CAPÍTULO V -

DESARROLLO DEL SISTEMA

5.1 Análisis del sistema

5.1.1 Requerimientos

A continuación se enlistan los requerimientos del EAPI y Venus:

Requerimientos EAPI:

- Ser un entorno de desarrollo que pueda ser migrado fácilmente entre distintas plataformas (Unix, Windows, Mac OS X, Linux).
- Permitir el rápido desarrollo de herramientas I-CASE basadas en el lenguaje de programación Java.
- Estar disponible en el Web.
- Implementar las capas del modelo de referencia de Integración propuesto por Roger Pressman [Pressman, 01].
- Presentar una interfaz amigable para el usuario.
- Proporcionar componentes de software para las herramientas I-CASE que se desarrollen.
- Que el sistema presente un buen diseño, para que futuros desarrolladores puedan incrementar nuevas funcionalidades fácilmente.

Requerimientos Venus:

- Construirlo utilizando el entorno que proporciona EAPI.
- Validar y evaluar la funcionalidad del EAPI.
- Permitir crear y editar diagramas de clase y de colaboración.
- Que los diagramas puedan ser almacenados y recuperados correctamente en y del depósito de proyectos.
- Poder imprimir los diagramas.
- Llevar un control de versiones de los proyectos desarrollados.
- Presentar una interfaz amigable para el usuario.
- Proporcionar al usuario un árbol gráfico que permita tener un orden en los diagramas creados (árbol de proyecto).
- Que el sistema presente un buen diseño, para que futuros desarrolladores puedan incrementar nuevas funcionalidades fácilmente.

5.1.2 Casos de Uso del EAPI

Después de analizar detalladamente los requerimientos del EAPI se identificaron nueve casos de uso básicos. Ver Fig. 5.1.

Caso de uso:		Acceso al EAPI.	
Actores:		Usuario, EAPI (iniciador).	
Propósito:		Permitir que un usuario tenga acceso al EAPI.	
Resumen:		Se valida que el usuario esté registrado en el EAPI, si aprueba la validación se le permite el acceso.	
Curso normal de los eventos			
Actor		**Sistema**	
Paso	Acción	Paso	Acción
2.	El usuario introduce su login y password, y presiona el botón aceptar.	1. 3. 4.	EAPI muestra una interfaz gráfica que permite que un usuario se identifique. EAPI busca el login y password en la base de datos. Al encontrar el login y password, se permite el acceso del usuario al sistema.
Cursos alternos			
Paso 4:	*El login y/o password no se encuentran en la base de datos* Se notifica al usuario, solicitándole que introduzca nuevamente sus datos.		
Notas			
Paso 1:	La interfaz consiste en dos áreas de texto (para login y password) y dos botones (aceptar y registro).		

Fig. 5.1 Casos de uso del EAPI

Caso de uso:	Registrar nuevo usuario.		
Actores:	Usuario (iniciador), EAPI.		
Propósito:	Permitir que un usuario se registre en el EAPI.		
Resumen:	Un usuario llena una forma de registro, EAPI la valida, y guarda los datos en la BD.		
Curso normal de los eventos			
Actor		**Sistema**	
Paso	Acción	Paso	Acción
1. 3. 5.	El usuario presiona el botón registro de la interfaz gráfica de acceso. El usuario llena la forma de registro y presiona el botón aceptar. El usuario presiona el botón aceptar.	 2. 4.	 EAPI muestra una forma de registro. EAPI muestra los datos con los que el usuario será dado de alta.
		6.	EAPI guarda los datos en la BD.
Cursos alternos			
Paso 3:	*El usuario no llenó todos los campos de la forma* Se notifica al usuario, solicitándole que introduzca los datos que falten. *Ya existe un usuario con el mismo login* Se le pide al usuario que proporcione uno nuevo. *Los passwords no coinciden* Se le notifica al usuario, pidiéndole que verifique su password.		
Notas			
Paso 2:	La forma de registro tiene los siguientes campos; nombre, login, password y confirmar password.		

Fig. 5.1 Casos de uso del EAPI (continuación)

Caso de uso:	Mostrar lista de proyectos.		
Actores:	Herramientas CASE (iniciador), EAPI.		
Propósito:	Permitir que una herramienta del ambiente muestre a los usuarios los proyectos contenidos en el EAPI (de forma gráfica).		
Resumen:	Una Herramienta solicita al EAPI una lista con los proyectos que necesita mostrar a un usuario, EAPI busca los proyectos en la BD y crea una ventana con el contenido de la búsqueda.		
Curso normal de los eventos			
Actor		**Sistema**	
Paso	Acción	Paso	Acción
1.	La herramienta CASE, solicita al entorno una lista de proyectos de un usuario determinado.	2. 3.	EAPI busca los proyectos en la base de datos. EAPI crea una interfaz gráfica con el contenido de la búsqueda.
Cursos alternos			
Paso 3:	*No se encontraron proyectos* Se crea un dialogo que indique que no hay proyectos		
Notas			
Paso 1:	También se puede crear una lista de los proyectos con privilegio de lectura o escritura.		

Fig. 5.1 Casos de uso del EAPI (continuación)

Caso de uso:	Administrar proyectos.
Actores:	Usuario (iniciador), EAPI.
Propósito:	Permitir que el usuario cambie los privilegios de sus proyectos, o los elimine del EAPI.
Resumen:	Un usuario selecciona la opción administrar proyectos, EAPI proporciona las opciones de cambiar privilegio y eliminar, el usuario selecciona una opción y EAPI actualiza los datos.

Curso normal de los eventos

Actor		**Sistema**	
Paso	Acción	Paso	Acción
1.	El usuario selecciona la opción de administrar proyectos		
		2.	EAPI muestra una lista con los proyectos que son propiedad del usuario
3.	El usuario selecciona un proyecto de la lista		
		4.	EAPI muestra una interfaz gráfica que permite que el usuario administre el proyecto seleccionado.
5.	El usuario selecciona alguna de las opciones que le muestra la interfaz		
		6.	EAPI guarda los cambios en la base de datos

Cursos alternos

Paso 6:	*El usuario desea eliminar un proyecto que está ocupado* Se impide que el usuario elimine el proyecto, indicándole la razón.

Notas

Paso 4:	El usuario puede elegir el privilegio que desea dar a su proyecto para otros miembros del EAPI (lectura, escritura o ninguno). También puede eliminar el proyecto del entorno.

Fig. 5.1 Casos de uso del EAPI (continuación)

Caso de uso:	Mostrar herramientas CASE.
Actores:	Usuario, EAPI (iniciador).
Propósito:	Mostrar a los usuarios del EAPI, una lista secuencial con las herramientas CASE del entorno.
Resumen:	Después de validar el acceso de un usuario, EAPI muestra una lista con todas las herramientas que puede ocupar. Cuando el usuario selecciona una herramienta CASE, EAPI crea una instancia de dicha herramienta.

Curso normal de los eventos

Actor		Sistema	
Paso	Acción	Paso	Acción
2.	El usuario selecciona una herramienta.	1. 3.	EAPI muestra una interfaz gráfica que le permite al usuario seleccionar una herramienta CASE. EAPI crea una instancia de dicha herramienta.

Caso de uso:	Mostrar manual.
Actores:	Usuario (iniciador), EAPI
Propósito:	Mostrar a un usuario el Manual de una herramienta contenida en el ambiente.
Resumen:	El usuario solicita el manual, en ese momento el sistema EAPI construye un manual con páginas html y lo muestra al usuario.

Curso normal de los eventos

Actor		**Sistema**	
Paso	Acción	Paso	Acción
1.	El usuario presiona el botón *Manual* de alguna de las herramientas del Entorno.	2. 3.	El EAPI construye el manual con páginas html, mostrando el índice en un árbol. Muestra al usuario el manual.

Cursos alternos

Paso 2:	*No se encuentran las páginas html* Se muestra el error, y se construye en manual con las paginas encontradas

Notas

Paso 2:	El que desarrolla la herramienta es el que indica donde están las páginas html que necesita su manual.

Fig. 5.1 Casos de uso del EAPI (continuación)

Caso de uso:	Imprimir.		
Actores:	Usuario (iniciador), EAPI.		
Propósito:	Imprimir la información generada por las herramientas del EAPI.		
Resumen:	El usuario presiona el botón imprimir, EAPI solicita al sistema la ventana de impresión predeterminada, y muestra al usuario las impresoras donde puede mandar el trabajo de impresión.		
Curso normal de los eventos			
Actor		**Sistema**	
Paso	Acción	Paso	Acción
1.	Presionar el botón *Imprimir* de alguna de las herramientas del Entorno.	2. 3.	EAPI solicita al sistema operativo la ventana de impresión predeterminada. EAPI muestra la ventana de impresión al usuario.
4.	El usuario configura su trabajo de impresión.		
		5.	EAPI manda al trabajo a la cola de impresión de la impresora seleccionada por el usuario.
Cursos alternos			
Paso 2:	*El sistema operativo no permite que se inicie un trabajo de impresión* Se notifica la situación al usuario		

Fig. 5.1 Casos de uso del EAPI (continuación)

Caso de uso:	Abrir proyecto.		
Actores:	Herramienta CASE (iniciador), EAPI.		
Propósito:	Permitir que una herramienta CASE tenga acceso a la información generada por otras herramientas del entorno.		
Resumen:	La herramienta CASE pide al EAPI la información que necesite de un proyecto.		
Curso normal de los eventos			
Actor		**Sistema**	
Paso	Acción	Paso	Acción
1.	La herramienta CASE solicita la información de un proyecto al EAPI.	2. 3.	El EAPI busca la información solicitada en la BD. EAPI le da a la herramienta CASE la información en un formato que es fácil de manipular.
Cursos alternos			
Paso 3:	*No se puede recuperar la información de la BD* Se muestra el error.		
Notas			
Paso 3:	En esta versión del EAPI se regresa la información en estructuras de datos que son muy sencillas de manipular (arreglos, vectores, árboles).		

Fig. 5.1 Casos de uso del EAPI (continuación)

Caso de uso:	Guardar proyecto.		
Actores:	Herramienta CASE (iniciador), EAPI.		
Propósito:	Prepara las tuplas necesarias en la BD y guarda la información generada por las herramientas CASE.		
Resumen:	La herramienta CASE pide al EAPI que guarde la información que contiene.		
Curso normal de los eventos			
Actor		**Sistema**	
Paso	Acción	Paso	Acción
1. 3.	La herramienta CASE solicita al EAPI que guarde la información en la base de datos. La herramienta CASE da la información al EAPI para que sea almacenada.	2.	EAPI prepara todas las tuplas que se necesitan para almacenar la información.
		4.	EAPI guarda la información en la BD.
Cursos alternos			
Paso 2:	*No se pueden generar la tuplas* se muestra el error y se cancela el proceso		
Paso 3:	*No se puede guardar la información en la BD* se muestra el error y se cancela el proceso		
Notas			
Paso 4:	En esta versión del EAPI se necesita que los datos que se van a guardar sean serializables (la información se guarda en bytes).		

Fig. 5.1 Casos de uso del EAPI (continuación)

5.1.3 Casos de uso de Venus

Para el I-CASE Venus se identificaron cuatro casos de uso básicos que se detallan en la Fig. 5.2

<table>
<tr><td colspan="2">Caso de uso:</td><td colspan="2">Agregar una hoja al árbol de proyecto.</td></tr>
<tr><td colspan="2">Actores:</td><td colspan="2">Usuario (iniciador), Venus.</td></tr>
<tr><td colspan="2">Propósito:</td><td colspan="2">Permitir que el usuario agregue hojas al árbol de proyecto.</td></tr>
<tr><td colspan="2">Resumen:</td><td colspan="2">Un usuario selecciona la opción nuevo diagrama, EAPI pide el nombre de la hoja (del árbol) que contendrá el conjunto de diagramas que el usuario editara, el usuario da un nombre y se agrega la hoja al árbol de proyecto.</td></tr>
<tr><td colspan="4">Curso normal de los eventos</td></tr>
<tr><td colspan="2">Actor</td><td colspan="2">Sistema</td></tr>
<tr><td>Paso</td><td>Acción</td><td>Paso</td><td>Acción</td></tr>
<tr><td>1.

3.</td><td>El usuario selecciona la opción nuevo diagrama.

El usuario da el nombre.</td><td>
2.

4.</td><td>
EAPI pide el nombre de la hoja que contendrá los diagramas.

Se agrega la hoja al árbol de proyecto.</td></tr>
<tr><td></td><td></td><td>5.</td><td>Por cada hoja se agrega un área de dibujo para crear/editar diagramas.</td></tr>
<tr><td colspan="4">Cursos alternos</td></tr>
<tr><td>Paso 3:</td><td colspan="3">Ya existe una hoja con ese nombre
Venus notifica el error, y pide otro nombre para la hoja.</td></tr>
<tr><td colspan="4">Notas</td></tr>
<tr><td>Paso 1:</td><td colspan="3">En esta versión de Venus sólo se pueden crear diagramas de clase y de colaboración.</td></tr>
</table>

Fig. 5.2 Casos de uso de Venus

Caso de uso:	Borrar una hoja del árbol de proyecto.
Actores:	Usuario (iniciador), Venus.
Propósito:	Permitir que el usuario elimine hojas del árbol de proyecto.
Resumen:	El usuario selecciona una hoja del árbol de proyecto, y le pide a Venus que sea eliminada.

Curso normal de los eventos

Actor		**Sistema**	
Paso	Acción	Paso	Acción
1. 2. 4.	El usuario selecciona una hoja del árbol de proyecto. El usuario selecciona la opción borrar diagrama. El usuario confirma la opción.	 3.	Venus le pide al usuario que confirme la decisión de eliminar la hoja (se perderá toda la información que contiene).
		5.	Venus elimina permanentemente la hoja del árbol.

Notas

Paso 5:	En esta versión de Venus, no existe la opción deshacer.

Fig. 5.2 Casos de uso de Venus (continuación)

Caso de uso:	Dibujar diagrama.
Actores:	Usuario (iniciador), Venus.
Propósito:	Permitir que el usuario dibuje un diagrama en el área de dibujo que contienen las hojas del árbol de proyecto.
Resumen:	El usuario selecciona una hoja del árbol de proyecto, Venus muestra el área de dibujo que tiene la hoja, el usuario dibuja diagramas utilizando la barra de herramientas que tiene el área de dibujo.

Curso normal de los eventos

Actor		**Sistema**	
Paso	Acción	Paso	Acción
1.	El usuario selecciona una hoja del árbol de proyecto	2.	Venus muestra el área de dibujo de la hoja.
3.	El usuario selecciona alguna de las opciones para dibujar de la barra de herramientas del área de dibujo.		
		4.	Lo que se dibuja depende de la opción seleccionada.

Notas

Paso 4:	La barra de herramientas de diagramas de colaboración tiene las siguientes opciones para dibujar: nuevo diagrama, agregar mensaje, y agregar nota. La barra de herramientas de diagramas de clase tiene las siguientes opciones para dibujar: nuevo diagrama, agregar atributo, agregar operación, y agregar nota.

Fig. 5.2 Casos de uso de Venus (continuación)

Caso de uso:	Editar diagrama.
Actores:	Usuario (iniciador), Venus.
Propósito:	Permitir que el usuario edite un diagrama en el área de dibujo que contienen las hojas del árbol de proyecto.
Resumen:	El usuario selecciona una hoja del árbol de proyecto, Venus muestra el área de dibujo que tiene la hoja, el usuario edita diagramas utilizando la barra de herramientas que tiene el área de dibujo.

Curso normal de los eventos

Actor		**Sistema**	
Paso	Acción	Paso	Acción
1. 3.	El usuario selecciona una hoja del árbol de proyecto. El usuario selecciona alguna de las opciones para editar de la barra de herramientas del área de dibujo.	2.	Venus muestra el área de dibujo de la hoja.
		4.	Lo que se edita depende de la opción seleccionada.

Notas

Paso 4:	La barra de herramientas de diagramas de colaboración y de clase tienen las siguientes opciones para editar: seleccionar/mover, cortar, copiar, y pegar.

Fig. 5.2 Casos de uso de Venus (continuación)

5.1.4 Diagramas de casos de uso

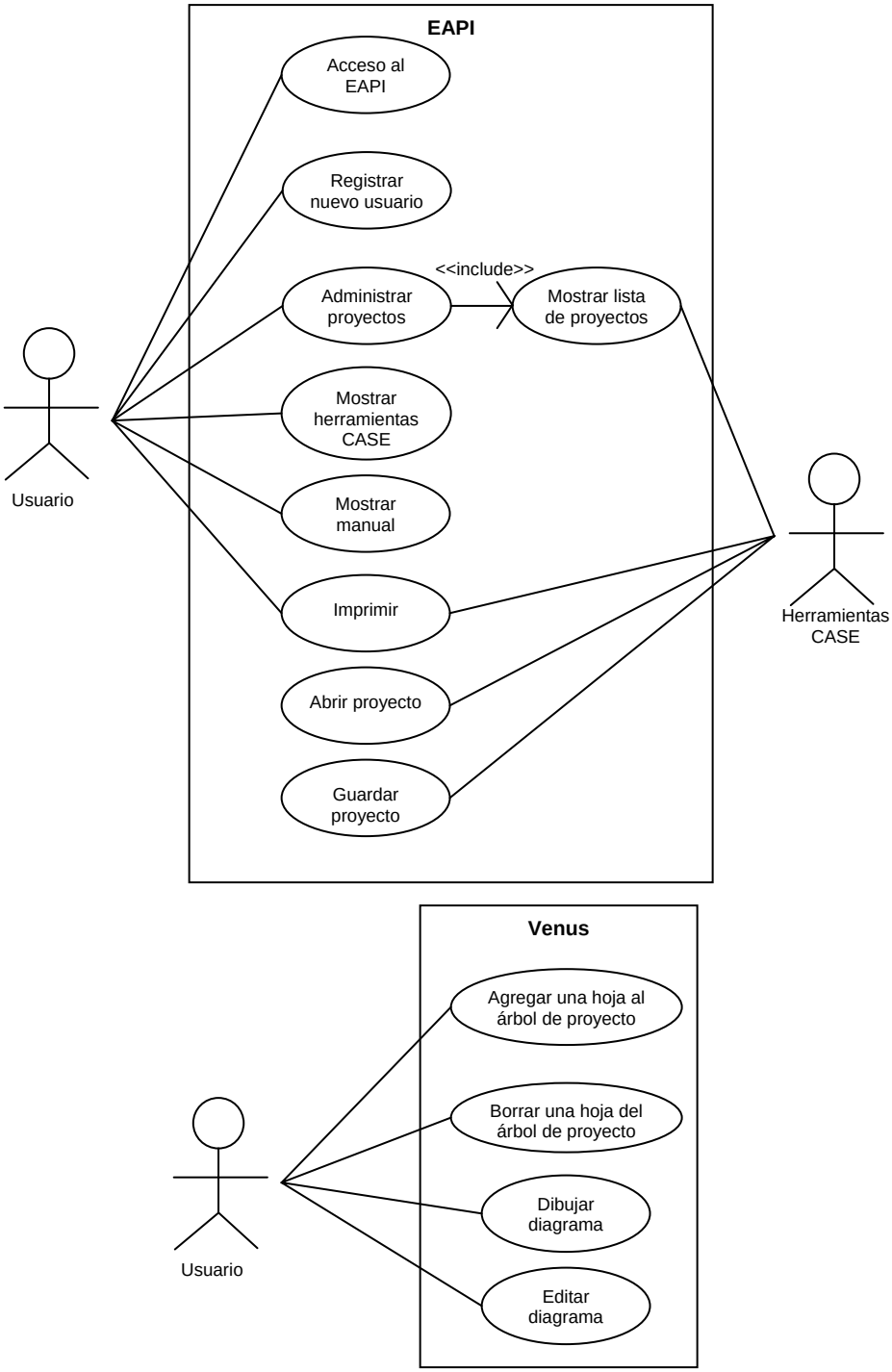

Fig. 5.3 Diagramas de casos de uso del proyecto

5.1.5 Modelo conceptual

Modelo conceptual del sistema

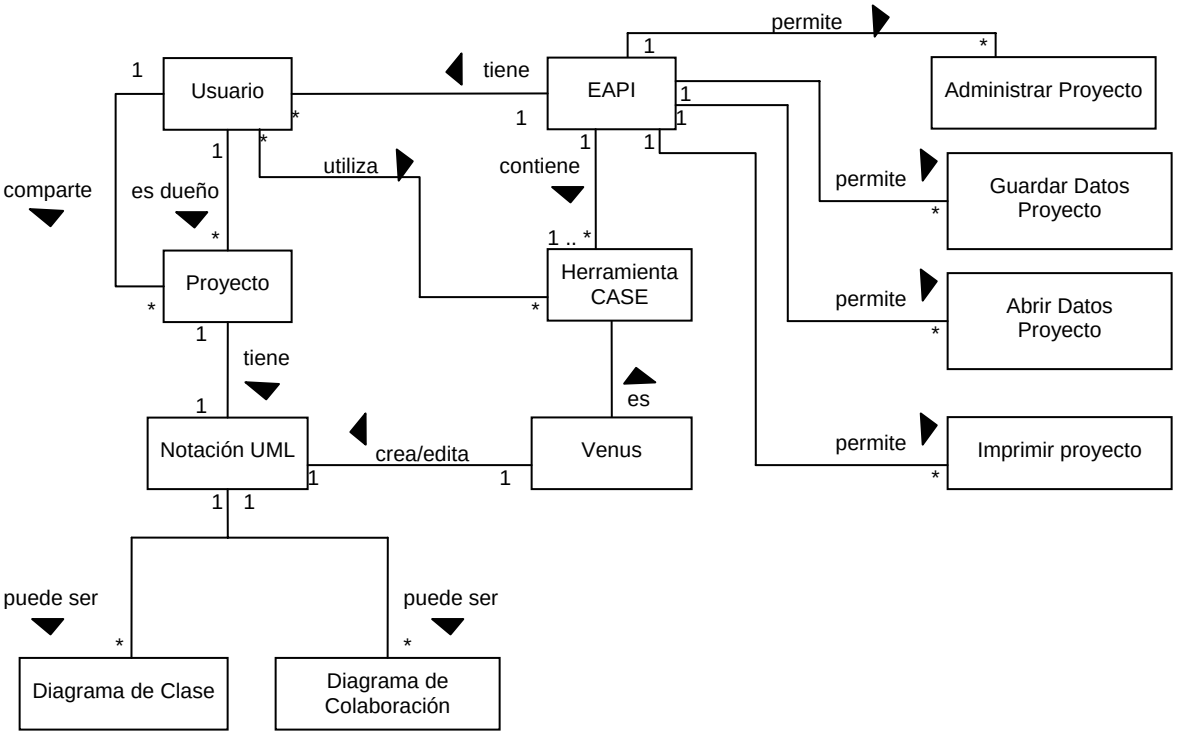

Fig. 5.4 Modelo conceptual del sistema

Modelo conceptual del árbol de proyecto (de Venus)

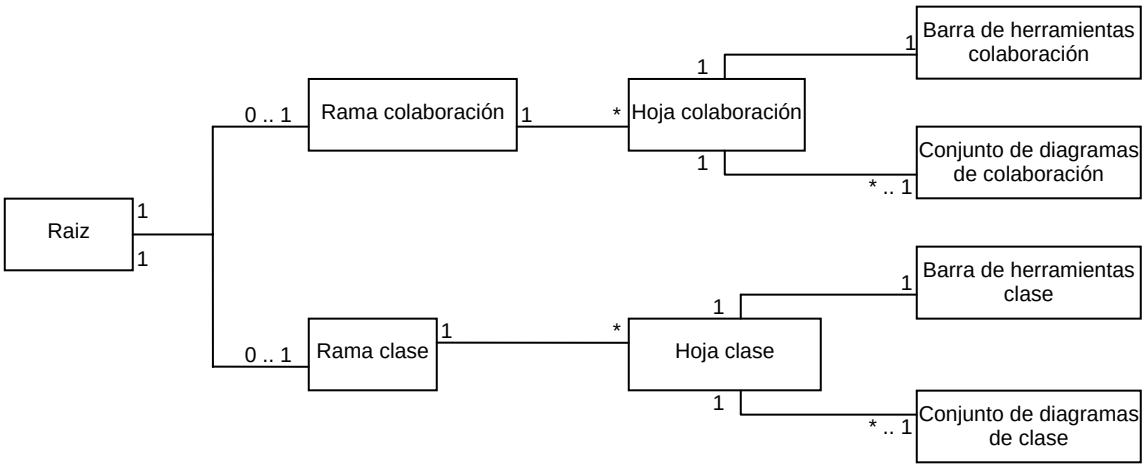

Fig. 5.5 Modelo conceptual del árbol de proyectos de Venus

5.1.6 Herramientas ocupadas

Las herramientas tecnológicas ocupadas para este proyecto fueron muy importantes en el resultado obtenido, de ahí la importancia de su elección. Después de analizar varias opciones elegí las siguientes herramientas:

5.1.6.1 Java

Java es una tecnología que simplifica considerablemente la construcción de sistemas multiplataforma, pues Java es un lenguaje OO para ser usado en Internet. Es distribuido, interpretado, robusto, seguro, concurrente (multithread) y gratuito. Proporciona el API de Swing para desarrollar interfaces gráficas de fácil uso, el JDBC (*Java DataBase Connectivity*), los archivos JAR (*Java Archive*), los JSP (*Java Server Pages*), y las firmas digitales. En el EAPI se utilizó el JSDK (*Java Standard Development Kit*) 1.4.2, ya que era el más reciente al terminar el proyecto.

5.1.6.2 MySQL

MySQL es un manejador de base de datos relacional robusto, de código abierto bajo la licencia GPL (*The GNU General Public License*). MySQL puede competir con sus famosas contrapartes comerciales (Oracle, DB2, Informix, Sybase) por sus características: no requiere de una gran número de recursos para funcionar, es extensible, robusto, rápido, multiplataforma, y fácil de administrar [Proal, 01]. En la cuenta Sun del proyecto se instaló y configuro la versión estándar 4.0.12 por ser la más reciente al elaborar el proyecto y la

primera versión estándar de MySQL que permite crear tablas tipo InnoDB, algo que antes requería recompilar el servidor de MySQL o instalar el MySQL-Max. Las tablas InnoDB proporciona lo siguiente:

- Recuperación automática ante fallas. Si MySQL se da de baja de una forma anormal, InnoDB automáticamente completará las transacciones que quedaron incompletas.

- Integridad referencial. Se pueden definir llaves foráneas entre tablas InnoDB relacionadas para asegurarse de que un registro no puede ser eliminado de una tabla si aún está siendo referenciado por otra tabla.

- Bloqueo a nivel de filas. Al usar tablas MyISAM, y tener consultas muy grandes que requiere de mucho tiempo, simplemente no se puede ejecutar más consultas hasta que terminaran las consultas que estaban en ejecución. En cambio, las tablas InnoDB usan bloqueo a nivel de filas para mejorar de manera notable el rendimiento.

- SELECTs sin bloqueo. InnoDB usa una técnica conocida como multi-versioning (similar a PostgreSQL) que elimina la necesidad de hacer bloqueos en consultas SELECT muy simples.

5.1.6.3 Tomcat

Tomcat es un servidor Web que permite ejecutar Servlets y JSP, de rápido crecimiento y uso dentro de la comunidad de programadores. Es gratuito, extensible y funciona el los sistemas operativos Unix y Windows. Para este proyecto se instaló y configuró la versión 5.0 por ser la que brindaba mayor soporte a las nuevas especificaciones de Java para los Servlets y JSP. El servidor Tomcat del EAPI se encuentra en “http://mailweb.udlap.mx:3310/”.

5.1.6.4 Red de Computadoras Sun

La dirección de Capacitación y Servicios en Sistemas (CASS) nos proporcionó una cuenta de usuario en la red de computadoras Sun (msqlroot) y dos puertos en el servidor mailweb.udlap.mx (3306, 3310) para implementar el EAPI. Elegí esta arquitectura para desarrollar el sistema por ser distribuida y estar basada en Unix.

5.1.7 Análisis de tecnologías Java

Uno de los objetivos principales del EAPI es permitir el rápido desarrollo de herramientas I-CASE basadas en el lenguaje de programación Java. Por este motivo se realizó un análisis detallado de tres de las principales tecnologías de Java (Applets, Frames, Servletes/JSP). Las conclusiones se sintetizan el la Fig. 5.6.

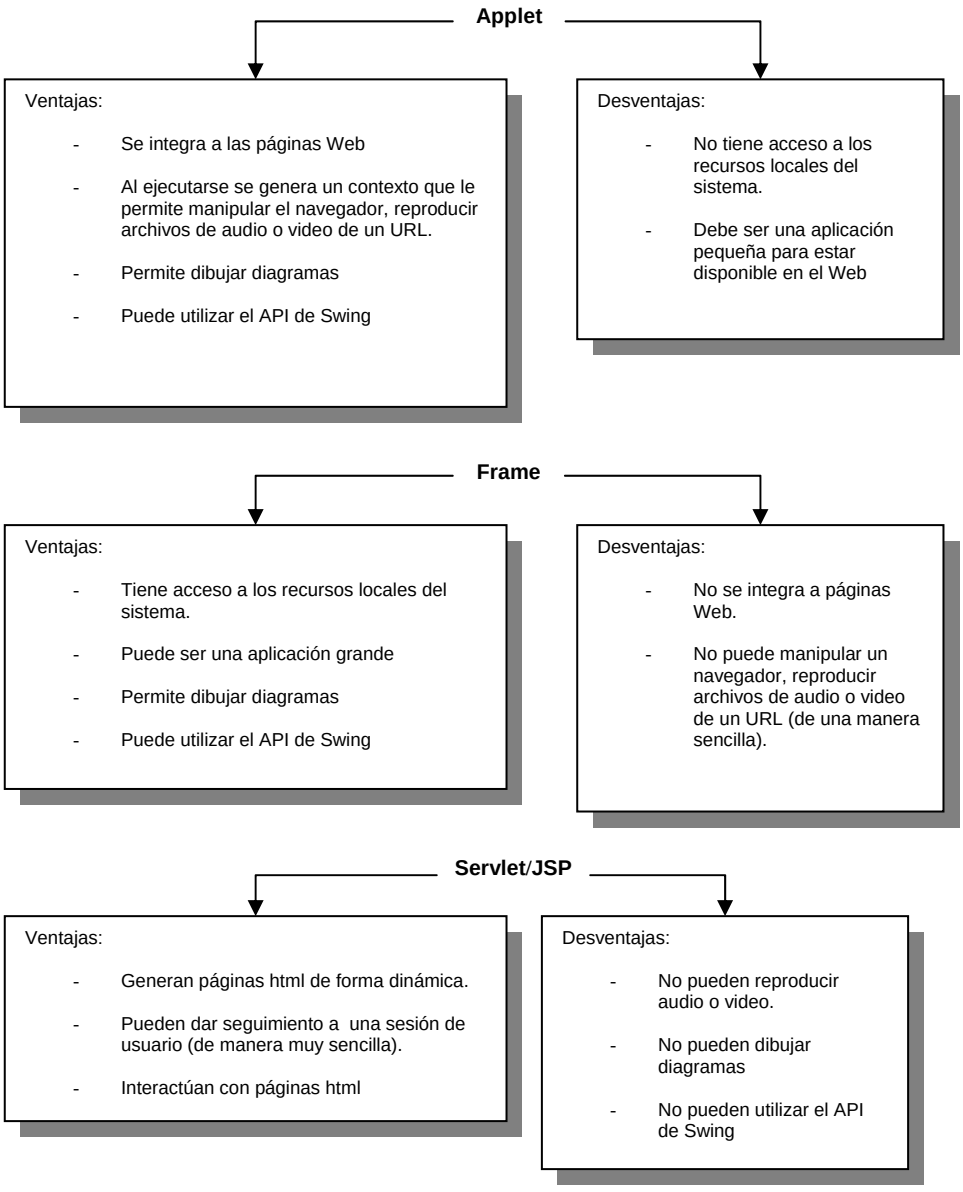

Fig. 5.6 Ventajas y desventajas de Applets, Frames, y Servlets/JSP

Después de evaluar las ventajas y desventajas de estas tecnologías se agregaron los siguientes requerimientos al EAPI:

- Implementar el acceso del EAPI con un JApplet.

- Permitir que los Frames que se generen en el EAPI tengan acceso al contexto de ejecución del JApplet de acceso, para que también puedan manipular el navegador, reproducir archivos de audio o video de un URL.

- Crear una firma digital para el EAPI, de esta forma todas las clases que se desarrollen en el EAPI tienen garantizado (si el usuario lo autoriza), el acceso a los recursos locales del sistema donde se ejecuten.

- Permitir que los Frames del ambiente tengan comunicación con los Servlets y JSPs.

- Realizar un diseño detallado que agregue las siguientes ventajas al EAPI:

    - *Reutilización*: Las clases se deben diseñar para ser reutilizadas por todas las herramientas contenidas en el entorno.
    - *Programación más sencilla*: Permitir que las herramientas CASE se puedan elaborar a partir de piezas proporcionadas por el entorno (componentes).
    - *Cuidar la integridad:* Las estructuras de datos sólo pueden utilizar métodos específicos (mayor seguridad).
    - *Mantenimiento más sencillo:* Las clases del EAPI deben efectuar sus funciones los más independientemente posible.

5.2 Diseño del sistema

5.2.1 Diagramas de paquetes

5.2.1.1 Paquete Acceso

El paquete acceso consta de tres clases (ver Fig. 5.7). *Acceso* muestra y valida una forma para acceder al sistema EAPI (es la clase principal del archivo “eapi.jar”), su función es poder ejecutar el Entorno sin necesidad de un browser (muy útil en la fase de desarrollo de herramientas CASE). La funcionalidad de la clase *AccesoApplet* es similar a la de *Acceso*, la única diferencia es que es invocada por un browser (netscape o explorer). *Registro* muestra y valida una forma de alta, si los datos pasan las pruebas de validación son almacenados en la base de datos.

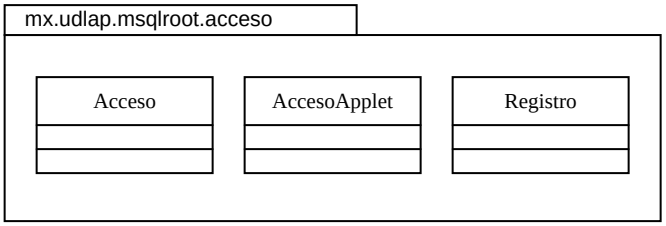

Fig. 5.7 Paquete acceso

5.2.1.2 Paquete Eapi

EAPI_Frame muestra una lista con las herramientas CASE del EAPI, permitiendo la comunicación del usuario con la herramienta que seleccionó. *ManualEapi* muestra el

manual de usuario del entorno. Y la clase *Usuario* almacena todos los datos de un usuario del EAPI (ver Fig. 5.8).

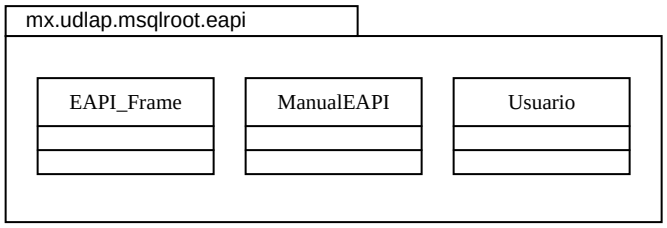

Fig. 5.8 Paquete Eapi

5.2.1.3 Paquete Proyecto

El paquete proyecto permite la comunicación de las herramientas CASE del Entorno con el depósito de proyectos (MySQL), y está compuesto por cuatro clases (ver Fig. 5.9). *AbrirProyecto* es una clase que tiene un constructor privado que es invocado por dos de sus métodos públicos; su método *mostrarDialogoAbrirProyecto* funciona como un selector de archivos (file chooser) invocado por las clases que necesitan que el usuario seleccione un proyecto del EAPI, dicho método regresa null cuando el usuario cancela la selección o cierra el dialogo (puede mostrar proyectos con privilegio de lectura o escritura, según se le especifique). El segundo método público de esta clase es *abrirDatosUML*, que es invocado por las clases que necesitan la información contenida en los diagramas UML generados por Venus (por el momento este método sólo es invocado por Venus).

La clase *AdministrarProyecto* es la responsable de cambiar los privilegios de un proyecto (lectura, escritura o ninguno), también es la encargada de eliminar un proyecto del

EAPI, antes de eliminar un proyecto verifica si alguna herramienta CASE está utilizando una parte del mismo (si esto ocurre no se elimina).

La última clase de este paquete es *GuardarProyecto*. Esta clase tiene dos funciones principales, la primera es preparar todas tablas que ocupan las herramientas CASE para almacenar la información que se genera por ellas (por el momento sólo se preparan las tablas del I-CASE Venus), y la segunda es guardar la información en el depósito de proyectos.

Fig. 5.9 Paquete proyecto

5.2.1.4 Paquete *Swing*

Todos los Frames y Panels del entorno extienden de las dos clases de este paquete (ver Fig 5.10). La clase IFrame contiene métodos que facilitan el uso del Frame, a demás permite que pueda invocar Servlets/JSP (por medio de un browser). Por otro lado la clase IPanel puede imprimir su contenido de una forma muy sencilla, sólo se necesita invocar a su método *mostrarDialogoImprimir*.

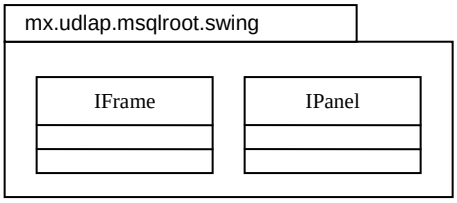

Fig. 5.10 Paquete swing

5.2.1.5 Paquete *Plaf*

En este paquete se encuentran las clases que modifican el *Look and Feel* (Aspecto y Sensación) de las herramientas contenidas en el entorno. Por el momento este paquete sólo contiene la clase *EAPILookAndFeel*, ver Fig. 5.11 (se espera que en próximas versiones se aumente el número de clases).

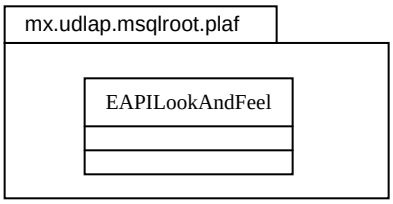

Fig. 5.11 Paquete plaf

5.2.1.6 Paquete *Theme*

Este paquete tiene dos clases (ver Fig. 5.12) que permiten personalizar el color (temas) del *Look and Feel* de las aplicaciones contenidas en el EAPI. La clase *BlueTheme* esta basada en mi color favorito, el azul. La otra clase del paquete es *Colors*, esta clase se

encarga de proporcionar un conjunto de colores para los temas. La lista de colores fue proporcionada por Karsten Lentzsch, desarrollador del proyecto JGoodies [Lentzsch, 03].

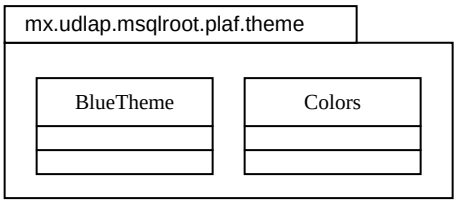

Fig. 5.12 Paquete theme

5.2.1.7 Paquete Útil

Este paquete tiene tres clases (ver Fig. 5.13). La primera es la clase *Conexion* que como su nombre lo indica permite abrir una conexión al DBMS MySQL. La clase *Manual* es un IFrame del que extienden los manuales de las herramientas CASE contenidas en el EAPI; tiene la ventaja que despliega la información en documentos HTML (es muy fácil hacer las páginas del manual, y permite hipertexto), y el contenido se organiza con un árbol. La última clase es *Static*, esta clase almacena un Applet para que cualquier IFrame pueda usarlo para desplegar un documento HTML, Servlet o JSP en el navegador (*browser*) origen de la aplicación.

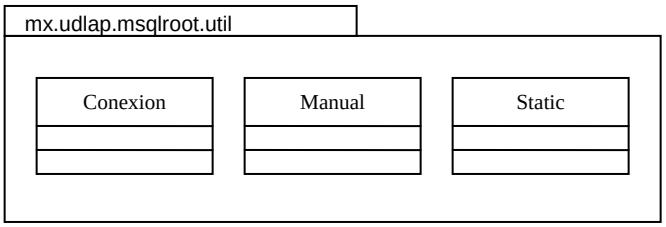

Fig. 5.13 Paquete útil

5.2.1.8 Paquete Venus

Para este paquete se diseñaron ocho clases (ver Fig. 5.14). La clase más importante es *Venus*, se encarga de generar la interfaz gráfica y de gestionar a las clases involucradas en este proceso. La clase *ManualVenus* (como su nombre lo indica) muestra el manual del CASE Venus. La clase *TreePanel* contiene la interfaz gráfica del árbol de proyecto, para crear dicha interfaz utiliza a *PanelColaboracion* (contiene un conjunto de diagramas de colaboración) y a *PanelClase* (contiene un conjunto de diagramas de clase), también hace uso de *PanelDeNoSeleccion*, en realidad esta clase es un auxiliar que es invocada cuando el usuario selecciona una parte del árbol que no tiene diagramas, por ejemplo, la raíz del árbol, la raíz de la rama de diagramas de clase o la raíz de la rama de diagramas de colaboración. Por último las clases *MouseListenerClb* y *MouseListenerClase*, permiten responder a los eventos del mouse en las instancias de PanelColaboración y PanelClase (respectivamente).

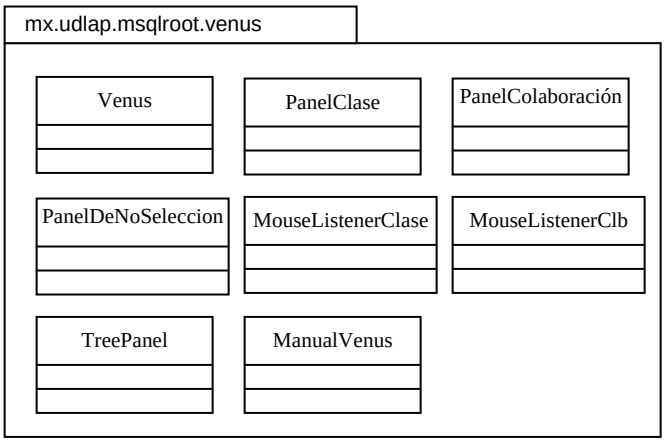

Fig. 5.14 Paquete venus

5.2.1.9 Paquete Diagrama

Este paquete contiene cuatro clases (ver Fig. 5.15) que permiten dibujar/editar los diagramas de colaboración y de clase. La clase *Diagrama* implementa los métodos y propiedades generales a los diagramas, como son, traslación, actualizar tamaño y selección. Las clases *DiagramaClase* y *DiagramaClb*, extienden directamente de *Diagrama* e implementan métodos y propiedades de cada uno de los tipos de diagrama que representan. La clase *TextGraphic*, permite crear/editar texto en la notación UML.

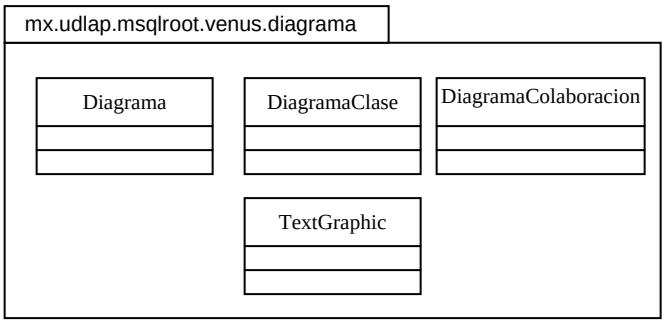

Fig. 5.15 Paquete diagrama

5.2.2 Diagramas de clase

Para desarrollar este proyecto se utilizó el proceso unificado de desarrollo de software (descrito en el capítulo III). Al momento de redactar este documento, el proyecto se encontraba en su tercer ciclo iterativo, y se habían desarrollado las clases descritas en la figura 5.16.

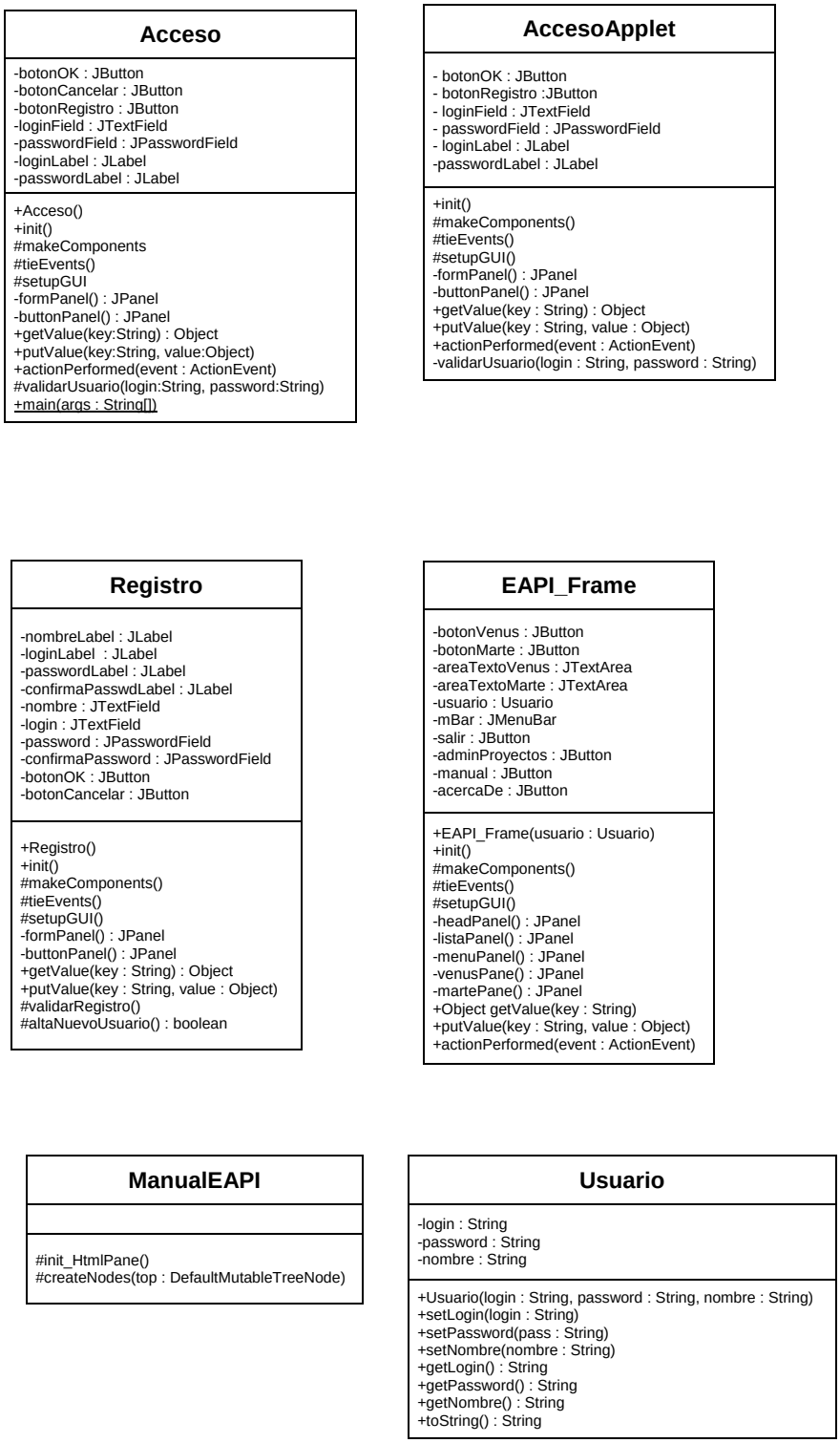

Fig. 5.16 Diagramas de clase del proyecto

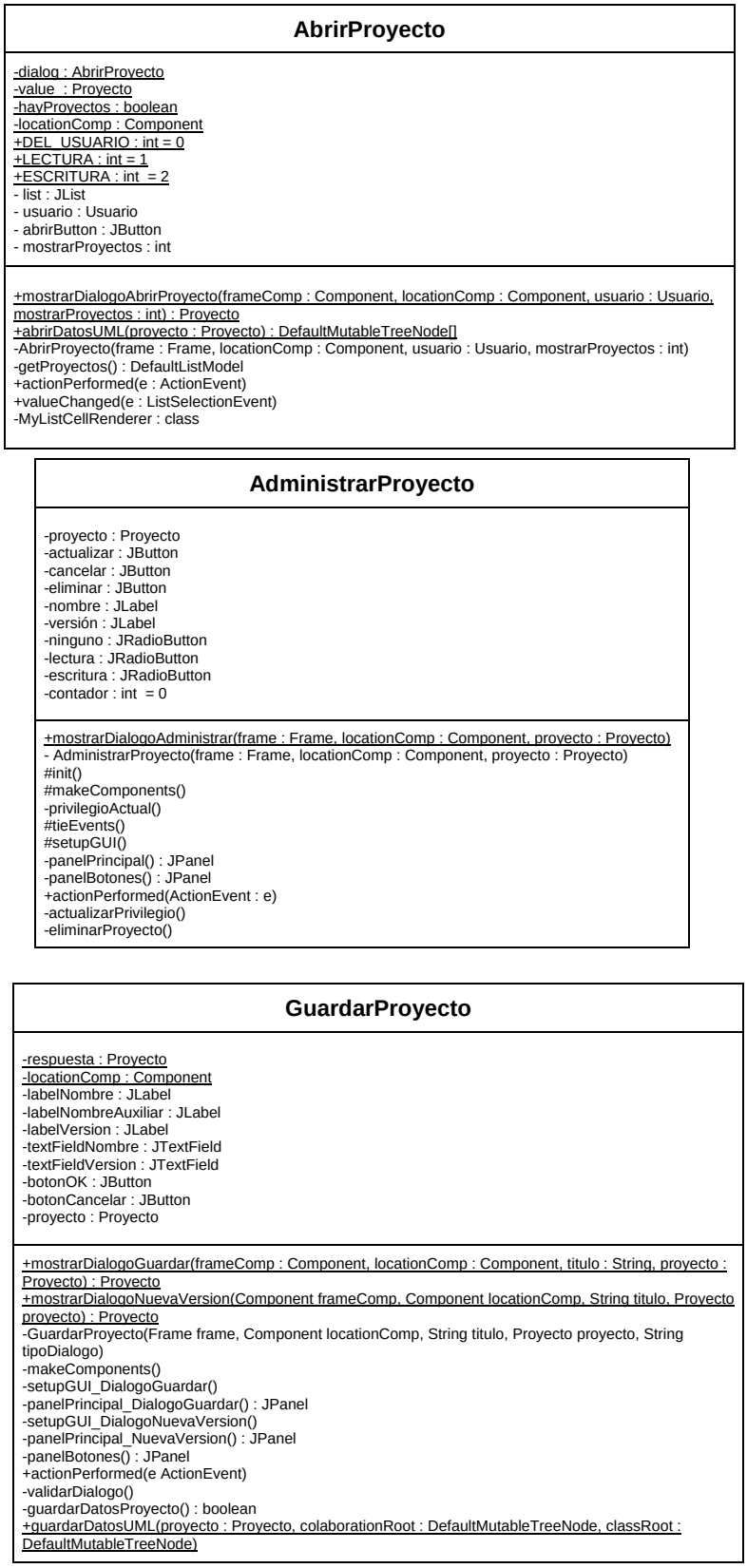

Fig. 5.16 Diagramas de clase del proyecto (continuación)

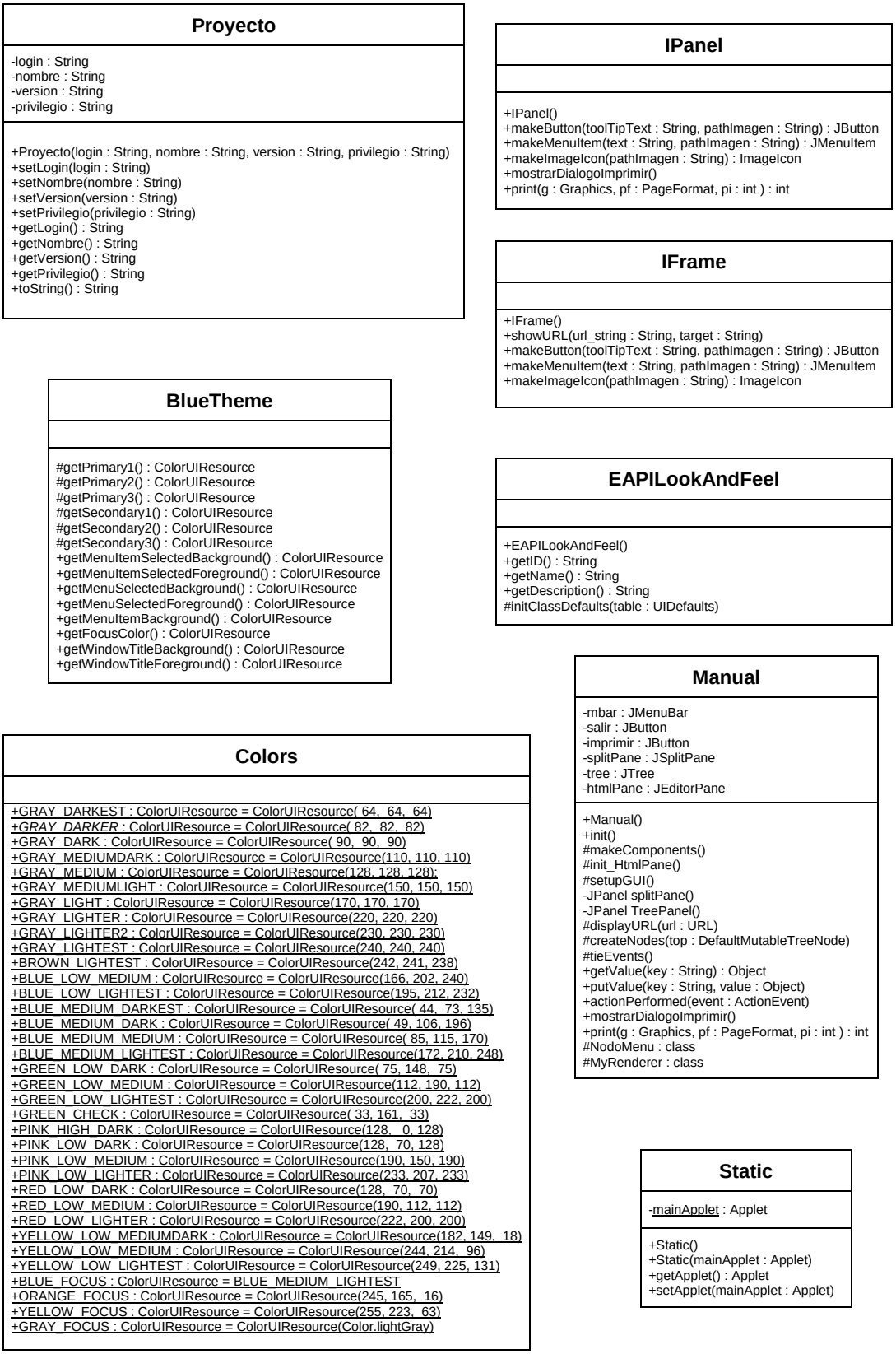

Fig. 5.16 Diagramas de clase del proyecto (continuación)

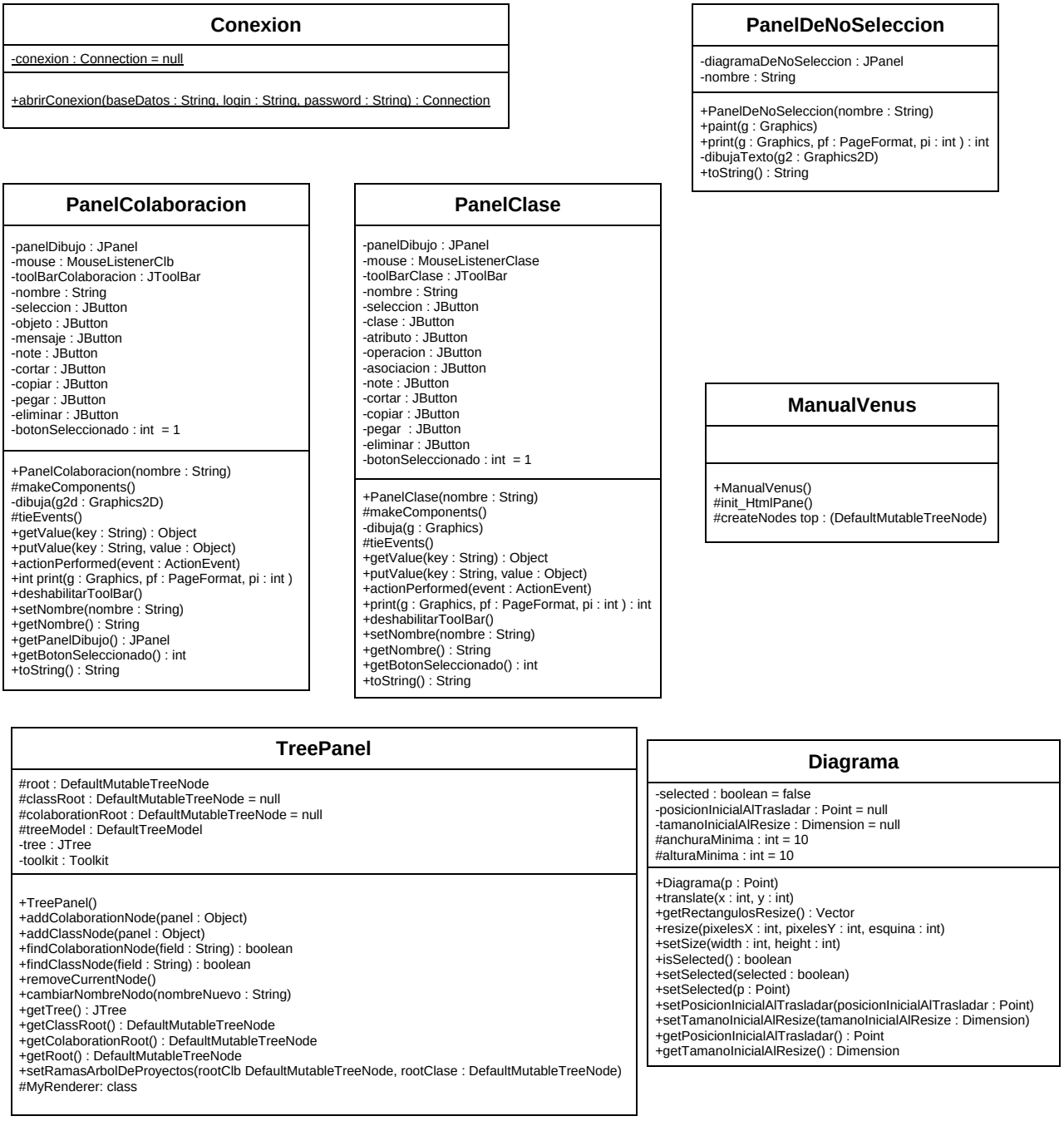

Fig. 5.16 Diagramas de clase del proyecto (continuación)

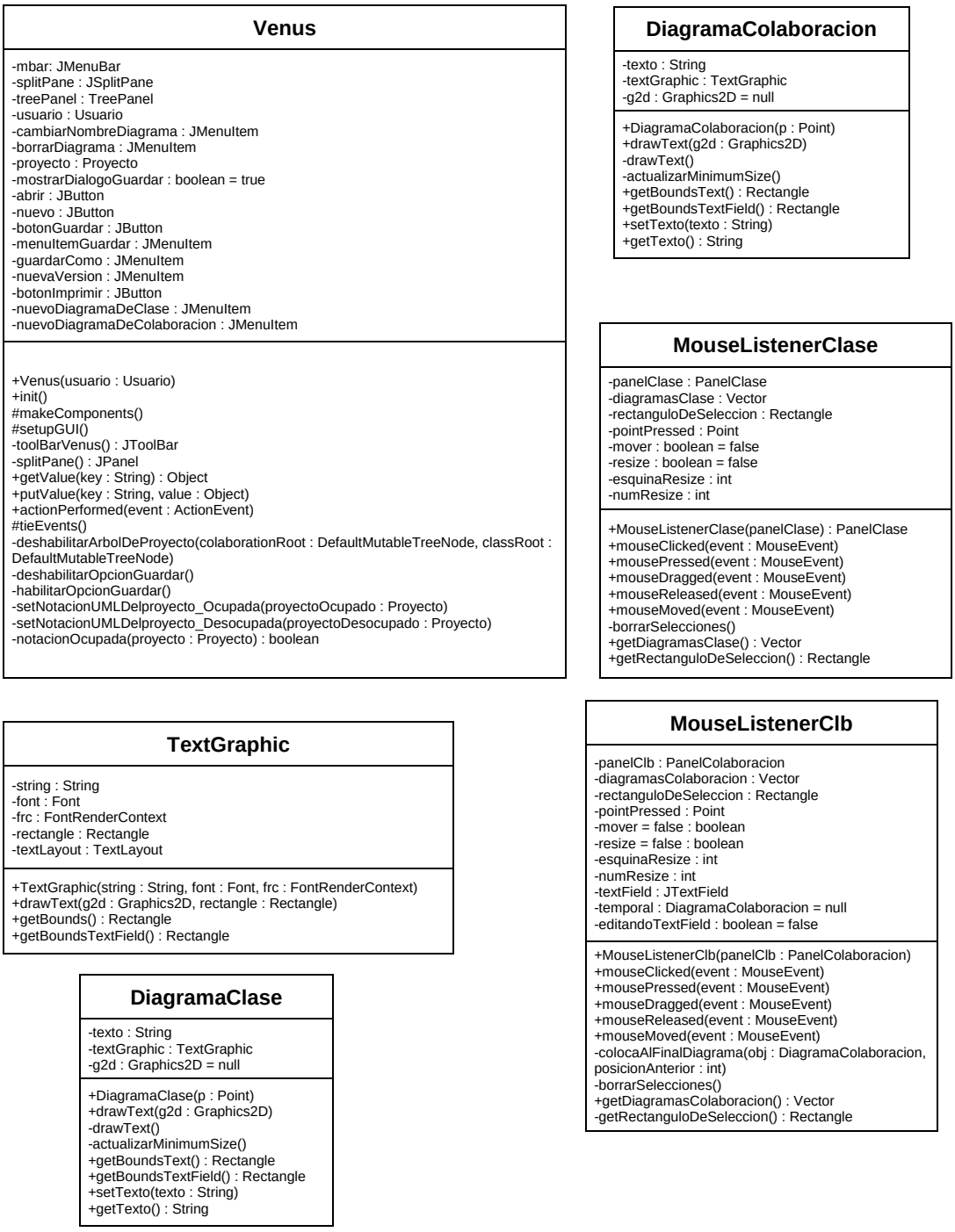

Fig. 5.16 Diagramas de clase del proyecto (continuación)

5.2.3 Análisis de herencias e implementación de interfaces

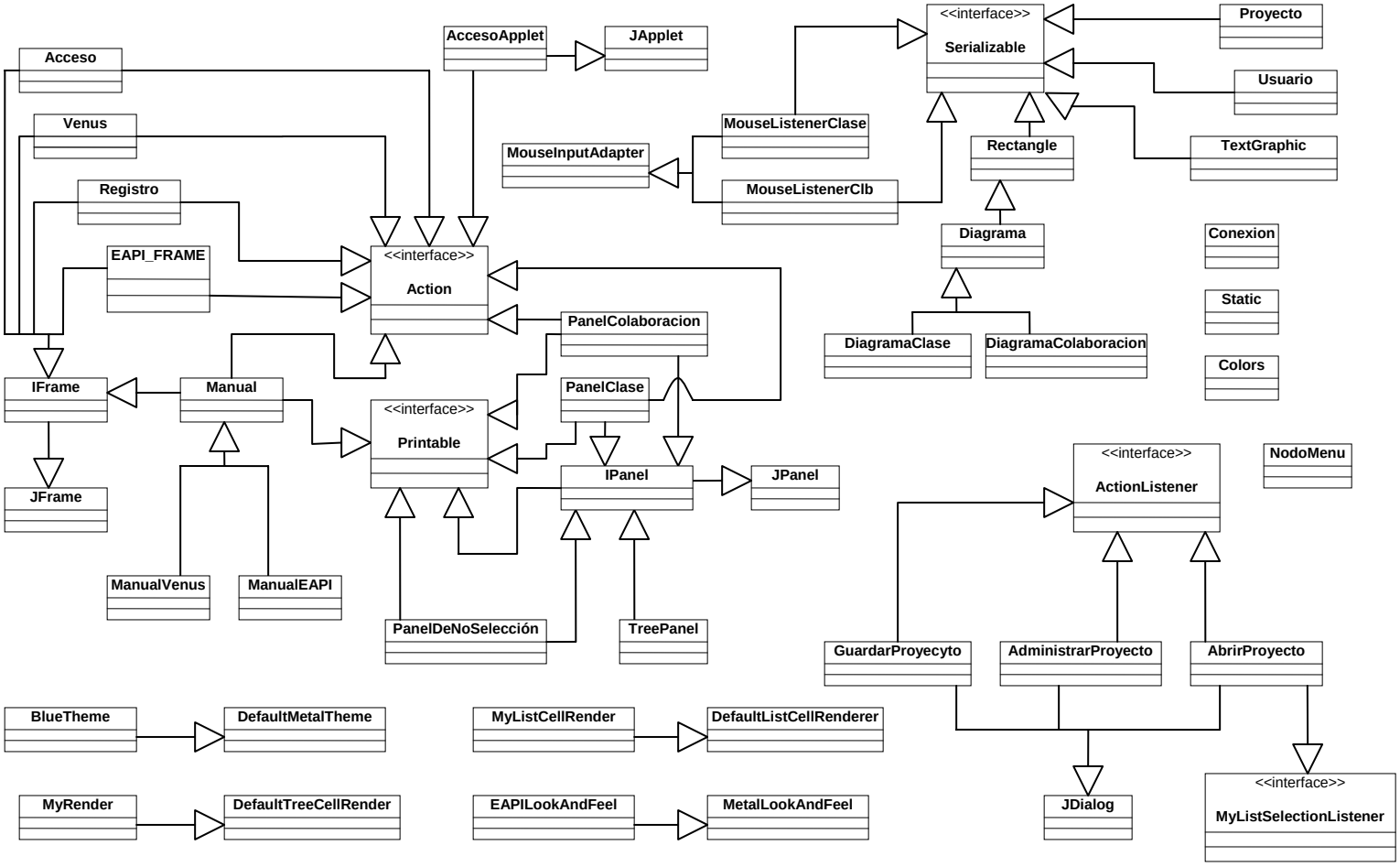

Fig. 5.17 Herencias e implementación de interfaces en el proyecto

5.2. Análisis de relaciones de las clases desarrolladas

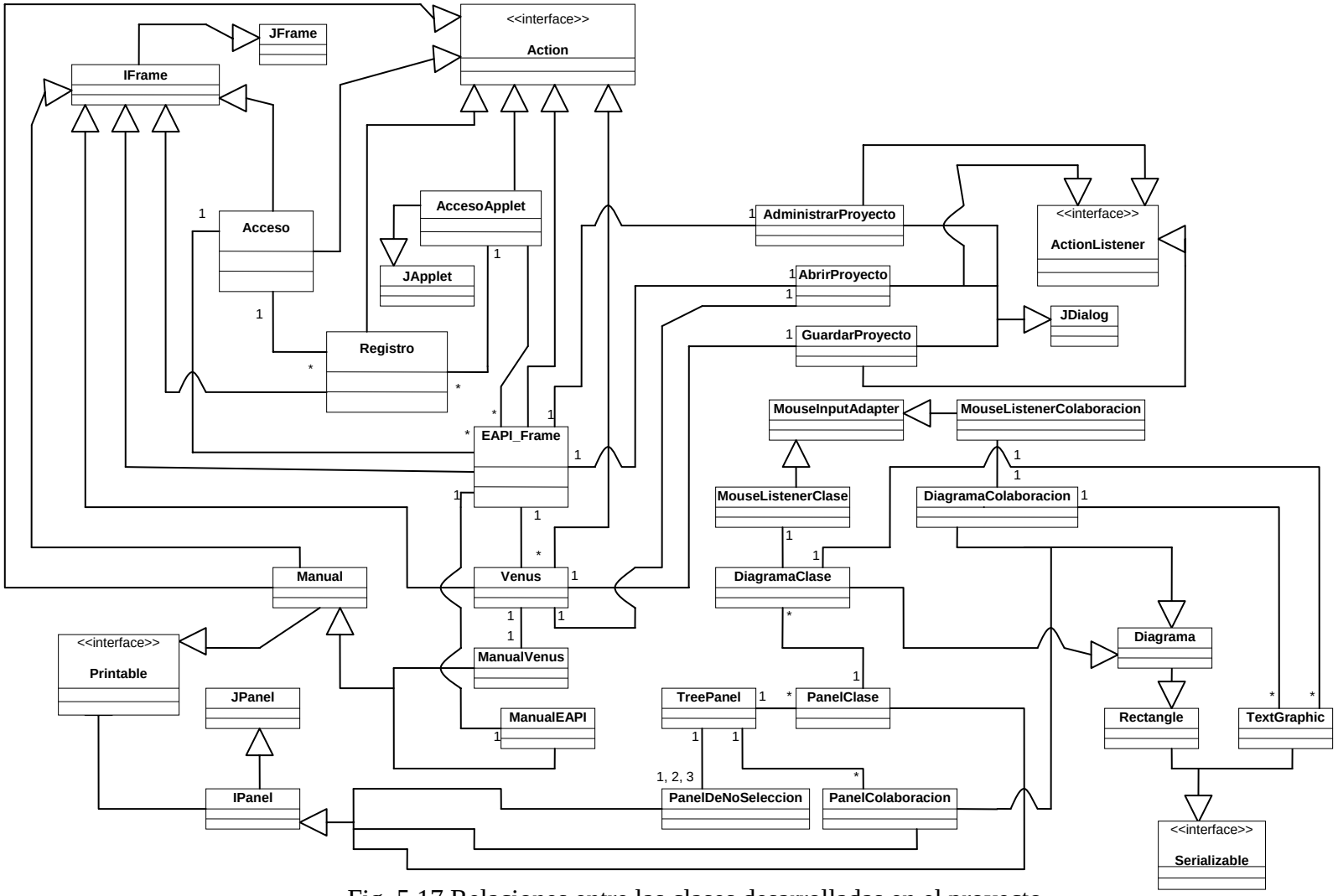

Fig. 5.17 Relaciones entre las clases desarrolladas en el proyecto

5.3 Implementación

5.3.1 Implementación del EAPI

5.3.1.1 Cuenta Sun

El primer paso en la implementación del EAPI fue configurar e instalar las herramientas que se utilizan en la cuenta msqlroot. En el sistema operativo Solaris se editó el archivo *.cshrc* agregándole las siguientes variables de ambiente:

```
#########################################################################
# Initial file to set proper environment edited by Joel Rea
#########################################################################
setenv MSQLROOT_HOME  /archivos/vol880-04/msqlroot
## Developers stuff
setenv JAVA_DIR   $MSQLROOT_HOME/j2sdk
setenv CLASSPATH :.:${HOME}:$HOME/html
setenv JAVA_HOME $MSQLROOT_HOME/j2sdk
#################################################################
##                    Configuracion de JAVA
#################################################################
## Definicion (minima) de CLASSPATH
#
setenv CLASSPATH :.:${HOME}:$HOME/html
setenv JAVA_HOME $MSQLROOT_HOME/j2sdk

## Driver de MySQL para java JDBC
#
setenv CLASSPATH ${CLASSPATH}:$HOME/mysql/driverMySQL/mm.mysql-2.0.4-bin.jar

## Archivo jar de TOMCAT
#
setenv CLASSPATH ${CLASSPATH}:$HOME/jakarta-tomcat/common/lib/servlet.jar

## path de los paquetes del EAPI
#
setenv CLASSPATH ${CLASSPATH}:$HOME/EAPI/bin/
setenv CLASSPATH ${CLASSPATH}:$HOME/EAPI/src/
```

Para poder trabajar cómodamente en el entorno se organizaron los archivos en seis carpetas (ver Fig. 5.18). En cada una de las carpetas se agregó un archivo “readme.txt” que explica detalladamente su contenido.

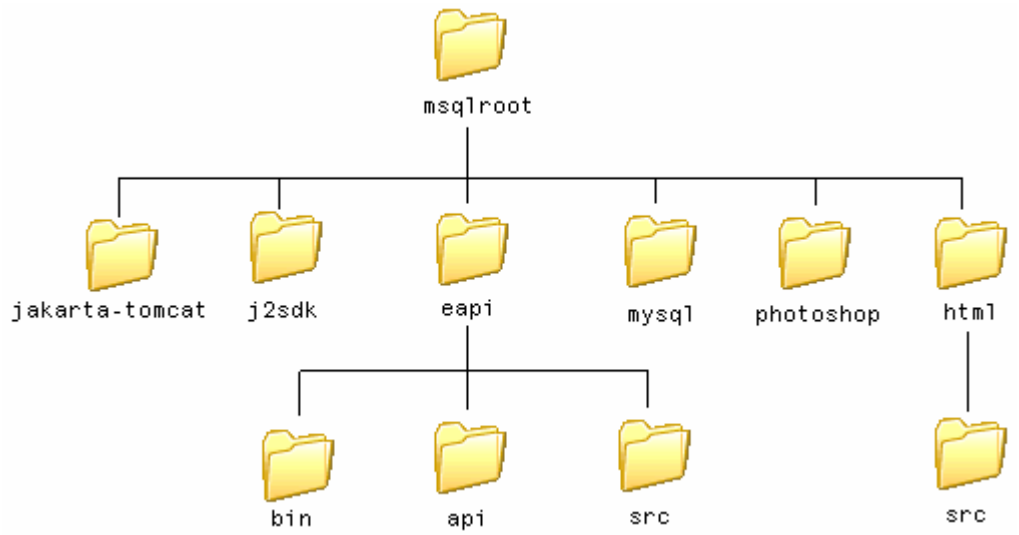

Fig. 5.18 Organización de archivos en la cuenta *msqlroot*

La carpeta “eapi" se subdivide en las carpetas “bin”, “api” y “src”. La carpeta “bin” contiene todos los archivos class del proyecto, en la carpeta src están todos los archivos fuente, y en la carpeta “api” se agregó el API (*Aplication Programming Interface*) de las clases desarrolladas para que forme parte de la documentación del proyecto. En la carpeta “html” están todos los archivos que forman la página Web del proyecto, y en “src” se encuentra el archivo “eapi.jar” (archivo que empaqueta todas las clases). La carpeta “photoshop” no contiene a la aplicación *Adobe Photoshop*, en realidad, en ella están los archivos PSD de las imágenes creadas para el EAPI, Venus y la página Web.

5.3.1.2 Modelo de integración

A continuación se describe brevemente como fueron implementadas cada una de las capas del modelo de referencia de integración de Roger Pressman [Pressman, 01] (Ver Fig. 2.2):

Capa de interfaz de usuario

Esta capa fue implementada por las clases ICaseLookAndFeel, Frame, IPanel, BlueTheme y Colors; estas clases permiten dar un mismo aspecto a todas las herramientas contenidas en el ambiente integrado (ver Fig. 5.19).

Fig. 5.19 Iconos utilizados en el EAPI

Capa de herramientas

Para lograr que el entorno sea accesible por un navegador la entrada al sistema es un JApplet implementado con la clase AccesoApplet, el contexto que genera se almacena en la clase Static. Esto permite que cualquier instancia de la clase IFrame (clase que extiende de JFrame) pueda manipular el navegador, invocar a un Servlet o JSP, y

reproducir audio o video de un URL (al igual que un Applet).

Para reducir el tamaño del sistema y mejorar el rendimiento del navegador que lo ejecute, se comprimieron y empaquetaron todas las clases e imágenes del EAPI en un archivo JAR (eapi.jar). Se utilizó el siguiente manifiesto para el archivo JAR:

```
Class-Path: libs/mm.mysql-2.0.4-bin.jar
Created-By: 1.4.2 (Sun Microsystems Inc.)
Main-Class: mx.udlap.msqlroot.acceso.Acceso
Implementation-Title: ICase
Implementation-Vendor: Fundación Universidad de las Américas, Puebla.
```

El archivo JAR fue firmado con un certificado digital (para mayor información de certificados y firmas digitales, ver el apéndice III) con duración de diez años (ver Fig. 5.20) para permitir que cualquier JApplet del EAPI tenga acceso a los recursos locales del sistema operativo (al igual que un Frame).

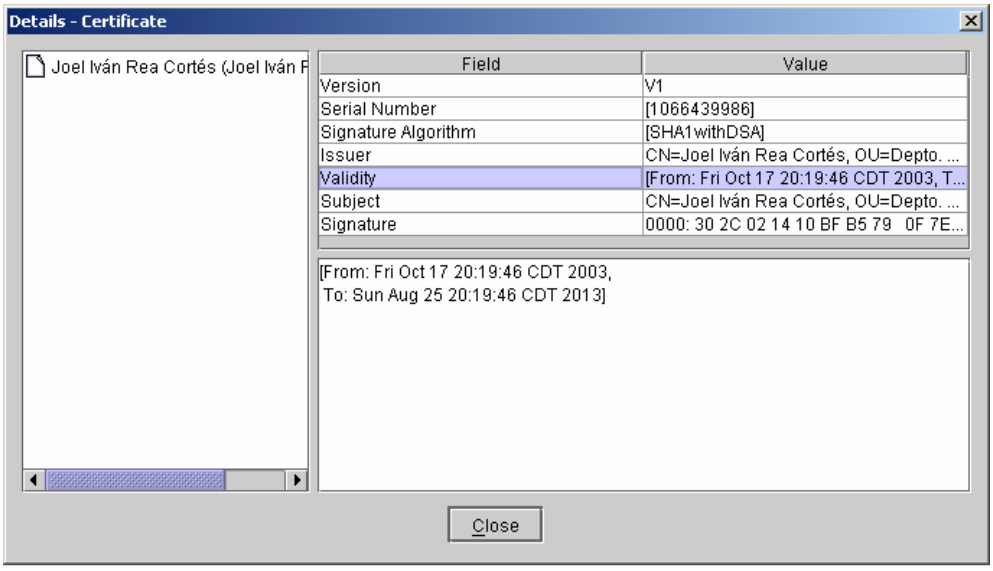

Fig. 5.20 Certificado digital del EAPI

Si un Servlet o JSP necesita dibujar, pueden solicitarlo a un JApplet o a un IFrame del EAPI. Para proveer una simplificación en la transferencia de datos entre las herramientas del entorno se implementaron las clases Usuario y Proyecto.

Capa de gestión de objetos

Para esta capa desarrollé las siguientes clases: Conexion, AbrirProyecto, AdministrarProyecto, GuardarProyecto y Proyecto. Estas clases sirven como estándares a las herramientas CASE del EAPI para conectarse a el depósito de proyectos. También se encargan del control de cambios y versiones (para conocer más detalles leer 5.2.1.3).

Capa de depósito compartido

Esta capa se implementó con MySQL, el demonio (o guardián) del manejador se encuentra activo en el servidor mailweb.udlap.mx en el puerto 3306. La BD (Base de Datos) principal del EAPI se llama icase, contiene los datos de los proyectos, y usuarios.

El modelo Entidad-Relación de la base de datos icase es relativamente sencillo, pues sólo abarca información básica de los proyectos, usuarios, y la notación UML que genera Venus (ver Fig. 5.21).

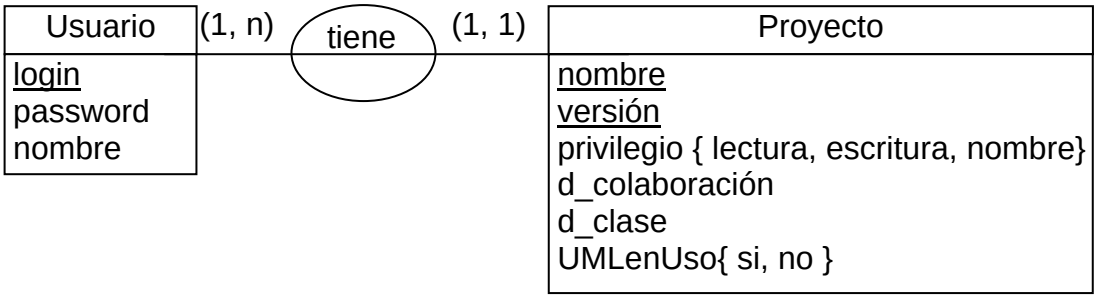

Fig. 5.21 Diagrama Entidad-Relación de la BD icase

A continuación se muestra el código SQL que se utilizó para crear las tablas:

```
mysql> CREATE TABLE usuario (
    -> login varchar(15) NOT NULL,
    -> password varchar(15) NOT NULL,
    -> nombre varchar(40) NOT NULL,
    -> PRIMARY KEY (login),
    -> KEY pkusuario(login) )
    -> Type = InnoDB;
Query OK, 0 rows affected (0.11 sec)

mysql> CREATE TABLE proyecto (
    -> login varchar(15) NOT NULL,
    -> nombre varchar(40) NOT NULL,
    -> version varchar(40) NOT NULL,
    -> privilegio varchar(10) default 'ninguno',
    -> d_colaboracion mediumblob,
    -> d_clase mediumblob,
    -> UMLenUso char(2) DEFAULT 'no',
    -> PRIMARY KEY(login,nombre,version),
    -> KEY pkusuario(login),
    -> FOREIGN KEY(login) REFERENCES usuario(login) ON UPDATE CASCADE
ON DELETE CASCADE,
    -> ) Type = InnoDB;
Query OK, 0 rows affected (0.12 sec)
```

5.3.1.3 Portabilidad y mantenimiento

El sistema EAPI se implementó con tecnologías multiplataformas (Java, Jakarta, MySQL) lo que permite una migración a cualquier sistema operativo basado en Unix o Windows. Otra característica interesante del sistema es que se encuentra modulado y documentado, cabe hacer notar que se invirtió mucho tiempo en este aspecto ya que la arquitectura del ambiente integrado aún es básica y se espera que posteriormente sea mejorada. De ahí la necesidad de excelente documentación.

5.3.1.4 Interfaces gráficas

Las siguientes clases del EAPI cuentan con interfaz gráfica (ver Fig. 5.22), Acceso, AccesoApplet, Registro, EAPI_Frame, ManualEAPI, AbrirProyecto, AdministrarProyecto, y GuardarProyecto.

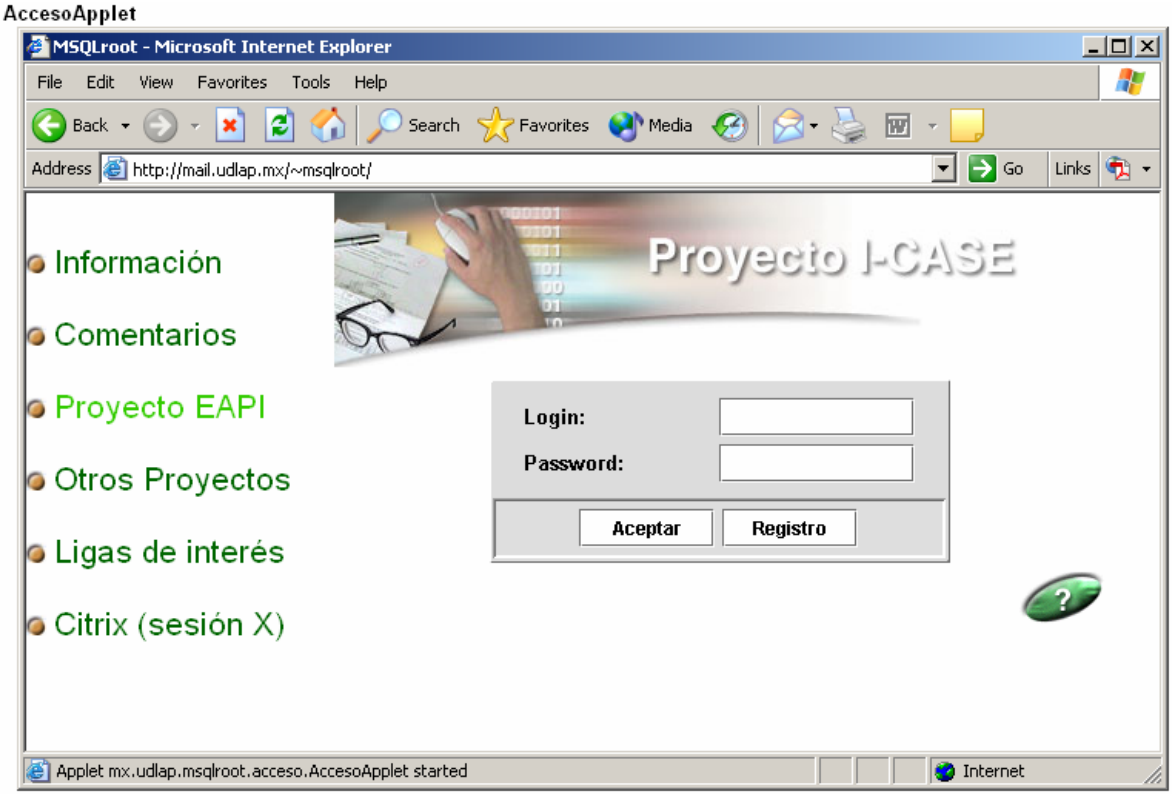

Fig. 5.22 Interfaces gráficas de las clases del EAPI

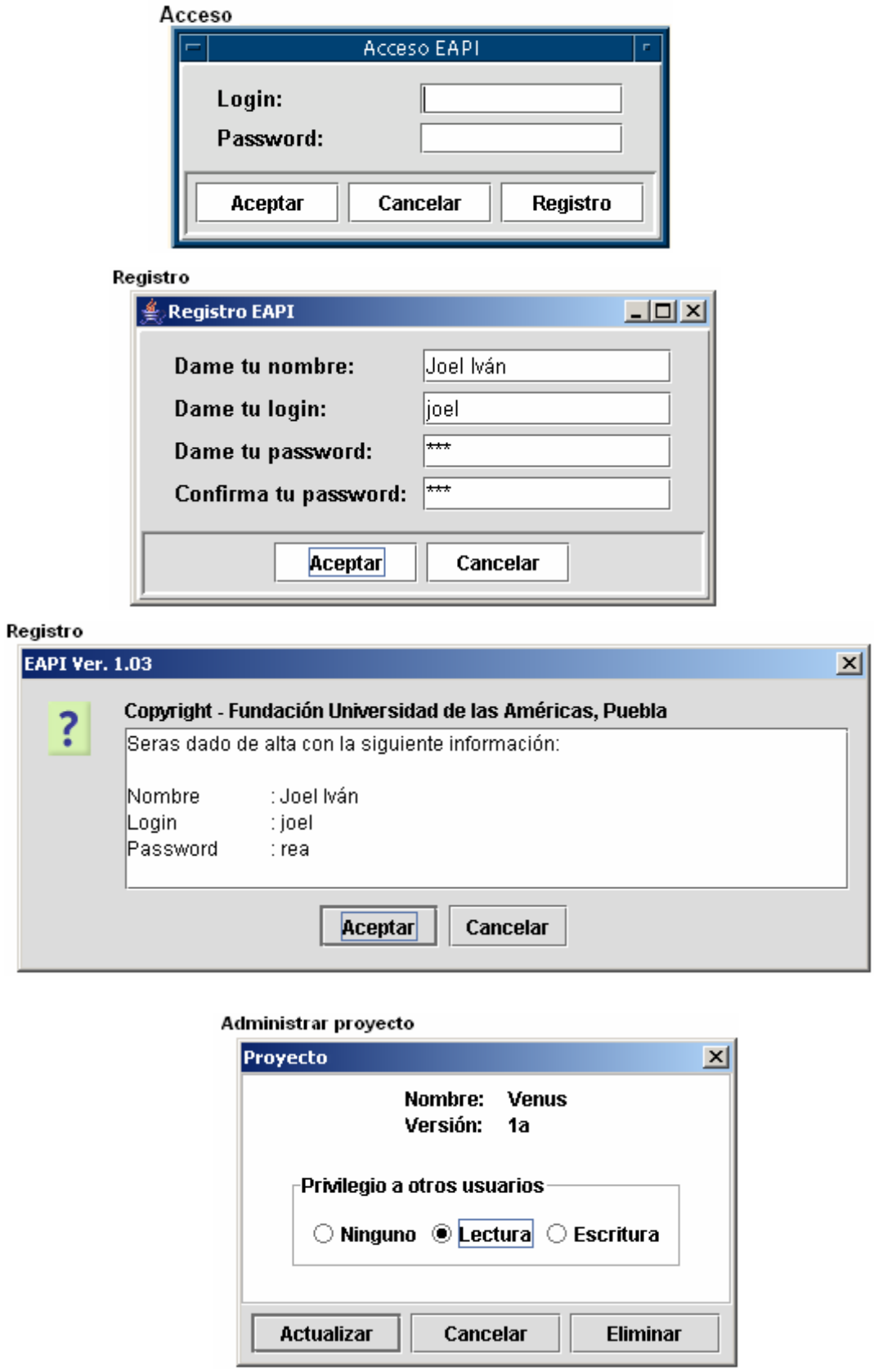

Fig. 5.22 Interfaces gráficas de las clases del EAPI (continuación)

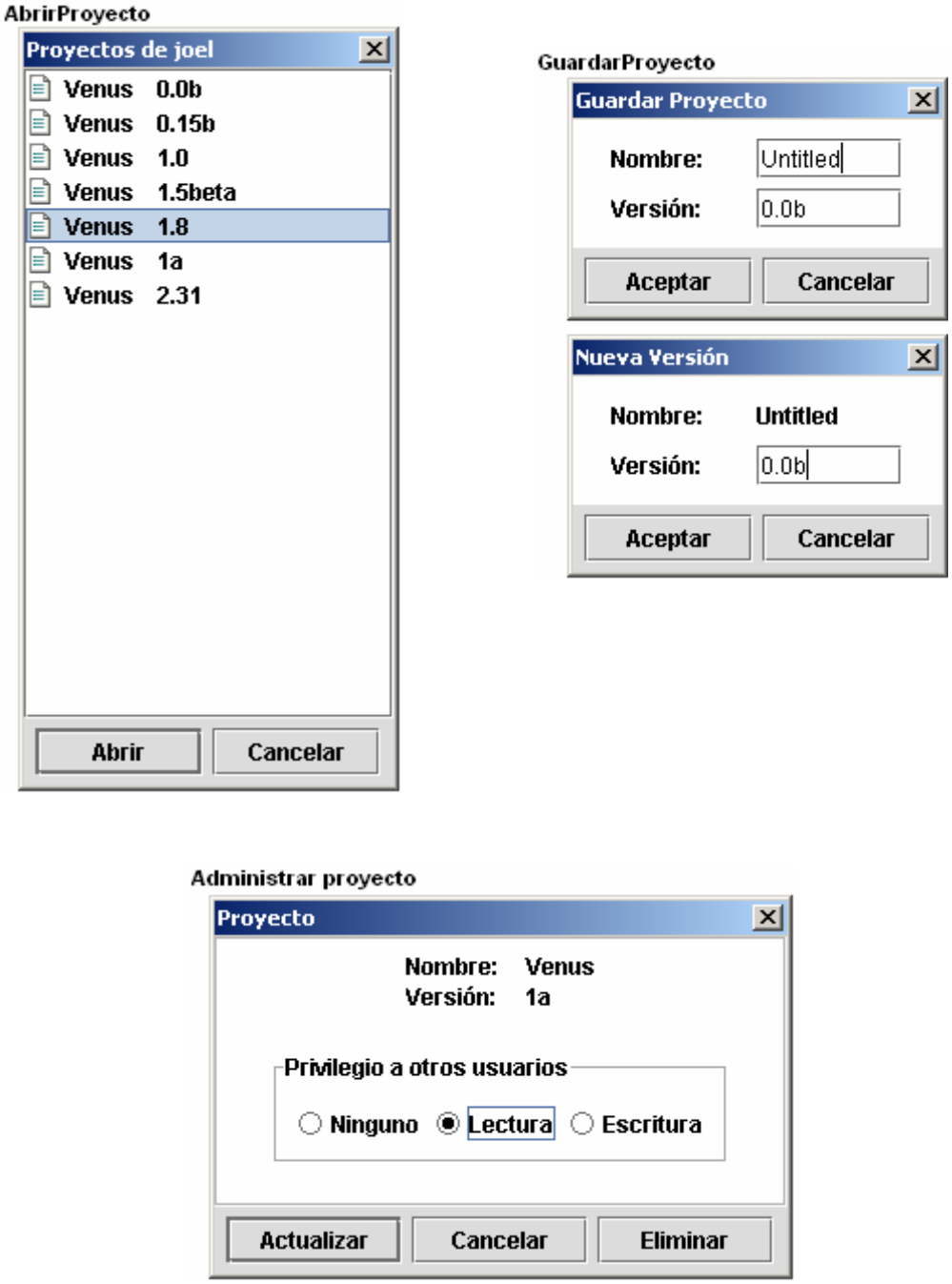

Fig. 5.22 Interfaces gráficas de las clases del EAPI (continuación)

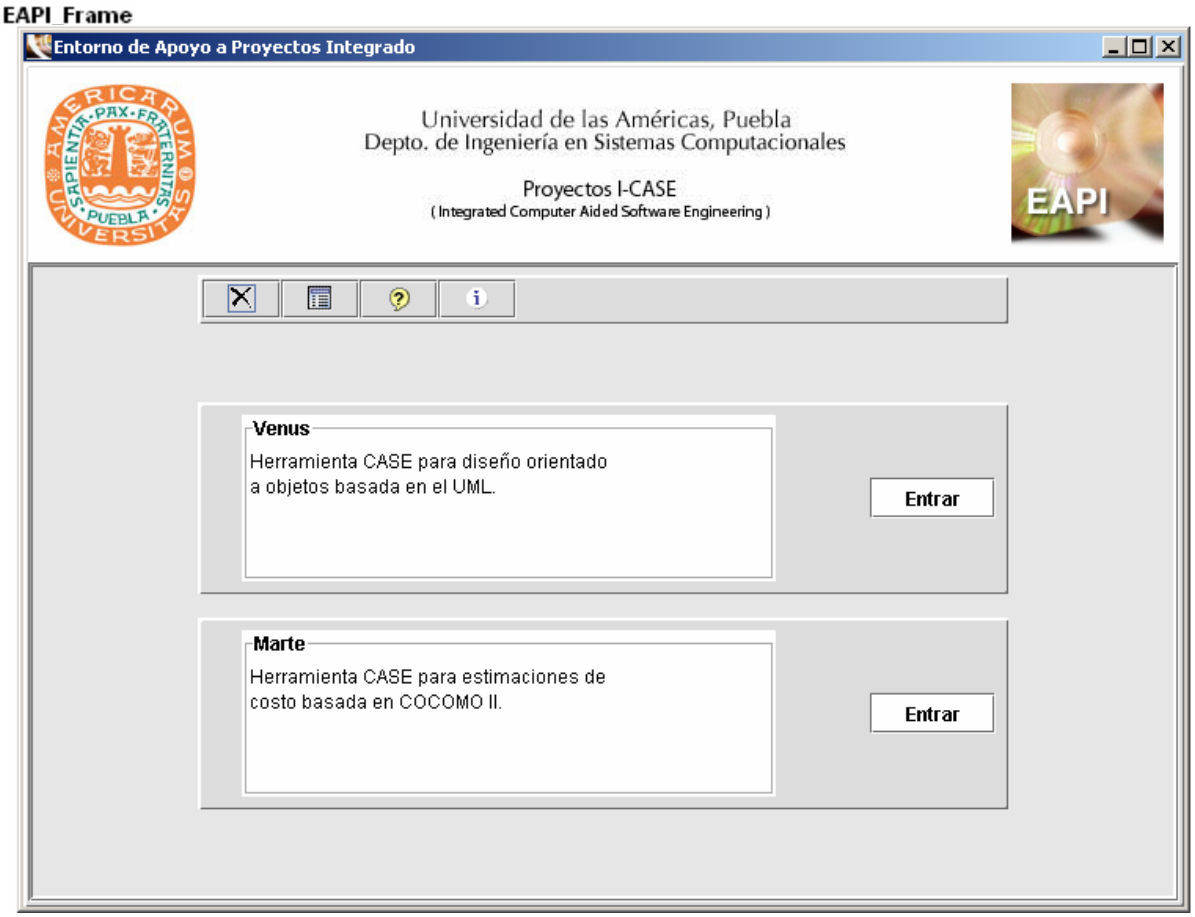

Fig. 5.22 Interfaces gráficas de las clases del EAPI (continuación)

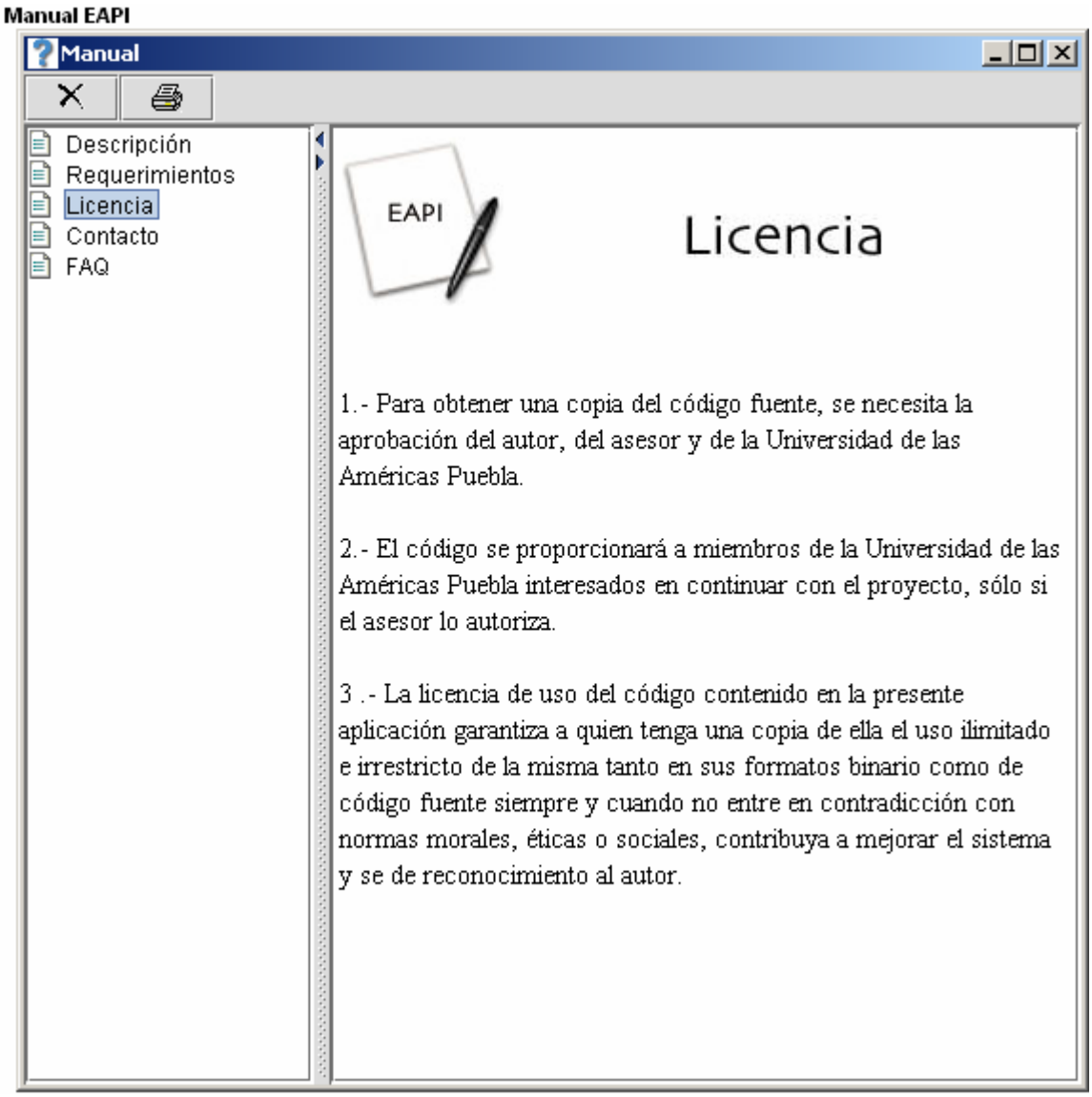

Fig. 5.22 Interfaces gráficas de las clases del EAPI (continuación)

5.3.2 Implementación de Venus

5.3.2.1 Descripción

Venus permite realizar diseño orientado a objetos utilizando la notación del UML

(*Unified Modeling Language*). Se limitó el alcance de la herramienta al soporte de diagramas de colaboración y de clase.

Para construir a Venus se utilizó el proceso unificado de desarrollo de software (*The Unified Software Development Process*), invirtiendo 5 meses en análisis-diseño, y 2 meses en codificación-pruebas. Venus ha sido la primera herramienta CASE construída en el Entorno de Apoyo a Proyectos Integrado (EAPI) que provee msqlroot.

5.3.2.2 Interacción Humano-Computadora

La interacción humano-computadora, o HCI (*Human-Computer Interaction*), se encarga de analizar y estudiar la relación que se presenta entre el ser humano con la computadora y sus sistemas, de tal forma que puedan diseñarse y desarrollarse las actividades conjuntas de forma productiva y segura.

La HCI de Venus está basada en el uso de menús, diálogos, y manipulación de diagramas. Al diseñar este tipo de comunicación es de suma importancia que las palabras utilizadas en los menús, el diseño de los íconos, y la manipulación de diagramas, sean completamente explícitos para los usuarios, lo que incrementó considerablemente la dificultad al implementar el sistema.

5.3.2.3 Menús e iconos

El diseño de iconos y menús, se basó en el sistema Microsoft Office (por ser

ampliamente usado), y en herramientas CASE existentes para editar/manipular diagramas UML (Rational, Argo, Fujaba y Magic Draw). Los menús e iconos de Venus se pueden apreciar en la figura 5.23.

Venus al igual que las herramientas comerciales que permiten crear diagramas UML, cuenta con un árbol gráfico en la parte izquierda, que permite dar un orden jerárquico a los diagramas creados por los usuarios. Ver figura 5.24.

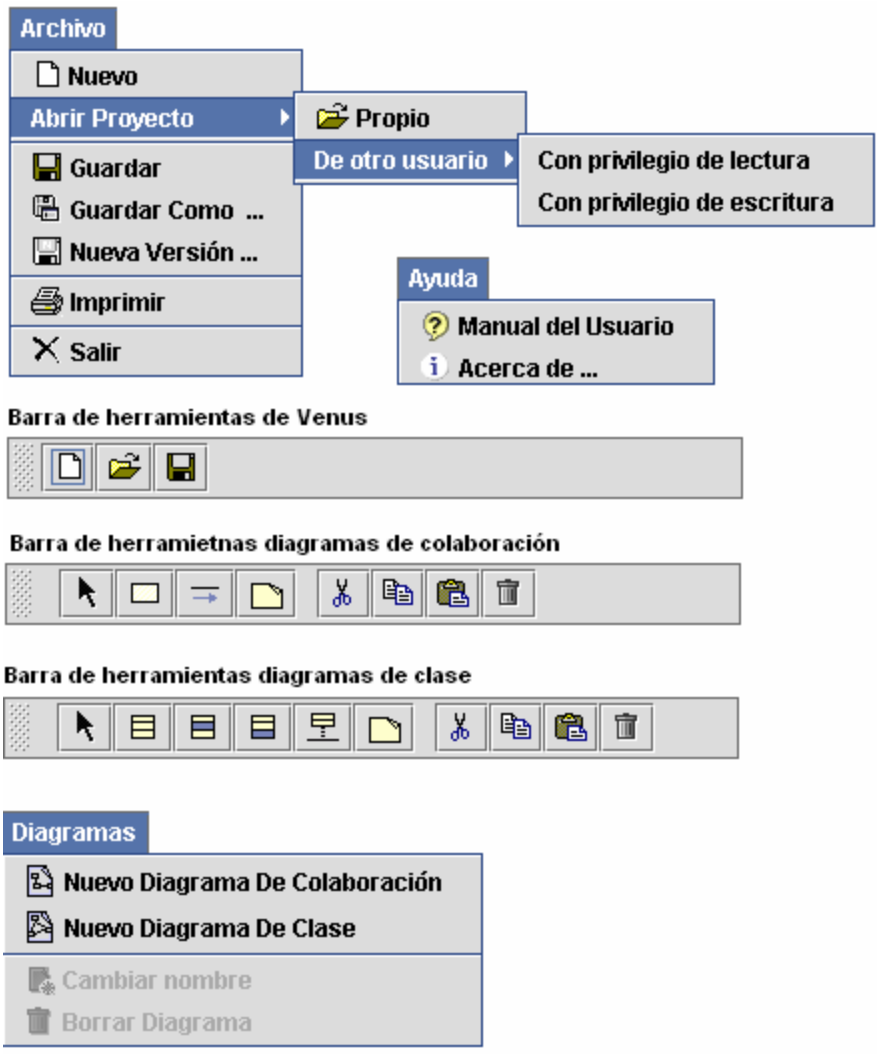

Fig. 5.23 Menús e iconos de Venus

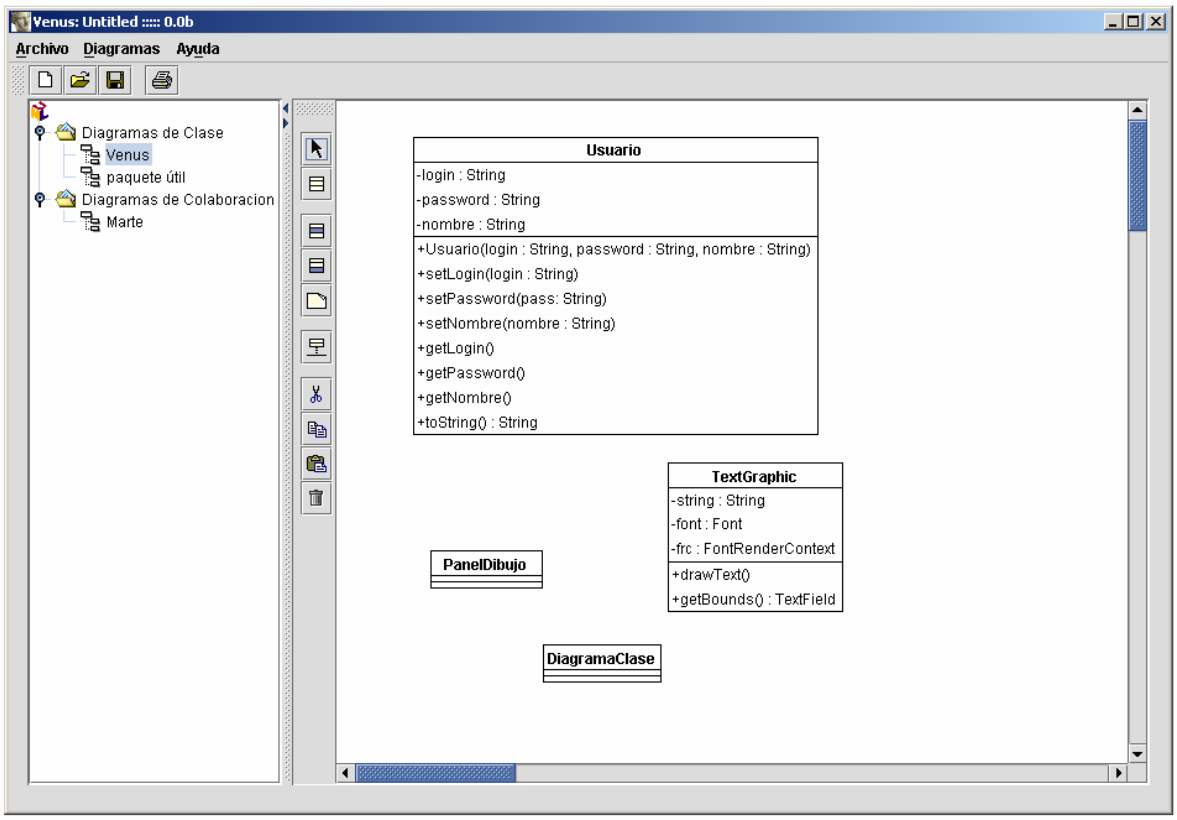

Fig. 5.24 Interfaz gráfica de Venus

5.3.2.4 Manipulación de diagramas

Para permitir que el usuario manipule cómodamente los diagramas se incluyeron en Venus características comunes de programas para crear/editar diagramas (no sólo de UML). Dichas características se ejemplifican en la figura 5.25.

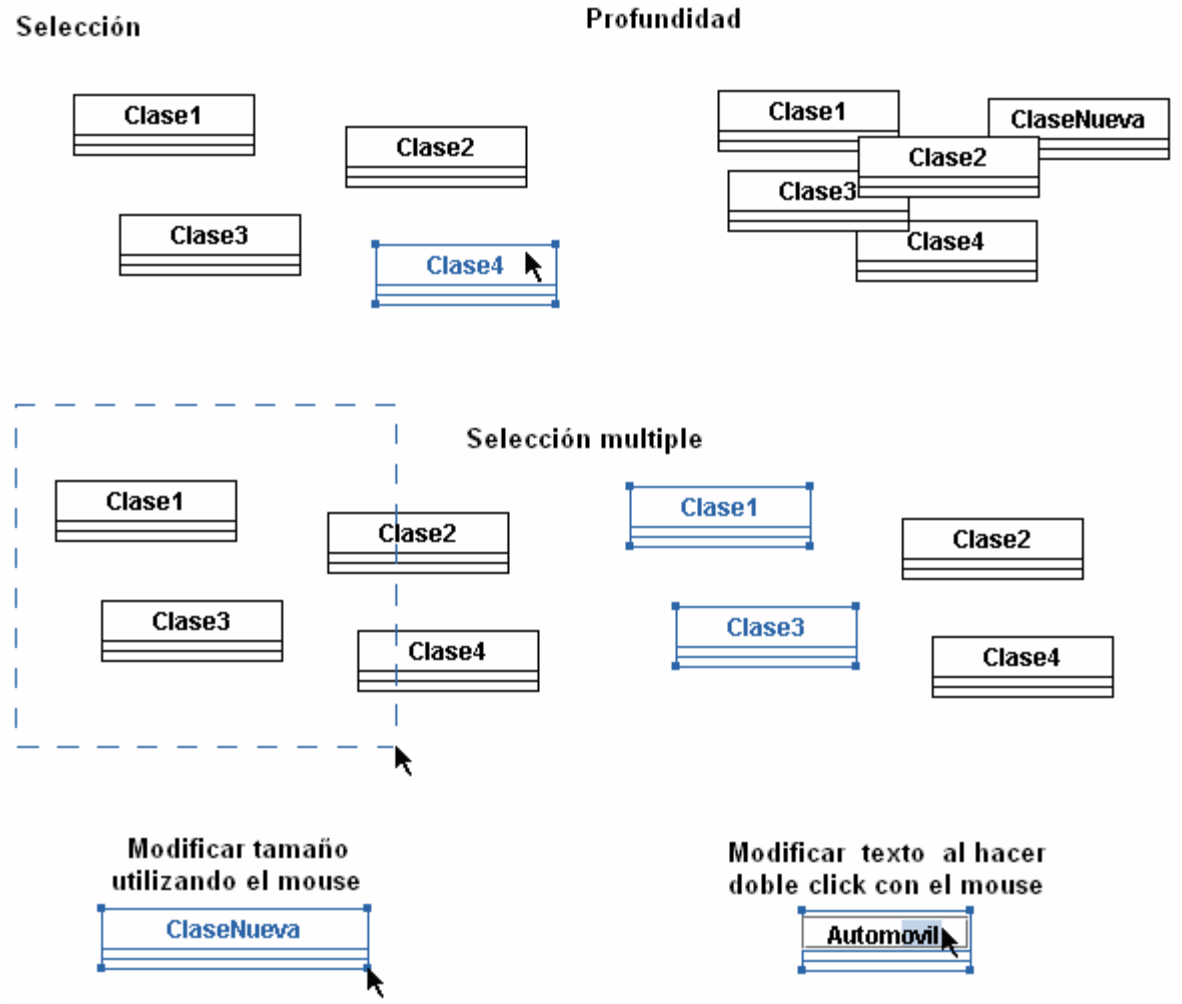

Fig. 5.25 Manipulación de diagramas en Venus

5.3.2.5 Manual del usuario

Un aspecto importante de todo sistema es proveer al usuario de un manual de fácil acceso, y que contenga la información necesaria para conocer las características del sistema así como su funcionamiento. El manual de Venus (ver Fig. 5.26) se implementó con la clase ManualVenus (extiende de la clase Manual que provee el EAPI).

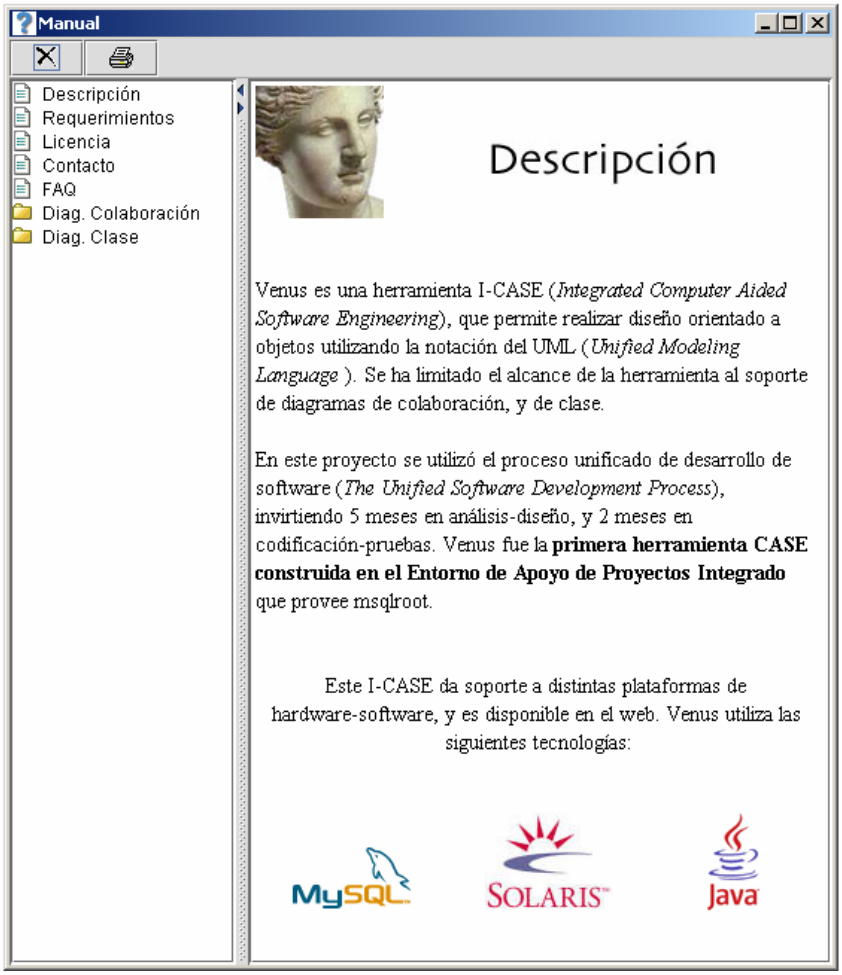

Fig. 5.26 Manual en línea de Venus

5.3.2.6 Impresión

Imprimir los diagramas generados por Venus y las páginas del manual, es muy sencillo, ya que se invocan a los métodos `mostrarDialogoImprimir()` y `print()` de la clase IPanel.

5.2.3 Clases desarrolladas

En total se desarrollaron doce clases para implementar el I-CASE Venus (la mayoría extiende o invoca clases del EAPI). Dichas clases se encuentran distribuidas en los paquetes “mx.udlap.msqlroot.venus” y “mx.udlap.msqlroot.venus.diagrama”.

El número tan pequeño de clases y el poco tiempo de codificación de Venus, se logró por el Ambiente Integrado que proporciona el EAPI. De esta forma se demuestra que el EAPI cumple con el objetivo de permitir el rápido desarrollo de herramientas CASE basadas en el lenguaje de programación Java.

La cantidad de datos involucrados al generar diagramas UML es muy grande, debido a esto se realizó un análisis detallado de cómo representar esa información en una base de datos relacional. Una manera sencilla (y práctica) de almacenar esa información es por medio de *bytes*, es decir, guardar los objetos Java que contienen los datos de los diagramas.

Toda la información que se genera en Venus forma parte de un JTree (internamente implementa un árbol), que contiene dos ramas (una para diagramas de colaboración, y otra para diagramas de clase) donde todas sus hojas son Vectores que contienen los datos. Tomando esto en cuenta, se generó un diagrama entidad-relación que considera como atributos de la entidad *proyecto*, la rama de diagramas de clase, y la rama de diagramas de colaboración, de dicho árbol (ver Fig. 5.21).

5.3.3 Página Web de “msqlroot”

A lo largo del desarrollo del proyecto se recabó una gran cantidad de información sobre herramientas CASE, UML, y tecnologías Java. Esa información esta disponible en la página de msqlroot (http://www.udlap.mx/~msqlroot, ver Fig. 5.27) para que pueda ser utilizada por cualquier persona interesada en estos temas (se solicitó a los buscadores Google, Yahoo y AltaVista que agreguen la página a su directorio).

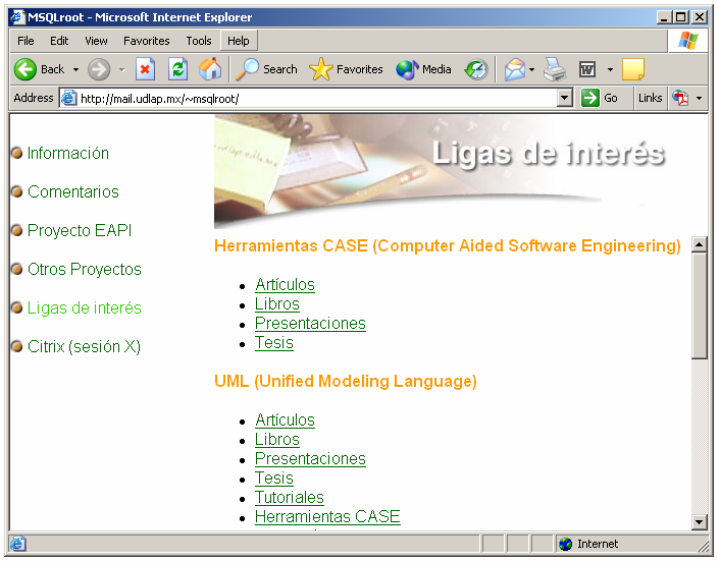

Fig. 5.27 Página Web de *msqlroot*

5.4 Conclusiones

Se demostró que es posible crear herramientas I-CASE a través de los componentes de software que proporciona el EAPI. Sin embargo, aún queda pendiente la verdadera integración de la información de ingeniería generada por las herramientas contenidas en el ambiente.