ANEXOS

TABLA DE CONTENIDO

CICLOS DE DESARROLLO **7**

A.1 ANALISIS Y DISEÑO DE LOS CASOS DE USO *7*
A.1.1 Caso de uso: Gestionar usuarios 7
A.1.1.1 Análisis 7
A.1.1.2 Diseño 9
A.1.2 Caso de uso general: Gestionar Modelos de Aprendizaje 12
A.1.2.1 Caso de uso: Gestionar Modelos de Aprendizaje 13
A.1.2.1.1 Análisis 13
A.1.2.1.2 Diseño 15
A.1.2.2 Caso de uso: Gestionar Estilos de Enseñanza-Aprendizaje 18
A.1.2.2.1 Análisis 18
A.1.2.2.2 Diseño 20
A.1.2.3 Caso de uso: Gestionar Cuestionarios 23
A.1.2.3.1 Análisis 23
A.1.2.3.2 Diseño 24
A.1.2.4 Caso de uso: Gestionar Preguntas y Respuestas 27
A.1.2.4.1 Análisis 27
A.1.2.4.2 Diseño 29
A.1.3 Caso de uso: Gestionar Asignaturas 33
A.1.3.1 Análisis 34
A.1.3.2 Diseño 35
A.1.4 Caso de uso: Gestionar Cursos 37
A.1.4.1 Análisis 37
A.1.4.2 Diseño 39
A.1.5 Caso de uso: Cambiar Clave 41
A.1.5.1 Análisis 41
A.1.5.2 Diseño 42
A.1.6 Caso de uso: Seleccionar tipos de actividad a monitorear 45
A.1.6.1 Análisis 45
A.1.6.2 Diseño 46
A.1.7 Caso de uso: Seleccionar estrategias de aprendizaje 49
A.1.7.1 Análisis 49
A.1.7.2 Diseño 51
A.1.8 Caso de uso: Modificar Perfil 55
A.1.8.1 Análisis 55
A.1.8.2 Diseño 57
A.1.9 Modelos Lógicos y Físicos 58
A.1.9.1 Modelo Entidad-Relación 59
A.1.9.2 Modelo Físico 61

BASES CONCEPTUALES **63**

B.1. E-LEARNING *63*

B.1.1 Diversas definiciones de E-Learning 63

B.2 SHARABLE CONTENT OBJECT REFERENCE MODEL - SCORM 64
B.2.1 Organización de SCORM 64
B.2.1.1 Libro: Vistazo General de SCORM 2004 64
B.2.1.2 Libro: Modelo de Agregación de Contenidos (CAM) de SCORM 64
B.2.1.3 Libro: Ambiente de Ejecución (RTE) de SCORM 65
B.2.1.4 Libro: Secuenciación y Navegación (SN) de SCORM 66
B.2.1.5 Resumen de los libros SCORM 67
B.2.2 Objetos de Contenido Compartible - SCO 68
B.2.2.1 Etiquetado y empaquetado de objetos de aprendizaje 68
B.2.2.2 Normativas y estándares para el tratamiento de contenidos 68
B.2.2.2 El lenguaje universal - XML 69
B.2.2.2 Cómo distribuir contenidos: Empaquetado e Intercambio de Contenidos 70
B.2.2.3 Cómo "hablar" con la plataforma / entorno de tiempo de ejecución 71
B.2.2.4 Cómo secuenciar contenidos: Secuenciación (IMS,SCORM) 71
B.2.3 Principales organizaciones involucradas en el proceso de estandarización. 72
B.2.3.1 LTSC (Learning Technology Standardization Committee) 72
B.2.3.2 IMS 72
B.2.3.3 ADL (Advanced Distributed Learning)/SCORM (Sharable Content Object Reference Model) 72
B.2.3.4 AICC 73
B.2.3.5 OKI (Open Knowledge Iniciative) 73
B.2.3.6 Iniciativas Europeas 74

B.2.3 AGENTES INTELIGENTES 75
B.2.3.1 ¿Qué es un Agente? 75
B.2.3.1 Arquitectura de Agentes 77
B.2.3.1.1 Arquitecturas para Agentes Deliberativos 77
B.2.3.1.2 Arquitecturas para Agentes Reactivos 79
B.2.3.1.3 Arquitectura de Red de Agentes 80
B.2.3.1.2.4 Arquitecturas para Agentes Híbridos 80

B.4 PATRONES DE DISEÑO (tomado de [PAT]) 83
B.4.1 Algo de Historia 83
B.4.2 ¿Qué es un Patrón de Diseño? 84
B.4.3 Describiendo un Patrón 84
B.4.4 Clasificación de Patrones 85
B.4.4.1 Patrones Creacionales 85
B.4.4.2 Patrones Estructurales 85
B.4.4.3 Patrones de Conducta 86
B.4.5 Aplicación de los patrones 86
B.4.5.1 Patrón Singleton 86
B.4.5.2 Patrón Observador 87
B.4.6 Otros Tipos de Patrones 88
B.4.6.1 Patrones de Arquitectura 89
B.4.6.2 Patrones de Implementación 89

BIBLIOGRAFIA 90

LISTA DE FIGURAS

Figura 1. Screen Shot del caso de uso Gestionar usuarios 9
Figura 2. Screen Shot del caso de uso Gestionar Modelos de Aprendizaje 15
Figura 3. Screen Shot del Caso de uso Gestionar Estilos de Aprendizaje 20
Figura 4. Screen Shot del Caso de uso Gestionar Cuestionarios 24
Figura 5. Screen Shot del Caso de uso Gestionar Preguntas y Respuestas 29
Figura 6. Screen Shot del caso de uso Gestionar Asignaturas 35
Figura 7. Screen Shot del caso de uso Gestionar Cursos 39
Figura 8. Screen Shot del caso de uso Cambiar Clave 42
Figura 9. Screen Shot del caso de uso seleccionar tipos de actividad a monitorear 46
Figura 10. Screen Shot del caso de uso Seleccionar estrategias de aprendizaje 51
Figura 11. Screen Shot del caso de uso Modificar Perfil 57
*Figura 12. Organización de SCORM (tomado de [**ADLOV**]) 64*
*Figura 13. Paquete de Contenidos (tomado de [**ADLOV**]) 65*
*Figura 14. Árbol de actividades (tomado de [**ADLOV**]) 67*
Figura 15. Interacción agente - ambiente 76
Figura 16. Características de los agentes 77
Figura 17. Diagrama UML del patrón Singleton 87
Figura 18. Diagrama UML del patrón Observador 88
Figura 19. Diagrama UML del patrón de Capas 89

LISTA DE TABLAS

Tabla A.1.1.1 Diagrama del Caso de Uso: Gestionar Usuarios 7
Tabla A.1.1.2 Análisis del Caso de uso: Gestionar Usuarios 8
Tabla D.1.1.1. Diseño del caso de uso Gestionar Usuarios 12
Tabla A.1.1.3 Diagrama General del Caso de uso Gestionar Modelos de Aprendizaje 13
Tabla A.1.1.4 Análisis del Caso de uso Gestionar modelos de aprendizaje 14
Tabla D.1.1.2. Diseño del caso de uso Gestionar Modelos 18
Tabla A.1.1.5 Análisis del Caso de uso Gestionar Estilos de Aprendizaje 19
Tabla D.1.1.3. Diseño del caso de uso Gestionar Estilos 22
Tabla A.1.1.6 Caso de uso Gestionar Cuestionarios 24
Tabla D.1.1.4. Diseño del caso de uso Gestionar Cuestionarios 27
Tabla A.1.1.7 Caso de uso Gestionar Preguntas y Respuestas 29
Tabla D.1.1.5. Diseño del caso de uso Gestionar Preguntas y Respuestas 33
Tabla A.1.1.8 Análisis del caso de uso Gestionar Asignaturas 35
Tabla D.1.1.6. Diseño del caso de uso Gestionar Asignaturas 37
Tabla A.1.1.9 Análisis del caso de uso Gestionar Cursos 38
Tabla D.1.1.7. Diseño del caso de uso Gestionar Cursos 41
Tabla A.1.1.10 Análisis del caso de uso Cambiar Clave 42
Tabla D.1.1.8. Diseño del caso de uso Cambiar Clave 44
Tabla A.1.1.11 Análisis del caso de uso seleccionar tipos de actividad a monitorear 46
Tabla D.1.1.9. Diseño del caso de uso seleccionar Tipos de Actividad a monitorear 48
Tabla A.1.1.12 Análisis del caso de uso Seleccionar estrategias de aprendizaje 51
Tabla D.1.1.10. Diseño del caso de uso Seleccionar estrategias de aprendizaje 55
Tabla A.1.1.13 Análisis del caso de uso Modificar Perfil 56
Tabla D.1.1.11. Diseño del caso de uso Modificar Perfil 58
Tabla A.1.1.14 Modelo Entidad-Relación 60
Tabla A.1.1.15 Modelo Físico 61
*Tabla B.1. Resumen de los libros SCORM (tomado de[**ADLOV**]) 67*

ANEXOS A

CICLOS DE DESARROLLO DE LA APLICACION

CICLOS DE DESARROLLO

A.1 ANALISIS Y DISEÑO DE LOS CASOS DE USO

A continuación se presenta el análisis y diseño de los casos de uso, que también intervienen en el ciclo de desarrollo del módulo del Sistema Tutor Inteligente (STI) apoyado en agentes inteligentes.

A.1.1 Caso de uso: Gestionar usuarios

A.1.1.1 Análisis

El diagrama de caso de uso planteado a continuación, presenta la gestión de toda la información asociada a los usuarios. Este caso de uso invoca al caso de uso Entrar al sistema.

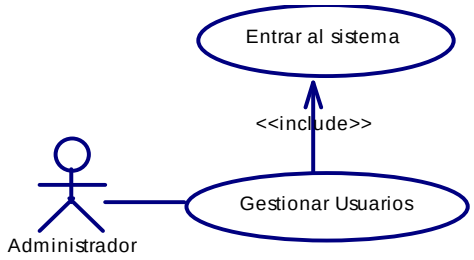
DIAGRAMA DEL CASO DE USO: GESTIONAR USUARIOS
En este diagrama se encuentra el actor Administrador, quien es el encargado de gestionar los usuarios asociados al proceso de aprendizaje en línea.


Tabla A.1.1.1 Diagrama del Caso de Uso: Gestionar Usuarios

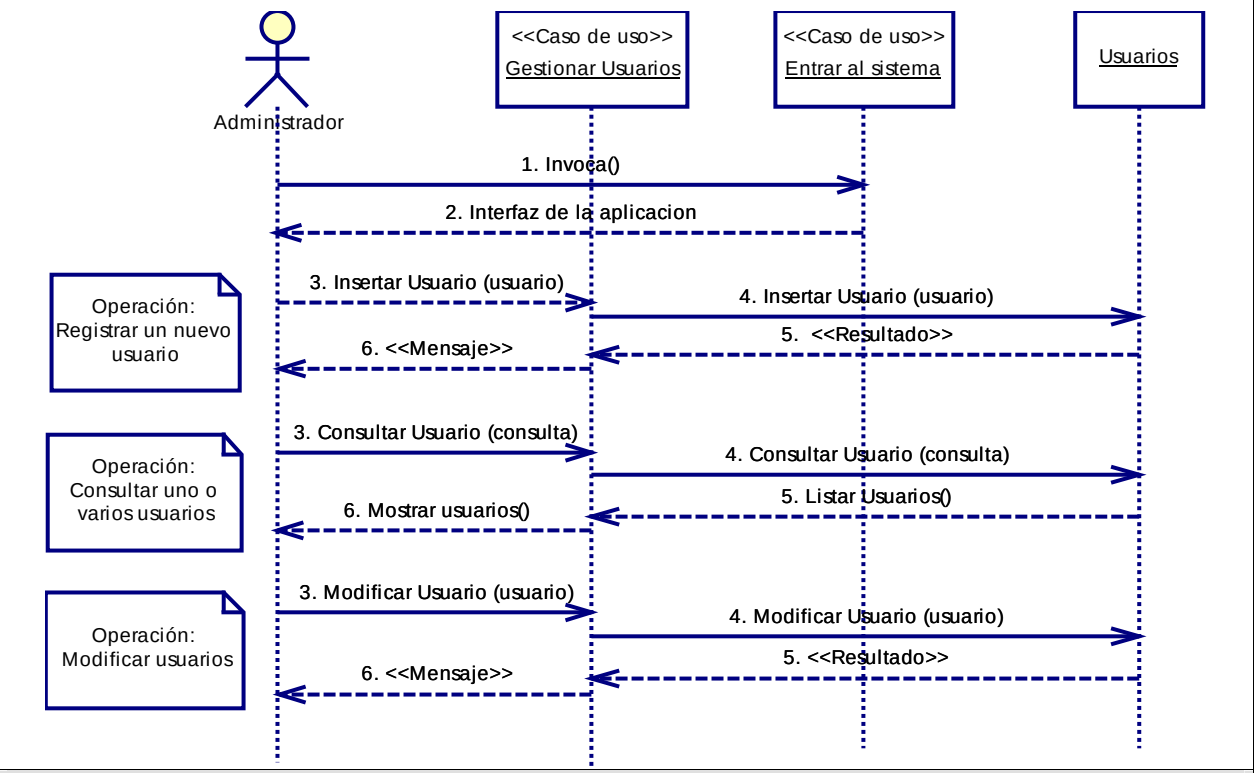
A continuación, se analizan cada una de las operaciones que conforman el caso de uso: Gestionar usuarios, mostrado en el diagrama anterior.

FORMATO CASO DE USO GESTIONAR USUARIOS	
Actores	
• Administrador (Iniciador)	
Este caso de uso comienza cuando el administrador invoca al caso de uso Entrar al sistema, con el fin de gestionar (registrar, consultar o modificar) usuarios. El sistema solicita la información relacionada con el usuario y realiza el respectivo proceso dependiendo de la operación que desee el administrador (registrar, consultar o modificar), luego envía un mensaje al administrador del resultado de la operación efectuada. Este caso de uso finaliza cuando el administrador: abandona el proceso, cierra la sesión o cierra la aplicación.	
Acción del Actor	**Respuesta del Sistema**
1. El administrador decide gestionar (registrar, consultar o modificar) uno o varios usuarios.	**2**. El sistema redirecciona al administrador a la interfaz de conexión.

FORMATO CASO DE USO GESTIONAR USUARIOS	
3. El administrador utiliza el caso de uso **Entrar al sistema**.	4. Si Entrar al sistema es exitoso, el sistema presenta la interfaz principal de la aplicación.
5. El administrador ingresa la información del usuario dependiendo de la operación de gestión (registrar, consultar o modificar) seleccionada.	**6**. El sistema realiza el respectivo proceso dependiendo de la operación que seleccionó el administrador, y le informa el resultado de la misma.

DIAGRAMA DE SECUENCIA DEL SISTEMA DEL CASO DE USO GESTIONAR USUARIOS

Este diagrama muestra todo el proceso necesario que debe hacer el actor administrador para realizar el caso de uso Gestionar Usuarios. El administrador gestiona (registra, consulta o modifica) usuarios. El sistema le envía un mensaje de información acerca del resultado de acuerdo a la operación efectuada.



MODELO CONCEPTUAL DEL CASO DE USO GESTIONAR USUARIOS

En el modelo conceptual, se destacan los conceptos más importantes del caso de uso Gestionar Usuarios. El objeto persistente o entidad que usa el caso de uso es: Usuarios. El control de este caso de uso está representado por el objeto Gestionar Usuarios, y la interfaz para la interacción con el actor Administrador es el objeto del mismo nombre.
El control realiza en la entidad Usuarios el proceso de gestión dependiendo de la operación seleccionada por el administrador.

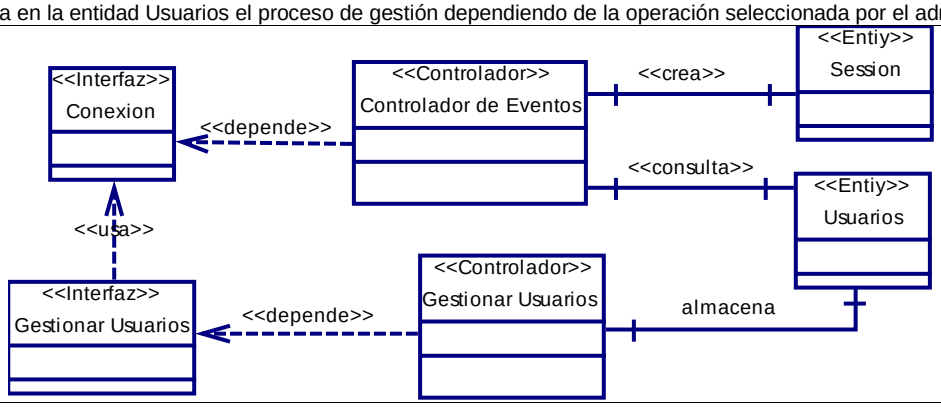


Tabla A.1.1.2 Análisis del Caso de uso: Gestionar Usuarios

A.1.1.2 Diseño

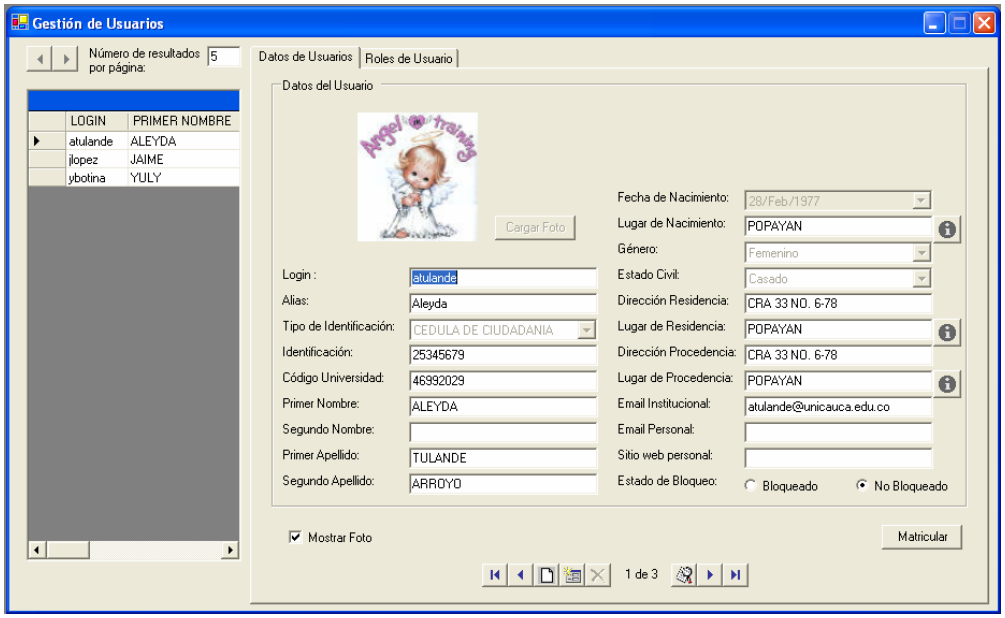
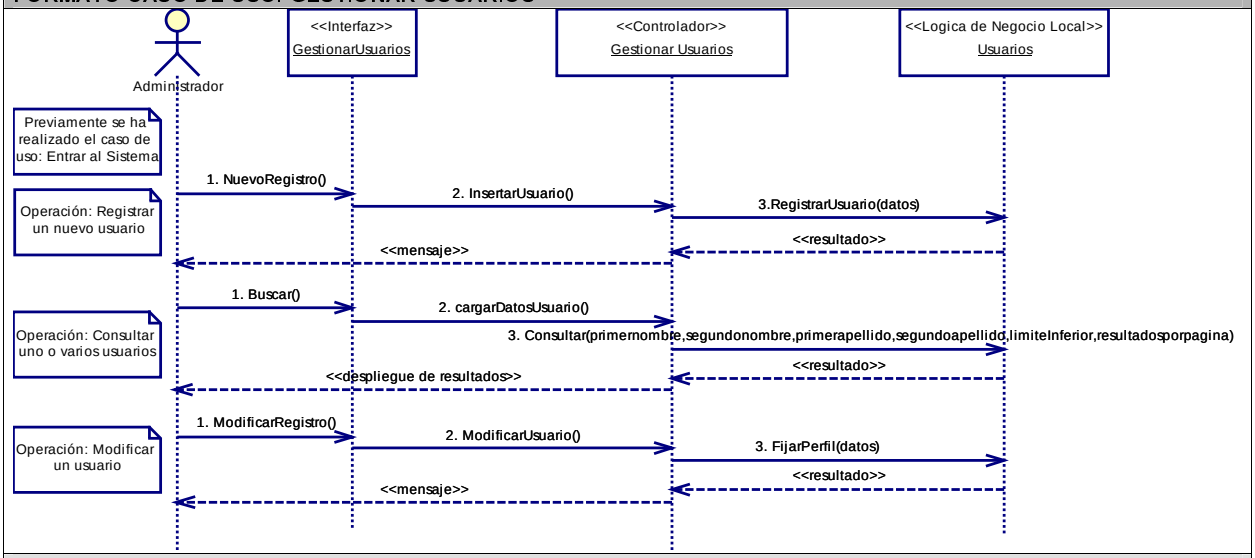


Figura 1. Screen Shot del caso de uso Gestionar usuarios

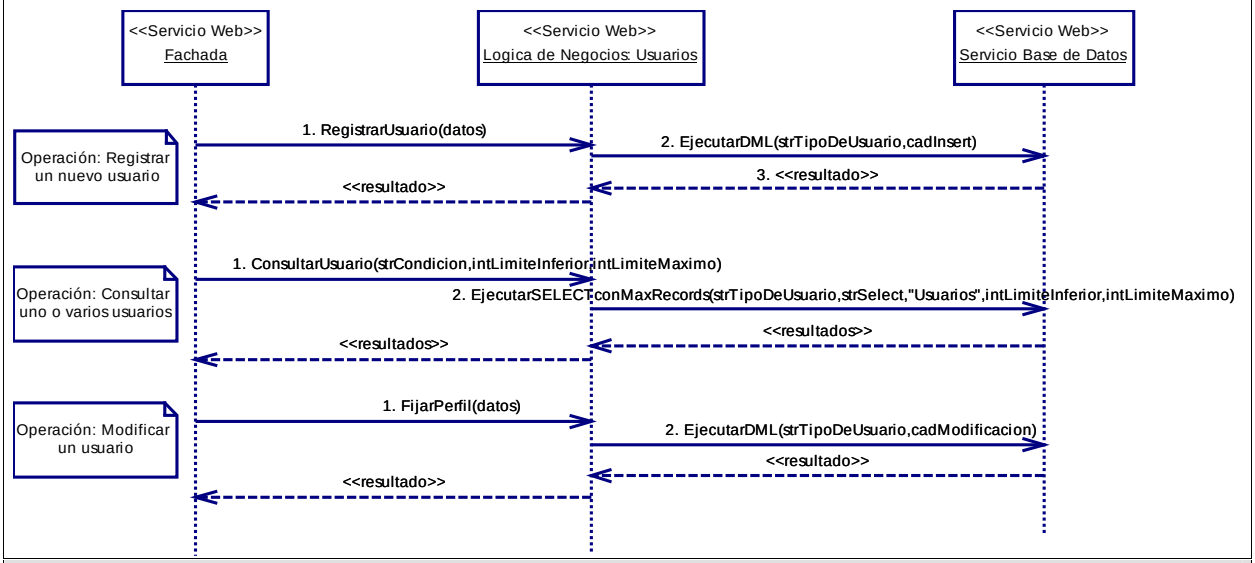
FORMATO CASO DE USO: GESTIONAR USUARIOS	
Actores	
Administrador (Iniciador)	
Este caso de uso comienza cuando el administrador decide Gestionar Usuarios. El sistema solicita primero al administrador que se loguee, invocando el caso de uso Entrar al Sistema. Después de este proceso el sistema presenta al administrador la interfaz principal, donde el administrador elige la opción "Gestionar Usuarios", que presenta la interfaz que gestiona los usuarios a través de tres operaciones básicas (registrar, consultar y modificar). Dependiendo de la operación que realice el administrador, el controlador envía los datos del usuario a la clase Usuarios de Lógica de Negocios Local, la cual se comunica y propaga los datos a través de la referencia Proxy del servicio Web Fachada, quien se comunica con el servicio Web Usuarios de la lógica de Negocios, el cual gestiona el usuario en la base de datos, ayudado del servicio Web Lógica de Servicios. Finalmente el sistema, envía un mensaje al administrador sobre el éxito de la operación. Este caso de uso finaliza cuando el administrador: abandona el proceso, cierra la sesión o cierra la aplicación.	
Acción del Actor	**Respuesta del Sistema**
1. El administrador decide gestionar usuarios.	2. El sistema redirecciona al administrador a la interfaz de Conexión.
3. El administrador utiliza el caso de uso Entrar al Sistema.	4. Si Entrar al sistema es exitoso, el sistema presenta la interfaz principal de la aplicación.
5. El administrador hace clic en alguno de los botones (adicionar, consultar o modificar) que permiten la gestión de los usuarios.	6. El sistema envía los datos del usuario y ejecuta el proceso de gestión de acuerdo a la operación efectuada, también retorna un mensaje del resultado de dicha operación.
CAPA DE LOGICA DE PRESENTACION: DIAGRAMA DE SECUENCIA DEL CASO DE USO GESTIONAR USUARIOS	
El administrador hace clic en alguno de los botones (adicionar, consultar o modificar) de la interfaz Gestionar Usuarios. Una vez que el administrador realiza la operación de su elección para gestionar los usuarios, esta información la envía el controlador a la clase Usuarios, a través de los métodos definidos para cada operación. Esta clase también está encargada de comunicarse y esperar respuesta del Servicio Web. Según el resultado, se despliega un mensaje al administrador.	

FORMATO CASO DE USO: GESTIONAR USUARIOS



CAPA DE LOGICA DEL NEGOCIO: DIAGRAMA DE SECUENCIA DEL CASO DE USO GESTIONAR USUARIOS

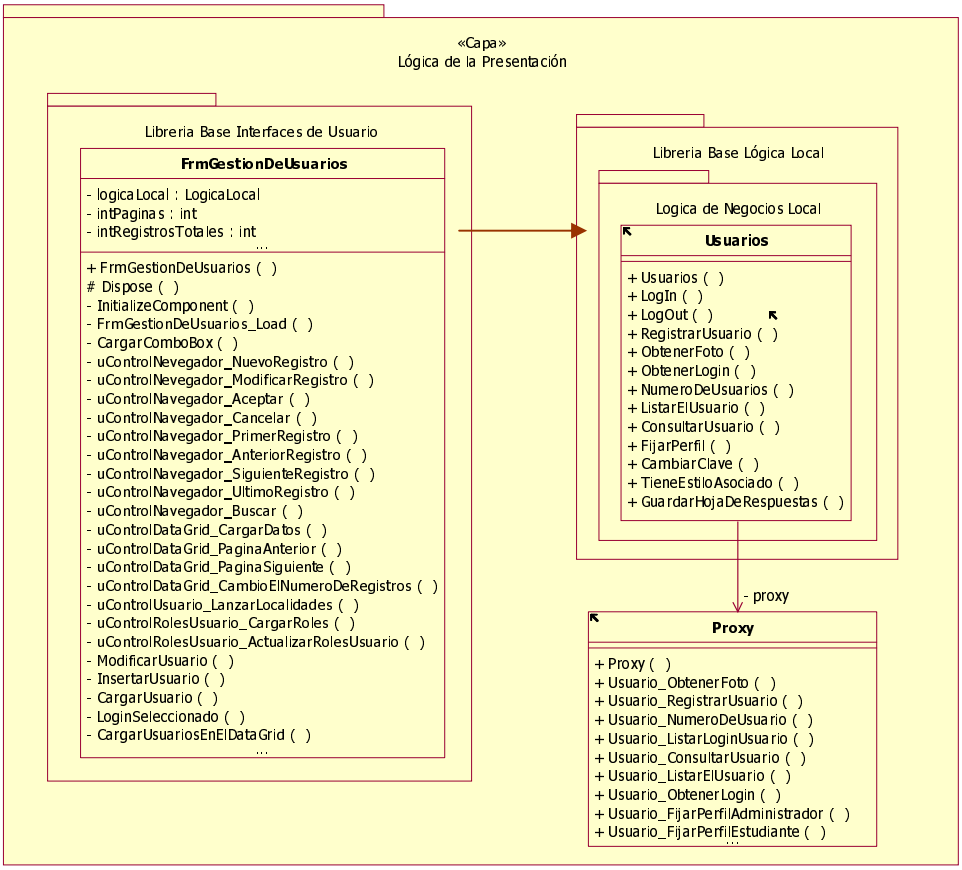
El servicio Web Fachada, recibe desde la capa de presentación, la información del usuario, este servicio Web delega al servicio Web Usuarios el proceso de gestión del usuario de acuerdo a la operación efectuada por el administrador, utilizando el patrón experto. El servicio Web Usuarios procesa los datos e invoca dependiendo de la operación de gestión efectuada al método respectivo de la clase de lógica Servicios base de datos, que se encarga de ejecutar el proceso de gestión contra la base de datos. Luego se informa del éxito o fracaso de dicho proceso a través del servicio Web hasta llegar a la capa de presentación.



CAPA DE LOGICA DE PRESENTACION: DIAGRAMA DE CLASES DEL CASO DE USO GESTIONAR USUARIOS

La interfaz Gestionar Usuarios se relaciona con el controlador a través de su código subyacente, que se relaciona con la lógica de negocios local a través de la clase Usuarios, que se relaciona con el Proxy de la Fachada, dado que Gestionar Usuarios utiliza métodos del Servicio Web de la aplicación.

FORMATO CASO DE USO: GESTIONAR USUARIOS



CAPA DE LOGICA DEL NEGOCIO Y SERVICIO: DIAGRAMA DE CLASES DEL CASO DE USO GESTIONAR USUARIOS

Fachada instancia un objeto de Usuarios para delegarle el proceso de gestión, existiendo una asociación directa entre ellas. El objeto Usuarios utiliza los métodos adecuados de la clase de Servicios de la base de datos de acuerdo a la operación de gestión, por lo que existe una asociación directa entre estas.

FORMATO CASO DE USO: GESTIONAR USUARIOS

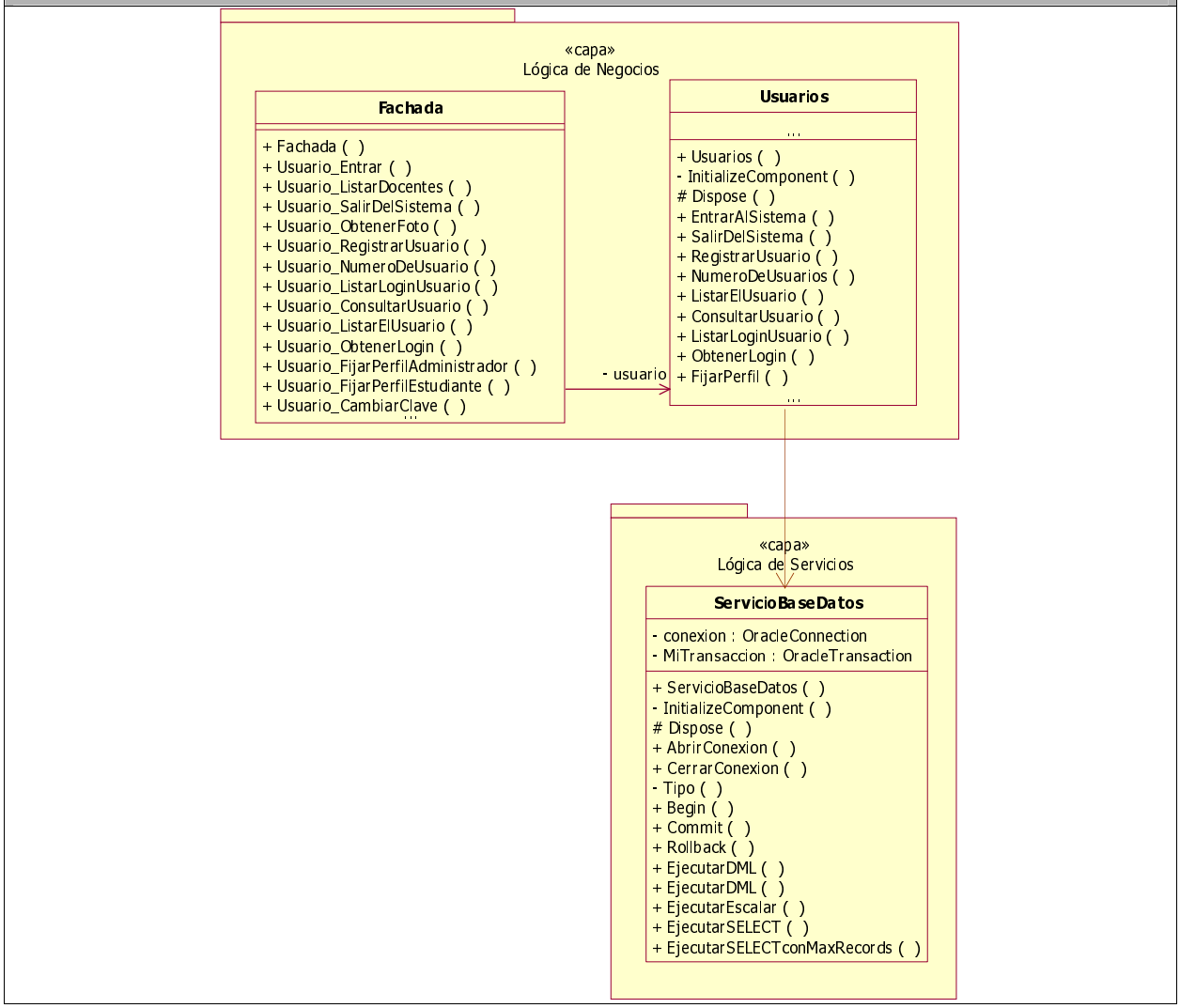


Tabla D.1.1.1. Diseño del caso de uso Gestionar Usuarios

A.1.2 Caso de uso general: Gestionar Modelos de Aprendizaje

El diagrama de caso de uso general planteado a continuación (Tabla A.1.1.3), presenta la gestión de toda la información asociada a los modelos de aprendizaje, como lo es: sus estilos, los cuestionarios con sus preguntas y respuestas elaborados para cada modelo. Este caso de uso invoca al caso de uso Entrar al sistema.

DIAGRAMA DEL CASO DE USO: GESTIONAR MODELOS DE APRENDIZAJE
En este diagrama se encuentra el actor Administrador, quien es el encargado de gestionar los modelos y estilos de aprendizaje, las preguntas y respuestas del o los cuestionarios asociados a los modelos.

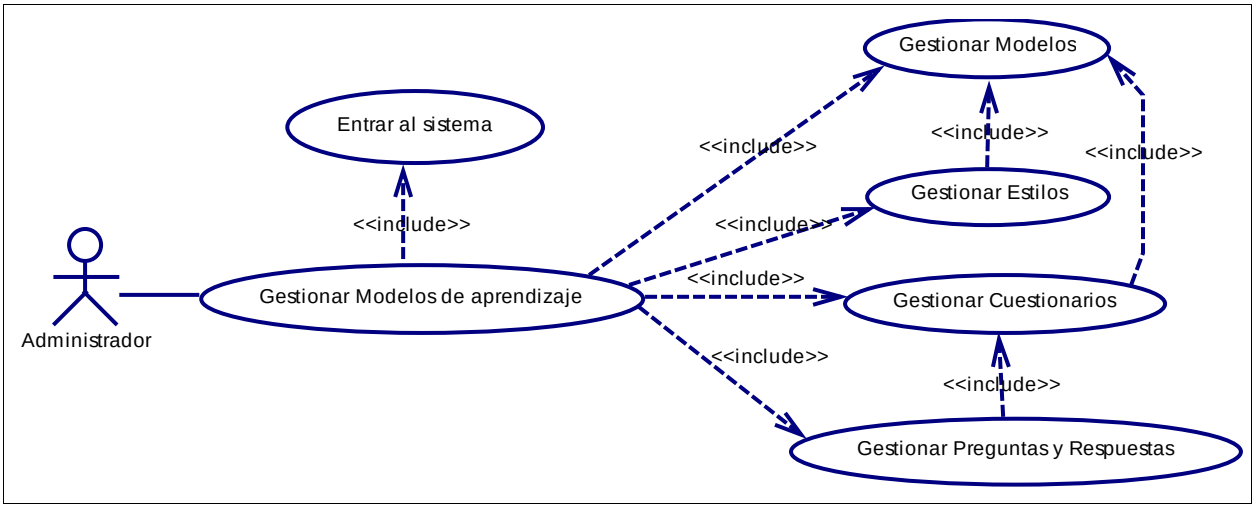


Tabla A.1.1.3 Diagrama General del Caso de uso Gestionar Modelos de Aprendizaje

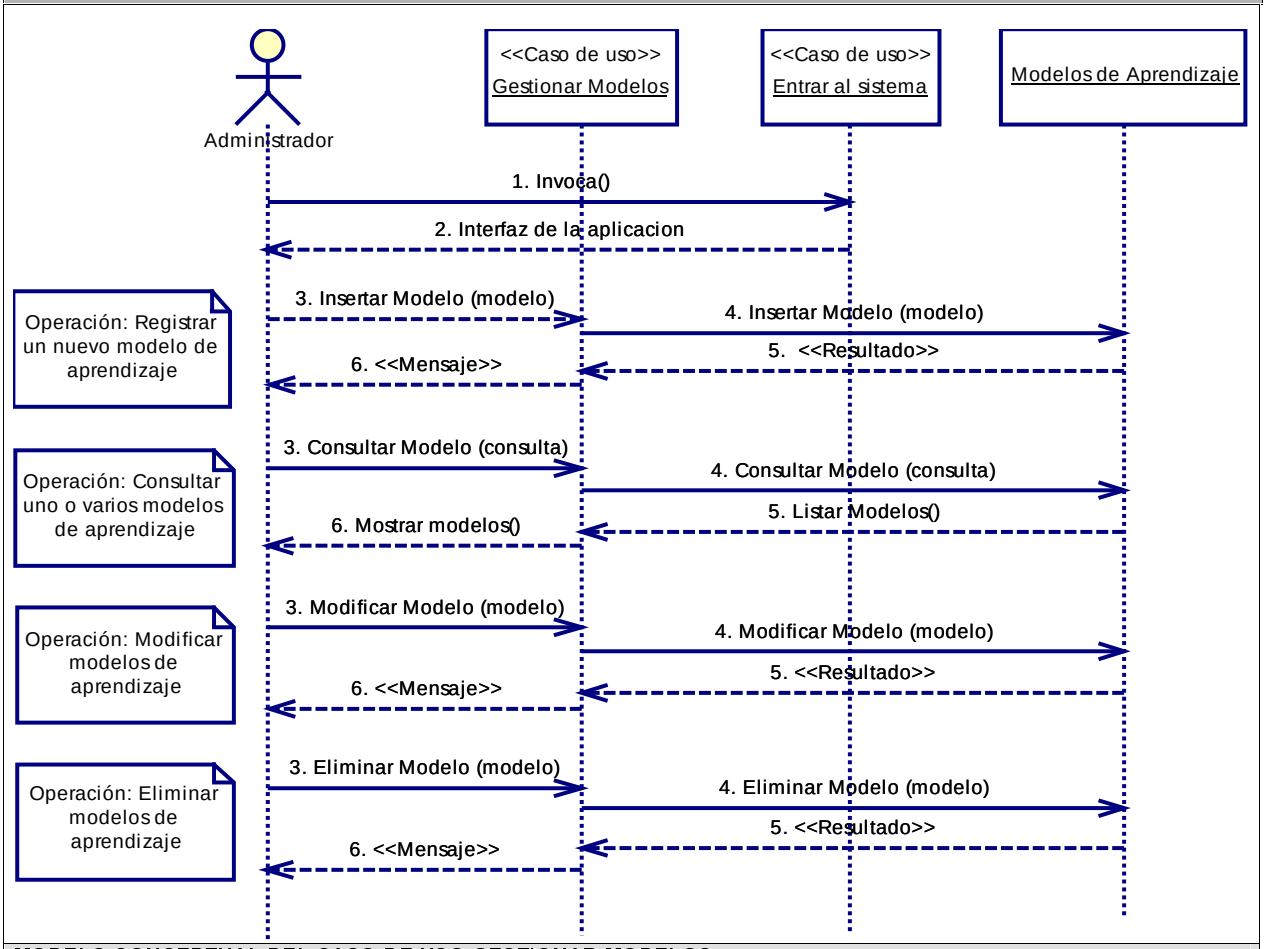
El análisis de cada uno de los casos de uso que conforman el caso de uso general se detalla a continuación.

A.1.2.1 Caso de uso: Gestionar Modelos de Aprendizaje

A.1.2.1.1 Análisis

FORMATO CASO DE USO GESTIONAR MODELOS	
Actores	
• Administrador (Iniciador)	
Este caso de uso comienza cuando el administrador invoca al caso de uso Entrar al sistema, con el fin de gestionar (registrar, consultar, modificar o eliminar) un modelo de aprendizaje que se utilizará en el proceso educativo. El sistema solicita la información relacionada con el modelo y realiza el respectivo proceso dependiendo de la operación que desee el administrador (registrar, consultar, modificar o eliminar), luego envía un mensaje al administrador del resultado de la operación efectuada. Este caso de uso finaliza cuando el administrador: abandona el proceso, cierra la sesión o cierra la aplicación.	
Acción del Actor	**Respuesta del Sistema**
1. El administrador decide gestionar (registrar, consultar, modificar o eliminar) un modelo de aprendizaje	**2**. El sistema redirecciona al administrador a la interfaz de conexión.
3. El administrador utiliza el caso de uso **Entrar al sistema**.	4. Si Entrar al sistema es exitoso, el sistema presenta la interfaz principal de la aplicación.
5. El administrador ingresa la información del modelo de aprendizaje dependiendo de la operación de gestión (registrar, consultar, modificar o eliminar) seleccionada.	**6**. El sistema realiza el respectivo proceso dependiendo de la operación que seleccionó el administrador, y le informa el resultado de la misma.
DIAGRAMA DE SECUENCIA DEL SISTEMA DEL CASO DE USO GESTIONAR MODELOS	
Este diagrama muestra todo el proceso necesario que debe hacer el actor administrador para realizar el caso de uso Gestionar Modelo. El administrador gestiona (registra, consulta, modifica o elimina) un modelo, luego de haber realizado los casos de uso necesarios. El sistema le envía un mensaje de información del resultado de acuerdo a la operación efectuada.	

FORMATO CASO DE USO GESTIONAR MODELOS



MODELO CONCEPTUAL DEL CASO DE USO GESTIONAR MODELOS

En el modelo conceptual, se destacan los conceptos más importantes del caso de uso Gestionar Modelos. El objeto persistente o entidad que usa el caso de uso es: Modelos de Aprendizaje. El control de este caso de uso está representado por el objeto Gestionar Modelos de Aprendizaje, y la interfaz para la interacción con el actor Administrador es el objeto del mismo nombre.
El control realiza en la entidad Modelos de aprendizaje el proceso de gestión dependiendo de la operación seleccionada por el administrador.

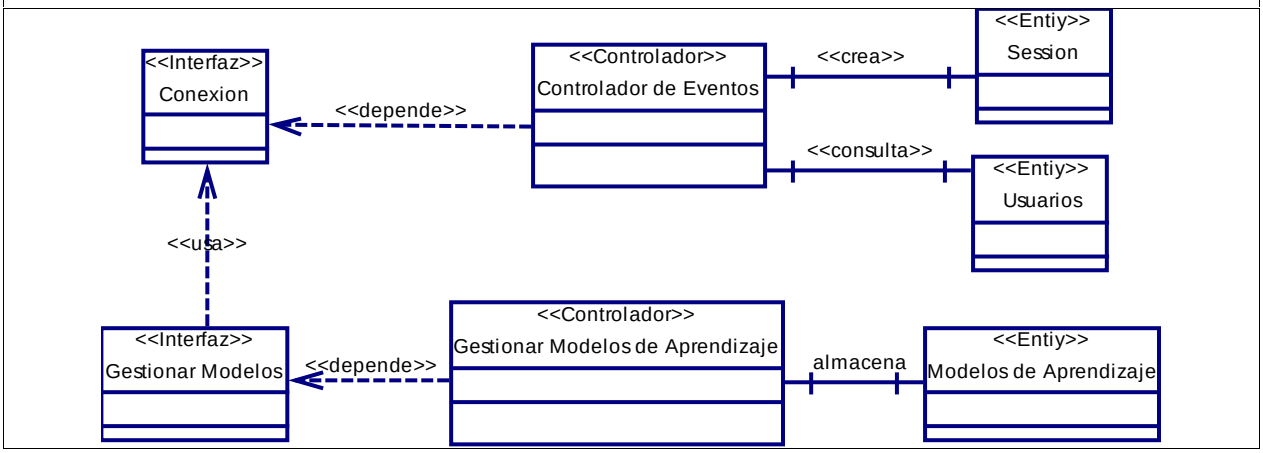


Tabla A.1.1.4 Análisis del Caso de uso Gestionar modelos de aprendizaje

A.1.2.1.2 Diseño

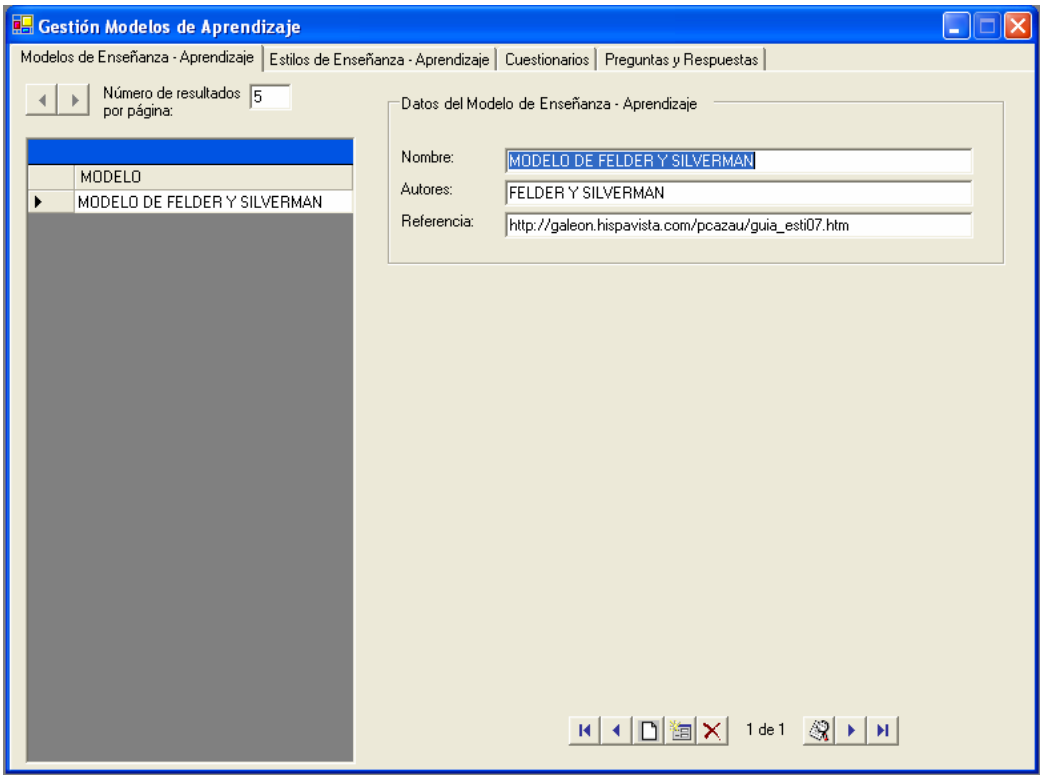
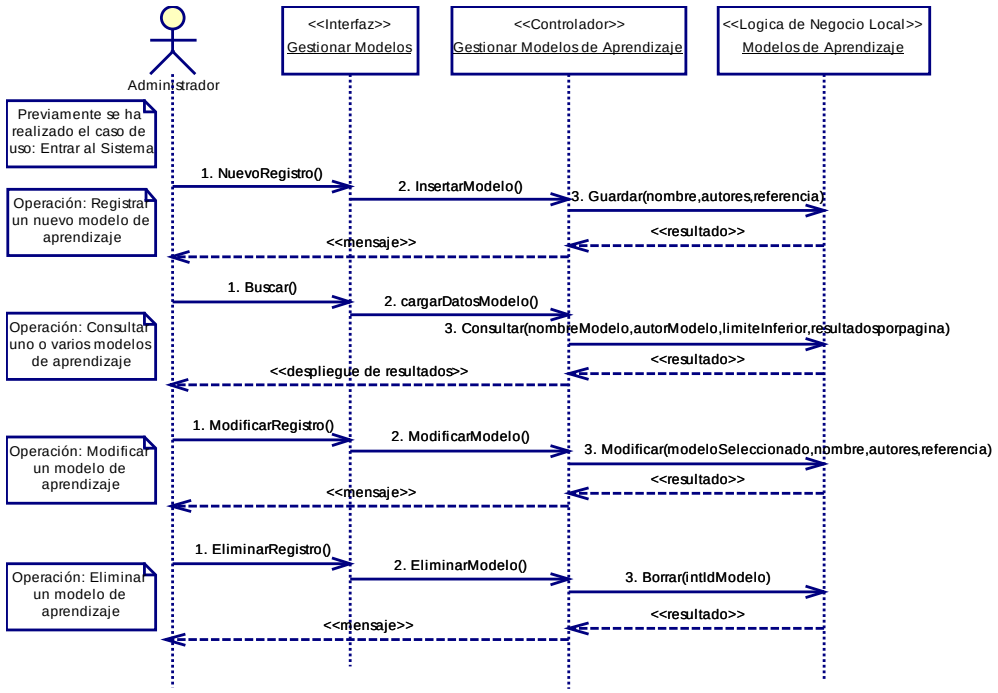


Figura 2. Screen Shot del caso de uso Gestionar Modelos de Aprendizaje

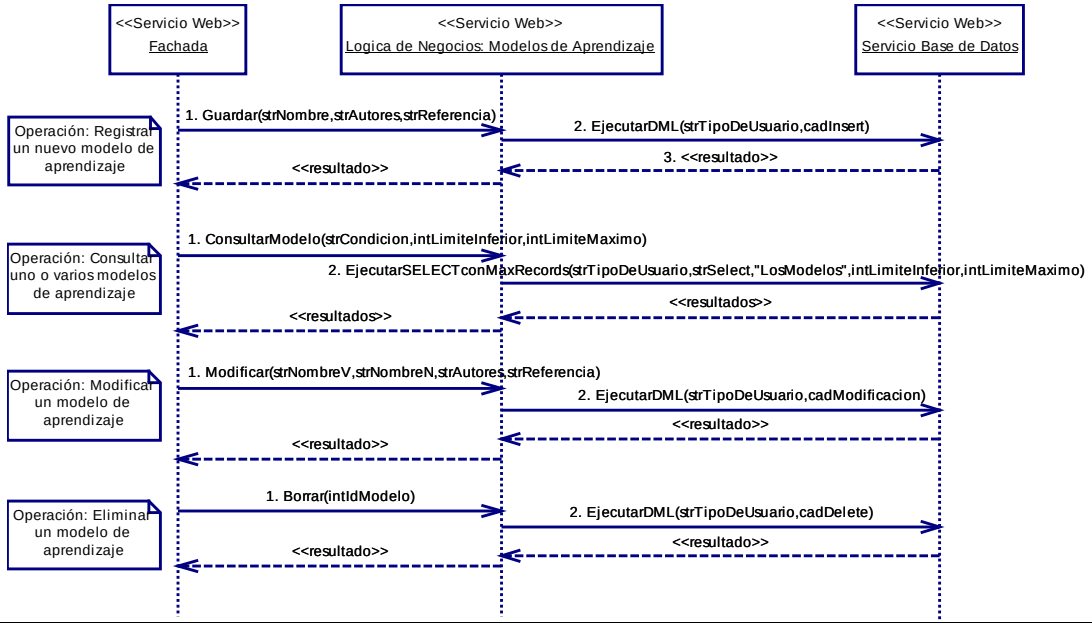
FORMATO CASO DE USO: GESTIONAR MODELOS	
Actores	
Administrador (Iniciador)	
Este caso de uso comienza cuando el administrador decide Gestionar Modelos. El sistema solicita primero al administrador que se loguee, invocando el caso de uso Entrar al Sistema. Después de este proceso el sistema presenta al administrador la interfaz principal, donde el administrador elige la opción "Modelos de Enseñanza-Aprendizaje", que presenta la interfaz que gestiona los modelos de aprendizaje a través de las cuatro operaciones básicas (registrar, consultar, modificar y eliminar). Dependiendo de la operación que realice el administrador, el controlador envía los datos del modelo a la clase Modelos de Aprendizaje de Lógica de Negocios Local, la cual se comunica y propaga los datos a través de la referencia Proxy del servicio Web Fachada, quien se comunica con el servicio Web Modelos de Aprendizaje de la lógica de Negocios, el cual gestiona el modelo en la base de datos, ayudado del servicio Web Lógica de Servicios. Finalmente el sistema, envía un mensaje al administrador sobre el éxito de la operación. Este caso de uso finaliza cuando el administrador: abandona el proceso, cierra la sesión o cierra la aplicación.	
Acción del Actor	**Respuesta del Sistema**
1. El administrador decide gestionar modelos de aprendizaje.	2. El sistema redirecciona al administrador a la interfaz de Conexión.
3. El administrador utiliza el caso de uso Entrar al Sistema.	4. Si Entrar al sistema es exitoso, el sistema presenta la interfaz principal de la aplicación.
5. El administrador hace clic en alguno de los botones (adicionar, consultar, modificar o eliminar) que permiten la gestión de los modelos de aprendizaje.	6. El sistema envía los datos del modelo y ejecuta el proceso de gestión de acuerdo a la operación efectuada, también retorna un mensaje del resultado de dicha operación.
CAPA DE LOGICA DE PRESENTACION: DIAGRAMA DE SECUENCIA DEL CASO DE USO GESTIONAR MODELOS	
El administrador hace clic en alguno de los botones (adicionar, consultar, modificar o eliminar) de la interfaz Gestionar Modelos. Una vez que el administrador realiza la operación de su elección para gestionar los modelos, esta información la envía el controlador a la clase Modelos de Aprendizaje, a través de los métodos definidos para cada operación. Esta clase también está encargada de comunicarse y esperar respuesta del Servicio Web. Según el resultado, se despliega un mensaje al administrador.	

FORMATO CASO DE USO: GESTIONAR MODELOS



CAPA DE LOGICA DEL NEGOCIO Y SERVICIOS: DIAGRAMA DE SECUENCIA DEL CASO DE USO GESTIONAR MODELOS

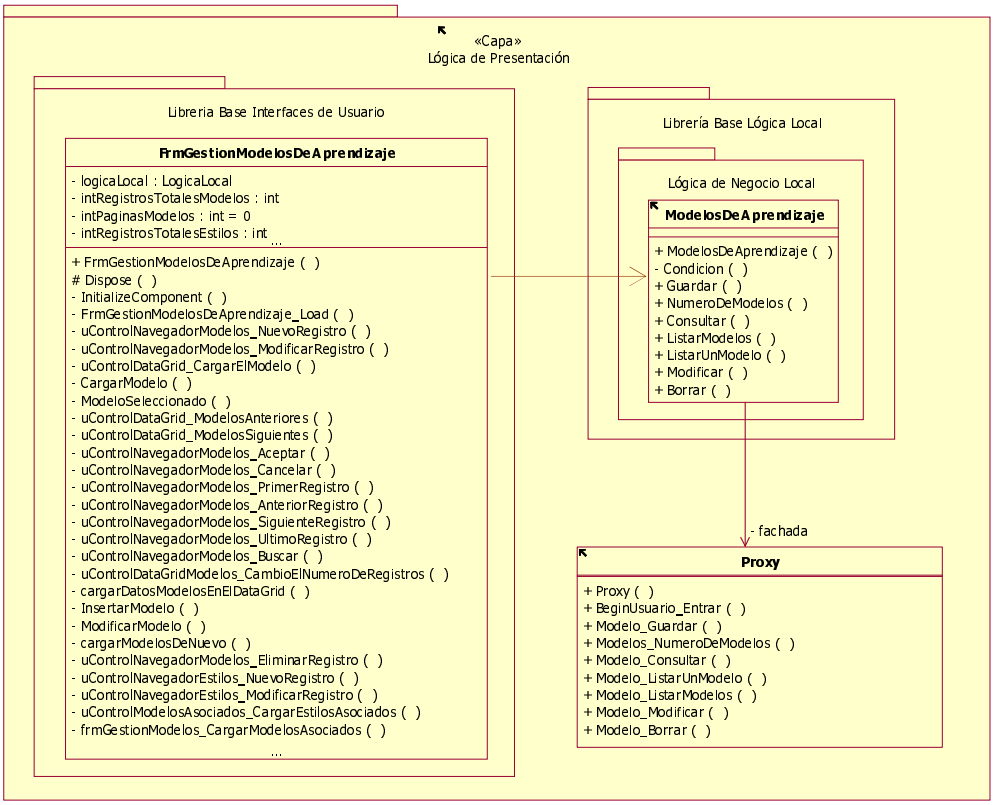
El servicio Web Fachada, recibe desde la capa de presentación la información del modelo de aprendizaje, este servicio Web delega al servicio Web Modelos de Aprendizaje el proceso de gestión del modelo de acuerdo a la operación efectuada por el administrador, utilizando el patrón experto. El servicio Web Modelos de Aprendizaje procesa los datos e invoca dependiendo de la operación de gestión efectuada al método respectivo de la clase de lógica Servicios base de datos, que se encarga de ejecutar el proceso de gestión contra la base de datos. Luego se informa del éxito o fracaso de dicho proceso a través del servicio Web hasta llegar a la capa de presentación.



FORMATO CASO DE USO: GESTIONAR MODELOS

CAPA DE LOGICA DE PRESENTACION: DIAGRAMA DE CLASES DEL CASO DE USO GESTIONAR MODELOS

La interfaz Gestionar Modelos se relaciona con el controlador a través de su código subyacente, que se relaciona con la lógica de negocios local a través de la clase Modelos de Aprendizaje, que se relaciona con el Proxy de la Fachada, dado que Gestionar Modelos utiliza métodos del Servicio Web de la aplicación.



CAPA DE LOGICA DEL NEGOCIO Y SERVICIOS: DIAGRAMA DE CLASES DEL CASO DE USO GESTIONAR MODELOS

Fachada instancia un objeto de Modelos de Aprendizaje para delegarle el proceso de gestión, existiendo una asociación directa entre ellas. El objeto Modelos de Aprendizaje utiliza los métodos adecuados de la clase de Servicios de la base de datos de acuerdo a la operación de gestión, por lo que existe una asociación directa entre estas.

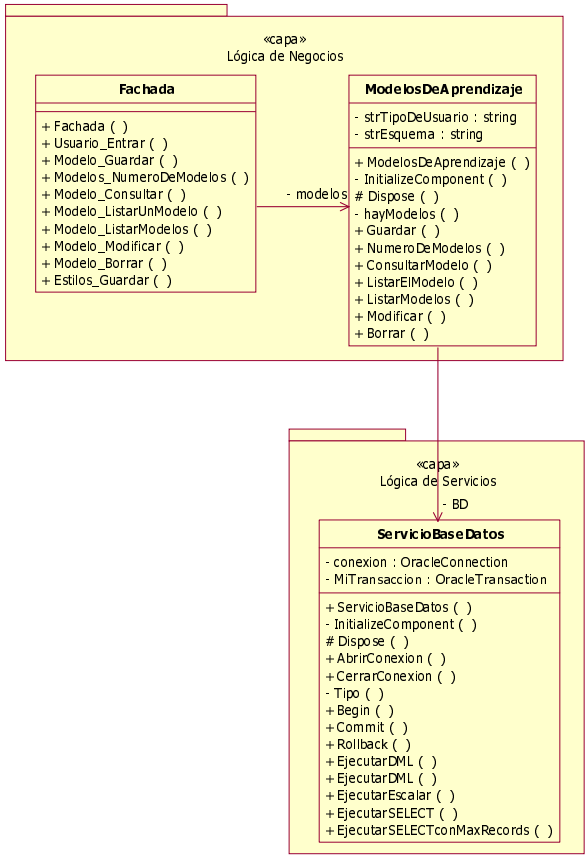
FORMATO CASO DE USO: GESTIONAR MODELOS


Tabla D.1.1.2. Diseño del caso de uso Gestionar Modelos

A.1.2.2 Caso de uso: Gestionar Estilos de Enseñanza-Aprendizaje

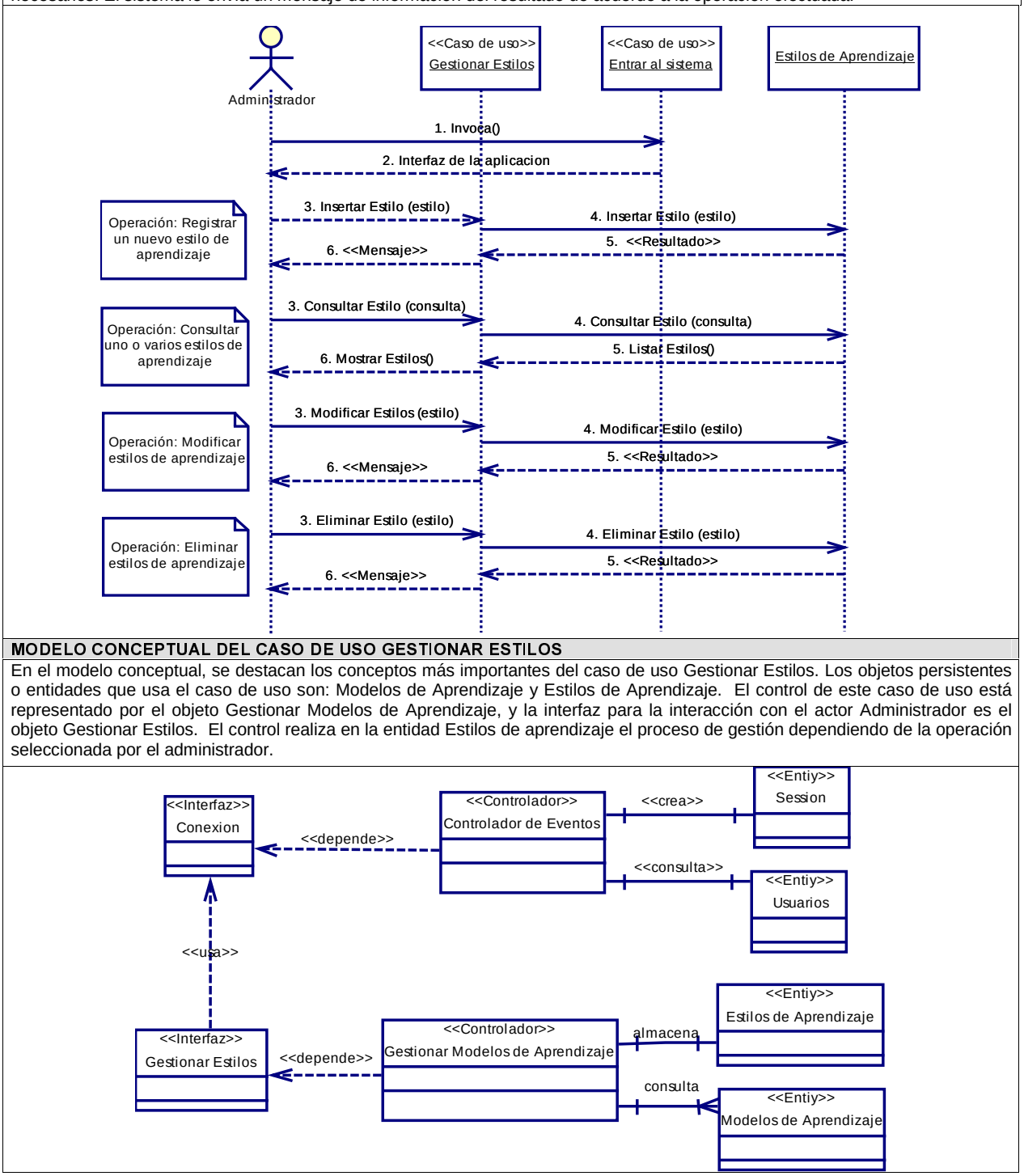
A.1.2.2.1 Análisis

FORMATO CASO DE USO GESTIONAR ESTILOS	
Actores	
• Administrador (Iniciador)	
Este caso de uso comienza cuando el administrador invoca al caso de uso Entrar al sistema, con el fin de gestionar (registrar, consultar, modificar o eliminar) un estilo de aprendizaje asociado a algún modelo de aprendizaje gestionado a través del caso de uso: Gestionar Modelos. El sistema solicita la información relacionada con el estilo de aprendizaje y realiza el respectivo proceso dependiendo de la operación que desee el administrador (registrar, consultar, modificar o eliminar), luego envía un mensaje al administrador del resultado de la operación efectuada. Este caso de uso finaliza cuando el administrador: abandona el proceso, cierra la sesión o cierra la aplicación.	
Acción del Actor	**Respuesta del Sistema**
1. El administrador decide gestionar (registrar, consultar, modificar o eliminar) un estilo asociado a un modelo de aprendizaje	**2**. El sistema redirecciona al administrador a la interfaz de conexión.
3. El administrador utiliza el caso de uso **Entrar al sistema**.	4. Si Entrar al sistema es exitoso, el sistema presenta la interfaz principal de la aplicación.
5. El administrador ingresa la información del estilo de aprendizaje dependiendo de la operación de gestión (registrar, consultar, modificar o eliminar) seleccionada.	**6**. El sistema realiza el respectivo proceso dependiendo de la operación que seleccionó el administrador, y le informa el resultado de dicha operación.

FORMATO CASO DE USO GESTIONAR ESTILOS

DIAGRAMA DE SECUENCIA DEL SISTEMA DEL CASO DE USO GESTIONAR ESTILOS

Este diagrama muestra todo el proceso necesario que debe hacer el actor administrador para realizar el caso de uso Gestionar Estilo. El administrador gestiona (registra, consulta, modifica o elimina) un estilo, luego de haber realizado los casos de uso necesarios. El sistema le envía un mensaje de información del resultado de acuerdo a la operación efectuada.



MODELO CONCEPTUAL DEL CASO DE USO GESTIONAR ESTILOS

En el modelo conceptual, se destacan los conceptos más importantes del caso de uso Gestionar Estilos. Los objetos persistentes o entidades que usa el caso de uso son: Modelos de Aprendizaje y Estilos de Aprendizaje. El control de este caso de uso está representado por el objeto Gestionar Modelos de Aprendizaje, y la interfaz para la interacción con el actor Administrador es el objeto Gestionar Estilos. El control realiza en la entidad Estilos de aprendizaje el proceso de gestión dependiendo de la operación seleccionada por el administrador.

Tabla A.1.1.5 Análisis del Caso de uso Gestionar Estilos de Aprendizaje

A.1.2.2.2 Diseño

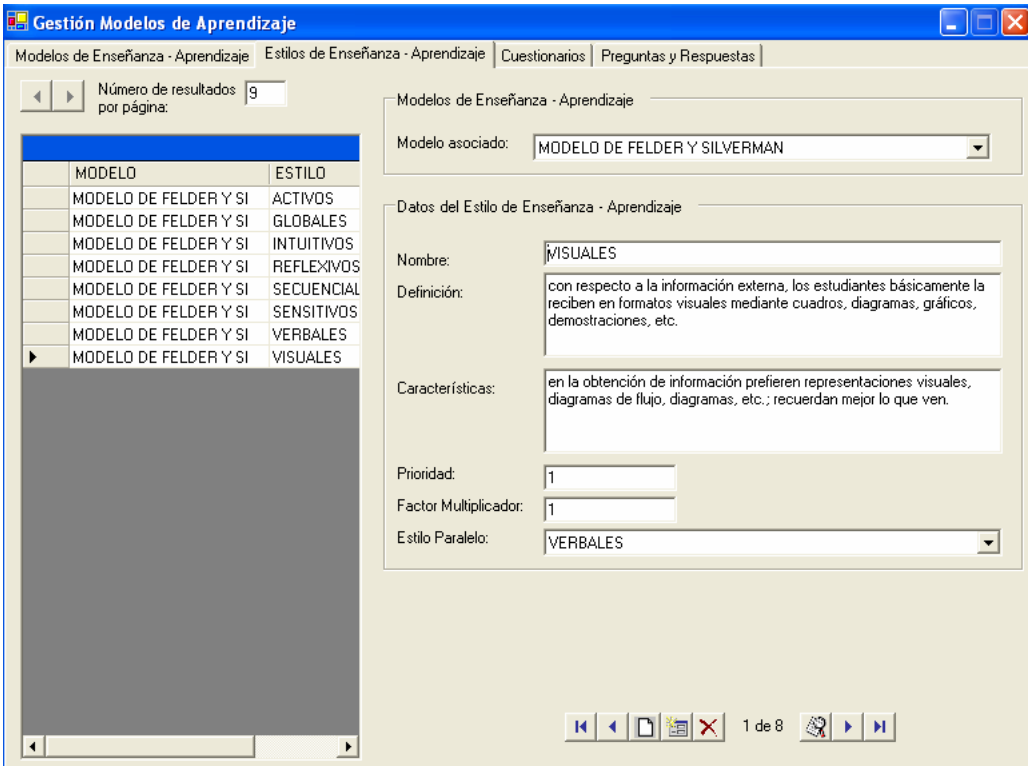
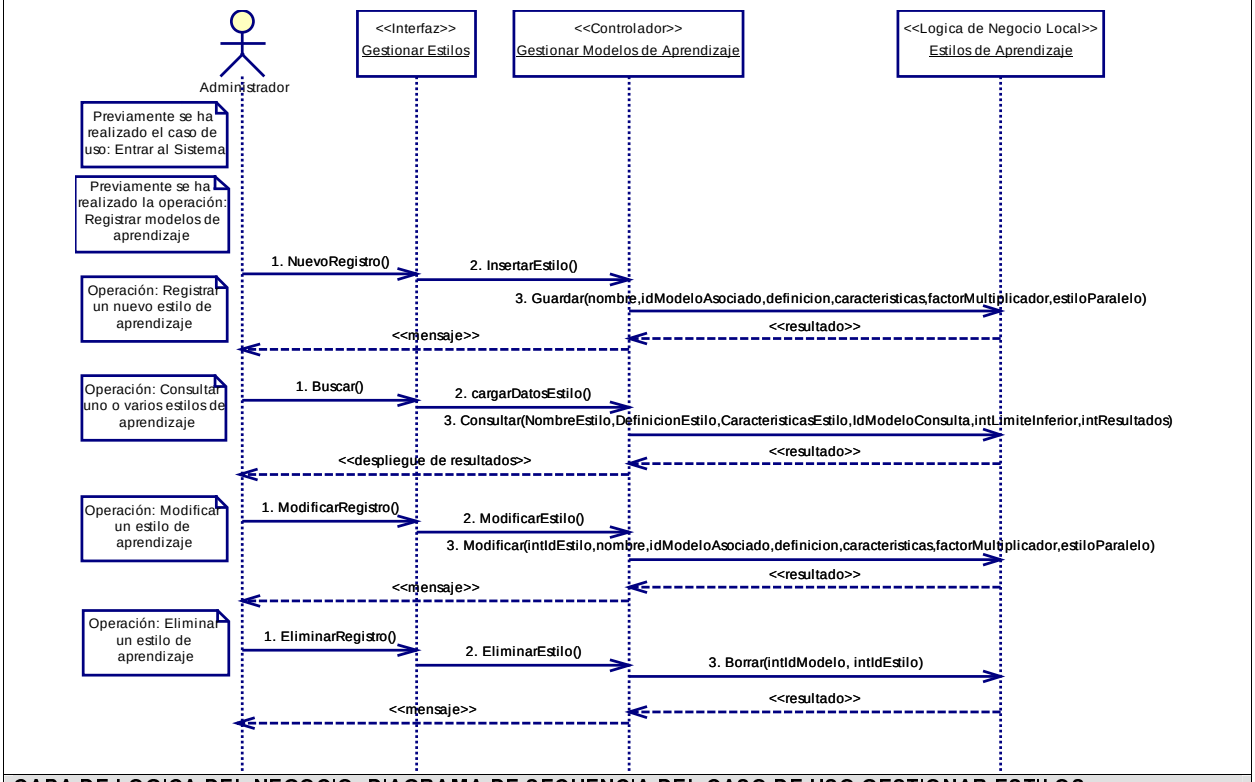


Figura 3. Screen Shot del Caso de uso Gestionar Estilos de Aprendizaje

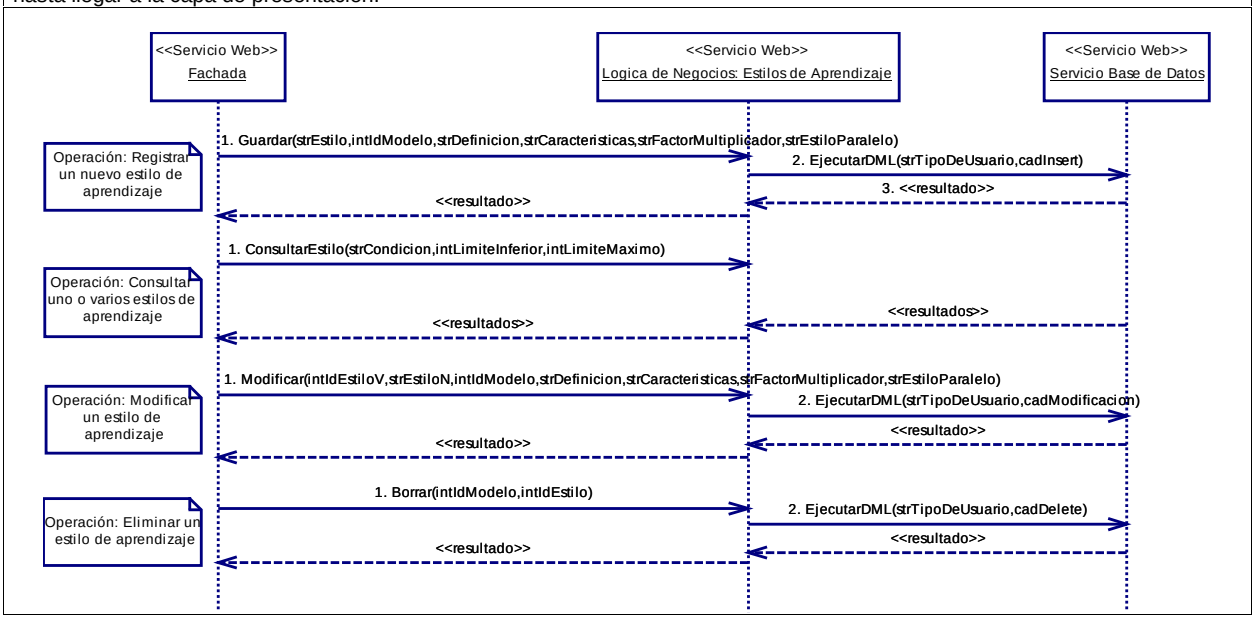
FORMATO CASO DE USO: GESTIONAR ESTILOS	
Actores	
Administrador (Iniciador)	
Este caso de uso comienza cuando el administrador decide Gestionar Estilos. El sistema solicita primero al administrador que se loguee, invocando el caso de uso Entrar al Sistema. Después de este proceso el sistema presenta al administrador la interfaz principal, donde el administrador elige la opción "Estilos de Aprendizaje", que presenta la interfaz que gestiona los estilos de aprendizaje a través de las cuatro operaciones básicas (registrar, consultar, modificar y eliminar). Pero para realizar la operación de registro de un estilo, el administrador debe haber gestionado un modelo de aprendizaje a través del caso de uso Gestionar Modelos. Dependiendo de la operación que realice el administrador, el controlador envía los datos del estilo a la clase Estilos de Aprendizaje de Lógica de Negocios Local, la cual se comunica y propaga los datos a través de la referencia Proxy del servicio Web Fachada, quien se comunica con el servicio Web Estilos de Aprendizaje de la lógica de Negocios, el cual gestiona el estilo en la base de datos, ayudado del servicio Web Lógica de Servicios. Finalmente el sistema, envía un mensaje al administrador sobre el éxito de la operación. Este caso de uso finaliza cuando el administrador: abandona el proceso, cierra la sesión o cierra la aplicación.	
Acción del Actor	**Respuesta del Sistema**
1. El administrador decide gestionar estilos de aprendizaje.	2. El sistema redirecciona al administrador a la interfaz de Conexión.
3. El administrador utiliza el caso de uso Entrar al Sistema.	4. Si Entrar al sistema es exitoso, el sistema presenta la interfaz principal de la aplicación.
5. El administrador hace clic en alguno de los botones (adicionar, consultar, modificar o eliminar) que permiten la gestión de los estilos de aprendizaje.	6. El sistema envía los datos del estilo y ejecuta el proceso de gestión de acuerdo a la operación efectuada, también retorna un mensaje del resultado de dicha operación.
CAPA DE LOGICA DE PRESENTACION: DIAGRAMA DE SECUENCIA DEL CASO DE USO GESTIONAR ESTILOS	
El administrador hace clic en alguno de los botones (adicionar, consultar, modificar o eliminar) de la interfaz Gestionar Estilos. Una vez que el administrador realiza la operación de su elección para gestionar los estilos, esta información la envía el controlador a la clase Estilos de Aprendizaje, a través de los métodos definidos para cada operación. Esta clase también está encargada de comunicarse y esperar respuesta del Servicio Web. Según el resultado se despliega un mensaje al administrador.	

FORMATO CASO DE USO: GESTIONAR ESTILOS

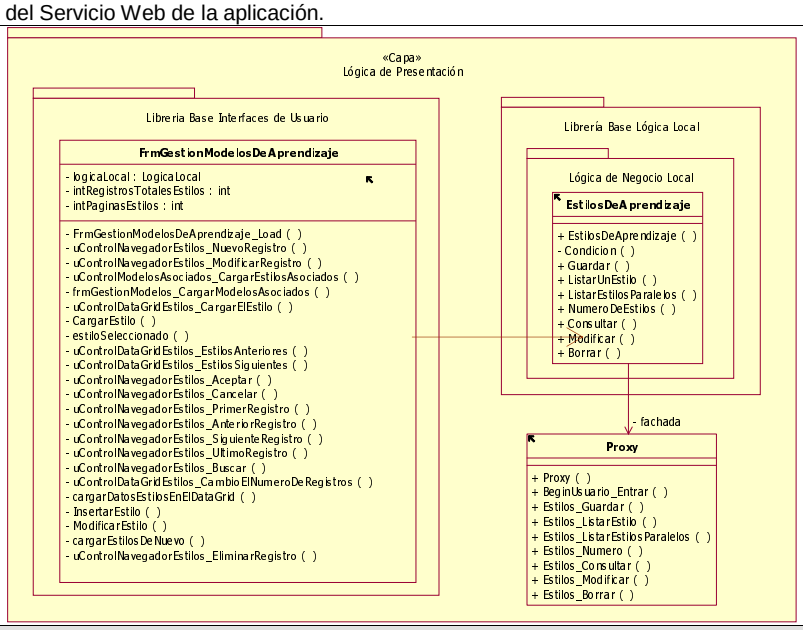


CAPA DE LOGICA DEL NEGOCIO: DIAGRAMA DE SECUENCIA DEL CASO DE USO GESTIONAR ESTILOS

El servicio Web Fachada, recibe desde la capa de presentación la información del estilo de aprendizaje, este servicio Web delega al servicio Web Estilos de Aprendizaje el proceso de gestión del estilo de acuerdo a la operación efectuada por el administrador, utilizando el patrón experto. El servicio Web Estilos de Aprendizaje procesa los datos e invoca dependiendo de la operación de gestión efectuada al método respectivo de la clase de lógica Servicios base de datos, que se encarga de ejecutar el proceso de gestión contra la base de datos. Luego se información el éxito o fracaso de dicho proceso a través del servicio Web hasta llegar a la capa de presentación.



FORMATO CASO DE USO: GESTIONAR ESTILOS
CAPA DE LOGICA DE PRESENTACION: DIAGRAMA DE CLASES DEL CASO DE USO GESTIONAR ESTILOS
La interfaz Gestionar Estilos se relaciona con el controlador a través de su código subyacente, que se relaciona con la lógica de negocios local a través de la clase Estilos de Aprendizaje, que se relaciona con el Proxy de la Fachada, dado que Gestionar Estilos utiliza métodos del Servicio Web de la aplicación.



CAPA DE LOGICA DEL NEGOCIO Y SERVICIO: DIAGRAMA DE CLASES DEL CASO DE USO GESTIONAR ESTILOS
Fachada instancia un objeto de Estilos de Aprendizaje para delegarle el proceso de gestión, existiendo una asociación directa entre ellas. El objeto Estilos de Aprendizaje utiliza los métodos adecuados de la clase de Servicios de la base de datos de acuerdo a la operación de gestión, por lo que existe una asociación directa entre estas.

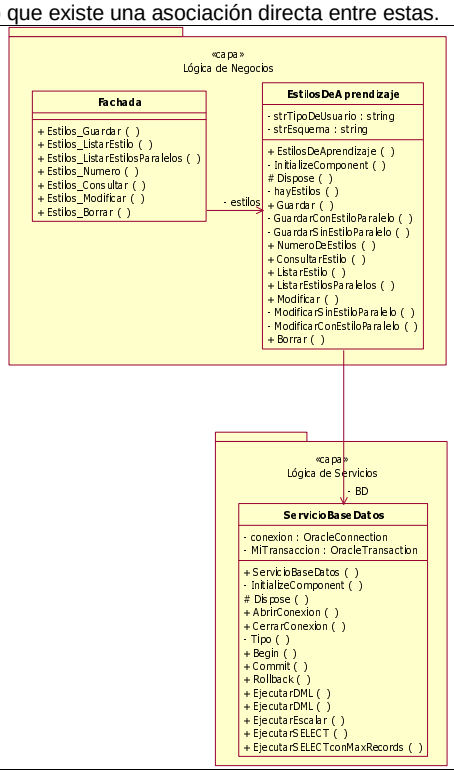
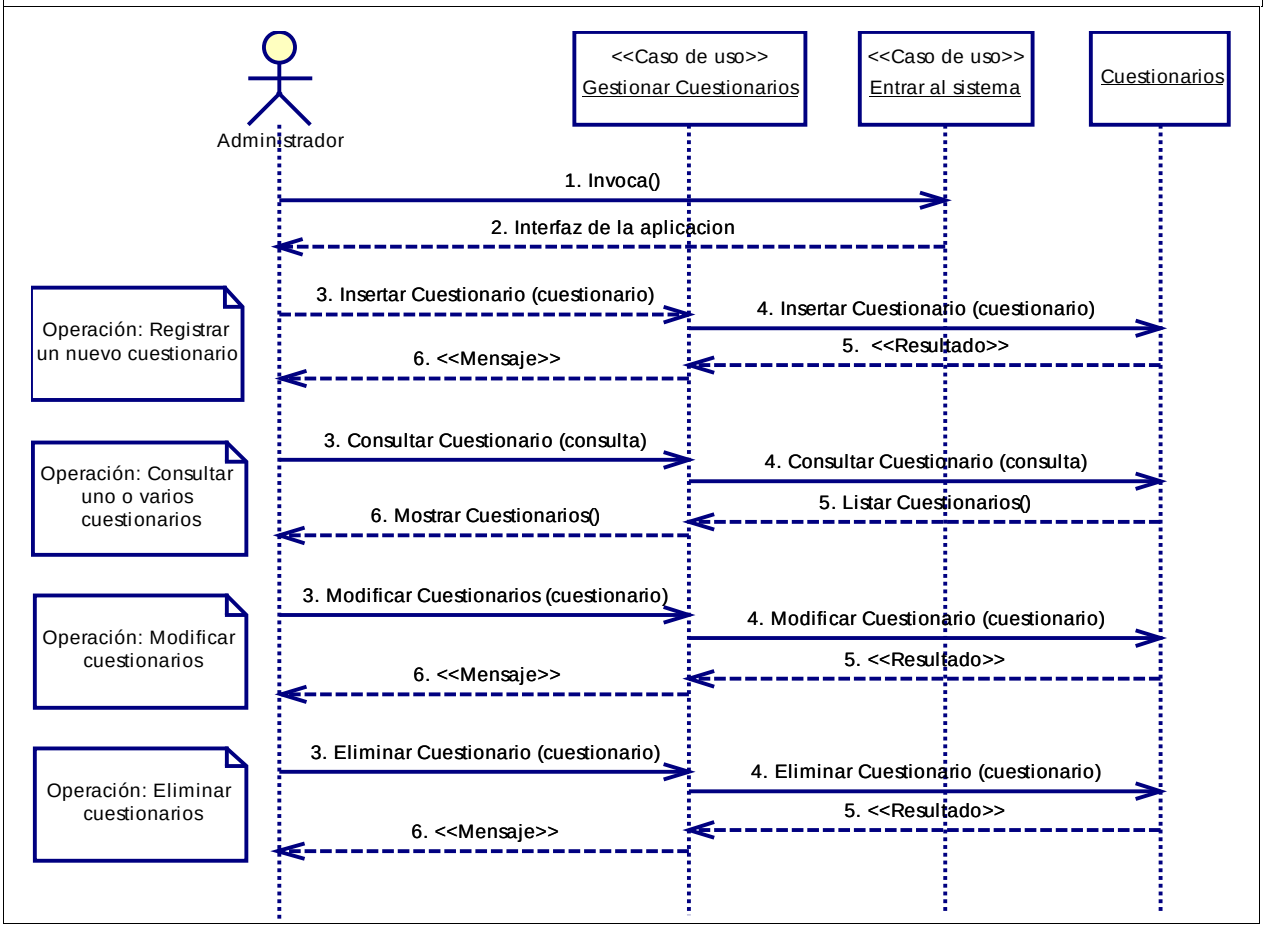


Tabla D.1.1.3. Diseño del caso de uso Gestionar Estilos

A.1.2.3 Caso de uso: Gestionar Cuestionarios

A.1.2.3.1 Análisis

FORMATO CASO DE USO GESTIONAR CUESTIONARIOS	
Actores	
• Administrador (Iniciador)	
Este caso de uso comienza cuando el administrador invoca al caso de uso Entrar al sistema, con el fin de gestionar (registrar, consultar, modificar o eliminar) un cuestionario asociado a algún modelo de aprendizaje gestionado a través del caso de uso: Gestionar Modelos. El sistema solicita la información relacionada con el cuestionario y realiza el respectivo proceso dependiendo de la operación que desee el administrador (registrar, consultar, modificar o eliminar), luego envía un mensaje al administrador del resultado de la operación efectuada. Este caso de uso finaliza cuando el administrador: abandona el proceso, cierra la sesión o cierra la aplicación.	
Acción del Actor	**Respuesta del Sistema**
1. El administrador decide gestionar (registrar, consultar, modificar o eliminar) un cuestionario asociado a un modelo de aprendizaje	**2**. El sistema redirecciona al administrador a la interfaz de conexión.
3. El administrador utiliza el caso de uso **Entrar al sistema**.	4. Si Entrar al sistema es exitoso, el sistema presenta la interfaz principal de la aplicación.
5. El administrador ingresa la información del cuestionario dependiendo de la operación de gestión (registrar, consultar, modificar o eliminar) seleccionada.	**6**. El sistema realiza el respectivo proceso dependiendo de la operación que seleccionó el administrador, y le informa el resultado de la misma.
DIAGRAMA DE SECUENCIA DEL SISTEMA DEL CASO DE USO GESTIONAR CUESTIONARIO	
Este diagrama muestra todo el proceso necesario que debe hacer el actor administrador para realizar el caso de uso Gestionar Cuestionario. El administrador gestiona (registra, consulta, modifica o elimina) un cuestionario, luego de haber realizado los casos de uso necesarios. El sistema le envía un mensaje de información del resultado de acuerdo a la operación efectuada.	



FORMATO CASO DE USO GESTIONAR CUESTIONARIOS
MODELO CONCEPTUAL DEL CASO DE USO GESTIONAR CUESTIONARIOS
En el modelo conceptual, se destacan los conceptos más importantes del caso de uso Gestionar Cuestionarios. Los objetos persistentes o entidades que usa el caso de uso son: Modelos de Aprendizaje y Cuestionarios. El control de este caso de uso está representado por el objeto Gestionar Modelos de Aprendizaje, y la interfaz para la interacción con el actor Administrador es el objeto Gestionar Cuestionarios. El control realiza en la entidad Cuestionario el proceso de gestión dependiendo de la operación seleccionada por el administrador.

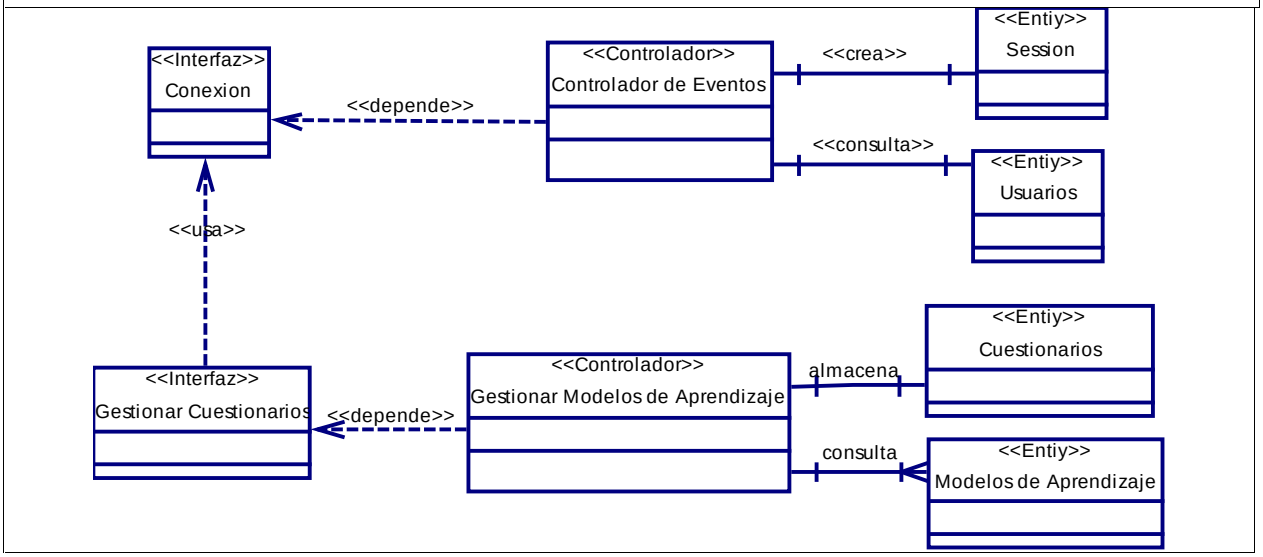


Tabla A.1.1.6 Caso de uso Gestionar Cuestionarios

A.1.2.3.2 Diseño

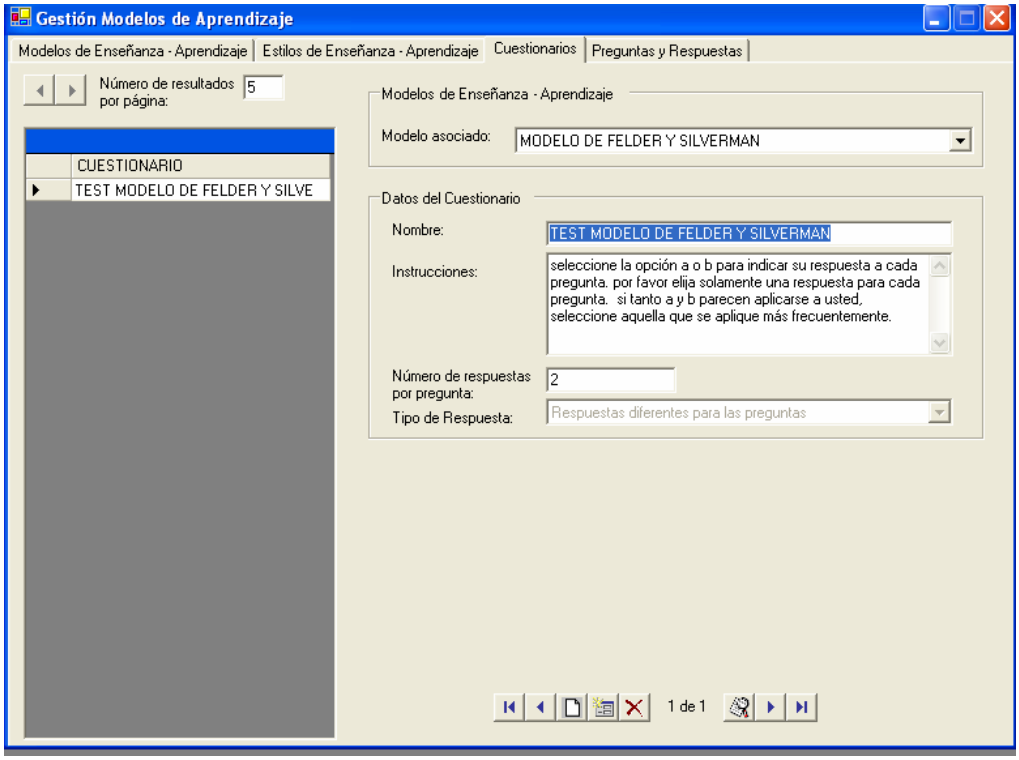
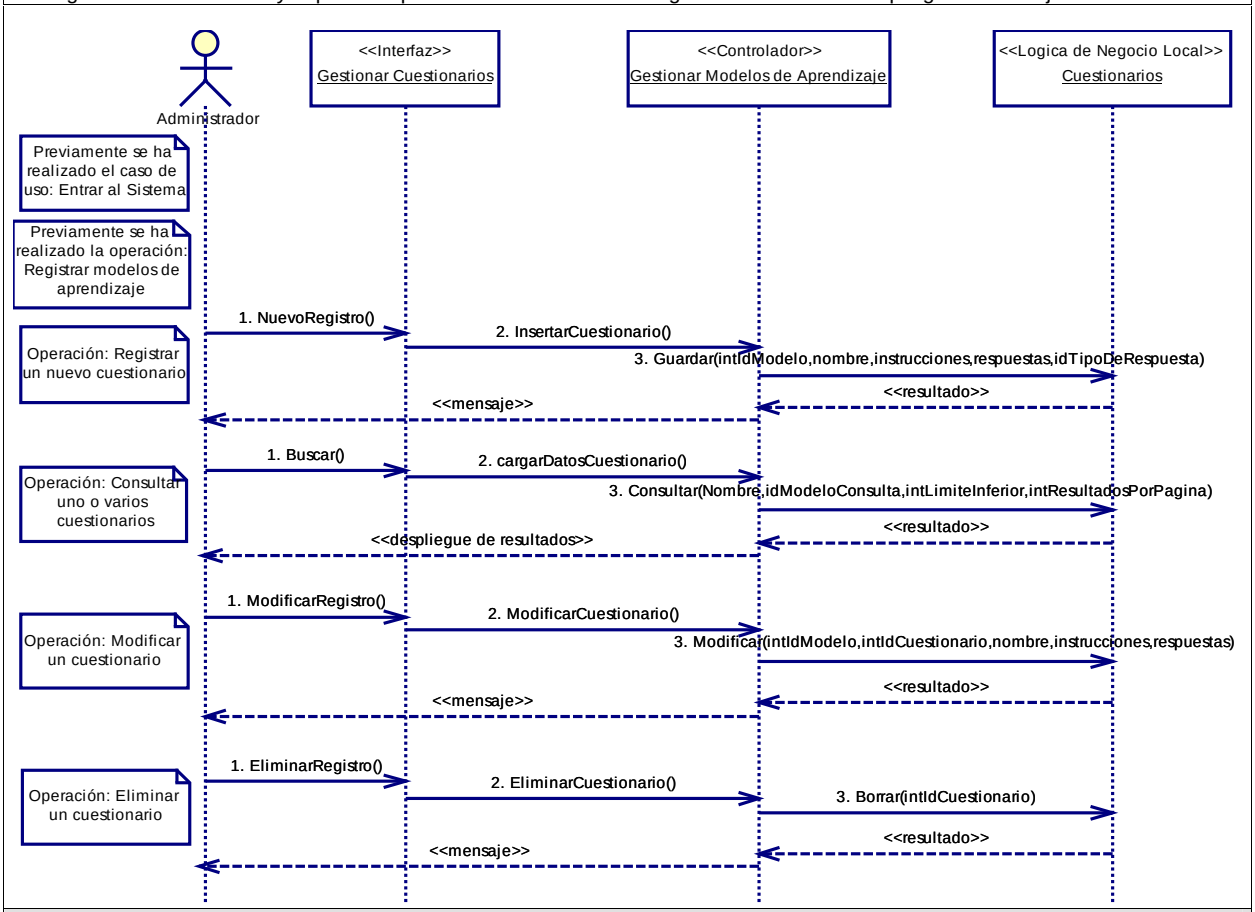


Figura 4. Screen Shot del Caso de uso Gestionar Cuestionarios

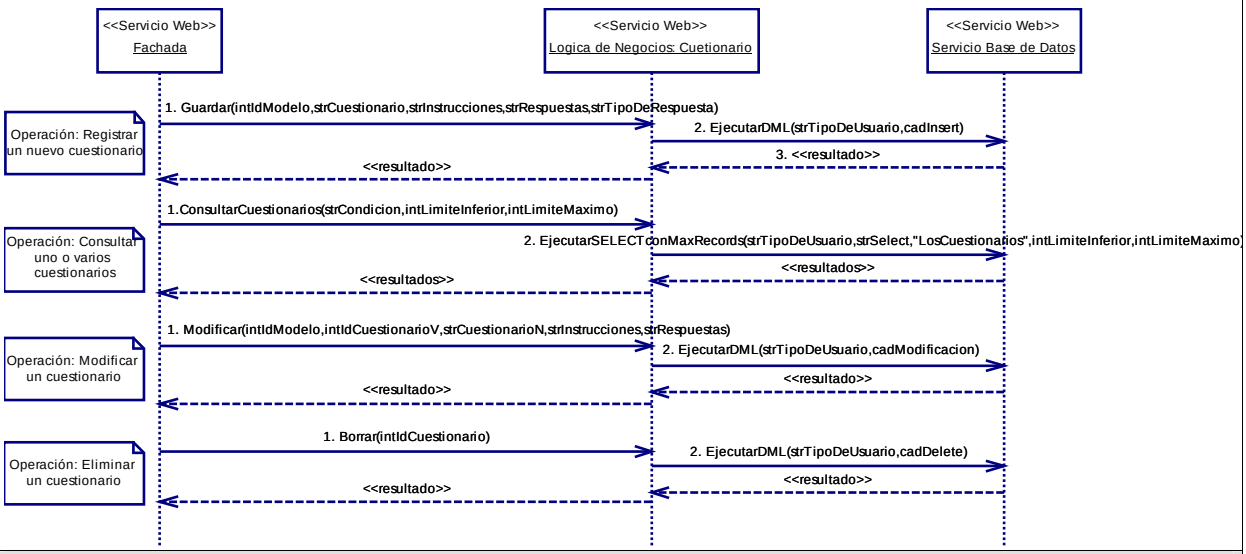
FORMATO CASO DE USO: GESTIONAR CUESTIONARIOS	
Actores	
Administrador (Iniciador)	
Este caso de uso comienza cuando el administrador decide Gestionar Cuestionarios. El sistema solicita primero al administrador que se loguee, invocando el caso de uso Entrar al Sistema. Después de este proceso el sistema presenta al administrador la interfaz principal, donde el administrador elige la opción "Cuestionarios", que presenta la interfaz que gestiona los Cuestionarios a través de las cuatro operaciones básicas (registrar, consultar, modificar y eliminar). Pero para realizar la operación de registro de un cuestionario, el administrador debe haber gestionado un modelo de aprendizaje a través del caso de uso Gestionar Modelos. Dependiendo de la operación que realice el administrador, el controlador envía los datos del cuestionario a la clase Cuestionarios de Lógica de Negocios Local, la cual se comunica y propaga los datos a través de la referencia Proxy del servicio Web Fachada, quien se comunica con el servicio Web Cuestionarios de la lógica de Negocios, el cual gestiona el cuestionario en la base de datos, ayudado del servicio Web Lógica de Servicios. Finalmente el sistema, envía un mensaje al administrador sobre el éxito de la operación. Este caso de uso finaliza cuando el administrador: abandona el proceso, cierra la sesión o cierra la aplicación.	
Acción del Actor	**Respuesta del Sistema**
1. El administrador decide gestionar cuestionarios asociados a los modelos de aprendizaje.	2. El sistema redirecciona al administrador a la interfaz de Conexión.
3. El administrador utiliza el caso de uso Entrar al Sistema.	4. Si Entrar al sistema es exitoso, el sistema presenta la interfaz principal de la aplicación.
5. El administrador hace click en alguno de los botones (adicionar, consultar, modificar o eliminar) que permiten la gestión de los cuestionarios.	6. El sistema envía los datos del cuestionario y ejecuta el proceso de gestión de acuerdo a la operación efectuada, también retorna un mensaje del resultado de dicha operación.
CAPA DE LOGICA DE PRESENTACION: DIAGRAMA DE SECUENCIA DEL CASO DE USO GESTIONAR CUESTIONARIOS	
El administrador hace clic en alguno de los botones (adicionar, consultar, modificar o eliminar) de la interfaz Gestionar Cuestionarios. Una vez que el administrador realiza la operación de su elección para gestionar los cuestionarios, esta información la envía el controlador a la clase Cuestionarios, a través de los métodos definidos para cada operación. Esta clase también está encargada de comunicarse y esperar respuesta del Servicio Web. Según el resultado se despliega un mensaje al administrador.	



CAPA DE LOGICA DEL NEGOCIO: DIAGRAMA DE SECUENCIA DEL CASO DE USO GESTIONAR CUESTIONARIOS

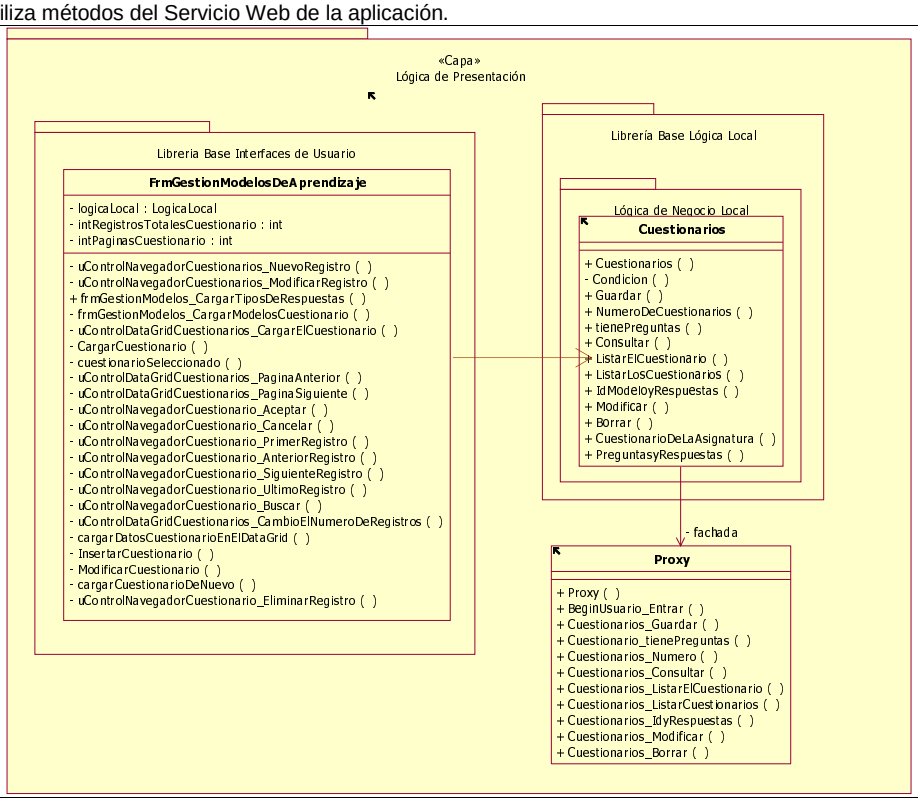
FORMATO CASO DE USO: GESTIONAR CUESTIONARIOS

El servicio Web Fachada, recibe desde la capa de presentación, la información del cuestionario, este servicio Web delega al servicio Web Cuestionarios el proceso de gestión del cuestionario de acuerdo a la operación efectuada por el administrador, utilizando el patrón experto. El servicio Web Cuestionario procesa los datos e invoca dependiendo de la operación de gestión efectuada al método respectivo de la clase de lógica Servicios base de datos, que se encarga de ejecutar el proceso de gestión contra la base de datos. Luego se información el éxito o fracaso de dicho proceso a través del servicio Web hasta llegar a la capa de presentación.



CAPA DE LOGICA DE PRESENTACION: DIAGRAMA DE CLASES DEL CASO DE USO GESTIONAR CUESTIONARIOS

La interfaz Gestionar Cuestionarios se relaciona con el controlador a través de su código subyacente, que se relaciona con la lógica de negocios local a través de la clase Cuestionarios, que se relaciona con el Proxy de la Fachada, dado que Gestionar Cuestionarios utiliza métodos del Servicio Web de la aplicación.



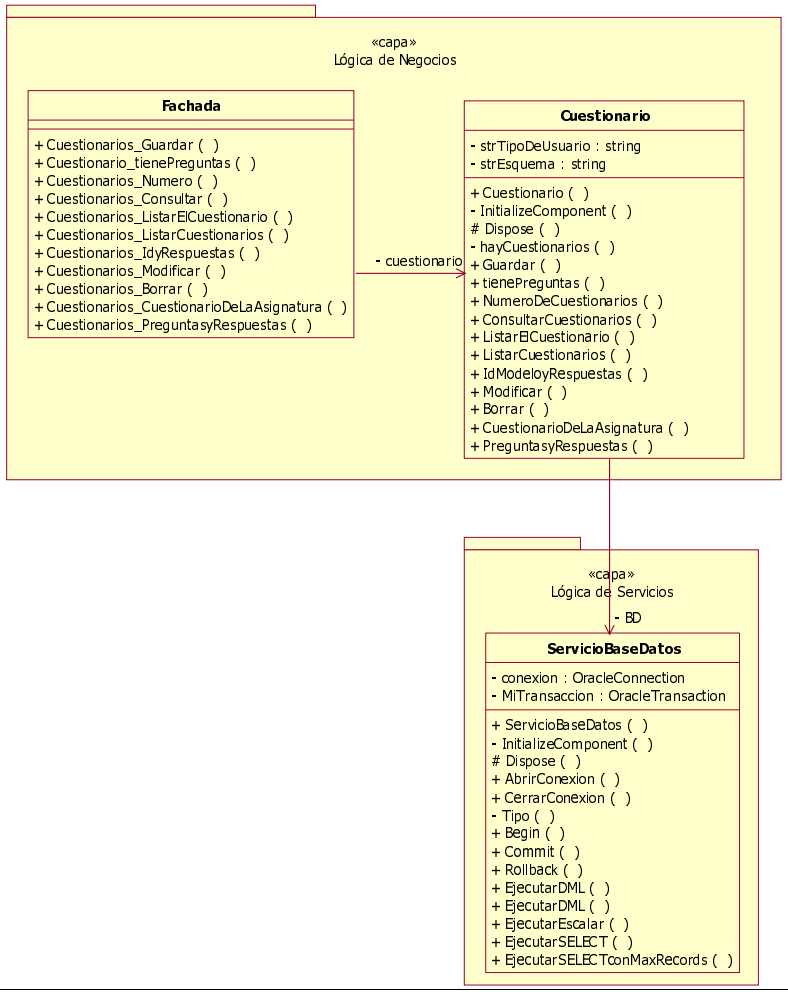
FORMATO CASO DE USO: GESTIONAR CUESTIONARIOS
CAPA DE LOGICA DEL NEGOCIO Y SERVICIO: DIAGRAMA DE CLASES DEL CASO DE USO GESTIONAR CUESTIONARIOS
Fachada instancia un objeto de Cuestionarios para delegarle el proceso de gestión, existiendo una asociación directa entre ellas. El objeto Cuestionarios utiliza los métodos adecuados de la clase de Servicios de la base de datos de acuerdo a la operación de gestión, por lo que existe una asociación directa entre estas.


Tabla D.1.1.4. Diseño del caso de uso Gestionar Cuestionarios

A.1.2.4 Caso de uso: Gestionar Preguntas y Respuestas

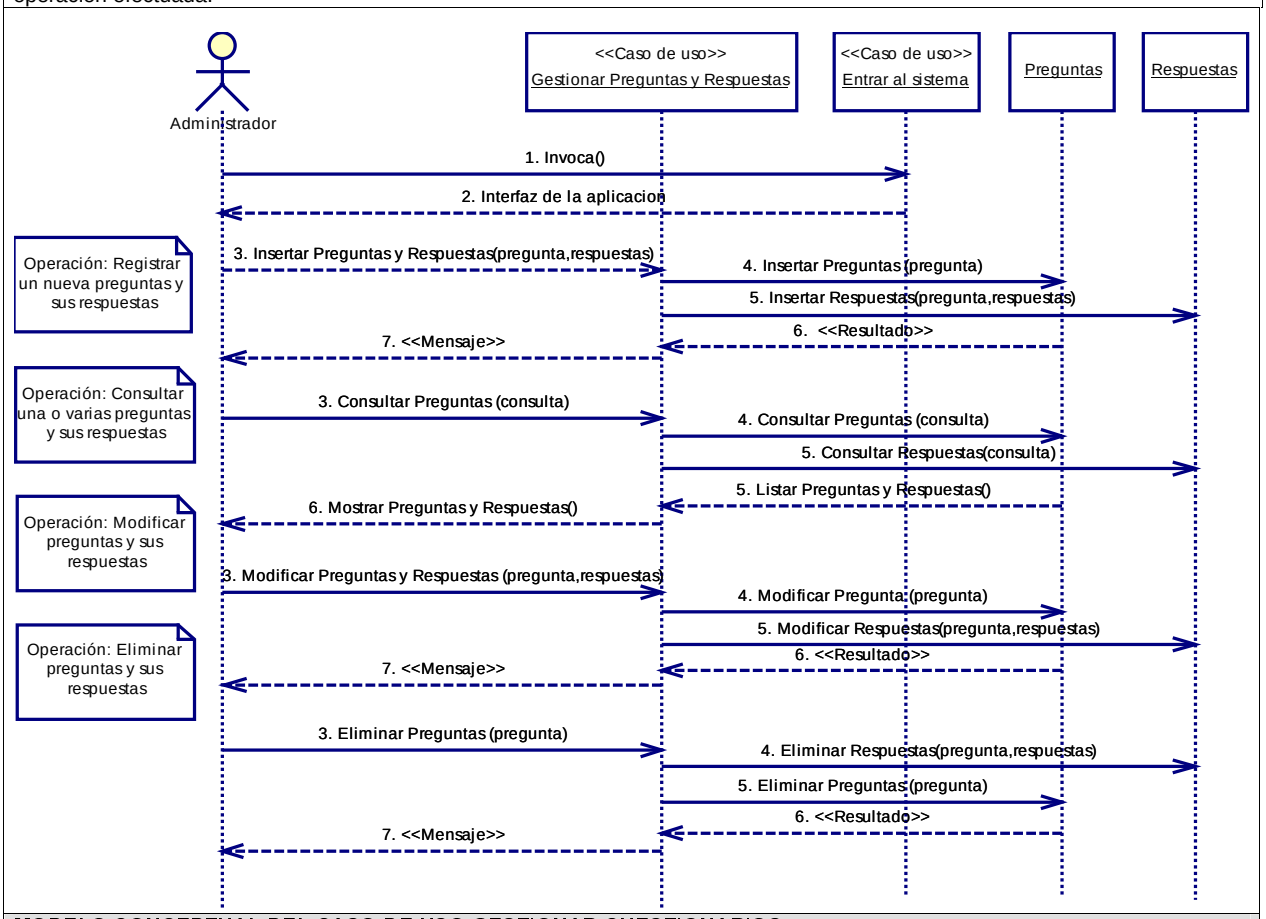
A.1.2.4.1 Análisis

FORMATO CASO DE USO GESTIONAR PREGUNTAS Y RESPUESTAS
Actores
• Administrador (Iniciador)
Este caso de uso comienza cuando el administrador invoca al caso de uso Entrar al sistema, con el fin de gestionar (registrar, consultar, modificar o eliminar) una pregunta y sus respuestas, asociadas a algún cuestionario gestionado a través del caso de uso: Gestionar Cuestionarios. El sistema solicita la información relacionada con la pregunta y sus respuestas y realiza el respectivo proceso dependiendo de la operación que desee el administrador (registrar, consultar, modificar o eliminar), luego envía un mensaje al administrador del resultado de la operación efectuada. Este caso de uso finaliza cuando el administrador: abandona el proceso, cierra la sesión o cierra la aplicación.

FORMATO CASO DE USO GESTIONAR PREGUNTAS Y RESPUESTAS	
Acción del Actor	**Respuesta del Sistema**
1. El administrador decide gestionar (registrar, consultar, modificar o eliminar) una pregunta y sus respuestas, asociadas a un cuestionario.	**2**. El sistema redirecciona al administrador a la interfaz de conexión.
3. El administrador utiliza el caso de uso **Entrar al sistema**.	4. Si Entrar al sistema es exitoso, el sistema presenta la interfaz principal de la aplicación.
5. El administrador ingresa la información de la pregunta y sus respuestas dependiendo de la operación de gestión (registrar, consultar, modificar o eliminar) seleccionada.	**6**. El sistema realiza el respectivo proceso dependiendo de la operación que seleccionó el administrador, y le informa el resultado de la misma.

DIAGRAMA DE SECUENCIA DEL SISTEMA DEL CASO DE USO GESTIONAR PREGUNTAS Y RESPUESTAS

Este diagrama muestra todo el proceso necesario que debe hacer el actor administrador para realizar el caso de uso Gestionar Preguntas y Respuestas. El administrador gestiona (registra, consulta, modifica o elimina) una pregunta y sus respuestas, luego de haber realizado los casos de uso necesarios. El sistema le envía un mensaje de información del resultado de acuerdo a la operación efectuada.



MODELO CONCEPTUAL DEL CASO DE USO GESTIONAR CUESTIONARIOS

En el modelo conceptual, se destacan los conceptos más importantes del caso de uso Gestionar Preguntas y Respuestas. Los objetos persistentes o entidades que usa el caso de uso son: Cuestionario, Preguntas y Respuestas. El control de este caso de uso está representado por el objeto Gestionar Modelos de Aprendizaje, y la interfaz para la interacción con el actor Administrador es el objeto Gestionar Preguntas y Respuestas.
El control realiza en las entidades Preguntas y Respuestas el proceso de gestión dependiendo de la operación seleccionada por el administrador.

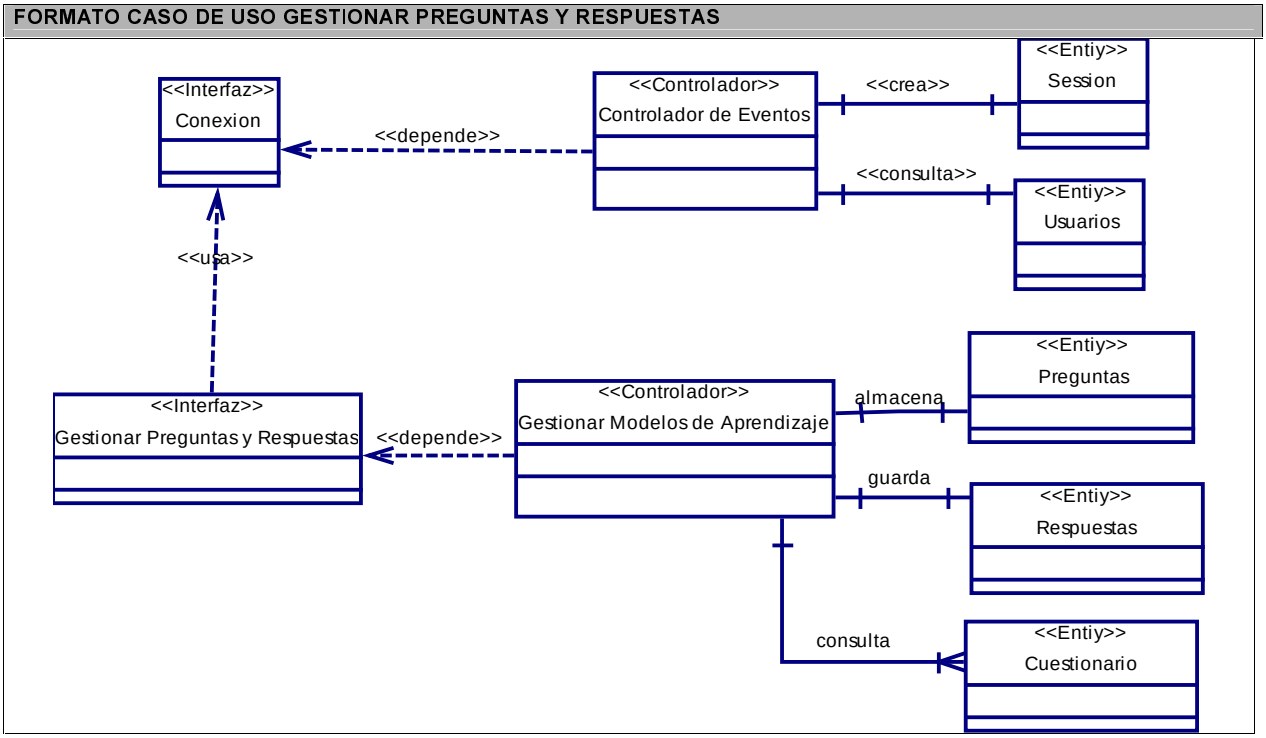


Tabla A.1.1.7 Caso de uso Gestionar Preguntas y Respuestas

A.1.2.4.2 Diseño

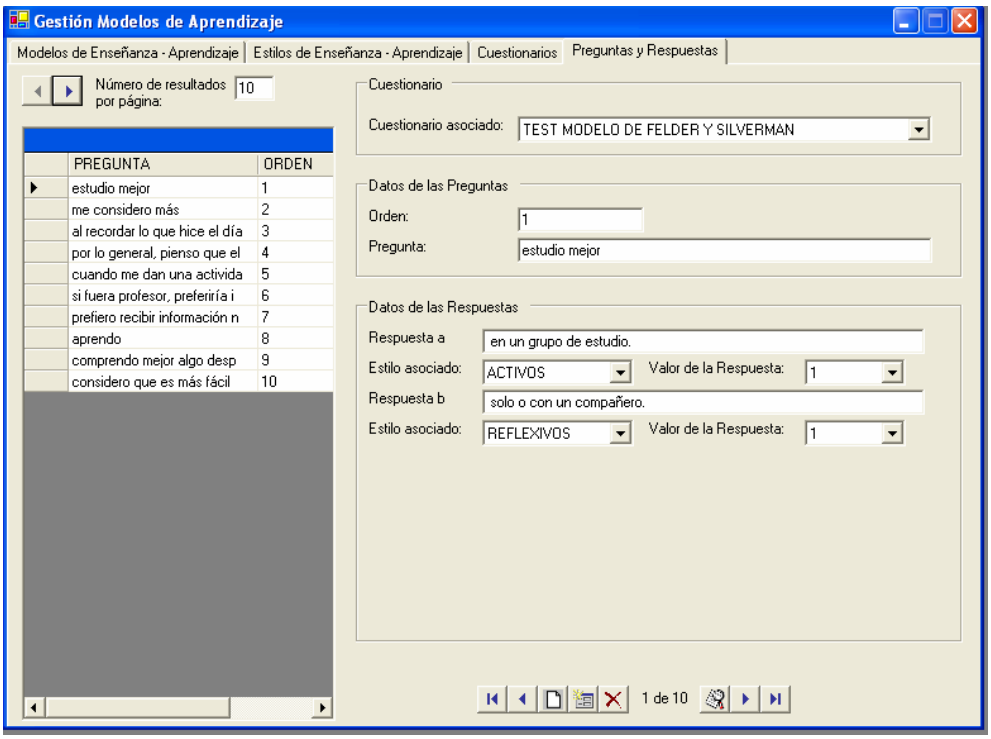
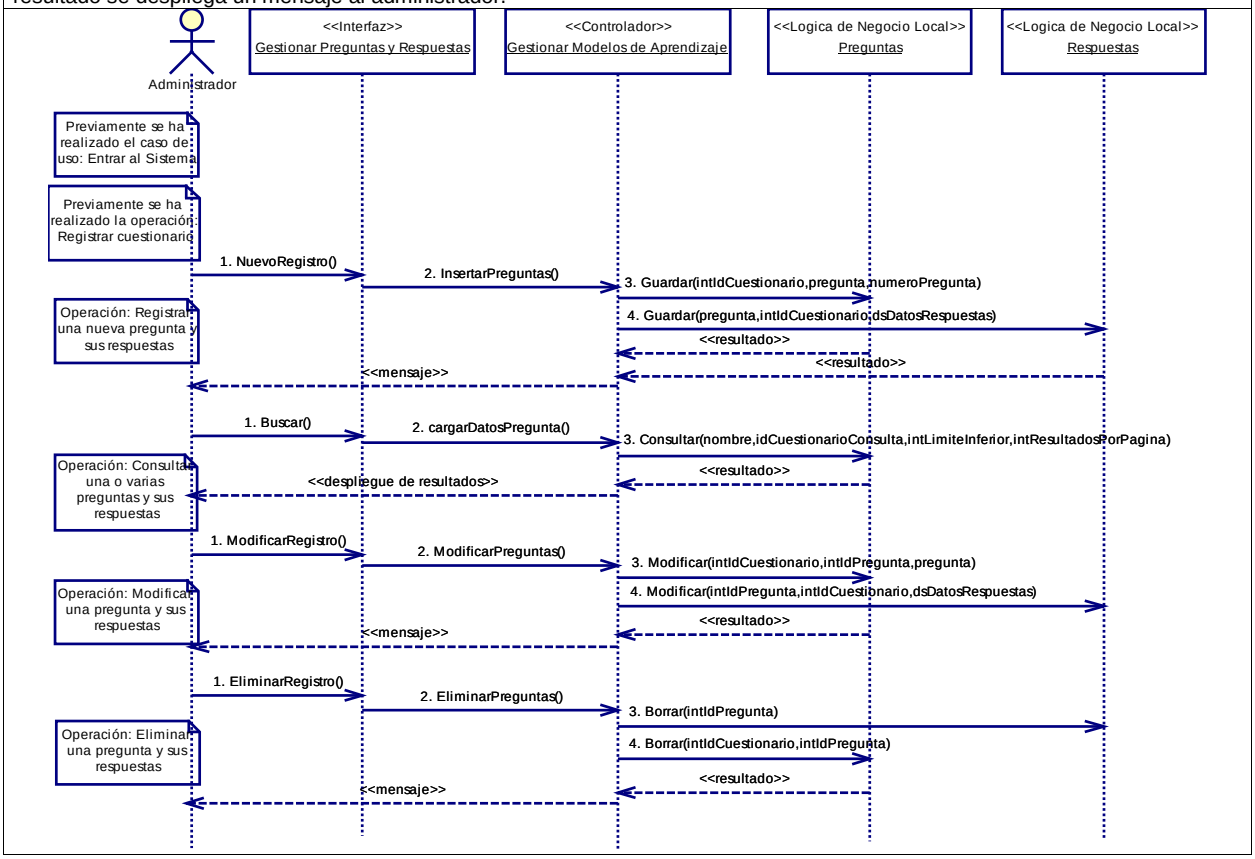


Figura 5. Screen Shot del Caso de uso Gestionar Preguntas y Respuestas

FORMATO CASO DE USO: GESTIONAR PREGUNTAS Y RESPUESTAS	
Actores	
Administrador (Iniciador)	
Este caso de uso comienza cuando el administrador decide Gestionar Preguntas y sus respuestas. El sistema solicita primero al administrador que se loguee, invocando el caso de uso Entrar al Sistema. Después de este proceso el sistema presenta al administrador la interfaz principal, donde el administrador elige la opción "Preguntas y Respuestas", que presenta la interfaz que gestiona las Preguntas y sus respuestas a través de las cuatro operaciones básicas (registrar, consultar, modificar y eliminar). Pero para realizar la operación de registro de una pregunta y sus respuestas, el administrador debe haber gestionado un cuestionario a través del caso de uso Gestionar Cuestionarios. Dependiendo de la operación que realice el administrador, el controlador envía los datos de la pregunta a la clase Preguntas y los datos de las respuestas a la clase Respuestas, clases de Lógica de Negocios Local, las cuales se comunican y propagan los datos a través de la referencia Proxy del servicio Web Fachada, quien se comunica con los servicios Web Preguntas y Respuestas de la lógica de Negocios, los cuales gestionan las preguntas y sus respuestas en la base de datos, ayudados del servicio Web Lógica de Servicios. Finalmente el sistema, envía un mensaje al administrador sobre el éxito de la operación. Este caso de uso finaliza cuando el administrador: abandona el proceso, cierra la sesión o cierra la aplicación.	
Acción del Actor	**Respuesta del Sistema**
1. El administrador decide gestionar preguntas y sus respuestas asociadas a un cuestionario.	2. El sistema redirecciona al administrador a la interfaz de Conexión.
3. El administrador utiliza el caso de uso Entrar al Sistema.	4. Si Entrar al sistema es exitoso, el sistema presenta la interfaz principal de la aplicación.
5. El administrador hace click en alguno de los botones (adicionar, consultar, modificar o eliminar) que permiten la gestión de las preguntas y sus respuestas.	6. El sistema envía los datos de las preguntas y sus respuestas y ejecuta el proceso de gestión de acuerdo a la operación efectuada, también retorna un mensaje del resultado de dicha operación.

CAPA DE LOGICA DE PRESENTACION: DIAGRAMA DE SECUENCIA DEL CASO DE USO GESTIONAR PREGUNTAS Y RESPUESTAS

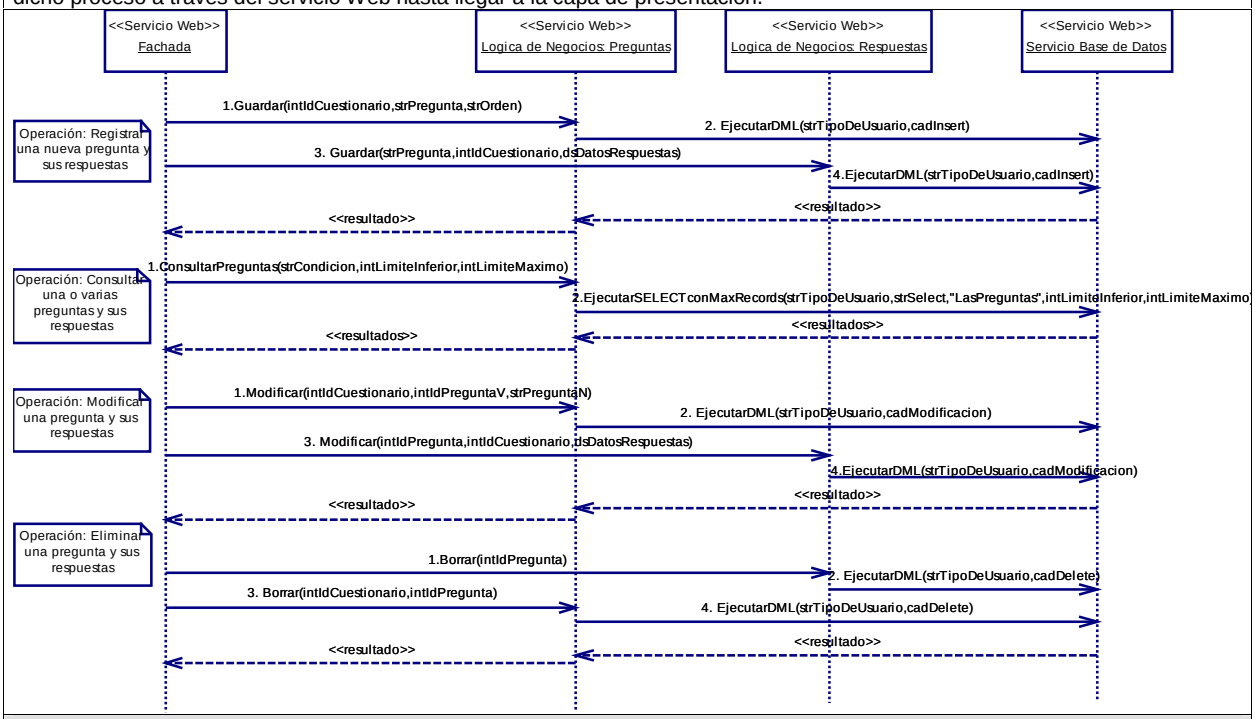
El administrador hace clic en alguno de los botones (adicionar, consultar, modificar o eliminar) de la interfaz Gestionar Preguntas y Respuestas. Una vez que el administrador realiza la operación de su elección para gestionar las preguntas y sus respuestas, esta información la envía el controlador a las clases Preguntas y Respuestas respectivamente, a través de los métodos definidos para cada operación. Estas clases también están encargadas de comunicarse y esperar respuesta del Servicio Web. Según el resultado se despliega un mensaje al administrador.



FORMATO CASO DE USO: GESTIONAR PREGUNTAS Y RESPUESTAS

CAPA DE LOGICA DEL NEGOCIO: DIAGRAMA DE SECUENCIA DEL CASO DE USO GESTIONAR PREGUNTAS Y RESPUESTAS

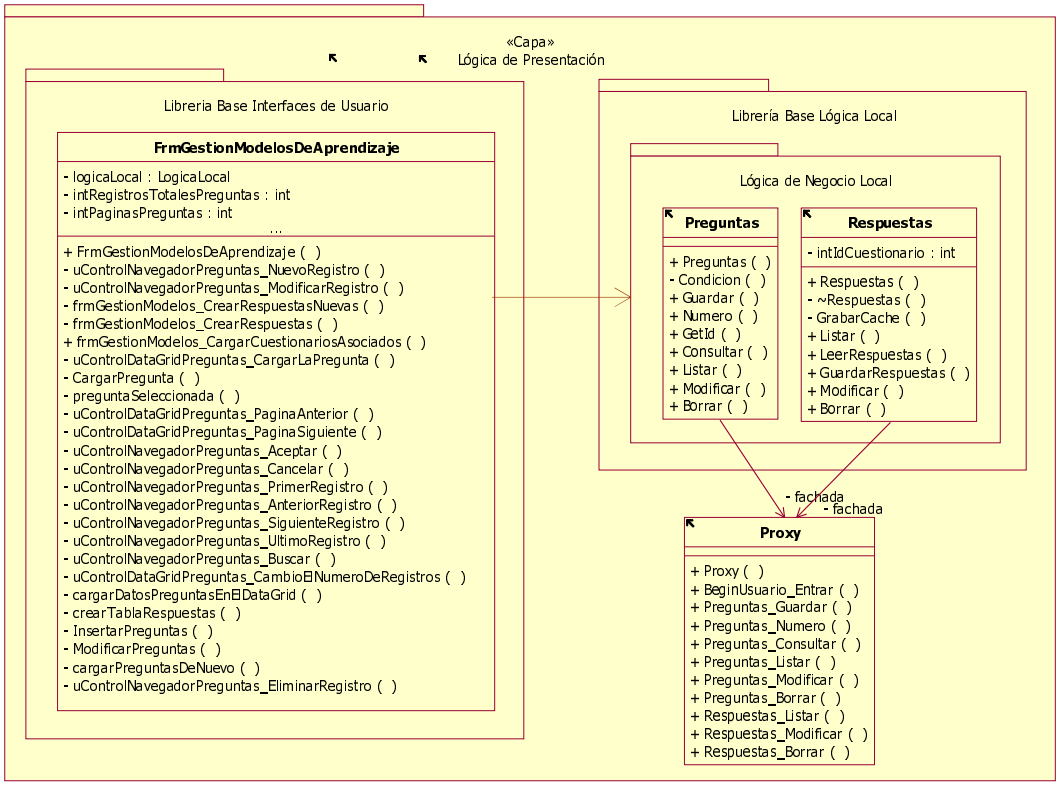
El servicio Web Fachada, recibe desde la capa de presentación, la información de las preguntas y sus respuestas, este servicio Web delega a los servicios Web Preguntas y Respuestas el proceso de gestión de las preguntas y respuestas de acuerdo a la operación efectuada por el administrador, utilizando el patrón experto. Los servicios Web Preguntas y Respuestas procesan los datos e invoca dependiendo de la operación de gestión efectuada al método respectivo de la clase de lógica Servicios base de datos, que se encarga de ejecutar el proceso de gestión contra la base de datos. Luego se información el éxito o fracaso de dicho proceso a través del servicio Web hasta llegar a la capa de presentación.



CAPA DE LOGICA DE PRESENTACION: DIAGRAMA DE CLASES DEL CASO DE USO GESTIONAR PREGUNTAS Y RESPUESTAS

La interfaz Gestionar Preguntas y Respuestas se relaciona con el controlador a través de su código subyacente, que se relaciona con la lógica de negocios local a través de las clases Preguntas y Respuestas, que se relaciona con el Proxy de la Fachada, dado que Gestionar Preguntas y Respuestas utiliza métodos del Servicio Web de la aplicación.

FORMATO CASO DE USO: GESTIONAR PREGUNTAS Y RESPUESTAS



CAPA DE LOGICA DEL NEGOCIO Y SERVICIO: DIAGRAMA DE CLASES DEL CASO DE USO GESTIONAR PREGUNTAS Y RESPUESTAS

Fachada instancia objetos de Preguntas y Respuestas para delegarle el proceso de gestión, existiendo una asociación directa entre ellas. Los objetos Preguntas y Respuestas utilizan los métodos adecuados de la clase de Servicios de la base de datos de acuerdo a la operación de gestión, por lo que existe una asociación directa entre estas.

FORMATO CASO DE USO: GESTIONAR PREGUNTAS Y RESPUESTAS

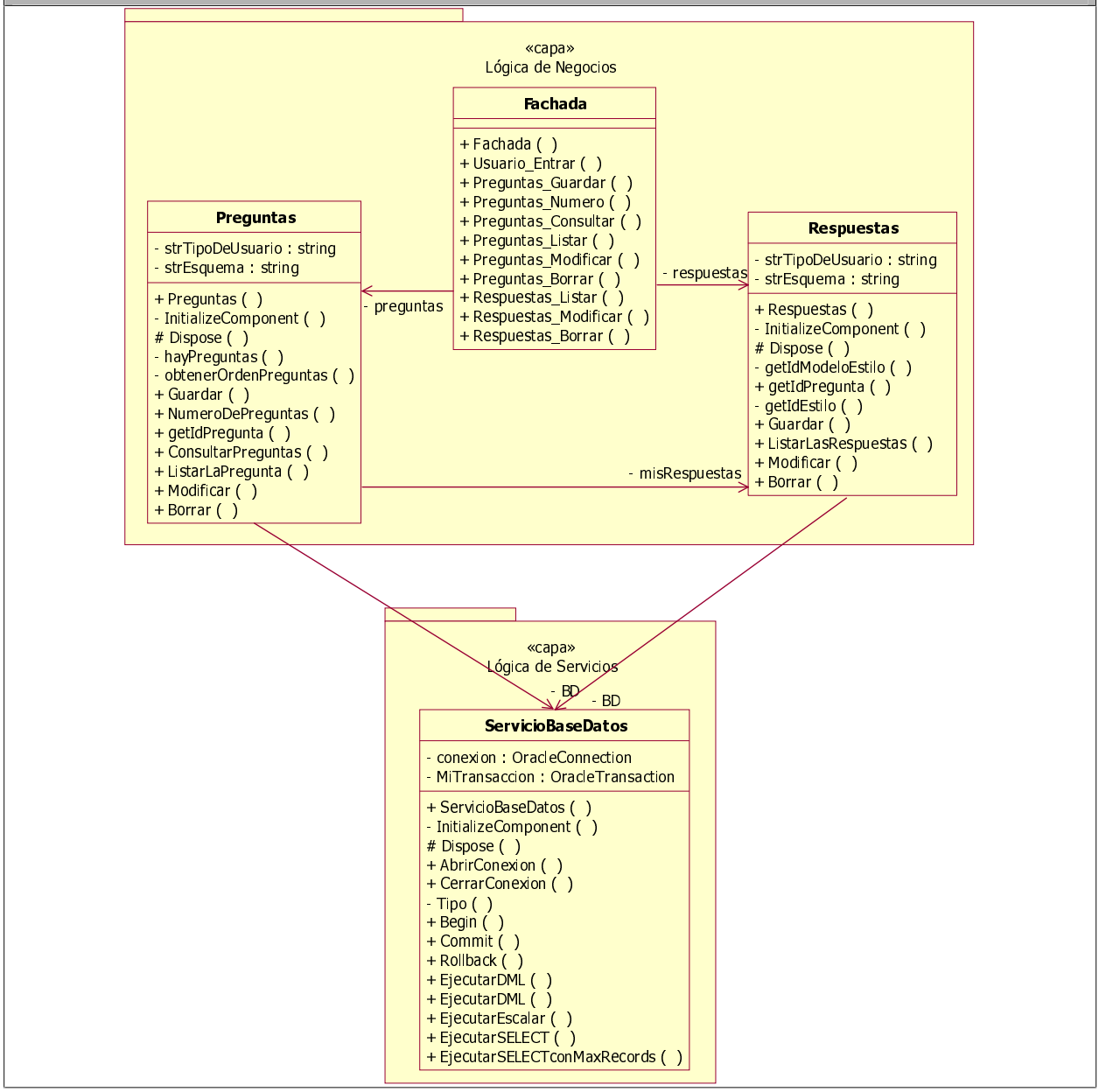


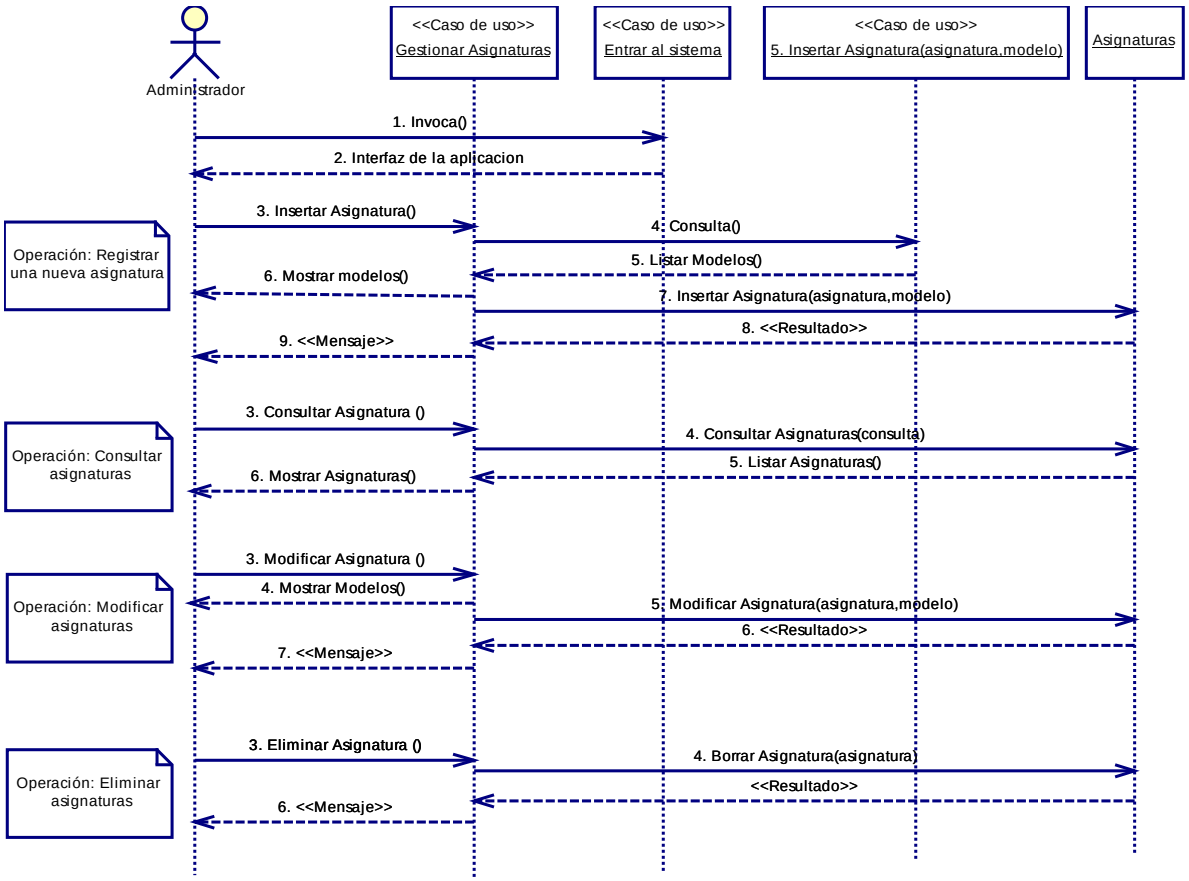
Tabla D.1.1.5. Diseño del caso de uso Gestionar Preguntas y Respuestas

A.1.3 Caso de uso: Gestionar Asignaturas

Este caso de uso presenta la gestión de toda la información asociada a las asignaturas a orientar a los estudiantes en su proceso educativo. Este caso de uso invoca al caso de uso Entrar al sistema.

A.1.3.1 Análisis

FORMATO CASO DE USO GESTIONAR ASIGNATURAS	
Actores	
• Administrador (Iniciador)	
Este caso de uso comienza cuando el administrador invoca al caso de uso Entrar al sistema, con el fin de gestionar (registrar, consultar, modificar o eliminar) asignaturas que se orientaran en el proceso educativo. El sistema solicita la información relacionada con las asignaturas y realiza el respectivo proceso dependiendo de la operación que desee el administrador (registrar, consultar, modificar o eliminar), luego envía un mensaje al administrador del resultado de la operación efectuada. Este caso de uso finaliza cuando el administrador: abandona el proceso, cierra la sesión o cierra la aplicación.	
Acción del Actor	**Respuesta del Sistema**
1. El administrador decide gestionar (registrar, consultar, modificar o eliminar) asignaturas.	**2**. El sistema redirecciona al administrador a la interfaz de conexión.
3. El administrador utiliza el caso de uso **Entrar al sistema**.	4. Si Entrar al sistema es exitoso, el sistema presenta la interfaz principal de la aplicación.
5. El administrador ingresa la información de la asignatura dependiendo de la operación de gestión (registrar, consultar, modificar o eliminar) seleccionada.	**6**. El sistema realiza el respectivo proceso dependiendo de la operación que seleccionó el administrador, y le informa el resultado de la misma.
DIAGRAMA DE SECUENCIA DEL SISTEMA DEL CASO DE USO GESTIONAR ASIGNATURAS	
Este diagrama muestra todo el proceso necesario que debe hacer el actor administrador para realizar el caso de uso Gestionar Asignaturas. El administrador gestiona (registra, consulta, modifica o elimina) una asignaturas, luego de haber realizado los casos de uso necesarios. El sistema le envía un mensaje de información de acuerdo a la operación efectuada.	



MODELO CONCEPTUAL DEL CASO DE USO GESTIONAR ASIGNATURAS
En el modelo conceptual, se destacan los conceptos más importantes del caso de uso Gestionar Asignaturas. El objeto persistente o entidad que usa el caso de uso es: Asignaturas. El control de este caso de uso está representado por el objeto Gestionar Asignaturas, y la interfaz para la interacción con el actor Administrador es el objeto del mismo nombre. El control realiza en la entidad Asignaturas el proceso de gestión dependiendo de la operación seleccionada por el administrador.

FORMATO CASO DE USO GESTIONAR ASIGNATURAS

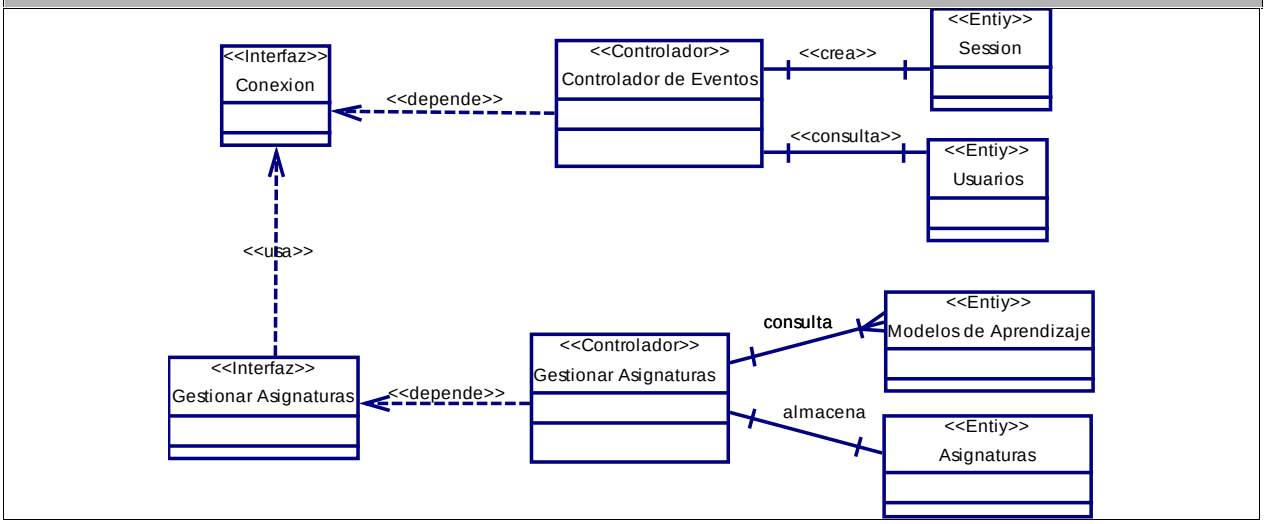


Tabla A.1.1.8 Análisis del caso de uso Gestionar Asignaturas

A.1.3.2 Diseño

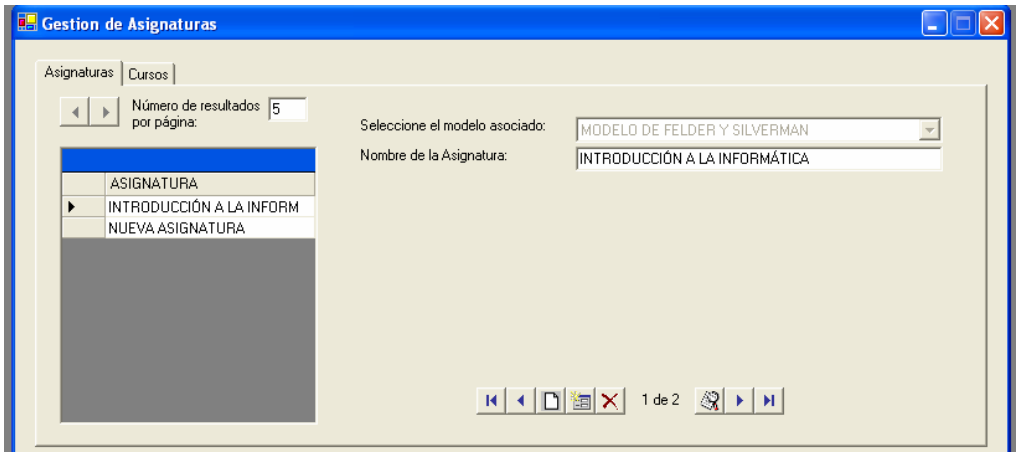


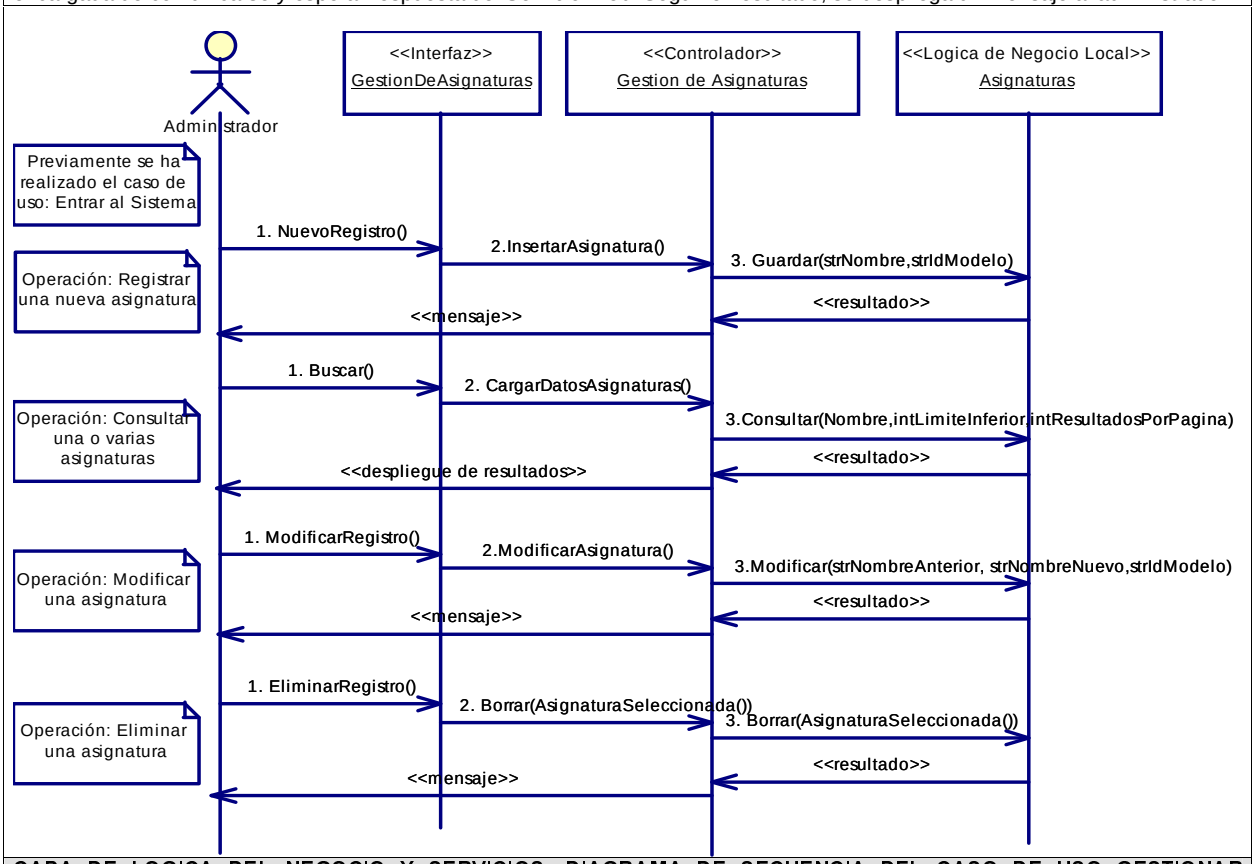
Figura 6. Screen Shot del caso de uso Gestionar Asignaturas

FORMATO CASO DE USO: GESTIONAR ASIGNATURAS	
Actores	
Administrador (Iniciador)	
Este caso de uso comienza cuando el administrador decide Gestionar Asignaturas. El sistema solicita primero al administrador que se loguee, invocando el caso de uso Entrar al Sistema. Después de este proceso el sistema presenta al administrador la interfaz principal, donde el administrador elige la opción "Asignaturas", que presenta la interfaz que gestiona las asignaturas a orientar, a través de las cuatro operaciones básicas (registrar, consultar, modificar y eliminar). Dependiendo de la operación que realice el administrador, el controlador envía los datos de la asignaturas a la clase Asignaturas de Lógica de Negocios Local, la cual se comunica y propaga los datos a través de la referencia Proxy del servicio Web Fachada, quien se comunica con el servicio Web Asignaturas de la lógica de Negocios, el cual gestiona la asignatura en la base de datos, ayudado del servicio Web Lógica de Servicios. Finalmente el sistema, envía un mensaje al administrador sobre el éxito de la operación. Este caso de uso finaliza cuando el administrador: abandona el proceso, cierra la sesión o cierra la aplicación.	
Acción del Actor	**Respuesta del Sistema**
1. El administrador decide gestionar asignaturas.	2. El sistema redirecciona al administrador a la interfaz de Conexión.

FORMATO CASO DE USO: GESTIONAR ASIGNATURAS	
3. El administrador utiliza el caso de uso Entrar al Sistema.	4. Si Entrar al sistema es exitoso, el sistema presenta la interfaz principal de la aplicación.
5. El administrador hace clic en alguno de los botones (adicionar, consultar, modificar o eliminar) que permiten la gestión de las asignaturas.	6. El sistema envía los datos de la asignatura y ejecuta el proceso de gestión de acuerdo a la operación efectuada, también retorna un mensaje del resultado de dicha operación.

CAPA DE LOGICA DE PRESENTACION: DIAGRAMA DE SECUENCIA DEL CASO DE USO GESTIONAR ASIGNATURAS

El administrador hace clic en alguno de los botones (adicionar, consultar, modificar o eliminar) de la interfaz Gestión de Asignaturas. Una vez que el administrador realiza la operación de su elección para gestionar las asignaturas, esta información la envía el controlador a la clase Asignaturas a través de los métodos definidos para cada operación. Esta clase también está encargada de comunicarse y esperar respuesta del Servicio Web. Según el resultado, se despliega un mensaje al administrador.



CAPA DE LOGICA DEL NEGOCIO Y SERVICIOS: DIAGRAMA DE SECUENCIA DEL CASO DE USO GESTIONAR ASIGNATURAS

El servicio Web Fachada, recibe desde la capa de presentación la información de la asignatura, este servicio Web delega al servicio Web Asignaturas el proceso de gestión de la asignatura de acuerdo a la operación efectuada por el administrador, utilizando el patrón experto. El servicio Web Asignaturas procesa los datos e invoca dependiendo de la operación de gestión efectuada al método respectivo de la clase de lógica Servicios base de datos, que se encarga de ejecutar el proceso de gestión contra la base de datos. Luego se informa del éxito o fracaso de dicho proceso a través del servicio Web hasta llegar a la capa de presentación.

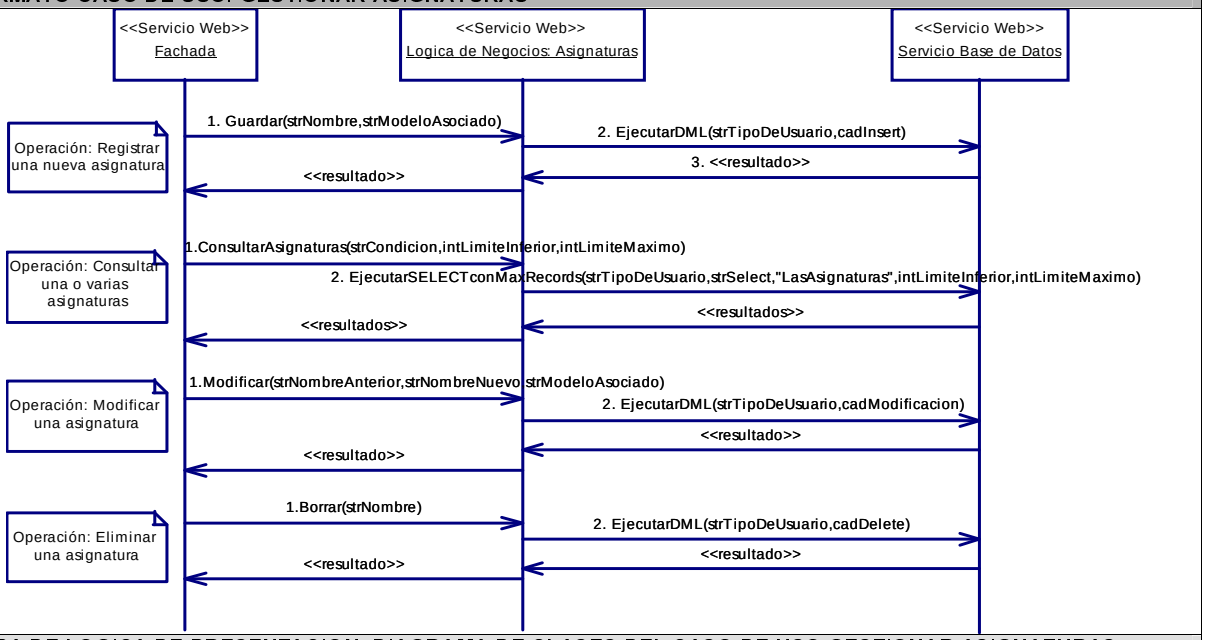
FORMATO CASO DE USO: GESTIONAR ASIGNATURAS

CAPA DE LOGICA DE PRESENTACION: DIAGRAMA DE CLASES DEL CASO DE USO GESTIONAR ASIGNATURAS
La interfaz Gestionar Asignaturas se relaciona con el controlador a través de su código subyacente, que se relaciona con la lógica de negocios local a través de la clase Asignaturas, que se relaciona con el Proxy de la Fachada, dado que Gestionar Asignaturas utiliza métodos del Servicio Web de la aplicación.
CAPA DE LOGICA DEL NEGOCIO Y SERVICIOS: DIAGRAMA DE CLASES DEL CASO DE USO GESTIONAR ASIGNATURAS
Fachada instancia un objeto de Asignaturas para delegarle el proceso de gestión, existiendo una asociación directa entre ellas. El objeto Asignaturas utiliza los métodos adecuados de la clase de Servicios de la base de datos de acuerdo a la operación de gestión, por lo que existe una asociación directa entre estas.

Tabla D.1.1.6. Diseño del caso de uso Gestionar Asignaturas

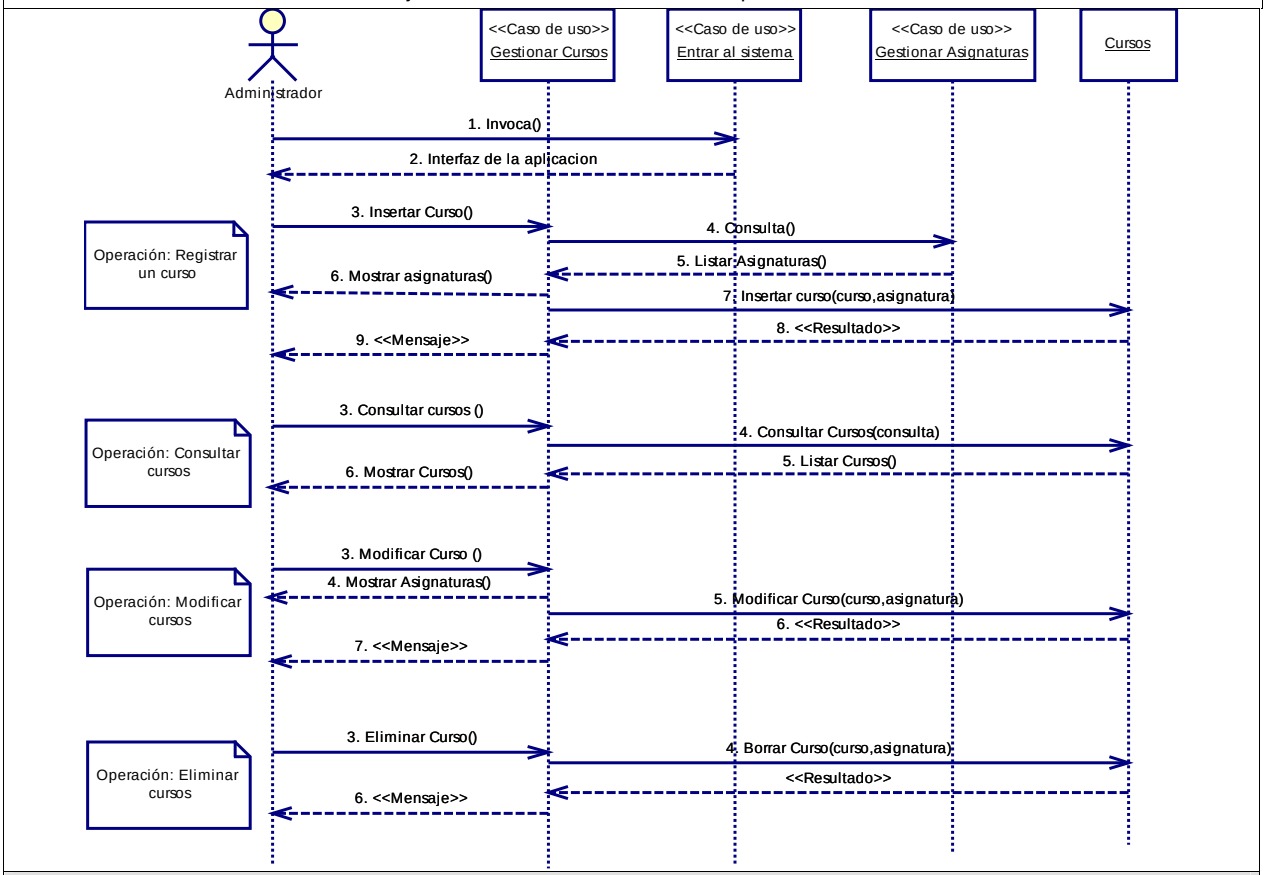
A.1.4 Caso de uso: Gestionar Cursos

Este caso de uso presenta la gestión de toda la información asociada a los cursos de las asignaturas a orientar. Este caso de uso invoca al caso de uso Entrar al sistema.

A.1.4.1 Análisis

FORMATO CASO DE USO GESTIONAR CURSOS	
Actores	
• Administrador (Iniciador)	
Este caso de uso comienza cuando el administrador invoca al caso de uso Entrar al sistema, con el fin de gestionar (registrar, consultar, modificar o eliminar) cursos asociados a las asignaturas que se orientaran en el proceso educativo. El sistema solicita la información relacionada con los cursos y realiza el respectivo proceso dependiendo de la operación que desee el administrador (registrar, consultar, modificar o eliminar), luego envía un mensaje al administrador del resultado de la operación efectuada. Este caso de uso finaliza cuando el administrador: abandona el proceso, cierra la sesión o cierra la aplicación.	
Acción del Actor	**Respuesta del Sistema**
1. El administrador decide gestionar (registrar, consultar, modificar o eliminar) cursos.	**2**. El sistema redirecciona al administrador a la interfaz de conexión.
3. El administrador utiliza el caso de uso **Entrar al sistema**.	4. Si Entrar al sistema es exitoso, el sistema presenta la interfaz principal de la aplicación.
5. El administrador ingresa la información del curso dependiendo de la operación de gestión (registrar, consultar, modificar o eliminar) seleccionada.	**6**. El sistema realiza el respectivo proceso dependiendo de la operación que seleccionó el administrador, y le informa el resultado de la misma.

FORMATO CASO DE USO GESTIONAR CURSOS
DIAGRAMA DE SECUENCIA DEL SISTEMA DEL CASO DE USO GESTIONAR CURSOS
Este diagrama muestra todo el proceso necesario que debe hacer el actor administrador para realizar el caso de uso Gestionar Cursos. El administrador gestiona (registra, consulta, modifica o elimina) un curso, luego de haber realizado los casos de uso necesarios. El sistema le envía un mensaje de información de acuerdo a la operación efectuada.



MODELO CONCEPTUAL DEL CASO DE USO GESTIONAR ASIGNATURAS
En el modelo conceptual, se destacan los conceptos más importantes del caso de uso Gestionar Cursos. El objeto persistente o entidad que usa el caso de uso es: Cursos. El control de este caso de uso está representado por el objeto Gestión de Asignaturas, y la interfaz para la interacción con el actor Administrador es el objeto del mismo nombre. El control realiza en la entidad Cursos el proceso de gestión dependiendo de la operación seleccionada por el administrador.

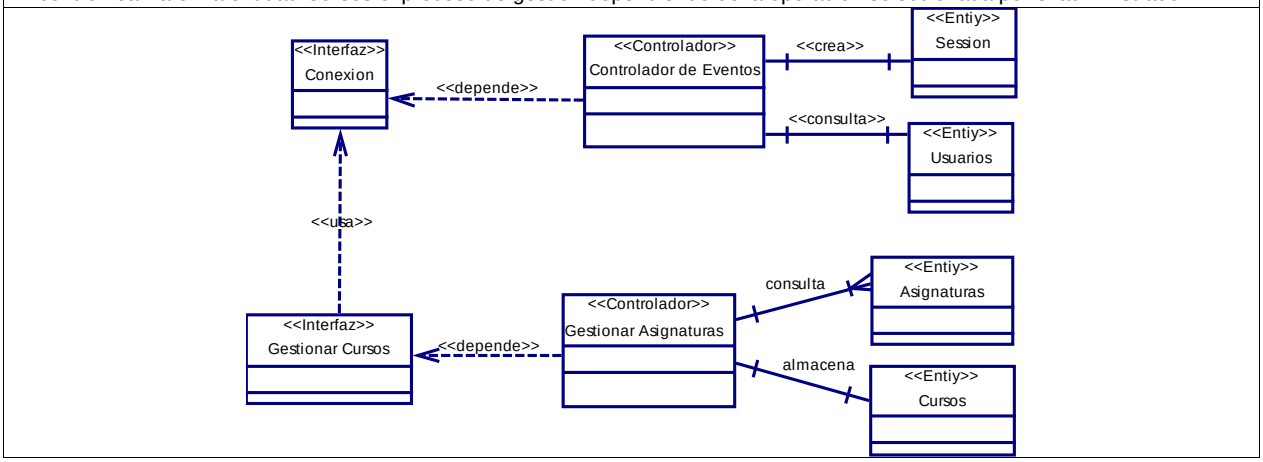


Tabla A.1.1.9 Análisis del caso de uso Gestionar Cursos

A.1.4.2 Diseño

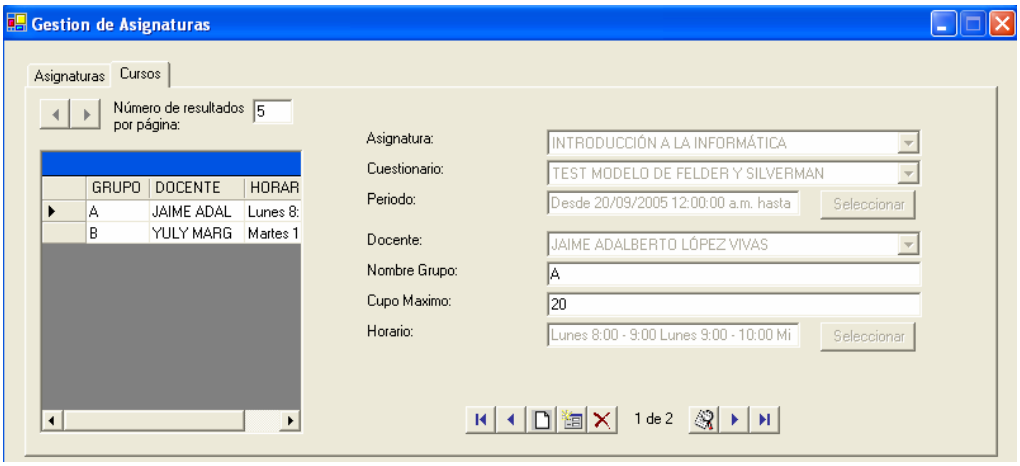
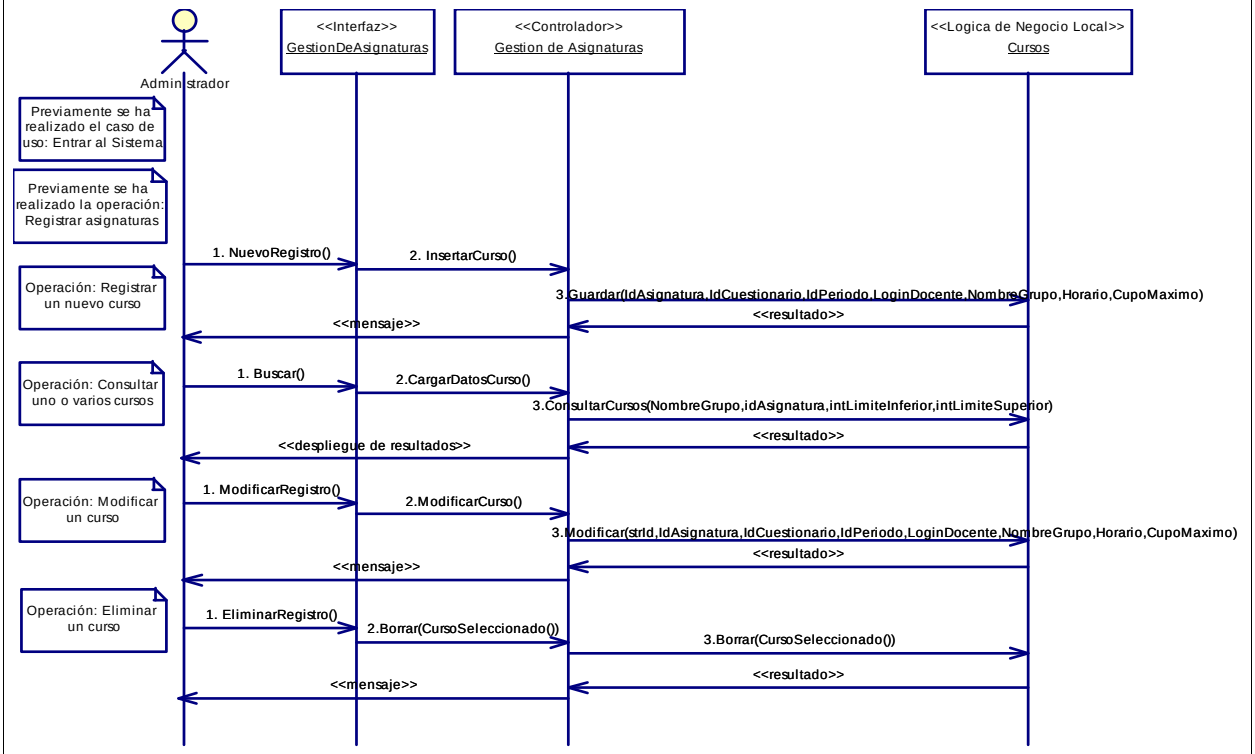


Figura 7. Screen Shot del caso de uso Gestionar Cursos

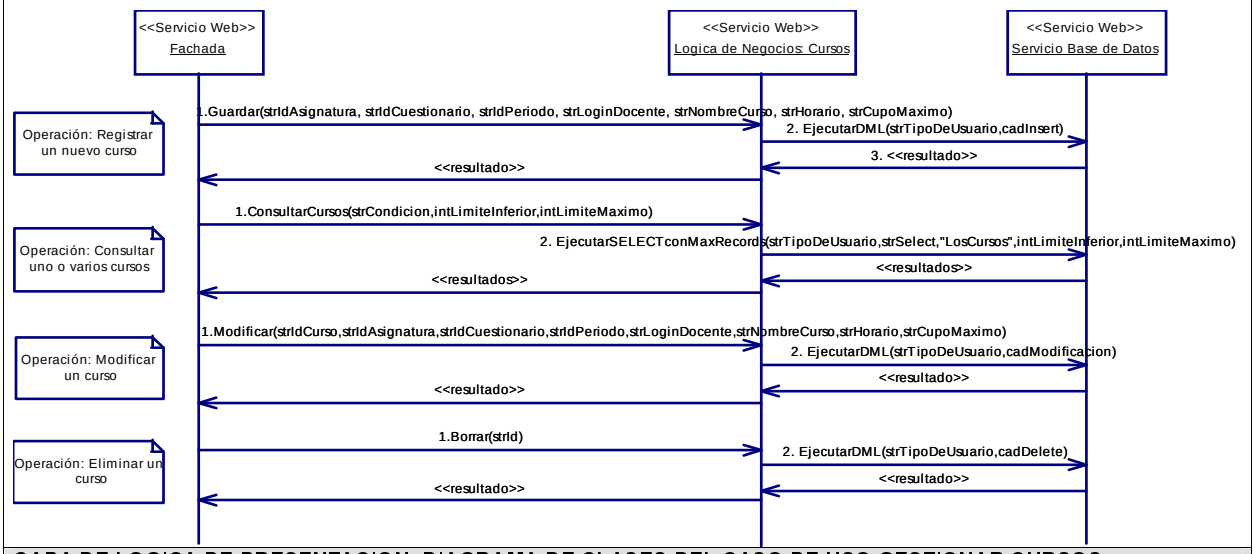
FORMATO CASO DE USO: GESTIONAR CURSOS	
Actores	
Administrador (Iniciador)	
Este caso de uso comienza cuando el administrador decide Gestionar los cursos de las asignaturas. El sistema solicita primero al administrador que se loguee, invocando el caso de uso Entrar al Sistema. Después de este proceso el sistema presenta al administrador la interfaz principal, donde el administrador elige la opción "Cursos", que presenta la interfaz que gestiona los cursos de las asignaturas a orientar, a través de las cuatro operaciones básicas (registrar, consultar, modificar y eliminar). Dependiendo de la operación que realice el administrador, el controlador envía los datos de los cursos a la clase Cursos de Lógica de Negocios Local, la cual se comunica y propaga los datos a través de la referencia Proxy del servicio Web Fachada, quien se comunica con el servicio Web Cursos de la lógica de Negocios, el cual gestiona los cursos en la base de datos, ayudado del servicio Web Lógica de Servicios. Finalmente el sistema, envía un mensaje al administrador sobre el éxito de la operación. Este caso de uso finaliza cuando el administrador: abandona el proceso, cierra la sesión o cierra la aplicación.	
Acción del Actor	**Respuesta del Sistema**
1. El administrador decide gestionar cursos.	2. El sistema redirecciona al administrador a la interfaz de Conexión.
3. El administrador utiliza el caso de uso Entrar al Sistema.	4. Si Entrar al sistema es exitoso, el sistema presenta la interfaz principal de la aplicación.
5. El administrador hace clic en alguno de los botones (adicionar, consultar, modificar o eliminar) que permiten la gestión de los cursos.	6. El sistema envía los datos del curso y ejecuta el proceso de gestión de acuerdo a la operación efectuada, también retorna un mensaje del resultado de dicha operación.
CAPA DE LOGICA DE PRESENTACION: DIAGRAMA DE SECUENCIA DEL CASO DE USO GESTIONAR CURSOS	
El administrador hace clic en alguno de los botones (adicionar, consultar, modificar o eliminar) de la interfaz que gestiona los cursos. Una vez que el administrador realiza la operación de su elección para gestionar los cursos, esta información la envía el controlador a la clase Cursos a través de los métodos definidos para cada operación. Esta clase también está encargada de comunicarse y esperar respuesta del Servicio Web. Según el resultado, se despliega un mensaje al administrador.	

FORMATO CASO DE USO: GESTIONAR CURSOS



CAPA DE LOGICA DEL NEGOCIO Y SERVICIOS: DIAGRAMA DE SECUENCIA DEL CASO DE USO GESTIONAR CURSOS

El servicio Web Fachada, recibe desde la capa de presentación la información del curso, este servicio Web delega al servicio Web Cursos el proceso de gestión del curso de acuerdo a la operación efectuada por el administrador, utilizando el patrón experto. El servicio Web Cursos procesa los datos e invoca dependiendo de la operación de gestión efectuada al método respectivo de la clase de lógica Servicios base de datos, que se encarga de ejecutar el proceso de gestión contra la base de datos. Luego se informa del éxito o fracaso de dicho proceso a través del servicio Web hasta llegar a la capa de presentación.



CAPA DE LOGICA DE PRESENTACION: DIAGRAMA DE CLASES DEL CASO DE USO GESTIONAR CURSOS

La interfaz que gestiona los cursos se relaciona con el controlador “Gestionar Asignaturas” a través de su código subyacente, que se relaciona con la lógica de negocios local a través de la clase Cursos, que se relaciona con el Proxy de la Fachada, dado que Gestionar Cursos utiliza métodos del Servicio Web de la aplicación.

FORMATO CASO DE USO: GESTIONAR CURSOS
CAPA DE LOGICA DEL NEGOCIO Y SERVICIOS: DIAGRAMA DE CLASES DEL CASO DE USO GESTIONAR CURSOS
Fachada instancia un objeto de Cursos para delegarle el proceso de gestión, existiendo una asociación directa entre ellas. El objeto Cursos utiliza los métodos adecuados de la clase de Servicios de la base de datos de acuerdo a la operación de gestión, por lo que existe una asociación directa entre estas.

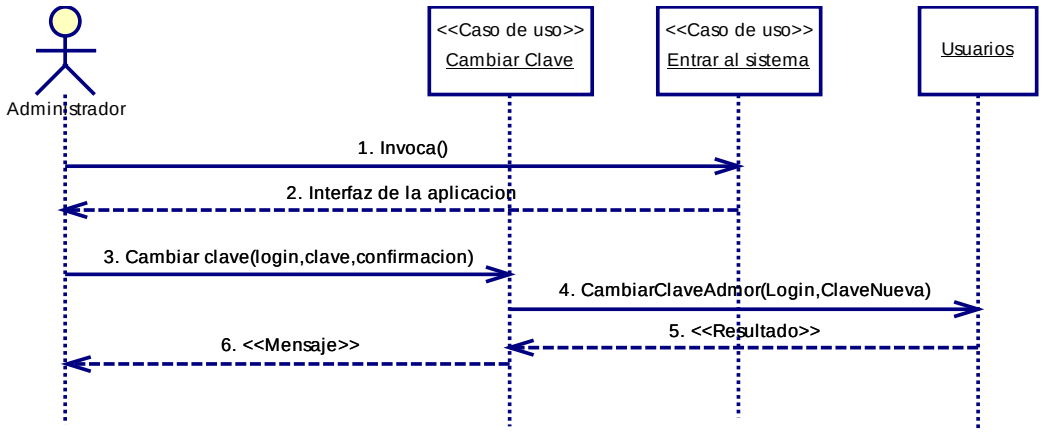
Tabla D.1.1.7. Diseño del caso de uso Gestionar Cursos

A.1.5 Caso de uso: Cambiar Clave

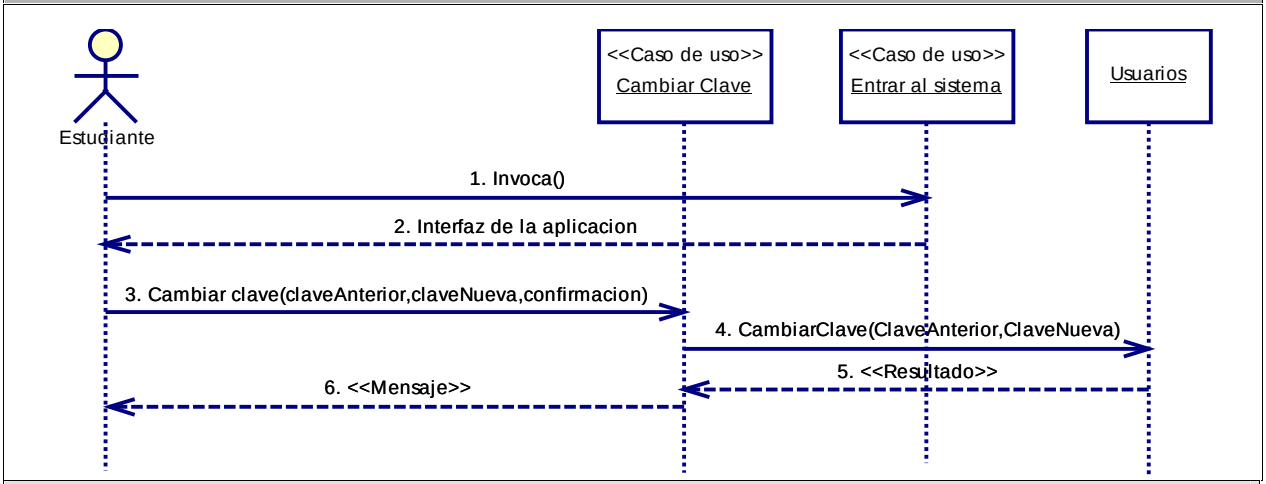
Este caso de uso presenta el proceso de cambio de clave de acceso efectuada por los actores: administrador y estudiante. El administrador efectúa cambio de clave de los usuarios registrados en el sistema, el estudiante efectúa cambio de clave de su clave de acceso personal. Este caso de uso invoca al caso de uso Entrar al sistema.

A.1.5.1 Análisis

FORMATO CASO DE USO CAMBIAR CLAVE	
Actores	
• Administrador (Iniciador), Estudiante(Iniciador)	
Este caso de uso lo utilizan los actores Administrador y Estudiante con un enfoque de uso diferente: Cuando el caso de uso es utilizado por el administrador, previamente se invoca al caso de uso Entrar al sistema, luego de esto se presenta la interfaz que permite cambiar la clave de los usuarios registrados en el sistema, en la que se solicita el login del usuario al cual se efectuará el proceso de modificación. Cuando el caso de uso es utilizado por el Estudiante, previamente se invoca al caso de uso Entrar al sistema, luego de esto se presenta la interfaz que permite cambiar su clave personal, en la que se solicita la clave anterior y el ingreso y confirmación de la nueva clave, con las que se efectuará el proceso de modificación. Este caso de uso finaliza cuando cada actor: abandona el proceso, cierra la sesión o cierra la aplicación.	
Acción del Actor	**Respuesta del Sistema**
1. El administrador o estudiante decide cambiar la clave de acceso al sistema.	**2**. El sistema redirecciona al administrador o estudiante a la interfaz de conexión.
3. El administrador o estudiante utiliza el caso de uso **Entrar al sistema**.	4. Si Entrar al sistema es exitoso, el sistema presenta la interfaz principal de la aplicación.
5. El administrador o estudiante ingresa la información requerida para el cambio de la clave.	**6**. El sistema realiza el respectivo proceso para cambiar la clave dependiendo del actor que efectúe dicha operación, también informa el resultado del mismo.
DIAGRAMA DE SECUENCIA DEL SISTEMA DEL CASO DE USO CAMBIAR CLAVE	
Este diagrama muestra todo el proceso necesario que debe hacer el actor administrador o estudiante para realizar el caso de uso Cambiar Clave. El administrador o estudiante gestiona el cambio de clave, luego de haber realizado los casos de uso necesarios. El sistema le envía un mensaje de información de acuerdo a la operación efectuada.	



FORMATO CASO DE USO CAMBIAR CLAVE



MODELO CONCEPTUAL DEL CASO DE USO CAMBIAR CLAVE

En el modelo conceptual, se destacan los conceptos más importantes del caso de uso Cambiar Clave. El objeto persistente o entidad que usa el caso de uso es Usuarios. El control de este caso de uso está representado por el objeto Cambiar Clave Administrador cuando el actor es el administrador y Cambiar Clave cuando el actor es el estudiante, y las interfaces para la interacción con los actores son los objetos del mismo nombre respectivamente.
El control realiza en la entidad usuarios el proceso de cambio de clave dependiendo del actor que la efectúe.

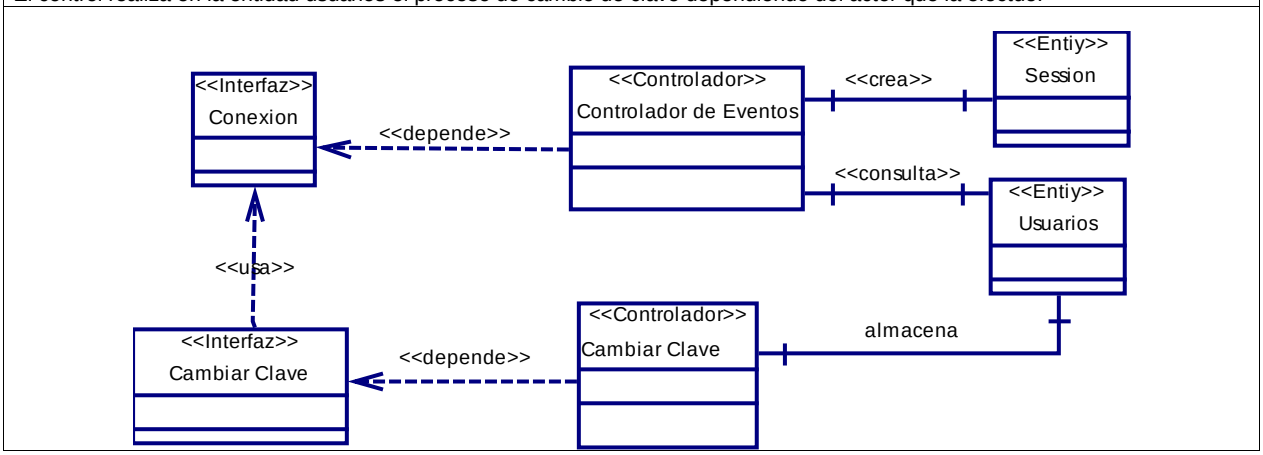


Tabla A.1.1.10 Análisis del caso de uso Cambiar Clave

A.1.5.2 Diseño

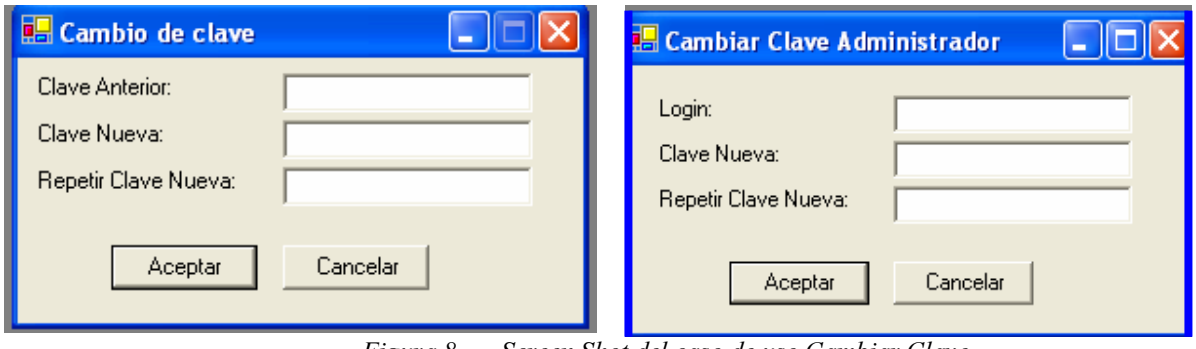
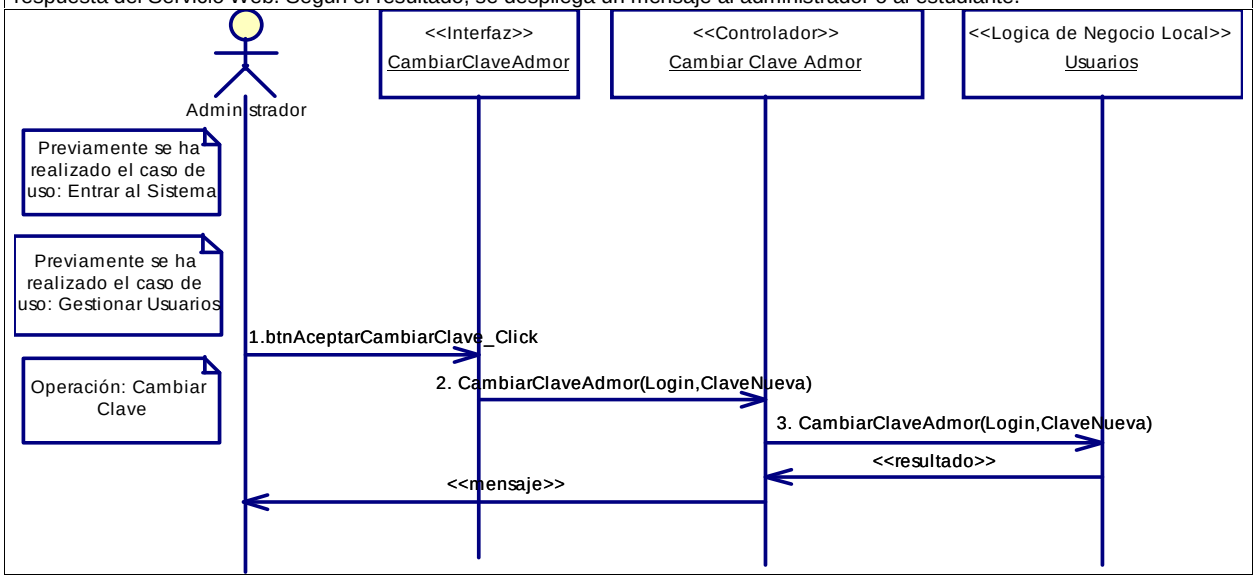
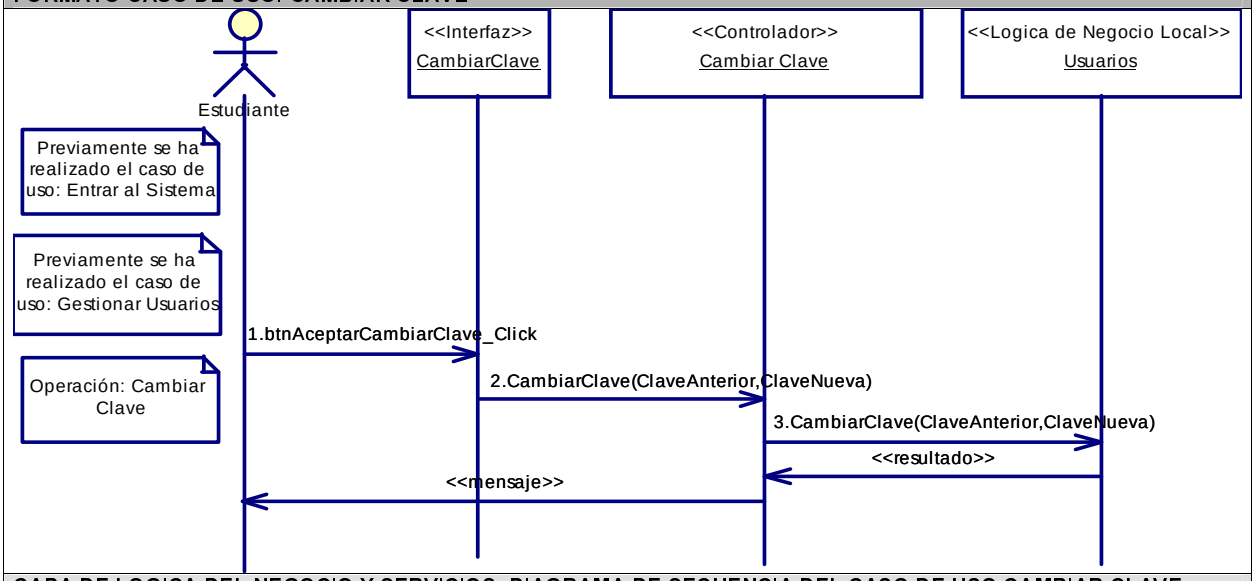


Figura 8. Screen Shot del caso de uso Cambiar Clave

FORMATO CASO DE USO: CAMBIAR CLAVE	
Actores	
Administrador (Iniciador), Estudiante (Iniciador)	
Este caso de uso comienza cuando el actor: administrador o Estudiante decide cambiar la clave de acceso al sistema. El sistema solicita primero a cada actor que se loguee, invocando el caso de uso Entrar al Sistema. Si el actor es el Administrador el sistema presenta la interfaz principal, donde éste elige la opción "Cambiar Clave", que presenta la interfaz que gestiona el cambio de clave, para lo cual ingresa el login del usuario, su nueva clave y confirmación. Si el actor es el Estudiante el sistema presenta la interfaz principal, donde éste elige la opción "Cambiar Clave", que presenta la interfaz que gestiona el cambio de clave, para lo cual ingresa la clave anterior, su nueva clave y confirmación. Dependiendo de la operación que realice cada actor, el controlador envía los datos del cambio de clave a la clase Usuarios de Lógica de Negocios Local, la cual se comunica y propaga los datos a través de la referencia Proxy del servicio Web Fachada, quien se comunica con el servicio Web Usuarios de la lógica de Negocios, el cual gestiona el cambio de clave en la base de datos, ayudado del servicio Web Lógica de Servicios. Finalmente el sistema, envía un mensaje a cada actor sobre el éxito de la operación. Este caso de uso finaliza cuando cada actor: abandona el proceso, cierra la sesión o cierra la aplicación.	
Acción del Actor	**Respuesta del Sistema**
1. El administrador o estudiante decide cambiar la clave de acceso al sistema.	2. El sistema redirecciona a cada actor a la interfaz de Conexión.
3. El administrador o Estudiante utiliza el caso de uso Entrar al Sistema.	4. Si Entrar al sistema es exitoso, el sistema presenta la interfaz principal de la aplicación.
5. El administrador o Estudiante realiza el cambio de clave de acceso al sistema.	6. El sistema envía los datos del cambio de clave y ejecuta el proceso de modificación, también retorna un mensaje del resultado de dicha operación.
CAPA DE LOGICA DE PRESENTACION: DIAGRAMA DE SECUENCIA DEL CASO DE USO CAMBIAR CLAVE	
El administrador o estudiante ingresa la información para el cambio de clave en la interfaz correspondiente a cada actor. Una vez que el administrador o estudiante realiza la modificación de la clave de acceso, esta información la envía el controlador a la clase Usuarios a través de los métodos definidos para cada operación. Esta clase también está encargada de comunicarse y esperar respuesta del Servicio Web. Según el resultado, se despliega un mensaje al administrador o al estudiante.	

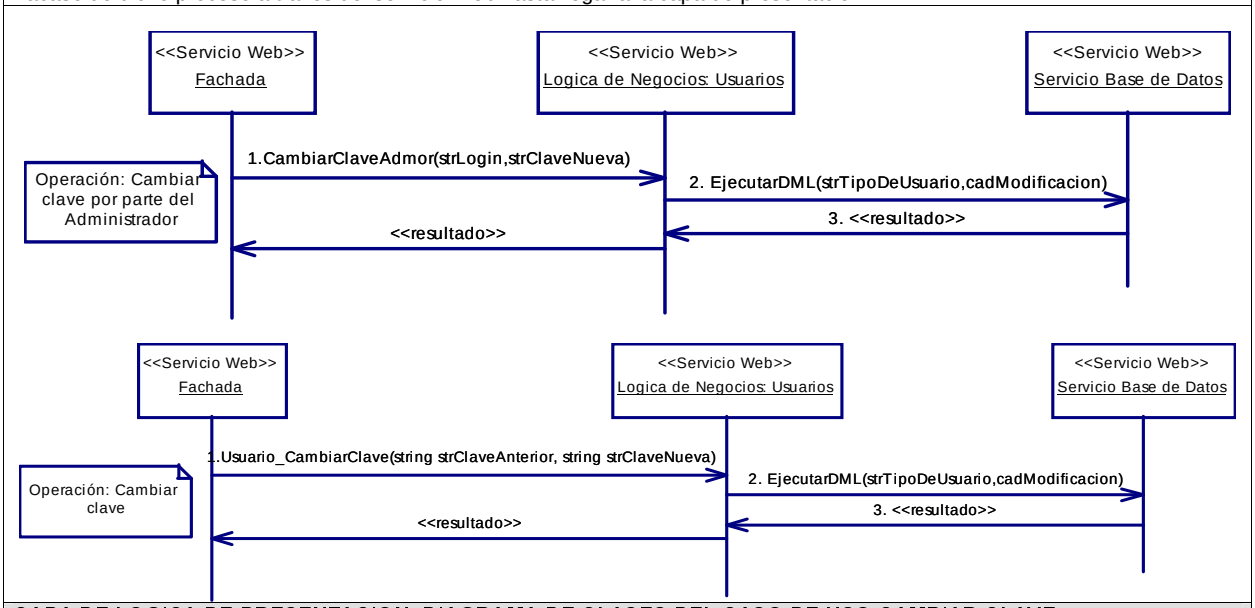


FORMATO CASO DE USO: CAMBIAR CLAVE



CAPA DE LOGICA DEL NEGOCIO Y SERVICIOS: DIAGRAMA DE SECUENCIA DEL CASO DE USO CAMBIAR CLAVE

El servicio Web Fachada, recibe desde la capa de presentación la información del cambio de clave, este servicio Web delega al servicio Web Usuarios el proceso de modificación de la clave de acceso dependiendo del actor que efectúe la operación, utilizando el patrón experto. El servicio Web Usuarios procesa los datos e invoca al método respectivo de la clase de lógica Servicios base de datos, que se encarga de ejecutar el proceso de gestión contra la base de datos. Luego se informa del éxito o fracaso de dicho proceso a través del servicio Web hasta llegar a la capa de presentación.



CAPA DE LOGICA DE PRESENTACION: DIAGRAMA DE CLASES DEL CASO DE USO CAMBIAR CLAVE

La interfaz que gestiona el cambio de clave se relaciona con el controlador "Cambio de Clave" a través de su código subyacente, que se relaciona con la lógica de negocios local a través de la clase Usuarios, que se relaciona con el Proxy de la Fachada, dado que Cambiar Clave utiliza métodos del Servicio Web de la aplicación. (La interfaz de la capa de presentación de este caso de uso es la misma que la del caso de uso Gestionar Usuarios).

CAPA DE LOGICA DEL NEGOCIO Y SERVICIOS: DIAGRAMA DE CLASES DEL CASO DE USO CAMBIAR CLAVE

Fachada instancia un objeto de Usuarios para delegarle el proceso de cambio de clave, existiendo una asociación directa entre ellas. El objeto Usuarios utiliza los métodos adecuados de la clase de Servicios de la base de datos, por lo que existe una asociación directa entre estas. (La interfaz de la capa de Negocio y Servicios de este caso de uso es la misma que la del caso de uso Gestionar Usuarios).

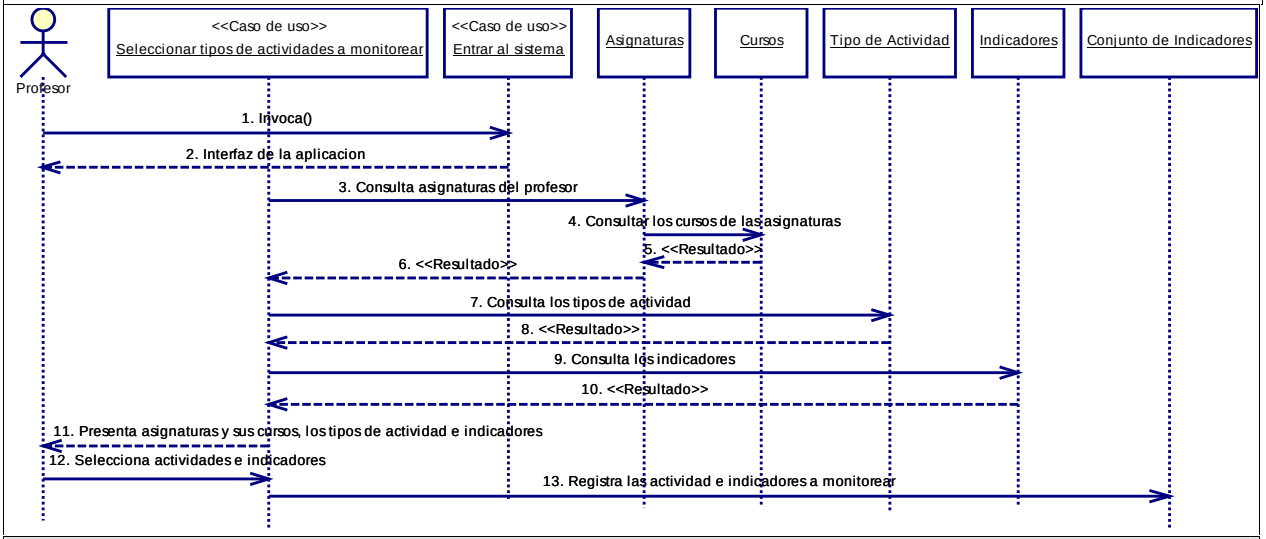
Tabla D.1.1.8. Diseño del caso de uso Cambiar Clave

A.1.6 Caso de uso: Seleccionar tipos de actividad a monitorear

A.1.6.1 Análisis

Este caso de uso le permite al profesor seleccionar los tipos de actividad a monitorear en el contenido del curso presentado al estudiante, dicha selección se realiza a través de la asociación del tipo de actividad con un conjunto de indicadores predefinidos en la herramienta. El análisis de este caso de uso es el siguiente:

FORMATO CASO DE USO SELECCIONAR TIPOS DE ACTIVIDADES A MONITOREAR	
Actores	
• Profesor (Iniciador)	
Este caso de uso comienza cuando el profesor invoca al caso de uso Entrar al sistema, con el fin de seleccionar los tipos de actividades a monitorear en la estructura de contenidos del curso que orienta. El sistema presenta solo las asignaturas y cursos que orienta el profesor y los tipos de actividad e indicadores predefinidos. Este caso de uso finaliza cuando el profesor: abandona el proceso, cierra la sesión o cierra la aplicación.	
Acción del Actor	**Respuesta del Sistema**
1. El profesor decide seleccionar los tipos de actividad a monitorear en la estructura de contenidos del curso a orientar.	**2**. El sistema redirecciona al profesor a la interfaz de conexión.
3. El profesor utiliza el caso de uso **Entrar al sistema**.	4. Si Entrar al sistema es exitoso, el sistema presenta la interfaz principal de la aplicación.
	5. El sistema visualiza solo las asignaturas y cursos que están a cargo del profesor, y también los tipos de actividad y el conjunto de indicadores previamente definidos.
6. El profesor elige la asignatura y curso al cual establecerá los tipos de actividad e indicadores ha utilizar en el proceso de monitoreo.	**7**. El sistema almacena dicho proceso.
DIAGRAMA DE SECUENCIA DEL SISTEMA DEL CASO DE USO SELECCIONAR TIPOS DE ACTIVIDADES A MONITOREAR	
Este diagrama muestra todo el proceso necesario que debe hacer el actor profesor para seleccionar y establecer los tipos de actividad e indicadores a monitorear en la estructura de contenidos a orientar.	



MODELO CONCEPTUAL DEL CASO DE USO SELECCIONAR ACTIVIDADES A MONITOREAR
En el modelo conceptual, se destacan los conceptos más importantes del caso de uso Seleccionar actividades a monitorear. El objeto persistente o entidad que usa el caso de uso es: Asignaturas, Cursos, Tipos de Actividad, Indicadores, Conjunto de Indicadores. El control de este caso de uso está representado por el objeto Tipo de Actividad, y la interfaz para la interacción con el actor profesor es el objeto del mismo nombre.

FORMATO CASO DE USO SELECCIONAR TIPOS DE ACTIVIDADES A MONITOREAR

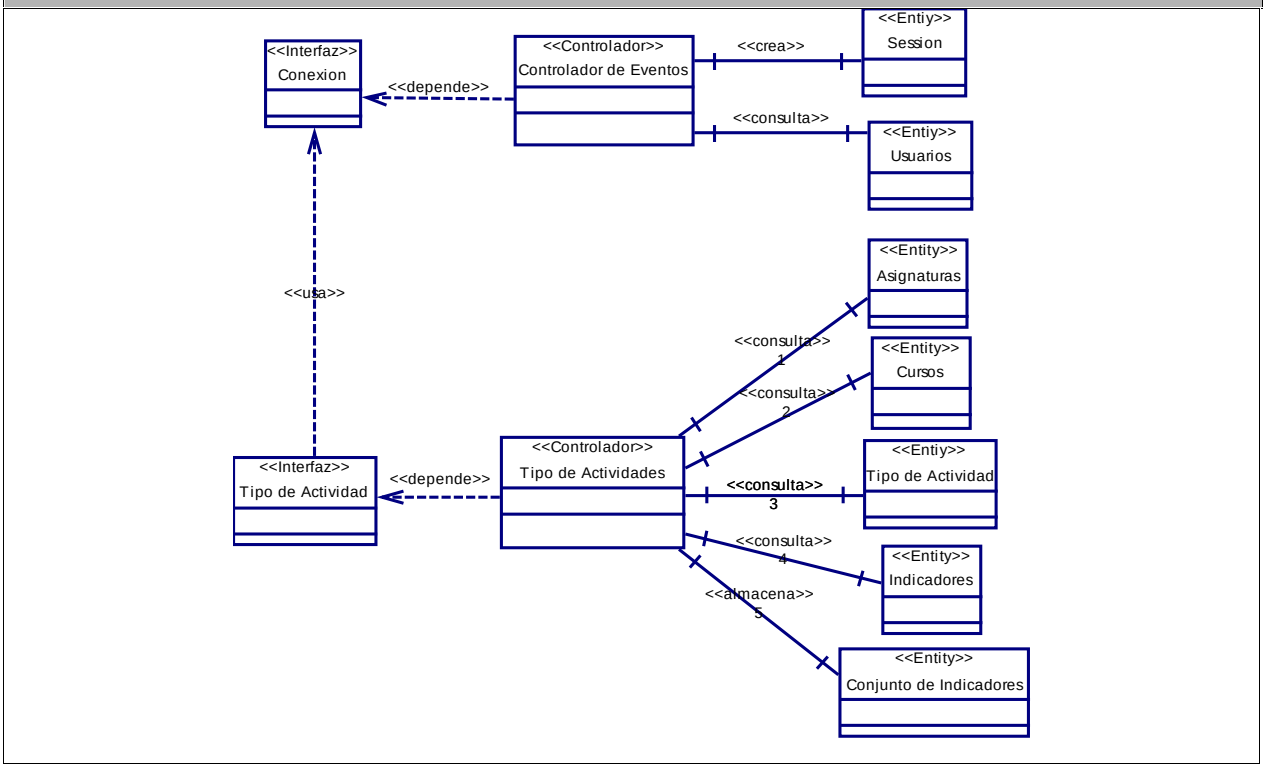


Tabla A.1.1.11 Análisis del caso de uso seleccionar tipos de actividad a monitorear

A.1.6.2 Diseño

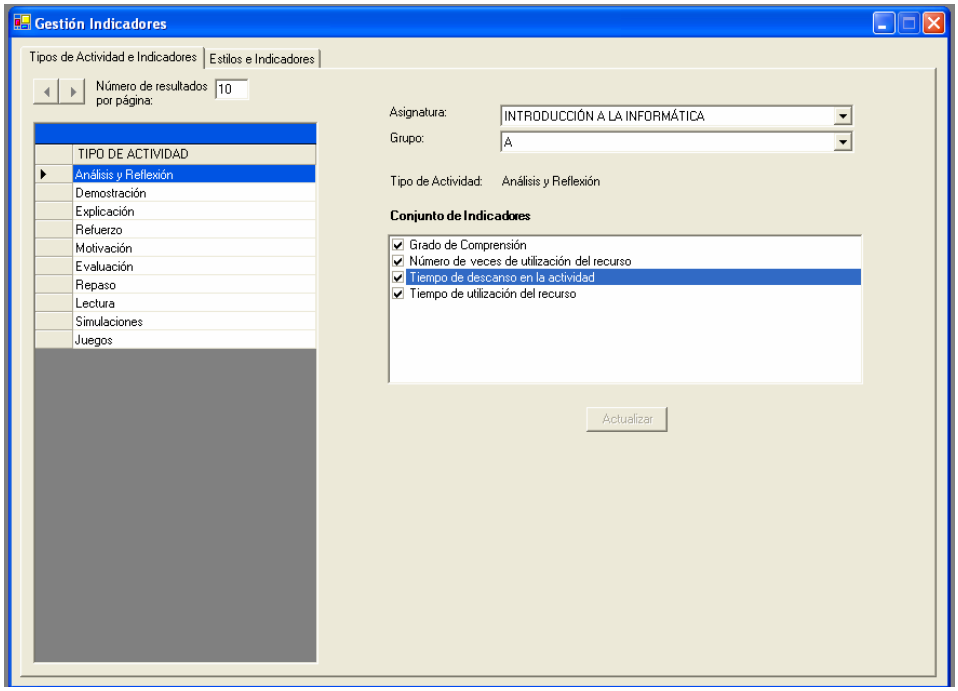
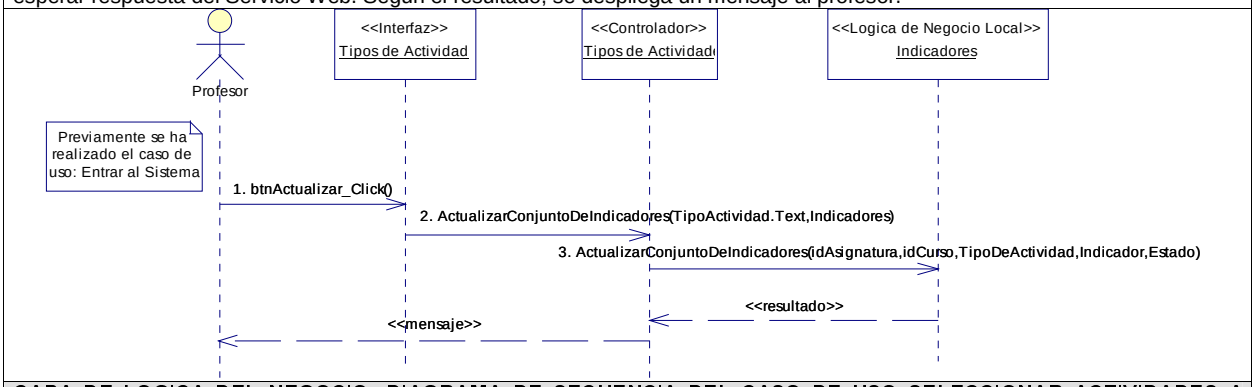


Figura 9. Screen Shot del caso de uso seleccionar tipos de actividad a monitorear

FORMATO CASO DE USO: SELECCIONAR ACTIVIDADES A MONITOREAR	
Actores	
Profesor (Iniciador)	
Este caso de uso comienza cuando el profesor decide Seleccionar las actividades a monitorear. El sistema solicita primero al profesor que se loguee, invocando el caso de uso Entrar al Sistema. Después de este proceso el sistema presenta al profesor la interfaz principal, donde el profesor elige la opción "Indicadores", que presenta la interfaz que gestiona la selección de los tipos de actividad a monitorear a través de un conjunto de indicadores, en alguno de los cursos de las asignaturas orientadas por el profesor. El controlador de la forma envía los datos a la clase Indicadores de Lógica de Negocios Local, la cual se comunica y propaga los datos a través de la referencia Proxy del servicio Web Fachada, quien se comunica con los servicios Web Tipo de Actividad e Indicadores de la lógica de Negocios, los cuales gestionan los tipos de actividad selecciona y su conjunto de indicadores en la base de datos respectivamente, ayudado del servicio Web Lógica de Servicios. Finalmente el sistema, envía un mensaje al profesor sobre el éxito de la operación. Este caso de uso finaliza cuando el profesor: abandona el proceso, cierra la sesión o cierra la aplicación.	
Acción del Actor	**Respuesta del Sistema**
1. El profesor decide seleccionar los tipos de actividad y conjunto de indicadores del proceso de monitoreo.	2. El sistema redirecciona al profesor a la interfaz de Conexión.
3. El profesor utiliza el caso de uso Entrar al Sistema.	4. Si Entrar al sistema es exitoso, el sistema presenta la interfaz principal de la aplicación.
5. El profesor hace clic en "Indicadores" del menú Planificación, que permiten seleccionar los tipos de actividad e indicadores.	6. El sistema envía los datos de la asignatura y su curso, y de los tipos de actividad e indicadores seleccionados para el monitoreo de las actividades de aprendizaje durante el proceso educativo, también retorna un mensaje del resultado de dicha operación.

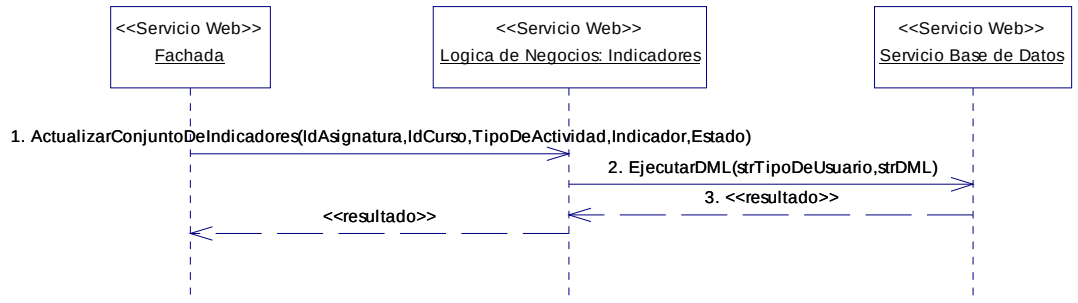
CAPA DE LOGICA DE PRESENTACION: DIAGRAMA DE SECUENCIA DEL CASO DE USO SELECCIONAR ACTIVIDADES A MONITOREAR

El profesor hace clic en la opción "Indicadores" del menú "Gestión". El sistema visualiza por medio de una interfaz: las asignaturas y cursos que orienta el profesor, también una lista de los tipos de actividad y del conjunto de indicadores. Una vez que el profesor realiza la operación de selección de las actividades de aprendizaje, esta información la envía el controlador a la clase Indicadores, a través de los métodos definidos para cada operación. Esta clase también está encargada de comunicarse y esperar respuesta del Servicio Web. Según el resultado, se despliega un mensaje al profesor.

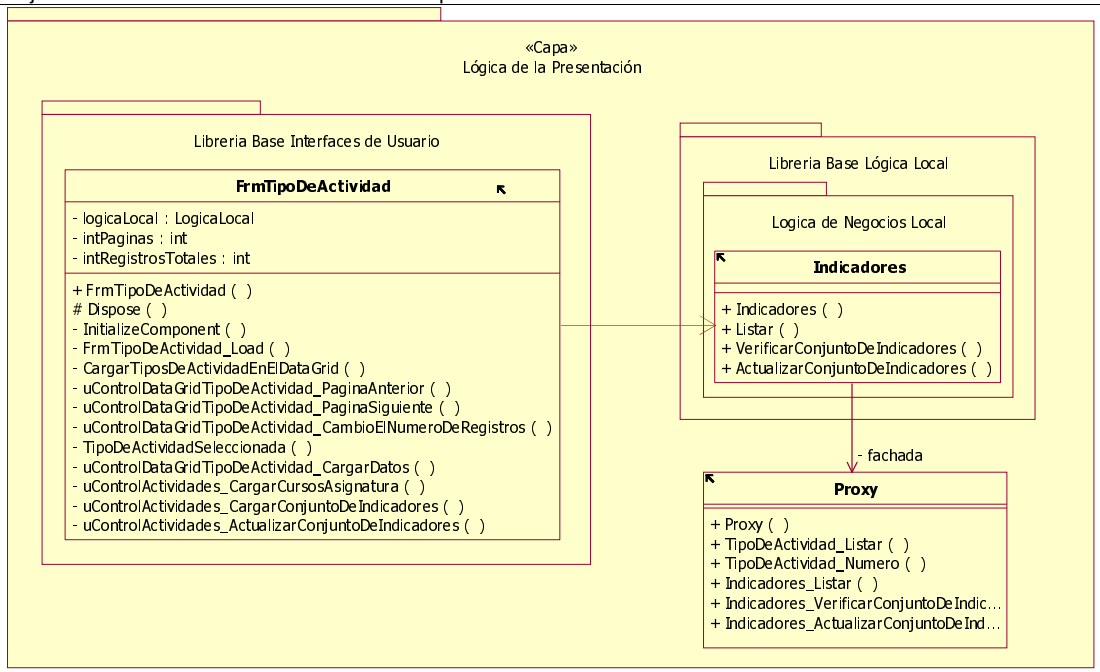


CAPA DE LOGICA DEL NEGOCIO: DIAGRAMA DE SECUENCIA DEL CASO DE USO SELECCIONAR ACTIVIDADES A MONITOREAR

El servicio Web Fachada, recibe desde la capa de presentación, la información referente a la selección de las actividades a monitorear, este servicio Web delega al servicio Web Indicadores el proceso de gestión, utilizando el patrón experto. El servicio Web Indicadores procesa los datos e invoca al método respectivo de la clase de lógica Servicios base de datos, que se encarga de ejecutar el proceso de gestión contra la base de datos. Luego se informa del éxito o fracaso de dicho proceso a través del servicio Web hasta llegar a la capa de presentación.



FORMATO CASO DE USO: SELECCIONAR ACTIVIDADES A MONITOREAR
CAPA DE LOGICA DE PRESENTACION: DIAGRAMA DE CLASES DEL CASO DE USO SELECCIONAR ACTIVIDADES A MONITOREAR
La interfaz Tipos de Actividad se relaciona con el controlador a través de su código subyacente, que se relaciona con la lógica de negocios local por medio de la clase Indicadores, que se relaciona con el Proxy de Fachada, dado que "Actividades de aprendizaje" utiliza métodos del Servicio Web de la aplicación.



CAPA DE LOGICA DEL NEGOCIO Y SERVICIO: DIAGRAMA DE CLASES DEL CASO DE USO SELECCIONAR ACTIVIDADES A MONITOREAR
Fachada instancia un objeto de Indicadores y Tipos de Actividad para delegarle el proceso de selección de tipos de actividad y sus indicadores, existiendo una asociación directa entre ellas. Los objetos Indicadores y Tipo de Actividad utilizan los métodos adecuados de la clase de Servicios de la base de datos de acuerdo a la operación de gestión, por lo que existe una asociación directa entre estas.

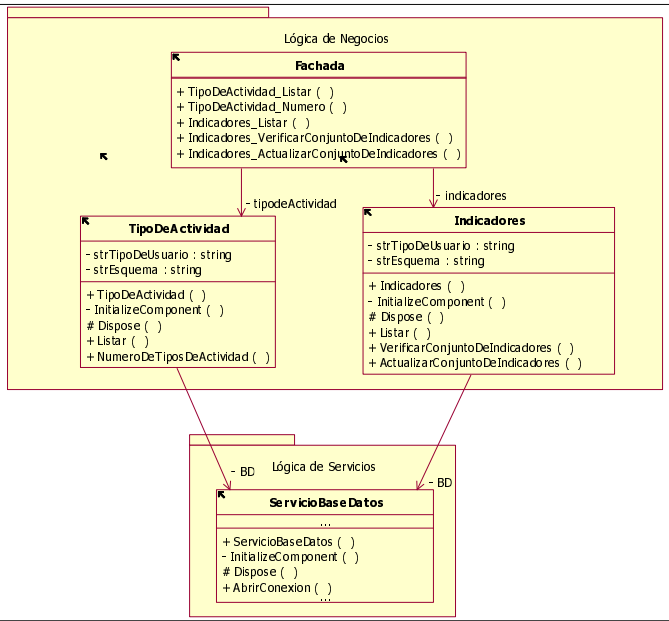


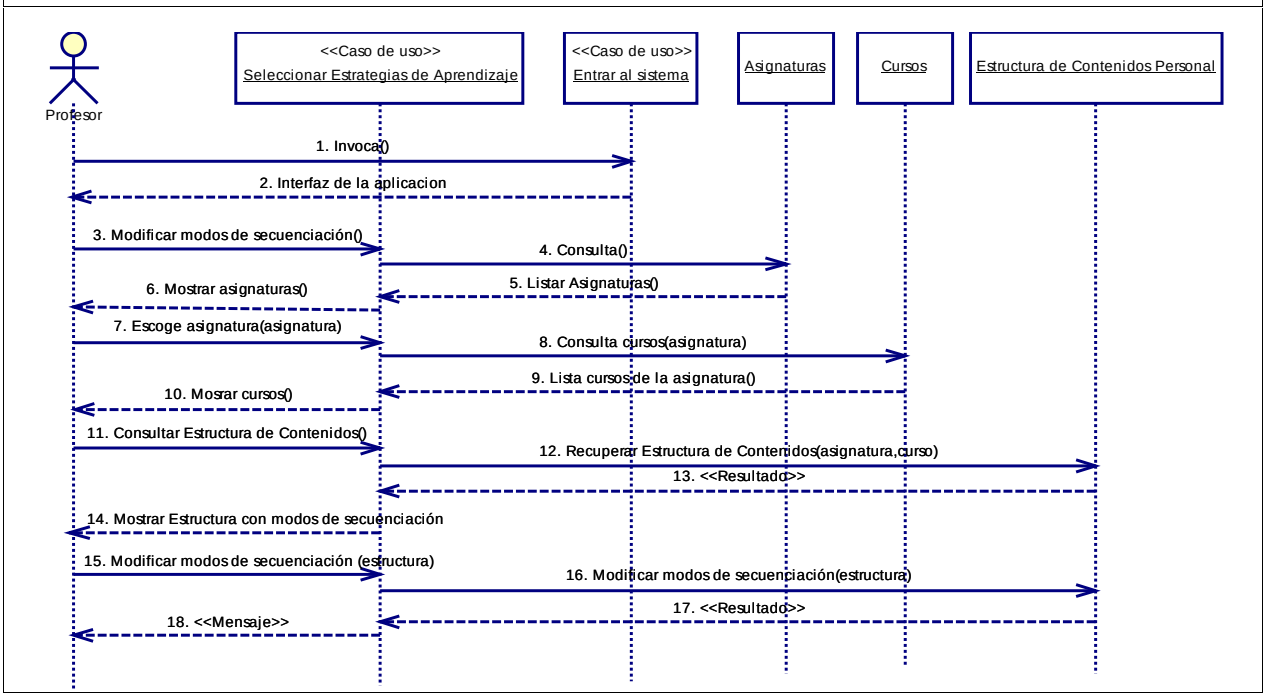
Tabla D.1.1.9. Diseño del caso de uso seleccionar Tipos de Actividad a monitorear

A.1.7 Caso de uso: Seleccionar estrategias de aprendizaje

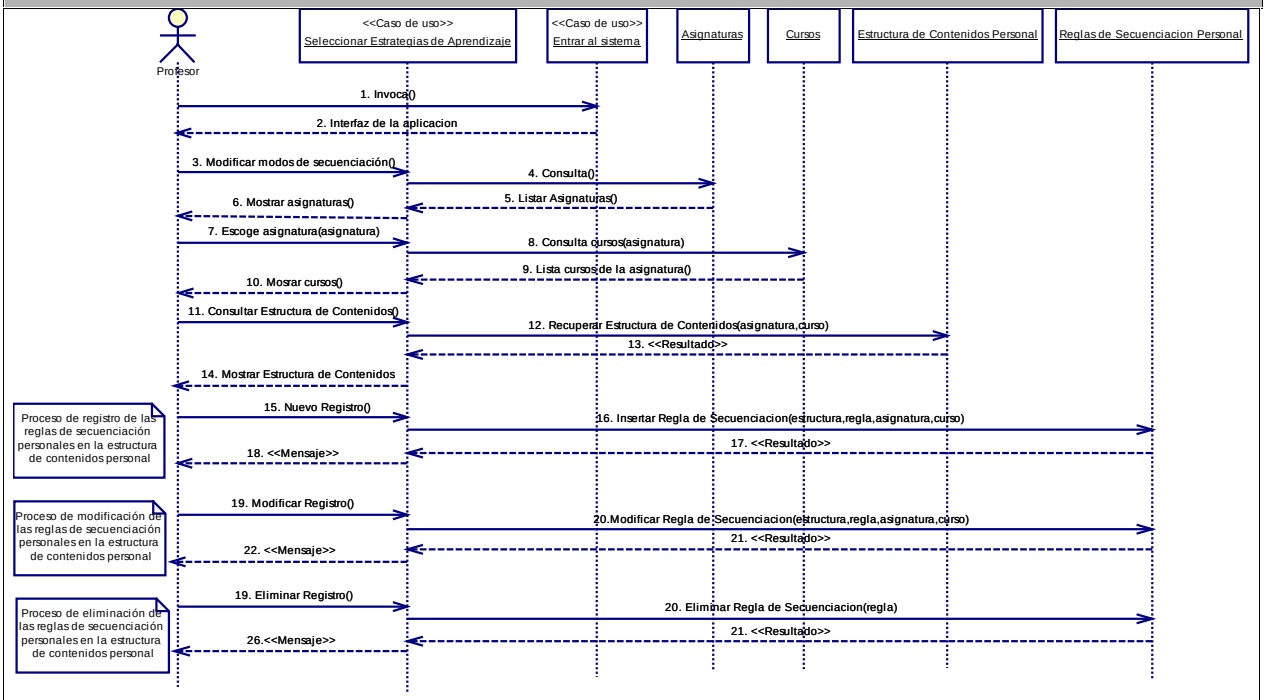
A.1.7.1 Análisis

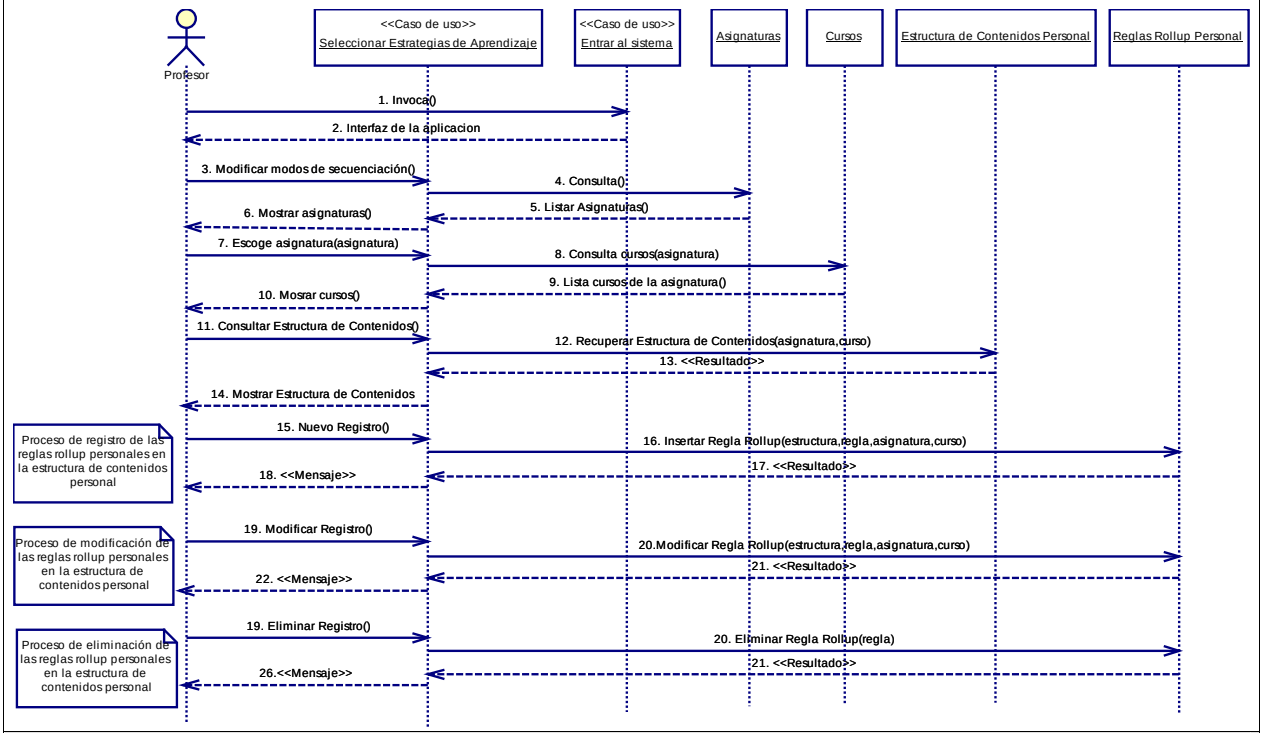
Este caso de uso le permite al profesor seleccionar las estrategias básicas de aprendizaje a utilizar en el contenido del curso a orientar. Dichas estrategias de aprendizaje son básicas y se reflejan por medio de la técnica de secuenciación de contenidos mediante la definición de los modos de secuenciación, reglas de secuenciación y reglas rollup que quiere que se ejecuten en la estructura de contenidos del curso a orientar. El análisis de este caso de uso es el siguiente:

FORMATO CASO DE USO SELECCIONAR ESTRATEGIAS DE APRENDIZAJE	
Actores	
• Profesor (Iniciador)	
Este caso de uso comienza cuando el profesor invoca al caso de uso Entrar al sistema, con el fin de seleccionar las estrategias básicas de aprendizaje que permiten la personalización de la estructura de contenidos personal. El sistema presenta la estructura de contenidos personal, la cual al ser copia, contiene las reglas de secuenciación establecidas en la estructura de contenidos original, el profesor modifica dichas reglas de secuenciación en la estructura de contenidos personal según cómo quiera orientar su curso. Este caso de uso finaliza cuando el profesor: abandona el proceso, cierra la sesión o cierra la aplicación.	
Acción del Actor	**Respuesta del Sistema**
1. El profesor decide seleccionar las estrategias de aprendizaje básicas a utilizar en la presentación del contenido del curso.	**2**. El sistema redirecciona al profesor a la interfaz de conexión.
3. El profesor utiliza el caso de uso **Entrar al sistema**.	4. Si Entrar al sistema es exitoso, el sistema presenta la interfaz principal de la aplicación.
	5. El sistema visualiza la estructura de contenidos personal asociada al profesor, utilizando el del caso de uso Armar estructura de contenidos personal.
6. El profesor elige las estrategias de aprendizaje (modos de secuenciación, reglas de secuenciación y reglas rollup) que quiere que se ejecuten en la estructura de contenidos personal durante el desarrollo del curso.	**7**. El sistema almacena dicho proceso.
DIAGRAMA DE SECUENCIA DEL SISTEMA DEL CASO DE USO SELECCIONAR ESTRATEGIAS DE APRENDIZAJE	
Este diagrama muestra todo el proceso necesario que debe hacer el actor profesor para seleccionar y establecer las estrategias de aprendizaje básicas a emplear en la estructura de contenidos personal.	



FORMATO CASO DE USO SELECCIONAR ESTRATEGIAS DE APRENDIZAJE





MODELO CONCEPTUAL DEL CASO DE USO SELECCIONAR ESTRATEGIAS DE APRENDIZAJE

En el modelo conceptual, se destacan los conceptos más importantes del caso de uso Seleccionar estrategias de aprendizaje. Los objetos persistentes o entidades que usa el caso de uso es: Estructura de Contenidos Personal, Reglas de Secuenciación Personal y Reglas Rollup Personal. El control de este caso de uso está representado por el objeto: Estructura de Contenidos Personal, y la interfaz para la interacción con el actor profesor es el objeto del mismo nombre.

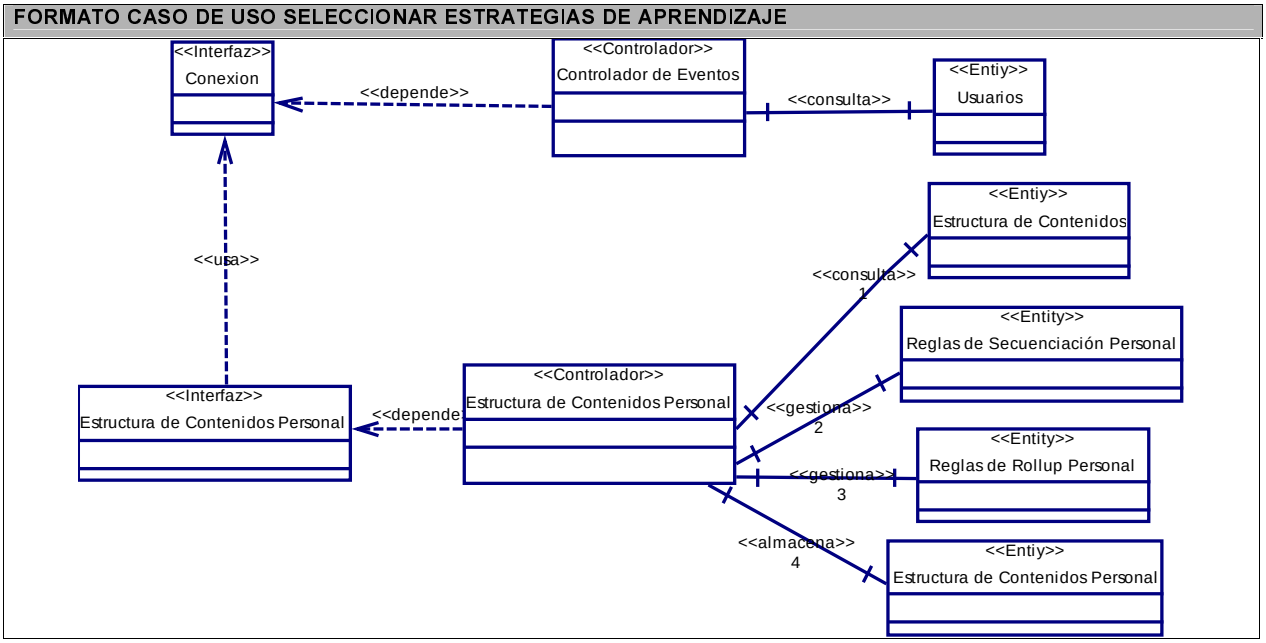


Tabla A.1.1.12 Análisis del caso de uso Seleccionar estrategias de aprendizaje

A.1.7.2 Diseño

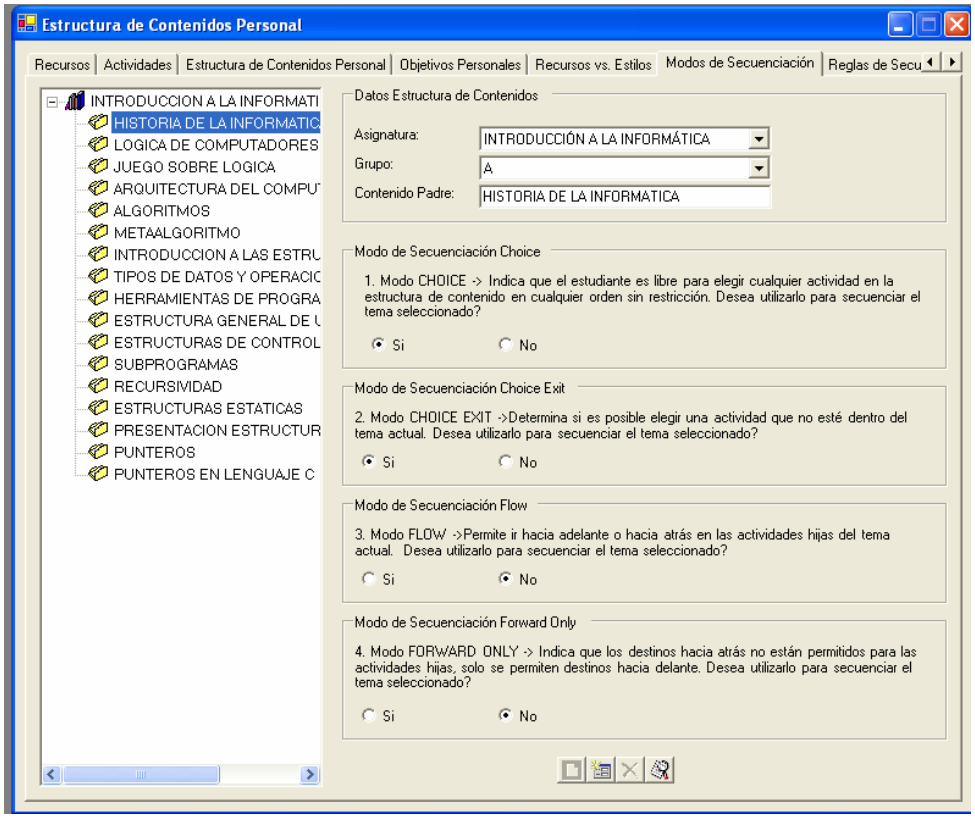
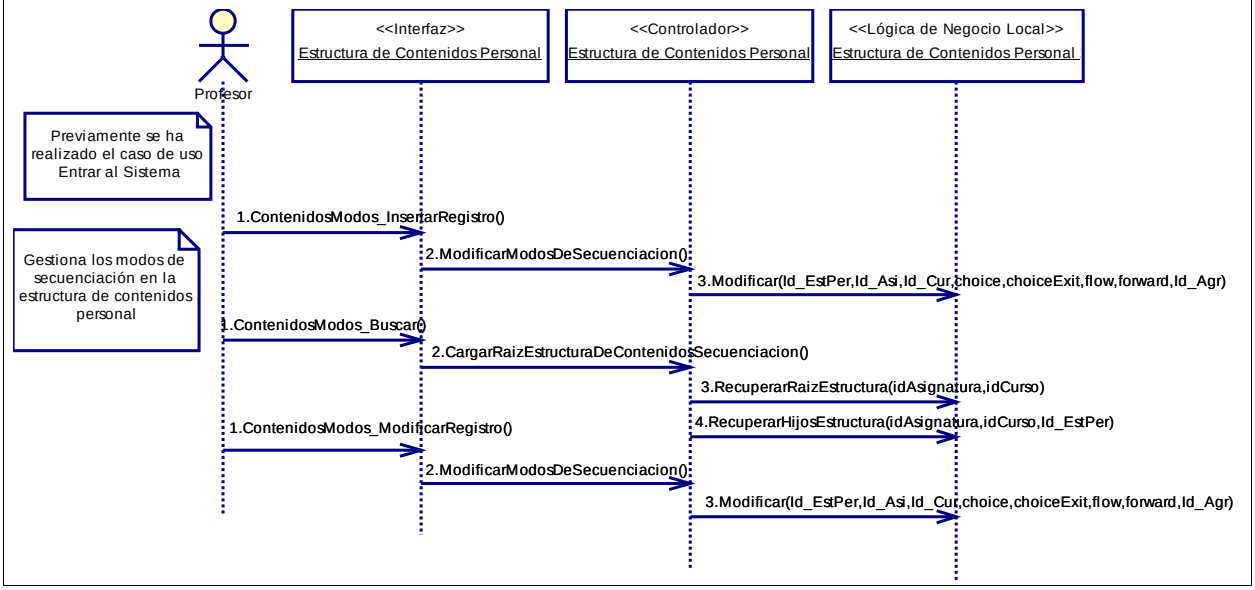


Figura 10. Screen Shot del caso de uso Seleccionar estrategias de aprendizaje

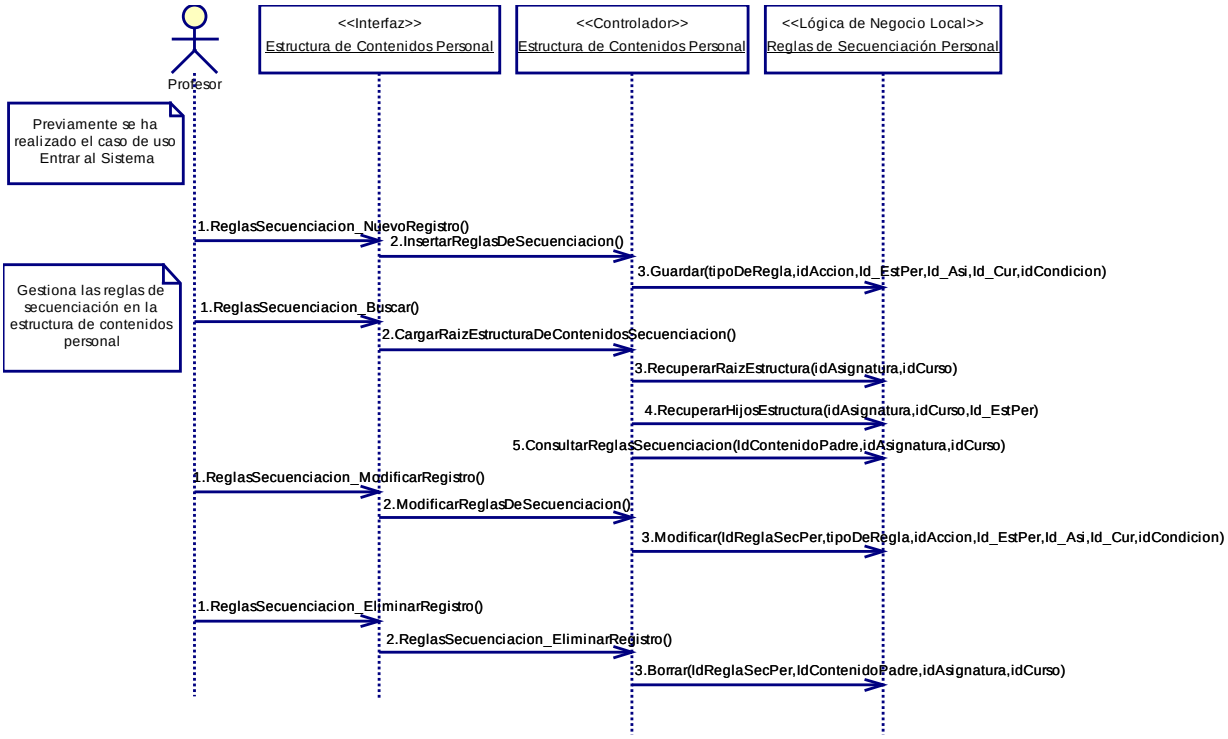
FORMATO CASO DE USO: SELECCIONAR ESTRATEGIAS DE APRENDIZAJE	
Actores	
Profesor (Iniciador)	
Este caso de uso comienza cuando el profesor decide seleccionar las estrategias de aprendizaje a utilizar en su estructura de contenidos. El sistema solicita primero al profesor que se loguee, invocando el caso de uso Entrar al Sistema. Después de este proceso el sistema presenta al profesor la interfaz principal, donde el profesor elige la opción "Estructura de Contenidos Personal", que presenta la interfaz que permite la selección de las estrategias de aprendizaje sobre una estructura de contenidos personal. El controlador de la forma envía los datos de los modos de secuenciación a la clase Estructura de Contenidos Personal, los datos de las reglas de secuenciación a la clase Reglas de Secuenciación Personal y los datos de las reglas rollup a la clase Reglas Rollup Personal que se encuentran en Lógica de Negocios Local, las cuales se comunican y propagan los datos a través de la referencia Proxy del servicio Web Fachada, quien se comunica con los servicios Web respectivos: Estructura de Contenidos Personal, Reglas de Secuenciación Personal y Reglas Rollup Personal de la lógica de Negocios, los cuales gestionan las estrategias de aprendizaje (modos, reglas de secuenciación y reglas rollup) seleccionas para dicha estructura de contenidos en la base de datos, ayudado del servicio Web Lógica de Servicios. Finalmente el sistema, envía un mensaje al profesor sobre el éxito de la operación. Este caso de uso finaliza cuando el profesor: abandona el proceso, cierra la sesión o cierra la aplicación.	
Acción del Actor	**Respuesta del Sistema**
1. El profesor decide seleccionar las estrategias de aprendizaje básicas a utilizar en la presentación del contenido del curso.	2. El sistema redirecciona al profesor a la interfaz de Conexión.
3. El profesor utiliza el caso de uso Entrar al Sistema.	4. Si Entrar al sistema es exitoso, el sistema presenta la interfaz principal de la aplicación.
5. El profesor utiliza Estructura de contenidos personal, reglas de secuenciación personal y reglas rollup personal, que permiten seleccionar las estrategias básicas de aprendizaje sobre una estructura de contenidos personal.	6. El sistema envía los datos, también retorna un mensaje del resultado de dicha operación.

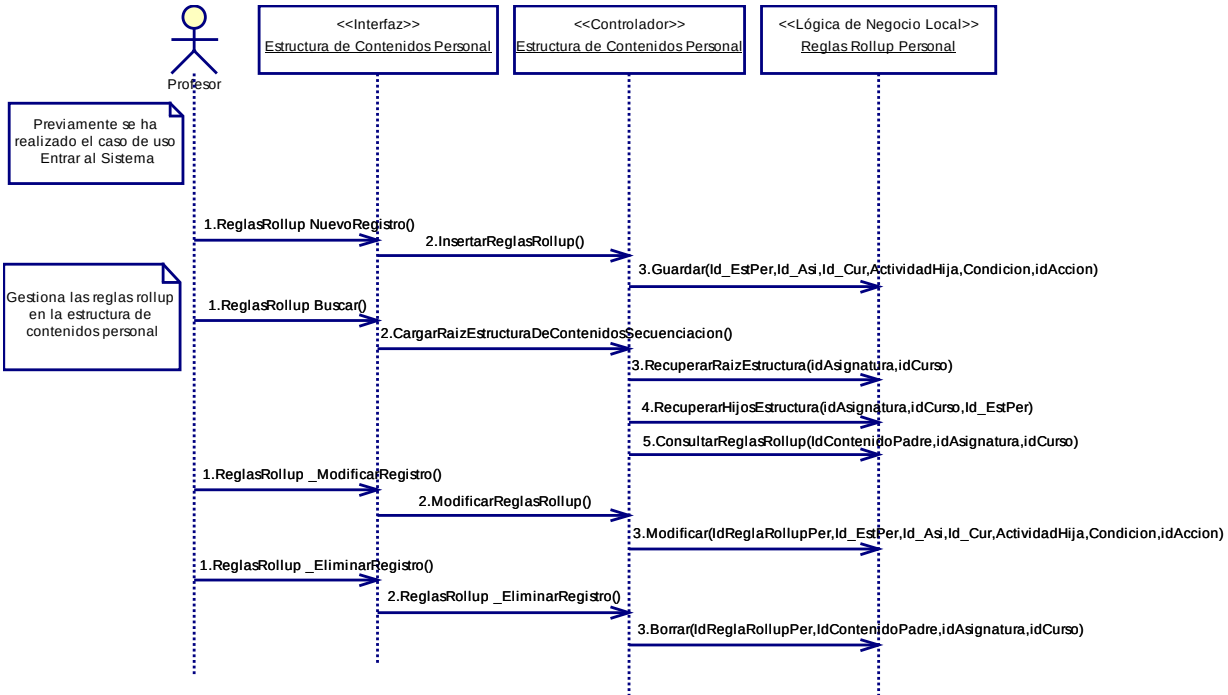
CAPA DE LOGICA DE PRESENTACION: DIAGRAMA DE SECUENCIA DEL CASO DE USO SELECCIONAR ESTRATEGIAS DE APRENDIZAJE

El profesor utiliza Estructura de Contenidos Personal, Reglas de Secuenciación Personal y Reglas Rollup Personal para seleccionar las estrategias básicas de aprendizaje a emplear sobre una estructura de contenidos personal. El sistema visualiza por medio de una interfaz: la estructura de contenidos personal junto con sus estrategias básicas predefinidas. Una vez que el profesor establece las estrategias a utilizar en su estructura personal, esta información la envía el controlador a las clases Estructura de Contenidos, Reglas de Secuenciación Personal y Reglas Rollup Personal, a través de los métodos definidos para cada operación. Estas clases también están encargadas de comunicarse y esperar respuesta del servicio Web respectivo. Según el resultado, se despliega un mensaje al profesor.



FORMATO CASO DE USO: SELECCIONAR ESTRATEGIAS DE APRENDIZAJE



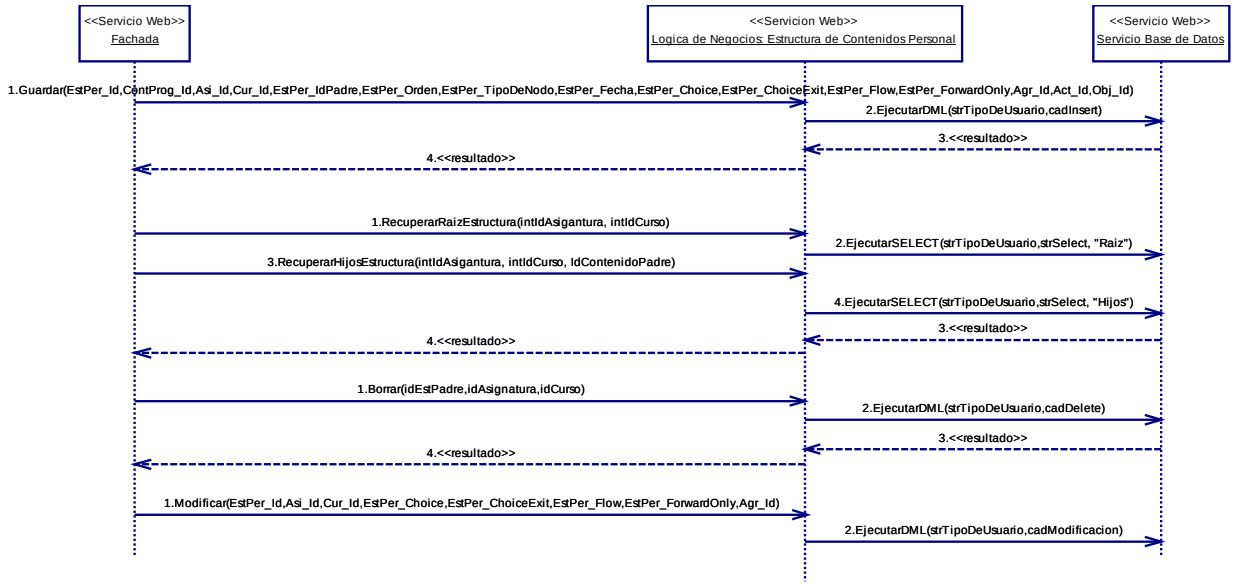


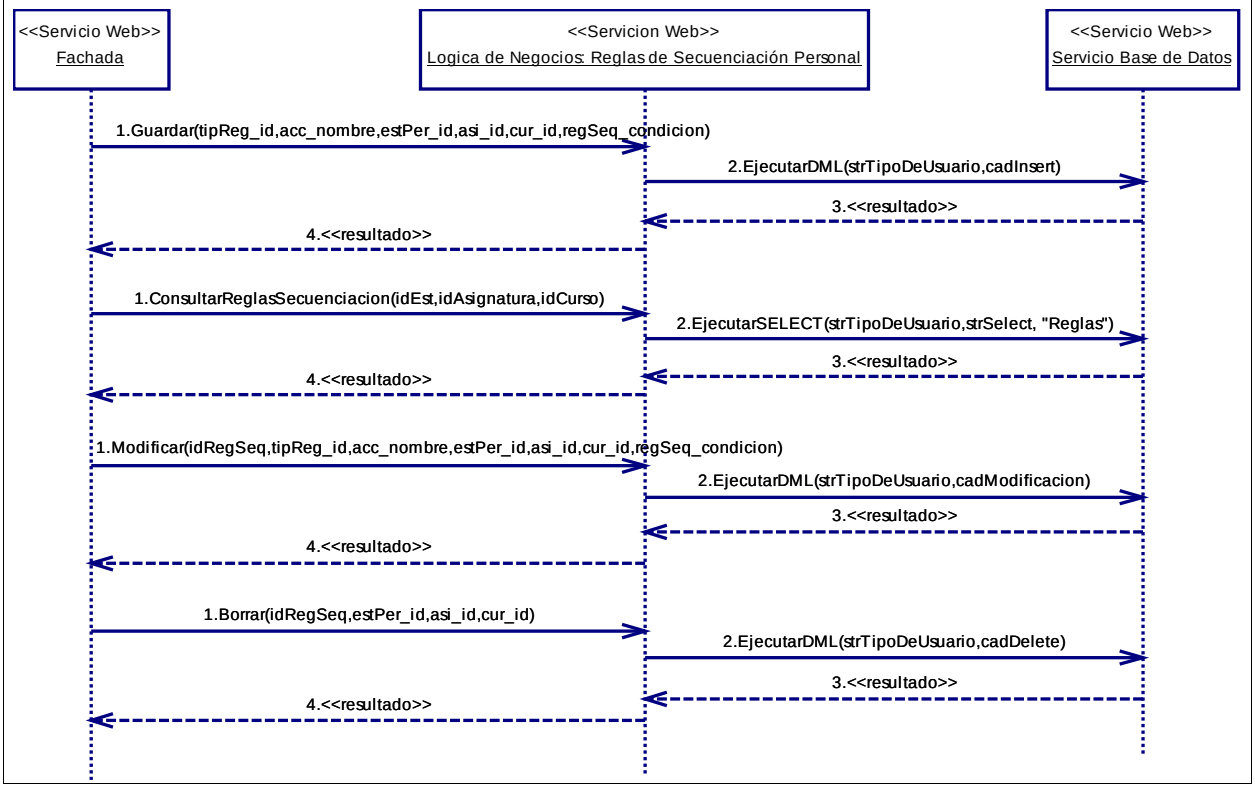
CAPA DE LOGICA DEL NEGOCIO: DIAGRAMA DE SECUENCIA DEL CASO DE USO SELECCIONAR ESTRATEGIAS DE APRENDIZAJE

El servicio Web Fachada, recibe desde la capa de presentación, la información referente a la selección de estrategia sobre una

FORMATO CASO DE USO: SELECCIONAR ESTRATEGIAS DE APRENDIZAJE

estructura de contenidos personal, este servicio Web delega a los servicios Web Estructura de Contenidos Personal, Reglas de Secuenciación Personal y Reglas Rollup Personal el proceso de gestión, utilizando el patrón experto. Los servicios Web Estructura de Contenidos Personal, Reglas de Secuenciación Personal y Reglas Rollup procesa los datos e invoca al método respectivo de la clase de lógica Servicios base de datos, que se encarga de ejecutar el proceso de gestión contra la base de datos. Luego se informa del éxito o fracaso de dicho proceso a través del servicio Web hasta llegar a la capa de presentación.





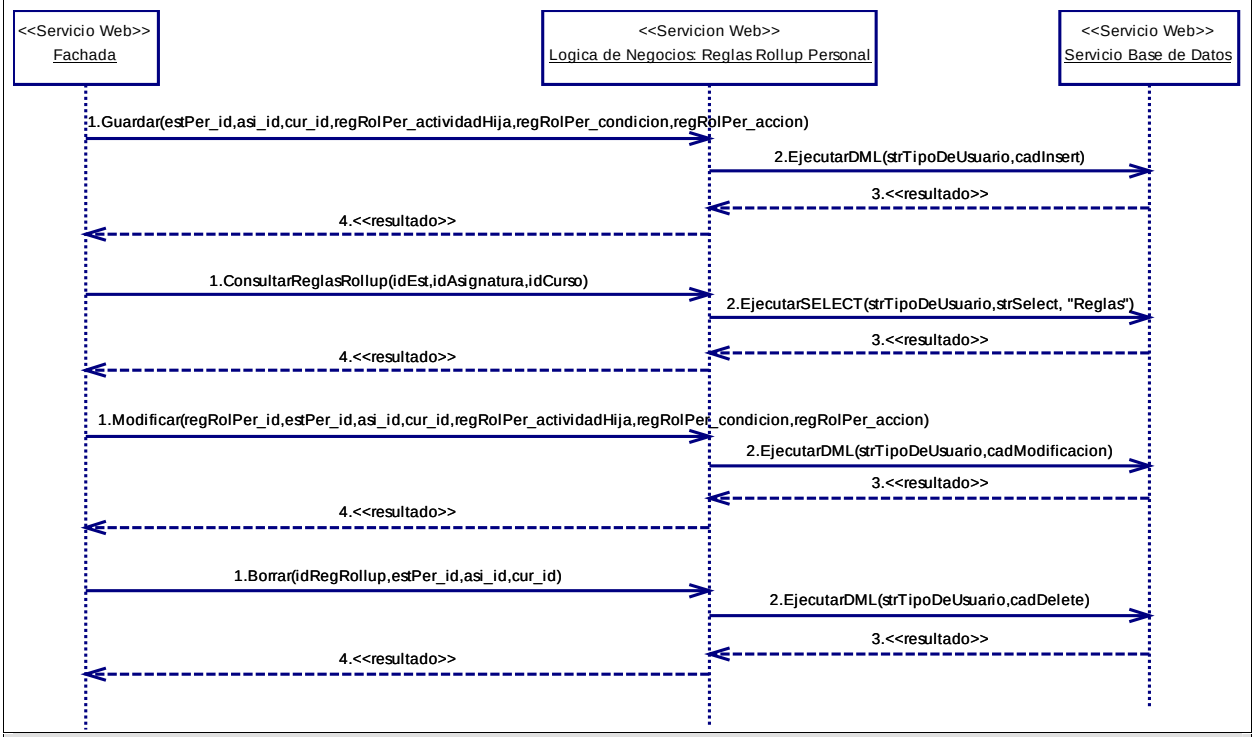
FORMATO CASO DE USO: SELECCIONAR ESTRATEGIAS DE APRENDIZAJE

CAPA DE LOGICA DE PRESENTACION: DIAGRAMA DE CLASES DEL CASO DE USO SELECCIONAR ESTRATEGIAS DE APRENDIZAJE
La interfaz Gestionar Estructura de Contenidos Personal se relaciona con el controlador a través de su código subyacente, que se relaciona con la lógica de negocios local por medio de las clases Estructura de Contenidos Personal, Reglas de Secuenciación Personal y Reglas Rollup Personal, que se relaciona con el Proxy de Fachada, dado que la selección de estrategias básicas de aprendizaje utiliza métodos del Servicio Web de la aplicación.
CAPA DE LOGICA DEL NEGOCIO y SERVICIO: DIAGRAMA DE CLASES DEL CASO DE USO SELECCIONAR ESTRATEGIAS DE APRENDIZAJE
Fachada instancia objetos de Estructura de Contenidos Personal, Reglas de Secuenciación Personal y Reglas Rollup Personal para delegarle el proceso de selección de estrategias de aprendizaje, existiendo una asociación directa entre ellas. Estos objetos utilizan los métodos adecuados de la clase de Servicios de la base de datos de acuerdo a la operación de gestión, por lo que existe una asociación directa entre estas.

Tabla D.1.1.10. Diseño del caso de uso Seleccionar estrategias de aprendizaje

A.1.8 Caso de uso: Modificar Perfil

Este caso de uso le permite al estudiante modificar la información de ciertos datos personales como dirección de residencia, correos, página Web personal, y así mantener actualizada la información de los usuarios registrados en el sistema.

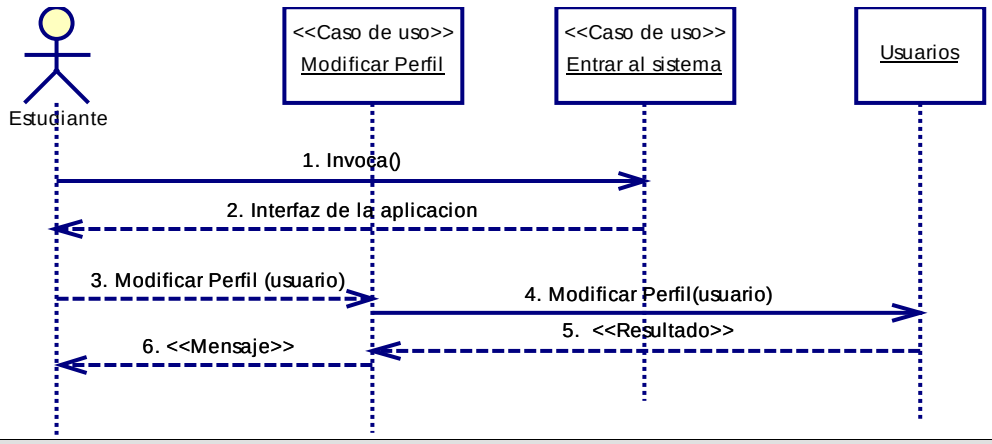
A.1.8.1 Análisis

FORMATO CASO DE USO MODIFICAR PERFIL	
Actores	
• Estudiante (Iniciador)	
Este caso de uso comienza cuando el estudiante invoca al caso de uso Entrar al sistema, con el fin de modificar su perfil (datos personales). El sistema solicita la información relacionada con la modificación del perfil del estudiante y realiza el respectivo proceso, luego envía un mensaje al estudiante del resultado de la operación efectuada. Este caso de uso finaliza cuando el estudiante: abandona el proceso, cierra la sesión o cierra la aplicación.	
Acción del Actor	**Respuesta del Sistema**

FORMATO CASO DE USO MODIFICAR PERFIL	
1. El estudiante decide modificar ciertos datos personales (perfil).	**2**. El sistema redirecciona al estudiante a la interfaz de conexión.
3. El estudiante utiliza el caso de uso **Entrar al sistema**.	4. Si Entrar al sistema es exitoso, el sistema presenta la interfaz principal de la aplicación.
5. El estudiante ingresa la información de los datos personales ha modificar.	**6**. El sistema realiza el respectivo proceso de modificación e informa el resultado de la misma.

DIAGRAMA DE SECUENCIA DEL SISTEMA DEL CASO DE USO MODIFICAR PERFIL

Este diagrama muestra todo el proceso necesario que debe hacer el actor estudiante para realizar el caso de uso Modificar Perfil. El estudiante ingresa la información actualizada de sus datos personales. El sistema le envía un mensaje de información acerca del resultado de la operación efectuada.



MODELO CONCEPTUAL DEL CASO DE USO MODIFICAR PERFIL

En el modelo conceptual, se destacan los conceptos más importantes del caso de uso Modificar Perfil. El objeto persistente o entidad que usa el caso de uso es: Usuarios. El control de este caso de uso está representado por el objeto Modificar Perfil y la interfaz para la interacción con el actor estudiante es el objeto del mismo nombre.
El control realiza en la entidad Usuarios el proceso de modificación efectuada por el estudiante.

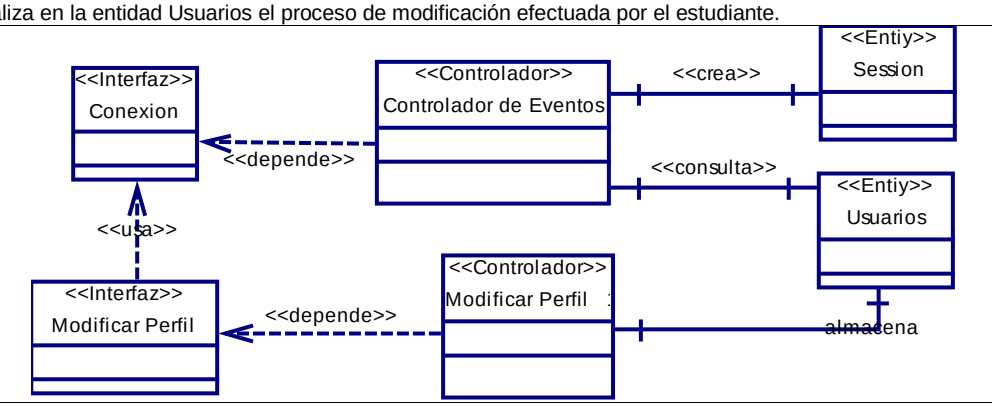


Tabla A.1.1.13 Análisis del caso de uso Modificar Perfil

A.1.8.2 Diseño

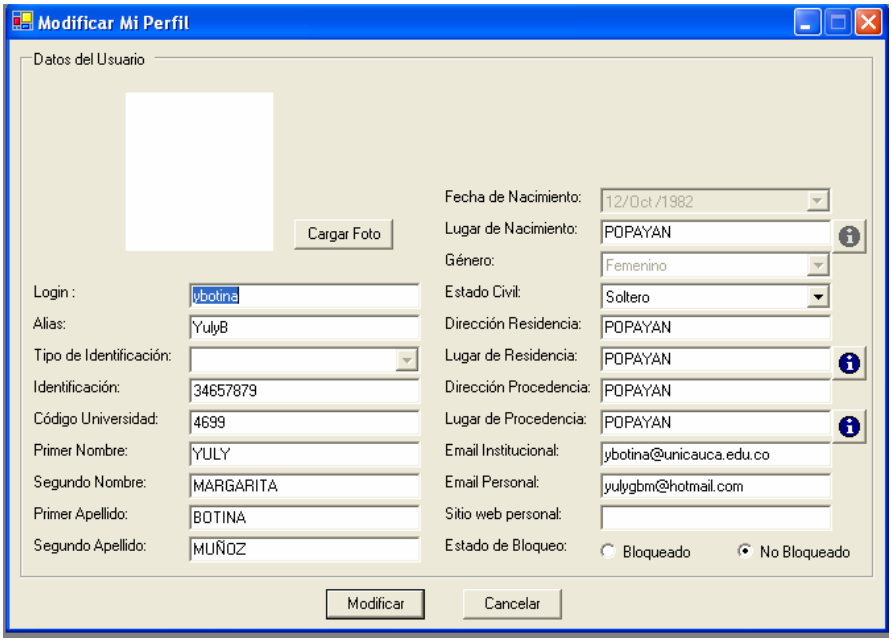


Figura 11. Screen Shot del caso de uso Modificar Perfil

FORMATO CASO DE USO: MODIFICAR PERFIL	
Actores	
Estudiante (Iniciador)	
Este caso de uso comienza cuando el estudiante decide modificar su perfil. El sistema solicita primero al estudiante que se loguee, invocando el caso de uso Entrar al Sistema. Después de este proceso el sistema presenta al estudiante la interfaz principal, donde el estudiante elige la opción "Modificar Perfil", que presenta la interfaz que modifica ciertos datos personales. Una vez realizada esta operación, el controlador envía los datos del estudiante a la clase Usuarios de Lógica de Negocios Local, la cual se comunica y propaga los datos a través de la referencia Proxy del servicio Web Fachada, quien se comunica con el servicio Web Usuarios de la lógica de Negocios, el cual modifica el perfil del estudiante en la base de datos, ayudado del servicio Web Lógica de Servicios. Finalmente el sistema, envía un mensaje al estudiante sobre el éxito de la operación. Este caso de uso finaliza cuando el estudiante: abandona el proceso, cierra la sesión o cierra la aplicación.	
Acción del Actor	**Respuesta del Sistema**
1. El estudiante decide modificar la información de ciertos datos personales.	2. El sistema redirecciona al estudiante a la interfaz de Conexión.
3. El estudiante utiliza el caso de uso Entrar al Sistema.	4. Si Entrar al sistema es exitoso, el sistema presenta la interfaz principal de la aplicación.
5. El estudiante modifica la información de ciertos datos personales.	6. El sistema envía los datos del estudiante y ejecuta el proceso de modificación de perfil, también retorna un mensaje del resultado de dicha operación.
CAPA DE LOGICA DE PRESENTACION: DIAGRAMA DE SECUENCIA DEL CASO DE USO MODIFICAR PERFIL	
El estudiante modifica la información de ciertos datos personales por medio de la interfaz Modificar Perfil. Una vez que el estudiante realiza dicha operación, esta información la envía el controlador a la clase Usuarios, a través de los métodos definidos para la operación. Esta clase también está encargada de comunicarse y esperar respuesta del Servicio Web. Según el resultado, se despliega un mensaje al estudiante.	

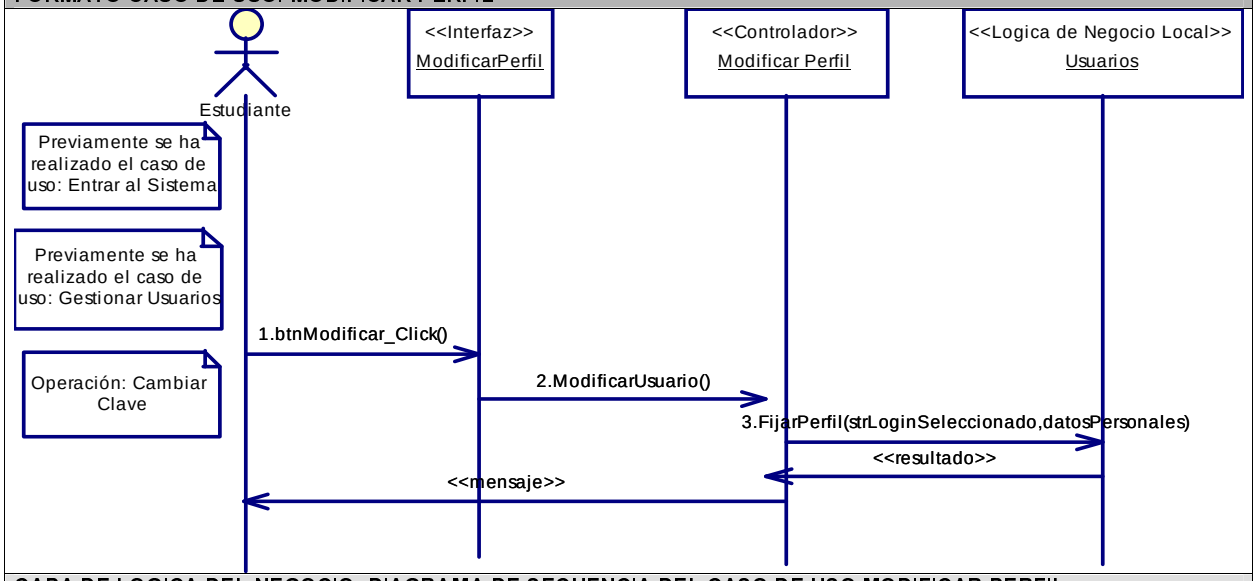
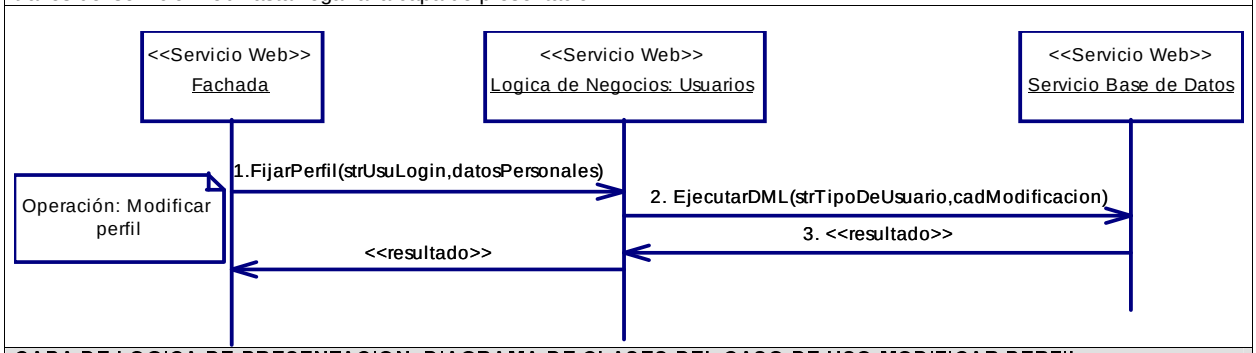
FORMATO CASO DE USO: MODIFICAR PERFIL

CAPA DE LOGICA DEL NEGOCIO: DIAGRAMA DE SECUENCIA DEL CASO DE USO MODIFICAR PERFIL
El servicio Web Fachada, recibe desde la capa de presentación, la información de los datos personales del estudiante, este servicio Web delega al servicio Web Usuarios el proceso de modificación del perfil del estudiante, utilizando el patrón experto. El servicio Web Usuarios procesa los datos e invoca al método respectivo de la clase de lógica Servicios base de datos, que se encarga de ejecutar el proceso de modificación contra la base de datos. Luego se informa del éxito o fracaso de dicho proceso a través del servicio Web hasta llegar a la capa de presentación.

CAPA DE LOGICA DE PRESENTACION: DIAGRAMA DE CLASES DEL CASO DE USO MODIFICAR PERFIL
La interfaz Modificar Perfil se relaciona con el controlador a través de su código subyacente, que se relaciona con la lógica de negocios local a través de la clase Usuarios, que se relaciona con el Proxy de la Fachada, dado que Modificar Perfil utiliza métodos del Servicio Web de la aplicación. (La interfaz de la capa de Presentación de este caso de uso es la misma que la del caso de uso Gestionar Usuarios).
CAPA DE LOGICA DEL NEGOCIO Y SERVICIO: DIAGRAMA DE CLASES DEL CASO DE USO MODIFICAR PERFIL
Fachada instancia un objeto de Usuarios para delegarle el proceso de modificación, existiendo una asociación directa entre ellas. El objeto Usuarios utiliza los métodos adecuados de la clase de Servicios de la base de datos de acuerdo a la operación de modificación, por lo que existe una asociación directa entre estas. (La interfaz de la capa de Negocio y Servicios de este caso de uso es la misma que la del caso de uso Gestionar Usuarios).

Tabla D.1.1.11. Diseño del caso de uso Modificar Perfil

A.1.9 Modelos Lógicos y Físicos

A continuación se presenta el modelo conceptual que permite representar los casos de uso planteados anteriormente y también el modelo físico que permite su persistencia en la base de datos.

A.1.9.1 Modelo Entidad-Relación

MODELO ENTIDAD - RELACION

El siguiente modelo se presenta los conceptos y relaciones abstraídos del entorno de educación en línea que permiten el proceso de adaptatividad de la presentación de contenidos para cursos de educación en línea por medio del STI.

- ***Usuarios***: Representa a los estudiantes, profesores, administradores y directivos involucrados en el proceso educativo.
- ***Matrículas***: Almacena la información de las asignaturas y sus cursos, matriculadas por el estudiante para ser cursadas.
- ***Asignaturas***: Representa las materias o temáticas que se enseñarán al estudiante.
- ***Cursos***: Representa los grupos establecidos para las asignaturas.
- ***Modelos de aprendizaje***: Representan las concepciones teóricas que buscan explicar el comportamiento humano frente al aprendizaje y el tipo de acción que puede resultar más eficaz en un momento dado.
- ***Estilos de aprendizaje***: Permiten conocer la conducta que tiene el estudiante durante su aprendizaje, es decir, su forma de percibir, procesar y analizar la información.
- ***Estilos de aprendizaje asociados***: Permite almacenar información de los estilos de aprendizaje en los cuales el estudiante puede estar clasificado.
- ***Cuestionarios***: Representa los cuestionarios asociados a los modelos de aprendizaje, que se le realizan al estudiante cuando cursa por primera vez una asignatura con el fin de establecer sus estilos de aprendizaje asociados.
- ***Preguntas***: Como su nombre lo indica representan la información de las preguntas asociadas a los cuestionarios o tests que se le realizan al estudiante.
- ***Respuestas***: Representan todas las opciones de respuestas establecidas para cada una de las preguntas. Cada respuesta esta asociada con un estilo de aprendizaje de un modelo.
- ***Hoja de Respuestas***: Cuando al estudiante se le realiza el cuestionario, éste llena una hoja de respuestas, con aquellas respuestas de su elección. Con esta información se establece inicialmente sus estilos de aprendizaje asociados, dado que cada respuesta se enfoca hacia un estilo de aprendizaje de un modelo en particular.
- ***Estructuras de Contenidos***: Representa la información original referente a la temática de los cursos, definidos mediante el estándar de SCORM.
- ***Estructuras de Contenidos Personal***: Igual a la estructura de contenidos, con la diferencia que la temática contiene los temas específicos que el profesor quiere desarrollar durante el curso.
- ***Estados Actividad Estudiante*** Registra el desempeño que tiene el estudiante durante su proceso educativo a través de la estructura de contenidos personal.
- ***Reglas de Secuenciación***: Son reglas que facilitan la presentación adaptativa a través de la forma en que se ordenan los contenidos de acuerdo a algún criterio, que dependerá de la meta y otras características del estudiante.
- ***Reglas Rollup***: Reglas que permiten establecer el estado de un contenido de la estructura de contenidos a partir del estado de sus actividades hijas afectadas.
- ***Acciones***: Representan las acciones que tienen las reglas de secuenciación.
- ***Reglas de Secuenciación personal***: Igual a la reglas de secuenciación, con la diferencia que se establecen sobre la estructura de contenidos personal.
- ***Reglas Rollup Personal:*** Equivalente a las reglas rollup, pero estas se utilizan con la estructura de contenidos personal.
- ***Tipos de Actividad:*** Representan las categorías en las cuales se pueden clasificar las actividades.
- ***Agregaciones:*** Representan los temas principales dentro de cada una de las estructuras de contenidos.
- ***Actividades:*** Representan las unidades de instrucción significantes de los temas por medio de las cuales se presenta al estudiante la información de cada uno de los temas a desarrollar, dado que las actividades facilitan la utilización de recursos.
- ***Objetivos:*** Representan los propósitos o metas que se buscan cumplir y/o lograr a través del desarrollo del contenido del curso.
- ***Indicadores:*** Representan el conjunto de parámetros a través de los cuales se realiza el proceso de monitoreo de las actividades del estudiante.
- ***Estados indicadores:*** Representa el estado que adquiere cada uno de los indicadores asociados a las actividades durante el proceso educativo de cada estudiante.
- ***Recursos:*** Representan los elementos por los cuales se presentan y explican los contenidos, estos pueden ser representaciones electrónicas de texto, imágenes, vídeos, entre otros.

Las relaciones entre las entidades se interpretan de la siguiente forma:

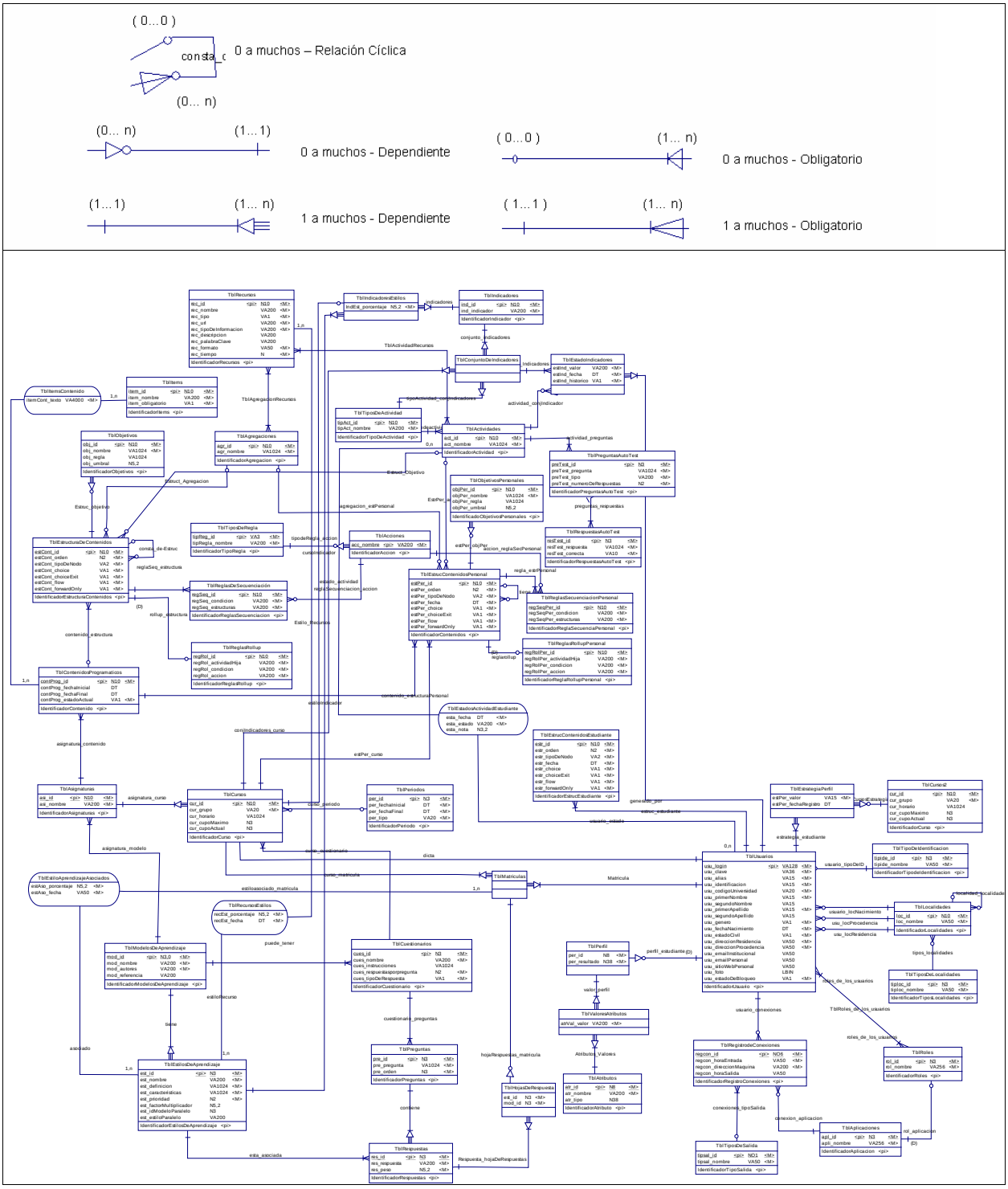


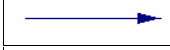
Tabla A.1.1.14 Modelo Entidad-Relación

A.1.9.2 Modelo Físico

MODELO FISICO

En este modelo las tablas contienen el tipo de dato y tamaño, las clases de atributos de integridad referencial. Las claves primarias, representadas por el atributo <pk>. Las claves foráneas representadas con el formato <fk#> y sus respectivas restricciones: claves foráneas que tienen el formato FK_[tabla Origen]_[tabla destino].

Las relaciones entre las tablas están representadas así:



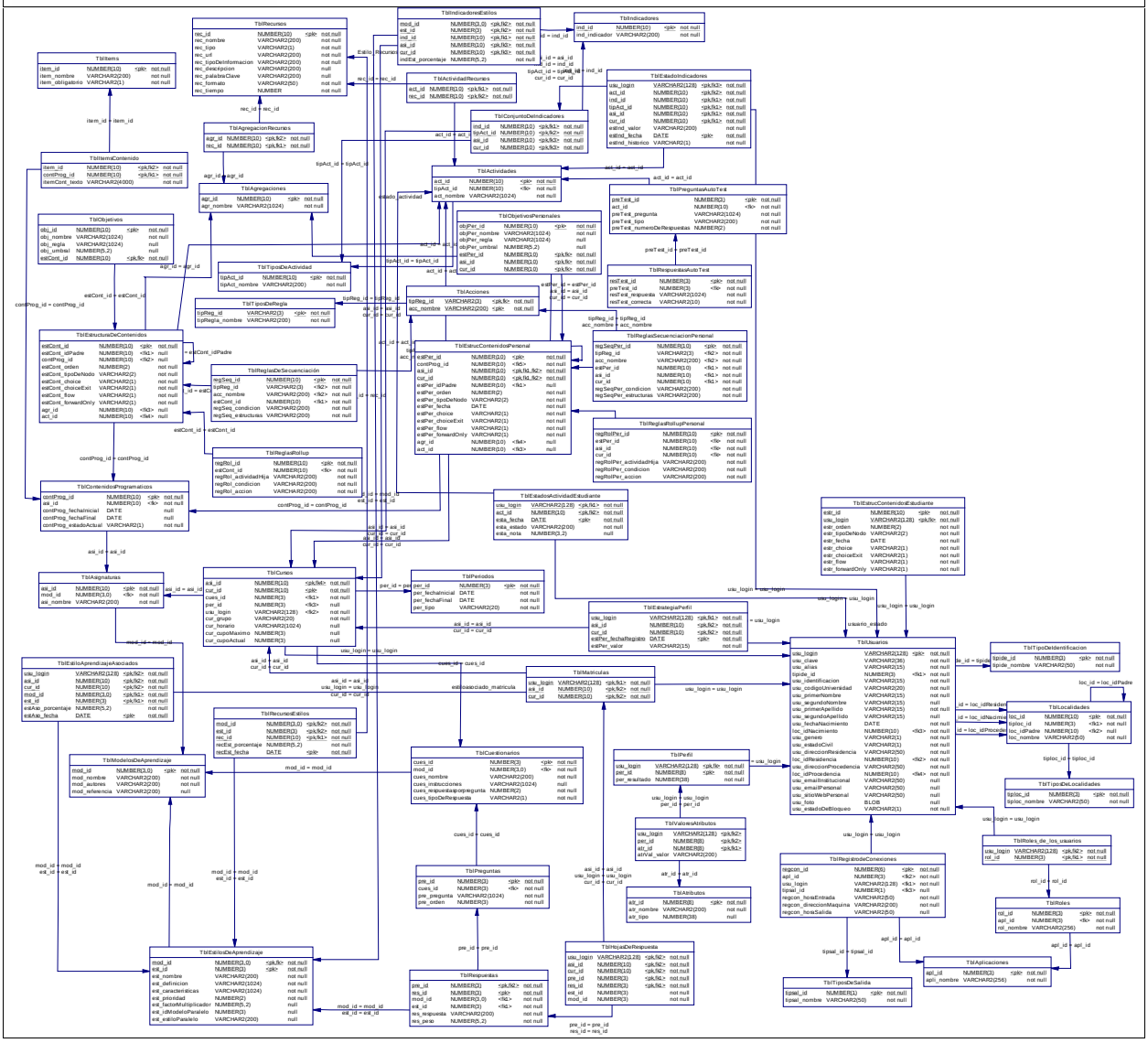


Tabla A.1.1.15 Modelo Físico

ANEXO B

BASES CONCEPTUALES

BASES CONCEPTUALES

B.1. E-LEARNING

B.1.1 Diversas definiciones de E-Learning

No existe una definición exacta del término E-Learning, algunas de las definiciones que brindan una idea acerca de su significado son:

- **E-Learning** es la convergencia de aprendizaje e Internet. – *Bank of America Securities.*
- **E-Learning** es el uso de tecnología de redes para el diseño, entrega, selección, administración y extensión del aprendizaje. – *Elliott Masie, The Masie Center*.
- **E-Learning** es el aprendizaje facilitado en Internet. Los componentes de E-Learning pueden incluir contenidos entregados en múltiples formatos, gestión de experiencias de aprendizaje, redes de trabajo de estudiantes, desarrolladores y expertos en contenidos. E-Learning provee rapidez en el aprendizaje a costos reducidos, incrementando el acceso a la educación con responsabilidades claras para todos los participantes en este proceso. – *Cisco Systems*.
- **E-Learning** es educación en línea entregada en forma síncrona (tiempo real, dirigida por un instructor) o asíncrona.
- **E-Learning** es el uso de la tecnología para administrar, diseñar, entregar, seleccionar, transmitir, adiestrar, soportar y extender todo tipo de aprendizaje.
- **E-Learning** traslada las experiencias de aprendizaje fuera de la tradicional aula de clases, esto es aprendizaje en cualquier momento y en cualquier lugar, sin barreras geográficas o de agenda, confiando en Internet para el acceso a los materiales de aprendizaje e interactuando con expertos y estudiantes semejantes

Por otra parte,

- **Técnicamente**: E-Learning es la entrega de material educativo por cualquier medio electrónico, como: Internet, audio, vídeo, red satelital, televisión interactiva, CD y DVD, entre otros medios.
- **Para los educadores**: E-Learning es el uso de tecnologías de redes y comunicaciones para diseñar, seleccionar, administrar, entregar y extender la educación.
- **Para los elocuentes:** E-Learning es el empleo del poder de la red mundial para proporcionar educación, en cualquier momento, en cualquier lugar.
- **Para los epigrafistas**: E-Learning representa la convergencia del aprendizaje e Internet.
- **Para los visionarios y futuristas:** E-Learning es a la educación convencional lo que el E-Business a los negocios ordinarios.

B.2 SHARABLE CONTENT OBJECT REFERENCE MODEL - SCORM

B.2.1 Organización de SCORM

SCORM es una colección de especificaciones y estándares que se han reunido dentro de una colección de "libros técnicos" (Ver figura 12). Cada uno puede verse como libros separados reunidos en una gran biblioteca. Casi todas las especificaciones y pautas se toman de otras organizaciones. Estos libros técnicos se agrupan bajo tres principales temas:

- Modelo de Agregación de Contenidos – Content Aggregation Model (CAM).
- Ambiente de Ejecución - Run-time Environment (RTE), y
- Secuenciación y Navegación - Sequencing and Navigation (SN).

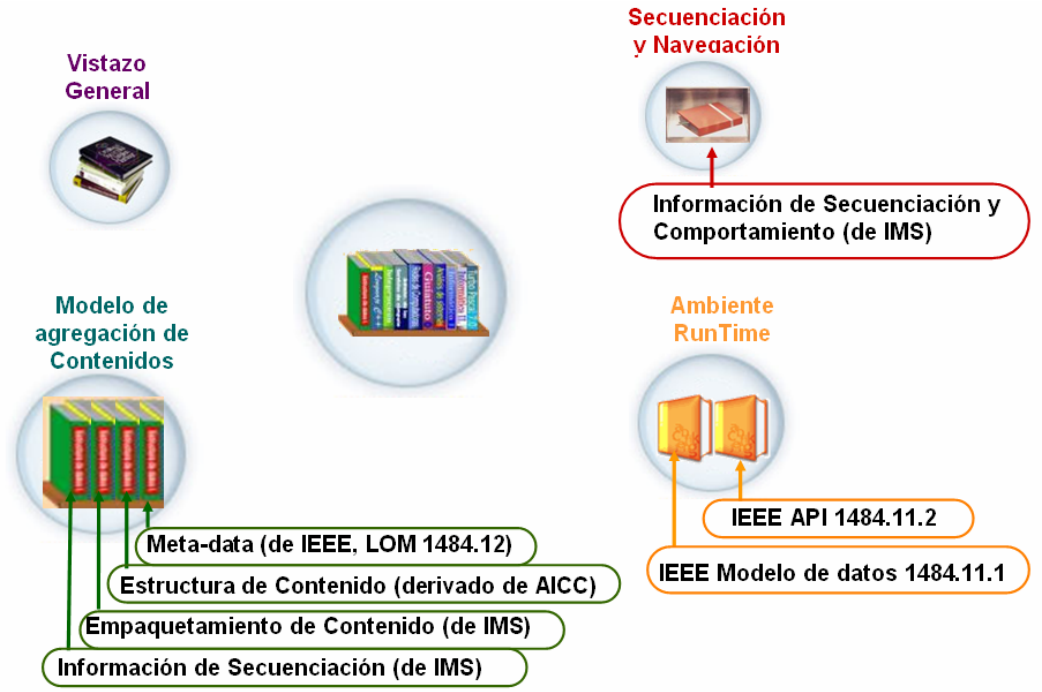


*Figura 12. Organización de SCORM (tomado de [**ADLOV**])*

B.2.1.1 Libro: Vistazo General de SCORM 2004

El libro Vistazo General de SCORM 2004 abarca la historia y objetivos de la iniciativa ADL y SCORM, incluyendo las especificaciones y los estándares que SCORM utiliza. También describe cómo los diferentes libros de SCORM se relacionan unos con otros.

B.2.1.2 Libro: Modelo de Agregación de Contenidos (CAM) de SCORM

El libro Modelo de Agregación de Contenidos (CAM), describe los componentes usados en una experiencia de aprendizaje, cómo se empaquetan estos componentes para intercambiarlos de sistema a sistema, cómo se describen estos componentes para habilitar búsquedas y hallazgos, y cómo definir las reglas de secuenciación para los componentes. El CAM proporciona almacenamiento consistente, etiquetado, empaquetado, intercambio y hallazgo de contenidos.

El libro CAM de SCORM, también define responsabilidades y requerimientos para construir agregaciones de contenidos (por ejemplo, cursos, lecciones, módulos, etc.). El libro contiene información sobre la creación de paquetes de contenidos, la aplicación de los metadatos de los componentes en el paquete de contenidos, la aplicación de los detalles de secuenciación y navegación en el contexto de un paquete de contenido.

Los metadatos SCORM describen los diferentes componentes del Modelo de Contenidos SCORM (agregación de contenidos, actividades, SCOs y Assets). Metadata, es una forma de etiquetado que mejora la búsqueda y descubrimiento de estos componentes.
Un paquete de contenidos, en sentido general, empaqueta objetos de contenido con una organización de contenidos, esto se describe en el manifiesto. Un paquete de Contenidos SCORM puede representar un curso, lección, módulo o puede ser simplemente una colección de objetos de contenidos relacionados. El Manifiesto, es una parte esencial de todos los Paquetes de Contenido SCORM, se define en el Lenguaje de Marcado Extensible (XML) y su nombre de archivo base es "imsmanifest.xml". Este archivo, describe los contenidos del paquete y puede incluir una descripción opcional de la estructura del contenido. (Ver Figura 13).

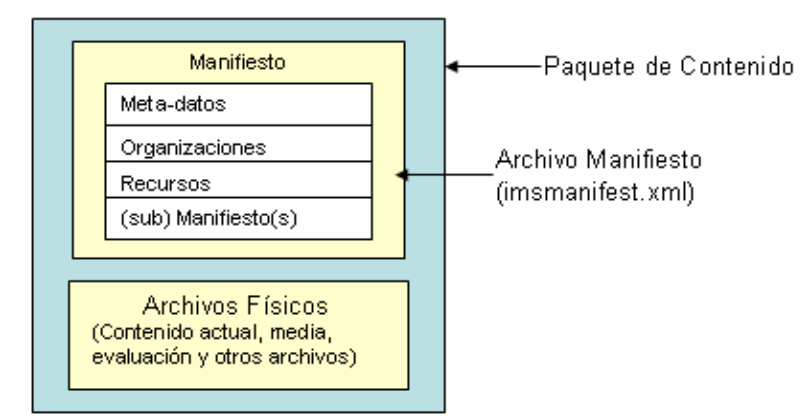


*Figura 13. Paquete de Contenidos (tomado de [**ADLOV**])*

Los Paquetes de Contenidos SCORM pueden incluir información adicional que describe cómo un LMS se proyecta para procesar el Paquete de Contenidos y sus contenidos. Algunos de estos elementos son utilizados por el modelo RTE de SCORM.

B.2.1.3 Libro: Ambiente de Ejecución (RTE) de SCORM

El libro RTE SCORM describe los requerimientos del Sistema Administrador de Aprendizaje (LMS) para gestionar el ambiente de ejecución (por ejemplo: procesos de lanzamiento de contenidos, comunicación entre contenidos y LMSs y elementos estándar del modelo de datos usados para entregar información acerca del aprendiz).

RTE abarca los requerimientos de los SCOs, el uso de la API y el ambiente de ejecución del modelo de datos SCORM. El propósito del RTE SCORM es proporcionar los medios para la interoperabilidad entre SCOs y LMSs. SCORM provee los medios para que el contenido de aprendizaje sea interoperable en múltiples LMSs, diferentes de las herramientas usadas para crear el contenido. Para que esto sea posible, debe existir una forma común para lanzar contenidos, una forma común para que el contenido se comunique con un LMS, y elementos de datos predefinidos que son intercambiados entre un LMS y el contenido durante su ejecución. Estos tres

componentes del RTE SCORM son definidos como Lanzador, Interfaz de Programa de Aplicación (API) y el Modelo de Datos. A continuación se presenta una breve descripción acerca de ellos.

- **Lanzador**, define la relación entre LMSs y contenidos SCORM, para que todo el contenido conforme SCORM confíe en un LMS conforme SCORM para ser entregado y desplegado al aprendiz. Con SCORM 2004, los LMSs han extendido las responsabilidades para determinar cuál contenido SCORM será entregado luego.
- **API SCORM**, provee un conjunto de funcionalidades predefinidas, que son acordadas por los vendedores de LMS y vendedores de la herramienta de autor de contenidos, para permitir la comunicación entre un LMS y los SCO lanzados. Estas funcionalidades complementan el proceso de lanzamiento por establecer un "protocolo" entre el SCO y el LMS que lo lanza. Además, esto permite que el contenido SCORM "fije" u "obtenga" datos en el LMS, como una evaluación de resultado, y para verificar y dirigir cualquier error que ocurra durante estos procesos.
- **El Modelo de Datos del Ambiente de ejecución SCORM,** provee el vocabulario que puede ser usado para transferir información, o para "fijar" y "obtener" datos desde y hacia un LMS cuando las funciones de la API SCORM son invocadas.

B.2.1.4 Libro: Secuenciación y Navegación (SN) de SCORM

El libro SN SCORM describe cómo el contenido conforme SCORM puede ser secuenciado a través de un conjunto de eventos de navegación iniciados por el aprendiz o por el sistema. La ramificación y circulación del contenido puede ser descrito por un conjunto predefinido de actividades, típicamente definidas en tiempo de diseño.

El libro SN SCORM también describe cómo un LMS conforme SCORM interpreta las reglas de secuenciación expresadas por el desarrollador del contenido a través de un conjunto de eventos de navegación y sus efectos en el ambiente de ejecución.

SN SCORM define un método para representar el comportamiento intencional de una experiencia de aprendizaje, de tal manera que cualquier LMS puede secuenciar continuamente las actividades de aprendizaje de una forma consistente.

SN SCORM modela los comportamientos requeridos y las funcionalidades que los LMSs conforme SCORM deben implementar para procesar la información de secuenciación en tiempo de ejecución. Más específicamente, describe la ramificación y circulación de las actividades de aprendizaje en términos de un Árbol de Actividades, basado en los resultados de las interacciones del aprendiz con los objetos de contenido y las estrategias de secuenciación.

Un Árbol de Actividades, es una estructura conceptual de actividades de aprendizaje gestionadas por el LMS para cada aprendiz. (Ver Figura 14). En SCORM, una actividad de aprendizaje puede referenciar objetos de contenidos que son entregados al aprendiz.

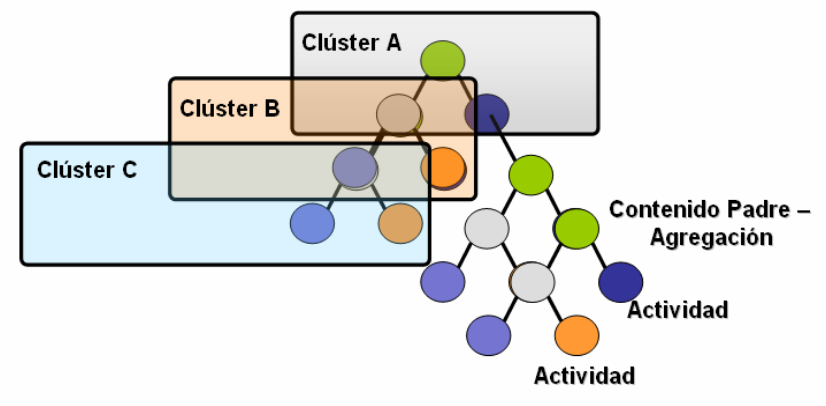


*Figura 14. Árbol de actividades (tomado de [**ADLOV**])*

El libro SN SCORM describe cómo los eventos de navegación iniciados por el aprendiz o por el sistema pueden ser disparados y procesados, resultando en la identificación de actividades de aprendizaje para entregar. Cada actividad de aprendizaje identificada para entregar tendrá un objeto de contenido asociado. El libro RTE SCORM describe cómo identificar objetos de contenidos lanzados.
La secuenciación de objetos de contenidos lanzados para un aprendiz y la estructura de contenidos proveen una experiencia de aprendizaje (interacción del aprendiz con los objetos de contenidos). El modelo RTE SCORM también describe cómo el LMS gestiona los resultados de la experiencia de aprendizaje y cómo ésta puede afectar el árbol de actividades.

B.2.1.5 Resumen de los libros SCORM

LIBROS QUE CUBRE SCORM			
Libro SCROM	**Conceptos que abarca**	**Lo clave de las tecnologías que cubre SCORM**	**Áreas de cubrimiento**
Vistazo General	Alto nivel de información conceptual.	Introducción a numerosos elementos de alto nivel de terminología SCORM.	Cubre áreas de los libros CAM, RTE y SN a un alto nivel.
Modelo de Agregación de Contenidos (CAM)	Ensamblado, etiquetado y empaquetado de contenidos de aprendizaje.	SCO, Assets, Agregación de Contenidos, Paquetes, Paquetes de Intercambio de Archivos (PIF), Metadatos, Manifiesto, Información de Secuenciación y Navegación.	SCOs y manifiestos, SCOs comunicados un LMS vía RTE. Manifiestos que contienen información de secuenciación y navegación.
Ambiente de Ejecución (RTE)	Gestión de LMSs y RTE, el cual incluye lanzamiento, comunicación de contenidos a LMS, seguimiento, datos de transferencia y error de encabezado.	API, instancia API, lanzados, métodos de sección, métodos de transferencia de datos, métodos de soporte, modelo temporal, Modelo de datos de tiempo de ejecución.	En el libro CAM son descritos los SCOs, objetos de contenidos, los cuales son usados por RTE.
Secuenciación y Navegación (SN)	Secuenciación y Navegación de contenidos.	Árbol de actividades, actividades de aprendizaje, información de secuenciación y navegación, modelo de datos de navegación.	Cómo la secuenciación y navegación afecta el contenido agrupado en un manifiesto.

*Tabla B.1. Resumen de los libros SCORM (tomado de[**ADLOV**])*

B.2.2 Objetos de Contenido Compartible - SCO

B.2.2.1 Etiquetado y empaquetado de objetos de aprendizaje

Para facilitar la búsqueda y descubrimiento de objetos de aprendizaje en un repositorio, y favorecer así su reúso, es necesario describir su contenido en una etiqueta y "pegarla" al objeto. La información contenida en la etiqueta de un objeto de aprendizaje es conocida como los metadatos (*metadata*) del objeto; es decir, datos acerca de los datos. Se trata de información descriptiva acerca del objeto.

La utilidad de los metadatos depende del uso de una nomenclatura común para describir el contenido de los objetos de aprendizaje. Los grupos de trabajo en estándares son los encargados de establecer la información suficiente y necesaria para describir un objeto de aprendizaje: qué contiene, dónde puede ser usado, nivel de dificultad, lenguaje, versión, etc. Entre los trabajos más importantes en este tema se encuentran los estándares emitidos por IMS, ADL y el IEEE.

No obstante, para alcanzar la interoperabilidad entre sistemas o pensar en una verdadera distribución del conocimiento (y del aprendizaje) se requiere que, además de que los sistemas "hablen" acerca de lo mismo, también usen un lenguaje común de comunicación. Aquí es donde entra el eXtensible Markup Language (XML), el lenguaje propuesto por el World Wide Web Consortium para reemplazar a HTML como el lenguaje del *Web.* [**W3C00**].

Finalmente, los objetos de aprendizaje son transmitidos entre diferentes sistemas o herramientas agrupados en paquetes de contenido educativo, correspondientes a lecciones, módulos, cursos, diplomados, etc. Una vez más, la organización de cada paquete se basa en estándares que facilitan el reúso y la interoperabilidad. Cada paquete tiene asociada una etiqueta en XML que describe características propias del paquete, particularmente la organización de los objetos de aprendizaje en el paquete y sus metadatos.

B.2.2.2 Normativas y estándares para el tratamiento de contenidos

Al hablar sobre un estándar *e-learning*, se refiere a un conjunto de reglas en común para todos los dedicados a la tecnología *e-learning*. Estas reglas especifican cómo los fabricantes pueden construir cursos en línea y las plataformas sobre las cuales son impartidos estos cursos de tal manera de que puedan interactuar unas con otras. Estas reglas proveen modelos comunes de información para cursos *e-learning* y plataformas de Sistemas de Gestión de Aprendizaje (LMS), que básicamente permiten a los sistemas y a los cursos compartir datos o "hablar" con otros. Esto también da la posibilidad de incorporar contenidos de distintos proveedores en un solo programa de estudios.

La obtención de un estándar formal se consigue como resultado de los esfuerzos combinados de numerosos organismos y consorcios que se agrupan de acuerdo a tres niveles de trabajo [**LIN01**]:

- **Nivel de especificación**: en este primer paso del proceso, se trabaja en la elaboración de recomendaciones basadas en el análisis de las necesidades de los propios participantes. El objetivo es proponer la especificación elaborada a la comunidad *e-learning*. Es en este nivel dónde se encuentra IMS (http://www.imsproject.org).
- **Nivel de validación**: en esta fase del proceso, se desarrollan nuevos productos que incorporan las especificaciones elaboradas en el paso anterior y se inician programas piloto con el fin de valorar la efectividad y aplicabilidad de la especificación. Así mismo, se crean modelos de referencia que muestran cómo las distintas especificaciones y estándares pueden ensamblarse para integrar un sistema *e-learning* global.

- **Nivel de estandarización**: es el paso final de la elaboración, las especificaciones que ya han sido validadas, son retomadas por los organismos oficiales de estandarización que se encargan de darles un último refinamiento, consolidación, clarificación de los requisitos que satisfacen y acreditación. Es importante distinguir entre la especificación (que es un proceso de trabajo en evolución) y el estándar acreditado (basado en implementaciones reales y las experiencias obtenidas).

Como ventajas de la estandarización se pueden numerar las siguientes [**EDU04**]:

- Garantizar el intercambio de contenidos entre diferentes entornos virtuales de formación.
- Permitir la búsqueda de contenidos por toda la red.
- Fomentar la profesionalización en la elaboración de contenidos.
- Iniciar sistemas de compra-venta de contenidos. Enlazar diferentes entornos de formación.

La estandarización de las tecnologías pretende facilitar la reutilización de recursos y la interoperabilidad entre sistemas y software heterogéneo.

Entre los trabajos más importantes se encuentran los estándares emitidos por IMS Global Learning Consortium (IMS), Advanced Distributed Learning Initiative (ADL), Institute of Electrical and Electronics Engineers (IEEE). Un elemento común y central en estos estándares es la propuesta de organizar el contenido educativo en la forma de objetos de aprendizaje.

Las normativas SCORM aportan un mecanismo que permite al contenido comunicarse con la plataforma. Este mecanismo se denomina **RTE** (RunTime Environment).

En el protocolo de comunicación, la normativa SCORM define un conjunto de campos/valores (**DATAMODEL**) que se pueden almacenar en la base de datos del servidor LMS.
Estos valores permiten:

- Personalizar el contenido (por ejemplo, visualizar una retroalimentación con el nombre de estudiante).
- Mejorar la navegación por el contenido (por ejemplo, guardar la ultima página vista).
- Registrar el seguimiento (guardar la puntuación para poder evaluar al estudiante).

B.2.2.2 El lenguaje universal - XML

Los entornos de E-Learning actuales se plantean como sistemas distribuidos y deben funcionar en tiempo real, tratar datos que provienen de sistemas heterogéneos, bases de datos, servicios de directorios y aplicaciones, e integrarse con otros entornos y sistemas empresariales.
El acceso a estos entornos se realiza por medio de multitud de dispositivos y navegadores diferentes.

Para poder integrar todas estas piezas se hace necesario un medio universal, neutro en cuanto a la plataforma, que permita describir, transportar y transformar datos entre los distintos sistemas distribuidos.

Todas las especificaciones, normas y estándares tecnológicos actuales usan para este propósito el lenguaje **XML**. Éste sirve para describir y transportar por la red las páginas Web (HTML). En el ámbito de E-Learning, es el lenguaje que se usa para describir prácticamente cualquier cosa.

La ventaja del uso del XML es su gran aceptación y el alto número de herramientas tecnológicas que hacen muy fácil el desarrollo y tratamiento informático (interpretar un documento XML).

B.2.2.2 Cómo distribuir contenidos: Empaquetado e Intercambio de Contenidos

Como respuesta a esta necesidad se han desarrollado varias normas; las más relevantes se enumeran a continuación:

- AICC: http://www.aicc.org/
- IMS Content Packaging Specification: http://www.imsglobal.org/
- SCORM: http://www.adlnet.org/

Ejemplo paquete SCORM / IMS

El paquete SCORM es un fichero ZIP (comprimido) que contiene:

- Uno o varios ficheros que describen por medio de XML el índice de los contenidos de un paquete (estos ficheros se denominan MANIFEST) que, además, aporta información sobre la forma de presentarlos.
- La información que permite habilitar contenidos en función de determinados resultados del estudiante en los ítems de contenido (prerrequisitos).
- Los recursos educativos, ficheros HTML, imágenes, etc.

¿Cómo funciona?

- El proveedor crea el índice de contenidos con las herramientas de producción de contenidos. Estas herramientas disponen de funcionalidades para empaquetar contenidos según las normas SCORM.
- El proveedor envía el empaquetado (curso.zip) al cliente.
- El cliente da de alta el curso en su plataforma e importa el empaquetado (funcionalidad estándar en sistemas compatibles SCORM), la plataforma copia los ficheros en el disco duro del servidor y da de alta en la base de datos el índice de contenidos del curso importado.
- El estudiante accede a la plataforma y selecciona el curso. Por su parte, la plataforma representa el índice de contenidos del curso con los enlaces para acceder al recurso correspondiente a cada ítem de contenido.

¿Cómo describir contenidos?

Conforme crece el número de contenidos almacenados en un repositorio, se hace necesario tener mecanismos que permitan la búsqueda, clasificación y organización de los contenidos.
La información sobre la información se denomina **MetaData**.
Para abordar este problema, el grupo de trabajo de IEEE LOM (Institute of Electrical and Electronic Engineers. Learning Object Metadata) ha creado un estándar para los Metadatos que hacen referencia al ámbito educativo. Este modelo fue adoptado por IMS y por ADL con algunos ajustes y modificaciones.

Ejemplo Metadata de SCORM / IMS

El mecanismo propuesto por IMS consiste en un conjunto de etiquetas XML que se añaden a los archivos XML del empaquetado y aportan la descripción de los recursos y los ítems del contenido, distribuyéndose en el empaquetado.

¿Cómo funciona?

El proveedor utiliza las herramientas de autoría para rellenar los campos Metadata asociados con los recursos e ítems del contenido.
La descripción de recursos sirve asimismo en el proceso de producción de los contenidos y permiten al proveedor gestionar eficazmente los recursos utilizados por todos los profesionales implicados en el proceso.
Cuando se genera el empaquetado, los Metadatos se añaden en formato XML. De esta forma, el fichero de empaquetado incluye Recursos, Estructura del Contenido y la descripción de los mismos.

Cuando el cliente importa el paquete de contenidos a la plataforma, los Metadatos se incorporan a la base de datos.

B.2.2.3 Cómo "hablar" con la plataforma / entorno de tiempo de ejecución

Normalmente se espera que los contenidos se comuniquen con el sistema de gestión para así recopilar los datos sobre la formación realizada por el estudiante. Igualmente, a menudo se precisa personalizar los contenidos en función de su perfil, resultados obtenidos entre otros. Esta comunicación implica una integración entre el contenido y el sistema de gestión y hace que el contenido sea dependiente de las tecnologías con las que se haya desarrollado el entorno de formación.
Con el propósito de que el contenido sea interoperable e independiente de las tecnologías del entorno que lo contiene, se han desarrollado varias normas:

- AICC (HACP y API)
- ADL SCORM (API)
- IEEE (API)

Ejemplo ADL SCORM
La propuesta de ADL dentro de su modelo SCORM persigue proporcionar un medio estándar para la interoperabilidad entre los objetos de contenidos (SCO), y los sistemas de gestión de aprendizaje (LMS).
Uno de los requerimientos de SCORM es que el contenido sea operativo en todos los entornos compatibles SCORM sin tener en cuenta las herramientas que se utilicen para crear o usar los contenidos.
Para que esto sea posible, ADL establece:

- Unos requerimientos (responsabilidades) a nivel de contenidos (SCO) y nivel de la entornos de gestión (LMS)
- Un mecanismo (interface) estándar de programación (modelo PLUGIN). El acceso a este mecanismo se realiza por medio del lenguaje JavaScript, que es un estándar en los entornos Web.
- Un conjunto de datos (DATAMODEL) que se pueden intercambiar y deben ser soportados por todas las plataformas (por ejemplo: puntuación, estado del aprendizaje, etc.)

¿Cómo funciona?
El proveedor usa herramientas que añaden programación JavaScript en determinadas partes del contenido para que comunique a la plataforma la puntuación del estudiante en cada unidad.
El cliente publica el contenido en la plataforma.
El estudiante entra en el curso y realiza las actividades obteniendo una puntuación. El curso ejecuta la programación JavaScript, comunicando a la plataforma la puntuación obtenida. La plataforma recoge la puntuación comunicada y la guarda en su base de datos.
El gestor de formación revisa las puntuaciones del estudiante en las unidades para poder dirigir el proceso de formación.

B.2.2.4 Cómo secuenciar contenidos: Secuenciación (IMS,SCORM)

Esta especificación describe por medio de unas etiquetas XML las reglas para controlar el flujo de las actividades educativas basándose en los resultados obtenidos por los estudiantes en sus interacciones con los contenidos didácticos.

Esto permite reutilizar los mismos contenidos creando distintas secuencias de navegación, así como experiencias formativas basadas en los resultados formativos de cada estudiante. Además, igual que las otras especificaciones de IMS, esta información de secuenciación debe ser intercambiable entre diferentes sistemas educativos con ayuda de herramientas de exportación e

importación.

B.2.3 Principales organizaciones involucradas en el proceso de estandarización.

B.2.3.1 LTSC (Learning Technology Standardization Committee)

El comité de estandarización de tecnologías aplicadas al aprendizaje, LTSC (*Learning Technology Standardization Committee*), perteneciente al *Institute of Electrical and Electronics Engineers* (IEEE), abarca prácticamente todos los aspectos del aprendizaje basado en computador.

Su principal misión es "desarrollar estándares técnicos, prácticas recomendadas y guías para componentes software, herramientas, tecnologías y métodos de diseño que faciliten el desarrollo, implantación, mantenimiento e interoperabilidad de sistemas educativos basados en computadores".

Los 15 subcomités que componen el LTSC están organizados en 5 áreas de trabajo:

1. Aspectos generales,
2. Aspectos relacionados con el contenido,
3. Aspectos relacionados con el aprendizaje,
4. Datos y metadatos y
5. Sistemas y aplicaciones de gestión.

Las primeras propuestas del LTSC están relacionadas con una arquitectura de sistema y un modelo de referencia para un sistema gestor de aprendizaje (LMS), un modelo de datos para describir el proceso de aprendizaje y metadatos educativos.

B.2.3.2 IMS

El proyecto IMS (IMS, 2003a; IMS, 2003b) fue creado en 1997 por EDUCASE, un consorcio de instituciones educativas y sus socios empresariales como una tentativa de desarrollar estándares abiertos para sistemas de enseñanza asistida por computador.

Los primeros trabajos del IMS se centraron en la definición de un modelo y una arquitectura para sistemas de aprendizaje distribuido. Sin embargo, sus esfuerzos se reorientaron rápidamente al percatarse de que primero necesitaban un modelo de datos adecuado para describir los recursos, estructuras y demás elementos manejados por los componentes de la arquitectura. Hoy en día los trabajos de IMS se centran en: metadatos, empaquetado de contenidos, definiciones de tests y cuestionarios, especificaciones de perfiles de estudiantes y gestión de grupos, recomendaciones para objetos educativos distribuidos, organización de cursos bajo enfoques pedagógicos y secuenciamiento.

B.2.3.3 ADL (Advanced Distributed Learning)/SCORM (Sharable Content Object Reference Model)

En noviembre de 1997 el Departamento de Defensa de los EE.UU. y la oficina de Ciencia y Tecnología de la Casa Blanca lanzaron la iniciativa ADL. ADL surge como respuesta a las necesidades de uno de los mayores consumidores de software del mundo y forma parte del esfuerzo que el gobierno norteamericano viene realizando con el objetivo de conseguir una enseñanza de calidad, en el que también están implicados los departamentos de Educación y Trabajo.

ADL se ha centrado desde un principio en el aprendizaje sobre Web. Su trabajo ha acompañado al de otras instituciones, principalmente IEEE, IMS y AICC, para buscar aquellos puntos críticos del aprendizaje sobre Web en los que sería recomendable especificar la interfaz consensuada.

ADL ha sido una de las organizaciones más activas en las reuniones y encuentros de estos organismos y fruto de ello es un conjunto de especificaciones que, bajo la denominación SCORM, modelo de referencia para objetos educativos software que puedan compartirse, propone un entorno de ejecución y un modelo de metadatos y estructuras de cursos. La versión 1.1 de SCORM fue publicada el 31 de enero de 2001.

ADL espera que el ámbito de aplicación de la especificación se vea ampliado en el futuro y que sirva como punto de convergencia para el resto de recomendaciones de otras instituciones implicadas en el proceso de estandarización. En 1999 el ADL estableció el Co-Lab como grupo de trabajo responsable de comprobar y validar las nuevas recomendaciones de la iniciativa ADL, verificar el grado de cumplimiento del estándar por parte de productos comerciales externos y realizar las certificaciones correspondientes. Asimismo, este grupo es el responsable de desarrollar prototipos conformes a SCORM y divulgar los resultados obtenidos.

A día de hoy, ADL es el modelo de referencia para las especificaciones de IMS, que a su vez es el productor de especificaciones para ADL.

B.2.3.4 AICC

El comité para CBTs (educación basada en el uso del computador -*ComputerBased Training)* de la industria de la aviación, *Aviation Industry* CBT *Committe* (AICC) (AICC, 1995; AICC, 1997) aparece como respuesta natural a las necesidades de una industria que consume una gran cantidad de software educativo para la formación de sus alumnos de piloto.

Las recomendaciones del AICC son publicadas en tres tipos de documentos: recomendaciones y guías AICC, e informes técnicos y documentos de trabajo. Los AGRs son documentos cortos que representan la postura oficial del AICC en las diferentes áreas que son objeto de estandarización.

Los trabajos del AICC contemplan, entre otros, la definición de requisitos hardware y software para los computadores de los estudiantes, los periféricos necesarios, los formatos aceptados para los elementos multimedia que componen los cursos, así como recomendaciones para las interfaces de usuario. Otra de sus principales aportaciones es su propuesta para entornos de ejecución. La recomendación del AICC en este sentido contempla sistemas autónomos en donde la comunicación es realizada a través de ficheros; sistemas de aprendizaje para Web, con una interfaz definida sobre el protocolo HTTP (*Hypertext Transfer Protocol*), Protocolo de Transferencia de Hipertexto; y finalmente, un esquema basado en una interfaz de programación que hace transparente el protocolo subyacente.

B.2.3.5 OKI (Open Knowledge Iniciative)

El proyecto OKI está patrocinado por la Fundación Mellon durante un periodo inicial de dos años. Liderado por el Instituto Tecnológico de Massachussets (*Massachussets Institute of Technology*) MIT, en colaboración con la Universidad de Stanford, cuenta con la participación de varias universidades americanas, entre ellas la Universidad de Dartmouth, la Universidad de Harvard, la Universidad de Carolina del Norte, la Universidad de Michigan, la Universidad de Pensilvania y la Universidad de Wisconsin.

El objetivo principal del proyecto cubre lo que en estos momentos es una necesidad básica para la comunidad educativa en la Web: diseñar y desarrollar una arquitectura abierta y extensible para los Sistemas de Gestión del Aprendizaje (*Learning Management Systems, LMS*).

Como resultado del proyecto se pretende elaborar una especificación que sirva como marco para la arquitectura de un LMS y desarrollar un conjunto de herramientas que implementen dicha arquitectura y prueben la viabilidad de la especificación.

La arquitectura OKI se caracteriza por dos capas independientes, compuestas de un conjunto de servicios básicos que se implementan mediante APIs *(Aplication Programming Interface*). La primera de estas capas se denomina Servicios Comunes y da soporte al conjunto de servicios básicos o de infraestructura de un LMS (administrativos, de autenticación, de definición de roles de usuario, etc.). En la segunda capa, Servicios Educativos, se define el conjunto de servicios que tienen que ver con la función educativa del LMS. El objetivo de conseguir un conjunto de APIs que no esté ligado a un servicio concreto, es crear una capa de enlace que independice el software educativo de la infraestructura, de tal manera que los cambios en un módulo determinado no afecten al resto. Cada uno de los servicios está también implementado como un módulo independiente que no comparte con el resto objetos ni interfaces.

Al definir un LMS como un conjunto de servicios básicos implementados mediante APIs independientes, el programador de la aplicación no necesita conocer los detalles particulares de cómo está implementado un determinado servicio. Además es posible realizar varias implementaciones de un mismo servicio sin modificar la aplicación: mientras la implementación de un determinado servicio mantenga su API; las implementaciones de un servicio pueden modificarse sin requerirse ningún cambio en la aplicación que utiliza el API.

Las organizaciones que lideran el desarrollo de especificaciones para la tecnología de enseñanza han firmado un acuerdo de colaboración y coordinación de sus actividades. Esta coalición informal incluye a ADL, OKI, IMS y SIF (*Schools Interoperability Framework*). Este grupo intenta formalizar sus actividades para conseguir enfocar sus esfuerzos hacia la consecución de un estándar común.

B.2.3.6 Iniciativas Europeas

Dentro de la Comunidad Europea se pueden identificar iniciativas relacionadas con la estandarización de la educación basada en computadores:

1. ARIADNE (http://www.aramisresearch.ch/e/6541.html,
2. GESTALT (http://www.fdgroup.co.uk/gestalt/),
3. PROMETEUS (http://prometeus.org/),
4. CEN/ISSS/LT (WS-LT, 2002) y
5. UNFOLD (http://www.unfold-project.net, http://www.mx.educaterra.com).

La alianza de redes europeas para la creación y distribución remota de contenidos para el aprendizaje, (***Alliance of Remote Instructional Authoring and Distribution Networks for Europe - ARIADNE***) se incluye dentro de las iniciativas del 4º programa marco de la Unión Europea.

Los principales campos de trabajo de ARIADNE son: telemática para educación y aprendizaje, metodologías para la creación, gestión y reutilización de elementos pedagógicos basados en computador, definición de programas de estudio basados en soportes telemáticos, y metadatos educativos.

Una de las principales contribuciones de esta iniciativa es una propuesta de metadatos educativos desarrollada en colaboración con el IMS.

Otra iniciativa europea es el proyecto **GESTALT, *Getting Educational Systems Talking Across Leading edge Technologies***. El sistema GESTALT pretende establecer un marco para el desarrollo de sistemas educativos distribuidos, heterogéneos, escalables y compatibles.

El objetivo genérico de la plataforma es permitir a los usuarios de un sistema de aprendizaje descubrir la existencia de recursos educativos, acceder a una referencia a esos recursos y entregarlos mediante una infraestructura de red convenientemente gestionada.

GESTALT ha realizado importantes aportaciones a la definición de modelos de datos para sistemas educativos basados en computador y redes telemáticas, en concreto en el campo de los metadatos y descripción de perfiles y preferencias de usuario. Ambos modelos han sido utilizados por los servicios de intermediación entre los proveedores de recursos educativos y sus consumidores, servicios que forman parte de la arquitectura de GESTALT.

La iniciativa **PROMETEUS, *Promoting Multimedia access to Education and Training in European Society***, reúne a más de 400 instituciones que buscan cubrir el hueco existente entre la investigación y las necesidades reales de los CBTs en el ámbito europeo. Hasta el momento han establecido 11 grupos de trabajo que tendrán como misión la publicación de guías de referencia, libros de referencia y recomendaciones sobre estándares en aquellos temas en los que están centrados. El modo de trabajo de PROMETEUS consiste en la utilización de listas de correo electrónico a través de las cuales cada uno de los grupos realiza las discusiones preliminares a los encuentros presenciales.

La iniciativa **UNFOLD**, es un proyecto de la Unión Europea para la adopción de estándares avanzados de *e-learning*. UNFOLD centra sus esfuerzos en IMS *Learning Desing* que es la única especificación abierta que proporciona soporte a múltiples usuarios y diferentes pedagogías. Un aspecto fundamental para el desarrollo del *e-learning* es que sea mejor aprendizaje. Progresar hacia este objetivo depende de la adopción de códigos abiertos. UNFOLD hará: hacer comprender, en el menor tiempo posible, las ventajas de los estándares abiertos; crear un motor para el desarrollo continúo de las prácticas *e-learning* en Europa y proveer de un modelo para la rápida realización de todos los beneficios de cualquier especificación. Las instituciones europeas involucradas en el proyecto son: La universidad Pompeu Fabra de Barcelona, España, Open University, Nederlands y Bolton Institute, UK.

Dentro del Comité Europeo para la Estandarización *(Comité Européen de Normalization, CEN*) se ubica el Sistema de Estandarización para la Sociedad de la Información (*Information Society Standardization System, ISSS*). Las actividades relacionadas con la estandarización educativa se desarrollan dentro del grupo de trabajo de tecnologías de aprendizaje (*Learning TechnologiesWorkshop, WS-LT*). Los esfuerzos principales de este grupo se centran en la reutilización e interoperabilidad de los recursos educativos, la colaboración en el aprendizaje, metadatos para contenidos educativos y calidad del proceso de aprendizaje (WS-LT, 2002).

B.2.3 AGENTES INTELIGENTES

B.2.3.1 ¿Qué es un Agente?

A pesar que los agentes inteligentes se han convertido en un área tecnológica que crece rápidamente, aún no existe un acuerdo sobre qué es exactamente un agente. En nuestro estudio encontramos muchas definiciones, la mayoría de estas se enfatizan en las características que poseen los agentes. Algunas de estas definiciones son:

Según Genesereth, “Una entidad es un agente software si y sólo si se comunica correctamente en un lenguaje de comunicación de agentes” [**GEN95**].

Para Coen (agente SodaBot): "los agentes son programas que se comprometen en diálogo, negocian y coordinan transferencia de información". [**COH89**]
Dado que estas definiciones solo se centran en un único aspecto como es la comunicación, se quedan cortas, además podríamos pensar que un humano se comunica con una máquina con un lenguaje de agentes, en ese tal caso ¿Quién seria el agente?, la máquina, el humano o los dos.

IBM afirma que: "los agentes inteligentes son entidades software que realizan algún conjunto de operaciones en beneficio de un usuario o de otro programa con cierto grado de independencia o autonomía, y realizando esto, emplean cierto conocimiento o representación de los deseos y objetivos del usuario", esta definición introduce características importantes como independencia y autonomía que otros paradigmas de programación no poseen, además habla de entidades que actúa en favor de otros con base al conocimiento que poseen sobre la tarea que desean realizar.

Rusell y Norving, definen agente como: "Un agente es cualquier cosa que puede ser vista como algo que percibe su ambiente a través de sensores y actúa sobre su ambiente a través de efectores" [**RUS95**].

Para Maes: "Los Agentes autónomos son sistemas computacionales que habitan en algún ambiente dinámico, complejo, monitorean y actúan de forma autónoma en este ambiente, y realizando esto logran un conjunto de objetivos o tareas para los cuales están diseñados" [**MAES95**].

Las anteriores definiciones se basan en el hecho que los agentes se encuentran dotados de un conjunto de sensores, que le ayudan a percibir el estado actual del ambiente en el que están inmersos y un conjunto de efectores que le ayudan a modificar el estado actual de dicho ambiente, como lo muestra la gráfica siguiente. (Figura 15).

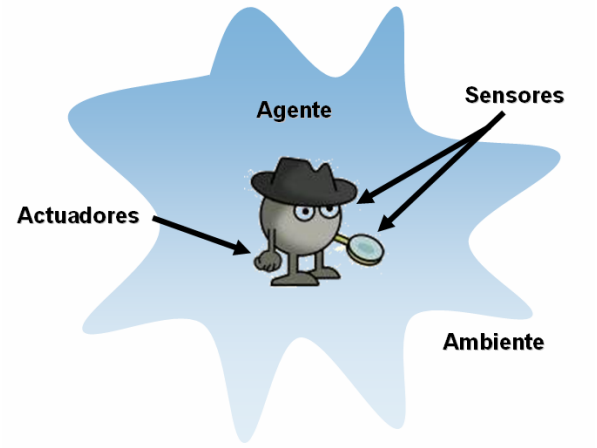


Figura 15. Interacción agente - ambiente

La definición de *Hayes-Roth* incluye implícitamente los conceptos anteriormente mencionados: "Los agentes inteligentes continuamente realizan tres funciones: percepción de condiciones dinámicas en el ambiente; acción para afectar condiciones en el ambiente; y razonamiento para interpretar las percepciones, resolver problemas, realizar inferencias, y determinar acciones" [**HAY95**]. En esta definición el comportamiento se presenta como una parte importante del agente, además se añaden una serie de pasos a seguir: monitorear eventos, razonar sobre los mismos, inferir posibles acciones y finalmente ejecutar alguna acción que puede ser en beneficio de un usuario.

Por último, la definición de Smith (el agente KidSim), introduce un nuevo componente: el conocimiento persístete. “Definamos un agente como una entidad software persistente dedicada a un propósito específico. ‘Persistente’ distingue a un agente de las subrutinas; los agentes tienen sus propias ideas acerca de cómo cumplir las tareas, su propia agenda. ‘Propósito específico’ los distingue de todas las aplicaciones multifunción; los agentes son típicamente más pequeños” [**SMI94**].

Todos estos autores adjudican más o menos las mismas características a los Agentes, en la mayoría de estas definiciones está presente la autonomía como característica principal, además mencionan el hecho que los agentes habitan determinado ambiente que puede ser estático o dinámico, algunas definiciones mencionan aspectos de sociabilidad, lo que introduce la necesidad de comunicación de los agentes con otros agentes y también con el usuario humano, además la definición del agente KidSim introduce el concepto de persistencia, la figura 16 agrupa las nociones que tratan de expresar las anteriores definiciones:

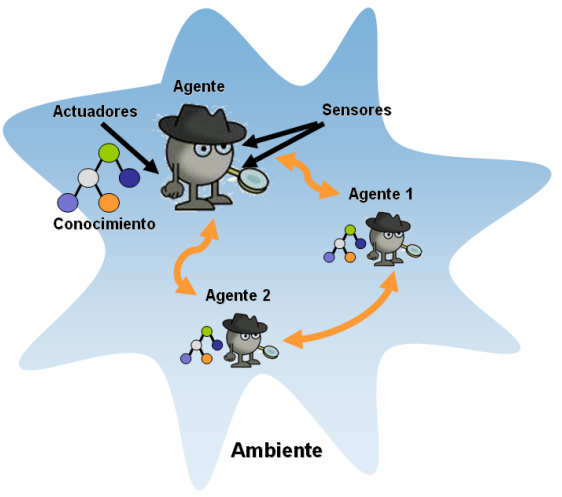


Figura 16. Características de los agentes

B.2.3.1 Arquitectura de Agentes

B.2.3.1.1 Arquitecturas para Agentes Deliberativos

Wooldridge define una arquitectura de agente deliberativo, como aquella que “contiene un mundo representado explícitamente y un modelo lógico del mismo, en la cual las decisiones son hechas por medio de un razonamiento lógico, basado en concordancia de patrones y manipulación simbólica” [**WOO95**], a continuación se presentan algunas de las arquitecturas de agentes deliberativas más conocidas:

Planning Agents

Desde inicios de los setenta, la comunidad de la Inteligencia Artificial dedicada al Planning (que es esencialmente programación automática; es decir, diseño de un curso de acción que, cuando ejecutado, resulta en el logro de algún objetivo deseado) ha estado fuertemente relacionado con el diseño de agentes. Parece razonable entonces, que muchas de las innovaciones en el diseño de agentes provengan de esa comunidad.

Tal vez el sistema mejor conocido basado en planning es STRIPS [**FIK71**]. La arquitectura de este sistema, contiene una descripción simbólica del mundo y el estado deseado de objetivos, y del conjunto de descripciones de las acciones que caracterizan las pre y post condiciones asociadas

con varias acciones. Tomando como base este conocimiento intenta encontrar una secuencia de acciones que lograrán el objetivo, utilizando un análisis simple. El algoritmo de planeación de STRIP es muy sencillo y se probó que es ineficiente en problemas de complejidad moderada.

Se han realizado varios intentos para construir agentes cuyo componente principal sea un planeador. Por ejemplo: el sistema Integrated Planning, Execution and Monitoring (IPEM) que está basado en un planeador sofisticado no lineal [**AMB88**]; el sistema AUTODRIVE que tiene planning agents operando en un ambiente altamente dinámico [**WOO93**]; el sistema PHOENIX que incluye agentes basados en planeadores [**COH89**].

IRMA

Intellince Resource-bounded Machine Architecture (IRMA) [**BRA98**]. Esta arquitectura tiene cuatro estructuras claves, de datos simbólicos: una librería de planes, una representación explícita de creencias, deseos e intenciones. Adicionalmente, la arquitectura tiene: un razonador, para razonamiento acerca del mundo; un analizador, para determinar qué planes deben ser utilizados para lograr las intenciones del agente; un filtrador de procesos; y un proceso de deliberación. El proceso de filtración es responsable de determinar el subconjunto de cursos de acción potenciales del agente, que tienen la propiedad de ser consistentes con las intenciones actuales del agente. La elección entre opciones que compiten lo realiza el proceso de deliberación.

HOMER

En [**VER90**] se argumenta que el establecimiento de tecnologías para agentes inteligentes está lo suficientemente desarrollada, como para construir un agente prototipo autónomo con habilidades lingüísticas, capacidades de planeación y acción, etc. Se desarrolló tal agente y se lo llamó HOMER. La arquitectura de HOMER es específica para el problema de la comunicación en lenguaje natural con el usuario; para ello incluye una base de vocabulario limitada a un subconjunto del idioma Inglés con cerca de 800 palabras y una memoria episódica también limitada. HOMER toma instrucciones del usuario que pueden contener referencias temporales moderadamente sofisticadas; puede planear cómo realizar sus instrucciones, y puede luego ejecutar sus planes, modificándolos como sea requerido durante la ejecución. El agente además, gracias a su memoria episódica, es capaz de responder a preguntas acerca de sus experiencias pasadas.

GRATE

GRATE es una arquitectura en capas en la que el comportamiento de un agente es guiado por actitudes mentales tales como creencias, deseos e intenciones colectivas [**JEN93**]. Los agentes se dividen en dos partes distintas: un sistema de nivel de dominio y una capa de cooperación y control. El primero resuelve problemas para la organización, sea esta en el dominio de controles industriales, de finanzas o transporte. El último es un controlador de meta nivel que opera en el nivel de dominio del sistema con la intención de asegurar que las actividades del agente a nivel del dominio sean coordinadas con aquellas otras dentro de la comunidad. La capa de cooperación está compuesta de tres módulos genéricos: un módulo de control que es la interfaz con la capa de dominio del sistema, un módulo de fijación de situaciones y un módulo de cooperación. Los módulos de fijación y cooperación proveen una implementación de un módulo de responsabilidad colectiva, que especifica cómo los agentes deberían actuar tanto localmente como hacia otros agentes comprometidos a cooperar en la resolución del problema.

BDI

Este tipo de arquitectura ve al sistema como un agente racional que tiene ciertas actitudes mentales, tales como: creencias, deseos e intenciones, representando respectivamente, los estados de información, motivacional y deliberativos del agente. Estas actitudes mentales

determinan el comportamiento del sistema y son críticos para lograr el desempeño adecuado u óptimo cuando la deliberación está sujeta a recursos limitados. [**GEO95**].

Las creencias de un agente representan el conocimiento del agente. El contenido del conocimiento puede ser cualquiera, por ejemplo conocimiento acerca del ambiente del agente o acerca de su historia. Los deseos son un conjunto de objetivos a largo plazo. Un objetivo es típicamente una descripción de un estado deseado del ambiente. Los deseos proveen al agente de la motivación para actuar. Los objetivos constituyentes de los deseos pueden ser contradictorios, entonces el sistema tiene que poder, de cierta forma, elegir qué objetivo alcanzar primero, es aquí donde aparecen las intenciones. Las intenciones pueden ser consideradas como un conjunto de planes para lograr los objetivos que constituyen los deseos.

Como se apunta en [**GEO95**], no es necesario, que un sistema especificado en términos de creencias, deseos e intenciones sea diseñado por medio de estructuras de datos correspondientes a cada uno de estos componentes. Sin embargo tal diseño podría simplificar la construcción, mantenimiento y verificación del sistema resultante.

B.2.3.1.2 Arquitecturas para Agentes Reactivos

Una arquitectura para agente reactivo es aquella que no incluye ningún tipo de modelo simbólico central del mundo, y no utiliza razonamiento simbólico complejo, algunas de las arquitecturas de agentes reactivos mas conocidas son:

Sumbsumption Architecture

Consiste en una jerarquía de comportamientos de logro de tareas [**BRO86**]. Cada comportamiento 'compite' con otros para ejercer control sobre el agente. Capas menores representan comportamientos de tipo más primitivo, (tal como evitar obstáculos, por ejemplo), y tienen precedencia sobre las capas superiores de la jerarquía. El sistema resultante es, en términos de porcentaje de computación que necesitan realizar, extremadamente simple, sin un razonamiento explícito del tipo encontrado en los sistemas simbólicos de inteligencia artificial.

PENGI

Chapman y Agre observaron en [**CHA86**], que la mayoría de las actividades cotidianas son 'rutinas', en el sentido que se requiere poco – o ningún - nuevo razonamiento. La mayoría de las tareas, una vez aprendidas, pueden ser desarrolladas de una manera rutinaria, con poca variación. Agre propuso que una arquitectura de agente eficiente podría ser basada en la idea de 'running arguments'. En forma cruda, la idea es que, como la mayoría de las decisiones son rutinarias, pueden ser codificadas a través de una estructura de bajo nivel (tal como un circuito digital), que sólo necesita actualización periódica, para manejar nuevos tipos de problemas. Su enfoque fue ilustrado con el sistema PENGI. PENGI es un juego de computadora simulado, con un sistema central controlado utilizando.

Situated Autómata

En el paradigma *situated autómata* [**KAE86**], un agente se especifica en términos declarativos. Esta especificación se compila luego a una máquina digital, que satisface la especificación declarativa. Esta máquina digital puede operar de una manera 'time-bounded'; no realiza ningún tipo de manipulación simbólica, y de hecho ninguna expresión simbólica se representa en la máquina. La lógica utilizada para especificar un agente es esencialmente una lógica modal de conocimiento. Un agente se especifica en términos de dos componentes: percepción y acción. Dos programas se utilizan para sintetizar a los agentes: RULER se emplea para especificar el componente referente a la percepción de un agente; mientras que GAPPS para el componente referente a la acción del agente.

RULER toma como entrada tres componentes:
'Una especificación de la semántica de las entradas del agente; un conjunto de hechos estáticos; y una especificación del estado de transiciones del mundo. El programador luego especifica la semántica deseada para la salida, y el compilador... sintetiza un circuito cuyas salidas tendrán la semántica correcta. ... Todo el conocimiento declarativo se reduce a un circuito bastante simple'

GAPPS toma como entrada un conjunto de reglas de reducción de objetivos, (esencialmente reglas que codifican información acerca de cómo los objetivos pueden ser satisfechos), y un objetivo de alto nivel, y genera un programa que puede ser traducido a un circuito digital para realizar el objetivo. Nuevamente, el circuito generado no representa o manipula expresiones simbólicas; toda la manipulación simbólica se realiza en tiempo de compilación.

B.2.3.1.3 Arquitectura de Red de Agentes

Pattie Maes ha desarrollado una arquitectura de agentes en la cual un agente se define como un conjunto de módulos de competencia [**MAES91**]. Estos módulos asemejan ligeramente el comportamiento de la arquitectura subsumption. Cada módulo es especificado por el diseñador en términos de pre- y post- condiciones, y un nivel de activación, que da una indicación real de la relevancia del módulo en una situación particular. Cuanto más alto es el nivel de activación de un módulo, hay mayor posibilidad que ese módulo influenciará el comportamiento del agente. Una vez especificado, un conjunto de módulos de competencia se compilan en una red de activación esparcida, en la cual los módulos se enlazan unos a otros de manera definida por las pre- y post-condiciones. El resultado de la ejecución puede ser un comando a una unidad efectora, o tal vez el aumento del nivel de activación de un módulo sucesor.

Existen obvias similitudes entre una arquitectura de red de agentes y arquitecturas de redes neurales. Tal vez la diferencia clave es que es difícil especificar cual es el significado de un nodo en una red neural; sólo tiene un significado en el contexto de la red en sí. A pesar de que los módulos de competencia se definen en términos declarativos, sin embargo, es mucho más fácil especificar cual es su significado.

B.2.3.1.2.4 Arquitecturas para Agentes Híbridos

Muchos investigadores [**GEO87**,**FER92**,**BUR92**] han sugerido que ni un enfoque completamente deliberativo ni uno completamente reactivo es adecuado para construir agentes. Argumentaron el caso de sistemas híbridos, que intentan unir los enfoques deliberativos y reactivos.

Un enfoque obvio es construir un agente compuesto por dos subsistemas: uno deliberativo, que contiene un módulo simbólico del mundo, que desarrolla planes y efectúa decisiones de la manera propuesta por la inteligencia artificial simbólica; y uno reactivo, que es capaz de reaccionar a eventos que ocurren en el ambiente sin necesitar un razonamiento complejo. A menudo, al componente reactivo se le da cierto grado de precedencia sobre el deliberativo, de tal manera a proveer una pronta respuesta a eventos ambientales importantes. En una arquitectura tal, los sistemas de control del agente se arreglan en una jerarquía, con las capas más altas tratando con información de mayor nivel de abstracción. Así, por ejemplo, las capas inferiores pueden mapear datos directamente a los efectores de salida, mientras que las capas superiores tratan con objetivos a largo plazo, las arquitecturas hibridas más conocidas son:

PRS

De la misma manera que IRMA, el PRS [**GEO87**] es una arquitectura basada en creencias, deseos e intenciones, que incluye una librería de planes, así como una explícita representación simbólica de las creencias, deseos e intenciones. Las creencias son hechos, tanto acerca del mundo exterior como del estado interno del sistema. Estos hechos son expresados mediante la clásica lógica de primer orden. Los deseos son representados como comportamientos del sistema (antes que como

una representación estática de estados de objetivos). Una librería de planes de un PRS contiene un conjunto de planes parcialmente elaborados, llamados áreas de conocimiento (*knowledge areas KAs),*cada uno de los cuales se asocia con una condición de invocación. Estas condiciones determinan cuando el KA debe ser activado. Los KAs pueden ser activados de una manera dirigida por objetivos o dirigida por datos; los KAs pueden ser también reactivos, permitiendo que el PRS responda rápidamente a cambios en su ambiente. El conjunto de KAs actualmente activos en un sistema representan sus intenciones. Estas varias estructuras de datos son manipuladas por un sistema intérprete, que es responsable de la actualización de las creencias, invocando KAs, y ejecutando acciones.

TouringMachines

La arquitectura consiste de subsistemas de percepción y acción, que realizan la interfaz directamente con el ambiente del agente, y de tres capas de control, contenidas en un framework de control, que media entre las capas. Cada capa es un proceso independiente, productor de actividad, que se ejecuta continuamente [**FER92**].

La capa reactiva genera cursos potenciales de acción en respuesta a eventos que ocurren demasiado rápido como para que otras capas los traten. Está implementada como un conjunto de reglas situación-acción, en el estilo de la arquitectura subsumption.

La capa de planeamiento construye planes y selecciona acciones para ejecutar a fin de satisfacer los objetivos del agente. Esta capa consiste de dos componentes: un planeador, y un mecanismo foco de atención. El planeador integra la generación del plan y la ejecución, y utiliza una librería de planes parcialmente elaborados, junto con un mapa topológico del mundo, a fin de construir planes que lograrán el principal objetivo del agente. El propósito del mecanismo foco de atención es limitar el porcentaje de información con los que el planeador debe tratar, y de esta forma aumentar su eficiencia. Logra esto filtrando información relevante procedente del ambiente.

La capa de modelado contiene representaciones simbólicas del estado cognitivo de otras entidades dentro del ambiente del agente. Estos modelos son manipulados a fin de identificar y resolver conflictos entre objetivos – situaciones donde el agente ya no puede lograr sus objetivos, como resultado de una interferencia inesperada.

Las tres capas son capaces de comunicarse unas con otras (vía paso de mensajes), y están contenidas en un framework de control. El propósito de este framework es mediar entre las capas, y en particular, tratar con acciones conflictivas propuestas por las diferentes capas. El framework de control logra esto utilizando reglas de control.

COSY

La arquitectura COSY [**BUR92**] es un BDI (creencias, deseos e intenciones) híbrido que incluye elementos tanto de PRS como de IRMA. La arquitectura tiene cinco componentes principales: (i) sensores; (ii) actuadores; (iii) comunicaciones; (iv) cognición; e (v) intención. Los primeros tres componentes son directos: los sensores reciben entradas perceptibles no comunicativas, los actuadores permiten al agente realizar acciones no comunicativas, y el componente comunicaciones permite al agente enviar mensajes. De los dos componentes restantes, el componente intención contiene objetivos a largo plazo, actitudes, responsabilidades y los elementos de control que toman parte en el razonamiento y la toma de decisiones del componente cognición, y el componente cognición es responsable de mediar entre las intenciones del agente y sus conocimientos acerca del mundo, y de elegir un acción apropiada para realizar. Dentro del componente cognición se encuentra la base de conocimiento que contiene las creencias del agente, y tres componentes procedurales: un componente de ejecución de script, un componente de ejecución de protocolo, y un componente de razonamiento, decisión y reacción. Un script es una fórmula o plan para lograr un objetivo. Los protocolos son diálogos que representan

frameworks de cooperación. El componente de razonamiento, decisión y reacción es tal vez el componente clave de COSY. Está hecho de un número de otros subsistemas, y es estructurado tal como PRS e IRMA. Se mantiene una agenda, que contiene un número de scripts activos. Estos scripts pueden ser invocados de una manera dirigida por objetivos, o de una dirigida por datos. Un componente filtrador selecciona entre scripts en competición para la ejecución.

Composicional

En una arquitectura composicional, todas las funcionalidades son diseñadas como una serie de componentes estructurados jerárquicamente, que interactúan, basados en tareas [**BRA97a**].

Las tareas se caracterizan en términos de sus entradas, sus salidas y su relación con otras tareas. La interacción y cooperación entre componentes, entre componentes y el mundo externo, y entre componentes y usuarios se especifica en términos de intercambio de información, secuenciamiento de información y dependencias de control. Los componentes en sí pueden ser de cualquier complejidad y pueden realizar cualquier función de dominio.

Modelos de tareas definen la estructura de arquitecturas composicionales: los componentes en una arquitectura composicional están directamente relacionados a tareas en una (des)composición de tareas. En una arquitectura composicional son especificados y modelados explícitamente los siguientes elementos:

- Una (des)composición de tareas: por cada tarea en una jerarquía de tareas un conjunto de subtareas puede ser especificada;
- Intercambio de información: se especifica como enlaces de información entre componentes. Cada enlace de información relaciona la salida de un componente con la entrada de otro;
- Secuenciamiento de tareas: se modela explícitamente dentro de los componentes como conocimiento de control de tarea. El conocimiento de control de tareas incluye no sólo conocimiento de qué tarea debe ser activada, cuando y cómo, sino también conocimiento sobre información de control asociado con la activación de la tarea y el porcentaje de esfuerzo que puede permitirse para lograr un objetivo dado.
- Delegación de subtareas: durante la adquisición del conocimiento una tarea como un todo se modela. Durante el proceso de modelado se toman decisiones como por ejemplo: qué tarea es desarrollada mejor y por cuál agente. Este proceso, que en general también puede ser llevado a cabo en tiempo de ejecución, resulta en la delegación de (sub)tareas a partes envueltas en la ejecución de la tarea; y
- Estructuras de conocimiento: durante la adquisición del conocimiento una estructura apropiada para el dominio del conocimiento debe ser proyectada. El significado de los conceptos utilizados para describir un dominio y las relaciones entre los conceptos y grupos de conceptos, debe ser determinado. Los conceptos se requieren para identificar objetos distinguibles en un dominio, pero además para expresar los métodos y estrategias empleadas para realizar la tarea.

BDI Composicional

En esta arquitectura, el modelo genérico de un agente con arquitectura composicional es refinado en un modelo BDI genérico racional, en el cual el agente es capaz de razonamiento explícito acerca de sus creencias, deseos e intenciones. El modelo BDI Composicional [**BRA97b**] está basado en un análisis de las tareas desarrolladas por un agente BDI. Tal análisis de tareas, resulta, en una composición (jerárquica) de tareas, que es la base para un modelo composicional.

En la arquitectura composicional, el modelo genérico establece las siguientes tareas necesarias para el agente:

- Control de sus propios procesos,
- Cumplimiento de sus tareas propias,
- Manejo de su interacción con el mundo,
- Manejo de su comunicación con otros agentes,

- Mantenimiento de información sobre el mundo, y
- Mantenimiento de información sobre otros agentes.

En la arquitectura BDI Composicional, cada una de las tareas anteriores es refinada, descomponiéndolas en tres componentes:

- Creencias del agente.
- Deseos del agente.
- Intenciones del agente.

Entonces la jerarquía de tareas presentada en el modelo genérico se extiende agregando los tres componentes anteriores. Por ejemplo, la tarea control de sus propios procesos se subdivide en las siguientes subtareas: determinación de creencias, determinación de deseos y determinación de intenciones. El resultado es un agente BDI más específico en el cual las dependencias entre creencias, deseos e intenciones se hacen explícitas.

B.4 PATRONES DE DISEÑO (tomado de [**PAT**])

B.4.1 Algo de Historia

Curiosamente, el concepto de patrón de diseño, no nace en el software, sino en otra actividad, en la arquitectura (la de construcción de edificios).

Christopher Alexander es el arquitecto que primero estudió el concepto de patrón (pattern), en el contexto de construcción de edificios y comunidades.
El escribió, ya en 1977: "Cada patrón describe un problema que ocurre una y otra vez en nuestro entorno, y además, describe el núcleo de la solución a ese problema, de tal manera, que podemos usar esa solución un millón de veces más en el tiempo, sin que tenga que ser la misma cada vez".
A fines de los ochenta, el término "arquitectura de software" (software architecture) era ya usado normalmente, y en las reuniones de especialistas, iba naciendo la idea de un manual para arquitectos de sistemas, una especie de guía o hasta enciclopedia, de las prácticas habituales en la construcción de sistemas.

Ya había existido algún antecedente ilustre, el conocido "The Art of Computer Programming" del notable Donald Knuth. Su intento de catalogar el conocimiento acumulado sobre la programación, estuvo centrado en los algoritmos.

Algunas otras publicaciones habían tratado otros aspectos, el libro seminal de Coplien "Advanced C++: Programming Styles and Idioms", mostró ciertas soluciones específicas a C++, por ejemplo, una que permitía construir una clase String (tan necesaria en un lenguaje que derivaba de C, donde el concepto de string como tipo primitivo no existía), de una forma eficiente.

Por otra parte, en los desarrollos en Smalltalk (más que un lenguaje o una tecnología, un ambiente de objetos), se comienzan a describir patrones, soluciones que ya se habían aplicado en problemas anteriores, notablemente en la propia librería base del lenguaje (como el caso del notify en Object, preludio de los actuales Observer). El mítico Kent Beck tuvo la idea de difundir el trabajo de Alexander, tan alejado al principio del software, entre la comunidad smalltalker, desde su columna en "The Smalltalk Report". Peter Coad también trabaja en esos tiempos en coleccionar patrones. En un escrito de 1992, "Object Oriented Patterns", en las comunicaciones de la asociación ACM (Association for Computer Machinery, **http://www.acm.org/**), comienza a presentar el uso de patrones en las etapas de análisis y diseño.

Pero es con el trabajo de Gamma, Helm, Johnson, Vlissides, "Design Patterns", en 1995, subtitulado "Elementos de software orientado a objetos reusable", cuando el tema se pone maduro. Estos autores, conocidos afectuosamente como "GoF" (la "Gang of Four", la pandilla de los cuatro), son los que toman el trabajo algo monumental, de hacer un catálogo de los patrones de diseño que hasta ese momento habían aparecido, y en una lista de 23 patrones, tratan de enumerar el conocimiento acumulado sobre el tema.

B.4.2 ¿Qué es un Patrón de Diseño?

Christopher Alexander menciona que un patrón, es "la solución a un problema".

Según "GoF" un patrón tendrá, cuatro elementos esenciales:

- **El nombre**: a cada patrón popular, se le asignado una denominación que permite que los entendidos en el tema, puedan conversar usando un diccionario común, permitiendo un mayor grado de abstracción y comunicación. Según "GoF", uno de los problemas que tuvieron, fue encontrar un nombre apropiado para cada patrón que catalogaron.
- **El problema**: explica el problema original, y su contexto. Puede describir desde detalles específicos, como algoritmos, o clases y estructuras que se han encontrado inflexibles a la hora de implementarse.

- **La solución**: la solución que un patrón describe, no necesariamente es detallada al nivel de implementación, sino que provee una descripción abstracta, una enumeración de elementos y sus relaciones, para solucionar el problema planteado.

- **Las consecuencias**: son los resultados de aplicar el patrón, los "trade-off", compromisos, que se tienen que aceptar al adoptar el mismo.

B.4.3 Describiendo un Patrón

He aquí un formato consistente, para describir un patrón, adoptado en el libro de Gamma:

Nombre del patrón y clasificación
Cada patrón tiene un nombre, y una categoría a la que pertenece. El "GoF" los clasifica en creacionales, estructurales y de conducta.

Intención
Cada patrón tiene una intención, una razón, una justificación, "¿qué problema de diseño trata de abordar?".

Motivación
Describe un escenario: un problema en particular que aparece, y que ayuda a entender la descripción algo más abstracta del patrón genérico.

Aplicación
En qué situaciones puede ser aplicado un patrón. Pueden darse ejemplos de malos diseños, que pueden beneficiarse de la aplicación de este patrón en particular.
Estructura
Se adopta un diagrama basado originariamente en la OMT ("Object Modeling Technique"), hoy se usa UML ("Unified Modeling Language", Lenguaje Unificado de Modelado).

Participantes

La lista de las clases y objetos que participan del patrón de diseño, y las responsabilidades que tienen.

Colaboraciones

Descripción de cómo los participantes colaborar para llevar a cabo sus responsabilidades en el patrón

Consecuencias

Explica cómo el patrón cumple con sus objetivos, y que compromisos se asumen.

Implementación

Esta es la parte que más puede variar, porque para cada patrón puede haber varias posibles implementaciones, incluso, diferentes implementaciones según la tecnología adoptada. Puede haber sutiles diferencias entre una implementación y otra, ligadas, por ejemplo, al lenguaje de implementación.

Código de ejemplo

Los buenos de la "GoF", siempre nos dan código de ejemplo de cada patrón, por ejemplo, en Smalltalk o en C++. Han aparecido luego libros con patrones aplicados a Java, y comienzan a aparecer las implementaciones .Net.

Usos conocidos

Un patrón no es tal, si no ha sido ya empleado en algún caso real. Se debe incluir por lo menos dos ejemplos conocidos de diferentes ámbitos.

Patrones relacionados

Puede que haya problemas parecidos, que tengan más de una solución. De ahí que muchos patrones estén relacionados entre sí: como el caso del Proxy y del Adapter. Se trata de explicar acá cuáles son las similitudes y las diferencias (no menos importantes), entre patrones relacionados.

B.4.4 Clasificación de Patrones

Gamma y sus colaboradores, clasifican los patrones de diseño en las siguientes categorías:

B.4.4.1 Patrones Creacionales

Abstraen el proceso de instanciación, ayudando a independizar a un sistema, de cómo sus objetos son creados. En general, tratan de ocultar las clases y métodos concretos de creación, de tal forma que al variar su implementación, no se vea afectado el resto del sistema.

Algunos ejemplos de este patrón en .NET, son:

- **Abstract Factory**: da una interfaz para crear objetos de alguna familia, sin especificar la clase en concreto.
- **Builder**: separa la construcción de un objeto complejo, de su representación. De esa manera, el mismo proceso de construcción puede crear diferentes resultados.
- **Factory Method**: se define una interfaz para crear objetos, como en el Abstract Factory, pero se delega a las subclases implementar la creación en concreto.
- **Prototype**: mediante una instancia prototípica, se consigue otras instancias de ese objeto.
- **Singleton**: consigue dar un solo objeto de la clase, en cualquier momento de la aplicación.

B.4.4.2 Patrones Estructurales

Se ocupan de cómo clases y objetos se agrupan, para formar estructuras más grandes.

Algunos ejemplos de este patrón en .NET, son:

- **Adapter**: permite convertir una interfaz de una clase, en otra, que es la esperada por algún cliente.
- **Bridge**: desacopla una abstracción de su implementación en concreto. Luego, se puede cambiar la implementación, o la abstracción, sin cambiar la otra.
- **Composite**: compone objetos en una estructura de árbol, donde los objetos compuestos se tratan de forma similar a los objetos simples.
- **Decorator**: agrega responsabilidad a un objeto dinámicamente, dando una alternativa a la extensión de una clase, en lugar de usar subclases.
- **Facade**: provee una interfaz unificada a un conjunto de funciones de un subsistema. Es una interfaz de alto nivel, para facilitar el uso del subsistema.
- **Flyweight**: permite compartir objetos, sin repetirlos en el sistema, eficientemente.
- **Proxy**: provee un subrogado a otro objeto, para controlar el acceso al mismo.

B.4.4.3 Patrones de Conducta

Más que describir objetos o clases, describe la comunicación entre ellos. Frecuentemente, describen las colaboraciones entre distintos elementos, para conseguir un objetivo.

Algunos ejemplos de este patrón en .NET, son:

- **Chain of Responsability**: desacopla el remitente de un mensaje, de su receptor, permitiendo que hayan varios objetos que tengan la oportunidad de manejar el requerimiento. Eso se consigue pasando el requerimiento por la cadena de objetos hasta llegar al encargado de atenderlo.
- **Command**: encapsula el requerimiento a un objeto, permitiendo incluso el "undo" de la operación.
- **Interpreter**: construye una representación de la gramática de un lenguaje, junto con su intérprete.
- **Iterator**: da un modo de acceder a los elementos de un objeto colección o similar, sin exponer su estructura interna.
- **Mediator**: permite la interacción de varios objetos, sin generar acoples fuertes en esas relaciones.
- **Memento**: sin necesitar de entrar en la estructura interna de un objeto, permite capturar su estado, para, por ejemplo, poder restaurarlo más adelante.
- **Observer**: define una relación uno a muchos, entre un objetos y otros que están interesados en sus cambios, de nuevo, sin caer en el acople entre los mismos.
- **State**: permite a un objeto cambiar su conducta cuando cambia su estado interno, simulando que cambia de clase.
- **Strategy**: define una familia de algoritmos, y los hace intercambiables.
- **Template Method**: define el esqueleto de una operación, cuyas operaciones más básicas, quedan delegadas en subclases.
- **Visitor**: permite recorrer una estructura (un árbol, por ejemplo), aplicando una operación a cada elemento.

B.4.5 Aplicación de los patrones

B.4.5.1 Patrón Singleton

Contexto

En algunas situaciones, se necesita que algunos datos sean accesibles desde el resto del sistema. Y también se necesita que esos datos sean únicos.

Problema
¿Cómo hacer que la instancia de un objeto sea accesible globalmente, y que sea única?

Fuerzas
Consideremos las fuerzas que intervienen en las posibles soluciones.

- Hay lenguajes que soportan la existencia de variables globales, como Visual Basic 6, o Visual C++. En estos lenguajes, esa variable reside en el espacio de nombre raíz, y pueden ser accedidas desde cualquier otra parte del sistema. Esto puede solucionar el requerimiento de que sea visible globalmente, pero no el de instancia única. Hay otros lenguajes que no tienen el concepto de variable global.
- Si se necesita que la instancia sea única, debemos tener control de la creación de las instancias de la clase. Habrá que implementar algún mecanismo para que no cualquiera pueda crear una instancia.

Solución
El patrón Singleton proporciona la siguiente solución:

- Hacer que la clase provea una instancia de sí misma.
- Permitir que otros objetos obtengan esa instancia, mediante la llamada a un método de la clase.
- Declarar el constructor como privado, para evitar la creación de otros objetos.

El diagrama UML correspondiente es muy simple:

Figura 17. Diagrama UML del patrón Singleton

El diagrama UML (Ver Figura 17) muestra que Singleton es una clase. Contiene una propiedad estática (el subrayado indica que es un método de clase, más que de instancia), que retorna un Singleton, un objeto de la misma clase. Según la notación UML el número 1 en la esquina superior derecha, indica que solamente habrá una instancia de esta clase. El signo "-" en el constructor, lo señala como privado. Esto garantiza que nadie aparte de la propia clase, pueda crear una instancia.

B.4.5.2 Patrón Observador

Contexto
Siempre que pensamos en objetos, podemos encapsular su estado y la implementación de su comportamiento. Esto permite reutilizarlo en otras aplicaciones, de forma natural. Pero un objeto no es un ente aislado. Es común que deba colaborar con otros para cumplir con una tarea. A veces, la colaboración es altamente acoplada: los objetos intervinientes conocen sus interfaces de antemano. En otras situaciones, algunos objetos solamente estarán interesados en los cambios que se ejecuten en otro determinado, sin que éste necesariamente tenga que conocer a los primeros. Este es el caso del patrón Modelo-Vista-Controlador, que permite separar el modelo que se esta manejando, de la forma de mostrarlo (la vista). La vista (un control en un formulario, un gráfico), puede que esté interesada en conocer cuándo cambia el modelo subyacente.

Problema
¿Cómo podemos notificar a otros objetos que el estado de un objeto determinado cambió, sin depender de qué clases son?

Fuerzas que intervienen
Una posible solución, es que el objeto que cambia, llame directamente a los objetos interesados. Pero esto crea una dependencia entre ellos, que debemos evitar, porque el acoplamiento dependiente que se crea, dificulta la reutilización y complica los cambios futuros en las relaciones.

Un cambio en la vista, si adoptamos la llamada directa, podría afectar entonces a la programación del modelo. La necesidad de evitar esa dependencia se torna importante. Por otra parte, si tenemos que agregar una nueva vista, tendremos que modificar el código del modelo, que avisa de sus cambios.

Por otra parte, la llamada directa puede ser el método más rápido de avisar los cambios. Habrá que considerar los efectos sobre el buen desempeño de la aplicación, al adoptar un esquema menos directo.

Solución
El patrón Observador, mantiene una lista de objetos interesados en enterarse de los cambios. Todos los interesados en observar, deben implementar una interfaz Observer (Observador).

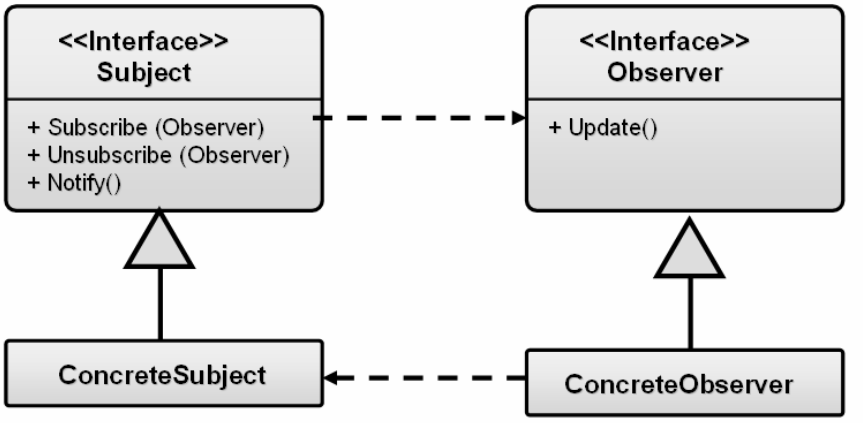


Figura 18. Diagrama UML del patrón Observador

En el diagrama (Ver Figura 18), aparece otra interfaz, Subject, que implementa todos los objetos que quieren ser observados. Al aplicar estas interfaces, nos independizamos de las clases en concreto, que más adelante pueden cambiar, sin afectar el esquema adoptado.

Pero resulta algo notable, que en .NET, existe otra forma de implementar el patrón, que ha nacido de la capacidad de definir eventos, y escuchadores ("handlers") para esos eventos.

B.4.6 Otros Tipos de Patrones
Además de los patrones de diseños, existen patrones de arquitectura y de implementación.

Un patrón de arquitectura, según Buschmann, es:
"Un patrón que expresa un esquema de organización estructural fundamental para un sistema de software. Provee un conjunto predefinido de subsistemas, especifica sus responsabilidades, e incluye reglas y guías para organizar las relaciones entre ellos".

B.4.6.1 Patrones de Arquitectura

PATRÓN DE CAPAS

Contexto

Estamos trabajando en un sistema grande y complejo. Y queremos manejar la complejidad vía la descomposición.

Problema

Cómo estructurar la información para soportar los requerimientos de mantenimiento, reusabilidad, escalabilidad y robustez.

Solución

Componer la solución en una serie de capas. Cada capa debe ocuparse de un nivel del problema, y debe tener poca cohesión con las demás. (Ver Figura 19).

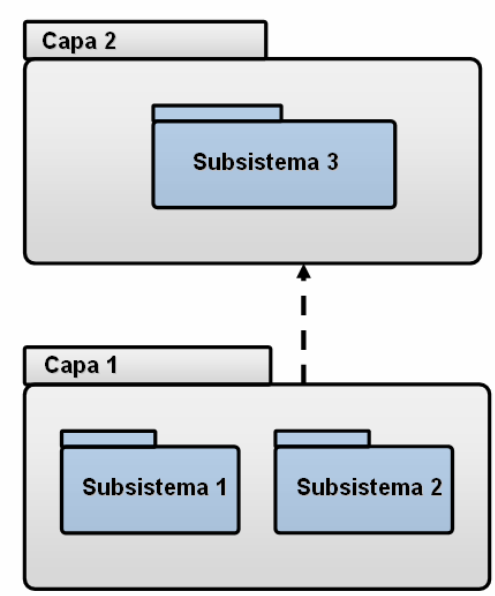


Figura 19. Diagrama UML del patrón de Capas

Este patrón tiene consecuencias no evidentes: el cambio en una capa, debería alterar en poco los cambios en las otras capas. En definitiva, como en el ejemplo del modelo ISO de redes, el cambiar la implementación de una capa, debe tener los mínimos efectos en el resto de la aplicación.

B.4.6.2 Patrones de Implementación

Patrones de implementación (Idioms): Un idiom es un patrón de bajo nivel, específico de un determinado lenguaje de programación. Describen como implementar aspectos particulares de los componentes, o de sus interrelaciones, utilizando las características de un determinado lenguaje.
Como ejemplo de un idiom se tiene la siguiente construcción en C++ para copiar cadenas de caracteres: while (*destino++ = *src++).

BIBLIOGRAFIA

[**ADLOV**] Advanced Distributed Learning (ADL) Sharable Content Object Reference Model (SCORM) 2004, 2nd Edition, Overview.

[**AMB88**] AMBORS, I. J. and STEEL, S., "Integrating planning, execution and monitoring". In Proceedings of the Seventh National Conference of artificial Intelligence, pp. 83-88, St. Paul, MN, 1988.

[**BRA98**] BRATMANN, M. E, ISRAEL, D.J. and POLLACK, M.E., "Plans and resource-bounded practical reasoning". Computational Intelligence, Vol. 4, Nº 1, pp. 349-355, 1998.

[**BRA97a**] BRAZIER, F.M.T, DUNIN-KEPLICZ, B.M., JENNINGS, N.R and TREUR, J., "DESIRE: Modelling multi-agent systems in a compositional formal framework", 1997.

[**BRA97b**] BRAZIER, F.M.T, DUNIN-KEPLICZ,B.M, TREUR,J and VERBRUGGE, R., "Modelling Internal Dynamic Behaviors of BDI Agents". Proceedings of the Third International Workshop on Formal models of Agents, Lectures Notes Springer Verlag, 1997.

[**BRO86**] BROOKS, R. A. "A robust layered control system for a mobile robot". IEEE Journal of Robotics and Automation, Vol. 2, Nº 1, pp. 14-23, 1986.

[**BUR92**] BURMEISTER, B. and SUNDERMEYER, K., "Cooperative problem solving guided by intentions and perception". In Werner, E. and Demazeau, Y., editors, Decentralized AI 3. Proceedings of the Third European Workshop on Modelling Autonomous Agents and Multi-Agent Worlds (MAAMAW-91), pp. 77-92. Elsevier Science Publishers B.V.: Amsterdam, The Netherlands. 1992.

[**CHA86**] CHAPMANN, D. and AGRE, P., "Abstract reasoningas emergent from concrete activity". In Georgeff, M. P and Lansky, A. L. , Proceedings of the 1986 Workshop , 1986.

[**COH89**] COHEN, P. R., GREENBERG, N. L., HART, D. N. and HOWE, A. E. "Tryal by fire Understanding the design requirements for agents in a complex environment", AI Magazine, Vol. 10, Nº 3, pp. 32-48, 1989.

[**EDU04**] EduStance**,** 2004. URL: http://www.edustance.com/

[**FER92**] FERGUSON, I. A., "Towards architecture for adaptive, rational, mobile agents". In Werner, E. and Demazeau, Y., editors, Decentralized AI 3. Proceedings of the Third European Workshop on Modelling Autonomous Agents and Multi-Agent Worlds (MAAMAW-91), pp. 249-262. Elsevier Science Publishers B.V.: Amsterdam, The Neth- erlands, 1992.

[**FIK71**] FIKES, R. E. and NILSSON, N. "STRIPS: A new approach to the application of theorem proving to problem solving". Artificial Intelligence, Vol. 5, Nº 2, pp.189-208, 1971.

[**GEN95**] GENESERETH, M. R. And KETCHPEL, S. P., "Software Agents", 1995.

[**GEO87**] GEORGEFF, M. P. and LANSKY, A. L., "Reactive reasoning and planning". In Proceedings of the Sixth National Conference on Artificial Intelligence (AAAI-87), pp. 677-682, Seattle, WA, 1987.

[**GEO95**] GEORGEFF, M. and RAO, A., "BDI Agents: From Theory to Practice". Proceedings of the First Internacional Conference on Multi-Agent Systems (ICMAS – 95), San Francisco, USA, Junio, 1995.

[**HAY95**] HAYES-ROTH, B., "An Architecture for Adaptive Intelligent System", Artificial Intelligence: Special Issue on Agents and Interactivity, 72, pp. 329-365, 1995.

[**JEN93**] JENNING, N., "Specification and implementation of a belief, desire joint intention architecture fro collaborating problem solving", Artificial Intelligence, Vol: 74, Nº 2, 1993.

[**KAE86**] KAELBLING, L. P., "An architecture for intelligent reactive systems". In Georgeff, M. P. and Lansky, A. L., editors, Reasoning About Actions & Plans. Proceedings of the 1986 Workshop, pp. 395-410. Morgan Kaufmann Publishers: San Mateo, CA., 1986.

[**LIN01**] LINDNER, R. Standarization Bodies at Work. *Prometheus Newsletter*, *6*, 2001.

[**MAES91**] MAES P., "Agents that reduce work and information overload", Communication of the ACM, Vol. 37, Nº. 7, pp. 31-40, 1991.

[**MAES95**] MAES, P., "Artificial Life Meets Entertainment: Life like Autonomous Agents", Communications of the ACM, Vol. 38, Nº. 11, pp. 108-114, 1995.

[**PAT**] Patrones de Diseño Artículo original en **http://www.ajlopez.com/**

[**RUS95**] RUSSELL, S. J. and NORVIG P., "Artificial Intelligence: A Modern Approach", Englewood Cliffs, NJ: Prentice Hall, 1995.

[**SMI94**] SMITH, D. C., CYPHER A. and SPHORER, J., "KidSim: Programming Agents Without a Programming Language", Communication of the ACM, Vol. 37, Nº. 7, pp. 55-67, 1994.

[**VER90**] VERE, S. and BICKMORE, T., "A basic agent", Computational Intelligence, Vol. 6, Nº 1, pp. 41-60, 1990.

[**WOO93**] WOOD, S. "Planning and Decision Making in Dynamic Domains". Ellis Horwood: Chichester, England. 1993.

[**WOO95**] WOOLDRIDGE, M. and JENNINGS, N. R. Intelligent agents: Theory and practice. The Knowledge Engineering Review, 10(2):115–152, 1995.

[**W3C00**] Extensible Markup Language (XML) 1.0, W3C Recommendation. World Wide Web Consortium. 2ª edición, 2000. URL http://www.w3c.org/TR/2000/REC-xml-20001006.