaz principal del proceso *otorgar cita* de la UMF32.

En esta fase se detectaron algunos defectos o limitaciones en la arquitectura. Por mencionar algunos, cuando un rol ejecuta una actividad y presiona un botón para pasar a la siguiente, pueden suceder dos cosas: a) se muestra un mensaje de error indicando que no se encuentra la interfaz para realizar la actividad o b) el sistema le entrega la interfaz de una actividad de otro proceso. El problema se debe a que el *Servlet* utilizado para acceder al sistema de coordinación de un proceso no permite la ejecución simultánea de varios sistemas de procesos diferentes. Este inconveniente se puede solucionar creando diferentes sesiones de trabajo para cada sistema de coordinación creado.

Una limitante de la arquitectura es que sólo permite el manejo de una entidad de información por actividad. Esta limitante es porque el esfuerzo realizado en el desarrollo de la arquitectura se centró más en proporcionar el ambiente de coordinación apropiado para los roles y sus procesos, sin adentrarse a fondo en la parte informacional.

VI.5.4 Utilización del sistema de coordinación

En esta sección se explican algunas actividades realizadas por los roles DH y AM, las cuales permitirán cumplir con el objetivo final del proceso que es dar una cita al DH para que sea atendido. En la Figura 48 se presenta la primera interfaz del DH en el estado *entregando*, la cual tiene un campo para introducir su número de afiliación (32984 en este ejemplo). Una vez introducido este dato, se envía la solicitud presionando el botón enviar y la interfaz de este rol cambia al estado *esperando*.

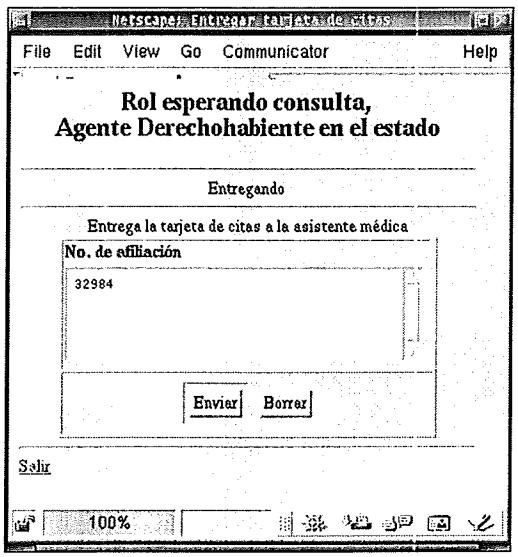

Figura 48. Interfaz del rol DH en el estado *entregando*.

Por su parte, el estado inicial de la AM es *esperando*. Este estado significa que la AM espera las solicitudes de citas de los DHs. La Figura 49 muestra la interfaz correspondiente.

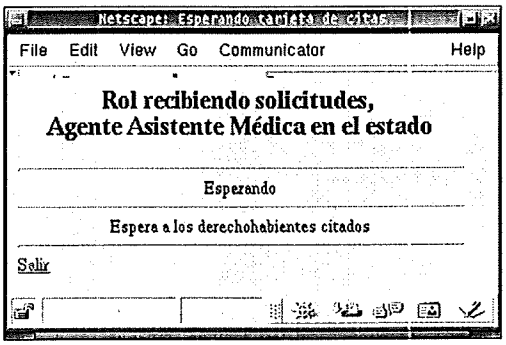

Figura 49. Interfaz del rol AM en el estado *esperando*.

Cuando la AM recibe la petición de una cita, su estado cambia a *verificando* (Figura 50). En este estado la AM ya recibió la solicitud realizada por el DH, y procede a verificar que esté afiliado a la institución. Si esto es cierto, entonces verifica si es un trabajador de la misma UMF32 o de otro lugar, ya que las actividades que debe realizar no son las mismas.

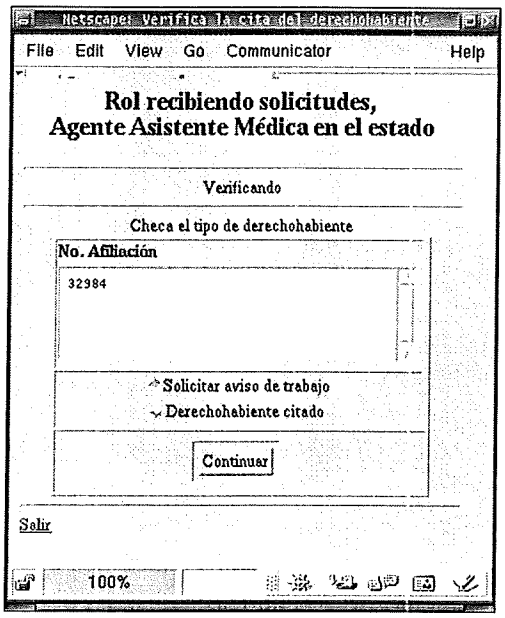

Figura 50. Interfaz del rol AM en el estado *verificando*.

Si el DH es trabajador de la UMF32, entonces aparece la interfaz de la Figura 51 donde se le solicita un comprobante de trabajo que demuestre que labora en esa institución.

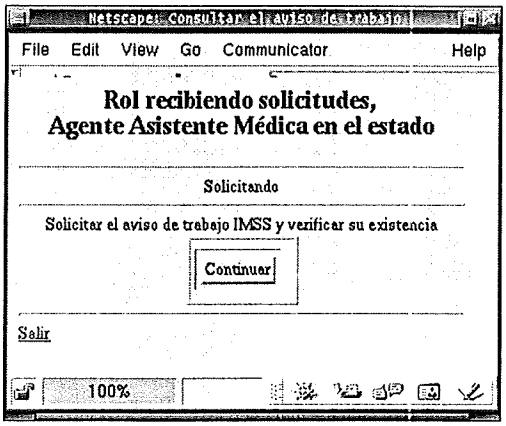

Figura 51. Interfaz del rol AM en el estado *solicitando*, cuando el DH es un trabajador de la UMF32.

Posteriormente, la AM envía la hora de la cita y el número de consultorio en el que será atendido el DH. En este ejemplo, se está citando al DH a las 16:30 en el consultorio No. 4 (Figura 52).

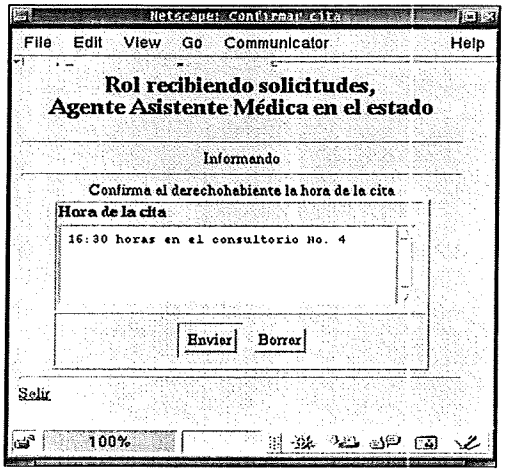

Figura 52. Interfaz del rol AM en el estado informando.

Si el DH no es trabajador de la UMF32, simplemente la AM selecciona la opción *Derechohabiente citado* de la Figura 50, checa la lista de citas (forma 4-30-6) y continua con la actividad de confirmar cita que se muestra en la Figura 52. Finalmente, el DH recibe el aviso de la cita para consulta médica mostrado en la Figura 53.

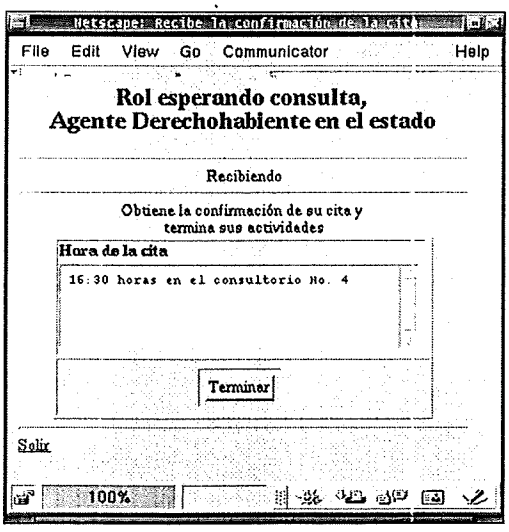

Figura 53. Interfaz del rol DH en el estado *recibiendo*.

VI.6 Resultados de las pruebas

En esta sección se presentan los resultados obtenidos del proceso de pruebas realizado a la arquitectura de coordinación. Los factores evaluados en esta prueba son el tipo de defecto y la severidad del mismo. Para el tipo de defecto se consideran los siguiente aspectos:

- Coordinación, si la interacción entre los roles de un proceso se realiza de manera adecuada.
- Interfaz, se refiere a la funcionalidad que ésta proporciona para que los roles ejecuten las actividades.
- Congruencia, esto significa que las entidades de información utilizadas corresponden a la actividad que se va a realizar.
- Persistencia, si el estado de los roles se mantiene aún cuando no están conectados al sistema.
- Seguridad, que los usuarios realicen actividades no permitidas o la funcionalidad del sistema no sea la esperada.
- Documentación, cuando no se muestra la información apropiadamente para entender lo que está pasando en el sistema.

En el aspecto de severidad del defecto se consideran dos clases:

- Menor, se refiere a que no se afecta el funcionamiento del sistema y que pueden ser corregidos con cambios mínimos.
- Mayor, en este se ve afectado el funcionamiento del sistema y los cambios a realizar involucran más trabajo y tiempo de programación.

En la Tabla VI se muestran los defectos encontrados derivados de las pruebas realizadas a la arquitectura de coordinación.

Tabla VI. Lista de defectos encontrados en la arquitectura de coordinación.

No.	Descripción	Prueba	Severidad	Tipo
1	No se muestran los errores de estructura del modelo base al ser analizado por la arquitectura.	A1	Menor	Seguridad
2	No se da un mensaje de error cuando hace falta una interfaz a un rol para que realice una actividad.	A1	Menor	Seguridad Interfaz
3	No se muestra una lista de los sistemas que ya han sido generados por la arquitectura	A2	Menor	Documentación
4	No se sabe si los demás roles de un proceso están conectados al sistema.	A3	Menor	Interfaz Coordinación Documentación
5	Se maneja una sola entidad de información por actividad.	A5	Menor	Seguridad
6	Cuando un rol realiza una actividad y presiona un botón para pasar a la siguiente, el sistema envía la actividad del rol de un proceso diferente.	A6	Mayor	Congruencia Seguridad Persistencia
7	La presentación de los datos en las interfaces se hacen a través de cuadros tipo áreas de texto (HTML).	A9	Menor	Interfaz Documentación
8	Un rol puede ser ejecutado por más de un usuario al mismo tiempo.	U1	Menor	Seguridad Interfaces
9	No es posible trabajar con dos sistemas de coordinación al mismo tiempo.	U2	Mayor	Seguridad Coordinación Congruencia

Los defectos anteriores fueron encontrados en la arquitectura de coordinación al momento de generar los sistemas para dar soporte a procesos y también al probar estos sistemas representando el papel de uno de los roles involucrados en los mismos. Se verificó si realmente se cumplía con las expectativas que se tenían como: la coordinación eficiente de los roles, la persistencia del estado de los roles en el sistema, la ejecución de actividades, entre otras.

Por otra parte, en el Capítulo III se definieron las características que debía cumplir una arquitectura para llevar a cabo su funcionalidad en el contexto de sistemas de soporte a

procesos organizacionales, de las cuales se seleccionaron aquellas que se deseaba satisfacer con la arquitectura propuesta. En la Tabla VII se presenta finalmente los requisitos que se satisfacen en este trabajo.

Tabla VII. Se muestran las características que se están satisfaciendo con la arquitectura.

Nivel	Característica	Nuestra Propuesta
1	Persistencia	√
1	Comunicación entre Roles	√
1	Coordinación entre Roles	√
1	Manejo de Estados	√
1	Manejo de Información	√
1	Controlador de Actividades	√
1	Ejecución Asíncrona	√
1	Roles	√
1	Actividades	√
1	Independencia de Plataforma	√
1	Orden	√
1	Responsabilidad en las Decisiones	√
1	Decisiones	√
1	Metas	√

Nivel	Característica	Nuestra Propuesta
2	Presencia	X
2	Síncrona	√
2	Rol Base	√
2	Tipo de Arquitectura	Centralizada
2	Lenguaje de Programación	X
2	Modelo Reusable	√
2	Interacción con el Web	√
2	Integración con otras Aplicaciones	X
2	Generación Automática del Programa	½
3	Integración de Herramientas	X

√ cuenta con esta característica.

X no cuenta con esta característica.

½ generación semiautomática

Como se puede observar, se cumple algunos de los requisitos establecidos inicialmente tal como utilizar una arquitectura centralizada para lo cual se apoya en la tecnología *Web*, satisfacer la necesidad de mantener la persistencia de los estados de los roles en los procesos, permitir el manejo de entidades de información para ejecutar las actividades, llevar a cabo la coordinación de cada uno de los roles ya sea con sus actividades

particulares o la asignación de una nueva actividad a un rol por acción de otro y utilizar el concepto de "estado" para controlar el comportamiento de los roles en un proceso.

En esta arquitectura no se contempló la idea de manejar un lenguaje de programación para desarrollar un sistema de coordinación, sino que utilizamos el lenguaje XML para representar un proceso en un modelo formal y a partir de éste, generar el sistema respectivo. El estado actual de la arquitectura no realiza una generación automática de todo el sistema de coordinación porque la creación de interfaces se realiza de manera separada, sin embargo, el ideal que se espera alcanzar es integrar todos los componentes completamente. Finalmente, no se integraron otras herramientas a la arquitectura de coordinación que apoyaran el trabajo de los roles en el proceso, ya que este proyecto sólo enfocó en la fase de coordinación de los roles.

VI.6.1 Valoración de resultados

Una manera de estimar los resultados obtenidos es realizando un análisis que permita detectar el grado de eficiencia del sistema evaluado. Analizando la Tabla VI, se observa que son pocos los errores considerados como mayores que constituyen un esfuerzo más grande de programación para resolverlos, en comparación con los defectos considerados como menores. La cantidad de estos últimos no afecta de manera significativa el funcionamiento de la arquitectura, siendo éstos problemas de presentación en las interfaces (usabilidad), el tipo de entidades de información que soporta la arquitectura, entre otros.

Aún con los defectos encontrados que se presentan bajo condiciones muy específicas, se puede decir que la arquitectura funciona de una forma aceptable, considerando como

aceptable: el hecho que la generación del sistema de coordinación de un proceso está presente, que se mantiene la persistencia del mismo y que la asignación de actividades a los roles se realiza de acuerdo con lo especificado en el modelo base.

VI.6.2 Solución a los defectos encontrados

En esta sección se dá una serie de posibles soluciones a los defectos arrojados por las pruebas, con el propósito de proporcionar un camino de como resolverlos. La Tabla VIII está formada por tres columnas: causa, que provoca el problema; defecto, cual es la situación que presenta el mal funcionamiento; y solución, una propuesta para resolver el problema de funcionalidad de la arquitectura.

Tabla VIII. Tabla que muestra posibles soluciones a los defectos encontrados.

Defecto	Causa	Posible solución
No se muestran los errores de estructura del modelo base al ser analizado por la arquitectura.	Los mensajes de error están direccionados a la salida estándar del sistema operativo.	Redireccionar los mensajes de error a la salida HTTP. Guardarlos en un archivo de texto.
No se muestra la interfaz a un rol para que realice una actividad.	No se creó la página HTML para la actividad de ese rol.	Crear la página HTML ó colocarla en el lugar correcto.
No se muestran los sistemas que ya han sido generados.	No se ha agregado a la página principal una lista de sistemas.	Agregar en la página principal una lista de los sistemas creados.
No se tiene consciencia presencial de los roles en el sistema.	No se cambia la interfaz inicial de un modelo creado.	Cambiar el estado de un rol a seleccionado en la interfaz inicial.
Se maneja una sola entidad de información por actividad.	Falta especificar la capacidad de recibir más entidades.	Crear un vector de entidades por actividad y definirlas en este vector
Cuando un rol termina una actividad, se le envía la actividad de otro rol diferente.	Generación de dos o más sistemas de coordinación al mismo tiempo.	Guardar cada sistema de coord. en una sesión de trabajo diferente dentro del sistema.
La presentación de los datos en las interfaces se hacen a través de cuadros tipo áreas de texto de HTML.	Así se consideró desde un inicio, por la longitud que pueden llegar a tener los campos.	Agregar a las interfaces cuadros de texto para desplegar la información y agregar soporte para este tipo de campos.
No es posible trabajar con dos sistemas de coordinación al mismo tiempo.	Los sistemas de coordinación no se guardan en sesiones de trabajo distintas.	Almacenar estos sistemas en áreas de trabajo diferentes.
Un rol puede ser ejecutado por más de un usuario al mismo tiempo.	Falta inhabilitar la opción de selección del rol en la interfaz.	Cuando se selecciona un rol, cambiar la pantalla con la opción inhabilitada.

VI.7 Conclusiones

Las pruebas realizadas han sido útiles porque ayudaron a detectar algunos defectos que minimizan en mayor o menor escala la funcionalidad del sistema. Se ha visto que las pruebas son un paso importante en el desarrollo de software pues entre más defectos se encuentren y sean solucionados, aumenta la calidad del producto y la funcionalidad que presenta se acerca más a la esperada por los clientes. Por esta razón, las pruebas dan la pauta para encontrar algunos problemas y proponer posibles soluciones en beneficio de la arquitectura de coordinación propuesta.

En el siguiente capítulo se presentan las conclusiones obtenidas con la realización de este proyecto de tesis y el trabajo a futuro que puede realizarse dentro del área de ingeniería de procesos, en particular el soporte a los mismos mediante el uso de sistemas de coordinación, sistemas de información, sistemas de flujo de trabajo, sistemas de simulación, entre otros.

Capítulo VII. Conclusiones y trabajo futuro

VII.1 Conclusiones

En este trabajo se presentó una arquitectura de coordinación para dar soporte a procesos organizacionales apoyados en la tecnología de "Internet", específicamente la "*Web*", bajo el ambiente cliente-servidor. Esta arquitectura permite el desarrollo de sistemas de coordinación de procesos de manera guiada y rápida. Guiada, porque el usuario no tiene que aprender un nuevo lenguaje de programación para desarrollar un sistema, y rápida, porque la generación del mismo está basada en la reusabilidad de la información que proporcionan los diagramas RAD plasmada en un modelo formal (base).

Con el propósito de identificar los requerimientos funcionales de la arquitectura y los componentes que la conforman, se desarrolló un prototipo del modelo de la cerveza y se llevó a cabo un caso de estudio en la Unidad de Medicina Familiar No. 32 de Ensenada, B.C. El caso de estudio fue de ayuda porque permitió identificar la problemática que presenta un proceso real cuando se desea aplicar algún tipo de soporte mediante TI. Se observó que no siempre es posible dar soporte a todas las partes que conforman un proceso, ya sea por la complejidad que representa o por no estar bien definidas las responsabilidades de cada uno de los roles. Sin embargo, se pudo detectar que hay partes en un proceso que sí son factibles de apoyar con tecnología cuando son claras las actividades que debe realizar cada rol y cuando se sabe con certeza cuales son los roles que colaboran entre sí para alcanzar los objetivos de la organización.

Por otro lado, el prototipo ayudó a identificar y determinar los componentes que integran la arquitectura de coordinación para que sea funcional, además sirvió para seleccionar la tecnología más apropiada de acuerdo a los objetivos planteados. La tecnología que se propone es utilizar un servidor *Web* con soporte a *Servlets* como una interfaz entre los usuarios que representan un rol en el proceso y el sistema de coordinación que apoyará la ejecución del mismo. Esto involucra la utilización de documentos HTML mediante las cuales los roles puedan realizar sus actividades. Este ambiente centralizado facilita el control de un proceso y las entidades de información (datos, documentos, etc.) requeridos. La implementación se llevó a cabo utilizando el lenguaje Java y los *Servlets*, ya que Java tiene la ventaja de acceder con facilidad a todas las clases y métodos de los *Servlets* sin problemas, es decir, la utilización de Java y *Servlets* no conlleva a riesgos tecnológicos de integración. Los *Servlets* por su parte, tienen la característica de minimizar la carga de trabajo del servidor porque a cada petición que recibe le asigna un hilo de ejecución diferente haciendo que la respuesta sea más rápida y que la cantidad de usuarios que se puedan atender al mismo tiempo sea mayor comparada con otras tecnologías como los programas CGI's (Common Gateway Interface).

Derivado de este prototipo surgieron algunos aspectos interesantes. En primer lugar, se vió la necesidad de contar con un administrador global de procesos para coordinar los esfuerzos e interacciones de cada uno de los roles. Además, se le dio a cada rol la responsabilidad de controlar las actividades que realiza en el proceso para hacer más eficiente y fácil la coordinación de los roles.

En segundo término, se observó la importancia del manejo de estados para coordinar el comportamiento de los roles y asignar las interfaces adecuadas para que éstos lleven a cabo la ejecución de sus actividades.

Dentro de las aportaciones de este trabajo se encuentran:

La definición de un modelo base en XML para representar el modelo de un proceso en forma estructurada y utilizarlo en la arquitectura para construir un sistema de coordinación. Este modelo tiene la propiedad de ser portable, ya que se puede utilizar en cualquier plataforma que tenga soporte de XML, debido a que es archivo de texto en forma estructura de acuerdo a las reglas establecidas en un documento conocido como DTD. Además, el modelo es reusable porque se pueden generar otro tipo de sistemas de soporte a procesos a partir de este modelo. Por ejemplo, generando sistemas de simulación para detectar posibles áreas problemáticas, agregando al modelo los elementos necesarios para construir este tipo de sistemas.

También podemos hablar del concepto de estados como un mecanismo para que los roles se encarguen de coordinar sus responsabilidades cumpliendo entre todos con el objetivo global de la organización. El diagrama RAD de un proceso tiene la información de los estados de manera implícita y nosotros la retomamos por medio de diagramas de transición de estados para entender el comportamiento de los roles a medida que ejecutan sus actividades y poder determinar cual es la secuencia de las mismas.

El utilizar la arquitectura para generar sistemas de coordinación implica que se reduzca el esfuerzo de desarrollo, ya que no se utiliza un lenguaje específico de programación como

tal, mediante el cual se defina cómo realizar las interacciones, cómo llevar a cabo el intercambio de información, cómo coordinar los roles, entre otros aspectos; sino que este trabajo lo realiza la arquitectura y simplemente los usuarios tienen la tarea de definir en un modelo base el proceso al que desean dar soporte. De esta manera, no se tiene que aprender un nuevo lenguaje de programación y el ciclo de desarrollo de software se minimiza porque el sistema no se implementa desde cero.

Por último, para generar un sistema de coordinación, se estableció una guía especificando los pasos a seguir, de tal manera que una vez cumplidos se tenga un sistema que dé soporte a un proceso. Esto permite que los roles realicen sus actividades de manera más eficiente y rápida mediante el uso de TI alcanzando los objetivos de la organización.

VII.2 Trabajo futuro

El estado actual de la arquitectura de coordinación, así como en la mayoría de los sistemas que se desarrollan, permite que su funcionalidad siga evolucionando y mejorando para hacer más fácil la generación de sistemas. Por lo tanto, a continuación se presentan algunas propuestas de mejoras.

- En la parte informacional, actualmente se están utilizando objetos de datos o entidades para manejar la información de un proceso, por lo tanto, se propone hacer más robusta esta parte, permitiendo el uso de documentos complejos o grandes como los de word, excel, bases de datos, etc., con el propósito de que se facilite el intercambio de información entre los roles participantes en un proceso. Para esto será necesario

establecer un mecanismo apropiado de acceso y entrega de documentos dentro de la arquitectura.

- Utilizar otro tipo de aplicaciones o sistemas para apoyar el trabajo realizado por los roles. Por ejemplo, correr aplicaciones como el word, excel, bases de datos, sistemas de información, etc., ya sea soportado por la arquitectura (internamente) o comunicándose con esas aplicaciones para extraer información o trabajar con ella.
- Agregar un mecanismo para mantener la persistencia de un proceso, aún después de que el servidor de *Web* que se utiliza en el sistema de coordinación termine su ejecución de manera inesperada. Esto se debe a que la persistencia del proceso se mantiene mientras el servidor de *Web* se encuentra funcionando. Lo anterior puede resolverse si se almacenan físicamente en el servidor los objetos utilizados en el sistema, guardando la información del proceso en una base de datos para después recuperarla o definiendo un documento XML con la estructura necesaria para guardar la información actual de un proceso, y que se cargue en el sistema cuando se necesite ejecutar nuevamente.
- Integrar la arquitectura a la herramienta CAPETool para que se vayan conformando los módulos que darán soporte a cada una de las etapas de la Ingeniería de Procesos.
- Crear un editor gráfico para construir las interfaces de los roles participantes en un proceso, que permita agregar los elementos necesarios para ejecutar una actividad como botones, cuadros de texto, campos ocultos, etc.
- Desarrollar un mecanismo que permita relacionar de manera semiautomática la información que proporcionan los diagramas de transición de estados al modelo base de un proceso.

Literatura citada

Bruynooghe, R., Parker, J. y Rowles, J. 1991. *A System for Process Enactment.* Publicado por IEEE Computer Society Press.

Chan, Mark C. 1997. *1001 Tips para Programar con Java.* Editorial Mc Graw Hill.

Chan, Stephen L. 2000. *Information Technology in Business Processes.* Business Process Management Journal. Vol 6. No. 3.

Chan, Peng S. y Land, Carl. 1999. *Implementing Reengineering using Information Technology.* Business Process Management Journal. Vol 5. No. 4. Páginas 311-324.

Curtis, B., Kellner, M. y Over, J. 1992. *Process Modelling,* Communications of the ACM. Vol. 15, No. 9.

Feiler, Peter H. and Humphrey, Watts S. Febrero, 1993. *Software Process Development and Enactment: Concepts and Definitions.* IEEE Computer Society Press.

Fernström, C. y Lennart, F. 1991. *Integration Needs in Process Enacted Enviroments.* Publicado por IEEE Computer Society Press.

Flores Ríos, Brenda (2001). *A computer Aided Process Engineering Tool for the Study of Organizational Processes.* Encuentro Internacional de Computación en México 2001 (ENC'01). Sociedad Mexicana de Ciencia de la Computación.

Forrester, J.W. 1961. *Industrial Dynamics.* Cambridge, MA: MIT Press.

Fowler, Martin. 1999. *UML Destilled.* Editores Booch, Jacobson, Rumbaugh. Second Edition.

Gladwin, B. Diciembre, 1994. *Modelling Business Processing with Simulations Tools.* In proceedings of the 1994 Winter Simulation Conference. Pg. 114-121.

Greenwood, R.M., Robertson, I., Snowdon, R.A., Warboys, B.C. 1998. *Active Models in Business.* University of Manchester. Department of Computer Science.

Grudin, J. 1994. *Eight Challenges for Developers.* Communications of the ACM. Vol. 37, No. 1.

Hammer, M. 1990. *Reegineering Work: Don't Automate, Obliterate.* Harvard Business Review, Julio-Agosto.

Hammer, M. y Champy, J. 1993. *Reengineering the Corporation: A Manifesto for Business Revolution.* New York: HarperCollins.

Kawalek, Peter. 1997. *A Method for Designing the Software Support for Coordination.* Tesis de Doctorado en la Universidad de Manchester. Depto. de Ciencias Computacionales. Inglaterra.

Miers, Derek. 1996. *Use of Tools and Technology Within a BPR Initiative.* Coulson-Thomas, Colin. Business Process Re-egineering: Myth & Reality. Editorial Koga Page. Páginas: 142-165.

Ould, Martin A. 1993. *Process Modelling with RADs.* The Journal of the IOPT Club for the Introduction of Process Technology. Volumen 2. Número 2.

Ould, Martin A. 1995. *Business Processes.* Editorial Wiley.

Pressman, Roger S. 1993. *Ingeniería de Software.* Editorial McGraw-Hill.

Phalp, Keith y Shepperd, Martin. 1994. *A Programatic Approach to Process Modelling*. Lecture Notes in Computer Science. Third European Workshop, EWSPT'94. Editorial Spring-Verlag. Páginas: 65-68.

Roberts, C. 1993. *Coordination in Business Process*. The Journal of the IOPT Club for the Introduction of Process Technology. Volumen 2. Número 2.

Rojas Iñiguez, Fernando. 1998. *Simulation and Enactment in Process Modelling,* Tesis de Maestría en la Universidad de Manchester. Depto. de Ciencias Computacionales. Inglaterra.

Rossbach, Peter, Schreiber, Hendrik. 2000. *Java Server and Servlets*. Building Portable Web Applications. Editorial Addison-Wesley. USA.

Rumbaugh James, Jacobson Ivar y Booch Grady. 2000. *El Lenguaje de Modelado Unificado. Manual de Referencia*. Editorial Addison-Wesley. España.

Schill, A., Mittasch, C. 1996. *Workflow Management Systems on Top of OSF DCE and OMG CORBA*. Distributed Systems Engineering Journal, Vol. 3, No. 4, pp. 250-262.

Sommerville, Ian. 1996. *Software Engineering*. Editorial Addison-Wesley.

Warboys, B., Kawalek, P., Robertson, I, and Greenwood, M. 1998. *Business Information Systems*: *A process approach*. Editorial McGraw-Hill.

Warboys, B. 1989. *The IPSE 2.5 Project: Process Modelling as the basis for a Support Enviroment*. Universidad de Manchester.

Yeomans, B. 1996. *Enhancing the World Wide Web*. Reporte de proyecto. Universidad de Manchester.

Yu, L. 1990. *A Coordination Based Approach for Modelling Office Workflow and references. In Business Process Modelling*. Eds. B. Scholz-Reiten and E. Stickel. Springer Verlag. Berlin-Heidelberg.

VII.3 Direcciones de interés

AIIM. 2000. *Contenidos HTML en Páginas Web.*
URL: http://www.aiim.org/chapters/minnesota/ExpoPres/MCrocker-Image/sld021.htm.

Bergsten, Hans. 2000. *An Introduction to Java Servlets.*
URL: http://webdevelopersjournal.com/articles/intro_to_servlets.html

Das, S., Kochut, K., Miller, J., Scheth, A. y Worah, D. 2001. *ORBWork: A Reliable Distributed CORBA-based Workflow Enactment System for METEOR$_2$.*
URL: http://citeseer.nj.nec.com/das97orbwork.html

Groiss, Herbert y Eder, Johann. 2001. *Bringing Workflow Systems to the Web.*
URL: http://www.ifi.uni-klu.ac.at/PUBLICATIONS

Joeris, Gregor. 2001. *Decentralized and Flexible Workflow Enactment Based on Task Coordination Agents.* URL: http://www.aois.org/2000/Joeris-abstract.html

KBS. 2000. *Idef Methods.* Technical report, Knowledge Base Systems, Inc. URL: http://www.idef.com/

Nicolas, P. 1998. *Agent-based Workflow Automation.*
URL: http://www.ikonodine.com/whitewkflow/agents.html.

Universidad de Manchester. *Documentos Técnicos.*
URL: http://processweb.cs.man.ac.uk/#documents.

Ultimus. Octubre, 2001. *Benefits of using Ultimus.*

URL: http://www.ultimus.com/ultben.htm#lag_time

APENDICE A

Introducción

El 19 de enero de 1943 se publicó en el Diario Oficial la Ley del Seguro Social, donde se determina que la finalidad de la seguridad social es garantizar el derecho humano a la salud, la asistencia médica, la protección de los medios de subsistencia y los servicios sociales necesarios para el bienestar individual y colectivo (Ley General de Salud, 1997).

Como instrumento básico de la seguridad social se establece el Seguro Social y para administrarlo y organizarlo, se decreta la creación de un organismo público descentralizado, con personalidad y patrimonio propios, denominado Instituto Mexicano del Seguro Social (IMSS).

La misión implica una decidida toma de postura en favor de la clase trabajadora y sus familiares; misión tutelar que va mucho más allá de la simple asistencia pública y tiende a hacer realidad cotidiana el principio de la solidaridad entre los sectores de la sociedad y del Estado hacia sus miembros más vulnerables.

El primer nivel de atención médica que brinda el IMSS se constituye en la Unidad Médico Familiar (UMF), siendo ésta un módulo de atención para consulta externa y un nivel resolutivo donde se atiende al 85% de la población derechohabiente (DH) que acude a los servicios de medicina externa, auxiliares de diagnóstico, tratamiento y medicina preventiva. Los DHs que acuden periódicamente son los que tienen una atención directa con el médico familiar (MF).

La Unidad Médico Familiar No. 32 (UMF32) está compuesta por una plantilla de 16 médicos familiares: 8 en el turno matutino y 8 en el vespertino, cada uno con su respectiva asistente médica (AM); además para ambos turnos una coordinadora de asistentes médicos y un coordinador de médicos familiares.

La UMF32 actualmente recibe solicitudes para atención médica de consulta externa por parte de trabajadores asegurados, pensionados o beneficiados a través de previa cita (personal, correo electrónico, fax o teléfono) o presentándose de manera expontánea.

La necesidad de la UMF32 es la de mejorar sus procesos (en especial el de atención médica) y darle soporte a los mismos, además la unidad ha ido creciendo día con día y con ella la inquietud de usar Tecnología de Información para ofrecer de esta forma un mejor servicio a los derechohabientes.

Definición del problema

El proceso de atención médica de la UMF32 brinda una visión real de la problemática que enfrentan las organizaciones actuales en el uso de tecnología de información (soporte técnico), aspectos sociales tales como la capacitación del personal en el uso de nueva tecnología, cambio de cultura, entre otros; y por último, el proceso mismo.

Los problemas anteriores pueden ser analizados y/o resueltos a través de la filosofía de procesos dando soporte al realizar cambios mínimos en los subprocesos o un rediseño radical (innovación) del mismo.

La aportación principal que se dará a la UMF32, es la documentación completa y detallada del proceso de atención médica utilizando ingeniería de procesos. Esto permitirá el hecho de contar con información que contenga los aspectos técnicos y sociales, el modelado del

proceso, un pre-análisis y un reporte de la problemática encontrada hasta este momento, acompañada de una propuesta de posibles cambios o mejoras a dicho proceso.

Para lograr esto es importante establecer los siguientes objetivos:

- Definir el proceso de atención médica junto con sus objetivos.
- Identificar los agentes y roles del proceso de atención médica.
- Obtener las responsabilidades de los agentes involucrados.
- Evaluar el desempeño de las actividades realizadas.
- Identificar problemas o situaciones que se presenten durante el proceso para proponer soluciones que permitan mejorarlo.
- Brindar soporte en el proceso del cambio por medio de propuestas.

Descripción textual del proceso

A continuación se presenta una descripción de los subprocesos de citas, recepción, atención médica, interconsulta y farmacia creados a partir de la concentración de la información recabada y proporcionada durante las entrevistas antes descritas.

El subproceso de citas inicia cuando se abre la unidad de medicina a las 6:00 am. A partir de ese momento los DHs que desean recibir atención médica pueden depositar sus tarjetas de citas en unas cajas localizadas en cada consultorio para facilitar el trabajo de las AMs al momento de asignar las citas a los DHs. Los primeros que depositen sus tarjetas de citas serán los primeros que se les dé atención médica. Se atienden un máximo de 26 pacientes

por turno en cada consultorio pero dos lugares están reservados para DHs que laboran en la unidad. Si estos lugares no son solicitados por los trabajadores, entonces se asignan a los DHs que hayan llevado su tarjeta por la mañana. Después de las 14:00 pm pueden acudir personalmente o llamar por teléfono a la UMF32 para saber si alcanzaron cita con su MF. A las personas de la tercera edad por norma se les asigna el día y hora de la cita sin pasar por éste proceso (cita previa).

A continuación se describe el subproceso de recepción a derechohabientes con cita previa, de compañía o citados, realizado por la asistente médica dentro de la Unidad de Medicina Familiar No. 32.

La asistente médica recibe del derechohabiente (paciente, familiar o acompañante) con cita previa, de compañía o citado, la tarjeta de citas para realizar una solicitud de atención médica. En caso de que el DH sea un trabajador se debe solicitar el aviso de trabajo IMSS (2) 003 o SEC 007 y verificar su vigencia. Tanto el trabajador de alguna compañía como el DH que acude con forma de cita previa se debe registrar en la forma de control e informe de consulta externa 4-30-6 (forma 4-30-6), anotando en ella el número de afiliación y nombre del DH de acuerdo al orden de llegada. Si es un DH citado solamente consulta en la forma 4-30-6 para verificar que la cita corresponda al número de afiliación y nombre del mismo.

Posteriormente para todos los casos, confirma al DH el otorgamiento de la atención médica y lo invita a permanecer en la sala de espera. La AM busca en el archivero los expedientes clínicos MF-1 de los registrados en la forma 4-30-6 y verifica que estén completos y en orden. En caso de que alguno no se encuentre (lo cual ocurre esporádicamente), la AM

orienta y envía al DH al área de archivo clínico en el departamento de control de prestaciones para la verificación de vigencia de derechos.

Antes de iniciar con la atención médica la AM entrega al MF la forma 4-30-6 que contiene la lista de derechohabientes de ese turno y día.

Mientras se encuentra recibiendo a los DHs para entrar al consultorio, investiga entre la población en edad fértil el uso de algún método anticonceptivo. Por último, le proporciona al MF los expedientes clínicos MF-1 en el orden de llegada de los derechohabientes y notifica cuales no usan método anticonceptivo.

Enseguida se explica el subproceso de atención médica ejecutado por el médico familiar en la UMF32.

El MF recibe primeramente la forma 4-30-6 con los datos de los DHs que atenderá, después recibe al DH y verifica su nombre y matrícula para identificar sus necesidades. Le comunica a la AM la exploración que va a realizar para que lo asista y prepare al DH. La AM realiza la toma de temperatura y de somatometría al DH (toma de la presión, pulso y de frecuencia cardiaca y respiratoria). Una vez hecho esto, la AM informa los resultados al MF para que sean registrados en la forma 4-30-6 y en el expediente clínico MF-1.

El MF identifica los problemas de salud, registra los datos de la exploración física en el expediente clínico MF-1 y clasifica el problema. Registra el diagnóstico del DH en la nota médica MF-6. Identifica y valora el riesgo reproductivo de mujeres en edad fértil (MEF). Proporciona un diagnóstico para el problema de salud del DH y determina las medidas para resolver el problema. En caso de que sea necesario realizar análisis de laboratorio o

interconsulta proporciona la información y documentación (forma MF-8 y 4-30-2) necesaria para que el DH realice lo que el médico especifique.

En caso de que el DH asegurado amerite incapacidad temporal para el trabajo, el MF le extiende el certificado de incapacidad, con previa orientación, entrega al DH primera y segunda copia y el original al área de control de certificados de incapacidad y recetarios de la unidad, además lo registra también en la nota médica.

Finalmente registra en la forma 4-30-6 el diagnóstico y el número de recetas otorgadas. Una vez que finalizan las consultas se entregan los expedientes clínicos MF-1 y la forma 4-30-6 a la AM.

El subproceso de interconsulta se presenta cuando el DH ha sido atendido en varias ocasiones por su MF y el problema de salud rebasa la capacidad resolutiva de la unidad. En ese caso, se envía al DH con un especialista para que lo revise y valore a más profundidad. Lo anterior se hace por medio de la solicitud de interconsulta (forma 4-30-8), se registra al DH en la nota médica (MF-6) y se entrega a la AM quien orienta al DH sobre los pasos a seguir. La AM verifica la vigencia de derechos y la autorización del coordinador de médicos familiares (CMF). En caso de alguna duda por parte del CMF se hace una revaloración conjunta con el MF. Se tiene como fechas límites los miércoles antes de las 19:00 hrs para que los MF envíen las solicitudes de interconsulta. Una persona designada por el CMF se encarga de llevar las solicitudes a la Unidad de Medicina Familiar No. 8 (UMF8) que es donde se brinda el servicio de interconsulta. Esta persona recibe posteriormente respuesta a las solicitudes y entrega al DH la forma 4-30-8 con certificación de vigencia y la cita otorgada con el especialista. Finalmente se orienta al DH para el uso

adecuado de los servicios de interconsulta y el MF asigna una cita para darle seguimiento a su estado físico.

El subproceso que se presenta en la farmacia del IMSS inicia una vez que los DHs han sido atendidos por el MF se les entregan las recetas respectivas para aliviar sus problemas de salud. El DH puede surtir estas recetas en la farmacia de la unidad que está localizada en la entrada principal. Ahí, los DHs entregan sus recetas y se les proporcionan los medicamentos indicados. Uno de los inconvenientes que se pueden presentar en la farmacia es la falta de medicamentos básicos para la unidad, que se terminan por la gran demanda de los mismos.

En la Figura 54 se representan los subprocesos definidos mediante una gráfica rica plasmando de manera visual las líneas de comunicación entre los agentes y los formatos que se utilizan.

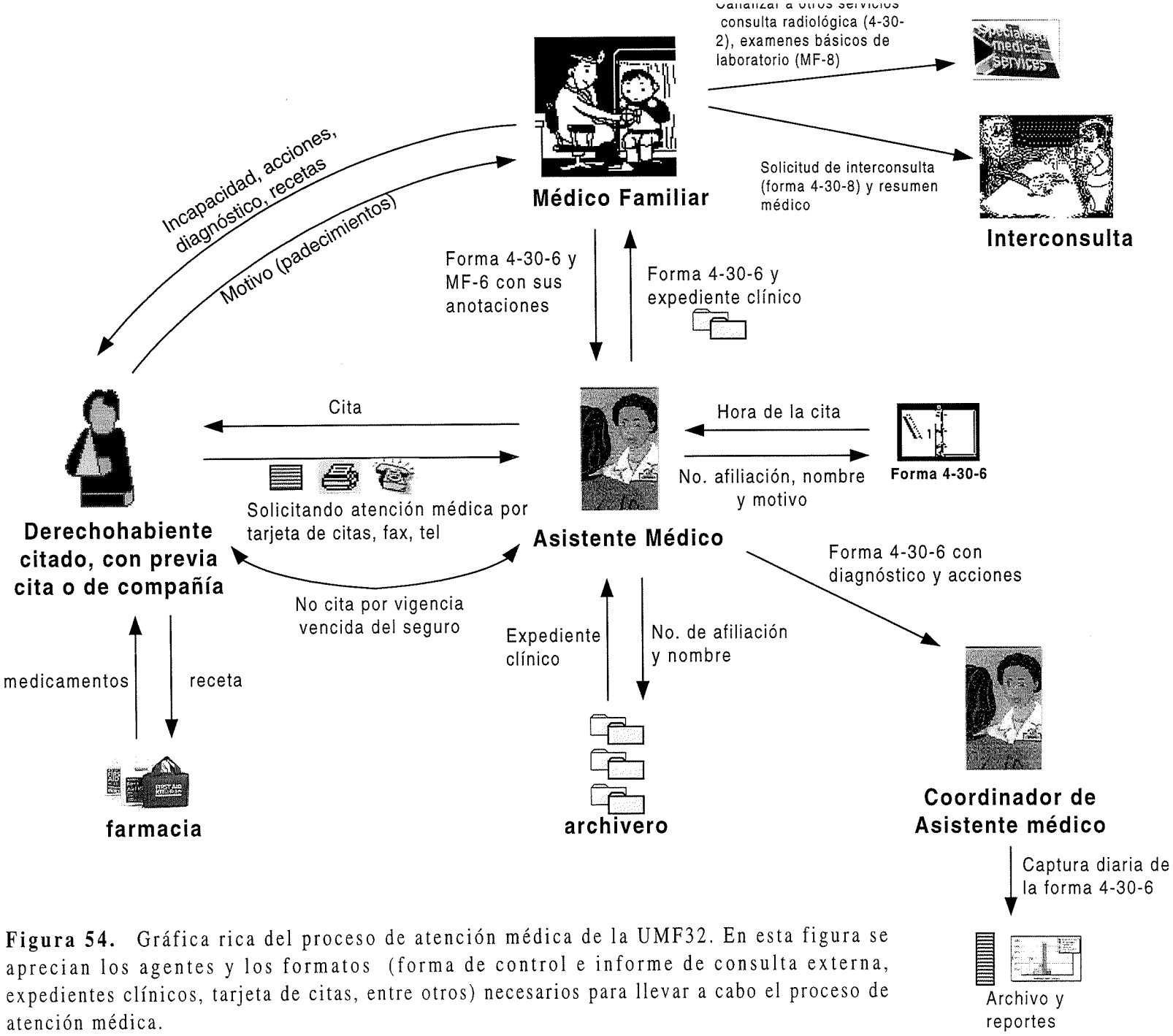

Figura 54. Gráfica rica del proceso de atención médica de la UMF32. En esta figura se aprecian los agentes y los formatos (forma de control e informe de consulta externa, expedientes clínicos, tarjeta de citas, entre otros) necesarios para llevar a cabo el proceso de atención médica.

Aspectos sociales y técnicos

La perspectiva socio-técnica describe a las organizaciones como una composición de un sistema social y un sistema técnico. La manera más simple de caracterizar cada uno de estos sistemas sería estableciendo que los sistemas sociales están compuestos de personas y sus preocupaciones, y que los sistemas técnicos por herramientas (tales como sistemas de cómputo, dispositivos de comunicación, etc.) y sus capacidades.

El diseño de sistemas socio-técnicos involucra entonces la unión del subsistema social (optimización para la satisfacción y motivación en el trabajo) y el subsistema técnico (optimizado para la eficiencia), de tal manera que se apoyen mutuamente.

Por lo anterior es importante señalar que dentro de un cambio organizacional se requerirá frecuentemente de la optimización de ambos subsistemas social y técnico. Partiendo de este punto se investigaron los posibles aspectos socio-técnicos existentes en el proceso de atención médica de la UMF32.

Aspectos Sociales

Los aspectos sociales involucran a las personas que trabajan para una organización y se enfoca en sus necesidades (relaciones personales, buen ambiente de trabajo, etc.) para cumplir y realizar su trabajo junto con sus habilidades, políticas y roles. Los aspectos sociales encontrados en la UMF32 fueron:

- Existe sobrepoblación de derechohabientes en el turno vespertino que desean atención médica, lo que genera una saturación de trabajo para el personal médico y espacios libres para el mismo proceso de atención en el turno matutino.

- No existe el módulo de orientación y quejas que proporcione información a los derechohabientes en el turno vespertino.
- Las responsabilidades del asistente médico son limitadas y orientadas más a la administración con lo que se evitan problemas con los derechohabientes y el IMSS.
- Se presentan temporadas donde no se cuenta con los suficientes medicamentos en la farmacia del IMSS y no es posible surtir las recetas de los derechohabientes para que alivien su padecimiento.
- Una actitud positiva por parte del MF y la AM con la finalidad de dar confianza al DH y brindar un mejor servicio.
- El deseo de que la medicina sea más preventiva que curativa con lo cual se disminuiría el número de enfermos y recursos utilizados para su atención.
- El entorno cultural de los derechohabientes, asistentes médicos y médicos familiares afecta positiva o negativamente las relaciones que existen entre ellos.
- Es necesario orientar y capacitar adecuadamente a las AMs eventuales para que la atención al DH siga siendo cordial y efectiva.
- Es necesario orientar y capacitar adecuadamente a las AMs de base para cuando se presenten situaciones donde ellas tengan que tomar decisiones sin consultar a la CAM o al CMF, estas sean correctas y oportunas resolviendo la problemática de ese momento.
- Los aspectos a considerar para el puesto de AM son: preparatoria terminada, aprobar los exámenes psicométricos y de conocimientos generales (aprovechamiento), dependiendo de los resultados se acepta o rechaza a la candidata.
- El área de atención médica cuenta actualmente con 8 consultorios, un médico familiar y un asistente médico por consultorio, un coordinador de asistentes médicos y un

coordinador de médicos familiares para satisfacer la demanda de la población afiliada al IMSS.

- Hasta el 30 de Abril del presente en el documento SUI (Sistema Único de Información) el número de DHs registrados a la UMF32 son 55,000 pero solamente se encuentran adscritos 46,000 a medicina familiar.

Aspectos Técnicos

El sistema técnico se enfoca en actividades, herramientas, configuraciones, datos, automatización y roles. Se puede describir de una manera más simple diciendo que los aspectos técnicos de una organización se refieren a sus procedimientos y la tecnología utilizada. Los aspectos técnicos encontrados en la UMF32 fueron:

- La tecnología de información con que se cuenta actualmente son máquinas de escribir mecánicas y papelería que en algunos casos no permiten que el proceso de atención médica, al menos en el área administrativa, sea más eficiente y rápido.
- No existen respaldos electrónicos o físicos de los expedientes clínicos MF-1 de los derechohabientes como una manera de asegurar siempre y en todo momento, su disponibilidad.
- Se cuenta con el servicio de citas por correo electrónico para las maquiladoras u otras organizaciones (CFE, UABC, entre otras), proporcionándoles un mejor servicio a través de la utilización de esta tecnología. Existe una computadora utilizada para la recepción de los correos electrónicos.

- La adquisición de equipo de cómputo es mediante el área de organización y calidad médica ubicada en la Ciudad de México por lo cual los recursos autorizados a las instituciones del IMSS son limitados.
- La tarjeta de citas es un pequeño papel de cartulina de 20 x 15 cm. doblado a la mitad, con información capturada a máquina y a mano. La durabilidad de la tarjeta no se garantiza al no estar protegida por algún plástico u otro material. La información que contiene es: fecha de entrega, número de folio, nombre del derechohabiente, unidad médico familiar, número de consultorio y turno.
- En el proceso de atención médica no existe el apoyo de alguna computadora para facilitar el trabajo ya sea de los MFs para el llenado de las formas, de las AMs para el control de los DHs ó de la CAMs para el concentrado de las actividades diarias y los informes semanales que tiene que presentar a la dirección de la UMF32.

Modelado del proceso

En ésta etapa se representa el proceso de atención médica utilizando técnicas diagramáticas como los diagramas del sistema y de cardinalidad, de flujo de datos y diagramas Rol-Actividad, entre otros.

Catálago de Usuarios

El catálogo de usuarios contiene los datos y las actividades de los agentes involucrados en el proceso dentro de la UMF32. Las siguientes tarjetas contienen información personal de los mismos:

Descripción del área	UMF32 Turno Vespertino
Dirección	Avenida Blancarte entre calle 2da. y calle 3ra.
	Col. Blancarte C.P. 22880
Tel conmutador	178 87 01

Educación e Inestigación Médica	Ext: 141
	Dra. Julia Mora Altamirano
Actividades	1. Normalizar los procesos de formación, capacitación y desarrollo del personal para la atención médica 2. Regularizar lo relativo a la información y documentación en salud 3. Formalizar y supervisar las promociones, desarrollo y difusión de proyectos de investigación en la UMF32

Consultorio: 3	Ext: 121
Médico Familiar	Dra. Laura Martínez Carrillo
Actividades	(Tabla IX)
Asistente Médico	Marisela Rodríguez Calvo
Actividades	(Tabla X)

Consultorio: 4	Ext: 120
Médico Familiar	Dr. Jesús Salinas Méndez
Correo electrónico	
Actividades	(Tabla IX)
Asistente Médico	Eventual
Actividades	(Tabla X)

Consultorio: 8	Ext:

Médico Familiar	Dra. Ana Bertha Cruz Toledo
Actividades	(Tabla IX)
Asistente Médico	Norma Páez
Actividades	(Tabla X)

Coordinación de Médicos Familiares	Ext:

Coordinador de MF	Dr. Roberto Juaréz
Actividades	3. Realizar informes diarios, semanales y mensuales del proceso de atención médica de la UMF32 utilizandolas formas 4-30-6 y 4-30-8. 4. Realizar 16 informes mensuales de los diferentes programas para ser enviados a la dirección. 5. Orientar a los DHs en los problemas que se presenten o escuchar sus inconformidades. 6. Revisar la forma 4-30-8 de los DHs que van a consulta a otra unidad UMF8 para que la información vaya completa y que exista una congruencia diagnóstica entre el tratamiento y el cuadro clínico. 7. Evaluar la atención integral y la congruencia clínico diagnóstica terapéutica por grupo etario. 8. Motivar y sugerir a los MFs para que se apeguen a los programas nacionales existentes. 9. Revisar los diagnósticos de los MFs utilizando la forma 4-30-6 principalmente los casos que comprometan la medicina preventiva. 10. Vigilar la calidad y la calidez de todos los empleados del área de atención médica (AM, CAM y MF).

Coordinación de Asistentes Médicos	Ext: 126
Coordinadora de AM	Ma. Lourdes Alpízar
Actividades	1. Coordinar y supervisar las actividades de las AMs. 2. Resolver cualquier problemática que se presente con los DHs como definir citas u orientarlos en el proceso de atención médica. 3. Suplir las actividades del asistente médico cuando sea necesario. 4. Realizar el conteo de las consultas que se presenten con la finalidad de realizar reportes. 5. Capacitar y/o actualizar a los asistentes médicos de nuevo ingreso o existentes cada 6 meses. 6. Hacer un reporte concentrando las actividades del día, para realizar posteriormente un informe semanal para entregarlo a la Dirección.

Todos los médicos familiares realizan las siguientes actividades:

Tabla IX. Tabla de actividades del médico familiar en la UMF32.

Actividades de los Médicos Familiares
1. Revisar el estado general de salud del DH y la historia clínica especificada en el expediente clínico MF-1.
2. Enviar a los DHs con el especialista llenando la forma de interconsulta 4-30-8 con su motivo de diagnóstico.
3. Informar sobre el estado de salud del derechohabiente y orientarlo sobre el tratamiento a seguir para la curación de su padecimiento.
4. Llenar toda la papelería necesaria en cada atención médica tales como: forma 4-30-6, receta individual, nota médica, forma 4-30-8, incapacidades, entre otros.
5. Entregar a la AM los expedientes clínicos MF-1 y la forma de control e informe de consulta externa con las anotaciones realizadas.

Las asistentes médicos realizan las actividades que se muestran a continuación:

Tabla X. Tabla de actividades del asistente médico en la UMF32.

Actividades de las Asistentes Médicos
1. Reservar dos lugares en caso de que se presenten uno o dos trabajadores afiliados al IMSS que laboren en la UMF32.
2. Llenar en la forma de control e informe de consulta externa 4-30-6 (ver apéndice 3) la relación de DHs que solicitaron citas para atención médica.
3. Buscar los expedientes clínicos MF-1 de los DHs y verificar que estén en orden antes de pasar a consulta.
4. Tomar los signos vitales como temperatura, tensión arterial, peso y talla para uso del médico en el diagnóstico de los pacientes.
5. Entregar diariamente la forma 4-30-6 a la coordinadora de asistentes médicos.
6. Informar a las mujeres en edad fértil (MEF) de las alternativas de planificación familiar que existen.

Diccionario de Datos

Es un elemento básico de referencia para localizar los nombres y atributos de los datos utilizados en todo el proceso. Un diccionario de datos es un documento que coordina, recopila y confirma lo que un término específico significa para el personal de la organización (Doake et al., 1993).

Fuentes de Información.

A partir de las entrevistas realizadas, se revisaron documentos, manuales y normas relacionados al proceso, mismos que fueron además mencionados por los agentes entrevistados. Estas fuentes de información se presentan enseguida:

- Tarjeta de citas.

Es un pedazo de papel cartulina que sirve para especificar el número de afiliación del derechohabiente y las próximas citas programadas. Esta tarjeta la entrega el DH al AM para que pueda registrar su cita en la forma 4-30-6.

Toda tarjeta de citas contiene la siguiente información:

Nombre de la variable	**Tipo de variable**	**Longitud**	**Descripción**
Número de afiliación			
Fecha de entrega			formato día/mes/año
Nombre del DH			
UMF32			
Número de consultorio			
Turno			

- Forma de control e informe de consulta externa 4-30-6.

Esta forma la utilizan las AM, MF, CAM y el CMF. La AM anotar el número de afiliación y nombre del DH en máquina de escribir para que posteriormente el MF registre los casos que ameritan incapacidad o pase a otros servicios (laboratorio, interconsulta) y los diagnósticos correspondientes a cada atención médica.

La CAM utiliza esta forma para elaborar reportes tomando en cuenta diagnósticos, datos del derechohabiente (edad, sexo, método anticonceptivo en su caso), incapacidades y tiempo amparado por éstas.

La forma de control e informe de consulta externa debe de contener la siguiente información:

Nombre de la variable	Tipo de variable	Longitud	Descripción
Unidad Fecha matrícula del médico	Fecha		formato día/mes/año especificar si es titular o suplente
clave servicio			
consultorio			del 1al 8
turno			especificar matutino o vespertino
horas consultadas	Número		
clave presupuestal			
nombre del médico			
hora cita			la hora que el DH entra al consultorio
número progresivo			número de cita del 1 al 26
1			1a. vez
2			Citado
3			recibió consulta
4			pase a serv. De la unidad
5			pase a otra unidad
6			alta
7			receta
8			días incapacidad
9			planificación familiar
10			accidentes y lesiones
11			riesgos de trabajo
12			invalidez
13			materno infantil
14			semanas de gestación
15			procedimientos consultorio
Número de afiliación y nombre			denominación social y nombre del propietario o concesionario
Agregado Codificación motivo de la consulta			

- Receta Individual.

Cuando el MF ha explorado al DH y le informa de su padecimiento, es necesario expedir una receta indivual donde se indique el medicamento y modo de uso del mismo. En cada receta individual sólo se pueden indicar un máximo de dos medicamentos, por lo que el MF expedirá por cada 2 medicamentos una receta.

La receta individual esta compuesta de los siguientes elementos:

Nombre de la variable	**Tipo de variable**	**Longitud**	**Descripción**
Nombre y Número de Seguridad Social del Asegurado			Nombre del DH y Razón y denominación social del propietario o concesionario
Nombre del médico			
Matrícula			Del médico
Cant			Cantidad de medicamento
Clave			Del medicamento
T.D.			Clave para famacia
Cédula Profesional			Del médico familiar
Registro Secretaría de Salud Fecha Autorización Rp: Modo de uso			señalando dosis del medicamento indicación médica sobre vía y periodicidad del medicamento

- Expediente Clínico.

Los expedientes clínicos MF-1 son propiedad de la institución y del prestador de servicios médicos, sin embargo, y en razón de tratarse de instrumentos expedidos en beneficio de los derechohabientes, deberán ser conservados por un período mínimo de 5 años, contados a partir de la fecha del último acto médico.

La información contenida en el expediente clínico MF-1 será manejada con discresión y confidencialidad, atendiendo a los principios científicos y éticos que orientan la práctica médica y sólo podrá ser dada a conocer a terceros mediante orden de la autoridad competente.

Todo expediente clínico deberá contener los siguientes datos generales:

Nombre de la variable	**Tipo de variable**	**Longitud**	**Descripción**
Tipo,nombre y domicilio del establecimiento			nombre de la institución que pertenece
Datos Personales			nombre, sexo, edad y domicilio del derechohabiente
Razón social			razón y denominación social del propietario o concesionario
Exploración física			habitus exterior, signos vitales,
Antecedentes familiares Esquemas de vacunación Antecedentes Gineco Obstrétricos			antecedentes heredofamiliares, personales patológicos y no patológicos
Planificación Familiar			
Detección de Enfermedades Cronológico-Degenerativas			
Hora y fecha			cuándo se registró la nota en el expediente
Firma			quién elaboró la nota en el expediente

Modelo del sistema

Por medio de una descripción de las interacciones de los agentes se obtiene una vista estructurada de alto nivel del sistema. Se utilizan modelos conceptuales que describen a los agentes, las interacciones entre ellos y la cardinalidad. Ésta última se representa por un semicírculo si existe sólo un agente, o doble semicírculo si existen uno o más agentes. Éstas características se reflejan en la Figura 55 del modelo del sistema del proceso de atención médica. Este modelo muestra las interacciones del sistema. Se puede observar que el coordinador de asistentes médicos interactúa con las Asistente Médicas, el Coordinador de Médicos Familiares y los DHs.

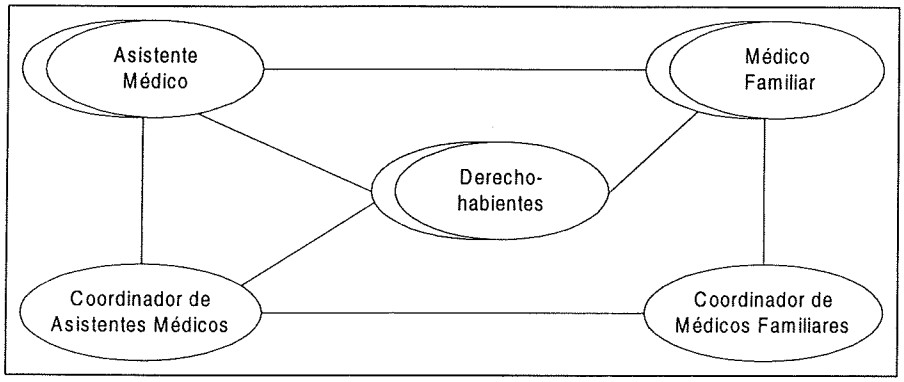

Figura 55. Diagrama del Sistema con Cardinalidad.

Diagrama de flujo de documentos físicos

En este tipo de diagramas se indica cual es la ruta que siguen los documentos y como fueron modificados en cada una de las actividades.

Los agentes se representan por una elipse, donde el nombre del agente va escrito dentro de ella. Las flechas de este diagrama indican la dirección que toma el flujo del documento. Sobre la flecha se encuentra el nombre del documento que fluye de un agente a otro. El diagrama de documentos físicos del proceso de atención médica se muestra en la Figura 56.

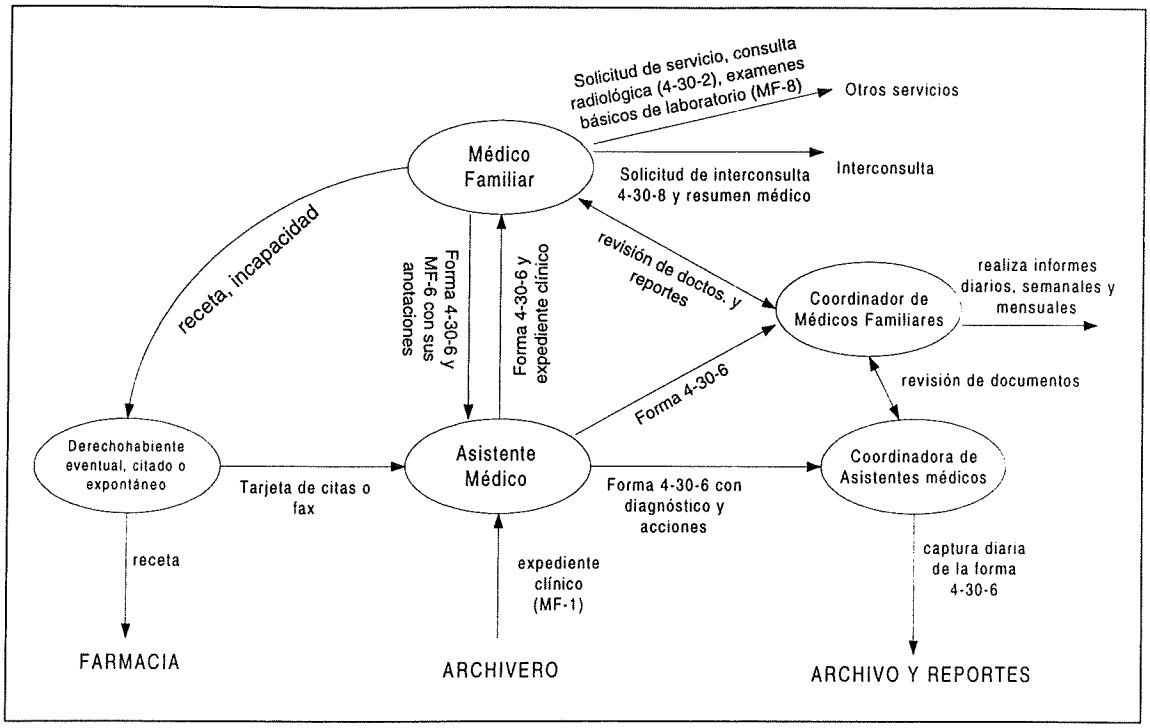

Figura 56: Diagrama de documentos físicos, donde se muestran la documentación y los formatos que manejan los agentes (representados por las elipses) de la UMF32.

Diagramas Rol Actividad (RAD)

La Figura 57 corresponde al RAD del subproceso de recepción para la atención médica desde la perspectiva de la AM, mientras que la Figura 58 se refiere al subproceso de atención desde la perspectiva del MF.

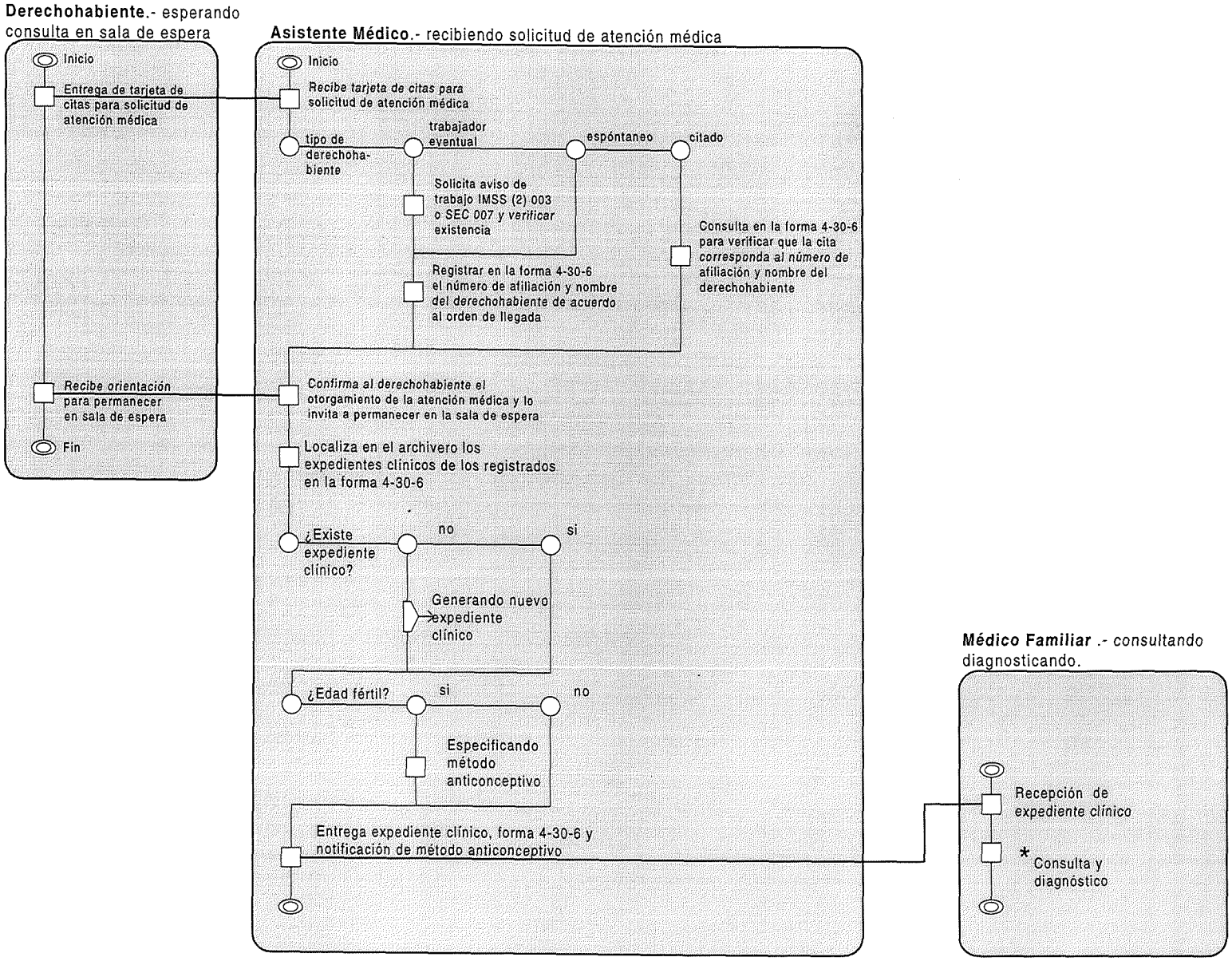

Figura 57. Diagrama RAD del proceso de recepción para atención médica desde la perspectiva de la asistente médica.

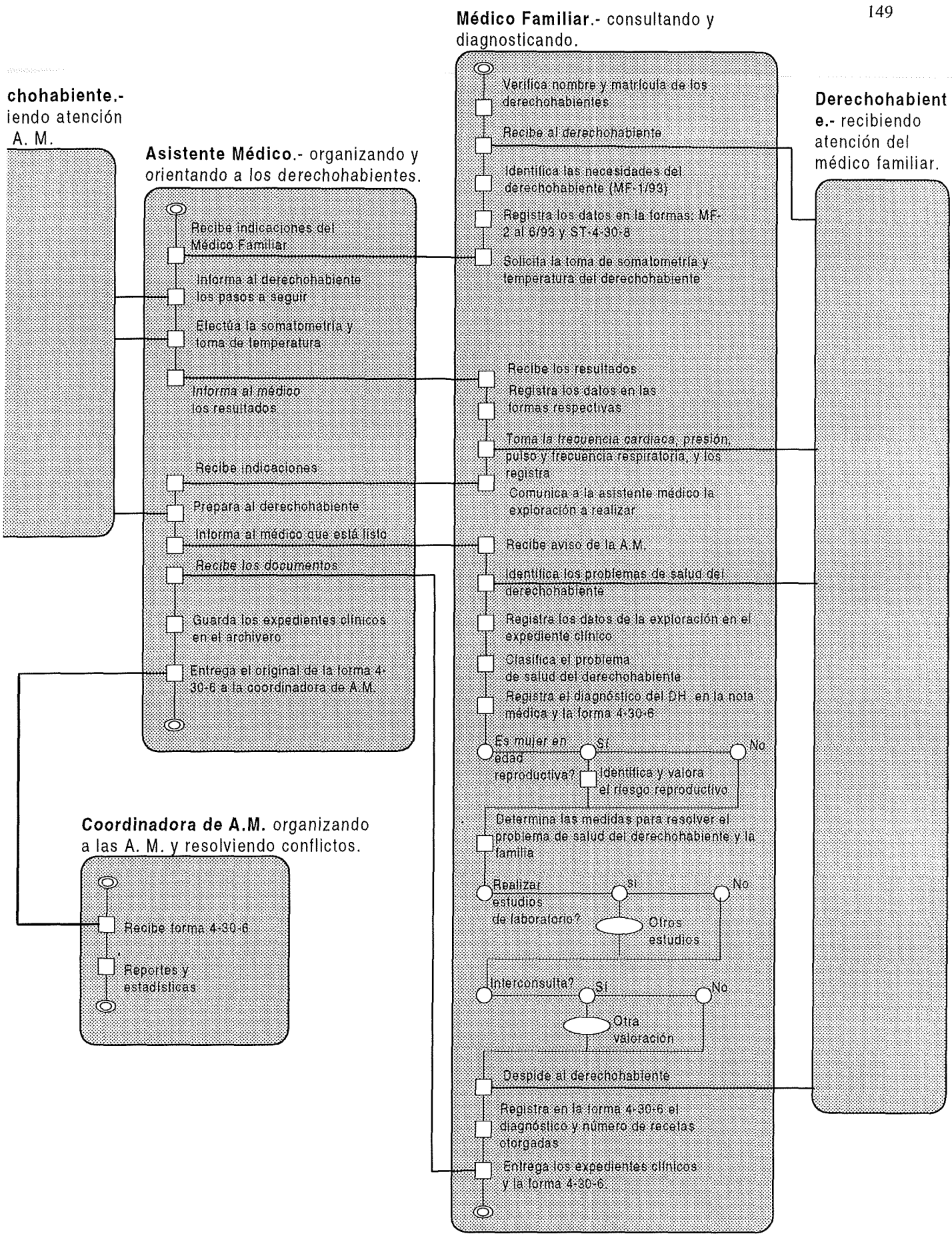

Figura 58. Diagrama RAD del proceso de atención médica desde la perspectiva del médico familiar

En las siguientes páginas se presenta dos de las formas utilizadas en la unidad de medicina familiar. La primera es la forma de control e informe de consulta externa (4-30-6) y la segunda es la forma de solicitud de interconsulta (4-30-8).

IMSS

INSTITUTO MEXICANO DEL SEGURO SOCIAL
DIRECCION DE PRESTACIONES MEDICAS

CONTROL E INFORME DE CONSULTA EXTERNA

UNIDAD	FECHA			MATRICULA DEL MEDICO		CLAVE SERVICIO	CONSULTORIO	TURNO	HORAS CONSULTA
	DIA	MES	AÑO	TITULAR	SUPLENTE				

CLAVE PRESUPUESTAL	NOMBRE DEL MEDICO

(4) PASE A SERVICIOS DE LA UNIDAD

1.- E.M.I.	8.- DIETOLOGIA
2.- PLANIF. FAM.	9.- TRAB. SOCIAL
3.- ENF. CRON. D.	10.- PSICOLOGIA
4.- MED. TRABAJO	11.- OPTOMETRIA
5.- ESPECIALIDAD	12.- LAB. CLIN.
6.- ESTOMATOLOGIA	13.- RAYOS X
7.- MED. PREV.	14.- OTROS

(9) PLANIFICACION FAMILIAR

NUEVO	METODO	SUB. RIESGO
1	PASTILLAS 1a. VEZ	7
2	PASTILLAS SUBS.	8
3	DIU 1a VEZ	9
4	DIU SUBS.	10
5	OTB SUB.	11
13	INYECTABLE 1a. VEZ	15
14	INYECTABLE SUBS.	16
6	VASECTOMIA 1a. VEZ	
12	VASECTOMIA SUBS.	

(10) ACCIDENTES Y LESIONES (LUGAR DONDE OCURRIO)

1.- HOGAR	
2.- TRABAJO	
3.- VIA PUBLICA	
4.- RECREACION	
5.- ESCUELA	
6.- OTROS	

(13) MATERNO INFANTIL

1	ATENCION PRENATAL
2	CONTROL PRENATAL
3	ATENCION PUERPERAL
4	CONTROL PUERPERAL
5	ATEN. NIÑO MENOR 1 AÑO
6	CONT. NIÑO MENOR 1 AÑO
7	ATENCION NIÑO 1 A 4
8	CONTROL NIÑO 1 A 4

No. PROGRESIVO	HORA CITA	1a. VEZ	CITADO	RECIBIO CONSULTA	PASE A SERV. DE LA UNIDAD	PASE A OTRA UNIDAD	ALTA	RECETA	DIAS INCAPACIDAD	PLANIF. FAMILIAR	ACCIDENTES Y LESIONES	RIESGOS DE TRABAJO	INVALIDEZ	MATERNO INFANTIL	SEMANAS DE GESTACION	PROCEDIMIENTOS CONSUL	NUMERO DE AFILIACION Y NOMBRE	AGREGADO	CODIFICACION MOTIVO DE LA CONSULTA
		1	2	3	(5)	6	7	8	(9)	(10)	11	12	(13)	14	15				
1																			
2																			
3																			
4																			
5																			
6																			
7																			
8																			
9																			

INSTITUTO MEXICANO DEL SEGURO SOCIAL

SUBDIRECCION GENERAL MEDICA

JEFATURA DE LOS SERVICIOS DE MEDICINA DEL TRABAJO

3 copias

INFORME MEDICO INICIAL

MT - 4 - 30 - 8

(PARA SER LLENADO POR EL MEDICO DE URGENCIAS O MEDICO FAMILIAR)

NUMERO DE AFILIACION		
APELLIDOS PATERNO Y MATERNO		
NOMBRE(S)	EDAD	SEXO ☐ M ☐ F
NOMBRE O RAZON SOCIAL DE LA EMPRESA		

1) FECHA DE ACCIDENTE				2) PRIMERA CONSULTA			
DIA	MES	AÑO	HORA	DIA	MES	AÑO	HORA

3) MECANISMO DEL ACCIDENTE

4) DESCRIPCION DE LA(S) LESION(ES)

5) DIAGNOSTICO(S)

6) TRATAMIENTO(S)

7) LESIONES O DEFECTOS PREVIOS AL ACCIDENTE EN RELACION A LAS LESIONES ACTUALES

8) SIGNOS Y SINTOMAS (MARQUE CON X)

☐ INTOXICACION ALCOHOLICA ☐ INTOXICACION POR ENERVANTES ☐ OTROS

DESCRIBIRLOS

9)

☐ HUBO RIÑA ☐ EXISTE SIMULACION ☐ SE PROVOCO LAS LESIONES INTENCIONALMENTE

10) INCAPACIDAD INICIAL

EN CASO DE EXPEDICION DE CERTIFICADO DE INCAPACIDAD TEMPORAL SE HARA EN LA RAMA DE E. G. Y SE ENGRAPARA EL TRIPLICADO A ESTA FORMA.

NUMERO DE FOLIO	A PARTIR DE:	DIA	MES	AÑO	NUMERO DE DIAS	

ENVIAR ESTE DOCUMENTO A MEDICINA DEL TRABAJO

EL PACIENTE PASA A SERVICIO DE _____

11) NOMBRE DEL MEDICO	CLAVE MEDICA	FIRMA DEL MEDICO

12) UNIDAD MEDICA Y DELEGACION

Apéndice B

En este apéndice se presenta los aspectos de diseño considerados en la arquitectura de coordinación como el documento DTD definido para crear el modelo base de un proceso, los diagramas de secuencia de la arquitectura y su respectivo diccionario de clases.

Definición del tipo de documento (DTD)

Este documento permite definir las reglas estructurales que van a dar forma al modelo base que se ha propuesto para un proceso. A continuación se muestra este documento el cual contiene los elementos necesarios para representar al modelo.

```
<!ELEMENT rad (rol+)>

<!ENTITY % coordenada "(x,y)">
<!ELEMENT x (#PCDATA)>
<!ELEMENT y (#PCDATA)>

<!ENTITY % elemento "(act+,ir_al?,terminador*)">
<!-- <!ELEMENT act (nom,edoactual,evento,edosigui?,%coordenada;,interaccion,textarea,button)> -->
<!ELEMENT act (nom,edoactual,edosigui?,%coordenada;,interaccion,cadena?)>
<!ATTLIST act id CDATA #REQUIRED>
<!ELEMENT nom (#PCDATA)>
<!ELEMENT edoactual (#PCDATA)>
<!ELEMENT evento (#PCDATA)>
<!ELEMENT edosigui (#PCDATA)>
<!ELEMENT interaccion (#PCDATA)>
<!ATTLIST interaccion cual CDATA #REQUIRED
                tipo (E|R) #REQUIRED>
<!ELEMENT cadena (#PCDATA)>
<!--<!ELEMENT textarea (#PCDATA)>
<!ELEMENT button (#PCDATA)> -->
<!ELEMENT ir_al (#PCDATA)>
<!ELEMENT terminador (%coordenada;)>

<!ENTITY % respuesta "(resp,%coordenada;)">
<!ATTLIST resp opcion CDATA #REQUIRED>
```

Continuación en la siguiente hoja...

```
<!ENTITY % alternativa "(condicional|paralela)">
<!ELEMENT condicional (%respuesta;,(%elemento; | %alternativa;)+)>
<!ATTLIST condicional que CDATA #REQUIRED>

<!ELEMENT paralela  (%respuesta;,(%elemento; | %alternativa;)+)>
<!ATTLIST paralela que CDATA #REQUIRED>

<!ELEMENT rol (agente, (%elemento; | %alternativa;)*)>
<!ATTLIST rol nombre CDATA #REQUIRED>
<!ATTLIST agente quien CDATA #REQUIRED>
```

Diagramas de secuencia de la arquitectura

Los siguientes dos diagramas corresponden al agente de actividades de un rol. La Figura 59 muestra el diagrama de secuencia que permite *realizar una actividad.* En éste, se pueden observar cinco objetos que a través del intercambio de mensajes se logra realizar una actividad. El objeto *aa1* se encarga de identificar y ejecutar la actividad actual. Si para cumplir con ésta, es necesario leer información adicional, el objeto *aa1* toma las entidades de información requeridas. Estas entidades están representadas por el objeto *ei1*. Finalmente, se ejecuta la actividad y si es conveniente realizar alguna operación sobre los datos, se llama al objeto *op1* del diagrama, y se procede a almacenar las entidades de información involucradas.

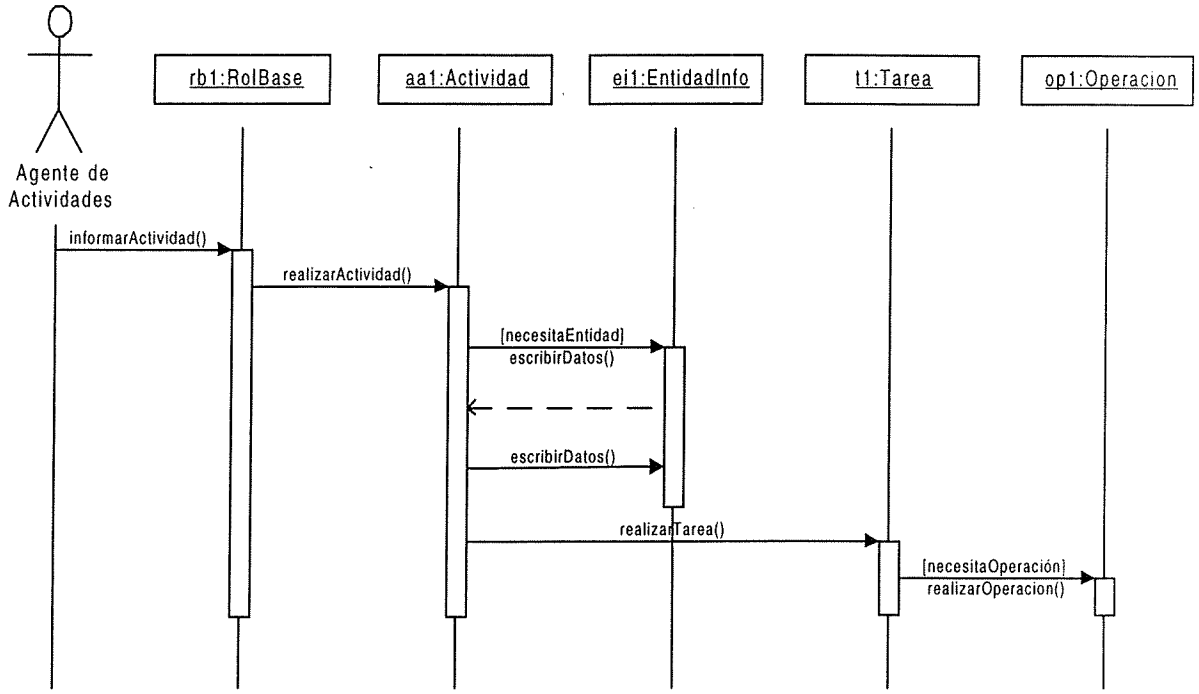

Figura 59. Diagrama de secuencia del caso de uso realizar actividad.

El diagrama que se presenta en la Figura 60, muestra la secuencia que se sigue para informar y asignar las actividades a los roles de un proceso. El objeto *apg1* solicita información al agente de actividades respecto a la tarea que va a realizar el rol representado por *rb1*. Este objeto solicita la siguiente actividad al agente *aa1*, que a su vez obtiene los datos necesarios para que el rol cuente con los elementos que le permitirán ejecutar la actividad. Una vez que se tienen las entidades de información, el objeto *aa1* solicita al agente de estados (objeto *ae1*) la siguiente actividad de acuerdo a lo que indica su tabla de estados.

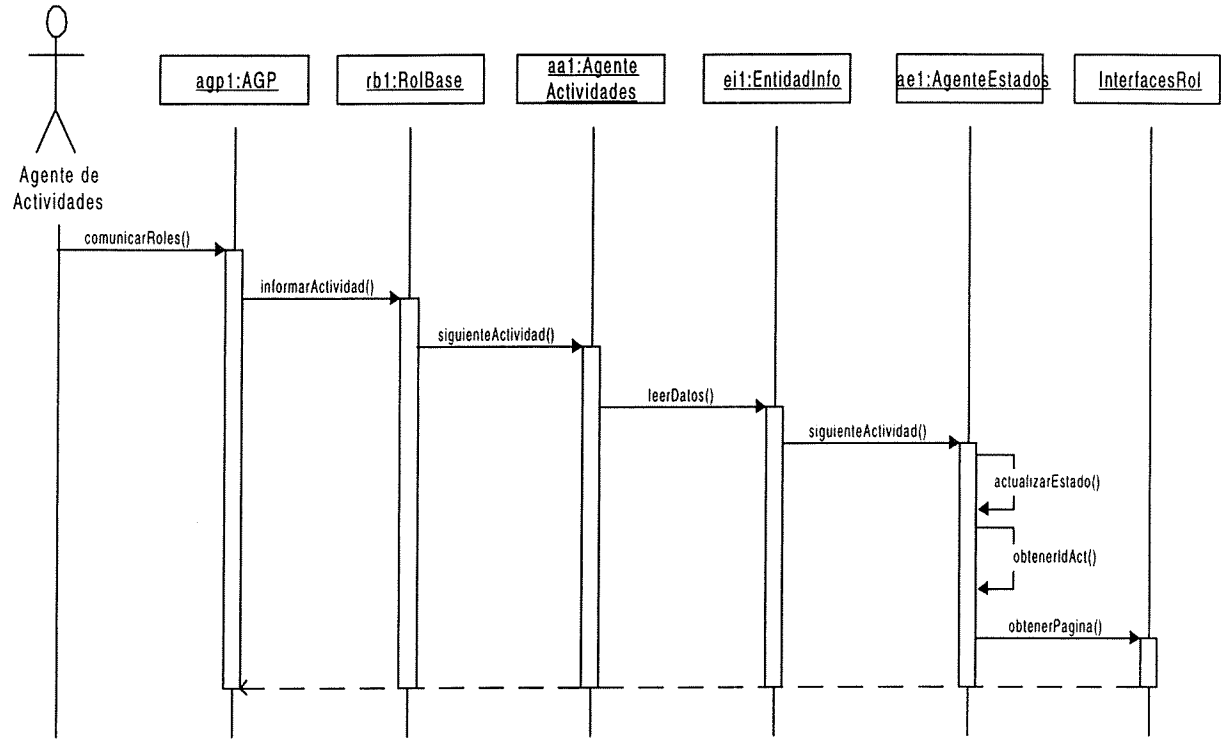

Figura 60. Diagrama de secuencia del caso de uso informar actividades al AGP.

La Figura 61 muestra el diagrama de secuencia del agente de estados para *recuperar interfaces* de un rol. Se puede observar en el diagrama que el objeto *aa1* obtiene las entidades de información que se requieren para una interfaz (objeto *ei1*) y solicita al agente de estados dos cosas: el estado del rol pàra esta actividad, y segundo, de acuerdo a este estado obtener la interfaz correspondiente con la cual va a interactuar el rol para ejecutar la actividad.

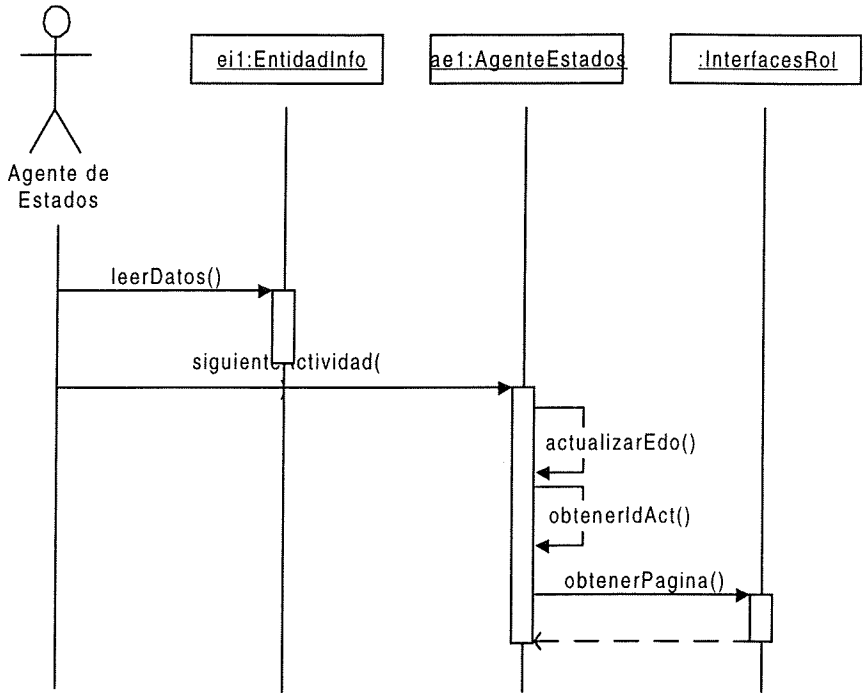

Figura 61. Diagrama de secuencia de los casos de uso del agente de estados.

Diccionario de clases de la arquitectura

En esta sección se definen las clases utilizadas para la implementación de la arquitectura de coordinación así como sus atributos y métodos.

Clase: *AGPS*, esta clase es un Servlet que se encarga de enviar la informacion necesaria a cada uno de los participantes de algun proceso.

Atributos: String URI, AGP manejador.

Métodos: init(ServletConfig config), doGet(HttpServletRequest req, HttpServletResponse res).

Clase: *AGP*, se utiliza para coordinar los roles involucrados en un proceso.

Atributos: String pathToInterf, String pathToModel, Vector roles = new Vector().

Métodos: AGP(String uri, String modelo, String interfaces), comunicarRoles(PrintWriter writer, String rol, String rec, String edo, Hashtable params), leerDoctoXML(PrintWriter writer).

Clase: *AnalizarProceso*, declara el tipo de analizador XML que se utilizará para verificar el modelo base.

Atributos: String pathToModel, String pathToInterf, Vector roles.

Métodos: AnalizarProceso(String toModel, String toInterf), obtenerProceso(), analizarModeloBase(PrintWriter writer).

Clase: *ManejarContenido*, realiza el análisis sintáctico del modelo base (XML) de un proceso.

Atirbutos: Hashtable rolInfo, String[][] componentes, int cont, private String pathToModel, private String pathToInterf, String nomVar. RolBase rol, Vector roles, String element, String id, String idAgente, int numEdos, String tipo.

Métodos: ManejarContenido(String toInterf), setDocumentLocator(Locator locator), startDocument(), endDocument(), processingInstruction(String target, String data), startPrefixMapping(String prefix, String uri), endPrefixMapping(String prefix), startElement(String namespaceURI, String localName, String rawName, Attributes atts), endElement(String namespaceURI, String localName, String rawName), characters(char[] ch, int start, int end), ignorableWhitespace(char[] ch, int start, int end), skippedEntity(String name), copiarActividad(String[][] comp, int cont, int numEdos), obtenerRoles().

Clase: *ManejarErrores*, define los métodos necesarios para disparar los mensajes de error que se presenten en el modelo base.

Métodos: warning(SAXParseException exception), error(SAXParseException exception), fatalError(SAXParseException exception).

Clase: *RolBase*, se utiliza para almacenar información básica de un rol.

Atributos: String rol, String agente, String id, boolean refrescar, AgenteActividades aa.

Métodos: crearRol(String toInterf, String[][] comp, String vars), definirRol(Hashtable rolInfo), obtenerIdAgente(), obtenerRol(PrintWriter writer), informarActividad(Hashtable params), informarActividad(PrintWriter writer, String edo, Hashtable params), informarActividad(PrintWriter writer, String edo), interaccion(), noInteraccion().

Clase: *AgenteActividades*, se utiliza para definir las actividades de un rol.

Atributos: EntidadInfo ei; AgenteEstados ae = new AgenteEstados().

Métodos: definirActividades(String nombre, String toInterf, String[][] comp), Object pedirDatos(), pedirActividad(), realizarActividad(), actualizarInfo(String dato), actualizarInfo(int dato), void actualizarInfo(double dato).

Clase: *AgenteEstados*, se utiliza para definir los estados por los que atraviesa un rol después de realizar sus actividades.

Atributos: PaginasHTML hp1, String tablaEstados[][], private int renglones, int columnas, String edoActual, int noIndex, String pathToInterf.

Métodos: definirEstados(String toInterf, String[][] comp), void crearTabla(String[][] comp), obtenerIdAct(String edo), estadoInicial(), verificarInter(), obtenerEstado(), actualizarEstado(String edo), siguienteActividad(PrintWriter writer, String edo, Object param).

Clase: *EntidadInfo*, se utiliza para crear objetos poder manejar distintos tipos de datos.

Atributos: String nomObjeto, Object oDato, String sDato, int iDato, double dDato.

Métodos: EntidadInfo(String nombre), Object leerDatos(), String nombreObjeto(), escribirDatos(Object info), escribirDatos(String info), escribirDatos(int info), escribirDatos(double info).

Diccionario de clases del prototipo

A continuación se describen las clases del modelo de la cerveza y se muestran sus atributos y métodos.

Clase: *Cliente*, en la cual se definen los atributos y métodos que le dan la funcionalidad requerida al rol con el mismo nombre para realizar y enviar órdenes.

Atributos: Orden orden, Rol rol1, Acceso acceso.

Métodos: enviarOrden(), recibirRespuesta(), crearOrden(String nombre, int cant).

Clase: *Minorista*, esta clase se encarga de definir los atributos y métodos que van a permitir a este rol interactuar con el Cliente y satisfacer sus demandas de productos, así como también, verificar constantemente la cantidad de artículos en almacén con el propósito de enviar una orden al Mayorista cuando el nivel del inventario esté bajo.

Atributos: int totalInventario, Orden o1, Orden o2, ProdSolicitados ps, ps2; Rol rol1, Acceso acceso.

Métodos: enviarOrden(Orden ordenClte), responderClte(), crearOrden(String nombre, int cant), inicializarInventario(), solicitarArticulos(), obtenerInventario(), recibirProductos(), verificarInventario(), recibirRespuesta(ProdSolicitados ps1).

Clase: *Mayorista*, esta clase se encarga de responder a las órdenes de productos del Minorista exclusivamente, y mientras no existan éstas, espera hasta el arribo de una orden.

Atributos: Rol rol1, Acceso acceso, ProdSolicitados ps.

Métodos: solicitarArticulos(Orden ordenMay), recibirProductos(), recibirRespuesta(String fecha, String comentario).

Clase: *Orden*, se utiliza para que los Clientes o Minoristas puedan enviar solicitudes de artículos a sus respectivos proveedores.

Atributos: int cantArticulos, String nombreClte.

Métodos: asignarNombreClte(String Nombre), asignarCantArticulos(int Cant), obtenerNombreClte(), obtenerCantArticulos().

Clase: *ProdSolicitados*, esta clase se encarga de guardar las órdenes de productos recibidas por el Minorista y Mayorista.

Atributos: String nombreClte, Respuesta resp.

Métodos: asignarNombreClte(String nomb), obtenerNombreClte(), crearRespuesta(int cant).

Clase: *Respuesta*, la función de esta clase es crear una respuesta a los roles que solicitan productos y enviarla a los mismos.

Atributos: String comentarios, int cantArticulos, String fechaEntrega.

Métodos: obtenerComent(), asignarComent(String comen), asignarCantArticulos(int cantidad), obtenerFechaEntrega(), int obtenerCantArticulos(), asignarFechaEntrega(String fe).

Clase: *servletJuCerv* (coordinador del proceso), esta clase tiene como propósito coordinar las interacciones y actividades realizadas por los agentes del proceso.

Atributos: Cliente c, String[] edoClte, Minorista min, String[] edoMin, Mayorista may, String[] edoMay.

Métodos: doGet(HttpServletRequest req, HttpServletResponse res), init(ServletConfig config).

Clase: *ViewUtil,* esta clase permite a los roles crear sus interfaces dependiendo de la actividad que deban realizar.

Métodos: ordenarClte(PrintWriter writer, String action), esperarClte(PrintWriter writer, String action), terminarClte(PrintWriter writer, String action, int cant).

Clase: *HTMLUtil,* esta clase es utilizada por ViewUtil para formar las interfaces de los roles.

Métodos: formatAttr(String name, String value), writeDocumentHeader(PrintWriter writer, String title, String action), writeDocumentHeaderMeta(PrintWriter writer, String title),

writeParamMeta(PrintWriter writer, String text), writeFooterMeta(PrintWriter writer), writeDocumentFooterMeta(PrintWriter writer), writeDocumentFooter(PrintWriter writer), writeTableHeader(PrintWriter writer), writeTableCell(PrintWriter writer, String title, String type, String name, String value).

Guía para utilizar la arquitectura de coordinación

Para construir un sistema de coordinación que brinde soporte a un proceso por medio de la arquitectura, primero se debe identificar este proceso y realizar todas las etapas de la ingeniería de procesos descritas en el capítulo II. Enseguida, se llevan a cabo los siguientes pasos:

Paso 1: Se debe crear el diagrama RAD del proceso en estudio (en caso de no existir). Este modelo nos permitirá entender el comportamiento del proceso y las interacciones existentes entre los roles involucrados.

Paso 2: Para cada uno de los roles del proceso se debe crear un diagrama de estados por medio del cual se identifique el comportamiento de los roles a medida que van ejecutando sus actividades. Estos diagramas son una pieza importante en la construcción del *modelo base del proceso* ya que por medio de los diferentes estados de un rol se podrá definir su comportamiento.

Paso 3: Para el proceso en estudio, es importante identificar las entidades de información que utilizan los distintos roles para realizar sus actividades. Por ejemplo, si en el proceso se

maneja una entidad de tipo orden, es necesario verificar qué información debe contener una orden, por ejemplo: nombre, cantidad de productos, fecha de pedido, entre otros. También se debe saber a cuáles roles pertenecen las entidades de información. La identificación de la relación entre roles y entidades es con la finalidad de definirlas en el *modelo base del proceso* y asociar las entidades al rol que le corresponde.

Paso 4: Tomando como referencia el diagrama RAD del proceso, se va a construir su *modelo base*. Para esto, se ha creado un documento DTD (proceso.dtd) el cual define la estructura que se debe seguir para crear el modelo utilizando el lenguaje XML (Extensible Markup Language). De acuerdo a esto se debe realizar lo siguiente:

Paso 4.1: Para cada uno de los roles se tiene que crear un submodelo en XML que represente al diagrama RAD. Es decir, la definición de un Rol representado por un rectángulo en el diagrama, debe estar contenido en un submodelo. La información que contiene un rol es: actividades, condicionales, interacciones, actividades paralelas, entre otras. Esta misma información es la que debe estar contenida en el submodelo de cada rol. Se recomienda utilizar una letra para identificar al agente de un rol dentro del submodelo. Para ello, en una etiqueta llamada *agente* definida en el DTD, existe el atributo *quien* que sirve como identificador para el agente. Por ejemplo: si un agente es un escritor el atributo *quien* tendría una "a" como identificador:

<rol nombre="escribiendo carta (ec)">
<agente quien="a"> escritor **</agente>**
.....
</rol>

De la misma forma, es recomendable asignar a los identificadores de las actividades de un rol el mismo que éste tiene asignado más un número. Para el ejemplo visto anteriormente sería algo como sigue:

<rol nombre="escribiendo carta (ec)">
<agente quien="a"> escritor </agente>
<act id="a1">
<nom> escribir carta </nom>
<edoactual> escribiendo </edoactual>
<evento> carta enviada </evento>
<edosiguiente> esperando </edosiguiente>
</act >
</act id="a2">
.....
</act>
.....
</rol>

Paso 4.2: Dentro del submodelo de un rol, es necesario agregar la información que proporcionan los diagramas de estados. Esto es, el comportamiento que tienen los roles durante la ejecución de sus actividades se debe ver reflejado aquí. Por ejemplo, un rol va a tener un *estado actual* para una actividad cuando la está realizando en este momentoy un *estado siguiente*, generado a partir de un evento determinado una vez que terminó de ejecutar la actividad. Esto se representa por las etiquetas siguientes:

<act id="a1">
<nom> escribir carta </nom>
<edoactual> escribiendo **</edoactual>**
<evento> carta enviada **</evento>**
<edosiguiente> esperando **</edosiguiente>**
</act >

Paso 4.3: También se debe agregar la información referente a la interacción de un rol con otro, en las respectivas actividades dentro de sus submodelos. Es decir, se tiene que agregar

una etiqueta de tipo *interacción* tanto al rol emisor (el que inicia la interacción) y al rol receptor (el que recibe la interacción). Esto se ejemplifica a continuación:

"Submodelo de un rol"

<act id="a1">
<nom> escribir carta </nom>
<edoactual> escribiendo </edoactual>
<evento> carta enviada </evento>
<edosiguiente> esperando </edosiguiente>
<interacción cual="b1" tipo="E"> b </interacción>
</act >

"Submodelo del segundo rol"

<act id="a1">
<nom> escribir carta </nom>
<edoactual> escribiendo </edoactual>
<evento> carta enviada </evento>
<edosiguiente> esperando </edosiguiente>
<interacción cual="a1" tipo="R"> a </interacción>
</act >

El atributo "cual" de esta etiqueta contiene el identificador de la actividad con la que se asocia, en este caso *b1* y *a1*. El atributo "tipo" puede contener dos valores: una *E* para la actividad que es emisora y una *R* para la actividad que es receptora.

Paso 4.4: En ocasiones, no es necesario describir todas las actividades dentro del submodelo porque algunas al momento de implementarlas en un sistema ya van implícitas. Por ejemplo, si se tiene en un RAD la actividad *escribir* para un rol y a continuación está una actividad *enviar*, esta última ya va implícita en la interfaz del rol que está realizando la tarea de *escribir,* y puede estar representada en forma de un botón que sirva para enviar una respuesta o petición.

Paso 4.5: Una vez que se tienen los submodelos para cada rol, es necesario juntarlos en lo que será el *modelo base del proceso*. A continuación se definen la secuencia de pasos a través de un ejemplo:

Primero se crea un archivo con extensión "xml": *osca.xml*, que será el *modelo base*.

Segundo, se coloca la etiqueta de inicio del modelo para definir la versión del XML que se va a utilizar:

<?xml version="1.0" encoding="ISO-8859-1"?>

Tercero, se coloca la etiqueta para definir el tipo de documento, llamando al DTD que se va a utilizar para validar la estructura y los submodelos de los roles creados en el paso 4.1. Dentro de los corchetes se mandará llamar a las entidades correspondientes a cada uno de los roles de un proceso.

<!DOCTYPE rad SYSTEM "proceso.dtd"[

Cuarto, se crean entidades para cada uno de los submodelos realizados. En este ejemplo, existen dos submodelos que se van a llamar:

<!ENTITY *escritor* SYSTEM "writer.xml">

<!ENTITY *colaborador* SYSTEM "collaborator.xml">

]>

Quinto, finalmente se crea el cuerpo del modelo base mandado llamar a las entidades definidas anteriormente:

<rad>

&escritor;

&colaborador;

</rad>

Paso 4.6: Una vez que se ha definido el *modelo base de un proceso*, es necesario verificar que esté bien formado. Para hacer esto se utiliza un programa como analizador (parser) el cual envía mensajes de error a la consola en caso de que el modelo no esté bien construido. Este mensaje muestra la línea en la que se produce un error para poder corregirlo.

Paso 5: Una vez que se tiene el *modelo base* en XML del proceso y se ha validado mediante el analizador, es necesario construir las interfaces que contengan los elementos necesarios para que los roles puedan ejecutar sus actividades. Debe existir para cada estado de un rol una interfaz diferente de tal manera que puedan realizar todas sus actividades.

Paso 5.1: En la forma de una interfaz (documento HTML) debe estar incluida una etiqueta de tipo acción en la cual se va indicar la ruta a la cual el rol debe enviar la petición. Para el caso de esta arquitectura el URI (Identificador Universal de Recursos) será el siguiente:

<FORM METHOD="GET" ACTION="/servlet/AGPServlet">

EL URI en esta forma es "/servlet/AGPServlet".

Paso 5.2: Si el rol que envía la petición interactúa con otro rol entonces dentro de la definición de la forma (interfaz de la actividad de un rol) se deben agregar dos campos como los siguientes:

<INPUT type="hidden" name="rol" value="a">
<INPUT type="hidden" name="receptor" value="b">

La primera línea indica mediante un atributo *oculto* (hidden) quién es el rol que inicia la interacción con otro. Se pueden observar dos atributos: *name* que define a un rol y *value* en el cual se indica el identificador de este rol.

La segunda línea define al rol con el cual se va a interactuar. De la misma forma se utiliza un campo oculto y se definen dos atributos: *name* que tiene como valor "receptor" indicando que es un rol receptor y *value* que tiene como valor el identificador del receptor, en este caso "b".

Paso 5.3: Se debe agregar a cada interfaz una etiqueta mediante la cual se pueda desconectar del sistema un agente. Esta etiqueta debe contener el URI del servidor al cual se manda la petición, un atributo *salir* con el valor "yes" y un atributo *rolesmod* que contenga como valor el nombre del documento HTML en la que están definidos los roles de un proceso para acceder a su sistema de coordinación. Esto se realiza con la finalidad de desconectar al agente del sistema y presentarle la interfaz principal. A continuación se muestra la etiqueta:

Salir

El valor de URI es "/servlet/AGPServlet?salir=yes&rolesmod=modeloOSCA.html".

El valor de salir es "yes".

El valor de rolesmod es "modeloOSCA.html".

Requerimientos de instalación

La arquitectura de coordinación propuesta en este trabajo debe cumplir con algunos requisitos de instalación que permitan la generación de sistemas de coordinación. Entre éstos se encuentran los que se mencionan a continuación.

Equipo de cómputo.

Las características del equipo que se va a utilizar deben ser las siguientes: procesador Pentium II en adelante, un mínimo de 32 megabytes de memoria RAM, tarjeta de red y se necesita acceso a *Internet.*

Software requerido.

En una máquina se debe tener instalado un servidor *Web*, de preferencia *Apache*, que permita enviar y recibir la información que intercambian los roles en un proceso, además de enviar las interfaces HTML correspondientes, que ayudarán al usuario (agente representando un rol) a realizar sus actividades.

El servidor *Web* debe contar con soporte a *Servlets* para atender las peticiones de los agentes, y por otro lado, responder a éstas. El *Servlet* utilizado para este propósito está representado por la clase AGPS.

El equipo de cómputo necesita tener instalada la máquina virtual de java (JVM por sus siglas en inglés) para poder utilizar la arquitectura de coordinación. La versión de Java debe ser la JDK1.2 en adelante.

El equipo utilizado también debe tener instalado el soporte para el lenguaje XML por medio del API (*Application Program Interface*) Xerces de *Apache*.

Las clases que conforman la arquitectura de coordinación deben estar almacenadas en el directorio "servlet/" del servidor *Web* utilizado. De la misma manera, las interfaces y modelos base de cada uno de los procesos a los cuales se les ha dado soporte, estarán instalados en los directorios respectivos. Por ejemplo, si tenemos el modelo base del proceso *solicitar visa*, se puede crear un directorio que se llame "visa/" en la ruta "servidor web/servlet/interfaces/" almacenando dentro de "visa/" las interfaces del proceso en cuestión y su respectivo modelo base. La interfaz principal de la arquitectura se debe localizar en el directorio"servidor web/htdocs/arquitec/". Ésta contiene una lista de los modelos bases que están disponibles (ligas haciendo referencia a estos modelos) para generar un sistema de coordinación.

Por medio de los elementos mencionados anteriormente, será posible utilizar la arquitectura de coordinación para generar un sistema que dé soporte a un proceso.