UNIVERSIDAD POLITÉCNICA DE VALENCIA

Departamento de Sistemas Informáticos y Computación

GESTIÓN DE USUARIOS Y CONTROL DE ACCESO BASADO EN ROLES. UN CASO REAL

Marta Ruiz Server

TESIS DEL MASTER

Máster en Ingeniería del Software, Métodos Formales y Sistemas de Información

DIMENSIÓN INFORMÁTICA

Septiembre, 2007

Dirigida por:

Vicente Pelechano Ferragud

Índice general

pagina

Índice de tablas

Índice de figuras

Figura pagina

LISTA DE ABREVIATURAS

CRBAC	Core RBAC
DSDR	Dynamic Separation of Duty Relations
DI-SFM	Metodología creada por Dimensión Informática
HRBAC	Hierarchical RBAC
LDAP	Lightweight Directory Access Protocol
NIST	National Institute of Standards and Technology
RBAC	Role Based Access Control
RUP	Rational Unified Process
SSDR	Static Separation of Duty Relations
SSL	Secure Sockets Layer
UML	Unified Modeling Language

Capítulo 1
INTRODUCCIÓN

En este capítulo se presenta el entorno en el que se ha desarrollado el trabajo del *Máster en Ingeniería del Software, Métodos Formales y Sistemas de Información*. Este trabajo se ha desarrollado con una orientación profesional en el seno de la empresa Dimensión Informática y para un proyecto concreto de la Conselleria de Educación. El proyecto desarrolla un sistema de información para la gestión de centros educativos, y el trabajo se ha centrado en el control de acceso para dicho sistema.

Este capítulo se estructura de la siguiente forma: en primer lugar se presenta una introducción al trabajo realizado. En el segundo apartado se presenta el entorno en el que se ha desarrollando el trabajo, la empresa donde se ha realizado y una breve introducción al proyecto donde se enmarca. A continuación, se describe el problema, los objetivos y el diseño propuesto para finalizar con distintos métodos de control de acceso.

1.1. Introducción

El trabajo desarrollado en el máster tiene una orientación profesional. En él se expone un trabajo de tipo práctico relacionado con una de las actividades profesionales de un titulado en informática. El trabajo ha sido desarrollado en la empresa Dimensión Informática en el contexto de un proyecto real para la Conselleria de Educación. En Dimensión Informática realizo las actividades propias de un analista, como son la toma de requisitos y el modelado conceptual del sistema. En diciembre del 2006

finalicé una beca FPI, durante la cuál realicé los estudios del máster y del doctorado. Una vez encaminada mi tesis, entré a formar parte de Dimensión Informática en febrero del 2007. Desde entonces, estoy realizando mis tareas de analista en este proyecto.

Para la realización de este trabajo han sido de gran ayuda los conocimientos y capacidades desarrollados en el paso por la Universidad Politécnica de Valencia, no sólo los conocimientos adquiridos durante los estudios, sino también las investigaciones realizadas en el seno del grupo OO-Method.

De los cuatro años de investigación en el grupo OO-Method, se está desarrollando un trabajo de tesis doctoral donde se aborda la generación automática de servicios Web a partir de modelos conceptuales OO-Method/OOWS [1–3]. Esta propuesta está compuesta por una serie de grupos funcionales entre los que se encuentra la *Gestión de Usuarios*. Esta propuesta es un primer esbozo del trabajo presentado en este máster. El proceso presentado en esos trabajos ha sido extrapolado y extendido fácilmente para abordar la problemática encontrada durante este proyecto.

A raíz de las investigaciones realizadas en estos trabajos y la necesidad que ha surgido de llevar a cabo el modelado del control de acceso, ha surgido este trabajo. El hecho de que el cliente (en esta caso la Conselleria de Educación) utilice perfiles en su trabajo día a día, ha ayudado a la elección de continuar por la línea que se venía siguiendo en los trabajos [1–3]. Esta línea de trabajo propone un conjunto de operaciones para el control de acceso basadas en RBAC [4–8], un estándar para el control de acceso basado en roles.

Es importante que el control de acceso (también conocido como gestión de identidad o gestión de usuarios) se aborde desde el primer momento dentro del ciclo de vida, es decir, en las etapas de análisis y diseño. En los últimos años se ha visto como elementos software, aplicaciones, etc. no fueron concebidos con la gestión de usuarios como fase crítica del diseño, y se han tenido que acoplar posteriormente

elementos que la cubrieran existiendo así vulnerabilidades en las aplicaciones. Por lo tanto, uno de los objetivos de esta trabajo es minimizar la aparición de vulnerabilidades antes de que el producto vea la luz y añadir características que aumenten su calidad.

Por eso es importante que las operaciones necesarias para el control de acceso de un sistema de información se deben identificar en el modelado conceptual (espacio del problema), durante las etapa de requisitos, y así automáticamente definir el diseño de las operaciones de seguridad para que pueda llevarse fácilmente al espacio de la solución.

Durante el trabajo se van a utilizar los términos perfil y rol dependiendo de:

- Perfil: es la terminología que utiliza el cliente y por tanto cuando se esté realizando algún documento que vaya a ser entregado al cliente se utilizará este término.
- Rol: es la terminología utilizada en la jerga informática y por lo tanto será utilizada durante la explicación del trabajo siempre y cuando no sea documentación a entregar al cliente.

1.2. Entorno del trabajo

Este trabajo del máster se ha desarrollado en el seno del proyecto ATICA, en la empresa Dimensión Informática. El proyecto se está realizando dentro de la Unidad Temporal de Empresas (UTE) entre Telefónica Soluciones de Informática y Comunicaciones de España S.A.U. y Dimensión Informática S.L. para la realización de un *sistema de información para la gestión de centros educativos* destinado a la Conselleria de Educación de la Generalitat Valenciana.

Además, el trabajo presentado forma parte de los trabajos realizados en el contexto del grupo de investigación OO-Method: *Métodos de Producción del Software (Object-Oriented Methods for Software Development)* perteneciente al Departamento de Sistemas Informáticos y Computación de la Universidad Politécnica de Valencia,

en el que la autora ha participado activamente durante los últimos cuatro años. Los trabajos realizados en el grupo han permitido adquirir el conocimiento y las habilidades necesarias para transferir a la empresa parte de ese trabajo en un entorno real de desarrollo.

1.2.1. Dimensión Informática

Dimensión Informática es una empresa valenciana especializada en la fabricación de software. En julio de 2005 Dimensión Informática se integró en el Grupo AZERTIA, compañía multinacional de las TIC integrada en la Corporación IBV (IBERDROLA, BBVA). Actualmente, Dimensión Informática ha entrado a formar parte de la empresa Indra.

En la actualidad, Dimensión Informática cuenta con un equipo humano de más de 320 profesionales repartidos en sus oficinas de Baleares, Barcelona, Castellón, Madrid, Murcia, Vigo, Zaragoza y Valencia, siendo esta última su oficina principal donde dispone de unas instalaciones de más de 2.300 metros cuadrados.

Desde sus inicios en 1992, la trayectoria de Dimensión Informática ha estado orientada al desarrollo de Soluciones Tecnológicas multisectoriales. Sus servicios incluyen la planificación, diseño y construcción de sistemas de información, soluciones de integración, portales, soluciones propias (Suite Kewan) y a medida para cualquier compañía y sector.

En su larga experiencia ha realizado proyectos para más de 300 clientes nacionales e internacionales, públicos y privados. Un ejemplo es el diseño y desarrollo de la gestión de ticketing de uno de los buques insignia de la Comunidad Valenciana, la *Ciudad de las Artes y las Ciencias*; también la *Secretaría de Estado de Telecomunicaciones* y la *Sociedad de la Información* han confiado en Dimensión para la dirección de un proyecto que consigue la ventanilla única electrónica de la Administración Pública, unificando la información de todo los ciudadanos dispersa

en múltiples sistemas de información y ubicaciones geográficas; *AENA* gestiona sus incidencias en tiempo real con el desarrollo e implantación del sistema NOTIFES en los aeropuertos de Barajas, Palma de Mallorca y Tenerife. A nivel internacional, el *Instituto de Aeronáutica Civil de Cuba*, el organismo público del Gobierno cubano encargado de la gestión del espacio aéreo y los nueve aeropuertos del país, han elegido el ERP de producción propia Kewan.

1.2.2. ATICA: sistema de información para la gestión de centros educativos

La Conselleria de Educación quiere seguir avanzando en la evolución hacia la Administración Electrónica a través de una estrategia que permita poner a disposición de los colectivos implicados en el entorno educativo todos aquellos servicios que prestados de forma online resultan de gran interés y comodidad.

El principal objetivo de la iniciativa consiste en realizar la toma de requisitos de usuarios, el análisis y definición funcional de los procedimientos relacionados con la Gestión de Centros Educativos, con vistas a realizar el desarrollo e implantación de la nueva aplicación de Gestión de Centros Educativos.

El proyecto global consiste en el diseño y desarrollo de nuevas aplicaciones que permitan a la Conselleria de Educación de la Generalitat Valenciana disponer de la información de los centros de forma centralizada y en tiempo real.

Acceso al sistema

La solución propuesta es una aplicación centralizada, por lo que se podrá acceder a ella desde cualquier centro de estudios que tenga la conectividad adecuada. El acceso final a cada una de las funcionalidades será controlado por una serie de permisos asociados a perfiles de usuarios definidos de forma global, lo que permitirá garantizar la seguridad necesaria para los distintos niveles de confidencialidad de la información.

Gestión de usuarios del sistema

La gestión de usuarios del sistema se podrá gestionar de forma centralizada, creando los usuarios necesarios para la utilización del sistema y asociándoles los perfiles adecuados en función de las actividades que vaya a desarrollar cada usuario, previniendo de esta forma los accesos no deseados a información confidencial.

Gestor de Seguridad

El Gestor de Seguridad implementa las políticas de seguridad de acceso a los datos definidos para el sistema. La política de seguridad se establece en función de la definición de perfiles o roles (capacidad de actuación sobre los datos), áreas de la organización (determinando a qué tipo de datos se tiene acceso) y niveles dentro de la organización (que definen el ámbito de acceso a los datos).

Se va a utilizar una plataforma analítica que ofrece la ventaja de proporcionar un único punto de acceso para el usuario donde el gestor de seguridad enlazará con el sistema de autenticación de usuario. Una vez validado el usuario y obtenidos sus permisos, el gestor de seguridad, con la ayuda de los mecanismos de Single-Sign-On, limitará el ámbito de sus consultas de datos de forma adecuada.

La auditoria de los accesos del usuario al sistema, qué consultas ha efectuado y qué datos ha recibido, se efectuarán apoyándose a si mismo en las capacidades de auditoria de la base de datos complementándose con un desarrollo específico cuando sea necesario.

1.2.3. Características generales de un sistema de identidad y de usuarios

Con carácter general, un sistema de gestión de identidad y usuarios capacita a una organización para, entre otras cosas:

- Permitir la delegación de las tareas de administración y mantenimiento en grupos de usuarios designados a tal efecto por el administrador del sistema, de modo que puedan agilizarse dichas tareas pero siempre de forma segura.
- Posibilitar la aprobación y cancelación instantáneas de los derechos de acceso de los usuarios a todos los recursos, acrecentando la productividad de los usuarios por un lado, y evitando la fuga de información por el otro.
- Hacer posible la gestión automatizada de usuarios durante todo el ciclo de actividad del usuario, aumentando la eficiencia de los procesos y proporcionando un importante ahorro de costes.
- Generar automáticamente informes detallados para demostrar el cumplimiento de las normativas y las directivas de seguridad de aplicación en la organización.
- Obtener pruebas de auditoria.
- Controlar y gestionar de manera unificada y sencilla todos los derechos de acceso de los usuarios a cualquiera de los sistemas con los que trabajan.

Para ello, este tipo de sistemas actúa sobre cinco dimensiones bien diferenciadas:

- **Autenticación**, o la verificación segura de la identidad de usuarios.
- **Autorización**, o la gestión unificada del control de acceso.
- **Administración**, o gestión segura de activos y usuarios, centralizada y delegada.
- **Auditoria**, o el aseguramiento del cumplimiento de normativas.
- **Identidad Federada**, o la integración de modelos extranet (B2B, 2C, B2C) y modelos de negocio basados en servicios Web.

Funcionalidades específicas

En una solución de gestión de identidad y de usuarios podemos establecer tres grandes áreas en las que estructurar la funcionalidad específica que proporciona (ver figura 1–1). Cada una de estas áreas funcionales se complementa con las demás para conformar el sistema completo:

Figura 1–1: Áreas de la Gestión de Identidad

A continuación se caracteriza cada una de estas áreas a través de su funcionalidad específica:

- Mejora de acceso y servicio
  - El sistema instaura la gestión unificada y segura de la identidad de usuarios a través de perfiles, roles y niveles de acceso.
  - Los servicios federados permiten completar procesos de negocio o administrativos garantizando privacidad en los datos.
  - Implementa mecanismos de tipo *single sign-on* para acceso a distintos servicios sin necesidad de introducir las credenciales de usuario más de una vez. Esto posibilita que una persona represente un único usuario de aplicaciones en toda la organización, con un único *token* de autenticación (contraseña, firma, etc.).
  - Ofrece características de autoservicio que mejoran la calidad de servicio (recuperación de contraseñas, solicitudes de cambio periódico, etc).
- Incremento de la seguridad
  - Representa un punto unificado para el control y la gestión de los accesos a los recursos de la organización.

- Bloquea accesos no autorizados de usuarios internos o externos: proveedores, empleados, clientes, etc.
- Asegura el cumplimiento de la LOPD y otras regulaciones, mediante el registro y la firma digital de todas las transacciones.
- Genera informes de acceso y auditoria.

- Reducción de costes
  - Conlleva un aumento de los servicios online y de automatización de procesos, lo que comporta una reducción de intervenciones manuales de provisión o auditoria.
  - Automatiza procesos repetitivos, tediosos y proclives a la comisión de errores, aumentando la eficiencia de la organización.
  - Reduce costes de gestión al centralizar la administración de los usuarios y sus derechos de acceso en un solo lugar.

Descripción técnica

Este tipo de soluciones, por su naturaleza intrínseca actúan en general como un recubrimiento de los diferentes sistemas de la organización permitiendo su uso sin necesidad de que éstos hayan de sufrir modificación alguna.

se construyen mediante una arquitectura no invasiva, con soporte nativo de los protocolos y especificaciones estándar más extendidas (HTTP, LDAP, SQL, BerkeleyDB, CSV, etc.) y basada en tecnología de agentes y conectores que posibilita la integración con aquellos sistemas con los que no pueda éntenderse'de forma nativa.

Todo ello permite la integración de la gestión de provisión de usuarios y asignación de permisos entre sistemas y plataformas heterogéneas. En el caso de la Conselleria de Educación, estos sistemas y plataformas heterogéneas serían ATICA, Títulos, Ajuda, Exenval, Oracle, PostgreSQL, correo electrónico Postfix, directorios LDAP, sistemas operativos Windows/Linux/Unix, PKI de la *Generalitat Valenciana*, etc.

Esto permitirá a la Consellería concentrarse en la definición y confirmación de las políticas centrales de seguridad y gestión del accesos para todos los usuarios y aplicaciones, garantizando su cumplimiento y haciendo posible una uniformidad de criterios para todos los sistemas y plataformas.

1.3. Descripción del problema

El acceso de los usuarios a la aplicación debe contener determinada seguridad. Se debe proponer una solución generalista que permita un entorno de seguridad común para diferentes aplicaciones.

En cuanto a seguridad podemos diferenciar tres aspectos:

- **Gestión de la autenticación**: Dado un nombre de usuario y una contraseña, el sistema debe comprobar que el usuario es válido y la contraseña es correcta. A su vez, el sistema debe proporcionar a partir del usuario, un rol de conexión al sistema, el cuál servirá para configurar los posibles accesos a los diferentes servicios. Además, se debe de proporcionar el centro donde dicho rol es válido.
- **Gestión de acceso**: Dado un rol de conexión, se debe proporcionar un conjunto de accesos a los que tiene permiso este rol. Esta lista de accesos se corresponderá con elementos de la interfaz de usuario (entradas de menú y servicios).
- **Gestión de la activación/desactivación de elementos de la interfaz de usuario**: Dada una lista de accesos, debemos acceder a la vista para activar o desactivar aquellos componentes que sean necesarios de forma automática y sin tener que escribir esta lógica en las vistas.

1.4. Objetivos del trabajo

El objetivo general de este trabajo de tesis del máster es la incorporación del control de acceso basado en roles en el ciclo de vida proporcionado por DI-SFM (ver apartado 2.1).

Los objetivos específicos que se buscan alcanzar son los siguientes:

1. Estudiar el estado actual de los métodos de control de acceso.
2. Estudiar el estado actual de la metodología de DI-SFM.
3. Estudiar la necesidad que tiene la Conselleria de Educación en el sistema de información para la gestión de centros educativos, y de esta forma poder explotar la categorización que ya viene realizando la Conselleria de Educación sobre sus usuarios.
4. Definir un Modelo de Control de Acceso basado en Roles que permita describir la funcionalidad propia de este tipo de control de acceso. El Modelo deberá permitir además la división de funcionalidad a partir de la funcionalidad implementada en ATICA (parte implementada por DI) y la que deberá ser soportada por el LDAP (parte implementada por Telefónica).
5. Definir una estrategia de captura de requisitos para el control de acceso que permita obtener las operaciones necesarias.

1.5. Diseño propuesto

A continuación se muestra punto por punto cómo se resuelven los problemas comentados en el apartado 1.3.

- **Gestión de la autenticación**: para solucionar este problema la propuesta es comprobar si existe una sesión activa (o token de seguridad). Si no es así pedirá la validación del usuario a través del pantalla de acceso o de inicio de sesión. Recogerá el usuario y la contraseña y se conectará al servicio de validación del LDAP correspondiente. Éste devolverá el nombre del usuario, el/los roles de acceso y el/los centros a los que puede acceder dentro de un objeto llamado `sesion`.

- **Gestión de acceso**: para solucionar este problema se recoge la `sesion` y se busca en el repositorio de seguridad qué accesos válidos existen para cada rol del usuario. Existen diferentes aproximaciones a la hora de recuperar esta lista de accesos válidos:
  - Devolver todos los componentes a los que tiene acceso el rol. Se debe guardar la lista completa en caché para posteriores uso.
  - Devolver los componentes a lo que tiene acceso el rol dado el módulo al que va acceder. De esta forma solo se recupera lo que se necesita en cada momento. Esta opción suele usarse en entornos Web puesto que el consumo de memoria en el servidor es menor.

  Otro factor a tener en cuenta para solucionar este problema es qué componentes se van a tener en cuenta en esta lista de accesos.
  - Componentes a los que se tiene acceso.
  - Componentes a los que no se tiene acceso.
  - Todos los componentes susceptibles de tener seguridad con la información de si se debe tener acceso o no.

  Las dos primeras soluciones obligan a guardar en el repositorio todos los posibles componentes a los que van acceder los usuarios, incluso aquellos que van a ser accesibles por todos los usuarios. Por lo tanto la opción que se va a usar en el proyecto es la de guardar sólo aquellos componentes susceptibles de tener seguridad. La lista que devuelva este servicio será una serie de componentes a los cuales se deniega el acceso al rol indicado.
- **Gestión de activación/desactivación de elementos de la interfaz de usuario**: este problema se soluciona junto con el anterior y no debe afectar a la programación de la lógica de las vistas o de los servicios.

1.6. Métodos de control de acceso

A continuación se presentan varias propuestas de trabajos relacionados con el control de acceso para aplicaciones:

- Se pueden definir niveles de seguridad basándose en las habilidades de manipulación que tienen los usuarios, permitiendo a los diseñadores establecer quien puede hacer que, usando propiedades como acceso de grano fino o protección de enlaces (ver [9]).
- Se puede definir un control de acceso basado en contenidos como se realiza en algunas bibliotecas digitales (ver [10]), donde los permisos están codificados como tuplas (credencial, concepto, privilegio). Un concepto no representa un objeto sino un tópico que describe el contenido del objeto.
- Se puede definir un control de acceso basado en equipos (ver [11]), donde las páginas representan procesos y el contenido se tiene en cuenta para reflejar procesos y estados de objetos.
- Se puede definir un control de acceso basado en roles (ver [12]), donde los roles son como una función dentro del contexto de una organización. Los permisos se asignan a los roles, y los usuarios también son asignados a los roles. Los usuarios adquieren los permisos asociados a los roles que tienen asignados.

Tras el análisis de los métodos de control de acceso existentes y tras analizar el tipo de sistema a desarrollar (donde el cliente trabaja con perfiles ya definidos), este trabajo se basa en un control de acceso basado en roles (RBAC) [12]. RBAC es un estándar muy común y extendido como puede verse en la cantidad de trabajos desarrollados a su alrededor (ver [4–8, 13–15]).

Capítulo 2
METODOLOGÍAS Y ESTÁNDARES

En este capítulo se presentan las metodologías y estándares existentes y que han servido de base al trabajo realizado. Primero se presenta DI-SFM, una metodología creada por *Dimensión Informática* que guía el proceso a seguir para los sistemas que se desarrollan. A continuación se comenta brevemente Métrica 3, con la que es totalmente compatible DI-SFM, que ofrece a las Organizaciones un instrumento útil para la sistematización de las actividades que dan soporte al ciclo de vida del software. En el tercer apartado se presenta RBAC, el método de control de acceso en el que se basa la propuesta presentada en el trabajo. Para finalizar se presenta LDAP, un servicio de directorio utilizado por Telefónica Soluciones como repositorio de usuarios y roles.

2.1. Metodología DI-SFM

DI-SFM (ver [16] y [17]) es una metodología de desarrollo que guía el proceso a seguir, describiendo las actividades a realizar, los productos a obtener, los roles que intervienen en dicho proceso y las técnicas a utilizar, es decir, el uso de la metodología en un proyecto permite tener definido de forma precisa qué se debe hacer, cómo debe hacerse, quién debe hacerlo y en qué momento.

La metodología DI-SFM integra los aspectos más destacables de la metodología definida por Rational, RUP (Rational Unified Process) [18] y es compatible con los entregables establecidos con Métrica 3.0 [19] (Ministerio Administración Pública Española).

DI-SFM define:

- un proceso dirigido por modelos,
- centrado en la arquitectura,
- con un desarrollo incremental por iteraciones en que se realiza un control continuo de la calidad.

Las disciplinas más relevantes del proceso son: Requisitos, Análisis y Diseño, Implementación y Pruebas, por otro lado se encuentran la Gestión de la Configuración y del Cambio que son disciplinas horizontales a todo el proyecto.

2.1.1. Proceso

La metodología DI-SFM integra los aspectos más destacables del proceso unificado de Rational y la metodología Métrica 3.0, y los extiende incorporando mejores prácticas para su optimización. DI-SFM define un proceso Dirigido por Modelos (MDA), basado en módulos (desarrollo basado en módulos) y centrado en la Arquitectura.

La metodología define de forma precisa las características del proceso estableciendo quién hace qué, cómo y cuándo y proporcionando disciplinas en las que se integran los cuatro elementos principales del proceso: roles, actividades, productos y flujos de trabajo:

- Los **roles**, que responden a la pregunta ¿Quién?,
- Las **actividades** que responden a la pregunta ¿Cómo?,
- Los **productos**, que responden a la pregunta ¿Qué?
- Y los **flujos de trabajo** de cada disciplina que responden a la pregunta ¿Cuándo?

En los proyectos DI-SFM se promueve la comunicación continua con el cliente a lo largo del ciclo de la vida del proyecto involucrándose como un miembro más del equipo, priorizando los casos de uso y validando cada uno de los entregables que

se van obteniendo en las sucesivas iteraciones. Para llevar a cabo dichas validaciones y revisiones se mantendrán reuniones periódicas de seguimiento en las que se mostrará la situación actual del proyecto y se revisarán aspectos de gestión y planificación. Esta retroalimentación continua favorece la reducción de plazos y costes del proyecto e incrementa la calidad del software.

2.1.2. Disciplinas

La ingeniería del software, tal y como es concebida hoy en día, abarca multitud de actividades de diversa índole. DI-SFM categoriza estas actividades en lo que es llamado por la comunidad científica, disciplina. Una actividad se representa mediante un pentágono irregular mientras que la disciplina se muestra mediante un rectángulo que contiene actividades.

Una disciplina es un conjunto de actividades relacionadas con un área de interés mayor. Las disciplinas que contempla DI-SFM actualmente de la ingeniería del software son (ver 2–1): Requisitos, Análisis y Diseño, Implementación, Pruebas, Gestión de la configuración y del cambio y Gestión de proyectos. A continuación se describen las principales disciplinas para este trabajo (Requisitos, Análisis y Diseño e Implementación).

En los siguientes apartados se comentan brevemente las disciplinas involucradas en este trabajo.

Requisitos

DI-SFM define un proceso dirigido por Casos de Uso y ofrece guías en las que se detalla cómo se debe capturar, organizar y documentar los requisitos de forma óptima, y se establece cómo gestionar los cambios que ocurran en dichos requisitos durante la vida del proyecto. En DI-SFM los casos de uso no son sólo una técnica

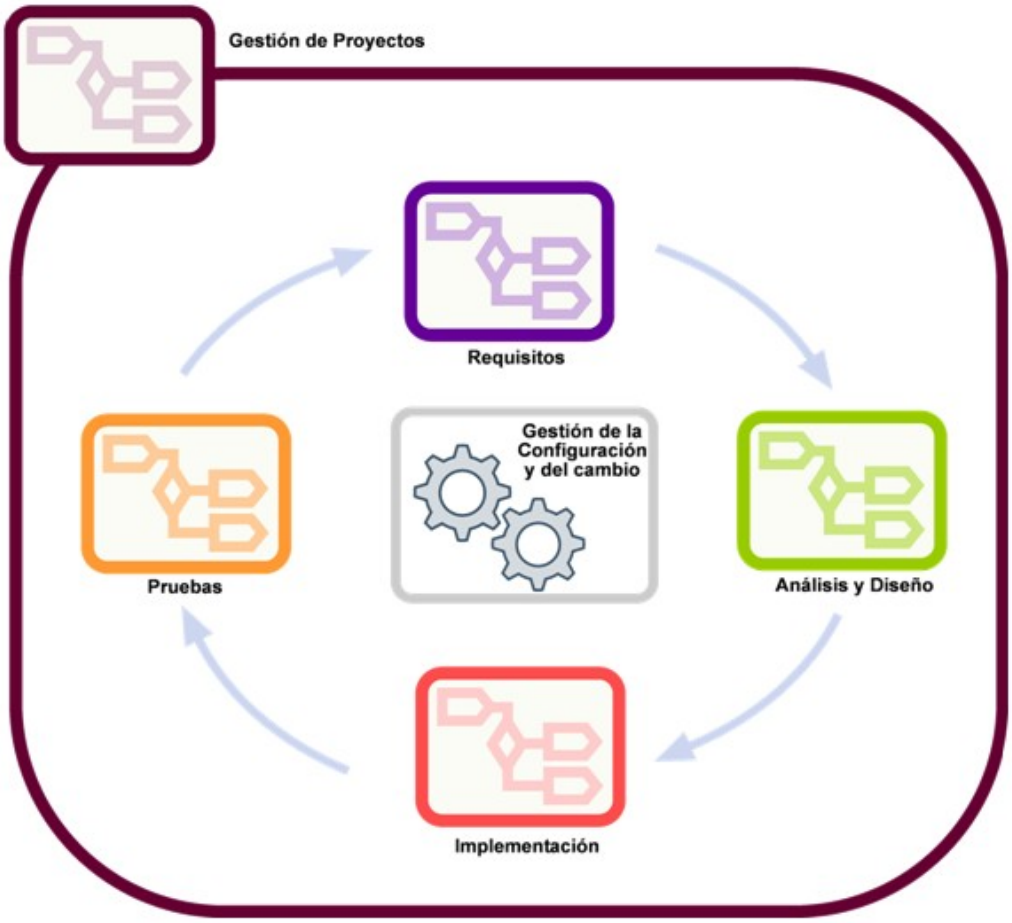

Figura 2–1: Disciplinas de la metodología DI-SFM

para especificar los requisitos del sistema, sino que también guían el diseño, implementación y prueba constituyendo un elemento integrador y una guía de trabajo permitiendo establecer trazabilidad entre los productos o artefactos que se generan en las diferentes actividades del proceso de desarrollo. En el proceso definido por DI-SFM se dedica un mayor esfuerzo a las actividades de Requisitos en las fases de Inicio y Elaboración.

Análisis y Diseño

DI-SFM sigue la aproximación de desarrollo dirigido por modelos MDA (Model Driven Architecture) que es un marco de trabajo definido por OMG (Object Management Group), en el que la clave es la importancia de los modelos en el proceso

de desarrollo de software. MDA propone la definición y uso de modelos a diferente nivel de abstracción, así como la posibilidad de la generación automática de código a partir de los modelos definidos y de las reglas de transformación entre dichos modelos. Con esta nueva visión, se puede obtener código a partir de modelos centrados en el dominio del problema e independientes de la plataforma de implementación, mediante transformaciones de modelos.

Como técnica de modelado DI-SFM utiliza UML (Unified Modeling Language) que es un lenguaje estándar para especificar, visualizar, construir y documentar los artefactos de un sistema software, lo que simplifica la complejidad que supone el diseño y facilita la construcción.

El caso de uso es el punto de partida para las actividades de análisis y diseño ya que a partir de su realización y especificación se obtiene el modelo de análisis. DI-SFM utiliza los diagramas de secuencia de UML para la realización del caso de uso (véase Glosario). Los diagramas de secuencia permiten la identificación de las clases y servicios utilizados en el caso de uso. Los diagramas de secuencia son utilizados para definir el modelo de análisis que contiene las clases de análisis y artefactos asociados, en el que las clases determinan la estructura y las operaciones necesarias para implementar las funcionalidades descritas en los Casos de Uso. Por otra parte los Diagramas de Estado detallarán el comportamiento para las clases que lo requieran. El modelo de análisis evolucionará posteriormente en el modelo de diseño que estará compuesto por diagramas de secuencia de diseño, diagramas de clases y algunos diagramas de estados; en el modelo de diseño se tienen en cuenta aspectos de implementación.

DI-SFM apuesta por la estrategia *divide y vencerás* realizando un desarrollo basado en módulos. De esta manera los sistemas de información son descompuestos en subsistemas y en módulos con interfaces bien definidas, que posteriormente serán ensamblados de manera sistemática para generar el sistema. Esta característica del

proceso de desarrollo permite que el sistema evolucione a medida que se obtienen o se desarrollan sus módulos.

Implementación

La implementación del código fuente resulta más cómoda ya que el uso de un framework y arquitecturas probadas y evaluadas facilitan la programación al desarrollador. Por otra parte, el desarrollador no empieza de cero ya que las transformaciones MDA aplicadas durante el Análisis y Diseño reducen y simplifican los elementos de código fuente a desarrollar.

En DI-SFM se recalca la necesidad de definir estándares de nomenclatura, los cuales facilitarán la identificación de los elementos y la mejor comprensión del código y el modelo de datos por parte de todo el equipo, además de eliminar posibles ambigüedades.

2.2. Métrica 3

La metodología MÉTRICA Versión 3 (para mayor detalle ver [19]) ofrece a las Organizaciones un instrumento útil para la sistematización de las actividades que dan soporte al ciclo de vida del software dentro del marco que permite alcanzar los siguientes objetivos:

- Proporcionar o definir Sistemas de Información que ayuden a conseguir los fines de la Organización mediante la definición de un marco estratégico para el desarrollo de los mismos.
- Dotar a la Organización de productos software que satisfagan las necesidades de los usuarios dando una mayor importancia al análisis de requisitos.
- Mejorar la productividad de los departamentos de Sistemas y Tecnologías de la Información y las Comunicaciones, permitiendo una mayor capacidad de adaptación a los cambios y teniendo en cuenta la reutilización en la medida de lo posible.

- Facilitar la comunicación y entendimiento entre los distintos participantes en la producción de software a lo largo del ciclo de vida del proyecto, teniendo en cuenta su papel y responsabilidad, así como las necesidades de todos y cada uno de ellos.
- Facilitar la operación, mantenimiento y uso de los productos software obtenidos.

La nueva versión de MÉTRICA contempla el desarrollo de Sistemas de Información para las distintas tecnologías que actualmente están conviviendo y los aspectos de gestión que aseguran que un Proyecto cumple sus objetivos en términos de calidad, coste y plazos. Su punto de partida es la versión anterior de MÉTRICA de la cual se han conservado la adaptabilidad, flexibilidad y sencillez, así como la estructura de actividades y tareas, si bien las fases y módulos de MÉTRICA versión 2.1 han dado paso a la división en Procesos, más adecuada a la entrada-transformación-salida que se produce en cada una de las divisiones del ciclo de vida de un proyecto. Para cada tarea se detallan los participantes que intervienen, los productos de entrada y de salida así como las técnicas y prácticas a emplear para su obtención.

En la elaboración de MÉTRICA Versión 3 se han tenido en cuenta los métodos de desarrollo más extendidos, así como los últimos estándares de ingeniería del software y calidad, además de referencias específicas en cuanto a seguridad y gestión de proyectos. En una única estructura la metodología MÉTRICA Versión 3 cubre distintos tipos de desarrollo: estructurado y orientado a objetos, facilitando a través de interfaces la realización de los procesos de apoyo u organizativos: Gestión de Proyectos, Gestión de Configuración, Aseguramiento de Calidad y Seguridad. La automatización de las actividades propuestas en la estructura de MÉTRICA Versión 3 es posible ya que sus técnicas están soportadas por una amplia variedad de herramientas de ayuda al desarrollo.

2.3. Role Based Access Control

El proceso de desarrollo de este estándar fue iniciado por el National Institute of Standards and Technology (NIST) debido a la necesidad existente entre los compradores de sistemas de información de una definición clara y precisa cuyas características de control de acceso no fueran sufriendo cambios. Durante muchos años los vendedores han estado implementando características de control de acceso basado en roles en sus sistemas sin un acuerdo general de cuáles eran las características que bebe tener todo sistema de control de acceso basado en roles (RBAC) [12]. Esta falta de consenso de un modelo causó incertidumbre y confusión sobre la utilidad y el significado de RBAC.

El estándar Role Based Access Control (RBAC) [6], es un estándar para el control de acceso basado en roles que aparecieron para resolver esta situación. RBAC está formado por dos partes principales: el *modelo de referencia* de RBAC y su *especificación funcional*.

El Modelo de Referencia de RBAC define conjuntos de elementos RBAC básicos (por ejemplo, usuarios, roles, permisos, operaciones y objetos) y relaciones como tipos y funciones que están incluidos en este estándar. El modelo de referencia de RBAC tiene dos propósitos:

- Definir las características de RBAC incluidas en el estándar. Identifica el conjunto mínimo de características que debe incluir todo sistema RBAC.
- Proveer de un lenguaje preciso y consistente, en términos de conjuntos de elementos y funciones para definir la especificación funcional.

La especificación funcional de RBAC especifica las características que se requieren en un sistema RBAC. Estas características se dividen en tres categorías: operaciones administrativas, consulta administrativa y funcionalidad a nivel de sistema.

- Las operaciones administrativas definen funciones que proveen con operaciones para crear, borrar y mantener los elementos RBAC y sus relaciones (por ejemplo, crear y borrar asignaciones de usuarios a roles).
- La consulta administrativa define funciones que proveen la capacidad de llevar a cabo operaciones de consulta sobre elementos RBAC y sus relaciones.
- La funcionalidad a nivel de sistema define funciones para la creación de sesiones de usuarios que incluyen la activación/desactivación de roles, etc.

A continuación se van a mostrar más en detalle tanto el modelo de referencia como la especificación funcional de RBAC, pero sólo se entra en detalle de aquellos modelos que se han utilizado en el trabajo.

2.3.1. Modelo de Referencia de RBAC

El modelo de referencia de RBAC se define en términos de cuatro modelos: Core RBAC, Hierarchical RBAC, Static Separation of Duty Relations y Dynamic Separation of Duty Relations.

- *Core RBAC* (CRBAC) define una colección mínima de elementos, conjuntos de elementos y relaciones para conseguir un sistema de control de acceso basado en roles. Incluye la asignación de usuarios a roles y de permisos a roles, consideradas fundamentales en cualquier sistema RBAC. Además, CRBAC introduce el concepto de la activación del rol como parte de la sesión de un usuario. Cualquier sistema RBAC requiere la existencia del CRBAC, pero el resto de componentes son independientes unos de otros y pueden ser implementados (o no) por separado.
- *Hierarchical RBAC* (HRBAC) añade relaciones para soportar jerarquía de roles. Una jerarquía es matemáticamente un orden parcial definiendo una relación de rango entre roles, donde los roles con más rango heredan los permisos de los roles de menor rango y los roles de menor rango heredan los usuarios de rangos superiores. Además, HRBAC se dirige hacia la asignación simple de usuarios y permisos a

los roles introduciendo el concepto de conjunto de usuarios y permisos autorizados para roles.

- *Static Separation of Duty Relations* (SSDR) añade relaciones exclusivas entre roles y asignaciones de usuarios. Debido al riesgo de inconsistencias con la separación estática de las relaciones de responsabilidades y la herencia de relaciones de una jerarquía de roles, el modelo SSDR define relaciones tanto de la existencia como de la ausencia de jerarquía de roles.
- *Dynamic Separation of Duty Relations* (DSDR) define relaciones exclusivas entre los roles que se activan como parte de la sesión de usuarios.

Debido a las características de la aplicación que se verán en capítulos posteriores, en el trabajo sólo se va a utilizar el modelo de Core RBAC, por lo que es él único que se comenta en detalle.

Core RBAC

El conjunto de elementos y relaciones del modelo de Core RBAC (CRBAC) se definen en la figura 2–2. CRBAC incluye un conjunto de cinco elementos básicos de datos llamados `usuarios`, `roles`, `objetos`, `operaciones` y `permisos`. El modelo RBAC completo se define fundamentalmente en términos de usuarios individuales que se asignan a roles y permisos que se asignan a roles. Por lo tanto, un rol es una forma de relacionar muchos a muchos usuarios y permisos. Además, el modelo CRBAC incluye un conjunto de sesiones donde cada sesión refleja un usuario y un subconjunto de roles activados que se asignan a ese usuario.

El significado de los elementos básicos de este modelo CRBAC son:

- `Objeto`: puede ser cualquier recurso del sistema que necesite control de acceso, como puede ser ficheros, impresoras, bases de datos, etc.
- `Operación`: funcionalidad ejecutable por el usuario.

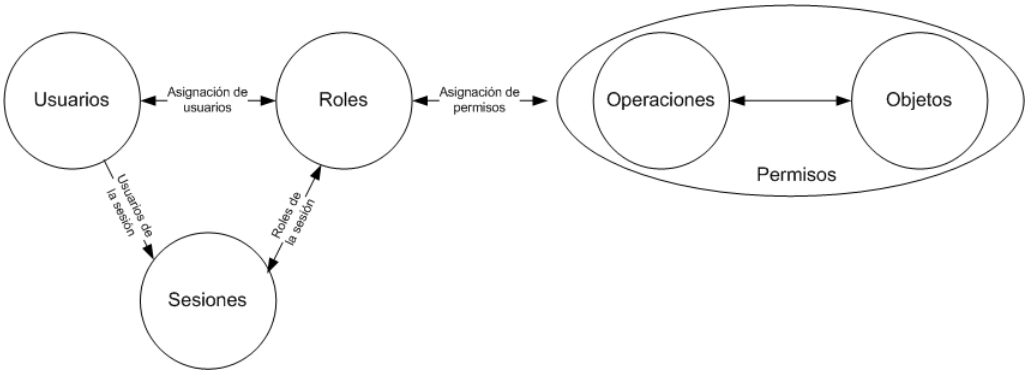

Figura 2–2: Elementos y relaciones del modelo de Core RBAC

- `Permiso`: consentimiento de llevar a cabo una operación sobre alguno de los objetos RBAC.
- `Rol`: función dentro del contexto de una organización.
- `Usuario`: persona que puede acceder al sistema.

Los tipos de operaciones y objetos que controla RBAC dependen del tipo de sistema en el cuál están implementados.

La figura 2–2 muestra gráficamente, además de los elementos básicos que componen RBAC, las relaciones existentes entre los distintos elementos. Estas relaciones son la asignación de usuarios a roles y la asignación de permisos a roles. Las flechas indican relaciones muchos-a-muchos (por ejemplo, un usuario puede ser asignado a uno o más roles, y un rol puede ser asignado a uno o más usuarios). Esto proporciona flexibilidad y granularidad a la asignación de permisos a roles y de usuarios a roles.

Cada sesión representa un usuario y sus roles, por ejemplo, un usuario establece una sesión durante la cual el usuario activa un subconjunto de roles que tiene asignados. Cada sesión está asociada a un único usuario y cada usuario puede estar asociado con una o más sesiones. Existen dos funciones relacionadas con la sesión, una que devuelve los roles activos en una sesión y otra que devuelve el usuario asociado a la sesión. Los permisos que tendrá el usuario en el sistema serán los permisos asignados a los roles que están activos durante la sesión del usuario.

2.3.2. Especificación Funcional de RBAC

La especificación funcional de RBAC especifica las operaciones para la creación y mantenimiento del conjunto de elementos y relaciones de RBAC. También especifica funciones para llevar a cabo consultas administrativas y funciones del sistema para crear y administrar los atributos de RBAC como sesiones de usuarios y tomar decisiones de control de acceso.

La especificación se realiza para cada uno de los cuatro modelos que conforman el estándar RBAC, pero en este trabajo solo se muestra la especificación funcional para CRBAC puesto que es el único modelo que se va a utilizar.

Core RBAC

A continuación se comentan aquellas operaciones necesarias para administrar correctamente los elementos que se encuentran en CRBAC.

- `AddUser`: esta operación crea un nuevo usuario RBAC. La operación solo es válida si el nuevo usuario no existe todavía como miembro del conjunto de `usuarios`. El siguiente esquema describe formalmente la operación:

$$AddUser(user : NAME) \triangleleft$$
$$user \notin USERS$$
$$USERS' = USERS \cup \{user\}$$
$$user\_sesions' = user\_sessions \cup \{user \mapsto \emptyset\} \triangleright$$

- `DeleteUser`: esta operación elimina un usuario existente de la base de datos de RBAC. La operación es válida si y solo si el usuario que va a ser borrado es miembro del conjunto de `usuarios`. Los datos de usuarios y roles que tenía asignados se actualizan. Es una decisión de implementación cómo proceder con la sesión del usuario a ser borrado. El sistema RBAC puede esperar a que el usuario termine su sesión normalmente o puede forzar su finalización. El siguiente esquema describe formalmente la operación:

$DeleteUser(user : NAME) \triangleleft$

$user \in USERS$

$[\forall s \in SESSIONS \bullet s \in user\_sessions(user)$

$\Rightarrow DeleteSessions(s)]$

$UA' = UA \setminus \{r : ROLES \bullet user \mapsto r\}$

$assigned\_users' = \{r : ROLES \bullet r \mapsto (assigned\_user(r)$

$\{user\})\}$

$USERS' = USERS \setminus \{user\} \triangleright$

- `AddRole`: esta operación crea un nuevo rol. La operación es válida si y solo si el nuevo rol no existe todavía como miembro del conjunto de `roles`. Inicialmente, el nuevo rol no tendrá ningún usuario ni ningún permiso asignado. El siguiente esquema describe formalmente la operación:

$AddRole(role : NAME) \triangleleft$

$role \notin ROLES$

$ROLES' = ROLES \cup \{role\}$

$assigned\_users' = assigned\_users \cup \{role \mapsto \emptyset\}$

$assigned\_permissions' = assigned\_permissions \cup \{role \mapsto \emptyset\} \triangleright$

- `DeleteRole`: esta operación elimina un rol existente de la base de datos de RBAC. La operación es válida si y solo si el rol que va a ser borrado es miembro del conjunto de `roles`. Es una decisión de implementación cómo proceder con las sesiones que tienen activo el rol que va a ser borrado. El sistema RBAC puede esperar a que la sesión termine normalmente o puede eliminar el rol de las sesiones mientras se permite que las sesiones continúen. El siguiente esquema describe formalmente la operación:

$DeleteRole(role : NAME) \triangleleft$

$role \in ROLES$

$[\forall s \in SESSIONS \bullet role \in session\_roles(s)$

$\Rightarrow DeleteSessions(s)]$

$UA' = UA \setminus \{u : USERS \bullet u \mapsto role\}$

$assigned\_users' = assigned\_users \setminus \{role \mapsto$

$(assigned\_users(role)\}$

$PA' = PA \setminus \{op : OPS, obj : OBJS \bullet (op, obj) \mapsto role\}$

$assigned\_permissions' = assigned\_permissions \setminus \{role \mapsto$

$assigned\_permissions(role)\}$

$ROLES' = ROLES \setminus \{role\} \triangleright$

- `AssignUser`: esta operación asigna un usuario a un rol. La operación es válida si y solo si el usuario existe en el conjunto de `usuarios`, el rol existe en el conjunto de `roles` y el usuario no ha sido todavía asignado al rol. El siguiente esquema describe formalmente la operación:

$AssignUser(user, role : NAME) \triangleleft$

$user \in USERS; role \in ROLES; (user \mapsto role) \notin UA$

$UA' = UA \cup \{user \mapsto role\}$

$assigned\_users' = assigned\_users \setminus \{role \mapsto (assigned\_users(role)\}$

$\cup \{role \mapsto (assigned\_users(role) \cup \{user\})\} \triangleright$

- `DeassignUser`: esta operación elimina la asignación de un usuario de un rol. La operación es válida si y solo si el usuario existe en el conjunto de `usuarios`, el rol existe en el conjunto de `roles` y el usuario ha sido asignado al rol. El siguiente esquema describe formalmente la operación:

$DeassignUser(user, role : NAME) \triangleleft$

$user \in USERS; role \in ROLES; (user \mapsto role) \in UA$

$[\forall s \in SESSIONS \bullet s \in user\_session(user) \land role$

$\in session\_roles(s) \Rightarrow DeleteSessions(s)]$

$UA' = UA \cup \{user \mapsto role\}$

$assigned\_users' = assigned\_users \setminus \{role \mapsto (assigned\_users(role)\} \cup$

$\{role \mapsto (assigned\_users(role) \cup \{user\})\} \triangleright$

- `GrantPermission`: esta operación asigna a un rol el permiso de llevar a cabo una operación sobre un objeto. La operación es válida si y solo si el par (operación, objeto) representa un permiso, y el rol es un miembro del conjunto de `roles`.El siguiente esquema describe formalmente la operación:

$GrantPermission(object, operation, role : NAME) \triangleleft$

$(operation, object) \in PERMS; role \in ROLES$

$PA' = PA \cup \{(operation, object) \mapsto role\}$

$assigned\_permissions' = assigned\_permissions \setminus \{role \mapsto$

$(assigned\_permissions(roles)\} \cup \{role \mapsto (assigned\_permissions(role)$

$\cup \{(operation, object)\})\} \triangleright$

- `RevokePermission`: esta operación cancela el permiso de llevar a cabo una operación sobre un objeto del conjunto de permisos asignados a un rol. La operación es válida si y solo si el par (operación, rol) representa un permiso, el rol existe en el conjunto de `usuarios` y el permiso está asignado al rol.El siguiente esquema describe formalmente la operación:

$RevokePermission(object, operation, role : NAME) \triangleleft$

$(operation, object) \in PERMS; role \in ROLES; ((operation, object)$

$\mapsto role) \in PA$

$PA' = PA \setminus \{(operation, object) \mapsto role\}$

$assigned\_permissions' = assigned\_permissions \setminus \{role \mapsto$

$(assigned\_permissions(role)\} \cup \{role \mapsto (assigned\_permissions(role)$

$\setminus \{(operation, object)\})\} \triangleright$

Soporte de funciones del sistema para Core RBAC

Además de las operaciones de administración de usuarios, roles y permisos, RBAC proporciona especificación funcional para la administración de las sesiones.

- `CreateSession(user, session)`: esta operación crea una nueva sesión con el usuario y un grupo de roles activos. La operación es válida si y solo si el usuario es miembro del conjunto de `usuarios` y el conjunto de roles activos es un subconjunto de roles asignados al usuario. El siguiente esquema describe formalmente la operación:

$$
\begin{aligned}
&CreateSession(user : NAME; ars : 2^{NAMES}; session : NAME) \triangleleft \\
&\quad user \in USERS; ars \subseteq \{r : ROLES | (user \mapsto r) \in UA\}; \\
&\quad\quad session \notin SESSIONS \\
&\quad SESSIONS' = SESSIONS\{session\} \\
&\quad user\_sessions' = user\_sessions \backslash \{user \mapsto user\_sessions(user)\} \cup \\
&\quad\quad \{user \mapsto (user\_sessions(user) \cup \{session\})\} \\
&\quad session\_roles' = session\_roles \cup \{session \mapsto ars\} \triangleright
\end{aligned}
$$

- `DeleteSession(user, session)`: esta operación elimina la sesión del usuario. La operación es válida si y solo si la sesión es miembro del conjunto de `sesiones`, el usuario existe en el conjunto de `usuarios` y el usuario es el propietario de la sesión. El siguiente esquema describe formalmente la operación:

$$
\begin{aligned}
&DeleteSession(user, session : NAME) \triangleleft \\
&\quad user \in USERS; session \in SESSIONS; session \in user\_sessions(user) \\
&\quad user\_sessions' = user\_sessions \backslash \{user \mapsto user_s essions(user)\} \cup \\
&\quad\quad \{user \mapsto (user\_sessions(user) \setminus \{session\})\} \\
&\quad session\_roles' = session\_roles \backslash \{session \mapsto session\_roles(session)\} \\
&\quad SESSIONS' = SESSIONS \setminus \{session\} \triangleright
\end{aligned}
$$

- `AddActiveRole`: esta operación añade un rol como rol activo de la sesión cuyo usuario es un usuario. La operación es válida si y solo si el usuario es miembro del conjunto de `usuarios`, el rol es miembro del conjunto de `roles`, la sesión es miembro del conjunto de `sesiones`, el rol está asignado al usuario y la sesión pertenece al usuario. El siguiente esquema describe formalmente la operación:

$$AddActiveRole(user, session, role : NAME) \lhd$$
$$user \in USERS; session \in SESSIONS; role \in ROLES;$$
$$session \in user\_sessions(user)$$
$$(user \mapsto role) \in UA; role \notin session\_roles(session)$$
$$session\_roles' = session\_roles \backslash \{session \mapsto session\_roles(session)\} \cup$$
$$\{session \mapsto (session\_roles(session) \cup \{role\})\} \rhd$$

- `DropActiveRole`: esta operación elimina un rol de la sesión activa de un usuario. La operación es válida si y solo si el usuario es miembro del conjunto de `usuarios`, la sesión es miembro del conjunto de `sesiones`, el usuario es el propietario de la sesión y el rol es un rol activo de la sesión.El siguiente esquema describe formalmente la operación:

$$DropActiveRole(user, session, role : NAME) \lhd$$
$$user \in USERS; session \in SESSIONS; role \in ROLES$$
$$session \in user\_sessions(user); role \in sessions\_roles(session)$$
$$session\_roles' = session\_roles \backslash \{session \mapsto session\_roles(session)\} \cup$$
$$\{session \mapsto (session\_roles(session) \setminus \{role\})\} \rhd$$

- `CheckAccess`: esta operación devuelve un `boolean` que indica si el usuario de una sesión tiene permiso o no para llevar a cabo una operación sobre un objeto. La operación es válida si y solo si la sesión es miembro del conjunto de `sesiones`, el objeto es miembro del conjunto de `objetos` y la operación es miembro del conjunto de `operaciones`. El usuario de la sesión tiene permiso de llevar a cabo la operación si y solo si el permiso está asignado a (al menos) uno de los roles activos de la sesión.El siguiente esquema describe formalmente la operación:

$$ChekAcces(session, operation, object : NAME; out, result : BOOLEAN) \lhd$$
$$session \in SESSIONS; operation \in OPS; object \in OBJS$$
$$result = (\exists r : ROLES \bullet r \in session\_roles(session) \land$$
$$((operation, object) \mapsto r) \in PA) \rhd$$

Funcionalidad de consulta para Core RBAC

Core RBAC proporciona también operaciones de consulta sobre los elementos de RBAC y sus relaciones.

- `AssignedUsers`: esta operación devuelve el conjunto de usuarios asignados a un rol. La operación es válida si y solo si el rol es miembro del conjunto de `roles`. El siguiente esquema describe formalmente la operación:

$$AssignedUsers(role : NAME; out\ result : 2^{USERS}) \triangleleft$$
$$role \in ROLES$$
$$result = \{u : USERS | (u \mapsto role) \in UA\} \triangleright$$

- `AssignedRoles`: esta operación devuelve el conjunto de roles asignados a un usuario. La operación es válida si y solo si el usuario es miembro del conjunto de `usuarios`. El siguiente esquema describe formalmente la operación:

$$AssignedRoles(user : NAME; result : 2^{ROLES}) \triangleleft$$
$$user \in USERS$$
$$result = \{r : ROLES | (user \mapsto r) \in UA\} \triangleright$$

Soporte de funciones avanzadas del sistema para Core RBAC

Finalmente, Core RBAC proporciona funciones avanzadas para la consulta de los permisos.

- `RolePermissions`: esta operación devuelve el conjunto de permisos (operación, objeto) de un rol. La operación es válida si y solo si el rol es miembro del conjunto de `roles`.El siguiente esquema describe formalmente la operación:

$$RolePermissions(role : NAME; result : 2^{PERMS}) \triangleleft$$
$$role \in ROLES$$
$$result = \{op : OPS; obj : OBJS | ((op, obj) \mapsto role) \in PA\} \triangleright$$

- `UserPermissions`: esta operación devuelve el conjunto de permisos que un usuario tiene asignados a través de sus roles. Esta operación es válida si y solo si el usuario

es miembro del conjunto de `usuarios`.El siguiente esquema describe formalmente la operación:

$$UserPermissions(user : NAME; result : 2^{PERMS}) \triangleleft$$
$$user \in USERS$$
$$result = \{r : ROLES, op : OPS; obj : OBJS | (user \mapsto r) \in$$
$$UA \wedge ((op, obj) \mapsto r) \in PA \bullet (op, obj)\} \triangleright$$

- `SessionRoles`: esta operación devuelve el conjunto de roles activos asociados a una sesión. La operación es válida si y solo si la sesión es miembro del conjunto de `sesiones`.El siguiente esquema describe formalmente la operación:

$$SessionRoles(session : NAME; out\ result : 2^{ROLES}) \triangleleft$$
$$session \in SESSIONS$$
$$result = session_roles(session) \triangleright$$

- `SessionPermissions`: esta operación devuelve el conjunto de permisos de una sesión, es decir, los permisos asignados a los roles activos de dicha sesión. La operación es válida si y solo si la sesión es miembro del conjunto de `sesiones`. El siguiente esquema describe formalmente la operación:

$$SessionPermissions(session : NAME; out\ result : 2^{PERMS}) \triangleleft$$
$$session \in SESSIONS$$
$$result = \{r : ROLES, op \in OPS; obj \in OBJS | r \in session\_roles(session)$$
$$\wedge\ ((op, obj) \mapsto r) \in PA \bullet (op, obj)\} \triangleright$$

- `RoleOperationsOnObject`: esta operación devuelve el conjunto de operaciones que se permiten llevar a cabo sobre un objeto para un rol. La operación es válida si y solo si el rol es miembro del conjunto de `roles` y el objeto es miembro del conjunto de `objetos`. El siguiente esquema describe formalmente la operación:

$$RoleOperationsOnObject(role : NAME; result : 2^{OPS}) \triangleleft$$

$role \in ROLES$

$obj \in OBJS$

$result = \{op : OPS | ((op, obj) \mapsto role) \in PA\}$ ▷

- `UserOperationsOnObject`: esta operación devuelve el conjunto de operaciones que se permiten llevar a cabo sobre un objeto para un usuario, obtenidas a través de los roles que tiene asignados. Esta operación es válida si y solo si el usuario es miembro del conjunto de `usuarios` y el objeto es miembro del conjunto de `objetos`.El siguiente esquema describe formalmente la operación:

$UserOperationsOnObject(user : NAME; obj : NAME; result : 2^{OPS})$ ◁

$user \in USERS$

$obj \in OBJS$

$result = \{r : ROLES, op : OPS | (user \mapsto r) \in UA \wedge$

$((op, obj) \mapsto r) \in PA \bullet op\}$ ▷

2.4. LDAP

El Lightweight Directory Access Protocolo (LDAP) (para más información consultar [20] y [21]) es un servicio de directorio con repositorio de usuarios y roles. LDAP define un método estándar para acceder y actualizar información en un directorio.

LDAP dirige las solicitudes de una aplicación cliente a la infraestructura de directorios, proporcionando de esta forma la protección contra fallos, el equilibrio de carga y la seguridad necesaria.

Además, LDAP define el contenido de los mensajes que se intercambian el cliente LDAP y el servidor LDAP. Los mensajes especifican las operaciones que pide el cliente, las respuestas del servidor y el formato de los datos de los mensajes. Los mensajes LDAP son transportados sobre TCP/IP, un protocolo orientado a conexión, por lo que también se definen operaciones para conectar/desconectar una sesión entre el cliente y el servidor.

Por este motivo, los directorios basado en LDAP son muy utilizados en las granes empresas. Para ellas es muy importante que pueda soportar jerarquías de entradas orientadas a objetos, además de buscar y modificar atributos sobre TCP/IP. También es muy importante el modelo de la lógica, indicar que se define con los mensajes y sus tipos, cómo se organiza el directorio, qué operaciones están permitidas, como se protege la información, etc.

2.4.1. Qué es un directorio

Un directorio es un listado de información sobre objetos ordenado en cualquier orden, el cuál da información detallada sobre cada objetos. Ejemplos comunes son la guía telefónica de una ciudad o el catálogo de una biblioteca.

En términos informáticos, un directorio es una base de datos especializada, también llamada repositorio de datos, que almacena información sobre objetos.

Los directorios permiten a los usuarios o a las aplicaciones encontrar recursos que tienen ciertas características necesarias para una tarea en particular. Por ejemplo, un directorio de usuarios puede ser usado para buscar el correo electrónico de una persona o su número de fax.

2.4.2. Directorio de seguridad

Cuando se mandan datos sobre una red insegura, internamente o externamente, puede necesitar proteger información sensible durante el transporte. También suele necesitar saber quién está pidiendo la información y quien la está enviando. El termino seguridad que se trata LDAP en [21] cubre cuatro aspectos:

- Autenticación: asegurarse de que el cliente es (máquina o persona) realmente quien dice ser.
- Integridad: asegurarse que la información que llega es realmente la misma que se envía.

- Confidencialidad: protección de la información por medio de encriptación de datos.
- Autorización: asegurarse que el cliente está realmente autorizado a realizar la petición. Este normalmente se realiza asignando control de acceso.

2.4.3. LDAP y RBAC

Como se ha visto, la autorización en un LDAP se realiza asignando un control de acceso. En este trabajo se utiliza LDAP junto con RBAC para cubrir esta parte de la seguridad.

Para implementar RBAC en una aplicación Web [22], se utiliza el servidor de directorios LDAP como servidor de perfiles (roles) que contiene la información de los perfiles. La información de los perfiles que existe en el servidor de perfiles se utiliza para el control de acceso a través del LDAP de una forma segura (normalmente sobre SSL).

2.5. Conclusiones

En este capítulo se han presentado las metodologías y estándares en los que se sitúa el trabajo que se desarrolla en este máster. A lo largo del capítulo se han analizado las propuestas relacionadas con la intención de facilitar el entendimiento de la solución propuesta.

En primer lugar se ha estudiado la metodología DI-SFM. En esta metodología se ha detectado la necesidad de incluir artefactos que faciliten el control de acceso. Además, se ha presentado la metodología en la cuál está basada, Métrica 3.

Para abordar de forma precisa el control de acceso se ha considerado una de las aproximaciones más utilizadas, RBAC. Debido a la importancia que tiene en el trabajo esta metodología, es este capítulo se ha analizado el conjunto de operaciones propuestas por RBAC.

Para finalizar, se ha presentado LDAP para situar el trabajo adecuadamente. Este directorio de de usuarios y roles es utilizado en el trabajo como el directorio encargado de gestionar los usuarios y pasar a la aplicación aquellos datos necesarios para garantizar que se cumple la seguridad.

Capítulo 3
GESTIÓN DE IDENTIDAD Y CONTROL DE ACCESO EN ATICA

En este capítulo se presenta una propuesta para la gestión de identidad y el control de acceso basado en roles. Esta propuesta es la solución que se está siguiendo en la empresa *Dimensión Informática* para el desarrollo de sus productos actuales.

La propuesta se ha desarrollado mediante un proceso de abstracción de la propuesta de RBAC, simplificando su modelo de referencia y extendiendo las operaciones de gestión de la identidad y control de acceso basado en roles para acoplarlo a las necesidades del proyecto ATICA.

El capítulo se estructura de la siguiente forma: en primer lugar, se introduce la gestión de identidad y control de acceso en ATICA. A continuación, se presenta la adaptación de RBAC a las necesidades detectadas en ATICA, seguido de la especificación funcional del sistema de información y del ámbito de implementación de las operaciones.

3.1. Introducción

La solución final del *sistema de información para la gestión de centros educativo* se aborda desde dos frentes distintos. Por un lado tenemos la gestión de identidad implementado por Telefónica Soluciones (que a partir de ahora será referido como LDAP) y por otro el control de acceso a la aplicación implementado por Dimensión Informática (que a partir de ahora será referido como ATICA).

Cuando un usuario quiere iniciar sesión (ver figura 3–1), lo primero que hace es invocar el servicio `IniciarSesionUsuario` a través de una secuencia segura. Con esto, `IniciarSesionUsuario` envía la notificación al LDAP, que será el encargado de comprobar que existe el usuario y que la contraseña es válida.

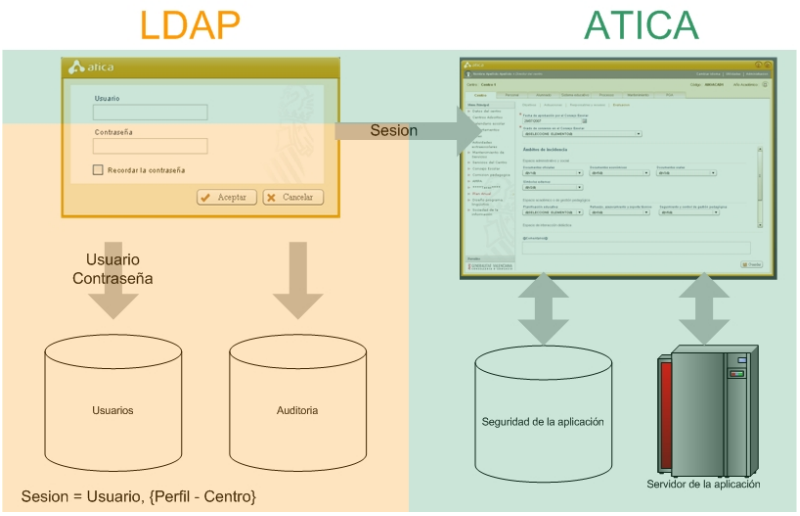

Figura 3–1: Comunicación entre LDAP y Atica

La contraseña se pasa al LDAP en forma de una clave secreta no reversible generada mediante una función hash (es una fórmula matemática a través de la cual, dada una cadena de caracteres, se obtiene un resultado único. La modificación de un solo carácter altera el resultado y, obtenido un resultado no se puede calcular la cadena de caracteres original). LDAP, al recibir el paquete de autenticación, comprueba la existencia del usuario y realiza la misma función hash sobre la contraseña que tiene almacenada y compara ambos resultados. Si concuerdan, devuelve un *token* con los permisos del usuarios (que más adelante llamaremos *Sesión*). Este token está formado por el usuario, y uno o varios conjuntos formados por perfil/es a los que está asociado el usuario y el/los centros en los que se aplica dicho perfil.

3.2. Adaptación de RBAC a las necesidades detectadas en ATICA

Como se ha comentado en el capítulo anterior (ver en apartado 2.3), RBAC es un estándar para el control de acceso basado en roles. La noción básica de RBAC es que los permisos se asignan a roles y los usuarios también se asignan a roles. Según el significado que da RBAC de los elementos básicos de su modelo, podemos crear una equivalencia entre sus elementos y los que conforman el proyecto ATICA.

- `Objeto`: puede ser cualquier recurso del sistema que necesite control de acceso. En el proyecto los recursos que necesitan control de acceso son las operaciones, las entradas de menú e incluso las pantallas que las contienen.
- `Operación`: funcionalidad ejecutable por el usuario. Con esta definición se refiere a que tipo de funcionalidad se puede realizar sobre los objetos. En el caso del proyecto, la única operación definida es si tiene o no acceso a un objeto. Por lo tanto, este elemento se podría suprimir sin que el resto de elementos sufrieran cambios.
- `Permiso`: consentimiento de llevar a cabo una operación sobre alguno de los objetos RBAC. En nuestro caso, al no tener operaciones se simplifica a tener o no permiso para acceder al objeto.
- `Rol`: función dentro del contexto de una organización. En el caso del proyecto están los directores, secretarios, administrativos y jefes de estudios a nivel del centro, y a nivel de servicios centrales y territoriales tenemos gestión de centro o gestión de personal entre otros.
- `Usuario`: persona que puede acceder al sistema. Como cualquier usuario del proyecto.

A continuación se muestra una tabla resumen donde se pueden ver las equivalencias encontradas entro los elementos de RBAC y los elementos que conforman el proyecto de ATICA(ver tabla 3–1).

Tabla 3–1: Elementos de RBAC en ATICA

RBAC	ATICA
usuarios	usuarios del Centro, Servicios Centrales y Servicios Territorials
roles	perfiles, actores o roles
objetos	pestañas, menús, pantallas o botones
operaciones	–
permisos	permisos

Por lo tanto, el modelo de RBAC se simplificaría y quedaría como muestra la figura 3–2, donde se puede ver que la diferencia se encuentra en que los permisos se asignan a los objetos directamente. Esta asignación es directa porque, como se ha comentado anteriormente, las operaciones por defecto son acceso o no acceso (existencia de permiso o ausencia del mismo) y por lo tanto se pueden obviar, simplificando de esta forma el sistema a implementar.

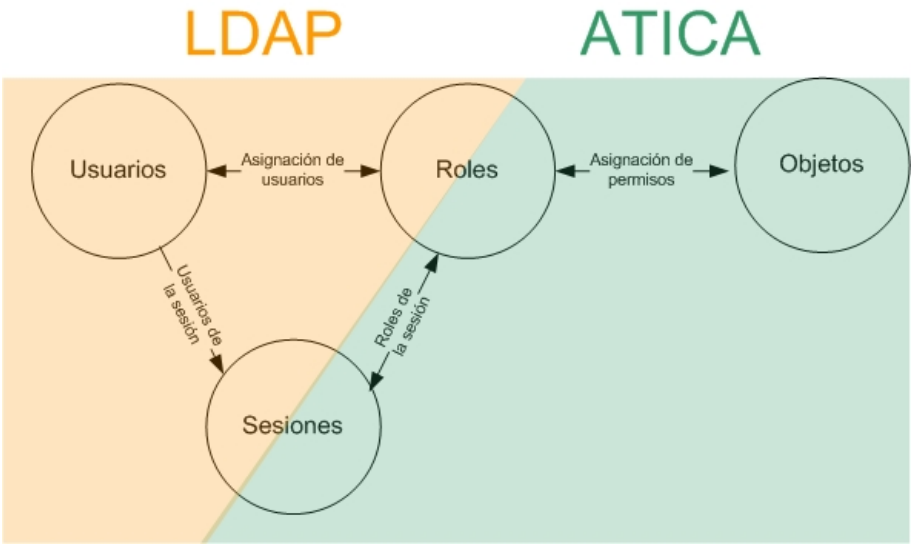

Figura 3–2: Modelo simplificado de RBAC

En resumen, el sistema va a estar formado por una serie de usuarios asignados a roles. Dichos roles tendrán asignados unos permisos (acceso o denegación de acceso) sobre unos objetos. Estos objetos son desde entradas de menú hasta operaciones sobre las clases. Además, cada usuario podrá tener activas una serie de sesiones que

estarán formadas por el usuario, el rol (y en el caso de ATICA también el centro de trabajo del usuario).

Como se puede ver en la figura 3–2, la parte de gestión de usuarios, asignación de roles a usuarios y creación de las sesiones del usuario estará totalmente soportada por la parte de LDAP, y la asignación de permisos a objetos, y conocer durante la ejecución de la aplicación qué permisos tiene un rol y que roles están activos en la sesión estará soportado por ATICA.

3.3. Especificación Funcional del sistema de información para la gestión de centros educativos

La especificación funcional de RBAC, como se ha visto en el capítulo anterior, da soporte funcional al modelo de referencia. Como el proyecto aquí presentado utiliza un modelo simplificado del modelo de referencia de RBAC, la especificación funcional de ATICA puede utilizar parte de la especificación funcional de RBAC.

A la especificación funcional de RBAC ha de sumarse las operaciones identificadas en los trabajos de investigación realizados por la autora del trabajo (ver [1–3]). De esta manera se extiende la especificación funcional de RBAC para cubrir todas las necesidades detectadas para las aplicaciones Web que se implementan hoy en día.

Primero, se van a ver y categorizar las operaciones presentadas en la especificación funcional de RBAC para a continuación presentar las operaciones identificadas en los trabajos de investigación. Las operaciones de la especificación funcional de RBAC se presentan en castellano dado que el idioma del resto de funcionalidad del proyecto está en castellano. Además, los nombres siguen la nomenclatura utilizada durante el proyecto, dada por la guía del analista.

De la especificación funcional de CRBAC se utilizan las siguientes operaciones:

- De las operaciones para administrar correctamente los elementos, se van a utilizar:
  - `AddUser`, que en el proyecto se renombra por `CrearUsuario`

  - `DeleteUser`, que en el proyecto se renombra por `EliminarUsuario`
  - `AddRole`, que en el proyecto se renombra por `CrearRol`
  - `DeleteRole`, que en el proyecto se renombra por `EliminarRol`
  - `AssignUser`, que en el proyecto se renombra por `AsignarUsuarioARol`
  - `DeassignUser`, que en el proyecto se renombra por `DesasignarUsuarioDeRol`
  - `GrantPermission`, que en el proyecto se renombra por `AsignarPermisoARol`
  - `RevokePermission`, que en el proyecto se renombra por `DesasignarPermisoDeRol`

- De las operaciones que dan soporte al sistema para CRBAC, se van a utilizar:
  - `CreateSession`, que en el proyecto se renombra por `IniciarSesion`
  - `DeleteSession`, que en el proyecto se renombra por `CerrarSesion`

  Las operaciones `CheckAccess`, `AddActiveRole` y `DropActiveRole` no se van a implementar dado que los roles que se van a asignar a una sesión van a ser siempre los roles del usuario.
- De la funcionalidad de consulta para CRBAC, se van a utilizar:
  - `AssignedUsers`, que en el proyecto se renombra por `BuscarUsuariosDeRol`
  - `AssignedRoles`, que en el proyecto se renombra por `BuscarRolesDeUsuario`
- Por último, de la funcionalidad de soporte de funciones avanzadas, se van a utilizar:
  - `RolePermissions`, que en el proyecto se renombra por `BuscarPermisosDeRol`
  - `SessionRoles`, que en el proyecto se renombra por `BuscarRolesDeSesion`
  - `SessionPermissions`, que en el proyecto se renombra por `BuscarPermisosDeSesion`

  Las operaciones `UserPermissions`, `RoleOperationsOnObject` y `UserOperationsOnObject` son operaciones que no van a ser utilizadas en el proyecto por no existir en el modelo simplificado de CRBAC las operaciones. Además, el resto de operaciones se utiliza pero sin tener en cuenta la variable `operación` de RBAC.

En este conjunto de operaciones, además de estar definidas únicamente las operaciones de creación, borrado y algunas de recuperación, le faltan varias operaciones más. Por un lado, hacen falta operaciones que permitan modificar los datos existentes tanto de la información de los usuarios como de los permisos u objetos. Además, hacen faltan operaciones para la identificación de usuarios (para poder interactuar con el sistema) u operaciones para la administración de objetos susceptibles de tener control de acceso. Las operaciones que extiende la funcionalidad de RBAC son:

- Modificación de datos:
  - `ModificarUsuario`
  - `ModificarContraseña`
  - `ModificarRol`
- Identificación de usuarios:
  - `RecordarContraseña`
- Administración de objetos:
  - `CrearObjetos`
  - `ModificarObjetos`
  - `EliminarObjetos`

Por lo tanto, las operaciones identificadas para el proyecto ATICA (o cualquier otro proyecto realizado en Dimensión Informática con unas características similares) tanto del modelo Core RBAC [4] como de los trabajos de investigación realizados en [1–3], se clasifican en los siguientes tipos:

1. Operaciones que dan soporte a la administración de usuarios (ver figura 3–3), algunas basadas en el modelo de RBAC y otras en los trabajos de investigación: `CrearUsuario`, `EliminarUsuario`, `ModificarUsuario` y `ModificarContraseña`.

Figura 3–3: Operaciones de administración de usuarios

2. Operaciones que proveen soporte a la administración de sesiones (identificación de usuarios en el sistema), basadas tanto en RBAC como en los trabajos de investigación (ver figura 3–4): `IniciarSesion`, `CerrarSesion` y `RecordarContraseña`.

Figura 3–4: Operaciones de administración de sesiones

3. Operaciones que dan soporte a la administración de objetos (software en el caso de ATICA), como extensión a la funcionalidad definida por RBAC (ver figura 3–5): `CrearObjetos`, `ModificarObjetos` y `EliminarObjetos`.

Figura 3–5: Operaciones de administración de objetos

4. Operaciones que dan soporte a la administración de roles de usuarios, basadas en el modelo de RBAC, (ver figura 3–6): `CrearRol`, `EliminarRol`, `ModificarRol`, `AsignarUsuarioARol`, `DesasignarUsuarioDeRol`, `BuscarUsuariosDeRol` y `BuscarRolesDeUsuario`.

Figura 3–6: Operaciones de administración de roles

5. Operaciones que dan soporte a la administración de permisos, basadas en el modelo de RBAC, (ver figura 3–7): `AsignarPermisosARol`, `DesasignarPermisosDeRol`, `BuscarPermisosDeRol` y `BuscarPermisosDeSesion`.

Figura 3–7: Operaciones de administración de permisos

3.4. Ámbito de implementación de las operaciones

Al estar la solución dividida en dos (LDAP por Telefónica y ATICA por Dimensión Informática), a continuación se presenta una tabla resumen (ver tabla 3–2)

donde se muestra para cada operación en que parte será implementada. Como se ha comentado anteriormente, en el LDAP se encuentran todo lo relacionado con usuarios, sesiones y asignación de usuarios a roles. En la parte de ATICA se encuentran los roles, con los permisos y objetos. Por lo tanto, las operaciones que involucren a los usuarios y a la creación de sesiones, serán implementadas en la parte del LDAP mientras que las de los permisos y los objetos estarán implementadas en ATICA. La implementación de la creación, modificación y eliminación de roles serán operaciones compartidas por los dos ámbitos (LDAP y ATICA) puesto que los roles son necesarios en los dos ámbitos. Un resumen gráfico de esta tabla se puede ver en la figura 3–8.

Tabla 3–2: Implementación de las operaciones en LDAP o en ATICA

Operación	LDAP	ATICA
`CrearUsuario`	X	
`EliminarUsuario`	X	
`ModificarUsuario`	X	
`ModificarContraseña`	X	
`CrearRol`	X	X
`EliminarRol`	X	X
`ModificarRol`	X	X
`AsignarUsuarioARol`	X	
`DesasignarUsuarioDeRol`	X	
`BuscarUsuariosDeRol`	X	
`BuscarRolesDeUsuario`	X	
`AsignarPermisoARol`		X
`DesasignarPermisoDeRol`		X
`BuscarPermisosDeRol`		X
`BuscarPermisosDeSesion`		X
`BuscarRolesDeSesion`		X
`IniciarSesionUsuario`	X	
`CerrarSesion`	X	
`RecordarContraseña`	X	
`CrearObjeto`		X
`ModificarObjeto`		X
`EliminarObjeto`		X

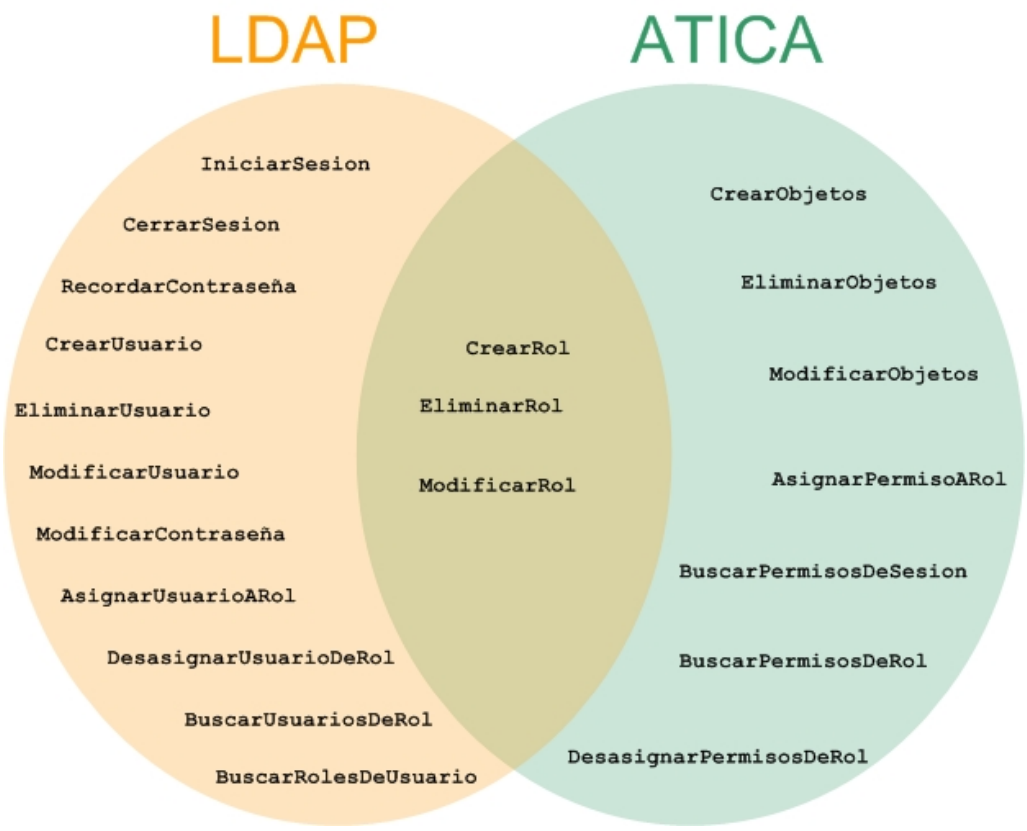

Figura 3–8: Ámito de implementación de las operaciones

3.5. Conclusiones

En este capítulo se ha presentado una especificación funcional para el control de acceso basado en roles. Esta especificación se ha definido a través de un proceso de abstracción de la especificación funcional ofrecida por RBAC. Además, esta especificación ha sido extendida con nuevas operaciones que la enriquecen y completan. Por último, se ha ha presentado el ámbito de implementación de cada una de las operaciones de la especificación funcional que están siendo llevadas a la práctica en el contexto de este trabajo de máster y del proyecto ATICA.

La especificación funcional presentada está abierta a la incorporación de nuevas operaciones y a la modificación de las existentes. Esta especificación pretende incorporar las características esenciales de la mayoría de sistemas con control de acceso basado en roles.

Capítulo 4
EXTENSIÓN DE DI-SFM PARA EL CONTROL DE ACCESO

En este capítulo se presenta una propuesta para extender la metodología DI-SFM (ver el apartado 2.1). Esta propuesta se ha desarrollado tras analizar las necesidades detectadas en el proyecto ATICA, un sistema de información para la gestión de centros educativos para la Conselleria de Educación. Estas extensiones cubren la propuesta completa, uniendo tanto la parte presentada para Dimensión Informática (implementada en ATICA, ver capítulo 3) como la parte a implementar por Telefónica (implementada en el LDAP), para conseguir una visión global de las necesidades, se presenta una extensión a la metodología.

El capítulo se estructura de la siguiente forma: en primer lugar, tras introducir las extensiones propuestas para DI-SFM, se presenta la extensión propuesta para la disciplina de Requisitos. Esta propuesta está basada en una batería de preguntas a realizar al cliente para capturar sus necesidades en el ámbito de la seguridad. A continuación, se presenta la extensión propuesta para la disciplina de Análisis. Esta propuesta consta del diseño de los casos de uso, clases y pantalla relacionadas con los requisitos capturados.

4.1. Introducción

Una vez presentadas las operaciones necesarias para el control de acceso basado en roles para el sistema de información de la gestión de centros educativos, a continuación se presenta la extensión de la metodología DI-SFM para introducir en la etapa

de requisitos y en la de diseño los artefactos necesarios para conseguir capturar las necesidades del usuario. En este apartado no se va a tener en cuenta que para este caso en particular Dimensión Informática no es propietaria de la parte del LDAP y se van a proponer las extensiones para un proyecto genérico donde puede o no puede pertenecer por completo a Dimensión Informática.

Por lo tanto, las extensiones propuestas son dos (ver figura 4–1): una primera para la disciplina de requisitos y una segunda para la disciplina de análisis. Para la disciplina de requisitos se propone introducir un asistente que guía al usuario, a través de una colección de preguntas, para conseguir capturar sus necesidades sobre seguridad en el ámbito del control de acceso basado en roles. En cuanto a la disciplina de análisis, se proponen el modelado de las operaciones presentadas en el apartado anterior.

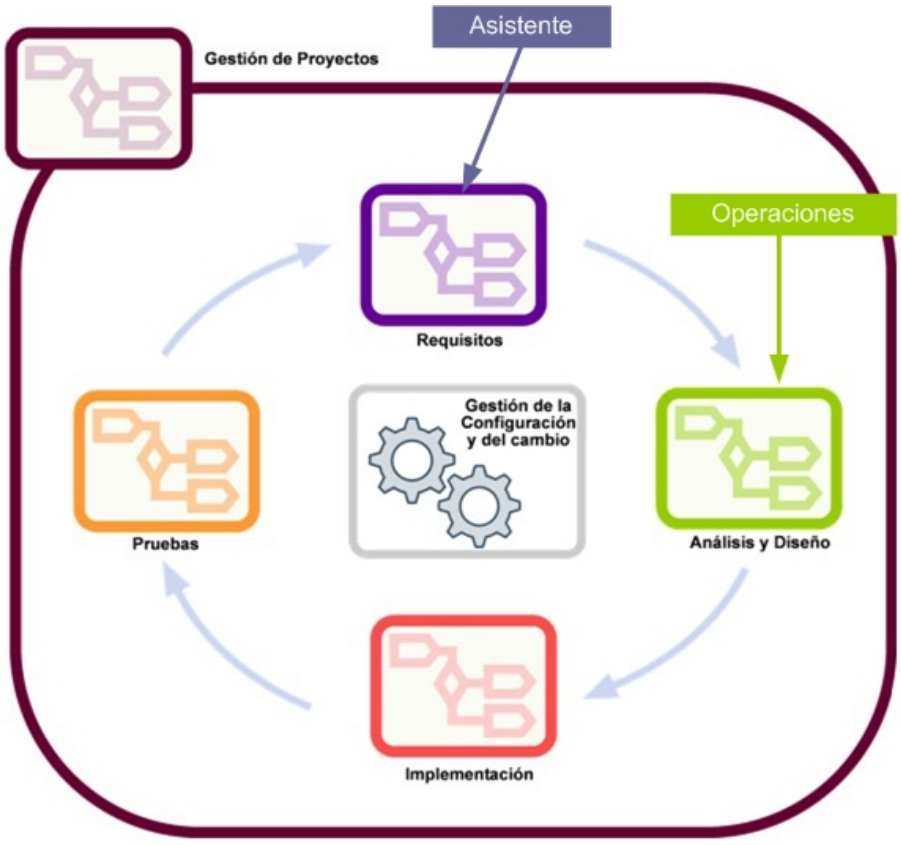

Figura 4–1: Elementos extendidos a incluir en DI-SFM

4.2. Extensión de la disciplina de Requisitos

En el contexto de la empresa, es interesante proporcionar guías metodológicas que ayuden en el diseño e implementación de las operaciones. En este apartado, se presenta un asistente que mejora y complementa el diseño de las operaciones a incluir en el sistema y su implementación.

El asistente, que actualmente se entrega al usuario como plantilla a rellenar manualmente, pero que se está implementando y probando una implementación del mismo, recoge información a través de preguntas, las cuales permiten determinar adecuadamente ciertas características funcionales de la aplicación, tales como qué operaciones se añaden, si se añaden o eliminan ciertos parámetros de las operaciones principales, etc.

Revisando las operaciones definidas se pueden deducir una serie de preguntas que ayudarán a identificar ciertas características imprescindibles para identificar las necesidades del cliente. Las características que se pueden capturar con este asistente son, entre otras, la posibilidad de dependencia entre las operaciones, el tipo de aplicación y el tipo de usuarios que van a utilizar el sistema. Estas preguntas se aplicarán siguiendo la secuencia de acciones propuesta a continuación.

4.2.1. Un asistente para mejorar la toma de requisitos

En este subapartado se presentan las preguntas con sus posibles respuestas. Como se ha comentado anteriormente, actualmente se está entregando al usuario en forma de cuestionario a rellenar en una hoja o digitalmente pero se está desarrollando y probando una herramienta asistente que automáticamente definirá las operaciones y sus parámetros.

En la figura 4–2 se puede ver un esquema de las preguntas que realiza el asistente para el control de acceso basado en roles. Cada rectángulo representa una pregunta. El nombre de las preguntas indica el orden y el color indica si la pregunta se

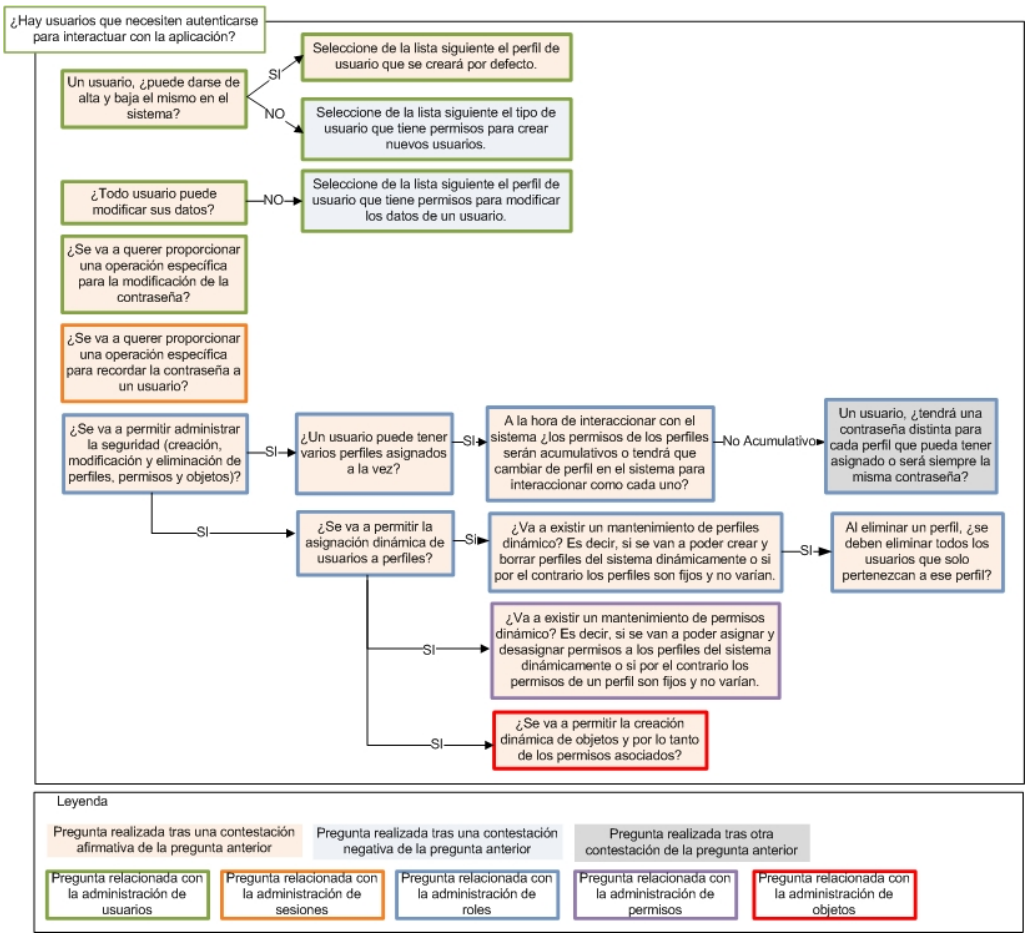

Figura 4–2: Asistente para el grupo funcional *Gestión de Usuarios*

realiza tras contestar afirmativamente la pregunta anterior (rectángulos con fondo en naranja claro), si la pregunta se realiza tras contestar negativamente la pregunta anterior (rectángulos con fondo azul claro) o si la respuesta a la pregunta anterior era otra (rectángulos con fondo de color gris). El color del rectángulo también da información importante para su comprensión: si es de color naranja indica que es una pregunta relacionada con la identificación de usuarios, si es de color verde indica que es una pregunta relacionada con la administración de usuarios y si es de color azul esta relacionada la administración de roles y permisos.

1. La primera pregunta que debe efectuarse, aunque básica, tiene como objetivo detectar si las operaciones de identificación de usuarios en el sistema son necesarias.

- Pregunta: ¿Hay usuarios que necesiten autenticarse para interactuar con la aplicación?
- Respuesta:
  - Si
  - No
- Consecuencia de la respuesta:
  - Si la respuesta es *Si*, se incluyen las operaciones de `IniciarSesion`, `CerrarSesion` y `RecordarContraseña` en el sistema. La operación `IniciarSesion` devuelve un `hash` (llamado `sesión`) que permite identificar la sesión y el rol o perfil del usuario. Todos las operaciones incluirán como parámetro de entrada la `sesión`. Además se seguirán realizando las siguiente preguntas.
  - Si la respuesta es *No*, no se incluirán las operaciones anteriores, el resto de operaciones no incluirán el parámetro `sesión`, ni se continuarán haciendo las preguntas de este grupo funcional
- Respuesta por defecto: La respuesta a esta pregunta se puede deducir fácilmente de las primeras reuniones con el usuario y será determinante para ver si se pasa este cuestionario o no.

2. Para conocer si el sistema va a permitir que sea el propio usuario quien se de da alta o por si el contrario solo lo va a poder realizar el administrador del sistema.
   - Pregunta: Un usuario, ¿puede darse de alta y baja el mismo en el sistema?
   - Respuesta:
     - Si
     - No
   - Consecuencia de la respuesta:

- Si la respuesta es *Si*, la operación `CrearUsuario` no necesita el parámetro de entrada *sesión* y `EliminarUsuario` dará de baja al usuario que la llama. Además se pasa a la pregunta $2a$.
- Si la respuesta es *No*, Se pasa a la pregunta $2b$.

- Respuesta por defecto: Si
- Depende de las siguientes preguntas:
  - Pregunta 1, si la respuesta ha sido *Si*

a) En el caso de los usuarios que pueden darse de alta ellos mismo y que el sistema tenga varios roles, se elegirá el rol con el que se crea el usuario por defecto.

- Pregunta: Seleccione de la lista siguiente el perfil de usuario que se creará por defecto.
- Respuesta: Listado con los perfiles de usuario identificados en el sistema.
- Consecuencia de la respuesta: Será el perfil de usuario con el que se creará un nuevo usuario si no se indica otro perfil distinto.
- Respuesta por defecto: No tiene respuesta por defecto.
- Depende de las siguientes preguntas:
  - Pregunta 1, si la respuesta ha sido *Si*
  - Pregunta 2, si la respuesta ha sido *Si*

b) En el caso de los usuarios que no pueden darse de alta ellos mismo en el sistema, se elegirá el rol que tiene permisos para crear nuevos usuarios.

- Pregunta: Seleccione de la lista siguiente el tipo de usuario que tiene permisos para crear nuevos usuarios.
- Respuesta: Listado con los perfiles de usuario identificados en el sistema.
- Consecuencia de la respuesta: Será el rol con permiso para crear nuevos usuarios.

- Respuesta por defecto: No tiene respuesta por defecto.
- Depende de las siguientes preguntas:
  - Pregunta 1, si la respuesta ha sido *Si*
  - Pregunta 2, si la respuesta ha sido *No*

3. La siguiente pregunta a realizar sirve para detectar si un usuario puede modificar sus datos.
   - Pregunta: ¿Todo usuario puede modificar sus datos?
   - Respuesta:
     - Si
     - No
   - Consecuencia de la respuesta:
     - Si la respuesta es *Si*, se crea la operación `ModificarUsuario`.
     - Si la respuesta es *No*, se realiza la pregunta $3a$.
   - Respuesta por defecto: Si
   - Depende de las siguientes preguntas:
     - Pregunta 1, si la respuesta ha sido *Si*

   a) La siguiente pregunta a realizar sirve para detectar quien puede modificar los datos de un usuario.
      - Pregunta: Seleccione de la lista siguiente el perfil de usuario que tiene permisos para modificar los datos de un usuario.
      - Respuesta: Listado con los roles de usuario identificados en el sistema. Además, como primer ítem de la lista aparecerá la opción *Ninguno*.
      - Consecuencia de la respuesta: Será el rol de usuario que tendrá permiso para modificar los datos de un usuario.
      - Respuesta por defecto: Ninguno.
      - Depende de las siguientes preguntas:

- Pregunta 1, si la respuesta ha sido *Si*
- Pregunta 3,si la respuesta ha sido *No*

4. La siguiente pregunta a realizar sirve para detectar si un usuario puede cambiar su contraseña.
   - Pregunta: ¿Se va a querer proporcionar una operación específica para la modificación de la contraseña?
   - Respuesta:
     - Si
     - No
   - Consecuencia de la respuesta:
     - Si la respuesta es *Si*, se crea la operación `ModificarContraseña`.
     - Si la respuesta es *No*, ninguna.
   - Respuesta por defecto: Si.
   - Depende de las siguientes preguntas:
     - Pregunta 1, si la respuesta ha sido *Si*

5. La siguiente pregunta a realizar sirve para identificar si se manda un correo al usuario para recordarle su contraseña en caso de solicitarlo.
   - Pregunta: ¿Se va a querer proporcionar una operación específica para recordar la contraseña a un usuario?
   - Respuesta:
     - Si
     - No
   - Consecuencia de la respuesta:
     - Si la respuesta es *Si*, se crea la operación `RecordarContraseña`.
     - Si la respuesta es *No*, ninguna.
   - Respuesta por defecto: Si.

- Depende de las siguientes preguntas:
  - Pregunta 1, si la respuesta ha sido *Si*

6. Existen sistemas donde se permite la administración de roles y permisos. Esta pregunta permite identificar si se va a necesitar ofrecer esta funcionalidad.
   - Pregunta: ¿Se va a permitir administrar la seguridad (creación, modificación y eliminación de perfiles y permisos)?
   - Respuesta:
     - Si
     - No
   - Consecuencia de la respuesta:
     - Si la respuesta es *Si*, se publican las operaciones de administración de usuarios determinadas por la realización de las preguntas $6a$, $6b$ y sus sucesivas.
     - Si la respuesta es *No*, no se publicarán las operaciones de administración de roles.
   - Respuesta por defecto: Si
   - Depende de las siguientes preguntas:
     - Pregunta 1, si la respuesta ha sido *Si*

   a) Existen aplicaciones donde un mismo usuario puede tener asignados varios roles a la vez, lo que implica tener que tomar una serie de decisiones.
      - Pregunta: ¿Un usuario puede tener varios perfiles asignados a la vez?
      - Respuesta:
        - Si
        - No
      - Consecuencia de la respuesta:
        - Si la respuesta es *Si*, se realiza la pregunta $6a1$.

- Si la respuesta es *No*, solo se permitirá la asignación de un rol a cada usuario.

- Respuesta por defecto: Si
- Depende de las siguientes preguntas:
  - Pregunta 1, si la respuesta ha sido *Si*
  - Pregunta 6, si la respuesta ha sido *Si*

1) Si un usuario puede tener asignados a varios roles a la vez, implica de nuevo tener que tomar una serie de decisiones.
   - Pregunta: A la hora de interaccionar con el sistema ¿los permisos de los perfiles serán acumulativos o tendrá que cambiar de perfil en el sistema para interaccionar como cada uno?
   - Respuesta:
     - Acumulativos
     - No acumulativos
   - Consecuencia de la respuesta:
     - Si la respuesta es *Acumulativos*, se incluirán todos los roles de un usuario en el parámetro de `sesión`.
     - Si la respuesta es *No acumulativos*, solo se se incluirá el rol con el que el usuario haga el `IniciarSesion` en el parámetro de `sesión`. Además se procede a realizar la pregunta $6a1a$.
   - Respuesta por defecto: Si
   - Depende de las siguientes preguntas:
     - Pregunta 1, si la respuesta ha sido *Si*
     - Pregunta $6a$, si la respuesta ha sido *Si*

a' Si los roles de un usuario no van a ser acumulativos, es interesante saber si tendrán distintas contraseñas para poder llevar a cabo ciertas acciones o no.

- Pregunta: Un usuario, ¿tendrá una contraseña distinta para cada perfil que pueda tener asignado o será siempre la misma contraseña?
- Respuesta:
  - Distinta
  - Misma
- Consecuencia de la respuesta:
  - Si la respuesta es *Distinta*, al llevar a cabo la operación `Iniciar-Sesion`, no hace falta incluir el perfil de usuario porque se puede deducir de la contraseña utilizada.
  - Si la respuesta es *Misma*, al llevar a cabo la operación `Iniciar-Sesion`, hace falta incluir el tipo de usuario porque al tener la misma contraseña para todos los roles no se puede deducir de la contraseña utilizada.
- Respuesta por defecto: Distinta
- Depende de las siguientes preguntas:
  - Pregunta 1, si la respuesta ha sido *Si*
  - Pregunta $6a1a$, si la respuesta ha sido *No Acumulativo*

b)

- Pregunta: ¿Se va a permitir la asignación dinámica de usuarios a perfiles?
- Respuesta:
  - Si
  - No
- Consecuencia de la respuesta:

- Si la respuesta es *Si*, aparecen las operaciones `AsignarUsuarioARol` y `DesasignarUsuarioDeRol`. Además se realizan las preguntas $6b1$, $6b2$ y $6b3$.
- Si la respuesta es *No*, no se realiza ninguna pregunta más referente.

- Respuesta por defecto: Si
- Depende de las siguientes preguntas:
  - Pregunta 1, si la respuesta ha sido *Si*
  - Pregunta 6, si la respuesta ha sido *Si*

1) La siguiente pregunta a realizar sirve para configurar el mantenimiento de roles. Si la pregunta anterior ha sido contestada negativamente, esta pregunta no tendría sentido realizarla.
   - Pregunta: ¿Va a existir un mantenimiento de perfiles dinámico? Es decir, si se van a poder crear y borrar perfiles del sistema dinámicamente o si por el contrario los perfiles son fijos y no varían.
   - Respuesta:
     - Dinámico
     - Estático
   - Consecuencia de la respuesta:
     - Si la respuesta es *Dinámico*, las operaciones `CrearRol`, `ModificarRol`, y `BorrarRol` aparecerán como operaciones del sistema. Además se realiza la pregunta $6b1a$.
     - Si la respuesta es *Estático*, no se realiza ninguna acción.
   - Respuesta por defecto: Estático
   - Depende de las siguientes preguntas:
     - Pregunta 1, si la respuesta ha sido *Si*
     - Pregunta $6b$, si la respuesta ha sido *Si*

a' Si se va a realizar el mantenimiento dinámico de roles, implica tener que tomar una serie de decisiones.

- Pregunta: Al eliminar un perfil, ¿se deben eliminar todos los usuarios que solo pertenezcan a ese perfil?
- Respuesta:
  - Eliminar usuarios
  - Reasignar usuarios en otro perfil
  - No permitir eliminar el perfil
- Consecuencia de la respuesta:
  - Si la respuesta es *Eliminar usuarios*, al eliminar un rol del sistema se borrarán todos los usuarios que solo estén asociados a ese rol.
  - Si la respuesta es *Reasignar usuarios en otro perfil*, al eliminar un rol del sistema permitirá reubicar a los usuarios que solo estén asociados a ese rol a otro rol existente.
  - Si la respuesta es *No permitir eliminar el perfil*, no permitirá borrar el rol si este tiene algún usuario asignado.
- Respuesta por defecto: No permitir eliminar el perfil
- Depende de las siguientes preguntas:
  - Pregunta 1, si la respuesta ha sido *Si*
  - Pregunta $6b1$, si la respuesta ha sido *Si*

2) La siguiente pregunta a realizar sirve para configurar el mantenimiento de permisos. Si la pregunta 6 ha sido contestada negativamente, esta pregunta no tiene sentido realizarla.
   - Pregunta: ¿Va a existir un mantenimiento de permisos dinámico? Es decir, si se van a poder asignar y desasignar permisos a los perfiles

del sistema dinámicamente o si por el contrario los permisos de un perfil son fijos y no varían.

- Respuesta:
  - Dinámico
  - Estático
- Consecuencia de la respuesta:
  - Si la respuesta es *Dinámico*, las operaciones `AsignarPermisosARol` y `DesasignarPermisosDeRol`. No se realiza ninguna pregunta más.
  - Si la respuesta es *Estático*, no se realiza ninguna acción ni se realiza ninguna pregunta más.
- Respuesta por defecto: Dinámico
- Depende de las siguientes preguntas:
  - Pregunta 1, si la respuesta ha sido *Si*
  - Pregunta $6b$, si la respuesta ha sido *Si*

3) La siguiente pregunta a realizar sirve para configurar el mantenimiento de los objetos software susceptibles de tener permisos en el control de acceso (menús de la aplicación, servicios, etc.). Si la pregunta 6 ha sido contestada negativamente, esta pregunta no tiene sentido realizarla.
   - Pregunta: ¿Se va a permitir la creación dinámica de objetos y por lo tanto de los permisos asociados? Es decir, si se van a poder crear y eliminar objetos del sistema dinámicamente o si por el contrario los objetos son fijos y no varían.
   - Respuesta:
     - Dinámico
     - Estático

- Consecuencia de la respuesta:
  - Si la respuesta es *Dinámico*, las operaciones `CrearObjeto`, `ModificarObjeto` y `EliminarObjeto`. No se realiza ninguna pregunta más.
  - Si la respuesta es *Estático*, no se realiza ninguna acción ni se realiza ninguna pregunta más.
- Respuesta por defecto: Dinámico
- Depende de las siguientes preguntas:
  - Pregunta 1, si la respuesta ha sido *Si*
  - Pregunta $6b$, si la respuesta ha sido *Si*

4.3. Extensión de la disciplina de Análisis

Una vez vistas las operaciones necesarias para la gestión de identidad y control de acceso para el proyecto ATICA (ver apartado 3.3), en este apartado se va a completar su especificación tal y como está definido en DI-SFM para la disciplina de análisis.

Como se ha comentado en el apartado 2.1 durante la introducción a la disciplina de análisis de DI-SFM, esta disciplina tiene una serie de tareas:

1. Especificación de Casos de uso
2. Modelado de las clases
3. Diseño de interfaces

A continuación, se va a presentar la especificación de los casos de uso para el control de acceso, en segundo lugar se van a introducir las clases del modelo de seguridad indicando sus atributos, operaciones y relaciones. El diseño de los interfaces se presenta en el anexo A.

4.3.1. Especificación de los casos de uso

Una vez conocidos los requisitos del cliente, se procede a modelar el sistema por los casos de uso. En definitiva, se crea un caso de uso por cada una de las acciones que se puedan llevar a cabo en nuestro sistema (ver figura 4–3).

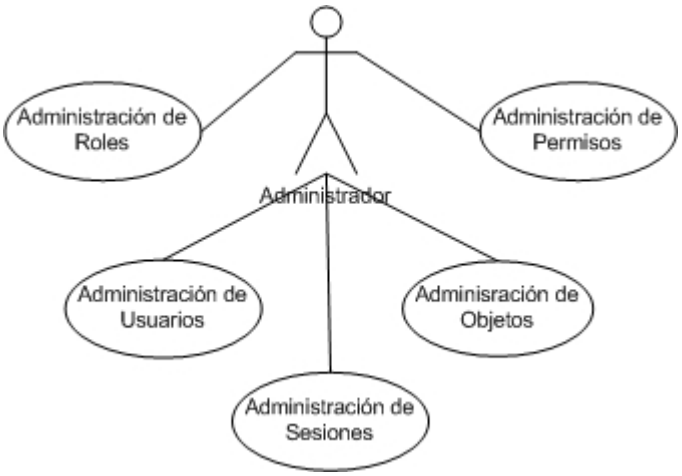

Figura 4–3: Diagrama de Casos de uso para el control de acceso basado en roles

En resumen, las acciones que se pueden llevar a cabo son:

- Administración de Sesiones: Este caso de uso se encarga de crear y eliminar la sesión de un usuario. Adicionalmente, se encarga de recordar la contraseña. Las operaciones que lo implementan son:
  - IniciarSesion
  - CerrarSesion
  - recordarContraseña
- Administración de Usuarios Este caso de uso se encarga de crear y eliminar la sesión de un usuario. Adicionalmente, se encarga de recordar la contraseña. Las operaciones que lo implementan son:
  - CrearUsuario
  - EliminarUsuario
  - ModificarUsuario
  - ModificarContraseña

- Administración de Roles Este caso de uso se encarga de crear y eliminar los roles del sistema. Adicionalmente, se encarga de asignar y desasignar usuarios a los roles además de realizar diversas búsquedas. Las operaciones que lo implementan son:
  - CrearRol
  - EliminarRol
  - AsignarUsuarioARol
  - DesasignarUsuarioDeRol
  - BuscarUsuariosDeRol
  - BuscarRolesDeUsuario
- Administración de Permisos Este caso de uso se encarga de asignar y desasignar permisos a los roles. Adicionalmente, se encarga de realizar diversas búsquedas. Las operaciones que lo implementan son:
  - AsignarPermisosARol
  - DesasignarPermisosDeRol
  - BuscarPermisosDeSesion
  - BuscarPermisosDeRol
- Administración de Objetos Este caso de uso se encarga de crear y eliminar objetos del sistema. Las operaciones que lo implementan son:
  - CrearObjeto
  - EliminarObjeto

4.3.2. Modelado de las clases

Tras la especificación de los casos de uso, el siguiente paso es modelar las clases involucradas (ver figura 4–4), definiendo sus atributos y operaciones. Como se puede apreciar en el modelo de clases, el sistema de seguridad está formado por un

conjunto de perfiles que tienen acceso a objetos software y a servicios. Estos objetos software pueden ser de varios tipos: Pestaña principal, Menú de la izquierda, Pestañas secundarias, Pantalla y botones (para más detalle ver anexo A).

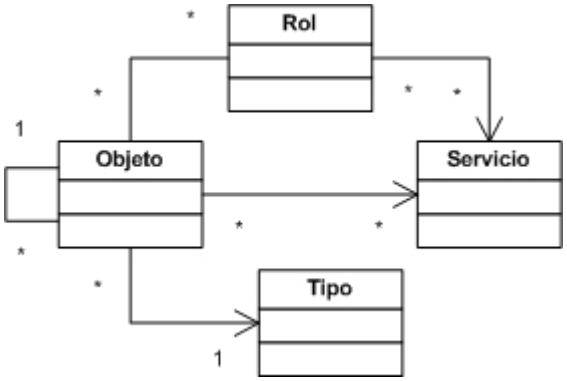

Figura 4–4: Modelo de control de acceso

A continuación se detalla la especificación funcional de cada una de las operaciones del modelo (identificadas de los casos de uso):

Administración de Sesiones

La especificación de las operaciones encargadas de la administración de sesiones son:

- IniciarSesion
  - **Descripción:** esta operación crea una nueva sesión con el usuario y un grupo de roles activos. La operación es válida si y solo si el usuario es miembro del conjunto de `usuarios` y el conjunto de roles activos es un subconjunto de roles asignados al usuario.
  - **Parámetros de entrada:** Datos del inicio de sesión. Estos datos dependerán de la toma de requisitos. Por ejemplo, para el proyecto ATICA los datos de entrada para el inicio de sesión son `usuario` y `contraseña`. La aparición del parámetro `Rol` dependerá de si un usuario puede pertenecer a más de un rol o no y de si cada rol tiene una contraseña igual o distinta. Si cada rol tiene una contraseña

distinta no es necesario especificar el rol porque viene dado para la contraseña del usuario.

- **Parámetros de salida:**Si los datos de entrada se corresponden con un usuario, se crea una sesión en el sistema y se devuelve un hash `sesion` que identifica dicha sesión. Para el proyecto ATICA, este parámetro `sesion` incluye el usuario y su colección de roles por centro.
- **Nota:** Por el artículo 18.2 de la LOPD, en lo que se refiere a la autenticación:

  > "Se limitará la posibilidad de intentar reiteradamente el acceso no autorizado al SI"

  El cumplimiento de esta ley se realiza mediante el bloqueo de cuentas al sistema. Por defecto se bloquea una cuenta cuando transcurridos varios intentos (tres en nuestro caso), en un tiempo determinado, no se haya introducido una contraseña correcta.

  En este caso, la cuenta quedará bloqueada durante un determinado tiempo o hasta que un administrador la desbloquee. La cuenta bloqueada supone la denegación de acceso al sistema aún cuando la contraseña sea correcta.

- CerrarSesion
  - **Descripción:** esta operación elimina la sesión del usuario. La operación es válida si y solo si la sesión es miembro del conjunto de `sesiones`, el usuario existe en el conjunto de `usuarios` y el usuario es el propietario de la sesión.
  - **Parámetros de entrada:** `sesion`.
  - **Parámetros de salida:** 0 si ha ido todo bien o -1 en caso contrario.
- RecordarContraseña
  - **Descripción:** esta operación envía un correo al usuario para recordarle su contraseña.
  - **Parámetros de entrada:** `usuario` y `respuesta`.
  - **Parámetros de salida:** Ninguno.

Administración de Usuarios

La especificación de las operaciones encargadas de la administración de usuarios son:

- CrearUsuario
  - **Descripción:** esta operación crea un nuevo usuario. La operación solo es válida si el nuevo usuario no existe todavía como miembro del conjunto de `usuarios`. El nuevo usuario no puede tener activa ninguna sesión en el momento de su creación.
  - **Parámetros de entrada:** Datos del `usuario`.
  - **Parámetros de salida:** 0 si ha ido todo bien o -1 en caso contrario.
- EliminarUsuario
  - **Descripción:** esta operación elimina un usuario existente de la base de datos del sistema. La operación es válida si y solo si el usuario que va a ser borrado es miembro del conjunto de `usuarios`. Es una decisión de implementación cómo proceder con la sesión del usuario a ser borrado. El sistema puede esperar a que el usuario termine su sesión normalmente o puede forzar su finalización.
  - **Parámetros de entrada:** Identificador del `usuario`.
  - **Parámetros de salida:** 0 si ha ido todo bien o -1 en caso contrario.
- ModificarUsuario
  - **Descripción:** esta operación modifica los datos de un usuario. La operación es válida si y solo si el usuario que va a ser modificado es miembro del conjunto de `usuarios`.
  - **Parámetros de entrada:** Datos del `usuario`.
  - **Parámetros de salida:** 0 si ha ido todo bien o -1 en caso contrario.
- ModificarContraseña
  - **Descripción:** modifica la contraseña de un usuario. La operación es válida si y solo si el usuario que va a ser modificado es miembro del conjunto de `usuarios`.

- **Parámetros de entrada:** Identificador del `usuario`, `contraseña actual` y dos veces la `nueva contraseña`.
- **Parámetros de salida:** 0 si ha ido todo bien o -1 en caso contrario.

Administración de Roles

La especificación de las operaciones encargadas de la administración de roles son:

- CrearRol
  - **Descripción:** esta operación crea un nuevo rol. La operación es válida si y solo si el nuevo rol no existe todavía como miembro del conjunto de `roles`. Inicialmente, el nuevo rol no tendrá ningún usuario ni ningún permiso asignado.
  - **Parámetros de entrada:** Datos del `Rol`
  - **Parámetros de salida:** 0 si ha ido todo bien o -1 en caso contrario.
- EliminarRol
  - **Descripción:** esta operación elimina un rol existente de la base de datos. La operación es válida si y solo si el rol que va a ser borrado es miembro del conjunto de `roles`. Es una decisión de implementación cómo proceder con las sesiones que tienen activo el rol que va a ser borrado. El sistema puede esperar a que la sesión termine normalmente o puede eliminar el rol de las sesiones mientras se permite que las sesiones continúen.
  - **Parámetros de entrada:** Identificador del `Rol`
  - **Parámetros de salida:** 0 si ha ido todo bien o -1 en caso contrario.
- ModificarRol
  - **Descripción:** esta operación modifica los datos del `Rol` indicado. La operación es válida si y solo si el rol que va a ser modificado es miembro del conjunto de `roles`.
  - **Parámetros de entrada:** Datos del `Rol`.
  - **Parámetros de salida:** 0 si ha ido todo bien o -1 en caso contrario.

- AsignarUsuarioARol
  - **Descripción:** esta operación asigna un usuario a un rol. La operación es válida si y solo si el usuario existe en el conjunto de `usuarios`, el rol existe en el conjunto de `roles` y el usuario no ha sido todavía asignado al rol.
  - **Parámetros de entrada:** Identificador del `Rol` e identificador del `usuario`.
  - **Parámetros de salida:** 0 si ha ido todo bien o -1 en caso contrario.
- DesasignarUsuarioDeRol
  - **Descripción:** esta operación elimina la asignación de un usuario de un rol. La operación es válida si y solo si el usuario existe en el conjunto de `usuarios`, el rol existe en el conjunto de `roles` y el usuario ha sido asignado al rol.
  - **Parámetros de entrada:** Identificador del `Rol` e identificador del `usuario`.
  - **Parámetros de salida:** 0 si ha ido todo bien o -1 en caso contrario.
- BuscarUsuariosDeRol
  - **Descripción:** esta operación devuelve el conjunto de usuarios asignados a un rol. La operación es válida si y solo si el rol es miembro del conjunto de `roles`.
  - **Parámetros de entrada:** Identificador del `Rol`
  - **Parámetros de salida:** Conjunto de `usuarios` asignados a un `rol`.
- BuscarRolesDeUsuario
  - **Descripción:** esta operación devuelve el conjunto de roles asignados a un usuario. La operación es válida si y solo si el usuario es miembro del conjunto de `usuarios`.
  - **Parámetros de entrada:** Identificador del `usuario`.
  - **Parámetros de salida:** Conjunto de `roles` asignados a un `usuario`.

Administración de Permisos

La especificación de las operaciones encargadas de la administración de permisos son:

- AsignarPermisosARol

- **Descripción:** esta operación asigna a un rol el permiso sobre un objeto. La operación es válida si y solo si el objeto es susceptible de soportar un permiso, y el rol es un miembro del conjunto de `roles`.
- **Parámetros de entrada:** Identificador del `rol` e identificador del `objeto`
- **Parámetros de salida:** 0 si ha ido todo bien o -1 en caso contrario.

- DesasignarPermisosDeRol
  - **Descripción:** esta operación elimina permisos a cada uno de los roles para ejecutar las operaciones que se consideren necesarias. Esta operación en principio sólo la podrá realizar el administrador del sistema. esta operación cancela el permiso de llevar a cabo una operación sobre un objeto del conjunto de permisos asignados a un rol. La operación es válida si y solo si el par (operación, rol) representa un permiso, el rol existe en el conjunto de `usuarios` y el permiso está asignado al rol.
  - **Parámetros de entrada:** Identificador del `rol` e identificador del `objeto`
  - **Parámetros de salida:** 0 si ha ido todo bien o -1 en caso contrario.
- BuscarPermisosDeSesion
  - **Descripción:** esta operación devuelve el conjunto de permisos de una sesión, es decir, los permisos asignados a los roles activos de dicha sesión. La operación es válida si y solo si la sesión es miembro del conjunto de `sesiones`.
  - **Parámetros de entrada:** Identificador del `Rol` e identificador del `Objeto`
  - **Parámetros de salida:** 0 si ha ido todo bien o -1 en caso contrario.
- BuscarPermisosDeRol
  - **Descripción:** esta operación devuelve el conjunto de permisos sobre objetos de un rol. La operación es válida si y solo si el rol es miembro del conjunto de `roles`.
  - **Parámetros de entrada:** Identificador del `rol`
  - **Parámetros de salida:** Conjunto de `objetos` sobre los que tiene permiso un `rol`.

- Administración de Objetos La especificación de las operaciones encargadas de la administración de objetos son:
  - CrearObjeto
    - **Descripción:** esta operación crea un nuevo objeto. La operación es válida si y solo si el nuevo objeto no existe todavía como miembro del conjunto de `objetos`. Inicialmente, el nuevo objeto no tendrá ningún permiso asignado.
    - **Parámetros de entrada:** Datos del `Objeto`
    - **Parámetros de salida:** 0 si ha ido todo bien o -1 en caso contrario.
  - ModificarObjeto
    - **Descripción:** esta operación modifica los datos del `objeto` indicado. La operación es válida si y solo si el objeto que va a ser modificado es miembro del conjunto de `objetos`.
    - **Parámetros de entrada:** Datos del `Objeto`
    - **Parámetros de salida:** 0 si ha ido todo bien o -1 en caso contrario.
  - EliminarObjeto
    - **Descripción:** esta operación elimina un objeto existente de la base de datos. La operación es válida si y solo si el objeto que va a ser borrado es miembro del conjunto de `objetos`. Es una decisión de implementación cómo proceder con las sesiones que tienen activo el obejto que va a ser borrado. El sistema puede esperar a que la sesión termine normalmente o puede eliminar el objeto de las sesiones mientras se permite que las sesiones continúen.
    - **Parámetros de entrada:** Identificador del `Objeto`
    - **Parámetros de salida:** 0 si ha ido todo bien o -1 en caso contrario.

Una representación gráfica de las operaciones que se generan descritas anteriormente se puede ver en la figura 4–5.

```
<wsdl:portType name="ControlAccesoSoap">
    ...
    <wsdl:operation name="IniciarSesion">
    <wsdl:operation name="CerrarSesion">
    <wsdl:operation name="RecordarContraseña">
    <wsdl:operation name="CrearUsuario">
    <wsdl:operation name="EliminarUsuario">
    <wsdl:operation name="Modificar">
    <wsdl:operation name="ModificarContraseña">
    <wsdl:operation name="CrearRol">
    <wsdl:operation name="EliminarRol">
    <wsdl:operation name="Modificar">
    <wsdl:operation name="AsignarUsuarioARol">
    <wsdl:operation name="DesasignarUsuarioDeRol">
    <wsdl:operation name="BuscarUsuariosDeRol">
    <wsdl:operation name="BuscarRolesDeUsuario">
    <wsdl:operation name="CrearObjetos">
    <wsdl:operation name="EliminarObjetos">
    <wsdl:operation name="ModificarObjetos">
    <wsdl:operation name="AsignarPermisoARol">
    <wsdl:operation name="DesasignarPermisosDeRol">
    <wsdl:operation name="BuscarPermisosDeRol">
    <wsdl:operation name="BuscarPermisosDeUsuario">
    <wsdl:operation name="BuscarPermisosDeSesion">
    ...
</wsdl:portType>
```

Figura 4–5: Operaciones para el control de acceso

4.4. Conclusiones

En este capítulo se han presentado dos extensiones para la metodología DI-SFM. La primera de las extensiones consiste en un asistente para guiar la toma de requisitos en relación al control de acceso al sistema. La segunda, presenta una especificación funcional para la disciplina de análisis. Estas extensiones permiten representar y generalizar los conceptos introducidos en el control de acceso basado en roles.

Se ha presentado un conjunto de preguntas que permiten al analista capturar las necesidades del cliente sobre el control de acceso durante la toma de requisitos y no durante la implementación del sistema.

Para completar la propuesta, se ha presentado la especificación funcional de las operaciones que se identifican automáticamente mediante el cuestionario rellenado

por el cliente. Para cada una de las operaciones se ha detallado su descripción, parámetros de entra y parámetros de salida.

Capítulo 5
CONCLUSIONES Y TRABAJOS FUTUROS

En este trabajo del *Máster en Ingeniería del Software, Métodos Formales y Sistemas de Información* se ha presentado cómo se está abordando el control de acceso en el proyecto que lleva la UTE Dimensión Informática con Telefónica para la Conselleria de Educación. La solución propuesta en este proyecto pretende ser genérica para ser usada en los siguientes proyectos que se realicen en la empresa.

Esta propuesta introduce dos nuevas extensiones en la metodología DI-SFM (metodología que siguen los proyectos realizados en Dimensión Informática). La primera de las extensiones se introduce en la disciplina de Requisitos. Esta primera extensión consta de un conjunto de preguntas a realizar al cliente durante la toma de requisitos para recoger sus necesidades en el ámbito del control de acceso basado en roles. La segunda de las extensiones se introduce en la disciplina de Análisis. Esta segunda extensión consta de la definición automática de las operaciones identificadas de la toma de requisitos.

La adopción de la propuesta junto con un sistema integrado y automatizado de gestión de identidades permite disfrutar de importantes mejoras de la eficacia, seguridad y flujo de procesos asociados con la autenticación y autorización de usuarios.

En el caso de la Conselleria de Educación, posibilitará que la gestión de usuarios y sus derechos de acceso puedan manejarse de forma transparente en plataformas y sistemas heterogéneos, asegurando al tiempo el cumplimiento de las directivas y normativas de seguridad que sean de aplicación, y proporcionando las pruebas de auditoria necesarias para certificarlo.

El trabajo desarrollado tiene continuidad y puede ampliarse mediante:

- La realización de una herramienta asistente (ver figura 5–1) para generar automáticamente la definición de los casos de uso y de las operaciones para ser reutilizadas en la disciplina de modelado.

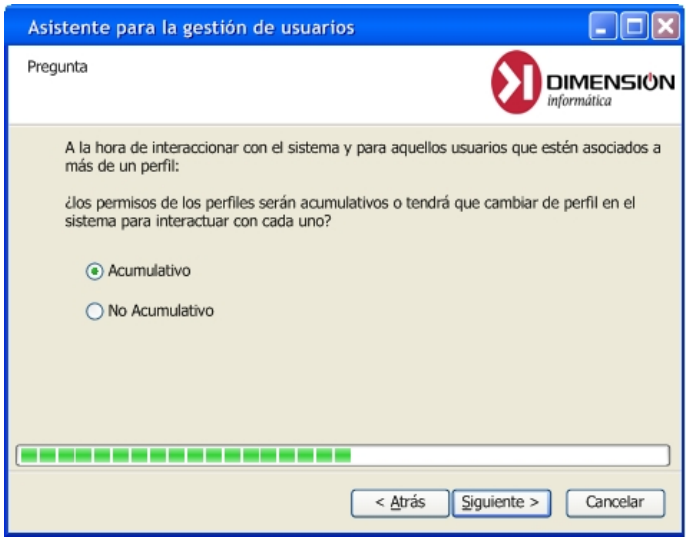

Figura 5–1: Pantalla interior del asistente

- La definición de la implementación de las operaciones, dependiendo de las respuestas de los cliente a las preguntas realizadas en la toma de requisitos, para poder incluirlo en en el generador.
- La depuración y/o extensión del asistente, refinando las preguntas existentes o introduciendo nuevas.
- La incorporación de los servicios de RBAC para así ampliar la gama de permisos ofertada.

Por último, es importante recordar que, en la actualidad, el trabajo desarrollado en la tesis del máster está siendo incorporado en un sistema real en la empresa Dimensión Informática. Conforme se vaya avanzando en el proyecto, la propuesta irá avanzando con él sufriendo modificaciones y extensiones que la mejorarán.

APENDICES

Apéndice A
MÓDULO DE SEGURIDAD EN ATICA

En este anexo se presenta el módulo de seguridad para el sistema de la Conselleria de Educación (sólo la parte correspondiente a Dimensión Informática). A este módulo de seguridad se accede desde la pestaña Seguridad del menú principal y sólo aquellos roles que tengan permisos.

Las funcionalidades que se van a poder administrar desde el módulo de seguridad son las propuestas en el apartado 3.4 y están organizadas en los siguientes apartados (y cada uno de ellos ubicado en su correspondiente entrada de menú):

- Administración de Perfiles
- Administración de Objetos
- Administración de Permisos

La estructura del anexo consta de cuatro apartados, en primer lugar se introduce el módulo de seguridad. En segundo lugar se presenta la administración de Perfiles de usuario, seguido por la administración de objetos y finalizando con la administración de permisos.

A.1. Administración de Objetos

Los permisos pueden concederse o revocarse sobre los objetos susceptibles de tener control de acceso (ver figura A–1). Para cada objeto se define:

- Código del objeto: se utiliza el código que tiene el objeto en la implementación.
- Nombre: nombre del objeto.
- Descripción: breve descripción del objeto.

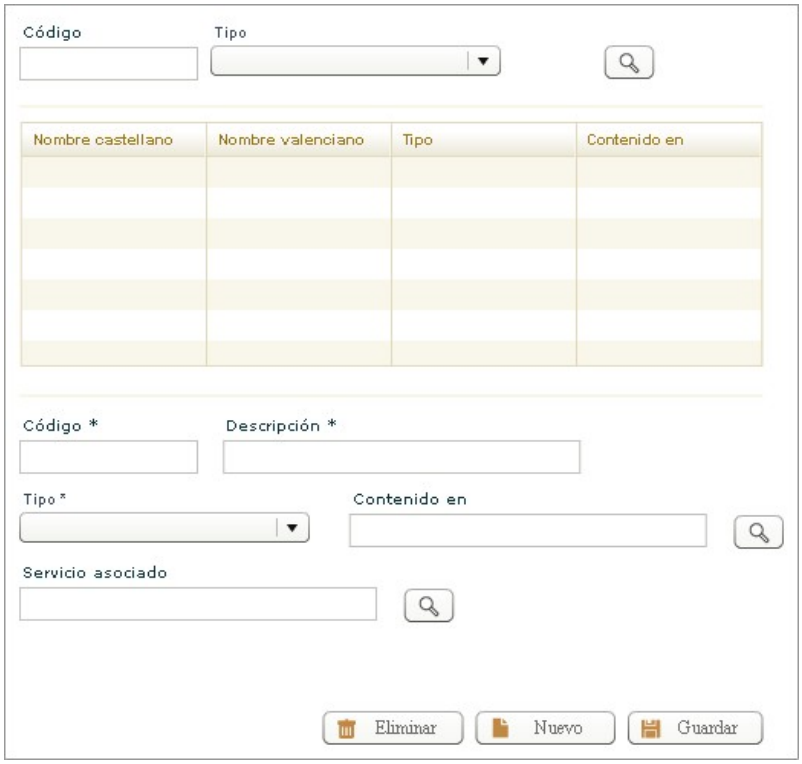

Figura A–1: Pantalla para la administración de objetos

- Tipo: indica el tipo de objeto (ver figura A–2).
  - Pestaña principal (indicada en la figura con un 1 y en rojo)
  - Menú de la izquierda (indicada en la figura con un 2 y en verde)
  - Pestañas secundarias (indicada en la figura con un 3 y en azul)
  - Pantalla (indicada en la figura con un 4 y en morado)
  - Botones (indicada en la figura con un 5 y en negro)
- Contenido en: indica el objeto que lo contiene (por ejemplo, si es un objeto menú de la izquierda aquí se indica en qué pestaña está incluido)
- Servicios: si está relacionado con algún servicio de la implementación.

A.2. Administración de Perfiles

Un perfil de usuario es una forma de organizar los permisos sobre las diferentes funcionalidades que se pueden llevar a cabo en la aplicación. De forma que el perfil

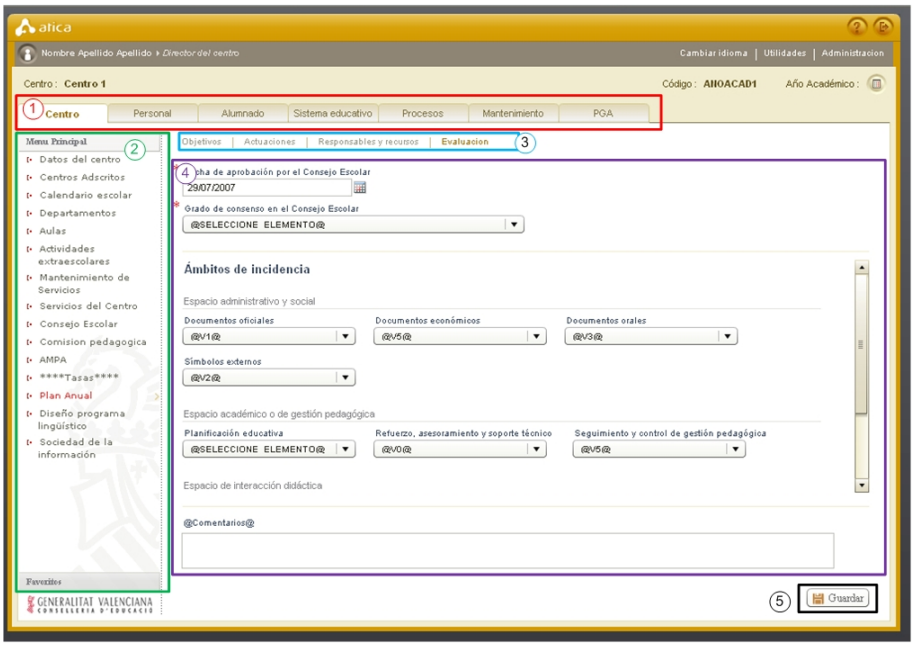

Figura A–2: Pantalla de ATICA donde se distinguen los distintos tipos de objetos

de usuario *X* puede acceder a las funcionalidades *Z, Y, V, ...* Por otro lado a los perfiles se les asocian usuarios, de esta manera sabemos a qué funcionalidad puede acceder un usuario.

Para acceder a las funcionalidades asociadas a un perfil de usuario se selecciona la entrada de menú Perfiles. Para cada perfil se define únicamente el nombre del perfil en castellano y en valenciano (por ser una aplicación bilingüe).

Una vez vistos los atributos de los perfiles, a continuación se presentan las operaciones implementadas en ATICA:

- Dar de alta un perfil
- Modificar un perfil
- Dar de baja un perfil
- Consultar los perfiles del sistema: todos, por nombre ...
- Asociar/Desasociar permisos al perfil

En figura A–3 se puede ver la pantalla de administración básica de perfiles. Desde la pantalla se permite buscar, dar de alta, dar de baja y modificar los datos de

un perfil. Una vez accionada la búsqueda, se rellena el grid con los resultados y seleccionando uno se puede modificar o eliminar el perfil.

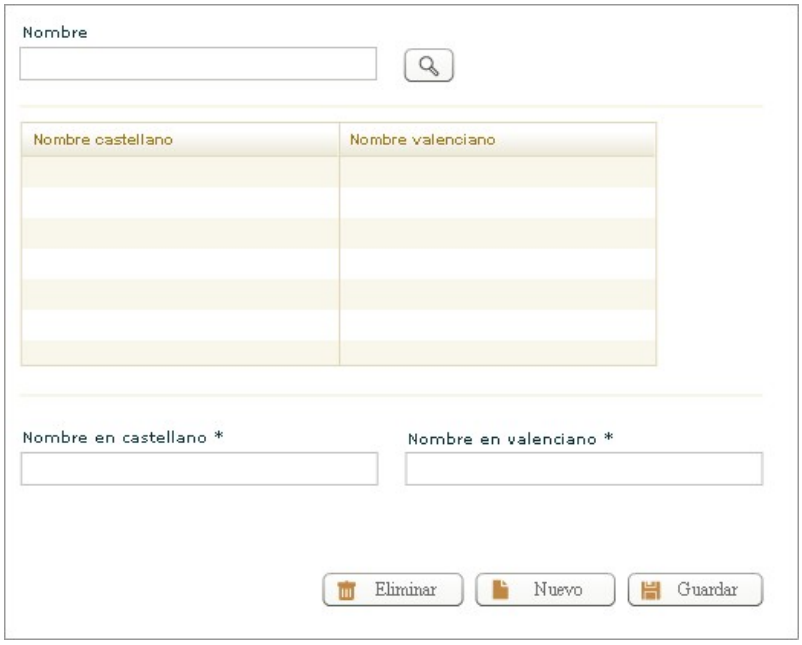

Figura A–3: Pantalla para la administración de perfiles

A continuación se muestran los perfiles existente en ATICA (ver figura A–4), que están divididos entre aquellos perfiles que se van a conectar desde un centro, desde servicios centrales y desde servicios territoriales.

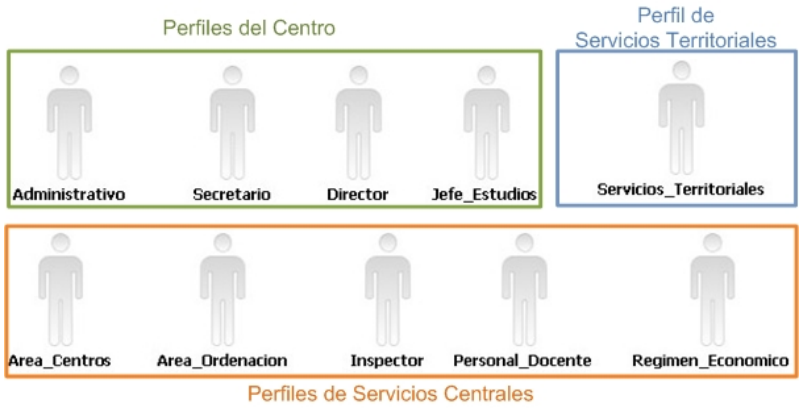

Figura A–4: Perfiles de ATICA

A.3. Administración de Permisos

Los permisos se utilizan para describir un conjunto de posibles acciones a realizar por el usuario o componentes accesibles. Estas acciones y componentes permiten la configuración de permisos a los perfiles. Se puede acceder a la administración de los permisos a través de la entrada de menú Permisos. Desde este menú se pueden activar o desactivar funcionalidades. El objetos que esté activado será accesible por los usuarios del perfil y los que no estén activados a los usuarios del perfil ni les aparecerá en pantalla.

En figura A–5 se puede ver la pantalla de administración básica de los permisos. Desde la pantalla se permite indicar para un perfil, sobre qué objetos del sistema tiene permiso.

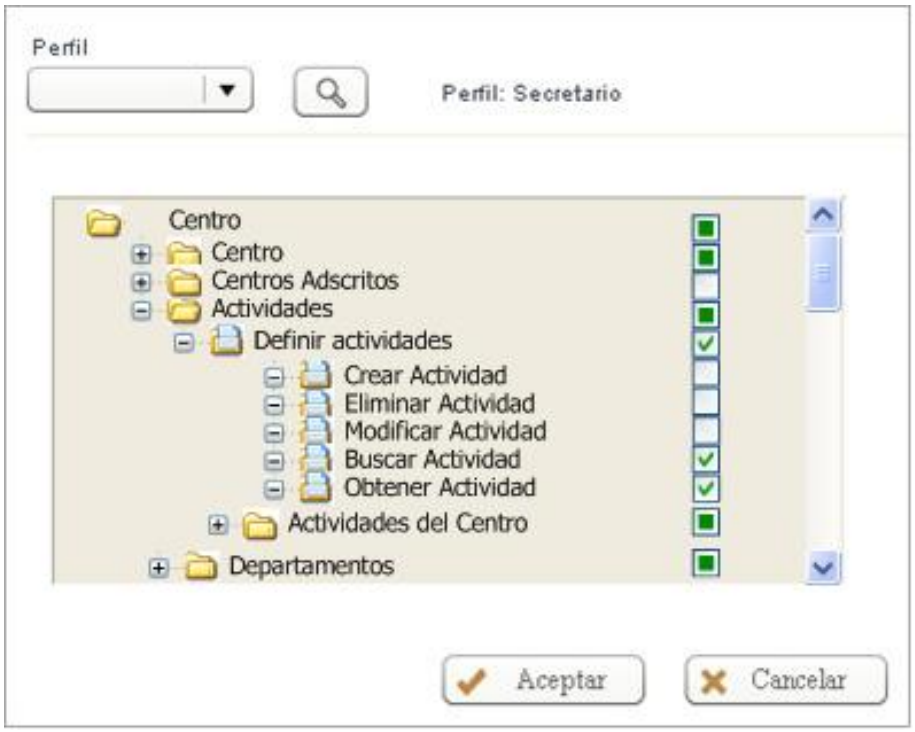

Figura A–5: Pantalla para la administración de permisos

A.4. Conclusiones

En este apéndice se ha presentado el diseño de las interfaces para la administración de perfiles, objetos y permisos. Se han mostrado los elementos que las forman y algunas de las páginas que forma el proyecto.

Bibliografía

[1] Marta Ruiz, Pedro Valderas, Victoria Torres, and Vicente Pelechano. A model driven approach to design web services in a web engineering method. In *CAiSE Short Paper Proceedings*, 2005.

[2] Marta Ruiz, Pedro Valderas, and Vicente Pelechano. Applying a web engineering method to design web services. In *ICSOC*, pages 576–581, 2005.

[3] Marta Ruiz and Vicente Pelechano. Model driven design of web services operations using web engineering practices. In *Emerging Web Services Technology*, 2007.

[4] David F. Ferraiol and D. Richard Kuhn. Role-based access controls. *15th NIST-NCSC National Compuer Security Conference*, pages 554–563, Oct 1992.

[5] Ravi S. Sandhu, Edward J. Coyne, Hal L. Feinstein, and Charles E. Youman. Role-based access control models. *IEEE Computer*, 29(2):38–47, 1996.

[6] David F. Ferraiolo, Ravi Sandhu, Serban Gavrila, D. Richard Kuhn, and Ramaswamy Chandramouli. Proposed nist standard for role-based access control. *ACM Trans. Inf. Syst. Secur.*, 4(3):224–274, 2001.

[7] Ninghui Li and Ziqing Mao. Administration in role-based access control. In *ASIACCS '07: Proceedings of the 2nd ACM symposium on Information, computer and communications security*, pages 127–138, New York, NY, USA, 2007. ACM Press.

[8] Joon S. Park, Ravi Sandhu, and Gail-Joon Ahn. Rbac on the web. *ACM Transaction on Information and Systems Security*, 4(1), Feb. 2001.

[9] Pierangela Samarati, Elisa Bertino, and Sushil Jajodia. An authorization model for a distributed hypertext system. *IEEE Transactions on Knowledge and Data*

Engineering, 8(4):555–562, 1996.

[10] N. R. Adam, V. Atluri, E. Bertino, and E. Ferrari. A content-based authorization model for digital libraries. *IEEE Transactions on Knowledge and Data Engineering*, 14(2):296–315, 2002.

[11] Weigang Wang. Team-and-role-based organizational context and access control for cooperative hypermedia environments. In *HYPERTEXT '99: Proceedings of the tenth ACM Conference on Hypertext and hypermedia : returning to our diverse roots*, pages 37–46, New York, NY, USA, 1999. ACM Press.

[12] American National Standards Institute (ANSI). Inc. role-based access control. *ANSI INCITS 359–2004*, 2004.

[13] Rafae Bhatti, Elisa Bertino, Arif Ghafoor, and James B. D. Joshi. Xml-based specification for web services document security. *Computer*, 37(4):41–49, 2004.

[14] Torsten Lodderstedt, David Basin, and Jürgen Doser. Secureuml: A uml-based modeling language for model-driven security. In *UML '02: Proceedings of the 5th International Conference on The Unified Modeling Language*, pages 426–441, London, UK, 2002. Springer-Verlag.

[15] Daniel Sanz, Paloma Díaz, and Ignacio Aedo. Towards integration of access control in the hypermedia development process. *Sistedes, JISBD*, 2007.

[16] Ausiàs Armesto, Grzegorz Loniewski, Auxiliadora Carlos, and Vicent Llorens. Incorporando el análisis estático de código en un proceso de integración contínúa. JTS 2007: Jornadas sobre el Testeo de Software, 2007.

[17] Ausiàs Armesto and Damián Vidal. Implantación de un proceso de testeo en una factoría de software. III Jornadas de Testeo de Software, 2006.

[18] Ivar Jacobson, Grady Booch, and James Rumbaugh. *The unified software development process*. Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA, 1999.

[19] METRICA version 3.0. Metodología de planificación, desarrollo y mantenimiento de sistemas de información. Technical report, Ministerio de Administraciones Públicas, España, 200.

[20] Timoth Howes, Mark Smith, and Gordon Good. *Understanding and Deploying LDAP Directory Services*. Macmillan Technical Publishing, 2003.

[21] Steven Tuttle et al. *Understanding LDAP design and implementation*. White Plains, NY : IBM International Technical Support Organization, c2004, 2004.

[22] Joon S. Park, Gail-Joon Ahn, and Ravi Sandhu. Role-based access control on the web using ldap. In *Das'01: Proceedings of the fifteenth annual working conference on Database and application security*, pages 19–30, Norwell, MA, USA, 2002. Kluwer Academic Publishers.