Proyecto Fin de Carrera

Ingeniería Informática

Universidad Carlos III de Madrid

Diseño e implementación del Sistema de Políticas de Privacidad para el proyecto CERTILOC

Alumno:	D. John Pater Martínez de Leiva
Tutor:	Dra. Dña. Ana Isabel González-Tablas Ferreres
Directores:	Dra. Dña. Ana Isabel González-Tablas Ferreres
	D. José María de Fuentes García-Romero de Tejada
Fecha:	Junio de 2009

Título: Diseño e implementación del "Sistema de Políticas de Privacidad para el proyecto CERTILOC

Autor: D. John Pater Martínez de Leiva

Tutor: Dra. Dña. Ana Isabel González-Tablas Ferreres

Directores: Dra. Dña. Ana Isabel González-Tablas Ferreres

D. José María de Fuentes García-Romero de Tejada

La defensa del presente Proyecto de Fin de Carrera se realizó el día 26 de Junio del 2009; siendo calificada por el siguiente Tribunal:

Presidente: Arturo Ribagorda Garnacho

Secretario: Agustín Orfila Díaz-Pabón

Vocal: Andrés Marín López

Habiendo obtenido la siguiente calificación:

Calificación

Presidente Secretario Vocal

Agradecimientos

Antes de comenzar la descripción detallada del actual Proyecto de Fin de Carrera, me gustaría dedicar unas líneas de agradecimiento a toda la gente que me ha ayudado y apoyado en su realización.

En primer lugar agradecer el tiempo e interés mostrado por los lectores del actual documento.

En segundo lugar me gustaría agradecer, con especial cariño y admiración, a mi familia. En particular a mi mujer y futura madre de mi hijo (nacerá en pocos meses) por todo el apoyo, por su comprensión y su cariño. Ha comprendido, en innumerables ocasiones, mi ausencia por estar desarrollando el actual proyecto. También a mi padre y mis hermanos por su apoyo, sin ellos nada de esto tendría sentido. Incluyo aquí a mis amigos y allegados, gracias a todos por estar ahí cada día y por vuestro apoyo.

Me gustaría hacer una mención especial para mi tutora del proyecto de fin de carrera, Anabel, por su comprensión apoyo y tiempo. Por haber comprendido que la planificación inicial no pudo ser cumplida y por no haber desistido con el proyecto. Por todo esto, y con todo mi cariño, te doy las gracias Anabel.

Agradecer también a Jose María (Chema) por haberme echado una mano para concluir el último estadio del proyecto, por haber sustituido a Anabel para la presentación de mi proyecto y por haber mostrado un apoyo incondicional y enérgico. Gracias Chema, eres una gran persona y te deseo lo mejor.

Por último, un agradecimiento general a la Universidad Carlos III de Madrid por haberme ofrecido la oportunidad de completar y finalizar mis estudios en Ingeniería Informática. En mi caso particular, el primer ciclo de la carrera (Ingeniería Técnica en Informática de Sistemas) fue realizado en la Universidad Complutense de Madrid. Tras un año de trabajo, me decidí a completar mi formación, con el segundo ciclo, en la Universidad Carlos III. La decisión para el cambio de universidad no fue casual, ni por necesidad sino que me fue recomendada de primera mano y con mucho acierto. La cantidad de conocimientos adquiridos en este segundo ciclo ha sido bastante amplia y no han sido únicamente abordados de manera teórica, sino que ha existido un componente práctico bastante amplio que me ha ayudado a completar los conocimientos técnicos y me están ayudando actualmente en el desarrollo de mi trabajo diario.

Antes de concluir, me gustaría hacer una mención especial a todos los trabajadores de Anasoft Solutions (empresa creada hace un año y medio, en la cual llevo la gerencia) y en particular a Sergio Jiménez. Agradeceros vuestra comprensión, vuestro esfuerzo y vuestro apoyo.

A mi futuro hijo, Nicolás, y a su madre, Ainhoa.

"Si se siembra la semilla con fe y se riega con perseverancia, sólo será cuestión de tiempo recoger sus frutos"

Thomas Carlyle (1795 – 1881).

Resumen

El presente documento describe el procedimiento seguido para la realización del Proyecto de Fin de Carrera, desarrollado por el alumno John Pater Martínez de Leiva, para la obtención del título de Ingeniería Informática.

El proyecto desarrollado se engloba dentro del contexto del proyecto de investigación CERTILOC. Dicho proyecto tiene como objetivo general el proporcionar un marco de seguridad y legalidad para varios métodos y tecnologías de localización espacio-temporal existentes hoy en día.

Particularmente, los módulos desarrollados en el proyecto que presentamos a continuación, aportan a CERTILOC un sistema de políticas de privacidad. Gracias a este sistema, los usuarios de CERTILOC pueden definir políticas que rigen la autorización del acceso a su información personal. Dicha información, suele estar relacionada con los dispositivos localizables, de los que el usuario es responsable.

El sistema de políticas de privacidad tiene 4 objetivos básicos:

- Evaluar si se debe permitir o denegar una determinada petición de autorización ante las políticas de privacidad activas en el sistema.
- Ofrecer un repositorio persistente para las políticas de privacidad definidas por los usuarios.
- Proporcionar a los usuarios una manera de gestionar sus políticas de privacidad.
- Permitir hacer un seguimiento de la actividad del sistema. Dicho seguimiento se hará en dos niveles: seguimiento del sistema o seguimiento de la aplicación.

Este sistema de políticas de privacidad ha sido desarrollado siguiendo la especificación XACML 1.0 definida por SUN y aceptada por OASIS (XACML - OASIS 2009) como estándar. El presente documento contiene información detallada acerca de dicha especificación y de la forma en que se ha integrado con el sistema de políticas de privacidad que nos ocupa.

A lo largo del resto del documento, se puede obtener información sobre el análisis, diseño, desarrollo e implementación seguidos para la realización de este Proyecto de Fin de Carrera.

Contenido

Índice de figuras

Índice de tablas

1 INTRODUCCIÓN

Hoy en día, se dispone de múltiples tecnologías que permiten localizar geográficamente un dispositivo, asociando el tiempo (fecha y hora) en que la información de localización fue recogida. Algunas de estas tecnologías están ampliamente extendidas, como son la tecnología GPS (Sistema de posicionamiento Global) o la GSM (Sistema Global para las comunicaciones Móviles).

Este tipo de tecnologías ha generado, en la última década, el surgimiento de un gran número de Servicios Basados en la Localización o LBS, del inglés "Location Based Services". Haciendo una descripción más concreta, los LBS son aquellos servicios que utilizan la posición geográfica de las entidades para proporcionar un valor añadido (Gonzalez-Tablas, y otros 2007).

El proyecto CERTILOC, del cual forma parte el presente Proyecto de Fin de Carrera, aporta un servicio completo para cubrir la certificación, la seguridad y la privacidad de la información intercambiada mediante distintas tecnologías de localización, en particular de las tecnologías GPS y la GSM.

El presente capítulo contiene, en primer lugar, una explicación extensa del contexto del proyecto CERTILOC.

En segundo lugar, se presentan los objetivos concretos del presente PFC.

Además, se introduce un glosario de términos utilizados a lo largo de todo el documento.

Por último se presenta un glosario completo de siglas, acrónimos y abreviaturas igualmente utilizadas a lo largo de todo el documento.

1.1 CONTEXTO: EL PROYECTO CERTILOC

Presentamos a continuación el contexto general del proyecto CERTILOC, del cual forma parte el presente Proyecto de Fin de Carrera.

1.1.1 INTRODUCCIÓN A CERTILOC

CERTILOC (Servicio de **CER**TIficación digital de la **LOC**alización) es un proyecto de investigación, financiado por el Ministerio de Educación y Ciencia, que se está llevando a cabo por el Grupo de Seguridad en las Tecnologías de la Información y las Comunicaciones de la Escuela Politécnica Superior de la Universidad Carlos III de Madrid.

CERTILOC responde a la unión de los conceptos de "**Certificación**" y "**Localización**" y tiene como objetivo principal, la propuesta de un modelo teórico de un sistema informático respetuoso con la **privacidad**, que permita **certificar digitalmente**, la **situación espacio-temporal** de un **dispositivo localizable** mediante distintas tecnologías de localización. Además propone la implementación de un demostrador real de dicho modelo.

La tesis doctoral de la tutora del presente Proyecto de Fin de Carrera (Gonzalez-Tablas Ferreres - Tesis 2005), presenta información detallada del modelo teórico de CERTILOC por lo que recomendamos al lector referirse a dicho documento en caso de querer ampliar la información que se presenta al respecto en el presente documento.

1.1.2 CERTILOC, UN SERVICIO DE CERTIFICACIÓN ESPACIO TEMPORAL RESPETUOSO CON LA PRIVACIDAD

La certificación digital de la información de localización manejada en el modelo teórico de CERTILOC, aporta varios aspectos y propiedades importantes a tener en cuenta:

- **Identidad**: permite identificar unívocamente a los actores de una comunicación. En el caso de CERTILOC nos permite identificar unívocamente a los dispositivos localizados. Es decir, cuando se recibe determinada información de la localización de un dispositivo, se puede asegurar y, lo que es más importante, demostrar que la información obtenida pertenece efectivamente al dispositivo localizado y no a otro.
- **Integridad**: permite comprobar que no se ha realizado manipulación intermedia alguna a la información transmitida en la comunicación. Es decir, se puede demostrar que la información obtenida en el destino (receptor) es exactamente igual a la emitida en el origen (emisor). En el caso de CERTILOC, nos permite demostrar que la información de localización obtenida de un determinado dispositivo, es exactamente la misma información emitida por el dispositivo localizado.
- **No repudio**: el no repudio nos permite demostrar la participación de ambas partes de una comunicación. El no repudio en origen, permite demostrar, al receptor de una comunicación, que cierto emisor efectivamente envió una información. Al contrario, el no repudio en destino, permite al emisor demostrar la recepción de la información por parte del destinatario. En el caso de CERTILOC, no sería posible alegar que un dispositivo localizado no haya tomado parte en el intercambio de la información de localización. Es decir, la entidad (ya sea una persona física o una entidad jurídica) responsable del dispositivo localizado, no podría renunciar a la información de localización obtenida desde su dispositivo.

Por otro lado, hoy en día, la **protección de datos personales** está completamente legislada y es de obligado cumplimiento tanto por las instituciones públicas como privadas (Agencia Española de Protección de Datos 2009).

Una de las razones que ha motivado esta situación es que en la última década y debido al constante crecimiento, desarrollo y uso de las tecnologías de la información y las comunicaciones, se ha prestado especial atención a la privacidad de los datos personales albergados en formatos electrónicos. Es por ello que en España, al igual que en gran parte de la Unión Europea y otros países desarrollados, se han desarrollado leyes específicas para regular y controlar la protección de los datos de carácter personal.

La Ley Orgánica 15/1999, de 13 de diciembre, de Protección de Datos de Carácter Personal en su primer artículo, indica que tiene por objeto garantizar y proteger, en lo que concierne al tratamiento de los datos personales, las libertades públicas y los derechos fundamentales de las personas físicas, y especialmente de su honor e intimidad personal y familiar (BOE núm. 298 1999). En el segundo artículo nos indica que será de aplicación a los datos de carácter personal registrados en soporte físico, que los haga susceptibles de tratamiento, y a toda modalidad de uso posterior de estos datos por los sectores público y privado.

En CERTILOC, la información intercambiada implica un acceso a información privada y personal que pertenece a las entidades (personas físicas o entidades jurídicas) responsables de los dispositivos localizados. Dado el carácter confidencial de la información que se maneja, CERTILOC necesita implementar ciertos mecanismos que aseguren la privacidad de dichas entidades. Es necesario que las entidades responsables puedan gestionar la privacidad de sus datos, debiendo permitirles indicar quién, cómo y cuándo puede hacer uso de la información relacionada con sus dispositivos, en particular de la información de su localización en un momento determinado del tiempo (lo que en CERTILOC se define como Información Espacio Temporal o IET). Además deben poder conocer en todo momento el uso que se está haciendo de su información.

La privacidad en el acceso a los datos, tal y como describe el modelo teórico de CERTILOC, debe proporcionarse mediante políticas de seguridad, gestionables por las entidades responsables de los distintos dispositivos localizables.

Por otro lado, el proyecto CERTILOC, además de plantear un modelo teórico para la certificación digital de la información de localización, persigue crear un demostrador real de dicho modelo teórico. El demostrador debe ser capaz de interactuar con varias tecnologías de

estimación de la posición y de cumplir con todos los aspectos posibles de seguridad de los que se especifican en el modelo teórico[1].

El demostrador de CERTILOC, en su conjunto, ofrece las siguientes funcionalidades:

- **Localización de dispositivos**: CERTILOC permite obtener la Información Espacio-Temporal (la localización, también conocida como IET) de los distintos dispositivos localizados. Dichos dispositivos funcionan bajo la tecnología GSM o GPS (en un futuro se pretende incluir la localización mediante RFID pero, actualmente, el prototipo de CERTILOC no soporta este tipo de tecnología).
- **Certificación digital de la localización**: CERTILOC permite generar un Certificado digital de la información Espacio-Temporal (o CET) obtenida de cierto dispositivo. Un certificado espacio-temporal (CET), es un documento electrónico que asocia cierto dispositivo o entidad con determinada información espacio-temporal, es decir, con el lugar y el momento en el que ésta se encontraba en el momento de la localización. En el caso de CERTILOC el formato de los CET se basa en el estándar SAML descrito por "OASIS Security Services" (OASIS 2009). CERTILOC se encarga del ciclo de vida completo de los certificados, es decir, de su generación, su conservación, su transferencia, su verificación y su eliminación (Memoria PFC - Calvo Martínez 2007) .
- **Respeto de la privacidad**: CERTILOC ofrece mecanismos, a las distintas entidades responsables de los dispositivos localizados, para garantizar la privacidad de su información personal, en particular de su localización en el tiempo. Estos mecanismos incluyen la gestión de políticas de privacidad, el seguimiento de la actividad de las mismas a lo largo de la vida del sistema y, como es lógico, su cumplimiento ante distintas peticiones de servicios implicadas con sus dispositivos.

El Proyecto de Fin de Carrera, cuyo desarrollo se describe en el presente documento, contempla la implementación real de varios módulos del demostrador de CERTILOC que conforman el sistema completo de políticas de privacidad.

1.1.3 CERTILOC Y LAS POLÍTICAS DE PRIVACIDAD

Las **políticas de privacidad** se pueden definir en función de varios parámetros. Los más importantes son el **rol** bajo el que el solicitante actúa, el **identificador del usuario** que solicita el servicio, el **dispositivo** sobre el que se solicita el servicio, la **acción** solicitada, la **información**

[1] No todos los aspectos de seguridad requeridos en el modelo teórico han podido ser implementados en el demostrador.

espacio-temporal relacionada con el dispositivo a localizar y la información temporal sobre el **momento de ejecución de la petición de autorización** hacia el sistema de políticas de privacidad. A continuación pasamos a describir en detalle cada uno de estos parámetros:

- El **rol** bajo el que actúa el solicitante de la petición de autorización. Se relaciona directamente con la finalidad para la que el solicitante utilizará la información obtenida. La solicitud de localización espacio-temporal de un dispositivo, ya sea de manera inmediata o de manera diferida, puede tener distintas motivaciones según la persona o entidad que está solicitando dicha información. Un ejemplo de un determinado **rol** de solicitud sería un gerente de una empresa que desea obtener la información de localización de su flota de automóviles para evaluar la actividad de sus comerciales. En este caso, el **rol** del solicitante sería el de "Jefe". CERTILOC debe permitir al usuario localizado definir bajo qué circunstancias desea que su jefe le pueda localizar. En esta situación sería lógico pensar que el usuario sólo desea ser localizado por personas con este rol durante su horario de trabajo y no fuera del mismo.
- El **identificador** del solicitante de la petición de autorización, es decir, del usuario solicitante del servicio. Cada usuario de CERTILOC tendrá un identificador único que le distinguirá del resto de usuarios. Las políticas, al igual que con el **rol** descrito en el párrafo anterior, deben poder especificar el identificador usuario que solicita el servicio y que por tanto genera la petición de autorización. Por ejemplo, un usuario responsable de un dispositivo podrá definir bajo qué circunstancias le puede localizar el usuario "Juan" (Juan es el identificador CERTILOC del solicitante).
- El **dispositivo** sobre el que se solicita el servicio que genera la petición de autorización. Los dispositivos que participen en CERTILOC tendrán un identificador unívoco que los diferenciará del resto. De esta manera, un usuario (entidad) responsable de un dispositivo pude definir bajo qué circunstancias se puede localizar ese dispositivo en concreto. Es decir, el usuario de CERTILOC "Juan" podrá definir políticas de privacidad que regulen el acceso a la IET de su dispositivo "GPS_DE_JUAN" (Juan sería el usuario responsable de dispositivo y GPS_DE_JUAN sería el identificador CERTILOC unívoco de un dispositivo GPS del que Juan es responsable). Las acciones de localización siempre se realizarán sobre un dispositivo concreto existente en el sistema CERTILOC.
- La **acción** a realizar en la petición de autorización. CERTILOC no sólo permite a sus usuarios realizar la **localización espacio-temporal** (IET) de dispositivos sino que además, CERTILOC permite a sus usuarios poder llevar a cabo acciones de **generación**, **descarga** y **eliminación** de Certificados Espacio Temporales (CET). El modelo propuesto por CERTILOC

permite definir políticas de privacidad que tengan en cuenta la **acción** que está llevando a cabo el usuario solicitante cuando intenta acceder a la información privada propiedad de otro usuario (ya sea la IET de uno de sus dispositivos o la información de sus CETs, que a su vez estarán relacionadas con los dispositivos de los que el usuario es el responsable). Por ejemplo, el usuario "Juan" podrá definir bajo qué circunstancias concretas se permite a otro usuario "**Obtener**" la **IET** de su dispositivo "GPS_DE_JUAN".

- La **información espacio-temporal (IET)**, tal y como se ha definido anteriormente, es la información sobre la localización espacial de un determinado elemento en un determinado momento. CERTILOC permite a sus usuarios definir políticas de privacidad que autoricen o denieguen una determinada petición de autorización, teniendo en cuenta la información espacio-temporal que ésta conlleve. Por ejemplo, un usuario responsable de un dispositivo podrá definir políticas que autoricen que éste sea localizado sólo si en el **momento de la localización**, el dispositivo se encuentra dentro de una **localización** o zona concreta. De esta manera, el usuario "Juan" podrá definir políticas para que su dispositivo "GPS_DE_JUAN" sólo se pueda localizar (Obtener IET) si éste se encuentra en la Comunidad de Madrid y en el momento de localizarlo, son entre las 08:00 am y las 20:00 pm.
- El **momento de ejecución de la petición** se relaciona con el momento exacto en el que se ejecuta la petición de acceso. Cabe remarcar que CERTILOC contempla la posible solicitud de **peticiones de información diferidas**. Es decir, peticiones donde se desea **acceder a la IET** de un dispositivo en un **determinado momento**, y se desean **consultar** los datos de IET recogidos en el momento de la petición, en otro **momento posterior**. De esta manera, un usuario puede localizar un dispositivo en un momento determinado y querer disponer de la información de localización en otro momento. De esta manera, un usuario responsable de dispositivo debe poder especificar políticas de privacidad donde se tenga en cuenta el momento concreto en el que se desea disponer de cierta IET recogida anteriormente. El usuario "Juan" puede indicar que sólo permite que el usuario "Pepe" disponga de la IET de su dispositivo "GPS_DE_JUAN" si se desea disponer de la IET entre las 08:00 am a las 21:00 pm. Si el usuario "Pepe" localiza el dispositivo "GPS_DE_JUAN" a las 13:00 pm y quiere consultar o disponer de la IET recogida a las 21:30, el sistema le debe denegar el acceso por mor de la política definida anteriormente por el usuario "Juan".

1.1.4 ARQUITECTURA PARA EL DEMOSTRADOR DE CERTILOC

En la arquitectura general (Figura 1) del demostrador real de CERTILOC, la mayor parte de la lógica y los datos se concentran en una sola aplicación denominada **núcleo**. El modelo

teórico permite que los servicios se ejecuten de manera independiente en contenedores o servidores distintos pero en esta primera implementación se ha decidido concentrar toda la funcionalidad en un solo contenedor – en una única aplicación – donde los distintos módulos interactúan entre sí mediante interfaces o fachadas definidas en las distintas clases Java (Java - Sun Microsystems , Inc. 2009) que los conforman.

El sistema de políticas de privacidad o **SPP** se encuentra dentro de dicho núcleo.

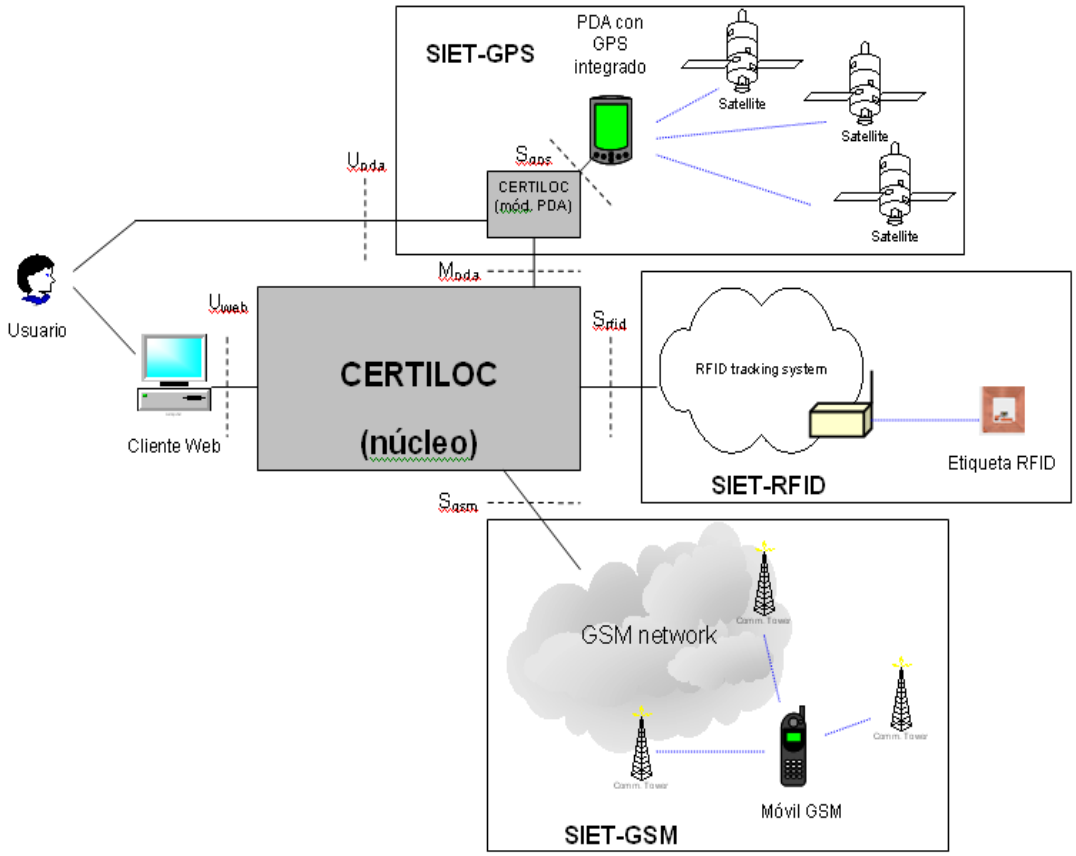


Figura 1. *Visión general del sistema CERTILOC*

De cara al usuario, los servicios que CERTILOC ofrece se concentran principalmente en una aplicación web (en PDA para los servicios de generación de IET de dispositivos GPS). La interfaz de usuario del **núcleo** deberá ser accedida mediante cualquier cliente web (navegadores web en general).

La primera implementación del demostrador real de CERTILOC, interactúa con dos servicios de información espacio-temporal, o sistemas de posicionamiento, para obtener la información de localización: PDAs conectados a dispositivos **GPS** y móviles **GSM** (RFID estaba incluido en el planteamiento inicial del demostrador pero no se ha implementado en esta primera versión). Cada una de estas tecnologías está correspondida con una interfaz que

permite comunicarse con los dispositivos implicados. A continuación introducimos cada una de estas interfaces:

- **S_{gps}** será la interfaz encargada de comunicarse con el Servicio de Información Espacio-Temporal basado en GPS – **SIET GPS**.
- **S_{gsm}** será la interfaz encargada de comunicarse con el Servicio de Información Espacio-Temporal basado en redes GSM – **SIET GSM**.
- **S_{rfid}** En un futuro será la interfaz encargada de comunicarse con el Servicio de Información Espacio-Temporal basado en sistemas RFID – **SIET RFID**.

En el caso de etiquetas RFID y de dispositivos GSM, será el **núcleo** quien interactúe directamente con los servicios de información espacio temporal. Para las PDAs, el **núcleo** no se comunicará directamente con los servicios de información espacio temporal sino que serán los propios dispositivos los que obtendrán su localización y se comunicarán con el **núcleo** para notificar sus respectivas informaciones.

La siguiente figura nos muestra un detalle de la arquitectura del demostrador de CERTILOC por módulos. En el apartado siguiente 1.1.5 se comenta a qué proyecto concreto se corresponden los acrónimos de cada módulo, además de poder consultarse su definición en el apartado 1.4 de Siglas, acrónimos y abreviaturas.

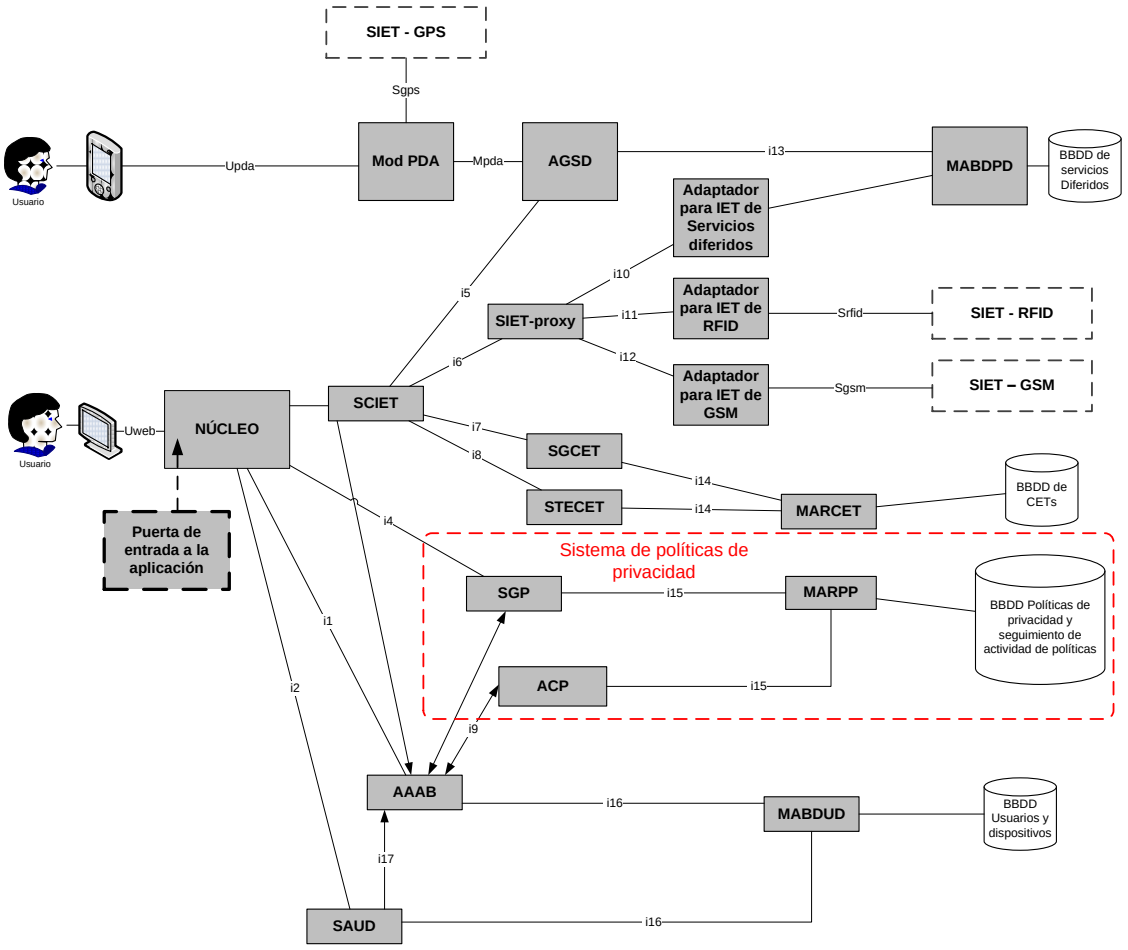


Figura 2. *Arquitectura por módulos del demostrador de CERTILOC*

Vemos remarcado el sistema de políticas de privacidad o SPP, que es el objeto de desarrollo del presente PFC. Tal y como se observa en la figura, la propuesta inicial del SPP contempla tres módulos:

- **SGP**: Sistema de Gestión de Políticas de privacidad – Será una aplicación web designada a la gestión de las políticas de privacidad y al seguimiento de la actividad de las políticas por parte de los usuarios de CERTILOC.
- **ACP**: Agente Custodio de la Privacidad – Será el sistema encargado de la evaluación de peticiones de autorización contra las políticas de privacidad definidas por los usuarios del sistema.
- **MARPP**: Módulo de Acceso al Repositorio de Políticas de Privacidad – Será una capa intermedia que gestionará un repositorio persistente con las políticas de privacidad definidas por los usuarios del sistema CERTILOC.

- **Base de datos de políticas de privacidad** o **RPP**: Será el repositorio físico de los datos persistentes de políticas de privacidad.

En el apartado 4.2 del presente documento se encuentra una explicación extensa de la arquitectura de diseño del sistema de políticas de privacidad del demostrador de CERTILOC.

1.1.5 EVOLUCIÓN DEL DEMOSTRADOR DE CERTILOC

La implementación del demostrador de CERTILOC ha sido dividida en cuatro Proyectos de Fin De Carrera. Cada uno de estos cuatro proyectos implementa una serie de aspectos básicos que, en su conjunto cumplen con el modelo teórico del proyecto CERTILOC.

Vemos a continuación un resumen de las distintas aportaciones de dichos proyectos:

- El primer proyecto desarrollado (Memoria PFC - Calvo Martínez 2007), sentaba las bases para el desarrollo del conjunto del demostrador incluyendo los siguientes módulos:
  - Un **módulo básico o núcleo,** que aportaba una base de entrada para acceder a todas las funcionalidades del demostrador de CERTILOC, y por lo tanto, al resto de módulos del sistema. Asociada a la creación de este módulo, se desarrolla la infraestructura necesaria que define la arquitectura del resto del demostrador siguiendo el patrón Modelo-Vista-Controlador (MVC).
  - Varios módulos para la **Gestión de la información espacio temporal** (**IET**), sobretodo en cuanto al manejo de dicha información y para adaptar la tecnología GSM al demostrador (recoger y gestionar los datos de localización) utilizando un simulador software de este tipo de redes.
  - Varios módulos para el **manejo de los certificados digitales** de información espacio-temporal, sobretodo en cuanto a la creación, el almacenamiento y la recuperación de **CETs.**
  - Simuladores para la autenticación de los usuarios y para el sistema de autorización de peticiones (Políticas de privacidad).
  - El desarrollo completo de este PFC, contemplaba los siguientes módulos de los presentados en la Figura 2: **Núcleo**, **SCIET**, **SIET-proxy**, **Adaptador para IET GSM**, **SIET-GSM, SGCET, STECET, MARCET, BBDD de CETs** (referirse al apartado 1.4 o al 4.2 para ver una definición de estos módulos).
- El segundo PFC (Memoria PFC - de Fuentes García-Romero de Tejada 2007), ampliaba el primero y aportaba nuevas funcionalidades:

- **Obtención de la información espacio temporal mediante la tecnología GPS**, realizado sobre dispositivos de posicionamientos reales y no simulados (PDAs con receptor GPS integrado).
- **Manejo de la información** del sistema **GPS** e **integración del mismo en el demostrador.**
- **Generación de CETs auto firmados.** Este formato más sencillo de CET generado en los módulos PDA, posteriormente se integra en un CET de los mencionados previamente.
- **Manejo de peticiones diferidas realizadas para los dispositivos PDA.**
- El desarrollo completo de este PFC, contemplaba los siguientes módulos de los presentados en la Figura 2: **Mod. PDA**, **SIET-GPS**, **AGSD**, **MABDPD**, **Adaptador para IET de Servicios diferidos, BBDD de Servicios Diferidos** (referirse al apartado 1.4 o al 4.2 para ver una definición de estos módulos).

- El tercer PFC (Memoria PFC - Gallo Martínez 2008) ampliaba y mejoraba ciertos aspectos del módulo básico o núcleo de CERTILOC y de la seguridad y gestión del conjunto de la aplicación incluyendo:
  - Desarrollo de las funcionalidades de **administración de usuarios** de CERTILOC y la administración de los **dispositivos localizables** (incluimos aquí los roles entre usuarios, y la asociación de responsabilidad de ciertos dispositivos a dichos usuarios o entidades)
  - Creación de un **repositorio persistente de datos de entidades, dispositivos** y relaciones entre los mismos
  - Creación del **sistema de autenticación de usuarios** en la interfaz Web, incluyendo autenticación mediante certificados digitales X.509 además de mediante contraseñas.
  - Creación del **sistema de control de acceso** (autorización), según el rol del usuario web (usuario o administrador), para la ejecución de las distintas funciones y servicios ofrecidos por el sistema. El sistema de control de acceso, delega la decisión de autorización al sistema de políticas de privacidad, ante peticiones de servicios que impliquen el procesamiento de información espacio temporal de un dispositivo, cuyo responsable es distinto del usuario que solicita el servicio.
  - Simulación del módulo de decisión de autorizaciones del sistema de políticas de privacidad

- Incorporación en CERTILOC de **mecanismos que mejoran la seguridad global del sistema** como son el establecimiento de sesiones seguras para los usuarios y autenticación de las mismas a lo largo de su interacción con el sistema, el establecimiento de conexiones seguras empleando SSL y el almacenamiento en las bases de datos de CERTILOC, de históricos de datos y de registros de acciones y eventos del sistema.
- El desarrollo completo de este PFC, contemplaba los siguientes módulos de los presentados en la Figura 2: **Núcleo (parcialmente)**, **SAUD**, **AAAB**, **MABDUD**, **BBDD Usuarios y Dispositivos** (referirse al apartado 1.4 o al 4.2 para ver una definición de estos módulos).

- El presente PFC aporta un sistema completo de privacidad mediante políticas. Este sistema de políticas de privacidad (SPP) se integra con el resto de módulos del demostrador de CERTILOC para aportar:
  - Un sub-sistema de **autorización de peticiones de autorización mediante políticas de privacidad**. Las peticiones de autorización vienen derivadas de la solicitud de distintos servicios por parte de los usuarios de CERTILOC. Este sub-sistema se encarga de recibir determinadas peticiones de autorización, desde otros módulos CERTILOC, y de evaluarlas contra las políticas de privacidad activas en el sistema. Además, este sistema toma parte en la recogida de datos sobre los eventos del sistema y la actividad de las distintas políticas para poder hacer un seguimiento posterior. Este subsistema se denomina **ACP** (Agente Custodio de las Políticas de Privacidad).
  - Un sistema completo de **gestión de políticas de privacidad** que permite crear, borrar, modificar, activar y desactivar cualquier elemento de uno o varios conjuntos de políticas a través de una interfaz web. Los usuarios pueden declarar el estado de una política como Activo o Inactivo para decidir si se aplica contra posibles peticiones de autorización que lleguen al **ACP**. Este subsistema es parte del sistema denominado **SGP** o Sistema de Gestión de la Privacidad.
  - Un **repositorio de datos persistentes** para las distintas políticas de privacidad y para los registros de su actividad ante las evaluaciones del **ACP**. Este repositorio es parte del sistema llamado **MARPP** o Módulo de Acceso al Repositorio de Políticas de Privacidad.

- Una **capa de acceso intermedio al repositorio** de datos persistente. Forma la otra parte del módulo MARPP. Se encarga de obtener distintos datos y vistas de los datos del repositorio de políticas de privacidad y de su actividad.
- Un sistema de **seguimiento de los eventos de sistema del SPP y de la actividad de las políticas de privacidad ante peticiones de autorización** de los distintos usuarios. Con respecto al seguimiento de las políticas de privacidad, es la otra parte del sistema SGP ya que se encarga de mostrar a los distintos usuarios responsables de dispositivos, la actividad del sistema y de las políticas de privacidad contra las peticiones de autorización recibidas por el ACP. Por otra parte, también registra y almacena los distintos eventos del sistema únicamente contemplando el funcionamiento interno de todos los módulos del SPP, sin poner en riesgo la privacidad de los usuarios (sólo registra eventos del funcionamiento del sistema y no de políticas de privacidad, peticiones o respuestas de autorización).
- Un sistema de traducción de formatos para transformar peticiones y respuestas de autorización al formato propuesto por el estándar XACML (XACML - OASIS 2009) y viceversa. Este estándar es el que se utiliza en CERTILOC para definir las políticas de privacidad. Este sistema se denomina Agente Gestor de Peticiones de Acceso o **AGPA**.

La siguiente figura muestra un resumen de la aportación de los distintos módulos de los anteriores proyectos de fin de carrera, y la integración del los módulos del presente PFC con los mismos.

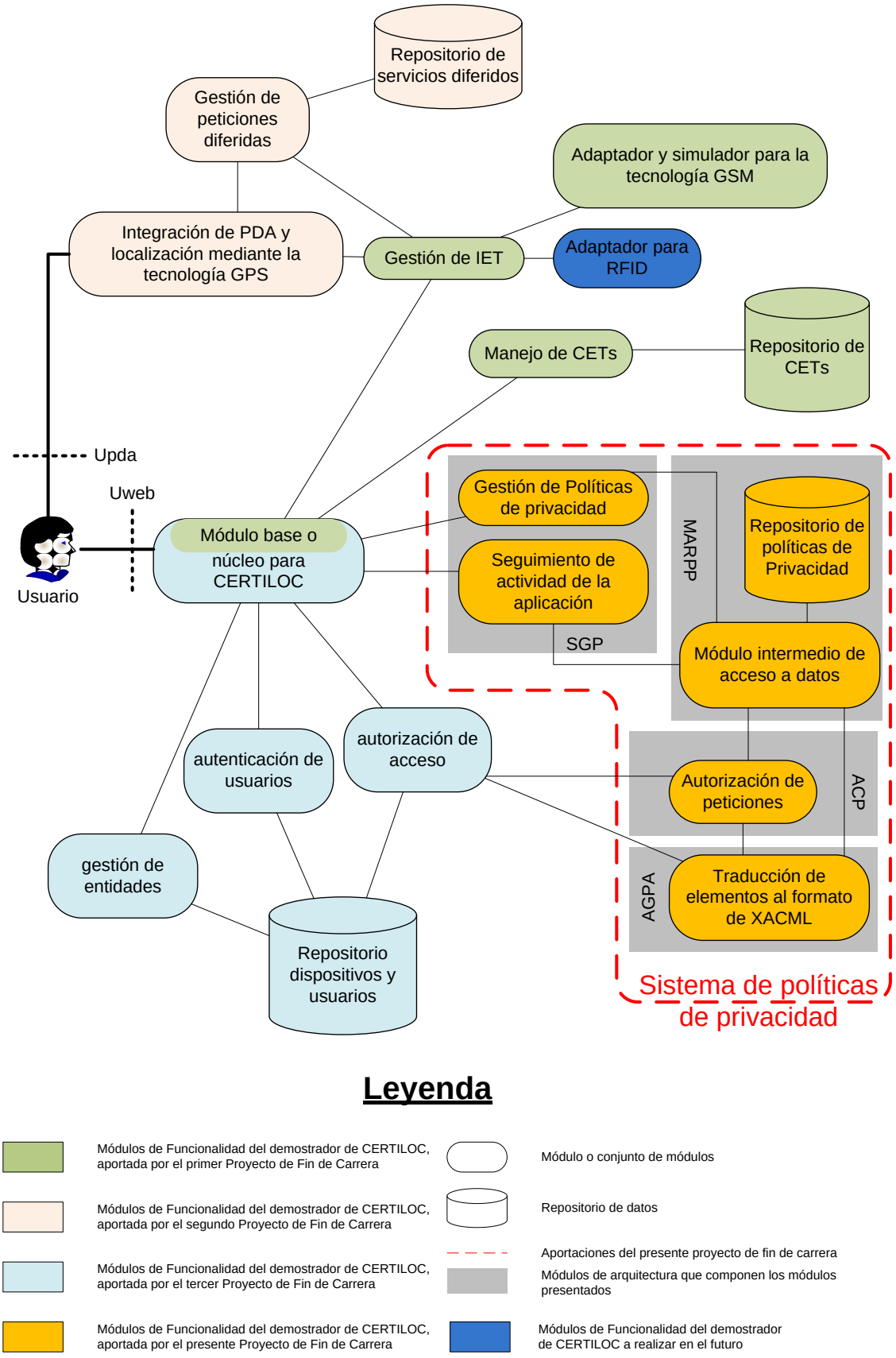


Figura 3. *Diseño general del demostrador de CERTILOC*

Tal y como se observa en la anterior figura, cada fase de desarrollo del demostrador de CERTILOC ha ido aportando cierta funcionalidad concreta, bien definida y separada del resto de proyectos. La coordinación de todos los proyectos ha supuesto un esfuerzo extra por parte de todos los autores, para comprender lo que cada uno debía realizar.

1.1.6 TECNOLOGÍAS PARA EL MARCO COMÚN DE DESARROLLO

Para facilitar la integración de los distintos módulos de CERTILOC, se han propuesto las siguientes tecnologías como base para los desarrollos de los distintos proyectos de fin de carrera:

- Java (Java - Sun Microsystems , Inc. 2009) como lenguaje de desarrollo
- MySQL (MySQL AB 2009) como motor de bases de datos para los repositorios de datos
- STRUTS (Struts - Apache Software Foundation 2009) como marco de desarrollo para la implementación de los módulos relacionados con la interfaz de usuario Web
- Apache Tomcat (The Apache Software Foundation 2009) como servidor Web para el alojamiento de los módulos relacionados con la interfaz de usuario Web

Además, en el presente PFC se ha decidido utilizar el **estándar XACML** (XACML - OASIS 2009) como base para la implementación del flujo de datos del sistema de políticas de privacidad y para su modelo de datos. El estándar **XACML** define un lenguaje común para la creación de políticas de seguridad orientadas al control de acceso a la información, utilizando para ello **XML**. Además, propone un modelo del flujo de datos que deben seguir los distintos agentes o actores que intervienen en la evaluación de autorización de las distintas peticiones, contra las políticas de seguridad del sistema.

Para facilitar la tarea de adaptación del sistema de políticas de privacidad al estándar, se utiliza un API desarrollado expresamente por Sun Microsystems para la integración de sistemas con el estándar XACML. Este API se denomina Sun's XACML Implementation (XACML - Sun Microsystems , Inc. 2009) y, al igual que el propio estándar XACML, su uso está muy poco extendido y es relativamente inmaduro.

Cabe remarcar que éste es uno de los mayores retos del presente PFC ya que **XACML** es un estándar con un uso poco extendido en la actualidad y ofrece una documentación y un soporte escaso y difícil de encontrar.

1.2 OBJETIVOS DEL PRESENTE PROYECTO

A modo de recopilación de lo comentado en los apartados anteriores, el sistema de políticas de privacidad de CERTILOC tiene los siguientes objetivos básicos:

- Evaluar si se debe permitir o denegar una determinada petición de autorización ante las políticas de privacidad activas en el sistema.
- Ofrecer un repositorio persistente para las políticas de privacidad definidas por los usuarios.
- Proporcionar a los usuarios una manera de gestionar sus políticas de privacidad.
- Permitir hacer un seguimiento de la actividad del sistema. Dicho seguimiento se hará en dos niveles: seguimiento del sistema o seguimiento de la aplicación.

Este sistema de políticas de privacidad debe desarrollarse siguiendo la especificación XACML 1.0 definida por SUN y aceptada por OASIS (XACML - OASIS 2009) como estándar.

1.3 GLOSARIO DE TÉRMINOS

- **Software libre**: Producto software cuya licencia es de libre distribución y que, en ocasiones, contiene un código fuente accesible y modificable. En cualquier caso, suele destacar que son productos de software gratuitos.
- **Framework**: Marco de trabajo para el desarrollo de aplicaciones y productos software.
- **Script**: Guión, generalmente utilizado para llevar a cabo una serie de instrucciones de manera secuencial.
- **Java Eclipse**: Entorno de desarrollo especialmente creado para el desarrollo de aplicaciones Java (Java - Sun Microsystems , Inc. 2009). Es un entorno versátil que puede valer para el desarrollo en otras plataformas.
- **Hardware**: Conjunto de los componentes que integran la parte material de una computadora (Diccionario - Real Academia de la Lengua Española 2009)
- **Software**: Conjunto de programas, instrucciones y reglas informáticas para ejecutar ciertas tareas en una computadora (Diccionario - Real Academia de la Lengua Española 2009)
- **Entorno de producción:** Entorno real y definitivo de una aplicación software.

- **Plugin:** componente instalable para ampliar la funcionalidad de una herramienta concreta.
- **Código abierto:** Aplicaciones cuyo código de programación es públicamente accesible y modificable para adaptarlo a proyectos personales o para aportar funcionalidad o mejoras a la pieza de código en cuestión.
- **Aplicativo:** Término utilizado para describir una función aplicada a una condición en el entorno de XACML y del modelo de datos de políticas privacidad de CERTILOC (apartado 4.3 del presente documento).
- **Uniform Resource Identifier**: Es una cadena compacta de caracteres que permite identificar unívocamente un recurso o elemento físico o abstracto (The Internet Society 1998).

1.4 SIGLAS, ACRÓNIMOS Y ABREVIATURAS

- LBS: "Location Based Services" o Servicios Basados en la Localización
- MÉTRICA: Metodología para el desarrollo de productos software.
- ACP: Agente Custodio de Políticas de privacidad.
- AGPA: Agente Gestor de Peticiones de Acceso.
- SCIET: Servicio de Certificación de Información Espacio-Temporal.
- MARPP: Módulo de Acceso al Repositorio de Políticas de Privacidad.
- SGP: Sistema de Gestión de Políticas de Privacidad.
- XACML: Extensible Access Control Markup Language.
- PFC: Proyecto de Fin de Carrera.
- CERTILOC: Sistema de Certificación de la Localización.
- POO: Programación Orientada a Objetos
- MVC: Modelo Vista Controlador
- Sgps: interfaz encargada de comunicarse con el servicio de información espacio-temporal basado en GPS
- Srfid: interfaz encargada de comunicarse con el servicio de información espacio-temporal basado en RFID
- Sgsm: interfaz encargada de comunicarse con el servicio de información espacio-temporal basado en GSM
- Urole: concepto de rol de usuario o User Role.
- J2EE: Java 2 Enterprise Edition

- RRHH: Recursos Humanos
- HW: Hardware
- SW: Software
- JSP: Java Server Page
- CET: Certificado de información Espacio Temporal
- IET: Información Espacio Temporal
- XML: Lenguaje de Marcado Extensible
- API: Interfaz de Programación de Aplicaciones
- PDP: Punto de Decisión de Políticas
- PEP: Punto de Información de Políticas
- PIP: Punto de Ejecución de Políticas
- PAP: Punto de Administración de Políticas
- IDE: Entorno de desarrollo unificado o integrado (del inglés Integrated Development Environment)
- URI: Identificador de Recurso Uniforme (Uniform Resource Identifier)
- CSS: Hojas de diseño en cascada ("Cascading Style Sheets"
- RFID: Radio Frequency IDentification – Identificación por radiofrecuencia
- GPS: Global Positioning System – Sistema de posicionamiento global
- GSM: Groupe Spécial Mobile - Sistema Global para las Comunicaciones Móviles
- PDA: Personal Digital Assistant - Asistente Digital Personal
- SPP: Sistema de Políticas de Privacidad
- AAAB**:** Agente de Autenticación y Autorización Básica
- AGSD: Agente Gestor de Servicios Diferidos
- i_n: Interfaces de comunicación entre los módulos componentes de CERTILOC
- MABDSD: Módulo de Acceso a la Base de Datos de Servicios Diferidos
- MABUD: Módulo de Acceso a la Base de Datos de Usuarios y Dispositivos
- MARCET: Módulo de Acceso al Repositorio de Certificados Espacio-Temporales
- Mód. PDA: Módulo para el PDA
- RCET: Repositorio de Certificados Espacio-Temporales
- RPP: Repositorio de Políticas de Privacidad
- SAUD: Servicio de Administración de Usuarios y Dispositivos
- SGCET: Servicio de Generación de Certificados Espacio-Temporales
- SIET-GPS: Servicio de Información Espacio-Temporal basado en GPS

- SIET-GSM: Servicio de Información Espacio-Temporal basado en redes GSM
- SIET-proxy: Proxy para los Servicios de Información Espacio-Temporal
- SIET-RFID: Servicio de Información Espacio-Temporal basado en sistemas RFID
- STECET: Servicio de Transferencia y Eliminación de Certificados Espacio-Temporales

2 GESTIÓN DEL PROYECTO

En el siguiente apartado se presentan todos los procedimientos seguidos para la gestión del presente Proyecto de Fin de Carrera. La gestión del proyecto ha permitido definir la planificación de las tareas básicas del proyecto así como definir las actividades principales y procedimientos necesarios a seguir para crear un proyecto consistente, que cumpla con la especificación del usuario.

Veremos una breve introducción a la gestión de proyectos software, la organización del proyecto, los riesgos potenciales que podrían desviar la planificación, los medios técnicos con los que se cuenta para el desarrollo del mismo, los mecanismos y procedimientos de control que se deben aplicar para cumplir con la planificación, la planificación inicial estimada y el seguimiento real final y, por último, un presupuesto detallado para evaluar los costes del desarrollo.

2.1 INTRODUCCIÓN A LA GESTIÓN

Todo proyecto software que se quiera desarrollar desde el punto de vista de la Ingeniería del software, debe contar con una gestión y una planificación inicial que permita cumplir exitosamente con las expectativas y objetivos del mismo en un tiempo determinado.

La gestión del proceso de desarrollo de cualquier componente o sistema informático, ya sea Software o Hardware, permitirá evaluar los posibles riesgos y costes del proyecto en sí mismo. Nosotros, como futuros ingenieros, debemos tener muy en cuenta dicha gestión y, debemos acuñar procesos de gestión, sólidos y fiables, que nos permitan plantear una perspectiva ingenieril para los proyectos y tareas a los que nos enfrentamos hoy en día y en el futuro.

Cabe destacar que, en cuanto a la gestión del presente proyecto, debido a la inexperiencia del autor del presente documento, y otros elementos externos, la desviación entre la planificación estimada y el seguimiento real ha sido notablemente significativa, como veremos en los siguientes puntos. No obstante, hay que comentar que este hecho no se recibe negativamente, todo lo contario, se observa como un punto positivo, desde la perspectiva formativa. Me ha ayudado a obtener y aprehender experiencia y a mejorar, potencialmente, los procesos de gestión para proyectos futuros y presentes.

2.2 ORGANIZACIÓN DEL PROYECTO

Veremos a continuación las directrices seguidas para la organización del proyecto. Comenzando por la metodología de desarrollo seleccionada para llevar a cabo el mismo y continuando por una descripción del modelo del proceso de desarrollo seguido.

2.2.1 METODOLOGÍA DE DESARROLLO

La metodología escogida para la gestión de tareas y seguimiento del desarrollo y configuración del presente Proyecto de Fin de Carrera ha sido MÉTRICA V.3 (Métrica V.3 - Consejo Superior de Administración Electrónica 2009).

MÉTRICA es una metodología de desarrollo, ampliamente utilizada en el territorio Español, específicamente en las administraciones públicas, que tiene varios años de vida. Tal y como se ha mencionado, se ha escogido la versión tres (V.3) de dicha metodología. Tal y como indica la especificación de MÉTRICA V.3: "*La metodología MÉTRICA Versión 3 ofrece a las Organizaciones un instrumento útil para la sistematización de las actividades que dan soporte al ciclo de vida del software*".

La aplicación de dicha metodología persigue los siguientes objetivos principales:

a) *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.*
b) *Dotar a la Organización de productos software que satisfagan las necesidades de los usuarios dando una mayor importancia al análisis de requisitos.*
c) *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.*
d) *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.*
e) *Facilitar la operación, mantenimiento y uso de los productos software obtenidos.*

No es objetivo del presente documento parafrasear la especificación completa de la mencionada metodología y recomendamos al lector la visita a la página web del Consejo

Superior de Administración Electrónica (Consejo Superior de Administración Electrónica 2009) si desea profundizar en el conocimiento de dicha especificación.

Además de proponer procesos detallados para el desarrollo de sistemas de información, MÉTRICA V3 (Métrica V.3 - Consejo Superior de Administración Electrónica 2009) propone unas interfaces para asegurar la planificación, sostenibilidad y calidad de los proyectos de desarrollo de software. Estas interfaces también serán adoptadas en el presente proyecto.

Por último, destacar que la adaptación del presente desarrollo a dicha especificación se hace de una manera abierta, es decir, se permitirán excepciones a la especificación de la metodología y se adaptarán algunas de las tareas y actividades que propone, siempre y cuando las circunstancias y características propias del Proyecto así lo exijan.

2.2.2 MODELO DEL PROCESO DE DESARROLLO

La metodología MÉTRICA, en su versión 3, propone un modelo de proceso de desarrollo basado en el modelo en cascada. El presente proyecto será desarrollado siguiendo este modelo de proceso añadiendo el **retroceso**. Esta será la primera adaptación particular que se realizará de dicha metodología. El **retroceso**, nos permitirá volver a fases anteriores del desarrollo, aun encontrándose el proceso de desarrollo en fases posteriores.

El modelo de desarrollo en cascada con retroceso se ve marcado por estar dividido en fases secuenciales. Cada una de las fases tiene un objetivo concreto y unos subproductos concretos y deben ser desarrolladas secuencialmente. Es decir, no se permite pasar a una fase posterior hasta haber concluido, como mínimo, una versión preliminar de la anterior. El proyecto actual exige el uso de estándares y tecnologías concretas para su implementación. El desconocimiento de algunos de estos estándares y tecnologías puede suponer que no se hayan considerado ciertos problemas en fases anteriores del proceso de desarrollo. El retroceso nos permitirá volver a fases anteriores para replantear el uso de estos estándares y tecnologías de una manera más precisa.

Este modelo está marcado por las siguientes fases:

a) **Análisis de requisitos:** Durante esta fase se analizarán los requisitos de los módulos necesarios para el desarrollo del sistema de políticas de privacidad de CERTILOC. Para llevar a cabo dicho análisis, se mantienen varias reuniones con la tutora del proyecto. Además, se estudiará el documento de análisis,

proporcionado por la misma tutora del proyecto, orientado al análisis del demostrador real de CERTILOC.

b) **Diseño de sistema:** En esta fase se descompone el sistema en módulos y se diseña el funcionamiento de cada uno de ellos. Además, se estudiará la especificación del estándar XACML y se buscarán librerías y utilidades que nos pudiesen valer para la implementación real del sistema de políticas de privacidad. El diseño a crear contendrá el diagrama de clases necesarias para la implementación de nuestro sistema.

c) **Codificación del software:** Basándose en la especificación de diseño realizada en la fase anterior, se codificarán las distintas clases necesarias para la implementación de la aplicación.

d) **Pruebas del sistema:** Durante esta fase se creará un banco de pruebas orientadas a encontrar errores de la implementación creada. Además, ante posibles errores encontrados, se refinará el código del software para corregirlos.

e) **Implantación:** Finalmente, se integrarán todos los módulos desarrollados con el resto de módulos del demostrador real de CERTILOC.

A continuación mostramos un diagrama que ilustra el modelo de desarrollo.

Figura 4. *Modelo de desarrollo del sistema*

2.3 GESTIÓN DE MEDIOS Y RIESGOS DEL PROYECTO

Presentamos a continuación los detalles de la gestión de medios y riesgos del presente proyecto. En este apartado podremos ver una especificación de los posibles riesgos que podrían provocar una desviación en la planificación inicial especificada para el proyecto. También tendremos oportunidad de precisar los medios con los que se cuentan para el presente desarrollo y la gestión de los mismos.

Por último, veremos los mecanismos de control y de seguimiento a adoptar para asegurar la mínima desviación en el desarrollo.

2.3.1 ENUMERACIÓN DE RIESGOS

Presentaremos los riesgos con formato tabular, siguiendo la siguiente especificación de detalles:

- **Identificador**: designará un código de identificación único para el riesgo.
- **Descripción**: designará una descripción breve del riesgo.
- **Probabilidad de ocurrir**: designará la probabilidad de la ocurrencia del riesgo contemplado. Podrá tener los valores: muy alto, alto, medio, bajo y muy bajo.
- **Impacto**: designará el efecto sobre la desviación de la planificación, que puede provocar la ocurrencia de dicho riesgo. Podrá tener los valores: severo, medio, bajo y muy bajo.
- **Mecanismos asociados**: designará uno o varios identificadores de mecanismos de control o de contingencia a aplicar, para sufragar el impacto en la desviación, ante la ocurrencia del riesgo.

Veamos a continuación los riesgos contemplados:

Identificador	R-001
Descripción	Falta de tiempo para el desarrollo a causa de desviarlo para la realización de otro proyecto. En este caso se contempla la creación real de una empresa dedicada a sistemas informáticos y al desarrollo e implantación de aplicaciones software orientada a la PYME
Probabilidad de ocurrir	Muy Alta
Impacto	Severo
Mecanismos asociados	M-001

Tabla 1. *Riesgo R-001*

Identificador	R-002
Descripción	Inhabilidad para el desarrollo del proyecto en caso de enfermedad psicológica o física. Incluimos en este riesgo el estrés y la fatiga.
Probabilidad de ocurrir	Media
Impacto	Severo
Mecanismos asociados	M-002, M-003

Tabla 2. *Riesgo R-002*

Identificador	R-003
Descripción	Falta de conocimientos técnicos o teóricos para la aplicación de las tecnologías designadas para el desarrollo.
Probabilidad de ocurrir	Media
Impacto	Severo
Mecanismos asociados	M-004, M-005

Tabla 3. *Riesgo R-003*

Identificador	R-004
Descripción	Falta de acceso a recursos materiales y lógicos necesarios para el desarrollo del proyecto.
Probabilidad de ocurrir	Baja
Impacto	Severo
Mecanismos asociados	M-006

Tabla 4. *Riesgo R-004*

Identificador	R-005
Descripción	Ignorancia sobre el tamaño real de la aplicación a desarrollar y del resultado del resto de implicados en el desarrollo del conjunto del proyecto CERTILOC.
Probabilidad de ocurrir	Alta
Impacto	Medio
Mecanismos asociados	M-007

Tabla 5. *Riesgo R-005*

Identificador	R-006
Descripción	Riesgo derivado de la implementación o especificación precarias o tempranas de las tecnologías designadas para el desarrollo del proyecto.
Probabilidad de ocurrir	Muy Alta
Impacto	Severo
Mecanismos asociados	M-008

Tabla 6.*Riesgo R-006*

2.3.2 MECANISMOS DE CONTROL PARA LA MINIMIZACIÓN DE RIESGOS

Tal y como exige MÉTRICA V3 (Métrica V.3 - Consejo Superior de Administración Electrónica 2009), en este apartado se determinan los mecanismos de control necesarios para minimizar los efectos de los riesgos.

Los mecanismos se presentarán de forma tabular, siguiendo la siguiente especificación de detalles:

- **Identificador**: designará un código de identificación único para el mecanismo.
- **Descripción**: designará una descripción breve del mecanismo.
- **Dificultad**: designará el grado de dificultad que implica la ejecución del mecanismo. Podrá tener los valores: **Alta, media, baja.**
- **Esfuerzo**: designará el grado de esfuerzo que implica la ejecución del mecanismo. Podrá tener los valores: **Alto, medio, bajo.**

Identificador	M-001
Descripción	Programación de jornadas diarias teniendo en cuenta la asignación de tiempo para el desarrollo del proyecto. Este tiempo se aplicará siempre en tiempos libres, fuera del horario laboral, a no ser que se encuentren momentos ociosos dentro de dicho horario.
Dificultad	Alta
Esfuerzo	Alto

Tabla 7.*Mecanismo de control M-001*

Identificador	M-002
Descripción	Programación de jornadas diarias teniendo en cuenta el descanso y el tiempo libre.
Dificultad	Media
Esfuerzo	Bajo

Tabla 8.*Mecanismo de control M-002*

Identificador	M-003
Descripción	Acudir a especialistas médicos para el tratamiento de enfermedades.
Dificultad	Baja
Esfuerzo	Medio

Tabla 9.*Mecanismo de control M-003*

Identificador	M-004
Descripción	Recurrir a la ayuda de terceros, orientados a la formación.
Dificultad	Baja
Esfuerzo	Bajo

Tabla 10. *Mecanismo de control M-004*

Identificador	M-005
Descripción	Obtención de especificaciones técnicas acerca de las tecnologías a utilizar. Lectura y acceso a documentos técnicos, incluyendo foros de internet y páginas web en general.
Dificultad	Medio
Esfuerzo	Alto

Tabla 11. *Mecanismo de control M-005*

Identificador	M-006
Descripción	Realización periódica de copias de seguridad y almacenamiento de las mismas en distintas ubicaciones y medios físicos.
Dificultad	Medio
Esfuerzo	Medio

Tabla 12. *Mecanismo de control M-006*

Identificador	M-007
Descripción	Mantener un contacto fluido con los responsables y resto de desarrolladores del proyecto CERTILOC.
Dificultad	Alto
Esfuerzo	Medio

Tabla 13. *Mecanismo de control M-007*

Identificador	M-008
Descripción	Contacto directo con los desarrolladores de tecnologías o especificaciones técnicas.
Dificultad	Alto
Esfuerzo	Medio

Tabla 14. *Mecanismo de control M-008*

2.3.3 GESTIÓN Y ENUMERACIÓN DE MEDIOS Y RECURSOS

Se describen a continuación y de una manera detallada, los medios y recursos que se utilizan para el desarrollo del proyecto.

Cada uno de estos medios o recursos contendrá las siguientes propiedades:

- **Identificador**: designará un código de identificación único para el recurso o medio.
- **Descripción**: designará una descripción breve del recurso o medio.

Identificador	MR-001
Descripción	Ordenador portátil HP Compaq NX 8220 con las siguientes especificaciones: • Procesador Intel (R) Pentium (R) M 1, 73 GHz • 512 MB de memoria RAM • Disco duro de 60 GB • Microsoft Office Project Professional 2007 instalado • Microsoft Office Visio Professional 2007 instalado • Ubuntu 6.06 LTS instalado • Microsoft Windows XP Professional Service Pack 3 instalado • Suite Microsoft Office 2007 instalado • Exadel Studio eclipse IDE instalado • Omondo UML for eclipse 3.2 instalado • MySQL Server 5 instalado • MySQL Navigator instalado • MySQL Admin instalado

Tabla 15. *Medio o recurso MR-001*

Identificador	MR-002
Descripción	Disco duro portátil Toshiba USB 2.0 40 GB

Tabla 16. *Medio o recurso MR-002*

Identificador	MR-003
Descripción	Servidor de almacenamiento clónico con las siguientes especificaciones: • Procesador Intel (R) Pentium 4(R) 2,55 GHz • 2 GB de memoria RAM • Disco duro IDE de 150 GB • Disco duro SATA de 300 GB • Microsoft Windows Server 2003 R2 SP2 instalado • Servicio "Compartir Archivos e Impresoras" instalado y funcionando correctamente • Servicio de "Terminal Server" instalado y funcionando correctamente

Tabla 17. *Medio o recurso MR-003*

Identificador	MR-004
Descripción	Servidor Goofy: Servidor web dedicado, perteneciente a la universidad Carlos III de Madrid. Contiene las siguientes especificaciones, entre otras: • Ubuntu Server 6.06 • Servidor Web Apache instalado y funcionando correctamente • Servidor de Bases de datos MySQL 5 instalado y funcionando correctamente

Tabla 18. *Medio o recurso MR-004*

Identificador	MR-005
Descripción	Conexión a Internet de banda ancha ADSL con conectividad total y derechos de administración.

Tabla 19. *Medio o recurso MR-005*

De momento se considera que estos recursos serán suficientes para el desarrollo del proyecto. En caso de surgir nuevas necesidades, este apartado se revisará y actualizará con la información de los nuevos medios y recursos a tener en cuenta.

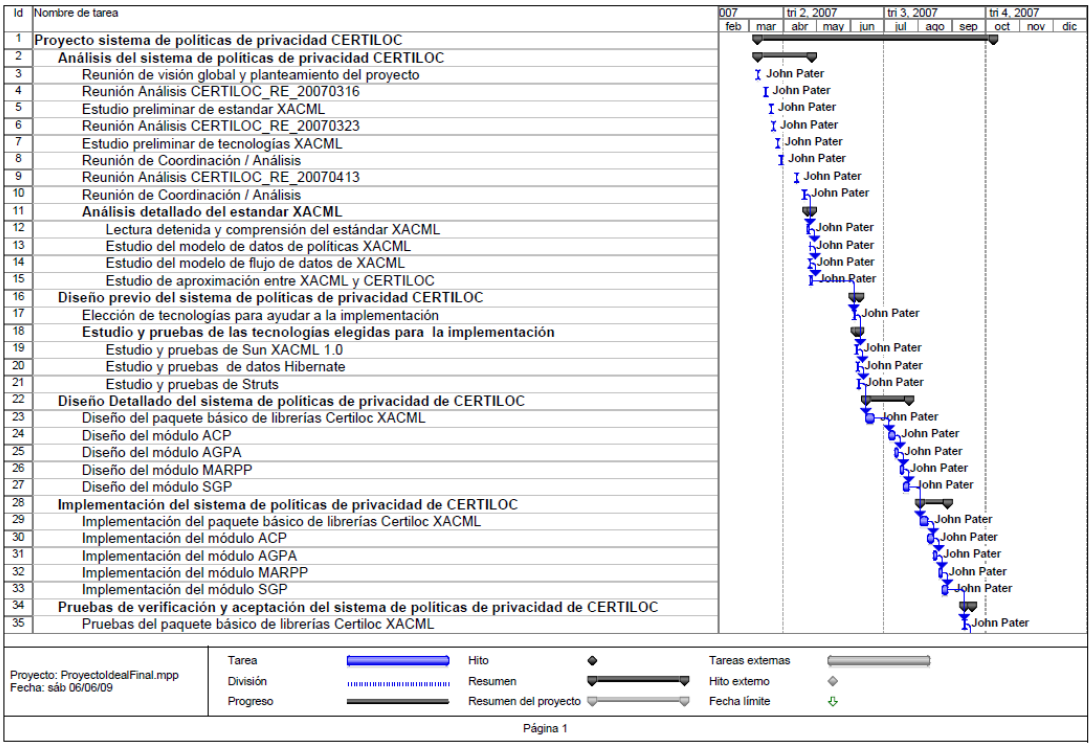
2.4 PLANIFICACIÓN DEL PROYECTO

En el presente apartado, veremos dos diagramas que representarán la planificación inicial para el desarrollo del proyecto y el seguimiento real del mismo.

Al final del presente apartado se incluye una conclusión sobre el ajuste de la realidad a la planificación y de las posibles causas ante desviaciones encontradas.

2.4.1 PLANIFICACIÓN INICIAL

Se presenta a continuación la planificación inicial para el desarrollo del proyecto.



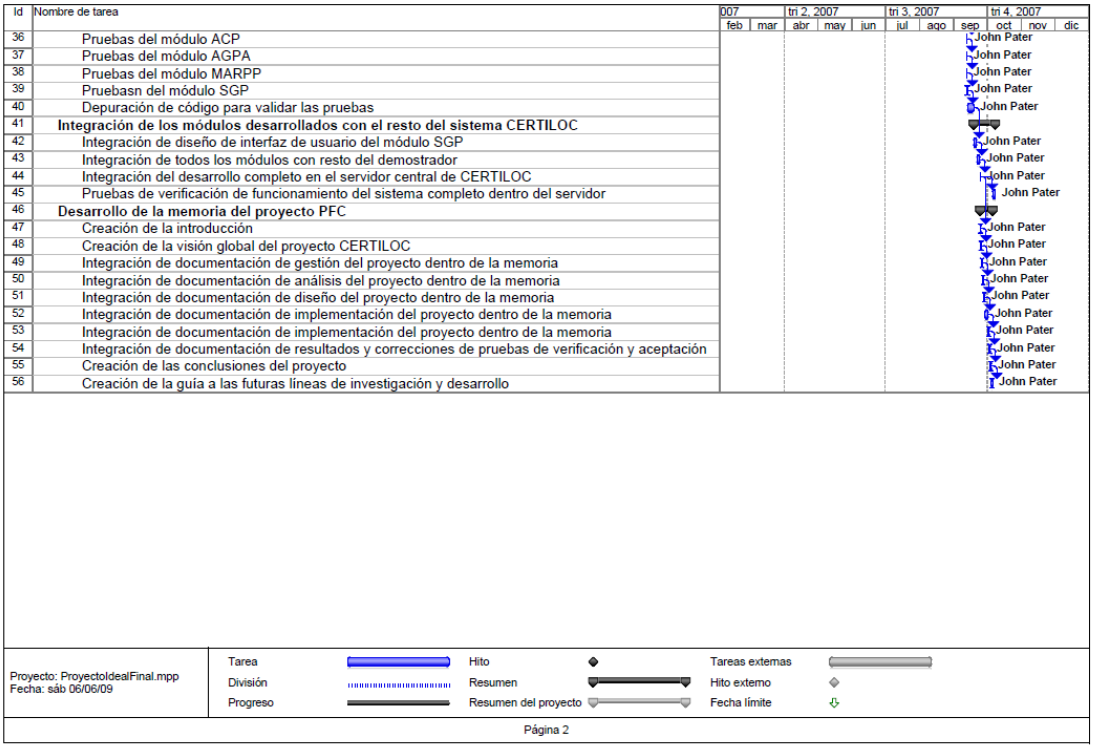


Figura 5. *Planificación inicial del proyecto*

Se ha planificado el proyecto con una estimación de unos 7,6 meses. Esta planificación es bastante optimista y da por sentado que se tendrán jornadas laborales de unas 3 horas

diarias, dedicadas íntegramente al proyecto, considerando 22 días laborables al mes. Al final tenemos un resultado de **502 horas de trabajo** repartidas en **168 días.**

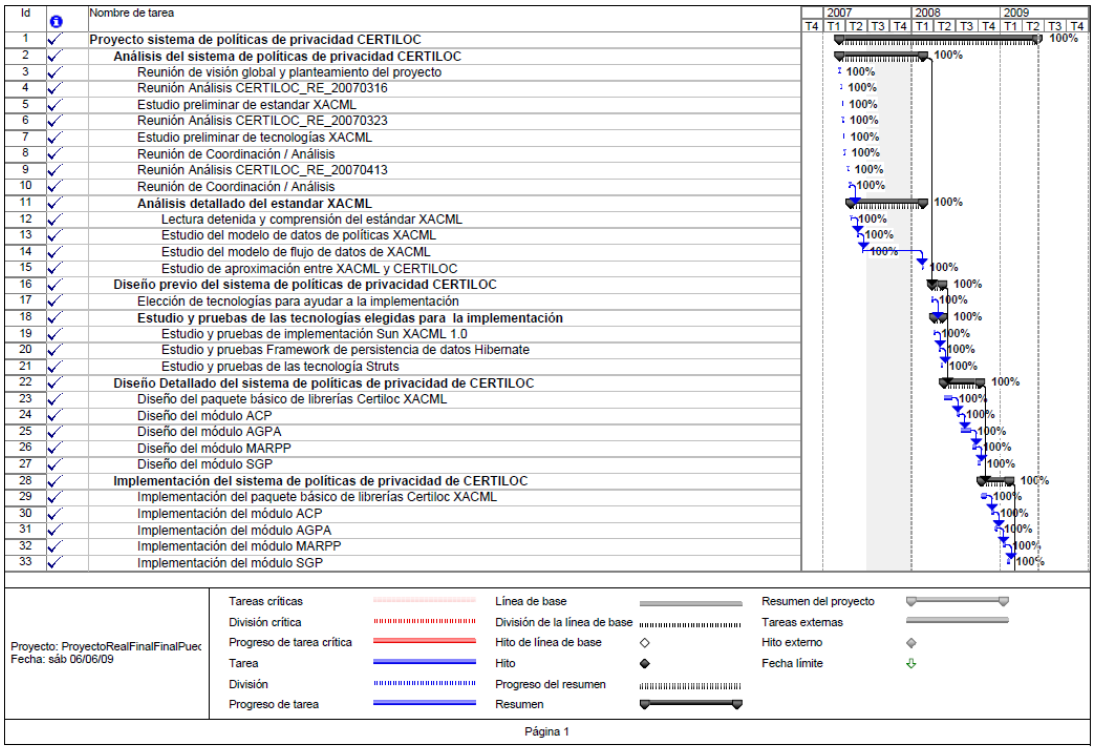
La siguiente tabla resume estos datos:

Actividad	Cantidad	Periodo
Análisis del sistema de políticas de privacidad CERTILOC	55,00	Horas
Diseño previo del sistema de políticas de privacidad CERTILOC	29,00	Horas
Diseño Detallado del sistema de políticas de privacidad de CERTILOC	136,00	Horas
Implementación del sistema de políticas de privacidad de CERTILOC	136,00	Horas
Pruebas de verificación y aceptación del sistema	40,00	Horas
Integración de los módulos desarrollados con el resto del sistema	34,00	Horas
Desarrollo de la memoria del proyecto PFC	72,00	Horas
Totales Desarrollo Horas	**502,00**	**Horas**
Total Desarrollo Días	**167,33**	**Días**
Total Desarrollo Meses	**7,61**	**Meses**

Tabla 20. *Resumen planificación inicial*

2.4.2 SEGUIMIENTO REAL

Se presenta a continuación el seguimiento real del proyecto desarrollado.



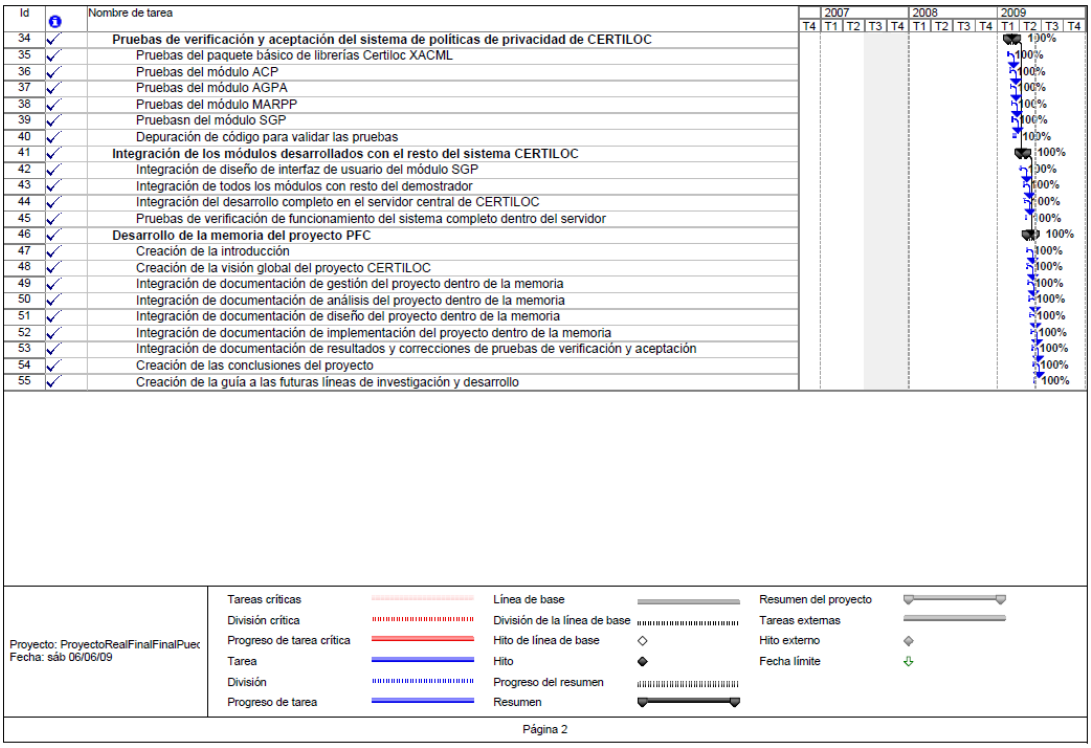


Figura 6. *Seguimiento real del proyecto*

La Figura 6 nos muestra un desarrollo real del proyecto de unos 27 meses y medio. Se ha considerado una media de jornada laboral de 1 hora diaria y 45 días libres. Se consideran 22 días laborables por mes. La siguiente tabla muestra un resumen de estos datos:

Actividad	Cantidad	Periodo
Análisis del sistema de políticas de privacidad CERTILOC	70,00	Horas
Diseño previo del sistema de políticas de privacidad CERTILOC	40,00	Horas
Diseño Detallado del sistema de políticas de privacidad de CERTILOC	150,00	Horas
Implementación del sistema de políticas de privacidad de CERTILOC	150,00	Horas
Pruebas de verificación y aceptación del sistema	40,00	Horas
Integración de los módulos desarrollados con el resto del sistema	34,00	Horas
Desarrollo de la memoria del proyecto PFC	80,00	Horas
Totales Desarrollo Horas	**564,00**	**Horas**
Total Desarrollo Días	**609,00**	**Días**
Total Desarrollo Meses	**27,68**	**Meses**

Tabla 21. *Resumen Seguimiento real*

Los datos mostrados en la tabla resumen (Tabla 21) nos revelan un resultado poco agradable. La planificación inicial se ha desviado completamente.

Veremos un detalle de la desviación en el siguiente apartado, junto con unas conclusiones generales.

2.4.3 COMPARATIVA Y CONCLUSIONES DE LA PLANIFICACIÓN Y EL SEGUIMIENTO

Se muestra a continuación una tabla que representa los datos de la desviación entre la planificación inicial y el seguimiento real. Más adelante sacaremos las veremos las conclusiones de los resultados obtenidos.

Actividad	Cantidad Estimada	Cantidad real	Periodo	Desviación
Análisis del sistema de políticas de privacidad CERTILOC	55,00	70,00	Horas	27,27%
Diseño previo del sistema de políticas de privacidad CERTILOC	29,00	40,00	Horas	37,93%
Diseño Detallado del sistema de políticas de privacidad de CERTILOC	136,00	150,00	Horas	10,29%
Implementación del sistema de políticas de privacidad de CERTILOC	136,00	150,00	Horas	10,29%
Pruebas de verificación y aceptación del sistema	40,00	40,00	Horas	0,00%
Integración de los módulos desarrollados con el resto del sistema	34,00	34,00	Horas	0,00%
Desarrollo de la memoria del proyecto PFC	72,00	80,00	Horas	11,11%
Totales Desarrollo en Horas	**502,00**	**564,00**	**Horas**	12,35%
Total Desarrollo en Días	**167,33**	**609,00**	**Días**	263,94%
Total Desarrollo en Meses	**7,61**	**27,68**	**Meses**	263,72%

Tabla 22. *Resumen de comparativa entre planificación inicial y seguimiento final*

Vemos que la mayor desviación está en el número de días estimados para el desarrollo. Sin embargo, el número de horas estimadas para cada tarea no ha sufrido tal desviación. Esto es porque el proyecto no se ha desviado en horas de desarrollo sino que se ha distribuido para realizarse en un periodo mayor de tiempo. Donde en la estimación inicial aplicábamos 3 horas diarias, íntegramente dedicadas al proyecto, en el seguimiento real hemos podido dedicar una media de una hora diaria por día laboral. Este hecho, multiplica por tres el número de días necesarios para la realización del proyecto. Si a esto le sumamos unos 45 días de vacaciones y unos 62 días de descanso, obtenemos el resultado final de **609 días de desarrollo**, con su consiguiente desvío de más del **250%**.

Evaluar el porqué de esta desviación tan grande no es una tarea complicada, pero reconocer el error en la planificación sí lo es. Los motivos principales para la desviación han sido varios (todos habían sido contemplados en los riesgos del proyecto) entre los que encontramos, ordenados por severidad e impacto en el proyecto:

- El principal motivo ha sido la **vida laboral**. A los tres meses de inicio del proyecto surgió la posibilidad de crear una empresa real dedicada a sistemas y desarrollos informáticos **(Riesgo R-001)**. El del autor del presente PFC fue, y sigue siendo, el de llevar a cabo la gerencia de la empresa. Teniendo en cuenta el caso de una empresa nueva, con una competencia alta, sin experiencia en la dirección de empresas (aunque sí en la vida laboral) y atravesando una crisis a nivel nacional e internacional, es lógico pensar que era imposible cumplir la planificación inicial y llevar a cabo el trabajo en la empresa. Se han aplicado los mecanismos de control asociados (**M-001**) a este riesgo pero no han sido suficientes para minimizar el impacto del riesgo.
- Secundariamente, se ha sufrido **estrés y fatiga** derivada de la actividad laboral **(Riesgo R-002)**. Se aplicaron los mecanismos de control definidos para minimizar el riesgo (**M-002 y M-003**) y ayudaron en gran medida a mitigarlo pero no evitaron la desviación.
- Además, las **tecnologías** a utilizar, y en particular **XACML**, tienen un uso poco extendido y, en consecuencia, una documentación muy precaria **(Riesgo R-003 y R-006)**. En este caso se ejecutaron sólo los mecanismos asociados al riesgo R-003. El mecanismo M-008, asociado al riesgo R-006 no dio los productos esperados (se pudo intentó contactar con los autores de XACML pero nunca se recibió una respuesta).
- Por último, se han sufrido algunas **complicaciones técnicas**. El equipo informático con el que se comenzó el desarrollo dejó de funcionar en el último semestre del proyecto y hubo que sustituirlo por uno nuevo, con la consiguiente restauración de copias de seguridad y reinstalación de aplicaciones **(Riesgo R-004)**. La ejecución del mecanismo de control asociado a este riesgo **(M-006)** fue crucial para minimizar su impacto.

La planificación inicial era demasiado optimista y ajustada con lo que se ha visto desviada de manera desmesurada.

La realización del presente PFC, ha ayudado a madurar al autor y también a tomar expectativas más reales y escépticas en proyectos, ya comercializados, en la empresa que se ha creado en el último año y medio.

2.5 COSTES Y PRESUPUESTO DEL PROYECTO

A continuación podremos ver una estimación de los costes derivados de la implementación del proyecto y el presupuesto estimado. Como es lógico entender esta estimación es una estimación simulada y, dado el carácter del proyecto, no se exigirá el pago de ninguno de los costes presentados a ninguna entidad pública o privada.

Cabe remarcar que el coste del proyecto se estima en euros (€) y viene derivado de la planificación y los recursos y medios físicos necesarios para la realización del mismo.

En este apartado, al igual que en el apartado de planificación se ha considerado un coste estimado derivado de la planificación inicial y un coste real derivado del seguimiento real del proyecto.

Sólo cabe comentar que se ha decidido relacionar el coste del proyecto con un valor monetario y no con un valor en esfuerzo para facilitar su lectura y comprensión.

El apartado se organiza de la siguiente manera. Primero se verá la estrategia seguida para la estimación de costes del proyecto. Más adelante veremos una estimación inicial de costes y posteriormente el presupuesto del proyecto. Se verá una pequeña aproximación del margen de beneficios esperado para el presente proyecto haciendo una comparación entre los costes y el presupuesto presentado al cliente. Por último, se verán los costes reales y, para terminar, se presentarán unas conclusiones sobre la desviación de costes.

2.5.1 ESTRATEGIA PARA EL CÁLCULO DE COSTES

Para poder enumerar los distintos costes del proyecto, se ha decidido dividirlos en tres grandes grupos: costes directos y costes indirectos. Los costes directos son aquellos que se derivan directamente de la implementación del proyecto, y los indirectos, los que se derivan de una manera indirecta.

Como costes directos podemos encontrar los siguientes: costes por RRHH, costes por HW y costes por SW.

Por otro lado, como costes indirectos podemos encontrar los siguientes: gastos en dietas, gastos de electricidad, gastos de transporte y gastos de comunicaciones (ADSL).

En general, el precio de cada elemento tangible (HW y SW) se considera **sin aplicar el 16% de IVA** ya que éste se devuelve anualmente a la empresa con lo que no supone un gasto.

2.5.1.1 Cálculo de costes de RRHH

Para calcular el coste en recursos humanos, hacemos una extrapolación del coste anual de un empleado con un rango profesional de titulado de grado superior hasta extraer el coste por hora de dicho empleado.

Para calcular el salario base, se ha considerado el convenio colectivo de empresas de consultoría y estudios de mercado (BOE 04-04-2009 2009). Según el ANEXO I de dicho

convenio, un Titulado Superior debe cobrar (cálculo del año 2007) **20.945,36 €/Año** en 14 pagas de 1.496,74 €. Además debe cobrar un plus de convenio equivalente a **1.465,24€/Año** en 14 pagas de 104,66 €. La suma nos da su salario base: **22.419,60€ por año (Salario Bruto)**.

Todos los salarios están expresados en importe bruto y no neto para facilitar el cálculo. Así mismo, para facilitar el coste por **seguros sociales** (recaen directamente sobre la empresa) que la empresa tendrá que pagar por el trabajador, éstos se consideran como **un tercio del salario bruto anual** (en general, los costes por seguros sociales oscilan entre un 32% y un 37% de la base de cotización por lo que un 33% del salario base bruto interanual nos da una aproximación bastante exacta del coste para la empresa).

Categoría profesional trabajador	Periodo de cálculo de coste	Salario Base	Seguros Sociales	Coste Total
Graduado Superior	Anual	22.419,60 €	7.473,20 €	29.892,80 €
Graduado Superior	Mensual (Anual/14)	1.601,40 €	622,77 €	2.224,17 €
Graduado Superior	Semanal (Mensual/4)	400,35 €	155,69 €	556,04 €
Graduado Superior	Diario (Semanal / 5)	80,07 €	31,14 €	111,21 €
Graduado Superior	Por Hora (Diario / 8)	10,01 €	3,89 €	**13,90 €**

Tabla 23. *Cálculo de costes de RRHH*

Por graduado superior se estima un coste de **13,90€** por hora de trabajo.

2.5.1.2 Cálculo de costes de HW

Para realizar una estimación de los costes de hardware se prorratean los precios iniciales de cada pieza de material, por el número de meses que durará el proyecto.

La siguiente tabla, se describe los periodos de vida útil para cada elemento. Esta tabla nos ayudará a estimar el coste prorrateado del uso de un determinado material durante un periodo de tiempo. La tabla muestra el coste por mes del material. Cabe remarcar que este coste es aproximado y pesimista ya que supone que cada pieza de hardware es utilizada exclusivamente en un proyecto. Para hacer un cálculo más realista, debería dividirse por el número de proyectos a los que se está aplicando. Aun así, este coste aproximado puede darnos una visión global del coste del proyecto.

Elemento	Coste inicial	Vida útil	Periodo	Coste por mes
Portátil HP Compaq NX 8220	1.200,00 €	24,00	Meses	**50,00 €**
Disco duro Toshiba USB 2.0 40 GB	60,00 €	48,00	Meses	**1,25 €**
Servidor de almacenamiento clónico	700,80 €	24,00	Meses	**29,20 €**
Portátil ASUS Z53JSeries (2ª Mano)	500,00 €	12,00	Meses	**41,67 €**

Tabla 24. *Cálculo de costes de HW*

2.5.1.3 Cálculo de costes de SW

Al igual que para los costes de HW, para realizar una estimación de los costes de hardware se prorratean los precios iniciales de cada pieza de material, por el número de meses que durará el proyecto.

Presentamos a continuación la tabla de cálculo de costes de SW asociada al presente proyecto. Esta tabla también elementos de libre distribución que tienen un coste asociado de 0 €.

Elemento	Coste inicial	Vida útil	Periodo	Coste por mes
Licencia Microsoft Office 2007	600,00 €	48,00	Meses	**12,50 €**
Licencia Microsoft Project 2007	600,00 €	48,00	Meses	**12,50 €**
Licencia Microsoft Visio 2007	600,00 €	48,00	Meses	**12,50 €**
Licencia Java Eclipse 3.2.2	0,00 €	48,00	Meses	**0,00 €**
Licencia Exadel Estudio	0,00 €	48,00	Meses	**0,00 €**
Licencia Omondo UML	0,00 €	48,00	Meses	**0,00 €**
Licencia Ubuntu Desktop 6.06 LTS	0,00 €	48,00	Meses	**0,00 €**

Tabla 25. *Cálculo de costes de SW*

2.5.1.4 Cálculo de costes Indirectos

Se presenta a continuación una tabla que expresa los costes indirectos por unidad, calculados para el presente proyecto.

Al igual que los costes por HW y SW, para hacer una aproximación más fina deberíamos dividir el coste de cada elemento por el número de proyectos que estén actualmente en desarrollo. Aun así, las cifras presentadas nos valen para hacer una estimación aproximada de los costes del presente proyecto.

Elemento	Coste por unidad	Unidad
Conexión a Internet de banda ancha ADSL	**52,00 €**	Mes
Luz y agua	**80,00 €**	Mes
Transporte	**1,08 €**	Km
Dietas y alimentos	**100,00 €**	Mes

Tabla 26. *Cálculo de costes indirectos*

2.5.2 ESTIMACIÓN INICIAL DE COSTES

Se presenta a continuación una tabla que contiene una estimación inicial de los costes del proyecto, utilizando para ello la estimación en tiempo de la planificación inicial y los cálculos de costes presentados en el apartado anterior.

Tipo de coste	Partida	Coste Unitario	Cantidad	Formato cantidad	Total
RRHH	Análisis del sistema de políticas de privacidad CERTILOC	13,90	55,00	Horas	764,56 €
RRHH	Diseño previo del sistema de políticas de privacidad CERTILOC	13,90	29,00	Horas	403,13 €
RRHH	Diseño Detallado del sistema de políticas de privacidad de CERTILOC	13,90	136,00	Horas	1.890,54 €
RRHH	Implementación del sistema de políticas de privacidad de CERTILOC	13,90	136,00	Horas	1.890,54 €
RRHH	Pruebas de verificación y aceptación del sistema	13,90	40,00	Horas	556,04 €
RRHH	Integración de los módulos desarrollados con el resto del sistema	13,90	34,00	Horas	472,64 €
RRHH	Desarrollo de la memoria del proyecto PFC	13,90	72,00	Horas	1.000,88 €
SW	Licencia Microsoft Office 2007	12,50	7,61	Meses	95,08 €
SW	Licencia Microsoft Project 2007	12,50	7,61	Meses	95,08 €
SW	Licencia Microsoft Visio 2007	12,50	7,61	Meses	95,08 €
SW	Licencia Java Eclipse 3.2.2	0,00	7,61	Meses	0,00 €
SW	Licencia Exadel Estudio	0,00	7,61	Meses	0,00 €
SW	Licencia Omondo UML	0,00	7,61	Meses	0,00 €
SW	Licencia Ubuntu Desktop 6.06 LTS	0,00	7,61	Meses	0,00 €
SW	Apache Tomcat Web Server	0,00	1,00	Licencia	0,00 €
SW	MySQL Server	0,00	1,00	Licencia	0,00 €
SW	Licencia Ubuntu Server LTS	0,00	1,00	Licencia	0,00 €
HW	Ordenador portátil HP Compaq NX 8220	50,00	7,61	Meses	380,30 €
HW	Disco duro portátil Toshiba USB 2.0 40 GB	1,25	7,61	Meses	9,51 €

Tipo de coste	Partida	Coste Unitario	Cantidad	Formato cantidad	Total
HW	Servidor de almacenamiento clónico	29,20	7,61	Meses	222,10 €
HW	Servidor web Goofy	900,00	1,00	Unidad	900,00 €
Indirecto	Conexión a Internet de banda ancha ADSL	52,00	7,61	Meses	395,52 €
Indirecto	Electricidad	80,00	7,61	Meses	608,48 €
Indirecto	Transporte	1,08	100,00	Km	108,00 €
Indirecto	Dietas y alimentos	100,00	7,61	Meses	760,61 €
RRHH	Instalación y configuración de Servidor Goofy	13,90	10,00	Horas	139,01 €
Total					**10.787,07 €**

Tabla 27. *Estimación de costes para el proyecto*

La tabla anterior nos muestra una estimación de unos costes que ascienden a **10.787,07€** para la realización completa del proyecto en unos siete meses y medio.

No se ha considerado coste alguno en concepto de dirección y coordinación del proyecto ya que la misma ha corrido a cargo del cliente (en este caso la tutora del presente PFC). No todos estos costes son imputables al cliente. Los costes no imputables, son costes de los que no se debe hacer cargo directamente al cliente en el presupuesto que se le presenta. Se consideran los siguientes costes como costes no imputables:

Tipo de coste	Partida	Coste Unitario	Cantidad	Formato cantidad	Total
SW	Licencia Microsoft Office 2007	12,50	7,61	Meses	95,08 €
SW	Licencia Microsoft Project 2007	12,50	7,61	Meses	95,08 €
SW	Licencia Microsoft Visio 2007	12,50	7,61	Meses	95,08 €
SW	Licencia Java Eclipse 3.2.2	0,00	7,61	Meses	0,00 €
SW	Licencia Exadel Estudio	0,00	7,61	Meses	0,00 €
SW	Licencia Omondo UML	0,00	7,61	Meses	0,00 €
SW	Licencia Ubuntu Desktop 6.06 LTS	0,00	7,61	Meses	0,00 €
HW	Ordenador portátil HP Compaq NX 8220	50,00	7,61	Meses	380,30 €
HW	Disco duro portátil Toshiba USB 2.0 40 GB	1,25	7,61	Meses	9,51 €
HW	Servidor de almacenamiento clónico	29,20	7,61	Meses	222,10 €
Indirecto	Conexión a Internet de banda ancha ADSL	52,00	7,61	Meses	395,52 €
Indirecto	Electricidad	80,00	7,61	Meses	608,48 €
Indirecto	Dietas y alimentos	100,00	7,61	Meses	760,61 €
Total					**2661,76 €**

Tabla 28. *Costes no imputables al cliente*

2.5.3 PRESUPUESTO DEL PROYECTO

Se presenta a continuación el presupuesto completo del proyecto. Este presupuesto sería el que presentaríamos al potencial cliente.

Para la presentación y cálculo del presupuesto real, se han tomado las siguientes consideraciones.

Se han considerado distintos roles de trabajador para cada fase del proyecto. En este caso, se considera que cada rol tiene un precio por hora distinto. De esta manera, podremos obtener un margen de beneficios mejor y podremos hacer una mejor negociación con el cliente. Así mismo, podremos crear un presupuesto más holgado que pueda hacer frente, de mejor manera, a desviaciones en el proyecto.

Presentamos a continuación la tabla de presupuestos para cada rol de desarrollo.

Rol	Descripción	Coste/Hora
Analista	Encargado de realizar el análisis de la aplicación	55,00 €
Ingeniero de software	Encargado de desarrollar la aplicación	30,00 €
Ingeniero de pruebas	Encargado de realizar y ejecutar el plan de pruebas de sistema	30,00 €
Operador de sistemas	**Encargado de los sistemas que soportarán la aplicación**	**20,00 €**

Tabla 29. *Presupuesto por roles de desarrollo*

Por otro lado, se incluye el 15% del total del presupuesto de desarrollo, en concepto de coordinación y dirección del proyecto.

Por último se ha considerado una posible instalación del sistema informático que soportará la aplicación.

Partida y conceptos presupuestados	Rol trabajador	Precio Unitario	Cantidad	Formato cantidad	Total €
Análisis del sistema de políticas de privacidad CERTILOC	Analista	55,00 €	55	Horas	3.025,00 €
Diseño previo del sistema de políticas de privacidad CERTILOC	Ingeniero de software	30,00 €	29	Horas	870,00 €
Diseño Detallado del sistema de políticas de privacidad de CERTILOC	Ingeniero de software	30,00 €	136	Horas	4.080,00 €
Implementación del sistema de políticas de privacidad de CERTILOC	Ingeniero de software	30,00 €	136	Horas	4.080,00 €
Pruebas de verificación y aceptación del sistema	Ingeniero de pruebas	30,00 €	40	Horas	1.200,00 €
Integración de los módulos desarrollados con el resto del sistema		30,00 €	34	Horas	1.020,00 €

Partida y conceptos presupuestados	Rol trabajador	Precio Unitario	Cantidad	Formato cantidad	Total €
Desarrollo de la Documentación del Proyecto	Ingeniero de software	30,00 €	72	Horas	2.160,00 €
Dirección y coordinación del proyecto	Director del proyecto	-	15%	Total Horas	2.465,25 €
Instalación y configuración de Servidor Goofy	Operador de sistemas	20,00 €	10,00	Horas	200,00 €
Apache Tomcat Web Server	-	0,00 €	1,00	Licencia	0,00 €
MySQL Server	-	0,00 €	1,00	Licencia	0,00 €
Licencia Ubuntu Server LTS	-	0,00 €	1,00	Licencia	0,00 €
Servidor web Goofy	-	1.232,88 €	1,00	Unidad	1.232,88 €
Transporte	-	1,48 €	100,00	Km	147,95 €
Total	-	-	-	-	20.481,07 €
IVA	-	-	-	-	3.276,97 €
Total + IVA	**-**	**-**	**-**	**-**	**23.758,04 €**

Tabla 30. *Presupuesto de la aplicación*

Tal y como se observa en la Tabla 30, el presupuesto total de la aplicación asciende a **23.758,04 euros**. En el siguiente apartado haremos una estimación de los beneficios derivados del desarrollo de la aplicación.

2.5.4 ESTIMACIÓN DE BENEFICIOS

Para hacer una estimación de beneficios se comparan la estimación de costes frente al presupuesto del cliente.

El cálculo del margen de beneficio está hecho sobre el precio de venta. Es decir, el porcentaje de beneficio derivado de la partida comparada, se obtiene considerando que el 100% es el precio de venta y no el precio de compra o coste. Cabe hacer una parada en este punto para explicar esto detalladamente.

Pondremos un sencillo ejemplo que nos ayude a comparar la diferencia del cálculo de margen según se haga desde el precio de venta o desde el precio de compra o de coste. En el siguiente ejemplo, queremos obtener un **margen de beneficios del 10%** de un **producto** que compramos o nos **cuesta a 1000** €. Si hacemos el cálculo del beneficio desde el precio de compra, obtendremos un **beneficio de 100 €** (equivalente al 10% de 1000). Sin embargo, haciendo el cálculo desde el punto de vista del precio de compra, el **margen de beneficios** es de **9,1 %** y **no del 10 %** que queríamos inicialmente. Esto se comprueba con una simple regla de tres: Si 1100 es el 100%, 1000 es el X%. Donde X sería igual a 90,90%. De ahí que el margen de beneficios sea **100% - 90,90% = 9,10%**. El cálculo del porcentaje de beneficio, en términos

comerciales, debe hacerse siempre desde el punto de vista del precio de venta y no del precio de compra. Para averiguar el porcentaje de beneficio de aplicamos la siguiente fórmula:

$$\%\ Margen\ de\ beneficio = 100\% - \left(Precio\ de\ coste * 100/_{Precio\ de\ venta}\right)$$

Por otro lado, si queremos averiguar el precio de venta que debemos dar a un producto para obtener un determinado margen de beneficios lo haremos con la siguiente fórmula:

$$Precio\ de\ venta = \frac{Precio\ de\ coste}{\left[1 - \left(\%\ Margen\ de\ beneficio/_{100}\right)\right]}$$

En el caso de **costes no imputables al cliente** habría que calcular el porcentaje beneficio como una pérdida directa. Para hacer el cálculo del margen de pérdida, obtenemos el porcentaje que representa la cantidad total no imputable al cliente frente a la cantidad de coste total.

Hecha esta aclaración pasemos a ver la estimación de beneficios.

Partida y conceptos presupuestados	Precio Unitario	Coste Unitario	Cantidad	Formato cantidad	Total Presupuestado	Total Coste	Margen de Beneficios
Análisis del sistema de políticas de privacidad CERTILOC	55,00 €	13,90 €	55	Horas	3.025,00 €	764,56 €	74,73%
Diseño previo del sistema de políticas de privacidad CERTILOC	30,00 €	13,90 €	29	Horas	870,00 €	403,13 €	53,66%
Diseño Detallado del sistema de políticas de privacidad de CERTILOC	30,00 €	13,90 €	136	Horas	4.080,00 €	1.890,54 €	53,66%
Implementación del sistema de políticas de privacidad de CERTILOC	30,00 €	13,90 €	136	Horas	4.080,00 €	1.890,54 €	53,66%
Pruebas de verificación y aceptación del sistema	30,00 €	13,90 €	40	Horas	1.200,00 €	556,04 €	53,66%
Integración de los módulos desarrollados con el resto del sistema	30,00 €	13,90 €	34	Horas	1.020,00 €	472,64 €	53,66%
Desarrollo de la Documentación del Proyecto	30,00 €	13,90 €	72	Horas	2.160,00 €	1.000,88 €	53,66%
Dirección y coordinación del proyecto	-		0,15	Total Horas	2.465,25 €	0 €	100%
Apache Tomcat Web Server	0,00 €	0,00 €	1	Licencia	0,00 €	0,00 €	0,00%
MySQL Server	0,00 €	0,00 €	1	Licencia	0,00 €	0,00 €	0,00%
Licencia Ubuntu Server LTS	0,00 €	0,00 €	1	Licencia	0,00 €	0,00 €	0,00%
Servidor web Goofy	1.232,88 €	900,00 €	1	Unidad	1.232,88 €	900,00 €	27,00%
Transporte	1,48 €	1,08 €	100	Km	147,95 €	108,00 €	27,00%
Instalación y configuración de Servidor Goofy	20,00 €	13,90 €	10	Horas	200,00 €	139,01 €	30,49%
Costes no Imputables al cliente			1		0,00 €	2.661,74 €	-24,68%
Total					**20.481,07 €**	**10.787,07 €**	**47,33%**

Tabla 31. *Estimación de beneficios derivados del desarrollo del proyecto*

Tal y como vemos en la Tabla 31, el margen de beneficios es del **47,33%**. En general podemos decir que es un margen alto. Un margen tan alto nos permite una desviación grande entre la estimación y la realidad. En el caso concreto de este proyecto, es un margen necesario ya que el proyecto utiliza tecnologías desconocidas cuyo uso puede desviarlo en gran medida.

2.5.5 COSTES REALES DEL PROYECTO

En el siguiente apartado se presentan los costes reales del proyecto. El cálculo de costes reales está hecho con los mismos datos que la estimación pero aplicando, esta vez, los

datos de tiempo de desarrollo obtenidos del seguimiento real del proyecto. Se muestra a continuación un resumen de la desviación en costes:

Partida	Coste Unit.	Cant. Estim	Cant. Real	Form. Cant.	Total Estim.	Total Real	Desv. Coste
Análisis del sistema de políticas de privacidad CERTILOC	13,90 €	55,00	70,00	Horas	764,56 €	973,07 €	**27,27%**
Diseño previo del sistema de políticas de privacidad CERTILOC	13,90 €	29,00	40,00	Horas	403,13 €	556,04 €	**37,93%**
Diseño Detallado del sistema de políticas de privacidad de CERTILOC	13,90 €	136,00	150,00	Horas	1.890,54 €	2.085,16 €	**10,29%**
Implementación del sistema de políticas de privacidad de CERTILOC	13,90 €	136,00	150,00	Horas	1.890,54 €	2.085,16 €	**10,29%**
Pruebas de verificación y aceptación del sistema	13,90 €	40,00	40,00	Horas	556,04 €	556,04 €	**0,00%**
Integración de los módulos desarrollados con el resto del sistema	13,90 €	34,00	34,00	Horas	472,64 €	472,64 €	**0,00%**
Desarrollo de la memoria del proyecto PFC	13,90 €	72,00	80,00	Horas	1.000,88 €	1.112,08 €	**11,11%**
Licencia Microsoft Office 2007	12,50 €	7,61	27,68	Meses	95,08 €	346,02 €	**263,94%**
Licencia Microsoft Project 2007	12,50 €	7,61	27,68	Meses	95,08 €	346,02 €	**263,94%**
Licencia Microsoft Visio 2007	12,50 €	7,61	27,68	Meses	95,08 €	346,02 €	**263,94%**
Licencia Java Eclipse 3.2.2	0,00 €	7,61	27,68	Meses	0,00 €	0,00 €	**0,00%**
Licencia Exadel Estudio	0,00 €	7,61	27,68	Meses	0,00 €	0,00 €	**0,00%**
Licencia Omondo UML	0,00 €	7,61	27,68	Meses	0,00 €	0,00 €	**0,00%**
Licencia Ubuntu Desktop 6.06 LTS	0,00 €	7,61	27,68	Meses	0,00 €	0,00 €	**0,00%**
Apache Tomcat Web Server	0,00 €	1,00	1,00	Lic.	0,00 €	0,00 €	**0,00%**
MySQL Server	0,00 €	1,00	1,00	Lic.	0,00 €	0,00 €	**0,00%**
Licencia Ubuntu Server LTS	0,00 €	1,00	1,00	Lic.	0,00 €	0,00 €	**0,00%**
Portátil ASUS Z53JSeries (2ª Mano)	41,67 €	0,00	55,00	Meses	0,00 €	208,33 €	**100,00%**
Ordenador portátil HP Compaq NX	50,00 €	7,61	27,68	Meses	380,30 €	1.384,09 €	**263,94%**
Disco duro portátil Toshiba USB 2.0 40 GB	1,25 €	7,61	27,68	Meses	9,51 €	34,60 €	**263,94%**
Servidor de almacenamiento clónico	29,20 €	7,61	27,68	Meses	222,10 €	808,31 €	**263,94%**
Servidor web Goofy	900,00 €	1,00	1,00	Unidad	900,00 €	900,00 €	**0,00%**
Conexión a Internet de banda ancha ADSL	52,00 €	7,61	27,68	Meses	395,52 €	1.439,45 €	**263,94%**
Electricidad	80,00 €	7,61	27,68	Meses	608,48 €	2.214,55 €	**263,94%**
Transporte	1,08 €	100,00	100,00	Km	108,00 €	108,00 €	**0,00%**
Dietas y alimentos	100,00 €	7,61	27,68	Meses	760,61 €	2.768,18 €	**263,94%**
Instalación y configuración de Servidor Goofy	13,90 €	10,00	10,00	Horas	139,01 €	139,01 €	**0,00%**
Totales					**10.787,07 €**	**18.674,45 €**	**73,12%**

Tabla 32. *Resumen de costes reales y desviación por costes*

Tal y como nos muestra la tabla anterior hay una **desviación de más del 70%** entre los costes estimados y los costes reales.

A continuación mostramos los datos sobre la desviación de los beneficios.

Partida y conceptos presupuestados	Total Presupuestado	Total Coste Real	Margen Esperado	Margen Real
Análisis del sistema de políticas de privacidad CERTILOC	3.025,00 €	973,07 €	74,73%	67,83%
Diseño previo del sistema de políticas de privacidad CERTILOC	870,00 €	556,04 €	53,66%	36,09%
Diseño Detallado del sistema de políticas de privacidad de CERTILOC	4.080,00 €	2.085,16 €	53,66%	48,89%
Implementación del sistema de políticas de privacidad de CERTILOC	4.080,00 €	2.085,16 €	53,66%	48,89%
Pruebas de verificación y aceptación del sistema	1.200,00 €	556,04 €	53,66%	53,66%
Integración de los módulos desarrollados con el resto del sistema	1.020,00 €	472,64 €	53,66%	53,66%
Desarrollo de la Documentación del Proyecto	2.160,00 €	1.112,08 €	53,66%	48,51%
Dirección y coordinación del proyecto	2.465,25 €	0,00 €	100 %	100 %
Apache Tomcat Web Server	0,00 €	0,00 €	0,00%	0,00%
MySQL Server	0,00 €	0,00 €	0,00%	0,00%
Licencia Ubuntu Server LTS	0,00 €	0,00 €	0,00%	0,00%
Servidor web Goofy	1.232,88 €	900,00 €	27,00%	27,00%
Transporte	147,95 €	108,00 €	27,00%	27,00%
Instalación y configuración de Servidor Goofy	200,00 €	139,01 €	30,49%	30,49%
Costes no Imputables al cliente	-	10.003,59 €	**-24,68%**	-52,68%
Total	**20.481,07 €**	**18.990,78 €**	**47,33%**	**7,28%**

Tabla 33. *Resumen de beneficios reales obtenidos*

Los datos sobre los márgenes de beneficio no son muy alentadores pero tampoco desoladores. Aunque vemos que todavía resta un 7% de beneficios, en realidad estaríamos hablando de unas pérdidas de aproximadamente 40% (47,33% - 7,28% = 40,05%), siempre que consideremos como pérdida el dinero que no hayamos podido ganar.

Veamos las causas de esta desviación:

- **Desviación en tiempo**: Al aumentar el tiempo del desarrollo, también han aumentado los costes, sobre todo los no imputables al cliente (HW, SW y costes indirectos).
- **Desviación por tecnología**: Para finalizar el proyecto se tuvo que comprar un nuevo equipo portátil de segunda mano. Esta compra ha aumentado, una vez más los costes no imputables al cliente.

La desviación de coste no es achacable al “cliente” o tutor del proyecto ya que no han surgido requisitos ocultos sino que las circunstancias del desarrollador no han permitido cumplir la estimación de costes inicial. En la vida real, al vender desarrollos, no se puede aplicar costes a los clientes en caso de no existir un cambio en los requisitos iniciales con lo que el impacto para la empresa, en este caso, habría sido grande. En caso que la empresa hubiese invertido dinero en algún otro proyecto, contando con un más de 8% del beneficio inicial estimado para este proyecto, estaría sufriendo pérdidas reales.

3 ANÁLISIS DEL PROYECTO

Presentamos en el siguiente apartado el análisis completo del sistema de políticas de privacidad de CERTILOC.

Podremos ver, además de una breve introducción, una descripción general del sistema de políticas de privacidad de CERTILOC, los diagramas de casos de uso del SPP, los requisitos recabados para el mismo, un análisis de las tecnologías a utilizar para el sistema destacando el análisis del estándar XACML y su adaptación a CERTILOC y por último, haremos una especificación del plan de pruebas de aceptación que nos permitirá evaluar si el sistema implementado cumple con la definición de todos los requisitos.

3.1 INTRODUCCIÓN

El análisis que se muestra a continuación, es el resultado de las reuniones de especificación mantenidas con la tutora del presente proyecto, la investigación personal del autor y la información extraída del documento "**Especificación de los Requisitos y análisis preliminar del sistema CERTILOC**" (Gonzalez-Tablas, y otros 2007).

En primer lugar se describen las características generales del sistema de políticas de privacidad de CERTILOC.

En segundo lugar, se pueden observar la especificación de actores del SPP y los casos de uso definidos para cada uno de ellos.

Más adelante, tal y como marca la metodología METRICA V3 (Métrica V.3 - Consejo Superior de Administración Electrónica 2009) se exponen los requisitos de software funcionales, no funcionales e inversos recabados para el desarrollo del sistema objeto del presente PFC.

Una vez vistos los casos de uso y requisitos definidos para el sistema, haremos un análisis de las tecnologías que se deben utilizar para el desarrollo del proyecto. Las tecnologías presentadas en esta fase del desarrollo, se resumen a las tecnologías a utilizar para el diseño y desarrollo del sistema de políticas de privacidad (SPP), y más en particular para su adecuación al estándar XACML.

Por último, se presenta una especificación preliminar del plan de pruebas del sistema. Este plan de pruebas, permitirá comprobar que el sistema desarrollado cumple correctamente con todos los requisitos y casos de uso definidos.

3.2 CARACTERÍSTICAS GENERALES DEL SISTEMA DE POLÍTICAS DE PRIVACIDAD DE CERTILOC

El sistema de políticas de privacidad o **SPP** tiene los siguientes objetivos:

- **Evaluar** si se debe permitir o denegar una determinada petición de autorización ante las políticas de privacidad activas en el sistema.
- Ofrecer un **repositorio** persistente para las políticas de privacidad definidas por los usuarios.
- **Proporcionar** a los usuarios una manera de gestionar sus políticas de privacidad.
- Permitir hacer un **seguimiento de la actividad** del sistema.

El SPP estará compuesto por tres partes bien diferenciadas.

Por un lado tendremos los servicios que se prestan directamente a los usuarios de CERTILOC a través de la interfaz de usuario Web:

- Hacer un seguimiento de la actividad de autorización relacionada con sus dispositivos y políticas de privacidad, incluyendo la petición de autorización que generó la actividad.
- Gestionar sus políticas de privacidad para crear, borrar, activar, desactivar o modificar sus políticas de privacidad.

Por otro lado tendremos los servicios que se prestan a otros módulos del demostrador de CERTILOC, en particular al módulo Agente de Autenticación y Autorización de Básica (**AAAB**):

- Evaluar la autorización de una determinada petición de autorización, contra las políticas de privacidad activas en el sistema
- Por último, tendremos que cumplir con las propuestas del estándar XACML:
- Tendremos que traducir entre el lenguaje nativo de CERTILOC para peticiones de autorización, respuestas de autorización y políticas de privacidad al formato del lenguaje XACML y viceversa
- Tendremos que cumplir, en la medida de lo posible, la arquitectura y el flujo de datos planteado por este estándar.

Las políticas de privacidad y peticiones de autorización de CERTILOC, deben poder especificar parámetros para los siguientes conceptos:

- El **rol** bajo el que actúa el solicitante de la petición de autorización. Tal y como se vio en el apartado 1.1.3 del presente documento, el rol del usuario solicitante se relaciona directamente con la finalidad para la que el solicitante utilizará la información obtenida. La solicitud de localización espacio-temporal de un dispositivo, ya sea de manera inmediata o de manera diferida, puede tener distintas motivaciones según la persona o entidad que está solicitando dicha información. Por ejemplo el rol entre un alumno y su director del proyecto de fin de carrera podría ser el de **tutor**. CERTILOC contemplará la definición de distintos roles entre usuarios de la aplicación (jefe, tutor u otros que quiera crear el usuario).
- El **identificador único** del solicitante de la petición de autorización, es decir, del nombre de usuario CERTILOC del usuario solicitante del servicio. Tal y como se vimos en el apartado 1.1.3, cada usuario de CERTILOC estará unívocamente identificado en el sistema por su identificador de usuario.
- El **dispositivo** sobre el que se realiza el servicio (la acción) que genera la petición de autorización. Todos los dispositivos en CERTILOC tienen un identificador único que los diferenciará del resto por lo que se pueden identificar de forma unívoca. Las acciones de localización (obtención de IET) siempre se realizarán sobre un dispositivo concreto existente en el sistema CERTILOC.
- La **acción** a realizar en la petición de autorización. Tal y como vimos en el apartado 1.1.3 CERTILOC permite a sus usuarios poder llevar a cabo distintas acciones o servicios: **obtención de IET, generación**, **descarga** y **eliminación** de **CETs**. El demostrador de CERTILOC debe permitir especificar la acción que se quiere realizar en el servicio solicitado en las políticas de privacidad, y que se incluya en las peticiones de autorización.
- La **IET** concreta del dispositivo sobre el que se realiza el servicio solicitado. Los usuarios de CERTILOC deben poder especificar en sus políticas de privacidad parámetros sobre la localización concreta de sus dispositivos así como del momento en que se realiza la recogida de datos de localización. De esta manera, un usuario responsable de un dispositivo, deberá poder especificar políticas de privacidad para indicar, por ejemplo, qué sólo quiere que se pueda localizar el dispositivo si se localiza entre las ocho de la mañana y las 11 de la noche, y sólo si el dispositivo se encuentra dentro de la Comunidad de Madrid.
- Por último, las políticas de privacidad deben poder tener en cuenta el **momento de ejecución de la petición** de autorización, es decir el momento en que se quiere disponer de cierta IET. Éste momento se relaciona con el momento exacto en el que se ejecuta la petición de acceso, es decir, cuando llega al agente custodio de las políticas de privacidad o ACP. En el caso de **peticiones diferidas**, CERTILOC va a permitir crear solicitudes de servicios de

localización de manera diferida. Es decir, peticiones donde se desea **acceder a la IET** de un dispositivo en un **determinado momento**, y se desean **consultar** los datos recogidos en el momento de la petición, en otro **momento posterior**.

Por otro lado, el SPP, debe permitir especificar **funciones o condiciones** concretas que utilicen los atributos presentados en el anterior listado. Es decir, debe existir la posibilidad de definir funciones condicionales que devuelvan verdadero o falso y determinen el resultado de la autorización. Por ejemplo, un usuario responsable de dispositivo puede permitir localizar uno de sus dispositivos sólo con la condición de que la petición de localización se ejecute entre las 8:00 y las 20:00 horas, o bien si el dispositivo se encuentra en una determinada localización espacio en el momento de obtener su IET.

Además de implementar los servicios descritos anteriormente, el sistema deberá registrar los distintos **eventos del sistema** (relacionado con el caso de uso CU-ADM-001) y los **eventos de aplicación**, es decir, de la actividad de las políticas de privacidad (relacionado con el caso de uso CU-RD-006).

El seguimiento de los eventos de sistema permitirá hacer un seguimiento de los eventos de sistema a lo largo del ciclo de vida del **SPP** y estará orientado a su revisión por parte de **Usuarios Administradores** de CERTILOC.

El seguimiento de la aplicación permitirá, a los **Usuarios Responsables de Dispositivo,** hacer un seguimiento exhaustivo de la manera en la que la aplicación maneja su información privada y personal y el funcionamiento de sus políticas de privacidad contra las distintas peticiones de autorización que llegan al sistema.

El sistema de políticas de privacidad de CERTILOC se dividirá en los siguientes subsistemas o módulos. Como podemos observar, cada uno tiene una razón de ser bien diferenciada:

- **AGPA**: Será el encargado de la conversión entre distintos contextos o lenguajes. El actual proyecto refleja dos contextos distintos para los datos de las políticas de privacidad: El contexto o formato de XACML y el contexto o formato de CERTILOC.
- **ACP:** Encargado de la evaluación de peticiones de autorización contra las distintas políticas de privacidad activas, definidas en el sistema.
- **MARPP**: Responsable de custodiar el repositorio de políticas de privacidad (RPP). Se encarga de la lectura y escritura de datos persistentes.

- **SGP:** Este sistema ofrece, a los usuarios finales de CERTILOC, la capacidad gestionar sus políticas de privacidad y acceder al seguimiento de las mismas, mediante una interfaz web.

3.3 ESPECIFICACIÓN DE CASOS DE USO

Se presentan a continuación los casos de uso definidos para el sistema de políticas de privacidad de CERTILOC. Hay que remarcar que en el presente PFC sólo se ha implementado el sistema de políticas de privacidad. Por lo tanto, en el presente apartado se presentan, única y exclusivamente, los casos de uso asociados con esta porción del sistema.

Si el lector desea obtener la especificación completa de casos de uso del conjunto del proyecto CERTILOC, debe ponerse en contacto con el Departamento de Informática de la Escuela Politécnica Superior de la Universidad Carlos III de Madrid (González-Tablas Ferreres, Ana Isabel) y solicitar el acceso a la documentación del resto de proyectos orientados a la implementación del demostrador de CERTILOC (Memoria PFC - de Fuentes García-Romero de Tejada 2007) (Memoria PFC - Calvo Martínez 2007) (Memoria PFC - Gallo Martínez 2008).

3.3.1 ESPECIFICACIÓN DE LOS ACTORES DEL SISTEMA

Dentro del marco del sistema de políticas de privacidad, se define un único actor que denominaremos “Usuario CERTILOC”. Este usuario principal, está subdividido en tres tipos de usuarios: “Responsable Dispositivo”, “Administrador del sistema” y “Solicitante de servicios” cuyas responsabilidades y funciones son distintas. Dependiendo de la acción que el usuario quiera llevar a cabo dentro de la aplicación, podrá acceder a la misma con distintos roles.

En la Figura 7 se muestra un diagrama de la jerarquía expuesta en el párrafo anterior.

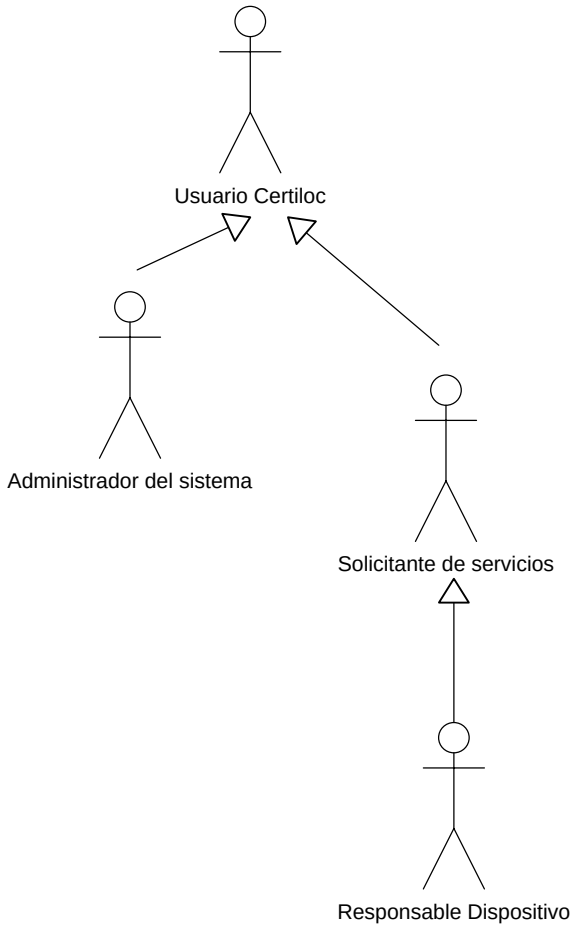


Figura 7. *Jerarquía de actores del sistema de políticas de privacidad* CERTILOC

A continuación pasamos a describir detalladamente cada uno de los actores presentados en el diagrama anterior:

- **Usuario** CERTILOC será cualquier actor que potencialmente interactúe con el sistema CERTILOC. Un "usuario CERTILOC" podrá realizar distintas funciones sobre el sistema dependiendo de su naturaleza y de los privilegios que tenga asociados.
- **Administrador del sistema:** será el actor encargado de administrar el sistema. Este actor realizará tareas de administración sobre el mismo, pudiendo llevar a cabo funciones tales como la gestión de los datos del sistema, la administración de los usuarios o visualizar eventos del sistema.
- **Responsable Dispositivo:** será aquel actor que posea, o sea responsable (valga la redundancia) de un dispositivo localizable. Por supuesto, dicho dispositivo debe estar relacionado con el sistema CERTILOC. Este actor podrá definir políticas de privacidad en cuanto al acceso a determinados servicios del sistema relacionados con su dispositivo. Además, este

actor debe tener acceso completo a los servicios asociados a los dispositivos de los que sea responsable directo.

- **Solicitante de servicios:** será aquel actor que solicite servicios de localización y de certificación, relacionados con dispositivos asociados al sistema CERTILOC. Por lo general, este actor llevará a cabo funciones de solicitud de localización espacio-temporal de dispositivos, generación de certificados de localización espacio-temporal, descarga de certificados de localización espacio temporal y otras.

En los siguientes apartados, veremos detalladamente los casos de uso asociados a cada uno de estos actores.

3.3.2 FORMATO PARA LA ESPECIFICACIÓN DE CASOS DE USO

Para presentar los casos de uso de los distintos actores del sistema de políticas de privacidad utilizaremos una tabla con información del caso de uso. Una vez presentados los casos de uso, se presentará un diagrama que representará una imagen visual descriptiva del actor implicado y la acción que éste realiza.

Se presenta a continuación la tabla a utilizar y la especificación de los detalles que contendrá.

Identificador	Identificador único para el caso de uso mostrado
Nombre	Nombre descriptivo para el caso de uso
Descripción	Descripción de los distintos pasos del caso de uso en cuestión.
Actor	Actor implicado en el caso de uso
Objetivos	Objetivo principal y secundarios del actor, al llevar a cabo el caso de uso
Condiciones	Especificación de las condiciones que influyen en el caso de uso presentado.

Tabla 34. *Formato de Tabla para la especificación de casos de uso*

Se presentan, a continuación, los casos de uso del sistema de políticas de privacidad para cada actor definido.

3.3.3 CASOS DE USO DEL ACTOR ADMINISTRADOR DEL SISTEMA

La Figura 8 muestra un resumen de los casos de uso del actor Administrador del sistema.

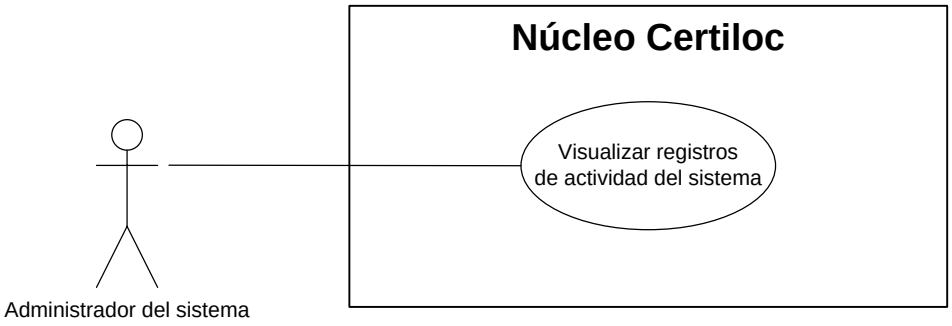


Figura 8. *Diagrama de casos del actor "Administrador del sistema"*

Identificador	CU-ADM-001
Nombre	Visualizar registros de actividad del sistema de políticas de privacidad
Descripción	• Un usuario accede al sistema como administrador • Presiona el botón "Acceder al histórico de Eventos del sistema de políticas de privacidad" • El sistema muestra un listado con los distintos ficheros de registro de eventos del sistema de políticas de privacidad • El usuario administrador pincha sobre uno de los ficheros y accede a descargarlo
Actor	Administrador del Sistema
Objetivos	• Proporcionar al usuario administrador una manera de acceder a los registros de actividad del sistema de políticas de privacidad.
Condiciones	• Los registros de actividad presentados pueden presentarse como documentos que el actor podrá descargar del servidor. • Los registros mostrados no deben mostrar información confidencial propia del resto de usuarios de CERTILOC

Tabla 35. *Caso de uso CU-ADM-001*

3.3.4 CASOS DE USO DEL ACTOR RESPONSABLE DE DISPOSITIVO

La Figura 9 muestra un resumen de los casos de uso del usuario responsable de dispositivo que se muestran en forma tabular a continuación.

Para el actor "Responsable de dispositivo":

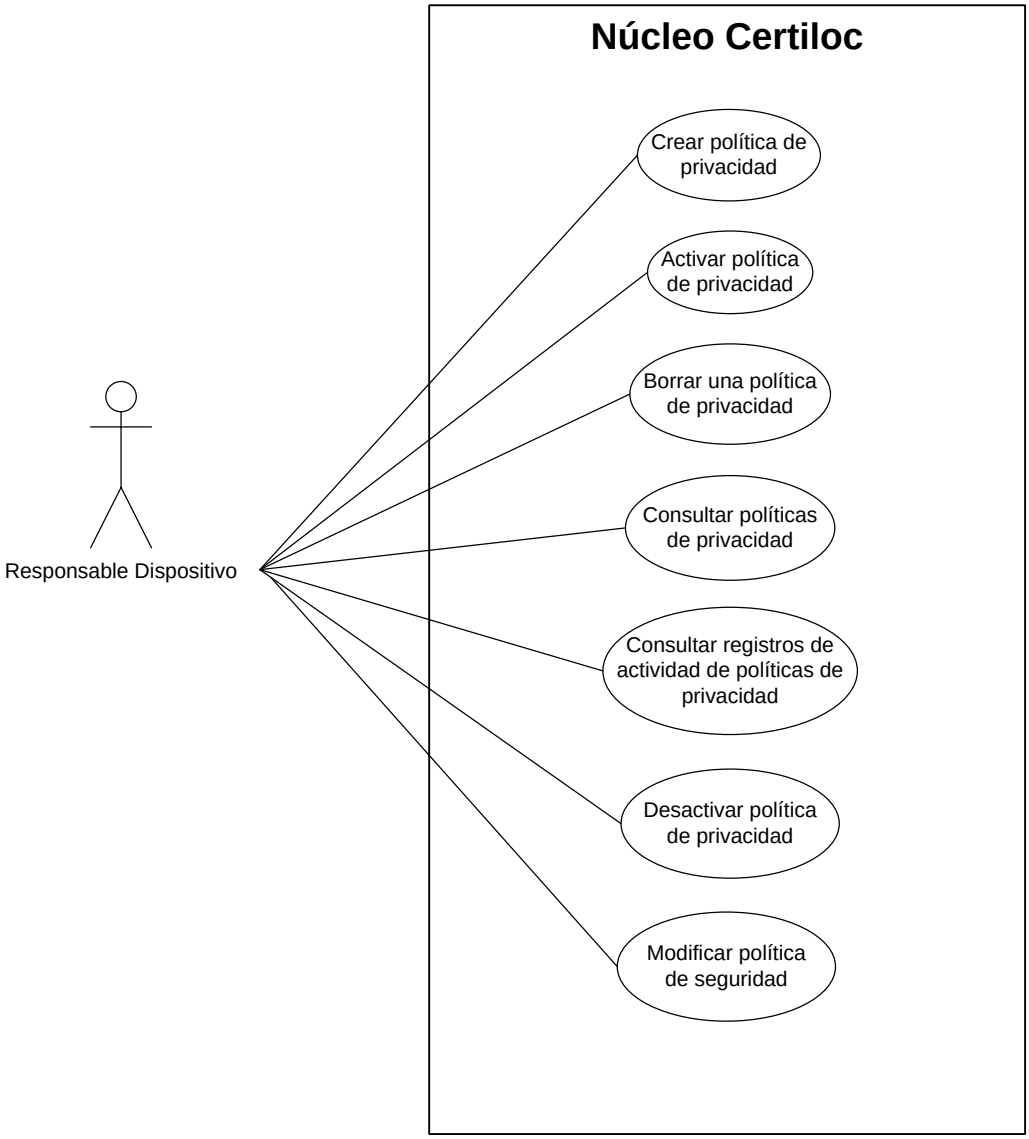


Figura 9. *Diagrama de casos del actor "Responsable Dispositivo"*

Identificador	CU-RD-001
Nombre	**Crear** política de privacidad
Descripción	• El actor entra en el sistema como usuario normal • Presiona el botón de acceso a la administración de políticas de privacidad • El usuario accede a sus políticas de privacidad presionando el botón preparado para tal acceso • Presiona el botón nuevo • El sistema muestra un formulario que el usuario debe rellenar • El usuario rellena los datos necesarios y presiona el botón guardar
Actor	Responsable Dispositivo
Objetivos	• Dar de alta una política de privacidad en el sistema que pase a interactuar contra peticiones de autorización.
Condiciones	• El usuario debe ser el propietario o responsable de la información relacionada con el objetivo de la política

Tabla 36. *Caso de uso CU-RD-001*

Identificador	CU-RD-002
Nombre	**Activar** política de privacidad
Descripción	• El actor entra en el sistema como usuario • Se dirige a la zona de administración de políticas de privacidad • El sistema muestra un listado de todas las políticas de privacidad del usuario • El usuario presiona sobre una política de privacidad inactiva en el sistema • El usuario presiona sobre el botón editar • El usuario cambia su estado a activa • El usuario presiona el botón guardar
Actor	Responsable Dispositivo
Objetivos	• Activar una determinada política de privacidad para que se empiece a aplicar ante peticiones relacionadas con el dispositivo relacionado.
Condiciones	• La política de privacidad debe pertenecer al actor que está activándola. • La política que se quiere activar debe estar previamente creada en el sistema.

Tabla 37. *Caso de uso CU-RD-002*

Identificador	CU-RD-003
Nombre	**Desactivar** política de privacidad
Descripción	• El actor entra en el sistema como usuario • Se dirige a la zona de administración de políticas de privacidad • El sistema muestra un listado de todas las políticas de privacidad del usuario • El usuario presiona sobre una política de privacidad activa en el sistema • El usuario presiona sobre el botón editar • El usuario cambia su estado a inactiva • El usuario presiona el botón guardar
Actor	Responsable Dispositivo
Objetivos	• Desactivar una determinada política de privacidad para que deje de aplicarse ante peticiones de autorización.
Condiciones	• La política de privacidad debe pertenecer al actor que está desactivándola. • La política que se quiere desactivar debe estar previamente creada en el sistema y encontrarse en estado activo.

Tabla 38. *Caso de uso CU-RD-003*

Identificador	CU-RD-004
Nombre	**Borrar** una política de privacidad
Descripción	• El actor entra en el sistema como usuario • Se dirige a la zona de administración de políticas de privacidad • El sistema muestra un listado de todas las políticas de privacidad del usuario • El usuario presiona sobre una política de privacidad • El usuario presiona sobre el botón Borrar
Actor	Responsable Dispositivo
Objetivos	• Borrar una determinada política de privacidad para que desaparezca del sistema.
Condiciones	• La política de privacidad debe pertenecer al actor que está borrándola. • La política que se quiere borrar debe estar previamente creada en el sistema.

Tabla 39. *Caso de uso CU-RD-004*

Identificador	CU-RD-005
Nombre	Consultar políticas de privacidad
Descripción	• El actor entra en el sistema como usuario • Se dirige a la zona de administración de políticas de privacidad • El sistema muestra un listado de todas las políticas de privacidad del usuario • El usuario presiona sobre una política de privacidad activa en el sistema navegando • El sistema le muestra los datos de esa política y un botón para volver al inicio de la raíz del árbol de políticas de privacidad del usuario
Actor	Responsable Dispositivo
Objetivos	• Ver sus políticas de privacidad
Condiciones	• Las políticas de privacidad deben pertenecer al actor que está visualizándolas.

Tabla 40. *Caso de uso CU-RD-005*

Identificador	CU-RD-006
Nombre	Consultar registros de actividad de políticas de privacidad
Descripción	• El actor entra en el sistema como usuario • Se dirige a la zona de administración de políticas de privacidad • El usuario presiona el botón "Acceder a registros de actividad" • El sistema muestra un listado de todos los registros de actividad relacionados con la información del usuario • El usuario presiona sobre un registro de actividad concreto • El sistema le muestra los datos de ese registro y un botón para volver al inicio de la raíz del árbol de registros de actividad
Actor	Responsable Dispositivo
Objetivos	• Mostrar al actor la actividad generada en el sistema, con origen en peticiones de autorización, y relacionada con sus dispositivos y políticas de privacidad.
Condiciones	• El actor sólo podrá visualizar los registros de actividad que vengan determinados por la actividad de sus políticas de privacidad y las peticiones recibidas cuyo objetivo era uno de sus dispositivos y que tuviera una política de privacidad relacionada y activa en el momento de la petición.

Tabla 41. *Caso de uso CU-RD-006*

Identificador	CU-RD-007
Nombre	**Modificar** política de privacidad
Descripción	• El actor entra en el sistema como usuario normal • Presiona el botón de acceso a la administración de políticas de privacidad • El usuario accede a sus políticas de privacidad presionando el botón preparado para tal acceso • Presiona sobre cualquiera de sus políticas de privacidad • El sistema muestra un formulario con los datos actuales de la política y listos para ser modificados • El usuario rellena los datos necesarios y presiona el botón guardar
Actor	Responsable Dispositivo
Objetivos	• Modificar una política de privacidad en el sistema.
Condiciones	• El usuario debe ser el propietario o responsable de la información relacionada con el objetivo de la política • La política a modificar debe estar creada en el sistema

Tabla 42. *Caso de uso CU-RD-007*

3.3.5 CASOS DE USO DEL ACTOR SOLICITANTE DE SERVICIOS

La Figura 10 muestra un resumen de los casos de uso definidos para el actor solicitante de servicios.

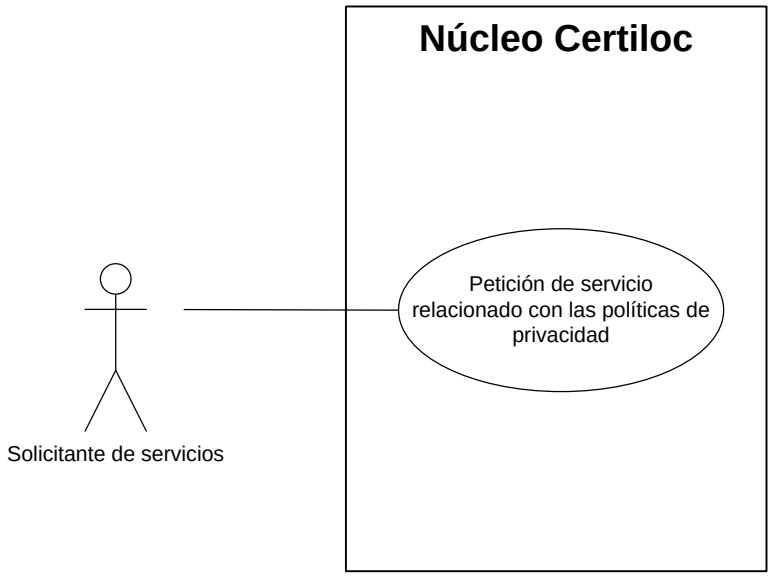


Figura 10. *Diagrama de casos del actor "Solicitante de servicios"*

Identificador	CU-SS-001
Nombre	Petición de servicio relacionado con las políticas de privacidad
Descripción	• El actor entra en el sistema como usuario • El usuario se dirige a la zona de peticiones de localización y servicios del sistema • El usuario rellena los datos necesarios para solicitar el uso de un servicio que se rige por el sistema de políticas de privacidad (Servicios ObtenerIET, DescargarCET o reanudar petición diferida) • El sistema decide, mediante las políticas de privacidad, autorizar o denegar la petición del usuario
Actor	Solicitante de servicios
Objetivos	• Aplicar las políticas de privacidad, dadas de alta y activadas en el sistema, para dar una respuesta de autorización o denegación ante la solicitud realizada.
Condiciones	• El usuario que solicita la autorización debe estar dado de alta en el sistema CERTILOC.

Tabla 43. *Caso de uso CU-SS-001*

3.4 REQUISITOS DE SOFTWARE

Se presentan en los siguientes apartados los requisitos recabados para el desarrollo del sistema de políticas de privacidad.

Tal y como propone Métrica V3 (Métrica V.3 - Consejo Superior de Administración Electrónica 2009), los requisitos están categorizados en funcionales y no funcionales. Además, se incluye un grupo de requisitos inversos para acabar de definir el sistema:

- **Requisitos funcionales**: Son los requisitos que definen la funcionalidad requerida para el proyecto desde el punto de vista del software y están divididos en requisitos de información y requisitos de operación. Los requisitos de información detallarán los datos concretos que manejará la aplicación y los operacionales indicarán las operaciones que se puedan llevar a cabo sobre los datos definidos.
- **Requisitos no funcionales**: Son requisitos que debe cumplir la implementación del sistema pero que no aportan funcionalidad a la misma. Los dividiremos en requisitos de rendimiento, requisitos operacionales, requisitos de seguridad, requisitos de mantenimiento, requisitos de interfaz, requisitos de recursos, requisitos de verificación, requisitos de aceptación, requisitos de documentación, requisitos de portabilidad, requisitos de calidad y requisitos de entrega.

- **Requisitos inversos**: Estos requisitos definen lo que la aplicación no debe realizar. Indirectamente, estos requisitos nos permiten definir completamente la aplicación ya que detallan concretamente lo que no debe implementarse.

3.4.1 FORMATO PARA LA ESPECIFICACIÓN DE REQUISITOS

Presentamos a continuación el formato que seguirán todos los requisitos mostrados en el documento.

Todos los identificadores de requisitos irán precedidos por "CERTILOC-PP-", para indicar que los requisitos pertenecen al sistema de políticas de privacidad del proyecto CERTILOC.

Cada uno de los requisitos presentados tendrá forma tabular y contendrá los siguientes datos:

Identificador	Identificador único del requisito en cuestión
Nombre	Nombre descriptivo para el requisito
Descripción	Descripción detallada del requisito
Prioridad	Prioridad de la implementación de este requisito frente a otros. Puede tener los valores Alta, Media o Baja.
Necesidad	Indica la necesidad, desde el punto de vista del cliente, de la implementación del requisito en cuestión. Puede tener los valores: Esencial, Deseable u Opcional.
Fuente	Indica la fuente de donde se extrae el requisito.
Pruebas de validación	Nos indica las pruebas que validarán el requisito en cuestión

Tabla 44. *Tabla de formato para la especificación de requisitos*

3.4.2 REQUISITOS FUNCIONALES

Se definen los siguientes requisitos funcionales para el sistema de políticas de privacidad.

3.4.2.1 Requisitos de información

Identificador	CERTILOC-PP-RFI-001					
Nombre	Políticas XACML					
Descripción	Las políticas de privacidad en CERTILOC seguirán el modelo de políticas del estándar XACML.					
Prioridad	Alta	☒	Media	☐	Baja	☐
Necesidad	Esencial	☒	Deseable	☐	Opcional	☐
Fuente	Creadores de CERTILOC					
Pruebas de validación	No Aplicable					

Tabla 45. *Requisito CERTILOC-PP-RFI-001*

Identificador	CERTILOC-PP-RFI-002					
Nombre	Peticiones de autorización recibidas desde otros módulos de CERTILOC					
Descripción	Las peticiones de autorización que se reciben desde otros módulos de CERTILOC hacia el ACP y que se evaluarán contra las políticas de privacidad del sistema, contienen los siguientes elementos: • **id del usuario** que solicita la petición • **información espacio temporal** del recurso objeto de la petición a localizar en el momento de la creación de la petición • **dispositivo** del que se solicita la petición • **acción** que se quiere realizar sobre el recurso de la petición • **petición con dispositivo original** en caso de peticiones diferidas • **petición a reasumir** en caso de peticiones diferidas • **certificado** involucrado en la petición en caso de peticiones relacionadas con un certificado El sistema ACP estará encargado de completar la información de la hora de creación de la petición y del rol del usuario que solicita la petición contra el usuario responsable del dispositivo para el cual se solicita.					
Prioridad	Alta	☒	Media	☐	Baja	☐
Necesidad	Esencial	☒	Deseable	☐	Opcional	☐
Fuente	Creadores de CERTILOC					
Pruebas de validación	No Aplicable					

Tabla 46. *Requisito CERTILOC-PP-RFI-002*

Identificador	CERTILOC-PP-RFI-003					
Nombre	Atributos de políticas CERTILOC					
Descripción	Los usuarios responsables de algún dispositivo de localización del sistema de políticas de privacidad deben poder especificar los siguientes datos en sus políticas: • **id** del usuario que solicita la petición • **rol** del usuario que solicita la petición • **acción** que se desear realizar sobre el recurso al que se solicita el acceso (**Obtener IET, Descargar CET, Generar CET, Borrar CET**) • **parámetros de designación de localización** o IET del dispositivo a localizar en el momento de la creación de la petición • **identificador del dispositivo** del que se solicita la petición • **hora** de la ejecución de la petición de autorización • **fecha** de la ejecución de la petición de autorización • **hora** del momento de la petición de la IET (este parámetro vale para peticiones diferidas) • **fecha** del momento de la petición de la IET (este parámetro vale para peticiones diferidas)					
Prioridad	Alta	☒	Media	☐	Baja	☐
Necesidad	Esencial	☒	Deseable	☐	Opcional	☐
Fuente	Creadores de CERTILOC					
Pruebas de validación	CERTILOC-PP-PA-004					

Tabla 47. *Requisito CERTILOC-PP-RFI-003*

Identificador	CERTILOC-PP-RFI-004					
Nombre	Respuestas de autorización de CERTILOC					
Descripción	Las respuestas a peticiones de autorización que devuelva el sistema de políticas de privacidad (SPP) contendrán los siguientes datos: • **respuesta** de autorización (Permitir o no Permitir) • **obligaciones** a cumplir en cuanto al uso de la información obtenida ante la respuesta a la solicitud de autorización					
Prioridad	Alta	☒	Media	☐	Baja	☐
Necesidad	Esencial	☒	Deseable	☐	Opcional	☐
Fuente	Creadores de CERTILOC					
Pruebas de validación	No Aplicable					

Tabla 48. *Requisito CERTILOC-PP-RFI-004*

Identificador	CERTILOC-PP-RFI-005					
Nombre	Registros de actividad de políticas de privacidad					
Descripción	Los registros de actividad de políticas de privacidad tendrán los siguientes atributos: • Petición que se ha recibido • Respuesta devuelta por el sistema • Política de privacidad que ha provocado la respuesta • Fecha y hora de la creación del registro • Usuario propietario del registro					
Prioridad	Alta	☒	Media	☐	Baja	☐
Necesidad	Esencial	☒	Deseable	☐	Opcional	☐
Fuente	Tutora del proyecto / creadores de CERTILOC					
Pruebas de validación	No Aplicable					

Tabla 49. *Requisito CERTILOC-PP-RFI-005*

Identificador	CERTILOC-PP-RFI-006					
Nombre	Estado de políticas de privacidad					
Descripción	• Los usuarios responsables de dispositivos deben poder definir el estado de las políticas de privacidad para reflejar si están activas o inactivas. • En caso que la política de privacidad esté en estado "activo", ésta se aplicará ante posibles peticiones de autorización. • En caso que la política de privacidad esté en estado "inactivo", ésta no se aplicará ante posibles peticiones de autorización.					
Prioridad	Alta	☒	Media	☐	Baja	☐
Necesidad	Esencial	☒	Deseable	☐	Opcional	☐
Fuente	Tutora del proyecto / creadores de CERTILOC					
Pruebas de validación	CERTILOC-PP-PA-006					

Tabla 50. *Requisito CERTILOC-PP-RFI-006*

3.4.2.2 Requisitos de operación

Identificador	CERTILOC-PP-RFO-001					
Nombre	Crear políticas de privacidad					
Descripción	• Los usuarios de CERTILOC que sean responsables de un dispositivo, deben poder crear políticas de privacidad para controlar las interacciones del resto de usuarios con sus elementos (dispositivos o certificados).					
Prioridad	Alta	☒	Media	☐	Baja	☐
Necesidad	Esencial	☒	Deseable	☐	Opcional	☐
Fuente	Creadores de CERTILOC					
Pruebas de validación	CERTILOC-PP-PA-001					

Tabla 51. *Requisito CERTILOC-PP-RFO-001*

Identificador	CERTILOC-PP-RFO-002					
Nombre	Resultado políticas de privacidad					
Descripción	• Las políticas de privacidad deben devolver una respuesta de autorización que sea Permitir o Denegar para indicar que se permite o se deniega la operación que se intenta llevar a cabo. Esta respuesta de autorización puede ir acompañada de unas obligaciones determinadas que se deben cumplir al utilizar la información obtenida mediante la respuesta.					
Prioridad	Alta	☒	Media	☐	Baja	☐
Necesidad	Esencial	☒	Deseable	☐	Opcional	☐
Fuente	Creadores de CERTILOC					
Pruebas de validación	CERTILOC-PP-PA-002, CERTILOC-PP-PA-003					

Tabla 52. *Requisito CERTILOC-PP-RFO-002*

Identificador	CERTILOC-PP-RFO-003					
Nombre	Activar y desactivar políticas de privacidad.					
Descripción	• Los usuarios que sean responsables de dispositivos deben poder cambiar el estado de sus políticas de privacidad de estado inactivo a activo y viceversa • Las políticas de privacidad con estado activo interactuarán con las peticiones de autorización. • Las políticas de privacidad con estado inactivo no interactuarán con las peticiones de autorización.					
Prioridad	Alta	☒	Media	☐	Baja	☐
Necesidad	Esencial	☒	Deseable	☐	Opcional	☐
Fuente	Creadores de CERTILOC					
Pruebas de validación	CERTILOC-PP-PA-006					

Tabla 53. *Requisito CERTILOC-PP-RFO-003*

Identificador	CERTILOC-PP-RFO-004
Nombre	Registros de actividad de políticas de privacidad
Descripción	• Cada vez que se recibe una petición de autorización, derivada de la solicitud de una petición de algún servicio que interactúa con las políticas activas en el sistema de políticas de privacidad, si la petición tiene relación con alguna política activa, se debe crear un registro de actividad que tenga los siguientes datos: ○ Petición que se ha recibido ○ Respuesta devuelta por el sistema ○ Política de privacidad que ha provocado la respuesta ○ Fecha y hora de la creación del registro
Prioridad	Alta ☒ Media ☐ Baja ☐
Necesidad	Esencial ☒ Deseable ☐ Opcional ☐
Fuente	Creadores de CERTILOC
Pruebas de validación	CERTILOC-PP-PA-005

Tabla 54. *Requisito CERTILOC-PP-RFO-004*

Identificador	CERTILOC-PP-RFO-005
Nombre	Registros de actividad de políticas de privacidad asociadas a usuario
Descripción	• Los usuarios que sean responsables de dispositivos deben poder revisar la actividad de sus políticas de privacidad mediante registros de actividad.
Prioridad	Alta ☒ Media ☐ Baja ☐
Necesidad	Esencial ☒ Deseable ☐ Opcional ☐
Fuente	Creadores de CERTILOC
Pruebas de validación	CERTILOC-PP-PA-005

Tabla 55. *Requisito CERTILOC-PP-RFO-005*

Identificador	CERTILOC-PP-RFO-006					
Nombre	Borrar política de privacidad					
Descripción	• Los usuarios responsables de dispositivos deben poder borrar políticas de de privacidad que les pertenezcan. • Cuando una política se borra en el sistema de políticas de privacidad, esta debe borrarse completamente de la aplicación.					
Prioridad	Alta	☒	Media	☐	Baja	☐
Necesidad	Esencial	☒	Deseable	☐	Opcional	☐
Fuente	Creadores de CERTILOC					
Pruebas de validación	CERTILOC-PP-PA-007					

Tabla 56. *Requisito CERTILOC-PP-RFO-006*

Identificador	CERTILOC-PP-RFO-007					
Nombre	Borrado automático de políticas de privacidad					
Descripción	• Cuando un usuario sea borrado del conjunto global de CERTILOC, todas sus políticas de privacidad deben ser borradas del mismo.					
Prioridad	Alta	☐	Media	☐	Baja	☒
Necesidad	Esencial	☐	Deseable	☒	Opcional	☐
Fuente	Creadores de CERTILOC					
Pruebas de validación	CERTILOC-PP-PA-015					

Tabla 57. *Requisito CERTILOC-PP-RFO-007*

Identificador	CERTILOC-PP-RFO-008					
Nombre	Autorización de peticiones contra dispositivos propios					
Descripción	• Todas las peticiones de autorización que realicen usuarios responsables de dispositivos contra dispositivos de los que sean responsables, deben ser siempre autorizadas.					
Prioridad	Alta	☐	Media	☐	Baja	☒
Necesidad	Esencial	☐	Deseable	☒	Opcional	☐
Fuente	Creadores de CERTILOC					
Pruebas de validación	No aplicable por prioridad del requisito.					

Tabla 58. *Requisito CERTILOC-PP-RFO-008*

Identificador	CERTILOC-PP-RFO-009					
Nombre	Consulta de políticas de privacidad					
Descripción	• Los usuarios responsables de dispositivos deben poder consultar sus políticas. En caso de no tener ninguna política de privacidad creada, el sistema les indicará que no tienen políticas. • Además, los usuarios podrán obtener una interpretación de sus políticas de privacidad con el formato nativo de XACML para poder hacer comprobaciones de corrección.					
Prioridad	Alta	☒	Media	☐	Baja	☐
Necesidad	Esencial	☒	Deseable	☐	Opcional	☐
Fuente	Creadores de CERTILOC					
Pruebas de validación	CERTILOC-PP-PA-008.					

Tabla 59. *Requisito CERTILOC-PP-RFO-009*

Identificador	CERTILOC-PP-RFO-010					
Nombre	Modificación de políticas de privacidad de CERTILOC					
Descripción	• Los usuarios responsables de dispositivos, que tengan políticas de privacidad dadas de alta en el sistema, deben poder modificar cualquiera de los parámetros definidos para estas políticas, exceptuando el identificador único que las identifica.					
Prioridad	Alta	☐	Media	☒	Baja	☐
Necesidad	Esencial	☐	Deseable	☒	Opcional	☐
Fuente	Autor del proyecto					
Pruebas de validación	CERTILOC-PP-PA-010.					

Tabla 60. *Requisito CERTILOC-PP-RFO-010*

Identificador	CERTILOC-PP-RFO-011					
Nombre	Definición de obligaciones asociadas a políticas de privacidad					
Descripción	• Los usuarios responsables de dispositivos, deben poder asociar obligaciones con sus políticas. Estas obligaciones se deben cumplir por el usuario que realiza la petición de autorización.					
Prioridad	Alta	☒	Media	☐	Baja	☐
Necesidad	Esencial	☒	Deseable	☐	Opcional	☐
Fuente	Creadores de CERTILOC					
Pruebas de validación	CERTILOC-PP-PA-011					

Tabla 61. *Requisito CERTILOC-PP-RFO-011*

Identificador	CERTILOC-PP-RFO-012					
Nombre	Registros de funcionamiento del sistema de políticas de privacidad					
Descripción	• Toda la actividad del sistema de funcionamiento de políticas de privacidad debe generar eventos del sistema que serán guardados en ficheros dentro del contenedor de la aplicación. • Estos registros de actividad no deben contener datos que amenacen la privacidad de los usuarios de CERTILOC y deben ceñirse única y exclusivamente al funcionamiento del sistema de políticas de privacidad.					
Prioridad	Alta	☒	Media	☐	Baja	☐
Necesidad	Esencial	☒	Deseable	☐	Opcional	☐
Fuente	Creadores de CERTILOC					
Pruebas de validación	No aplicable					

Tabla 62. *Requisito CERTILOC-PP-RFO-012*

Identificador	CERTILOC-PP-RFO-013					
Nombre	Acceso del administrador a los registros de funcionamiento del sistema					
Descripción	• Los usuarios administradores del sistema podrán acceder a los ficheros de registros de actividad de funcionamiento del sistema.					
Prioridad	Alta	☒	Media	☐	Baja	☐
Necesidad	Esencial	☐	Deseable	☒	Opcional	☐
Fuente	Creadores de CERTILOC					
Pruebas de validación	CERTILOC-PP-PA-013					

Tabla 63. *Requisito CERTILOC-PP-RFO-013*

Identificador	CERTILOC-PP-RFO-014					
Nombre	Funciones aplicables en las políticas					
Descripción	• Las políticas de privacidad del sistema de políticas de CERTILOC deben poder contener funciones de evaluación. • Se debe implementar como mínimo una función que evalúe parámetros relacionados con la fecha o con la hora y otra función que evalúe parámetros de localización.					
Prioridad	Alta	☒	Media	☐	Baja	☐
Necesidad	Esencial	☒	Deseable	☐	Opcional	☐
Fuente	Creadores de CERTILOC					
Pruebas de validación	CERTILOC-PP-PA-013					

Tabla 64. *Requisito CERTILOC-PP-RFO-014*

3.4.3 REQUISITOS NO FUNCIONALES

3.4.3.1 Requisitos de rendimiento

No se han definido requisitos de rendimiento concretos.

3.4.3.2 Requisitos operacionales

Identificador	CERTILOC-PP-RNFO-001					
Nombre	Interacción del usuario con el sistema					
Descripción	La interacción del usuario con el sistema de políticas de privacidad se realizará mediante el teclado y el ratón. La visualización será mediante la pantalla de su equipo PC.					
Prioridad	Alta	☒	Media	☐	Baja	☐
Necesidad	Esencial	☒	Deseable	☐	Opcional	☐
Fuente	Autor del implementador del sistema de políticas de privacidad					
Pruebas de validación	No Aplicable					

Tabla 65. *Requisito CERTILOC-PP-RNFO-001*

3.4.3.3 Requisitos de seguridad

Identificador	CERTILOC-PP-RNFS-001					
Nombre	Comprobación con políticas de privacidad					
Descripción	Para realizar una comprobación según políticas de privacidad: • se obtiene el identificador del dispositivo sobre el que se solicita el servicio (localizar, generar CET o descargar CET) • se obtiene el usuario responsable de dicho dispositivo • se van recorriendo todas las reglas activas pertenecientes al usuario responsable (propietario de la política) comprobando si los datos de la petición encajan con lo establecido en la regla. Si es así, se autoriza la petición. Si se recorren todas las reglas y no ha encajado en ninguna, se deniega la petición					
Prioridad	Alta	☒	Media	☐	Baja	☐
Necesidad	Esencial	☒	Deseable	☐	Opcional	☐
Fuente	Creadores de CERTILOC					
Pruebas de validación	No Aplicable					

Tabla 66. *Requisito CERTILOC-PP-RNFS-001*

3.4.3.4 Requisitos de mantenimiento

No se han definido requisitos de mantenimiento.

3.4.3.5 Requisitos de interfaz

Identificador	CERTILOC-PP-RNFI-001					
Nombre	Comunicación del usuario con el sistema de políticas de privacidad					
Descripción	El usuario podrá acceder al sistema de políticas de privacidad e interactuar con el mismo mediante un cliente Web que utilizará el protocolo de comunicaciones estándar HTTP.					
Prioridad	Alta	☒	Media	☐	Baja	☐
Necesidad	Esencial	☐	Deseable	☒	Opcional	☐
Fuente	Creadores de CERTILOC					
Pruebas de validación	No Aplicable					

Tabla 67. *Requisito CERTILOC-PP-RNFI-001*

Identificador	CERTILOC-PP-RNFI-002					
Nombre	Persistencia de datos del sistema de políticas de privacidad					
Descripción	Todos los datos referentes a políticas de privacidad del sistema a desarrollar, estarán alojados en una base de datos con motor SQL.					
Prioridad	Alta	☒	Media	☐	Baja	☐
Necesidad	Esencial	☒	Deseable	☐	Opcional	☐
Fuente	Autor del implementador del sistema de políticas de privacidad					
Pruebas de validación	No Aplicable					

Tabla 68. *Requisito CERTILOC-PP-RNFI-002*

3.4.3.6 Requisitos de recursos

Identificador	CERTILOC-PP-RRec-001					
Nombre	Hardware de alojamiento de CERTILOC					
Descripción	Goofy, la máquina donde se alojará el núcleo de CERTILOC es un servidor Super Micro SC743I645B con placa base H8DME2, dos procesadores AMD Opteron Dual Core, 2 GB de memoria RAM DDR2 y tres discos duros de 320 GB SATA-II. Goofy dispone de tarjeta de red Super Micro MCP55 Ethernet.					
Prioridad	Alta	☒	Media	☐	Baja	☐
Necesidad	Esencial	☒	Deseable	☐	Opcional	☐
Fuente	Creadores de CERTILOC					
Pruebas de validación	No Aplicable					

Tabla 69. *Requisito CERTILOC-PP-RRec-001*

No se han más definido recursos externos que deban interactuar directamente con el sistema de políticas de privacidad.

Cabe comentar que, otros módulos del sistema CERTILOC, sí interactúan con recursos externos tales como PDAs, etiquetas RFID, dispositivos GSM, etc.

3.4.3.7 Requisitos de verificación

Identificador	CERTILOC-PP-RNFV-001					
Nombre	Verificación del sistema de políticas de privacidad					
Descripción	Para dar por verificado el sistema de políticas de privacidad, se deben pasar un porcentaje del 90% de todas las pruebas definidas en el apartado 4.5.2 del presente documento de manera satisfactoria.					
Prioridad	Alta	☐	Media	☒	Baja	☐
Necesidad	Esencial	☒	Deseable	☐	Opcional	☐
Fuente	Autor del implementador del sistema de políticas de privacidad					
Pruebas de validación	No Aplicable					

Tabla 70. *Requisito CERTILOC-PP-RNFV-001*

3.4.3.8 Requisitos de aceptación

Identificador	CERTILOC-PP-RNFA-001					
Nombre	Aceptación del sistema de políticas de privacidad					
Descripción	Para dar por aceptado el sistema de políticas de privacidad, se debe haber realizado una correcta verificación del mismo. Además, el sistema debe superar al menos el 90% de todas las pruebas de aceptación que hayan sido definidas en el apartado 4.6 del presente documento.					
Prioridad	Alta	☐	Media	☒	Baja	☐
Necesidad	Esencial	☐	Deseable	☒	Opcional	☐
Fuente	Autor del implementador del sistema de políticas de privacidad					
Pruebas de validación	No Aplicable					

Tabla 71. *Requisito CERTILOC-PP-RNFA-001*

3.4.3.9 Requisitos de documentación

Identificador	CERTILOC-PP-RNFDoc-001					
Nombre	Memoria de proyecto de implementación de sistema de políticas de privacidad.					
Descripción	Al concluir la implementación y las pruebas del sistema de políticas de privacidad, el autor del mismo deberá realizar una memoria que resuma todas las actividades llevadas a cabo para su desarrollo.					
Prioridad	Alta	☒	Media	☐	Baja	☐
Necesidad	Esencial	☒	Deseable	☐	Opcional	☐
Fuente	Naturaleza del presente proyecto					
Pruebas de validación	No Aplicable					

Tabla 72. *Requisito CERTILOC-PP-RNFDoc-001*

3.4.3.10 Requisitos de diseño

Identificador	CERTILOC-PP-RNFD-001					
Nombre	Sistema operativo subyacente de alojamiento de CERTILOC					
Descripción	El sistema operativo subyacente al núcleo de CERTILOC será GNU/Linux. La distribución GNU/Linux seleccionada es UBUNTU Server.					
Prioridad	Alta	☒	Media	☐	Baja	☐
Necesidad	Esencial	☒	Deseable	☐	Opcional	☐
Fuente	Creadores de CERTILOC					
Pruebas de validación	No Aplicable					

Tabla 73. *Requisito CERTILOC-PP-RNFD-001*

Identificador	CERTILOC-PP-RNFD-002					
Nombre	Lenguaje de implementación del sistema de políticas de privacidad					
Descripción	El lenguaje de programación a utilizar para implementar el sistema de políticas de privacidad de CERTILOC será Java 1.5. El núcleo de CERTILOC también se implementará utilizando la plataforma Java Platform, Enterprise Edition 5.					
Prioridad	Alta	☒	Media	☐	Baja	☐
Necesidad	Esencial	☒	Deseable	☐	Opcional	☐
Fuente	Creadores de CERTILOC					
Pruebas de validación	No Aplicable					

Tabla 74. *Requisito CERTILOC-PP-RNFD-002*

Identificador	CERTILOC-PP-RNFD-003					
Nombre	Contenedor Web CERTILOC					
Descripción	Se utilizará Apache Tomcat 5.5 como contenedor de la capa web.					
Prioridad	Alta	☒	Media	☐	Baja	☐
Necesidad	Esencial	☒	Deseable	☐	Opcional	☐
Fuente	Creadores de CERTILOC					
Pruebas de validación	No Aplicable					

Tabla 75. *Requisito CERTILOC-PP-RNFD-003*

Identificador	CERTILOC-PP-RNFD-004					
Nombre	Servidor de bases de datos de CERTILOC					
Descripción	El gestor de las bases de datos será MySQL versión 5.0.26-12. La comunicación entre la capa de negocio y las bases de datos se realizará utilizando Java Database Connectivity API (JDBC).					
Prioridad	Alta	☒	Media	☐	Baja	☐
Necesidad	Esencial	☒	Deseable	☐	Opcional	☐
Fuente	Creadores de CERTILOC					
Pruebas de validación	No Aplicable					

Tabla 76. *Requisito CERTILOC-PP-RNFD-004*

Identificador	CERTILOC-PP-RNFD-005					
Nombre	Struts para desarrollo web					
Descripción	Se utilizará la tecnología Apache STRUTS como marco de desarrollo del núcleo de CERTILOC					
Prioridad	Alta	☒	Media	☐	Baja	☐
Necesidad	Esencial	☒	Deseable	☐	Opcional	☐
Fuente	Creadores de CERTILOC					
Pruebas de validación	No Aplicable					

Tabla 77. *Requisito CERTILOC-PP-RNFD-005*

Identificador	CERTILOC-PP-RNFD-006					
Nombre	Escalabilidad del sistema de políticas de privacidad					
Descripción	La implementación del sistema de políticas de privacidad debe realizarse teniendo siempre en cuenta la escalabilidad del proyecto. Además, se debe programar la aplicación de manera que sus sub-módulos pudiesen ejecutarse en distintas máquinas en el futuro.					
Prioridad	Alta	☐	Media	☒	Baja	☐
Necesidad	Esencial	☐	Deseable	☐	Opcional	☒
Fuente	Creadores de CERTILOC					
Pruebas de validación	No Aplicable					

Tabla 78. *Requisito CERTILOC-PP-RNFD-006*

Identificador	CERTILOC-PP- RNFD-007					
Nombre	Módulo SGP					
Descripción	• El sistema de políticas de privacidad debe contener un módulo que ofrecerá una interfaz web para la gestión (crear, borrar, modificar, activar y desactivar) de políticas de privacidad a los usuarios responsables de dispositivos de CERTILOC.					
Prioridad	Alta	☒	Media	☐	Baja	☐
Necesidad	Esencial	☒	Deseable	☐	Opcional	☐
Fuente	Creadores de CERTILOC					
Pruebas de validación	No aplicable					

Tabla 79. *Requisito CERTILOC-PP- RNFD-007*

Identificador	CERTILOC-PP- RNFD-008
Nombre	Módulo ACP
Descripción	• El sistema de políticas de privacidad debe contener un módulo denominado ACP que ofrecerá servicios de autorización a otros módulos de la aplicación. Dicha autorización debe venir determinada por las políticas de privacidad dadas de alta en el sistema y asociadas a los distintos usuarios CERTILOC que sean responsables de dispositivos
Prioridad	Alta ☒ Media ☐ Baja ☐
Necesidad	Esencial ☒ Deseable ☐ Opcional ☐
Fuente	Creadores de CERTILOC
Pruebas de validación	No aplicable

Tabla 80. *Requisito CERTILOC-PP-RNFD-008*

Identificador	CERTILOC-PP- RNFD-009
Nombre	Módulo MARPP
Descripción	• El sistema de políticas de privacidad debe contener un módulo denominado MARPP será el encargado de guardar las políticas de privacidad en un repositorio de datos persistente.
Prioridad	Alta ☒ Media ☐ Baja ☐
Necesidad	Esencial ☒ Deseable ☐ Opcional ☐
Fuente	Creadores de CERTILOC
Pruebas de validación	No aplicable

Tabla 81. *Requisito CERTILOC-PP- RNFD-009*

3.4.3.11 Requisitos de portabilidad

No se han definido requisitos de portabilidad para el proyecto.

3.4.3.12 Requisitos de calidad

Identificador	CERTILOC-PP-RNFCal-001					
Nombre	Estándar de codificación					
Descripción	Para la realización del código fuente, se deberá seguir el estándar de programación de la API de Java.					
Prioridad	Alta	☐	Media	☒	Baja	☐
Necesidad	Esencial	☐	Deseable	☒	Opcional	☐
Fuente	Autor del implementador del sistema de políticas de privacidad					
Pruebas de validación	No Aplicable					

Tabla 82. *Requisito CERTILOC-PP-RNCal-001*

3.4.3.13 Requisitos de entrega

Identificador	CERTILOC-PP-En-001					
Nombre	Entrega de CD Final					
Descripción	El proyecto completo, tanto de documentación como de desarrollo, se entregará reunido en un disco compacto.					
Prioridad	Alta	☐	Media	☒	Baja	☐
Necesidad	Esencial	☐	Deseable	☒	Opcional	☐
Fuente	Autor del implementador del sistema de políticas de privacidad					
Pruebas de validación	No Aplicable					

Tabla 83. *Requisito CERTILOC- PP-En-001*

3.4.4 REQUISITOS INVERSOS

Los requisitos inversos nos ayudarán a completar el análisis de requisitos indicando lo que la aplicación no debe hacer.

Se presentan a continuación los requisitos inversos para el sistema de políticas de privacidad.

Identificador	CERTILOC-PP-RI-001					
Nombre	No permitir navegar por políticas de privacidad de otro usuario					
Descripción	El sistema de políticas de privacidad, no debe mostrar, al usuario que ha ingresado en la aplicación, políticas de privacidad relacionadas con otros usuarios.					
Prioridad	Alta	☒	Media	☐	Baja	☐
Necesidad	Esencial	☒	Deseable	☐	Opcional	☐
Fuente	Creadores de CERTILOC					
Pruebas de validación	No Aplicable					

Tabla 84. *Requisito CERTILOC-PP-RI-001*

Identificador	CERTILOC-PP-RI-002					
Nombre	No permitir ver registros de actividad de otros usuarios					
Descripción	El sistema de políticas de privacidad, no debe mostrar, al usuario que ha ingresado en la aplicación, registros de actividad de políticas que no estén relacionadas con las políticas de privacidad del mismo usuario.					
Prioridad	Alta	☒	Media	☐	Baja	☐
Necesidad	Esencial	☒	Deseable	☐	Opcional	☐
Fuente	Creadores de CERTILOC					
Pruebas de validación	No Aplicable					

Tabla 85. *Requisito CERTILOC-PP-RI-002*

Identificador	CERTILOC-PP-RI-003					
Nombre	No permitir crear políticas de privacidad para dispositivos de los que no somos responsables					
Descripción	El sistema de políticas de privacidad, no debe permitir, al usuario que ha ingresado en la aplicación, crear políticas para dispositivos de los que el usuario no es el responsable.					
Prioridad	Alta	☒	Media	☐	Baja	☐
Necesidad	Esencial	☒	Deseable	☐	Opcional	☐
Fuente	Creadores de CERTILOC					
Pruebas de validación	CERTILOC-PP-PA-014					

Tabla 86. *Requisito CERTILOC-PP-RI-003*

Identificador	CERTILOC-PP-RI-004					
Nombre	El sistema de gestión de la privacidad o SGP no será responsable de autenticar usuarios de CERTILOC					
Descripción	El módulo SGP no se encargará de autenticar a los posibles usuarios de la aplicación. Esta funcionalidad será delegada en otro módulo de CERTILOC. Cuando los usuarios lleguen al gestor de políticas de privacidad (módulo SGP) ya estarán previamente autenticados en la aplicación.					
Prioridad	Alta	☐	Media	☐	Baja	☒
Necesidad	Esencial	☒	Deseable	☐	Opcional	☐
Fuente	Creadores de CERTILOC					
Pruebas de validación	No Aplica					

Tabla 87. *Requisito CERTILOC-PP-RI-004*

3.5 ANÁLISIS DE LAS TECNOLOGÍAS A UTILIZAR

En los siguientes apartados haremos un análisis de las tecnologías, a utilizar para el desarrollo del sistema de políticas de privacidad. Estas tecnologías se han escogido por convenio de los desarrolladores del marco del proyecto CERTILOC y otras motivaciones que veremos a continuación.

En primer lugar se presenta el estándar XACML (requisito **CERTILOC-PP-RFI-001**) de una manera detallada. Se considera necesario realizar un análisis conciso de la tecnología ya que es compleja y sólo conocida a grandes rasgos por los desarrolladores del proyecto. Se estudiará cómo encaja el estándar XACML en CERTILOC y se aportarán las distintas motivaciones que han generado su uso.

En segundo lugar se presenta la tecnología STRUTS (requisito **CERTILOC-PP-RNFD-005**). Esta tecnología se presenta de una manera superficial. Se recomienda al lector referirse a la documentación del primer PFC orientado al desarrollo del demostrador de CERTILOC (Memoria PFC - Calvo Martínez 2007) para ampliar los conocimientos de dicha tecnología.

3.5.1 EL ESTÁNDAR XACML 1.0

3.5.1.1 Introducción al estándar

XACML es la abreviatura de "eXtensible Access Control Markup Language", es decir, Lenguaje de marcado extensible para el control de acceso. Es una especificación de un lenguaje basado en XML y orientado a la definición de políticas de seguridad, peticiones y respuestas de autorización. La especificación de XACML fue desarrollada por Sun Microsystems y aceptada por OASIS como estándar (XACML - OASIS 2009). Además de un **modelo de datos**, propone un **modelo de flujo de datos** por distintos agentes que finalmente deriva en un resultado de autorización o denegación de una determinada petición de autorización.

El flujo de datos se basa en el intercambio de mensajes entre distintos agentes. Estos agentes son responsables, entre otras funciones, de **traducir las peticiones en el formato nativo** de la aplicación donde se implanta el sistema de seguridad **al formato de XACML**, para su posterior **evaluación contra las políticas**, que habrán sido **convertidas previamente** de su formato nativo al **formato de XACML**. En última instancia, la evaluación de una posible petición de autorización se realiza mediante documentos XML. Por último, para devolver una respuesta de autorización coherente al sistema donde se alberga el servicio de autorización, la **respuesta**, todavía en el formato del lenguaje XACML, debe ser previamente **convertida de vuetla al lenguaje nativo** de dicho sistema.

XACML surge de la necesidad de estandarizar el lenguaje e infraestructura de los sistemas de seguridad (relacionados con la realización de una determinada acción sobre un determinado recurso) que contemplan los distintos productos de la industria de los sistemas de información.

Los sistemas de seguridad de acceso a la información que han sido desarrollados hasta el momento, se han creado con el objetivo de ser aplicables en una gran cantidad de entornos operativos, esto provoca, en la mayoría de los casos, que sean sistemas de seguridad débiles o bien con pocas y, sobretodo, costosas posibilidades de expansión.

En las grandes empresas o en las entidades públicas, las políticas de privacidad pueden tener muchos elementos y muchos puntos de aplicación. Los elementos de las políticas de privacidad suelen ser gestionadas por distintos departamentos, como pueden ser el departamento de sistemas informáticos, el de recursos humanos, el legal, el departamento financiero, etc. Los puntos de aplicación de las políticas de privacidad también pueden ser muy variados, pasando por sistemas de correo electrónico, sistemas de acceso compartido a datos,

sistemas de acceso externo, etc. Como consecuencia de esto, las propuestas de cambio de las políticas de privacidad suelen tener un coste elevado y, en muchos casos, una funcionalidad no muy fiable. Además, cada vez con mayor frecuencia, los consumidores, reguladores e inversores de los sistemas de información, reclaman una mayor seguridad y privacidad y el uso de las "mejores prácticas" para su aplicación. Esta es la razón básica de la necesidad de la implementación de un lenguaje común para expresar las políticas de privacidad y es aquí donde encuentra su motivación el lenguaje XACML.

Podemos encontrar una introducción detallada de las motivaciones de XACML en el documento Oasis, eXtensible Access Control Markup Language (XACML) v 1.0, OASIS standard 18 Feb. 2003 (XACML - OASIS 2009) . Lo único que debemos remarcar es que XACML es un lenguaje pensado para unificar la redacción de políticas de privacidad de los sistemas de información, lo que nos va a permitir ofrecer a CERTILOC un sistema de privacidad sólido y fiable y que, además esté preparado para posibles interacciones futuras con otros sistemas de seguridad.

3.5.1.2 Motivaciones para su elección

Antes de pasar a describir los detalles del estándar, expondremos las razones por las que se ha decidido utilizar este estándar de seguridad para la definición del sistema de políticas de privacidad del demostrador de CERTILOC, y no otro.

En primer lugar, se ha escogido el estándar XACML, principalmente porque sus conceptos básicos coinciden en gran medida con los conceptos básicos de privacidad exigidos en el proyecto CERTILOC. En particular:

- El estándar ofrece un modelado de datos para definir **políticas** de control de acceso (denominadas "de seguridad" en el universo de XACML y "de privacidad" en el universo propuesto por CERTILOC) que rigen y controlan el acceso a la información de determinados recursos. Esto coincide con los Casos de Uso (aptdo. 3.3.4) del análisis del sistema de políticas de privacidad de CERTILOC.
- Por otro lado, tanto CERTILOC como XACML, consideran unas **obligaciones** que deben cumplirse una vez que se obtiene una respuesta para determinada petición de autorización. En CERTILOC las obligaciones estarían generalmente relacionadas con condiciones legales del uso de la IET o CET sobre el que se solicita la petición.
- XACML define distintos tipos **atributos** (CERTILOC-PP-RFI-003) para definir parámetros sobre:

- El actor que realiza la petición (“Usuario solicitante de servicios” en CERTILOC y sujeto en XACML). Aquí podemos incluir atributos con información del rol de usuario e identificador de usuario que solicita determinado servicio que implique al sistema de políticas de privacidad.
- La acción que se desea realizar sobre la información. En el caso de CERTILOC puede ser Obtener IET, Descargar CET, Generar CET, Borrar CET.
- El recurso sobre el que se solicita la petición de acceso. En CERTILOC sería el identificador del dispositivo sobre el que se realiza localización o relacionado con el CET sobre el que se quiere realizar la acción.
- El entorno bajo el que se realiza la petición. En XACML, los atributos de entorno están relacionados con las circunstancias de entorno concretas bajo las que ocurre la petición de autorización. En CERTILOC, podemos encontrar atributos de entorno como la localización, la fecha y hora de ejecución de la petición de autorización y la fecha y hora de la toma de los datos de IET.

- Por último, este estándar ofrece diferentes maneras de aplicar **funciones condicionales y lógicas**. En CERTILOC, se exige que se puedan aplicar distintas funciones a los datos de una determinada petición de autorización, contra los datos concretos definidos en una política de privacidad.

Otra de las principales motivaciones para la elección de este estándar tiene que ver con la complicación de la definición de un sistema de políticas de privacidad escalable y fiable. El estándar está desarrollado por expertos en seguridad y ayuda en gran medida a crear el sistema de políticas de privacidad ya que nos brinda un **modelo de datos** y un **modelo de flujo** de los mismos, lo cual nos ahorrará bastantes decisiones de diseño y desarrollo.

Por otro lado XACML es un estándar que ayuda a la unificación de los sistemas de seguridad basados en políticas de seguridad. Esto significa que, en un futuro, CERTILOC estará preparado para interactuar con otros sistemas de seguridad basados en este estándar de políticas de seguridad, lo que aumenta la escalabilidad global del proyecto CERTILOC, y más en particular del sistema de políticas de privacidad.

En último lugar se ha escogido el estándar XACML para coincidir con las especificaciones de privacidad propuestas por los autores del modelo del demostrador de CERTILOC (4). La elección del estándar concreto, fue hecha en primera instancia por dichos autores, como queda patente en la presentación de requisitos del software de la aplicación (CERTILOC-PP-RFI-001).

A pesar de tener buenas motivaciones para su elección, esta elección aumenta en gran medida el grado de dificultad del proyecto abordado en el presente PFC. XACML es un estándar complejo y, en ocasiones, poco trivial. Además se trata de un estándar poco extendido, lo que complica el acceso a recursos de información para poder comprenderlo completamente y el acceso a herramientas intermedias para su adaptación.

3.5.1.3 Detalles concretos del estándar

3.5.1.3.1 El modelo de datos de XACML

La figura que se presenta a continuación, presenta el modelo de datos tal y como se presenta en la especificación del estándar de XACML.

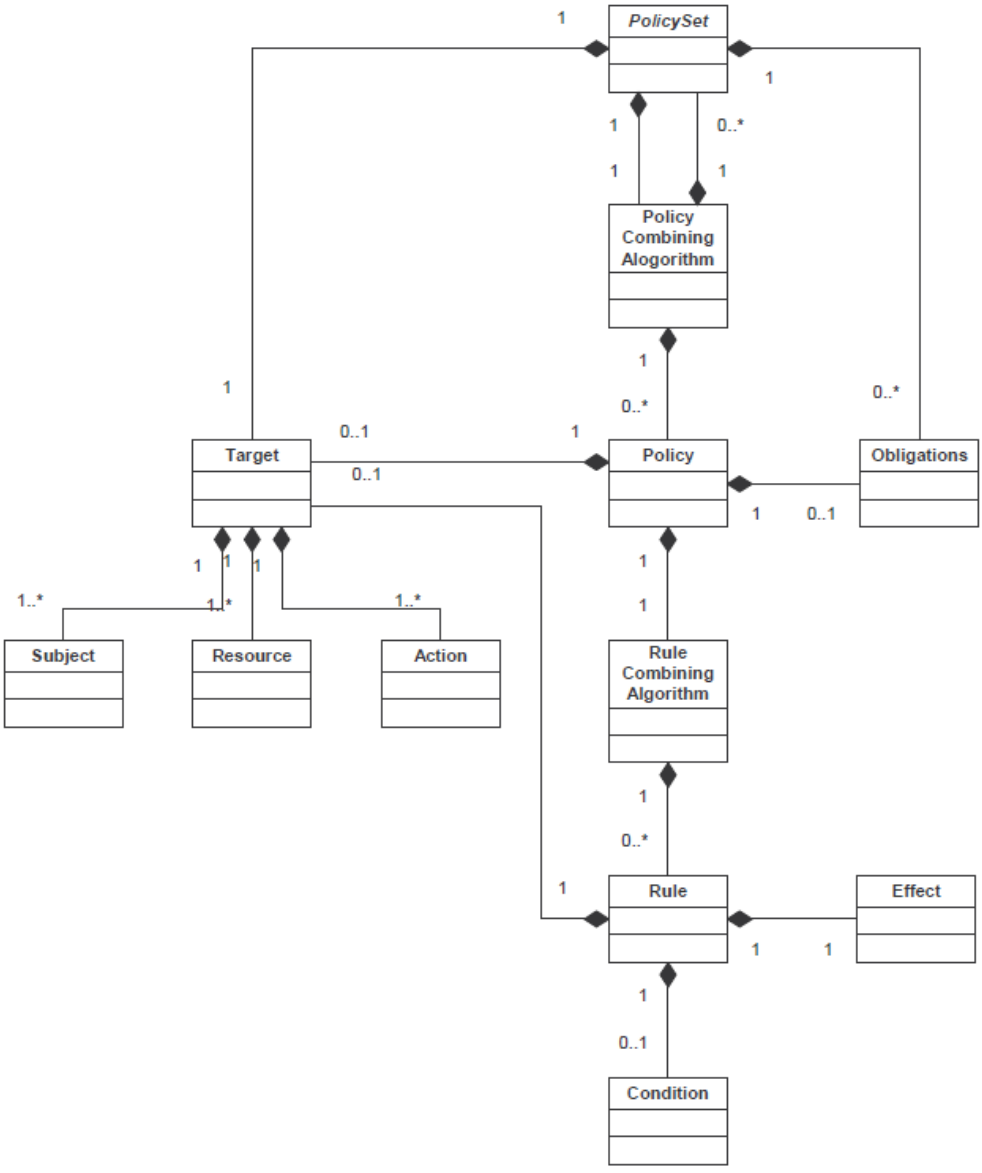


Figura 11. *El modelo de políticas de XACML (XACML - OASIS 2009)*

Veamos una definición de cada uno de los elementos que se aprecian en la figura anterior:

Nombre	Traducción	Descripción
PolicySet	Conjunto de políticas	Se define como un conjunto de políticas de privacidad, un algoritmo de combinación de políticas, un objetivo de aplicación y, opcionalmente, un conjunto de obligaciones.
Policy	Política	Se define como un conjunto de reglas, un algoritmo de combinación de reglas, un objetivo de aplicación, y, opcionalmente, un conjunto de obligaciones. Una política, puede ser un elemento de un conjunto de políticas.
Policy Combining Algorithm	Algoritmo de combinación de políticas	El procedimiento para combinar múltiples decisiones y obligaciones que provienen de distintas políticas
Obligations	Obligaciones	Obligaciones a cumplir cuando un usuario recibe una respuesta de autorización o denegación cuando realiza una petición de autorización.
Rule	Regla	Se define como un objetivo de aplicación, un efecto y una condición.
Rule Combining Algorithm	Algoritmo de combinación de reglas	El procedimiento para combinar múltiples decisiones que provienen de distintas reglas
Effect	Efecto	La consecuencia propuesta por el cumplimiento de una regla (Permitir o Denegar)
Condition	Condición	Una expresión de predicados – Una función que evalúa a Verdadero, Falso o Indeterminado

Nombre	Traducción	Descripción
Apply	Aplicativo	Aunque no aparece en la Figura 11, XACML define este elemento para poder combinar funciones en las condiciones y crear funciones complejas. Los aplicativos se incluyen dentro de los parámetros de las funciones de las condiciones y **aplican** nuevas funciones.
Target	Objetivo de aplicación	Un conjunto de parámetros sobre **sujetos**, **recursos** y **acciones**. En CERTILOC, el objetivo de una petición de autorización suele ser un dispositivo a localizar (en XACML un **recurso**). Sin embargo, las políticas de privacidad permiten a los usuarios aplicar ciertas políticas bajo ciertos criterios que pueden implicar información sobre el usuario que realiza la petición (en XACML el **sujeto**), la acción que se quiere realizar (en XACML la **acción**) o el dispositivo sobre el que se realiza la petición (en XACML el **recurso**). De esta manera un usuario puede, por ejemplo, aplicar un conjunto de reglas determinado a todas las peticiones que impliquen una **acción** de **Descargar CET**.
Subject	Sujeto	Un actor cuyos atributos pueden ser referenciados por un predicado. Es el sujeto que realiza la petición de autorización.
Resource	Recurso	Un dato, un servicio o un componente del sistema
Action	Acción	Acción que se quiere realizar con la petición de autorización.

Tabla 88. *Entidades del modelo de datos XACML*

3.5.1.3.2 Combinaciones de reglas y políticas mediante algoritmos de combinación

Las políticas aplicables a una determinada petición de autorización pueden estar compuestas por varios conjuntos de **reglas** o **políticas** individuales.

Los algoritmos de combinación indican cómo se deben combinar los resultados de cada una de las reglas aplicables frente a las demás, y de cada una de las políticas aplicables frente a las demás.

XACML propone tres elementos de "nivel superior": **reglas**, **políticas** y **conjuntos de políticas** (<Rule> <Policy> y <PolicySet> respectivamente):

- Las **reglas** contienen una expresión booleana que puede ser evaluada independientemente.
- Las **políticas** contienen un conjunto de reglas y especifican el mecanismo para combinar los resultados de estas reglas en caso que varias sean aplicables ante la misma petición de autorización.
- Los **conjuntos de políticas** contienen una serie de políticas y especifican el mecanismo para combinar los resultados de estas políticas en caso que varias sean aplicables.

Además, XACML define una serie de algoritmos de combinación, identificados por un id de combinación de reglas o políticas (**RuleCombiningAlgorithmId** y **PolicyCombiningAlgorithmId** respectivamente).

En cualquiera de los dos casos, los algoritmos de combinación indican cómo obtener una sola decisión de autorización ante los posibles resultados de los conjuntos de reglas o políticas aplicables para la misma petición. Los algoritmos del estándar contemplan los siguientes casos:

- La permisividad prevalece (**Permit-Overrides**): Si encontramos una decisión de Permitir, ésta debe tener prioridad ante otras decisiones obtenidas.
- La denegación prevalece (**Deny-Overrides**): Si encontramos una decisión de Denegar, ésta debe tener prioridad ante otras decisiones obtenidas.
- La primera decisión encontrada prevalece (**First-Applicable**): La primera decisión que encontramos será la que tendrá prioridad sobre otras decisiones obtenidas.
- Sólo una decisión permitida (**Only-one-Applicable**): Este algoritmo sólo es aplicable ante conjuntos de políticas y devuelve "No Aplicable" en caso de no encontrar ninguna política cuyo objetivo coincida con la petición e "Indeterminado" en caso de encontrar varias políticas cuyos objetivos coincidan con los datos de la petición.

3.5.1.3.3 Los actores del modelo de flujo de datos XACML

Veamos qué módulos son los principales actores del modelo de flujo de datos de XACML y su función.

Nombre	Abreviatura	Función
Punto de Administración de Políticas – Policy Administration Point	PAP	La entidad del sistema que crea políticas o conjuntos de políticas
Punto de Decisión de Políticas – Policy Decision Point	PDP	La entidad del sistema encargada de encontrar políticas y crear una respuesta de decisión.
Punto de Ejecución de Políticas – Policy Enforcement Point	PEP	La entidad del sistema encargada de llevar el control de acceso mediante la creación de peticiones de acceso y ejecuciones de decisiones de autorización.
Punto de Información de Políticas – Policy Information Point	PIP	La entidad del sistema que se encarga de obtener los valores de los atributos de entorno (fecha, hora, etc.) de una petición.
"Gestor de Contexto" – "Context Handler"	-	La entidad del sistema encargada de convertir peticiones, en el formato nativo del sistema en el que nos encontramos, al formato XACML y decisiones de autorización, en formato XACML, de vuelta al formato nativo.
"Servicio de obligaciones" – "obligations service"	-	La entidad del sistema que aporta servicios para la correcta aplicación de obligaciones.
Recurso – Resource	-	Recurso al que se desea acceder en una determinada petición de autorización.

Nombre	Abreviatura	Función
Entorno - Environment	-	El conjunto de atributos que son relevantes para la obtención de una decisión de autorización y que son independientes del sujeto que realiza la petición, el recurso que solicita y la acción que se quiere realizar.
Sujeto o Sujetos – Subject / Subjects	-	Sujeto o sujetos que solicitan la petición de autorización.

Tabla 89. *Actores principales del modelo de flujo de datos de XACML*

3.5.1.3.4 El modelo de flujo de datos de XACML

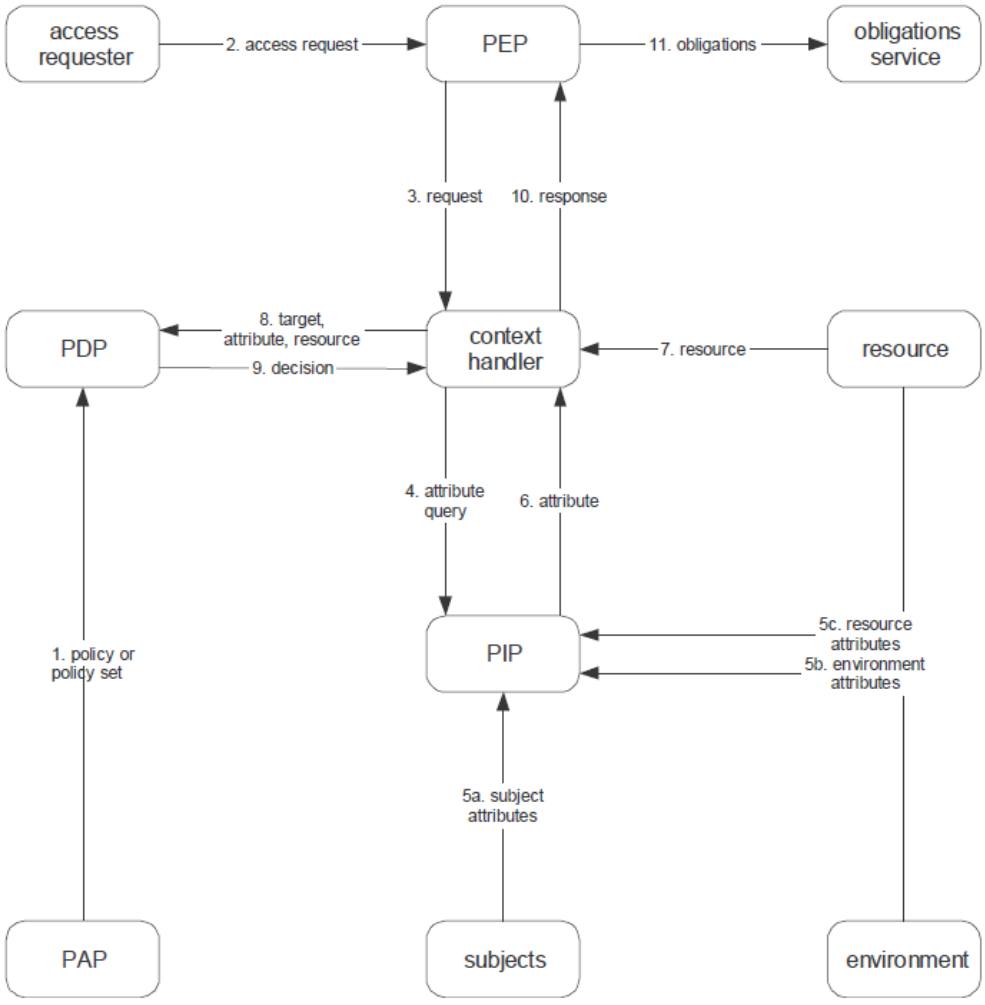


Figura 12. *Modelo de flujo de datos de XACML (XACML - OASIS 2009)*

Este modelo de flujo de datos se ejecuta siguiendo los pasos que describimos a continuación:

a) El módulo PAP define políticas y conjuntos de políticas que hace accesible al módulo PDP. Estos conjuntos de políticas, especifican el sistema de políticas completo para un determinado objetivo de petición.
b) El módulo "solicitante de acceso" (access requester) manda una determinada petición al módulo PEP.
c) El módulo PEP manda la petición de acceso al "gestor de contexto" (context handler) incluyendo, opcionalmente, atributos del sujeto que realiza la petición, de la acción a realizar y del recurso al que se quiere obtener acceso. En los pasos d, e y f, el "gestor de contexto" creará una petición con formato XACML.
d) El módulo PIP obtiene los atributos solicitados.
e) El módulo PIP devuelve los atributos solicitados al "gestor de contexto".
f) Opcionalmente, el "gestor de contexto" incluirá el recurso en el contexto de petición que generará.
g) El "gestor de contexto", envía una petición de decisión, incluyendo el objetivo ("target") de la misma, al módulo PDP. El módulo PDP, encuentra la política aplicable a ese objetivo y, opcionalmente, recupera del "gestor de contexto" los atributos ("attributes") y el recurso ("resource"). Una vez hecho esto, el módulo PDP evalúa la política contra la petición.
h) Una vez obtenida la respuesta, el módulo PDP devuelve un "contexto de respuesta" (response context) en formato XACML al "gestor de contexto".
i) El "gestor de contexto", convierte el contexto de respuesta al formato nativo, que comprende el módulo PEP, y se lo devuelve (response).
j) El módulo PEP, se encarga de cumplimentar las obligaciones especificadas en la respuesta.
k) Si se decide permitir la petición, el módulo PEP devuelve una respuesta de autorización permisiva, en caso contrario negará la autorización (no se muestra este paso en la figura del modelo de flujo).

3.5.1.4 Aproximación entre Certiloc y el estándar XACML

Para poder hacer una aproximación de cómo se integrarán los distintos aspectos del estándar al SPP de CERTILOC, debemos dividir el problema en varios sub-problemas:

- Por un lado, se debe adaptar el modelo de datos de políticas de privacidad de CERTILOC al modelo de datos de políticas de seguridad de XACML o bien ampliar el estándar para adaptarse a CERTILOC.
- Además, y dado que el estándar XACML es muy amplio (está concebido para cualquier sistema de seguridad), hay que escoger qué elementos concretos del estándar serán necesarios para la implementación concreta del sistema de políticas de privacidad de CERTILOC.
- Por otro lado, se deben integrar la arquitectura de módulos propuestos por CERTILOC y el modelo del flujo de datos de XACML. Es decir, se debe adaptar el flujo de datos de políticas de privacidad, peticiones y respuestas de CERTILOC al cumplimiento del estándar en la medida de lo posible (esto implica la creación de un nuevo módulo -AGPA- orientado a la traducción del contexto de CERTILOC al contexto de XACML y viceversa).
- Por último, durante la fase de diseño del sistema habrá que adaptar la herramienta intermedia o API utilizada para el manejo de datos de XACML y para la resolución de evaluaciones mediante un PDP. En el apartado 4.6.2.3 podemos ver las ampliaciones concretas realizadas al API utilizado para este objetivo.

3.5.1.4.1 Adaptación de los modelos de datos de XACML y de CERTILOC

Para crear el modelo de datos de las políticas de privacidad de CERTILOC, se replicará el modelo de datos propuesto por la herramienta intermedia o API que utilicemos para manejar los datos en el contexto de XACML, por lo que hasta que no lleguemos a la fase de diseño del sistema, no podremos aproximar ambos modelos de datos (en un principio deberían ser iguales).

Sin embargo hay ciertos aspectos relacionados con las diferencias entre XACML y CERTILOC que sí podremos contemplar en la fase de análisis y nos pueden ayudar a concebir la unión de CERTILOC y XACML.

Después de un estudio concienzudo del estándar y conociendo a fondo su modelo de datos, sabemos que hay un concepto que existe en CERTILOC y que no está contemplado en el estándar de XACML, y es que el estándar no distingue entre elementos **activos** e **inactivos**. En CERTILOC se debe contemplar este aspecto (caso de uso CU-RD-002). Por lo tanto, el modelo de datos concebido para CERTILOC, tendrá que tener en cuenta este nuevo concepto.

Por otro lado, donde CERTILOC sólo exige la utilización de **políticas de privacidad**, el estándar XACML nos brinda **conjuntos de políticas**, **políticas** y **reglas**. Para adecuar este

aspecto entre XACML y CERTILOC, consideraremos que cada usuario CERTILOC podrá poseer varios conjuntos de políticas en el sistema. Estos conjuntos de políticas incluirán políticas, y a su vez, las políticas contendrán reglas.

En cuanto a los objetivos de conjuntos de políticas, políticas y reglas, CERTILOC asumirá el mismo modelo que XACML. Es decir, los objetivos de conjuntos de políticas, políticas y reglas estarán compuestos por atributos sobre el usuario CERTILOC que solicita la operación (el **sujeto en XACML**), la acción a realizar (**la acción en XACML**) y el dispositivo sobre el que se realiza la petición (el **recurso en XACML**).

- Los atributos a contemplar acerca del usuario que solicita la operación serán el **rol** del usuario CERTILOC y su identificador (**id**).
- El atributo a contemplar acerca de la acción a realizar y el dispositivo sobre la que se realiza, será sólo el identificador (**id**).

CERTILOC también exige que las **políticas de privacidad** incluyan atributos relacionados con la IET, la fecha y hora de la ejecución de la petición de acceso a datos de IET y la fecha y hora de la recogida de la IET. XACML sólo permite incluir este tipo de atributos (de entorno) dentro de las condiciones y los aplicativos y no dentro de los objetivos de conjuntos de políticas, políticas y reglas. Por lo tanto, estos atributos se incluirán en atributos de entorno dentro de las condiciones de XACML. Además, las condiciones y aplicativos, podrán contener igualmente atributos de sujeto, acción y recurso.

Además de todo lo descrito anteriormente, CERTILOC exige que el sistema de políticas de privacidad pueda aplicar ciertas **funciones** que actúen sobre la información manejada por las políticas y las peticiones de autorización. En XACML, sólo se permite la introducción de funciones, que aplican operaciones sobre la información de peticiones y políticas, dentro de las **condiciones y aplicativos.** De esta manera, CERTILOC tendrá que resumir su uso a los elementos de condiciones y aplicativos. Es decir, los usuarios sólo podrán aplicar funciones que manejen información de peticiones y políticas dentro de las condiciones y los aplicativos de las políticas de privacidad. Además, XACML sólo permite introducir atributos de entorno en aplicativos y condiciones por lo que en CERTILOC, tendremos que resumir el uso de atributos de entorno (localización, fecha y hora de ejecución de petición e información espacio-temporal) a aplicativos y condiciones.

Por último, CERTILOC requiere que se puedan agregar **obligaciones** a las políticas de privacidad. XACML sólo nos permite asociar obligaciones a **políticas** y **conjuntos de políticas**

con lo que tendremos que los usuarios de CERTILOC sólo podrán asociar obligaciones con alguna de estas dos.

3.5.1.4.2 Elementos de CERTILOC y elementos de XACML

Para realizar un buen uso del estándar XACML, es necesario comprender qué atributos a considerar en las políticas y peticiones de CERTILOC están actualmente contemplados en el estándar XACML y cuáles no.

Para los parámetros o atributos no considerados en XACML, que existan en CERTILOC, XACML nos obliga a declarar un nombre de emisor nuevo. Dado que el nombre debe cumplir la especificación URI (The Internet Society 1998), se designa el siguiente emisor para atributos no considerados en XACML: ***urn:certiloc:1.0:issuer:admin@certiloc.com***.

Tal y como se indicaba en el requisito **CERTILOC-PP-RFI-003**, los usuarios de CERTILOC deben poder especificar los siguientes elementos en sus políticas de privacidad: **id del usuario** que solicita la petición (usuario solicitante de servicio), **rol del usuario** que solicita la petición, **acción** que se desear realizar, parámetros de designación del **localización o IET** del dispositivo, identificador del **dispositivo** sobre el que se solicita la petición, **hora** de la **ejecución** de la petición de autorización, **fecha** de la **ejecución** de la petición de autorización, **hora** del momento de la recogida de la **IET**, **fecha** del momento de la recogida de la **IET**.

A continuación mostramos una serie de tablas que indican qué atributos de XACML se han escogido para determinar los atributos de CERTILOC. La tabla contendrá los siguientes datos:

Concepto de CERTILOC	Atributo a considerar en CERTILOC
Atributo XACML escogido	Atributo de XACML escogido para el concepto a designar
Tipo de atributo	Indicará el tipo XACML del atributo tratado (en qué grupo de atributos se encuentra : Subject (de sujeto), Action (de acción), Environment (de entorno) y Resource (de recurso)
Ubicación del atributo	Dónde se podrán incluir estos atributos
Identificador XACML utilizado o creado	El identificador XACML designado para este atributo
Emisor utilizado	Para conceptos CERTILOC no considerados en XACML, el emisor XACML designado para este atributo

Tabla 90. *Especificación de conceptos CERTILOC contra atributos XACML*

Pasamos a continuación a ver cada una de las equivalencias entre conceptos de CERTILOC y atributos XACML.

Concepto de CERTILOC	Identificador del usuario solicitante de servicios
Atributo XACML escogido	**subject-id**
Tipo de atributo	Subject
Ubicación del atributo	Conjuntos de políticas, Políticas, Reglas, Condiciones o Aplicativos
Identificador XACML utilizado o creado	**urn:oasis:names:tc:xacml:1.0:subject: subject-id**
Emisor utilizado	Ninguno (existe en XACML)

Tabla 91. *Concepto: Identificador del usuario solicitante de servicios en XACML*

Concepto de CERTILOC	Rol CERTILOC del usuario del usuario solicitante de servicios
Atributo XACML escogido	**No existe en XACML**
Tipo de atributo	Subject
Ubicación del atributo	Conjuntos de políticas, Políticas, Reglas, Condiciones o Aplicativos
Identificador XACML utilizado o creado	**urn:certiloc:names:tc:xacml:1.0:subject:subject-role**
Emisor utilizado	urn:certiloc:1.0:issuer:admin@certiloc.com

Tabla 92. *Concepto: Identificador del usuario solicitante de servicios en XACML*

Concepto de CERTILOC	Acción a realizar mediante el servicio utilizado
Atributo XACML escogido	**action-id**
Tipo de atributo	Action
Ubicación del atributo	Conjuntos de políticas, Políticas, Reglas, Condiciones o Aplicativos
Identificador XACML utilizado o creado	**urn:certiloc:1.0:action:action-id**
Emisor utilizado	Ninguno (existe en XACML)

Tabla 93. *Concepto: Acción a realizar en XACML*

Concepto de CERTILOC	Localización de un determinado dispositivo
Atributo XACML escogido	**No existe en XACML**
Tipo de atributo	Environment
Ubicación del atributo	Condiciones o Aplicativos
Identificador XACML utilizado o creado	**urn:certiloc:1.0:action:environment:localizacion-iet**
Emisor utilizado	urn:certiloc:1.0:issuer:admin@certiloc.com

Tabla 94. *Concepto: Localización en XACML*

Concepto de CERTILOC	Identificador del dispositivo sobre el que se solicita la petición
Atributo XACML escogido	**resource-id**
Tipo de atributo	Resource
Ubicación del atributo	Conjuntos de políticas, Políticas, Reglas, Condiciones o Aplicativos
Identificador XACML utilizado o creado	**urn:oasis:names:tc:xacml:1.0:resource:resource-id**
Emisor utilizado	Ninguno (existe en XACML)

Tabla 95. *Concepto: Identificador del dispositivo objetivo de servicio en XACML*

Concepto de CERTILOC	Fecha y hora de la ejecución de servicio CERTILOC (de la ejecución de la petición de autorización)
Atributo XACML escogido	**No existe en XACML**
Tipo de atributo	Environment
Ubicación del atributo	Condiciones o Aplicativos
Identificador XACML utilizado o creado	**urn:certiloc:1.0:action:environment: fecha-peticion** **urn:certiloc:1.0:action:environment: hora-peticion**
Emisor utilizado	urn:certiloc:1.0:issuer:admin@certiloc.com

Tabla 96. *Concepto: Fecha y hora de ejecución de servicio en XACML*

Concepto de CERTILOC	Fecha y hora del momento de recoger la IET
Atributo XACML escogido	**No existe en XACML**
Tipo de atributo	Environment
Ubicación del atributo	Condiciones o Aplicativos
Identificador XACML utilizado o creado	**urn:certiloc:1.0:action:environment: fecha-peticion-iet** **urn:certiloc:1.0:action:environment: hora-peticion-iet**
Emisor utilizado	urn:certiloc:1.0:issuer:admin@certiloc.com

Tabla 97. *Concepto: Fecha y hora de ejecución de servicio en XACML*

Por otro lado, el estándar XACML contempla 4 tipos de respuestas ante una petición de autorización: Permitir, Denegar, No Aplicable ("Not Applicable") e Indeterminado ("Indeterminate"), sin embargo CERTILOC sólo considera 2 tipos de respuestas: Verdadero (Permitir) y Falso (Denegar). Para convertir los valores de las respuestas de autorización del formato XACML al formato nativo de CERTILOC, consideramos las siguientes equivalencias:

Valor de la respuesta en formato XACML	Equivalente en CERTILOC
Permitir	Verdadero (Permitir)
Denegar	Falso (Denegar)
No Aplicable	Falso (Denegar)
Indeterminado	Falso (Denegar)

Tabla 98. *Equivalencia entre valores de respuestas XACML frente a CERTILOC*

En cuanto a las funciones condicionales que manejan la información de políticas de privacidad y peticiones, cabe destacar que el estándar XACML incluye un gran número de funciones para aplicar sobre dicha información. Dado que la herramienta intermedia o API para el manejo de datos en formato XACML ya debe contener estas funciones, en un principio sólo será necesario ampliar el estándar con dos nuevas funciones (no contempladas en el estándar XACML).

CERTILOC exige la creación de al menos dos funciones condicionales que manejen información de peticiones y políticas: Una debe estar orientada al manejo de información espacial o de localización y la otra al manejo de información temporal (fechas u horas) tal y como describe el requisito **CERTILOC-PP-RFO-014**.

Por un lado, crearemos una función denominada **Localización dentro del área de un rectángulo.** Esta función debe recibir un argumento de localización y comprobar si éste se encuentra dentro del rectángulo definido por otros dos puntos en el plano que definen el punto superior derecho o cota superior y el punto inferior izquierdo o cota inferior del rectángulo. El identificador XACML a asignar a esta función será el siguiente:

- **urn:certiloc:1.0:function:location-in-rectangle-area**

Por otro lado, se incluirá la función **hora dentro de rango de horas**. Esta función permitirá evaluar si cierta hora, pasada por parámetro, se encuentra en el rango de otras dos horas. Por ejemplo, para evaluar si la hora 9:00 am está contenida en el rango de las 8:00 am a las 21:00 pm, la función devolverá verdadero. Sin embargo si recibe como parámetro las 7:30 am, devolverá falso ya que esta hora no se encuentra en el rango entre las 8 de la mañana y las 9 de la noche. El identificador XACML a asignar a esta función será el siguiente:

- **http://research.sun.com/projects/xacml/names/function#time-in-range**

Por último, dado que CERTILOC requiere que se pueda hacer un seguimiento de la actividad de las políticas en el sistema, tendremos que recurrir a la utilización de algoritmos de combinación de políticas (en conjuntos de políticas) y de combinación de reglas (en las políticas). XACML no incluye ningún algoritmo de combinación que permita registrar un seguimiento de la actividad de cierta política o regla por lo que tendremos que crear nuestros propios algoritmos de combinación.

En el caso de CERTILOC, incluiremos dos nuevos algoritmos de combinación que estarán basados en el algoritmo de permitir prevalece (“Permit Overrides”) de XACML para registrar la actividad de reglas y políticas. En el caso de combinación de políticas utilizaremos el siguiente nombre XACML para identificar éste algoritmo:

- **urn:certiloc:1.0:policy-combining-algorithm:permit-overrides**

Para la combinación de reglas utilizaremos el siguiente identificador XACML para el algoritmo:

- **urn:certiloc:1.0:rule-combining-algorithm:permit-overrides**

3.5.1.4.3 Adaptación de los modelos de flujo de datos de XACML y de CERTILOC

En el caso de la adaptación del flujo de datos de CERTILOC al de XACML, la herramienta intermedia o API no será tan determinante. Debemos tener en cuenta que el API a utilizar, sólo nos será útil para manejar los datos de peticiones, respuestas y políticas en el contexto o formato de XACML, sin embargo, el modelo de flujo de datos de XACML también contempla el intercambio de información y datos entre distintos agentes en el formato nativo del Sistema (no en el formato de XACML).

En esta fase, tenemos todos los datos necesarios para poder crear una integración de los módulos propuestos en la arquitectura general de CERTILOC y los distintos agentes que propone el estándar XACML.

La Figura 13 presenta el modelo de flujo de datos de XACML frente a los distintos módulos propuestos para la arquitectura del SPP de CERTILOC:

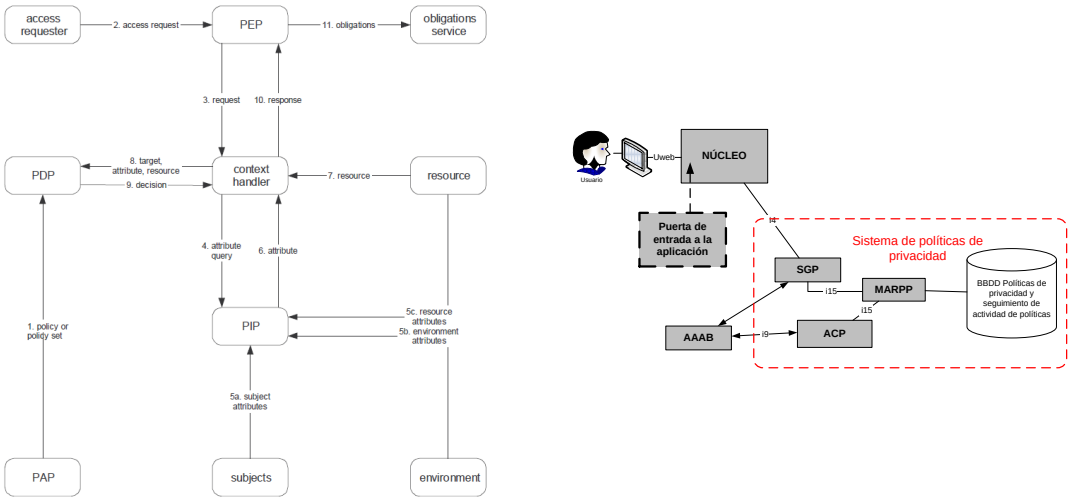


Figura 13. *Visión global de la arquitecturas del sistema de políticas de privacidad y el modelo de flujo de datos de XACML*

Recordamos que el modelo de flujo de datos de XACML plantea un cambio de contexto entre el lenguaje del modelo de datos del sistema original donde se ubica el sistema de políticas de privacidad y el lenguaje XACML.

En este caso, con la arquitectura planteada de antemano para el sistema de políticas de privacidad de CERTILOC, se estima que faltará un módulo que traduzca entre los distintos contextos (XACML y CERTILOC). Por lo tanto se debe introducir un nuevo módulo denominado **AGPA**, encargado de la traducción de peticiones, respuestas de autorización del formato nativo de CERTILOC al formato XACML y viceversa. Para traducir políticas de privacidad del

formato nativo de CERTILOC al formato de XACML, se delega en el módulo MARPP (Módulo de Acceso al Repositorio de Políticas de Privacidad).

Por lo tanto, la arquitectura final del módulo de políticas de seguridad se modificará y se integrará el modelo de flujo de datos de XACML con el de CERTILOC de la siguiente manera:

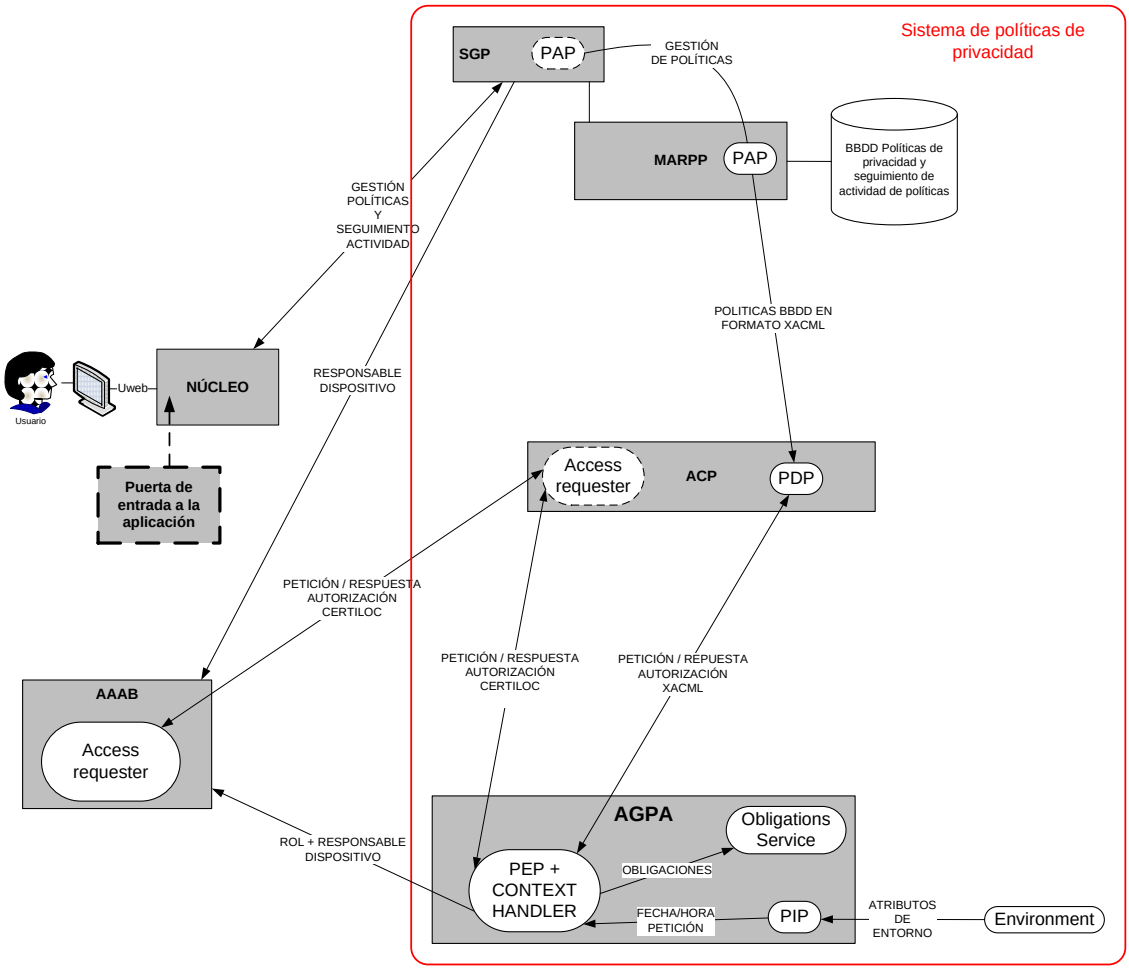


Figura 14. *Arquitectura de módulos propuesta para CERTILOC con SPP*

Se presenta a continuación una tabla donde se indica la ubicación de cada uno de los agentes del estándar XACML repartidos en la arquitectura de módulos de CERTILOC y la razón para esta ubicación.

Agente XACML	Módulo SPP	Razón para la asociación
Access requester	AAAB ACP	Tanto el agente XACML como el módulo de CERTILOC,, es el que genera la petición inicial de autorización
PEP	AGPA	Tanto el agente XACML como el módulo de CERTILOC, es el encargado de recibir la petición de autorización en formato nativo, traducirla al formato de XACML y pasarla al evaluador (PDP + CONTEXT HANDLER)
PDP	ACP	Tanto el agente XACML como el módulo de CERTILOC, es el encargado de evaluar cierta petición de autorización contra las políticas de privacidad del sistema
PAP	SGP MARPP	Tanto el agente XACML como el módulo de CERTILOC, es el punto donde se administran y generan de las políticas de privacidad. Dado que pertenece a dos módulos distintos en CERTILOC no se utilizarán las siglas presentadas en XACML para designar ningún sub-módulo del sistema de políticas
Obligations service	AGPA	Tanto el agente XACML como el módulo de CERTILOC, es el que, en última instancia, gestiona las obligaciones que implica cierto acceso a la información o recurso solicitados
Context handler	AGPA	Tanto el agente XACML como el módulo de CERTILOC es el encargado de ponerse en contacto con el PDP, ubicado en el ACP en CERTILOC
PIP	AGPA	El PIP es el encargado de recoger datos sobre el entorno de la petición (fecha y hora de ejecución de la petición en XACML) y el AGPA es el encargado de crear peticiones en formato XACML por lo tanto incluimos en ese módulo un PIP que informe de la fecha y la hora actuales al realizarse la petición

Tabla 99. *Aproximación entre el modelo de flujo de datos de XACML y la arquitectura de CERTILOC*

Para comprender completamente la Figura 14, explicamos aquí la motivación para las interacciones de los módulos del **SPP** (SGP, AGPA, ACP) **hacia** el **AAAB.** Son necesarias para

cumplimentar la siguiente información a la hora de evaluar las peticiones de autorización y de crear registros de actividad de políticas de privacidad:

- $\mathbf{U_{Role}}$ representa el concepto de rol de usuario. Tal y como hemos definido anteriormente en el presente documento, un rol de usuario es una agrupación de usuarios que determina la motivación de la utilización de la información accedida mediante el sistema de localización. El sistema de políticas de privacidad o SPP debe conocer, antes de llevar a cabo la evaluación de autorización, el rol que existe entre dos usuarios. Esta información está custodiada por el módulo AAAB.
- **Responsable del dispositivo**. En este caso, el SPP debe conocer quién es el usuario responsable del dispositivo para el que se está solicitando la petición de autorización. Una vez más, esta información está custodiada por el módulo AAAB.

3.5.2 APACHE STRUTS

STRUTS es una herramienta de soporte para el desarrollo de aplicaciones Web bajo el patrón ***MVC*** (Modelo – Vista – Controlador) (MVC - Sun Microsystems , Inc. 2009) que actúa siempre bajo la plataforma J2EE (Java 2, Enterprise Edition).

STRUTS nos va a permitir reducir, considerablemente, el tiempo de desarrollo de la interfaz web de usuario. Además, la compatibilidad de STRUTS con todas las plataformas en que Java Enterprise está disponible y su carácter de "software abierto o libre" (sus licencias son completamente gratuitas y el código fuente es accesible públicamente y modificable) hacen que esta herramienta sea altamente disponible.

STRUTS se basa en el patrón de diseño **MVC** (Modelo Vista Controlador). Este patrón de diseño está ampliamente extendido y cada vez se utiliza con mayor frecuencia.

Un problema común cuando se desarrollan aplicaciones bajo el patrón de diseño **MVC** es la creación de "controladores amplios" o "fat-controllers". Es decir, se utiliza un solo controlador para delegar toda la lógica de negocio y las peticiones de usuario, en vez de utilizar varios. Al utilizar un solo controlador, todo el desacoplamiento de la aplicación se puede volver en nuestra contra creando un acoplamiento fuerte en este componente (Struts - Apache Software Foundation 2009). Aquí es donde STRUTS encuentra su punto fuerte: Nos ayuda a desacoplar todo el control de la lógica de negocio mediante la creación de un varios controladores de peticiones ("Action Servlet") que evalúa y decide las acciones a tomar haciendo uso de un archivo XML configurable "struts-config.xml".

Para una explicación más extensa de STRUTS y el patrón de diseño MVC recomendamos al lector acceder al resto de documentación del proyecto de demostrador de CERTILOC (Memoria PFC - Calvo Martínez 2007).

3.6 PRUEBAS DE ACEPTACIÓN DEL SISTEMA

Se presenta a continuación los detalles de las pruebas de aceptación del sistema. En primer lugar se define el formato para la presentación de las pruebas y más adelante se ve el plan completo de pruebas de aceptación.

3.6.1 ESPECIFICACIÓN DEL PLAN DE PRUEBAS DE ACEPTACIÓN

Se define a continuación el plan de pruebas de aceptación del sistema de políticas de CERTILOC.

El plan de pruebas de aceptación está orientado a dar por válido el software desarrollado desde el punto de vista de cliente, es decir, las pruebas de aceptación deben ser aprobadas con éxito (requisitos CERTILOC-PP-RNFV-001 y CERTILOC-PP-RNFA-001) para verificar que el funcionamiento de la aplicación es el esperado por el cliente.

Cada una de las pruebas presentadas en el plan tendrá forma tabular y contendrá los siguientes datos:

Identificador	Identificador único de la prueba en cuestión
Propósito	Propósito de la prueba
Pasos	Los pasos a seguir para la realización de la prueba
Salida o estado esperado	Indica el resultado que debemos obtener o el estado del sistema, una vez realizados los pasos de la prueba.

Tabla 100. *Formato para la especificación de pruebas del sistema*

3.6.2 PLAN DE PRUEBAS DE ACEPTACIÓN

Se definen las siguientes pruebas de aceptación.

Identificador	CERTILOC-PP-PA-001
Propósito	Probar que se pueden crear políticas de privacidad en el sistema
Pasos	• Entrar en el sistema con un usuario responsable de dispositivos • Dirigirse al sistema de gestión de políticas de privacidad • Dar de alta un nuevo conjunto de políticas cuyo objetivo sea alguno de nuestros dispositivos
Salida o estado esperado	La política debe estar creada en el sistema y asociada al usuario con el que se dio de alta.

Tabla 101. *Prueba de aceptación CERTILOC-PP-PA-001*

Identificador	CERTILOC-PP-PA-002
Propósito	Probar que se pueden crear políticas que evalúen a Permitir y que. cuando están activas, permiten las peticiones de autorización que reciben
Pasos	• Entrar en el sistema con un usuario que tenga alguna política dada de alta y activa en el sistema • Crear una regla, con unos parámetros concretos, para la política y cuyo resultado sea Permitir • Entrar en el sistema con un usuario distinto • Hacer una petición que cumpla con las condiciones definidas en la política y regla anteriores
Salida o estado esperado	Se debe autorizar la operación que se está intentando ejecutar

Tabla 102. *Prueba de aceptación CERTILOC-PP-PA-002*

Identificador	CERTILOC-PP-PA-003
Propósito	Probar que se pueden crear políticas que evalúen a Denegar y que. cuando están activas, no permiten las peticiones de autorización que reciben
Pasos	• Entrar en el sistema con un usuario que tenga alguna política dada de alta y activa en el sistema • Crear una regla, con unos parámetros concretos, para la política y cuyo resultado sea Denegar • Entrar en el sistema con un usuario distinto • Hacer una petición que cumpla con las condiciones definidas en la política y regla anteriores
Salida o estado esperado	No se debe autorizar la operación que se está intentando ejecutar

Tabla 103. *Prueba de aceptación CERTILOC-PP-PA-003*

Identificador	CERTILOC-PP-PA-004
Propósito	Probar que se pueden crear políticas de privacidad con todos los parámetros descritos en los requisitos de operación de la aplicación.
Pasos	• Entrar en el sistema con un usuario responsable de algún dispositivo • Crear un conjunto de políticas con políticas, reglas (deben evaluar Permitir), condiciones y aplicativos que tengan en cuenta los siguientes parámetros: ○ **Identificador del usuario** que realiza la petición ○ **Rol del usuario** que realiza la petición ○ **Hora de la creación** de la petición ○ **Fecha de la creación** de la petición ○ **Hora de la realización** de la petición ○ **Fecha de la realización** de la petición ○ **Operación** que se quiere llevar a cabo en la petición ○ **Recurso** sobre el que se quiere realizar la operación de la petición ○ **Localización** del recurso sobre el que se solicita la operación de la petición • Entrar al sistema con otro usuario y generar distintas peticiones que cumplan con todos los parámetros definidos en el anterior conjunto de políticas.
Salida o estado esperado	Se deben autorizar todas las peticiones realizadas

Tabla 104. *Prueba de aceptación CERTILOC-PP-PA-004*

Identificador	CERTILOC-PP-PA-005
Propósito	Probar que las actividades de las políticas de privacidad creadas generan registros de actividad cuando son las responsables de una respuesta de autorización frente a una petición.
Pasos	• Entrar al sistema con un usuario responsable de algún dispositivo • Crear un conjunto de políticas cuyo objetivo sea un dispositivo concreto del usuario en cuestión • Crear una política asociada al anterior conjunto de políticas cuyo objetivo sea la acción "**ObtenerIET**" • Crear una regla que evalúe "**Permitir**" para todo aquel usuario que ostente el rol de "**tutor**" con respecto al usuario con el que creamos el conjunto de políticas • Entrar al sistema con un usuario que ostente el rol de "**tutor**" con respecto al usuario para el que hemos creado el conjunto de políticas anterior • Llevar a cabo una petición de "**ObtenerIET**" sobre el dispositivo para el cual se ha definido el conjunto de políticas anterior • Salir del sistema y volver a iniciar sesión con el usuario con el que creamos el conjunto de políticas inicial • Entrar al sistema de gestión de políticas de privacidad y dirigirnos a la sección reservada a registros de actividad de políticas
Salida o estado esperado	• El usuario que solicita la operación debe obtener una respuesta de autorización que evalúe a Permitir • Se deben haber creado registros de actividad con la fecha y hora de la solicitud de la petición y que reflejen correctamente el conjunto de políticas creado, la solicitud de petición recibida y la respuesta devuelta por el sistema de políticas de privacidad

Tabla 105. *Prueba de aceptación CERTILOC-PP-PA-005*

Identificador	CERTILOC-PP-PA-006
Propósito	• Probar que se pueden activar y desactivar políticas de privacidad. • Probar que cuando una política está en estado “inactivo” no se aplica frente a peticiones de autorización. • Probar que cuando una política está en estado “activo” se aplica frente a peticiones de autorización.
Pasos	• Entrar al sistema con un usuario (lo llamaremos usuarioA) responsable de algún dispositivo que tenga un solo conjunto de políticas asociado y dado de alta en el sistema, que contenga una regla que evalúe “Permitir” y cuyo estado sea “inactivo” • Entrar en el sistema con otro usuario (lo llamaremos usuarioB)y realizar una petición que cumpla todas las condiciones definidas en el conjunto de políticas anterior • Entrar de nuevo al sistema con el primer usuario (usuarioA) y poner el estado del único conjunto de políticas a “activo” • Entrar en el sistema con otro usuario (usuarioB) y realizar una petición que cumpla todas las condiciones definidas en el conjunto de políticas anterior • Entrar de nuevo al sistema con el primer usuario (usuarioA) y dirigirse al sistema de gestión de políticas de privacidad, al apartado reservado a registros de actividad de políticas
Salida o estado esperado	• Se debe autorizar la segunda petición realizada por el usuarioB • No se debe autorizar la primera petición realizada por el usuarioB. • Se debe comprobar, con el usuarioA, que sólo se ha creado un registro de actividad de políticas para la segunda petición del usuarioB

Tabla 106. *Prueba de aceptación CERTILOC-PP-PA-006*

Identificador	CERTILOC-PP-PA-007
Propósito	Probar que se pueden borrar políticas de privacidad en el sistema
Pasos	• Entrar en el sistema con un usuario responsable de algún dispositivo que tenga alguna política de privacidad dada de alta en el sistema • Dirigirse al sistema de gestión de políticas de privacidad • Dar de baja o borrar cualquier conjunto de políticas
Salida o estado esperado	La política de privacidad que se ha borrado no debe estar presente dentro de los conjuntos de políticas de privacidad del usuario.

Tabla 107. *Prueba de aceptación CERTILOC-PP-PA-007*

Identificador	CERTILOC-PP-PA-008
Propósito	Probar que se pueden consultar las políticas de privacidad dadas de alta en el sistema.
Pasos	• Entrar en el sistema con un usuario responsable de algún dispositivo que tenga alguna política de privacidad dada de alta en el sistema • Dirigirse al sistema de gestión de políticas de privacidad • Navegar por las políticas de privacidad creadas dadas de alta en el sistema
Salida o estado esperado	Poder ver y navegar por las políticas de privacidad asociadas al usuario en cuestión.

Tabla 108. *Prueba de aceptación CERTILOC-PP-PA-008*

Identificador	CERTILOC-PP-PA-009
Propósito	Probar que se pueden obtener una visión XACML de cualquier conjunto de políticas.
Pasos	• Entrar en el sistema con un usuario responsable de algún dispositivo que tenga alguna política de privacidad dada de alta en el sistema • Dirigirse al sistema de gestión de políticas de privacidad • Entrar en un conjunto de políticas cualquiera • Convertir el conjunto de políticas a formato XACML
Salida o estado esperado	Poder ver en formato XACML el conjunto de políticas seleccionado.

Tabla 109. *Prueba de aceptación CERTILOC-PP-PA-009*

Identificador	CERTILOC-PP-PA-010
Propósito	Probar que se puede modificar cualquier parámetro de las políticas de privacidad.
Pasos	• Entrar en el sistema con un usuario responsable de algún dispositivo que tenga alguna política de privacidad dada de alta en el sistema • Dirigirse al sistema de gestión de políticas de privacidad • Entrar en un conjunto de políticas cualquiera • Modificar distintos parámetros del conjunto de políticas en cuestión
Salida o estado esperado	• El parámetro modificado debe quedar guardado en el sistema. • El resto de parámetros del conjunto de políticas debe mantenerse como estaba antes de realizar esta prueba

Tabla 110. *Prueba de aceptación CERTILOC-PP-PA-010*

Identificador	CERTILOC-PP-PA-011
Propósito	Probar que se pueden asociar obligaciones a cualquier política de privacidad.
Pasos	• Entrar en el sistema con un usuario responsable de algún dispositivo que tenga alguna política de privacidad dada de alta en el sistema • Dirigirse al sistema de gestión de políticas de privacidad • Entrar en un conjunto de políticas cualquiera • Asociar una obligación al conjunto de políticas de privacidad en cuestión
Salida o estado esperado	• La obligación debe quedar asociada al conjunto de políticas seleccionado y guardada en el sistema.

Tabla 111. *Prueba de aceptación CERTILOC-PP-PA-011*

Identificador	CERTILOC-PP-PA-012
Propósito	Probar que los usuarios Administradores pueden acceder a los ficheros de registros de funcionamiento del sistema de políticas de privacidad.
Pasos	• Entrar en el sistema con un usuario considerado Administrador del sistema • Dirigirse a la zona de registros de funcionamiento del sistema • Acceder a la zona de registros de funcionamiento del sistema de políticas de privacidad • Descargar cualquier fichero de registro de funcionamiento asociado a alguno de los módulos del sistema de políticas de privacidad
Salida o estado esperado	• Comprobar que el fichero descargado contiene información sobre el funcionamiento del módulo seleccionado.

Tabla 112. *Prueba de aceptación CERTILOC-PP-PA-012*

Identificador	CERTILOC-PP-PA-013
Propósito	Probar que la aplicación permite crear por lo menos una función que evalúe parámetros relacionados con el tiempo y otra que evalúe parámetros relacionados con la localización.
Pasos	• Entrar en el sistema con un usuario que sea responsable de algún dispositivo • Dirigirse al sistema de gestión de políticas de privacidad • Crear un conjunto de políticas cuyo objetivo sea uno de los dispositivos del usuario • Crear una política dentro del conjunto de políticas anterior cuyo objetivo sea la operación “Obtener IET” • Crear una regla para la política anterior, que evalúe a Permitir y que no contenga objetivos. Se debe crear una condición que utilice la función “LocationInRectangleArea” que debe recibir como primer parámetro la localización IET de la petición, como segundo parámetro la coordenada “00,00,00” y como tercer parámetro la coordenada “100,100,00” • Crear otra regla para la política, que evalúe a Permitir y que no contenga objetivos. Se debe crear una condición que utilice la función “TimeInRangeFunction” que debe recibir como primer parámetro la hora de creación de la petición, como segundo parámetro la hora “08:00:00” y como tercer parámetro la hora “21:00:00” • Entrar al sistema con otro usuario y realizar una petición de “Obtener IET” para el dispositivo del conjunto de políticas definido, entre las 08:00:00 y las 21:00:00 • Realizar otra petición de “Obtener IET” para el mismo dispositivo (es necesario que el dispositivo se encuentre dentro del rectángulo definido por las coordenadas 00,00,00 y 100,100,00)
Salida o estado esperado	• Las dos peticiones deben autorizarse

Tabla 113. *Prueba de aceptación CERTILOC-PP-PA-013*

Identificador	CERTILOC-PP-PA-014
Propósito	Probar que la aplicación no permite crear políticas de privacidad que tengan en cuenta dispositivos que no sean responsabilidad del usuario que ha ingresado en la aplicación
Pasos	• Entrar en el sistema con un usuario que sea responsable de algún dispositivo • Dirigirse al sistema de gestión de políticas de privacidad • Crear un conjunto de políticas que tenga como objetivo un dispositivo que no pertenezca al usuario que ha ingresado en la aplicación
Salida o estado esperado	• La aplicación debe devolver un error indicando que el dispositivo para el que se intenta crear el objetivo del conjunto de políticas no es responsabilidad del usuario

Tabla 114. *CERTILOC-PP-PA-014*

Identificador	CERTILOC-PP-PA-015
Propósito	Probar que la aplicación borra todas las políticas de privacidad asociadas a un usuario cuando este es borrado del sistema.
Pasos	• Entrar en el sistema con un usuario que sea administrador • Dirigirse al sistema de gestión de usuarios • Borrar un usuario
Salida o estado esperado	• La aplicación debe borrar todas las políticas de privacidad existentes en el sistema y que pertenezcan al usuario borrado

Tabla 115. *CERTILOC-PP-PA-015*

4 DISEÑO DEL SISTEMA

Vemos, en el siguiente capítulo, el diseño de alto y bajo nivel del sistema de políticas de CERTILOC.

El análisis del sistema nos ha permitido comprender cuál es exactamente el problema a resolver. Por otro lado, hemos extraído las pautas generales en cuanto a cómo se debe resolver el problema y las restricciones que debemos cumplir para conseguir nuestro objetivo.

El diseño del sistema nos va a ofrecer una guía de cómo implementar dicho sistema de políticas de privacidad. El diseño que se presenta está dividido en dos grandes áreas: El diseño arquitectónico y el diseño de componentes de bajo nivel.

El diseño arquitectónico del sistema de políticas nos va a indicar cómo fusionar el diseño arquitectónico de los módulos de CERTILOC (los que conforman el sistema de políticas de privacidad de CERTILOC) con el modelo de flujo de datos de XACML.

Con el diseño detallado, veremos detalladamente cada componente de software a desarrollar. Para cada uno de ellos veremos: El diseño del paquete general que conforma el componente, el diseño de cada uno de los elementos que forman parte del paquete y los diagramas de secuencia de las funciones más destacadas de cada componente.

El capítulo se estructura de la siguiente manera:

En primer lugar se ve una breve descripción de los distintos patrones de diseño utilizados para el diseño del sistema.

En segundo lugar se puede ver el diseño arquitectónico de la aplicación, donde se aprecian, entre otros, el flujo de los datos entre los distintos módulos a desarrollar y el diseño de componentes de los mismos.

Seguidamente, se presentará el modelo de datos que utilizará el sistema de políticas de privacidad que muestra, entre otros, una especificación de alto nivel de cada una de las clases de dicho modelo.

A continuación se presentará una explicación de las distintas tecnologías escogidas para la implementación del SPP. Cabe destacar que se incluye aquí la elección de la herramienta intermedia o API para el manejo de datos de XACML y la evaluación de peticiones de autorización. Con su explicación, también se incluyen una serie de adaptaciones que ha tenido que sufrir el estándar para adaptarlo al modelo de negocio de CERTILOC.

Por último, se ve una especificación del diseño detallado del sistema.

4.1 PATRONES DE DISEÑO UTILIZADOS

Para poder crear un diseño reutilizable, ampliable y comprensible, se ha decidido incorporar varios patrones de diseño que nos ayudarán a resolver varios problemas concretos.

A continuación se describen cada uno de los patrones utilizados y el problema que nos ayudan a resolver. Cabe remarcar que las descripciones han sido extraídas del libro Head First: Design Patterns (Freeman 2004):

- **Singleton:** El patrón de diseño **singleton** (instancia única) está diseñado para restringir la creación de objetos pertenecientes a una clase. Su intención consiste en garantizar que una clase sólo tenga una instancia y proporcionar un punto de acceso global a ella. En el caso del sistema de políticas de privacidad de CERTILOC**, singleton** nos ayudará a crear una única instancia de los subsistemas que contiene (ACPSystem, AGPASystem, ACPSystem y MARPPSystem). De esta manera, crearemos un diseño que permita una posible implementación distribuida del sistema.
- **Facade**: El patrón de diseño Facade provee una interfaz unificada sencilla un conjunto de interfaces de un subsistema. Esta interfaz hará de intermediaria entre un cliente y una interfaz o grupo de interfaces más complejas. En el caso de CERTILOC, todos los sistemas (ACPSystem, AGPASystem, ACPSystem y MARPPSystem) del sistema de políticas de privacidad serán accedidos por una interfaz más sencilla.
- **Modelo – Vista - Controlador:** El objetivo de **MVC** (MVC - Sun Microsystems , Inc. 2009) es desacoplar el desarrollo de una aplicación dividiéndolo en las siguientes capas: la **vista** de la aplicación (interfaz de usuario), el **modelo de datos** que la soporta y los distintos **controladores** que manejan los eventos y las acciones de la aplicación y mueven la información del modelo de datos.

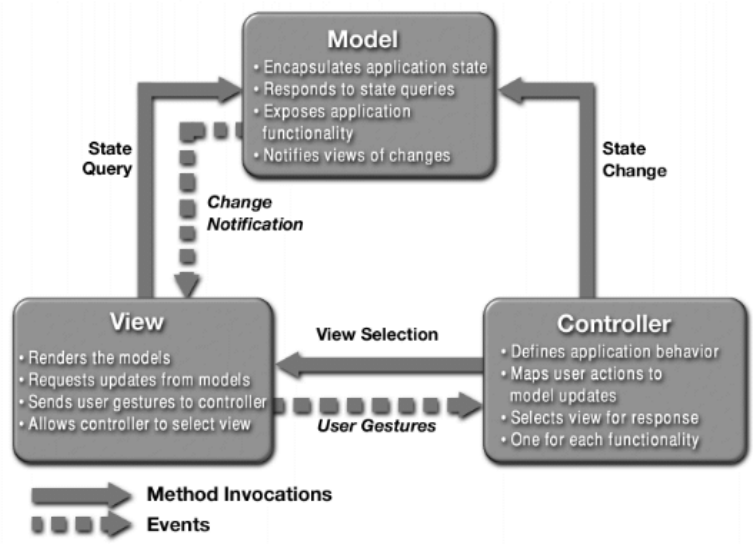


Figura 15. *Esquema del patrón de diseño MVC (MVC - Sun Microsystems , Inc. 2009)*

4.2 DISEÑO ARQUITECTÓNICO

Tal y como hemos explicado en la introducción, el diseño arquitectónico nos va a permitir definir cómo vamos a fusionar el modelo de flujo de datos de XACML con la especificación de diseño arquitectónico de los módulos que conforman el sistema de políticas de privacidad de CERTILOC.

Veamos a continuación un diagrama de alto nivel de los módulos a desarrollar para el sistema de políticas de privacidad de CERTILOC.

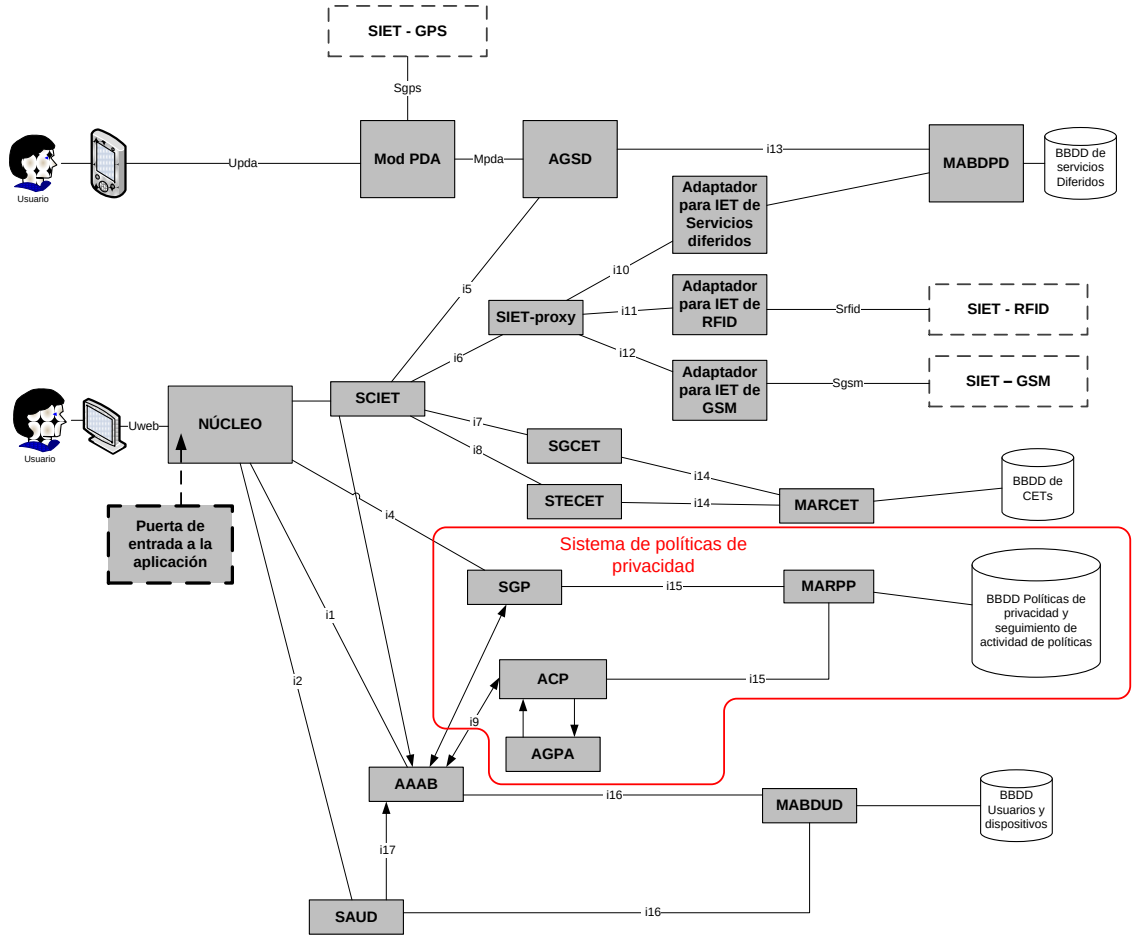


Figura 16. *Detalle de la arquitectura de CERTILOC dividida en módulos*

Los principales módulos que pueden identificarse en la arquitectura para el sistema son:

- **AAAB** Agente de Autenticación y Autorización Básica
- **ACP** Agente Custodio de la Privacidad
- **AGSD** Agente Gestor de Servicios Diferidos
- **AGPA** Agente Gestor de Peticiones de Acceso
- $\mathbf{i_n}$ Interfaces de comunicación entre los módulos componentes de CERTILOC
- **MABDSD** Módulo de Acceso a la Base de Datos de Servicios Diferidos
- **MABUD** Módulo de Acceso a la Base de Datos de Usuarios y Dispositivos
- **MARCET** Módulo de Acceso al Repositorio de Certificados Espacio-Temporales
- **MARPP** Módulo de Acceso al Repositorio de Políticas de Privacidad

- **Mód. PDA** Módulo para el PDA
- **RCET** Repositorio de Certificados Espacio-Temporales
- **RPP** Repositorio de Políticas de Privacidad
- **SAUD** Servicio de Administración de Usuarios y Dispositivos
- **SCIET** Servicio de Certificación e Información Espacio-Temporal
- **SGCET** Servicio de Generación de Certificados Espacio-Temporales
- **SGP** Servicio de Gestión de la Privacidad
- **SIET-GPS** Servicio de Información Espacio-Temporal basado en GPS
- **SIET-GSM** Servicio de Información Espacio-Temporal basado en redes GSM
- **SIET-proxy** Proxy para los Servicios de Información Espacio-Temporal
- **SIET-RFID** Servicio de Información Espacio-Temporal basado en sistemas RFID
- **STECET** Servicio de Transferencia y Eliminación de Certificados Espacio-Temporales

En la Figura 16 vemos como el usuario accede directamente al módulo del Sistema de Gestión de Políticas de privacidad (SGP) mediante la interfaz Uweb (requisito CERTILOC-PP-RNFI-001).

Tal y como vemos en la Figura 16, y en orden de flujo de datos, el módulo SGP se comunica con el Módulo de Acceso al Repositorio de Políticas de Privacidad (MARPP) para mostrar y escribir los datos de sus políticas de privacidad y de registros de actividad de políticas.

Por otro lado, el módulo AAAB (Memoria PFC - Gallo Martínez 2008) se comunica con el módulo Agente Custodio de Políticas de Privacidad (ACP) para realizar peticiones de autorización en el formato nativo de CERTILOC.

El módulo ACP se comunica con el módulo Agente Gestor de Peticiones de Acceso (AGPA) para convertir las peticiones de autorización desde el formato nativo de CERTILOC al formato Nativo de XACML. Una vez en formato XACML, el módulo AGPA devuelve la petición en formato XACML al ACP para que realice la validación contra las políticas albergadas en el sistema. Para extraer las políticas activas en el sistema de políticas de privacidad, el módulo ACP se comunica con el módulo MARPP para solicitar las políticas de privacidad activas en el sistema. El módulo devuelve dichas políticas al ACP para realizar la evaluación. Una vez obtiene

una respuesta, el módulo ACP devuelve la respuesta en formato XACML al módulo AGPA. Éste, convierte la respuesta al contexto o formato nativo de CERTILOC y devuelve la respuesta convertida al ACP que pasa a devolver la respuesta en contexto CERTILOC de vuelta al AAA. Esta respuesta contendrá información para permitir o denegar la autorización de la petición.

Veamos a continuación un diagrama que describe la arquitectura de los componentes que se van a desarrollar para el sistema de políticas de privacidad de CERTILOC y su interacción.

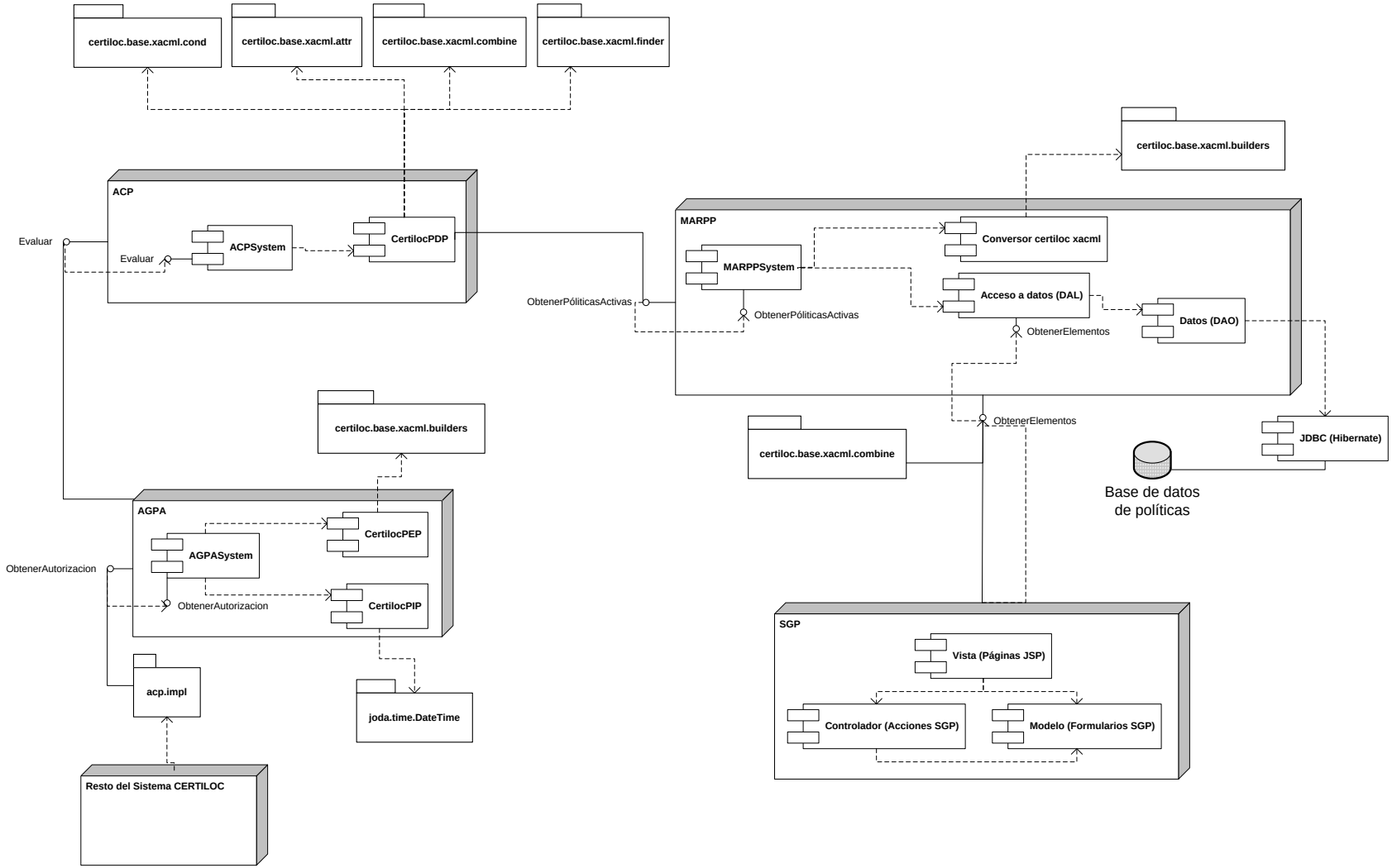


Figura 17. *Diseño detallado del sistema de políticas de privacidad de* CERTILOC

En la figura anterior vemos, abajo a la izquierda, cómo el resto del sistema CERTILOC [en un principio sólo el sistema AAA (Memoria PFC - Gallo Martínez 2008)] accede al **ACP** (Agente Custodio de la Privacidad) para hacer una petición de autorización. Recordamos al lector que estas peticiones de autorización provienen del resto del núcleo de CERTILOC cuando un usuario solicita un servicio que hace uso del sistema de políticas de privacidad.

A partir de ahí, el resto de módulos del sistema de políticas de privacidad se encargan de obtener la respuesta y devolverla al módulo de CERTILOC que la solicitó en primera instancia.

Tal y como marca el estándar XACML, las peticiones y las políticas deben ser convertidas al contexto de XACML antes de evaluar el resultado.

Para la conversión de la petición al contexto de XACML se ha implementado el sistema **AGPA** (Agente Gestor de Peticiones de Autorización). Una vez que se ha convertido a XACML, la petición se pasa al sistema **ACP** para su posterior evaluación contra las políticas de privacidad activas en el sistema.

Para la conversión de políticas de privacidad, persistentes en el sistema, al formato XACML, el sistema MARPP se encarga de su conversión antes de devolverlas al ACP.

Tanto para peticiones como para políticas, se dispone de un paquete de constructores (BUILDERS) que ayudan a la conversión, desde el formato o contexto nativo de CERTILOC**,** al formato XACML de las mismas.

A continuación veremos un diagrama que ayuda a comprender lo que se ha explicado en párrafos anteriores:

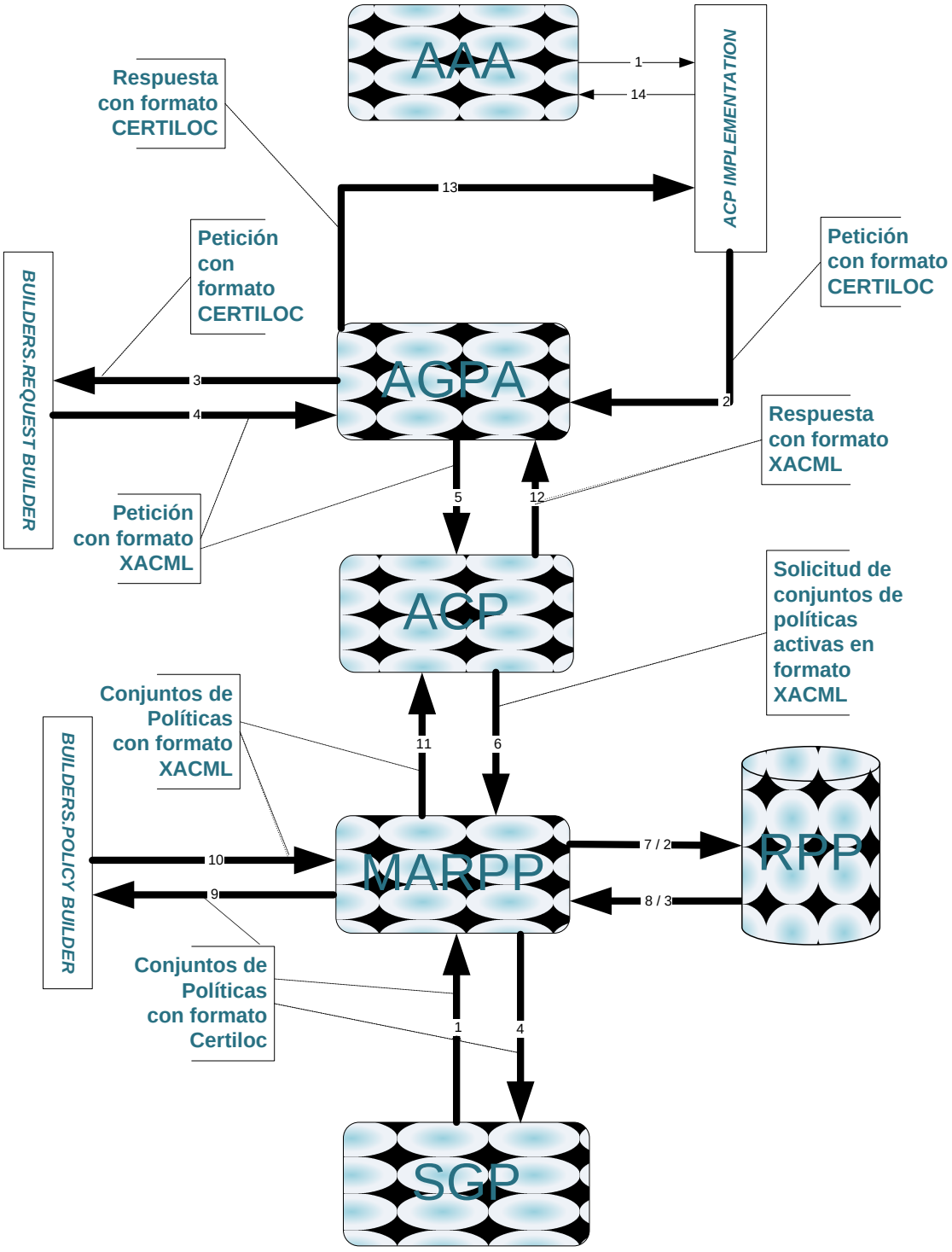


Figura 18. *Modelo de flujo de datos entre sistemas del sistema de políticas de privacidad*

La Figura 18 nos muestra la manera en que el sistema de políticas de privacidad va utilizando los distintos contextos o formatos de datos a lo largo de los diferentes módulos que lo conforman. Recordamos que existen dos formatos para la información (XACML y el formato nativo de CERTILOC). Si miramos la figura detenidamente, podremos observar qué formatos existen tienen cada uno de los mensajes intercambiados entre los distintos módulos.

A continuación vemos un detalle de la figura anterior que indica, de manera visual, los distintos contextos del sistema de políticas de privacidad. Estos contextos son el contexto de **XACML** y el contexto de CERTILOC. Además, se presenta otra figura donde se ven los nodos y componentes de la aplicación divididos por los contextos descritos.

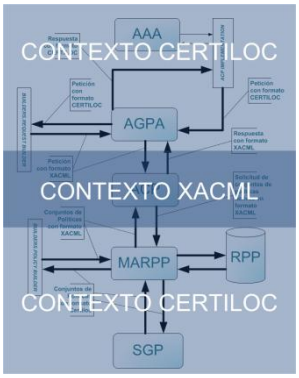


Figura 19. *Flujo de datos por contexto*

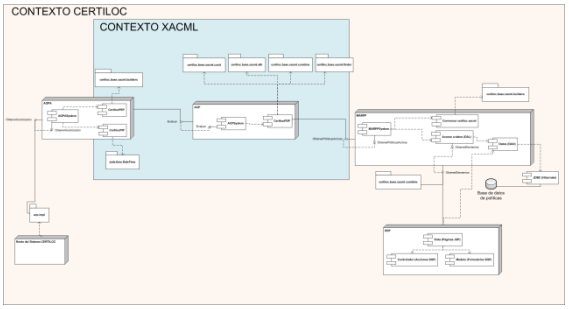


Figura 20. *Componentes por contexto*

4.3 MODELO DE DATOS DEL SISTEMA DE POLÍTICAS

Presentamos a continuación el modelo de datos del sistema de políticas de privacidad.

Este modelo está basado en el modelo de políticas de privacidad que define el estándar XACML, visto en el apartado 3.5.1.3.1 del presente documento. El modelo descrito en el estándar no cumplía todas las necesidades de CERTILOC con lo que se ha completado añadiendo nuevos campos a las clases que lo componen. En particular, se ha añadido un campo a las distintas clases que permite definir si están activas o no. De esta manera, cuando un elemento de un conjunto de políticas esté en estado inactivo, no influirá en la respuesta devuelta ante una petición de autorización.

Veamos un diagrama del modelo de datos (se divide la Figura 21 en dos partes de izquierda a derecha, para facilitar su lectura):

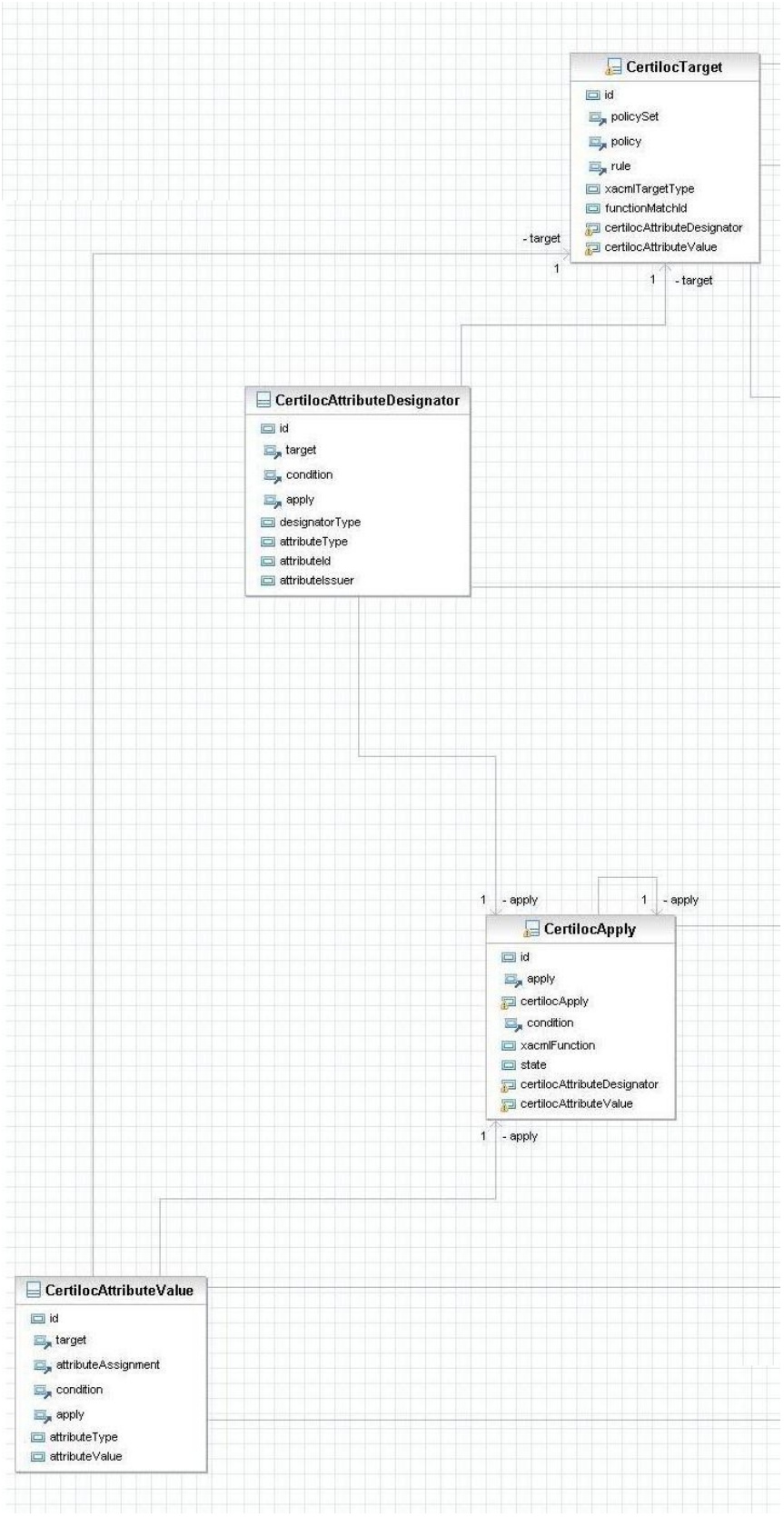

CertilocTarget
id
policySet
policy
rule
xacmlTargetType
functionMatchId
certilocAttributeDesignator
certilocAttributeValue
- target
1
1
- target
CertilocAttributeDesignator
id
target
condition
apply
designatorType
attributeType
attributeId
attributeIssuer
1
- apply
1
- apply
CertilocApply
id
apply
certilocApply
condition
xacmlFunction
state
certilocAttributeDesignator
certilocAttributeValue
1
- apply
CertilocAttributeValue
id
target
attributeAssignment
condition
apply
attributeType
attributeValue

Figura 21. *Modelo de datos del sistema de políticas de privacidad*

En el apartado 5.5.11.2 podemos ver una descripción completa de cada una de las clases que conforman el modelo y de sus atributos. En este apartado nos delimitaremos a explicar los conceptos de lo que cada clase representa.

4.3.1 LA CLASE CERTILOC POLICY SET

Esta clase representa un conjunto de políticas de privacidad. Cada conjunto de políticas estará compuesto por ciertos atributos, un conjunto de objetivos, un conjunto de políticas y un conjunto de obligaciones. Los objetivos indicarán contra qué peticiones se debe utilizar este conjunto de políticas y las obligaciones definirán las condiciones de uso de la información o recurso para el que se solicitan las distintas peticiones.

4.3.2 LA CLASE CERTILOC POLICY

Esta clase representa una política de privacidad concreta. Cada política estará compuesta por ciertos atributos, un conjunto de objetivos, un conjunto de reglas y un conjunto de obligaciones. Los objetivos indicarán contra qué peticiones se debe utilizar ésta política y las obligaciones definirán las condiciones de uso de la información o recurso para el que se solicitan las peticiones.

4.3.3 LA CLASE CERTILOC RULE

Esta clase representa una regla de privacidad concreta. Cada regla estará compuesta por ciertos atributos, entre ellos si la regla evalúa a permitir o denegar, un conjunto de objetivos y una (o ninguna) condición. Los objetivos definirán contra qué peticiones se debe aplicar esta regla y la condición (si existe) indicará bajo qué circunstancias concretas se aplica la regla en cuestión.

4.3.4 LA CLASE CERTILOC CONDITION

Esta clase representa una condición. Cada condición estará compuesta por ciertos atributos, entre los que encontramos la función que utiliza la condición, un conjunto de aplicativos (término definido en el apartado 1.3 del presente documento), un conjunto de especificadores de atributo y un conjunto de valores de atributo. Los aplicativos indicarán funciones aplicables a la propia condición, los especificadores de atributos indicarán parámetros de las peticiones a los cuales se les aplica la función deseada y los valores de atributos representan los valores concretos contra los que se aplica la función.

Cabe remarcar que estos aplicativos, tal y como marca el estándar XACML, no siempre serán necesarios por lo que una condición puede no necesitar contener aplicativos. Por otro lado, puede haber condiciones que no necesiten especificadores de atributos ni valores ya que realizan funciones que combinan aplicativos, por ejemplo al aplicar funciones OR y AND. En otros casos, un aplicativo puede representar uno de los parámetros de la función que se aplica en la condición y se puede acompañar de uno o varios valores de atributo.

Las funciones que implementan las condiciones siempre evaluarán a cierto o falso indicando si la condición se cumple o no.

Las condiciones y los aplicativos, permiten introducir parámetros del **entorno** bajo el que se evalúa la petición de autorización. El **entorno** define las condiciones puntuales de la petición como la hora y la fecha de realización de la petición.

4.3.5 LA CLASE CERTILOC APPLY

Esta clase representa un aplicativo. Cada aplicativo estará compuesto por ciertos atributos, entre los que encontramos la función que utiliza, un conjunto de aplicativos, un conjunto de especificadores de atributo y un conjunto de valores de atributo. Los aplicativos indicarán funciones aplicables al propio aplicativo, los especificadores de atributos indicarán parámetros, de las peticiones, a los cuales se les aplica la función especificada y los valores de atributos representan los valores concretos contra los que se aplica la función.

Al igual que las condiciones, los aplicativos pueden prescindir de otros aplicativos, conjuntos de atributos y valores de atributos según la función que utilicen.

Sin embargo, al contrario que las condiciones, un aplicativo puede utilizar funciones que devuelvan valores en vez de cierto o falso. Esto ocurre, por ejemplo, en aplicativos que utilizan funciones *one-and-only*. Estas funciones indican que el valor sólo puede ser uno. XACML permite, en ocasiones, introducir parámetros como conjuntos de atributos (o “bolsas de atributos” tal y como indica XACML – “Attribute Bag”). La función *one-and-only* nos indica que, en caso que el argumento debe ser un único valor y no varios.

4.3.6 LA CLASE CERTILOC TARGET

Esta clase representa un objetivo. Un objetivo define contra qué peticiones de autorización se aplican los conjuntos de políticas, las políticas o las reglas.

Un objetivo puede ser de tres tipos: **sujeto, acción,** y **recurso**. El tipo de objetivo indicará contra qué parte de la petición se comprueba la coincidencia. Además, definen una función para evaluar la coincidencia (igual, menor, mayor, etc.).

Por último, contendrán un especificador de atributo y un valor de atributo. El especificador de atributo permite definir el atributo de la petición contra la que se evalúa la coincidencia y el valor representa el valor contra el que se compara.

4.3.7 LA CLASE CERTILOC ATTRIBUTE DESIGNATOR

Esta clase representa un especificador de atributo. Los especificadores de atributos definen atributos presentados en las peticiones de autorización.

Entre los atributos de un especificador de atributo, encontramos su identificador. El identificador del atributo describe el atributo. Por ejemplo, dentro del elemento sujeto de la petición, podemos encontrar el rol, el nombre, la edad, etc. El identificador del especificador de atributo define cuál de estos atributos especifica (rol, nombre o edad – subject-role, subject-id y subject-age respectivamente).

Un especificador de atributo puede ser de cuatro tipos: **sujeto, acción, recurso** y **entorno**. El tipo indicará la parte de la petición donde se encuentra el especificador de atributo presentado.

Por otro lado, son de algún tipo de datos conocido (cadena, entero, decimal, etc.).

Por último, pueden contener un emisor. El emisor de un especificador atributo es la entidad que define dicho especificador de atributo (suele ser un especificador de atributo que no exista previamente en el estándar de XACML y que forma parte del universo de la aplicación para la que se crea el sistema de políticas de privacidad). En caso que el especificador forme parte del estándar XACML no hace falta designar un emisor. Por ejemplo, para los atributos definidos específicamente para CERTILOC, se utilizará el siguiente emisor de atributo: *urn:CERTILOC:1.0:issuer:admin@CERTILOC.com*. Sin embargo, para los especificadores de atributo definidos en el estándar de XACML, no se especifica ningún emisor de atributo.

4.3.8 LA CLASE CERTILOC ATTRIBUTE VALUE

Esta clase representa un valor de atributo. Los valores de atributos representan, como su propio nombre indica, los valores que toman ciertos atributos. Se representan con un tipo de datos conocido (cadena, entero, decimal, etc.) y con un valor.

4.3.9 LA CLASE CERTILOC OBLIGATION

Esta clase representa una obligación. Una obligación indicará las condiciones de uso de la autorización o negación de recurso ante una petición de autorización. Las obligaciones cuándo se deben presentar (al aceptar o denegar una petición) y un identificador de la obligación. El identificador será una descripción de la obligación, por ejemplo "*términos-legales-uso-datos"*. Contendrán un conjunto de asignaciones de atributo que representarán los detalles de la obligación.

4.3.10 LA CLASE CERTILOC ATTRIBUTE ASSIGNMENT

Esta clase representa una asignación de atributo. Una asignación de atributo definirá el valor concreto de una parte de una obligación.

El identificador será una descripción de la asignación de atributo. Por ejemplo, para la obligación *"términos-legales-uso-datos"* podemos tener dos partes: *"descripción"* que contendrá un párrafo descriptivo de la obligación y *"fecha-expiración"* que definirá cuándo expiran los términos legales de uso de los datos a los que se accede. Contendrán un valor de atributo para definir el valor concreto de la asignación de atributo.

4.3.11 LA CLASE XACML REQUEST BUILDER

Esta clase no representa una entidad del modelo de datos de políticas de CERTILOC, propiamente dicha, pero es importante observarla para poder comprender en conjunto el sistema de políticas de privacidad. Cabe destacar que esta clase es un útil para poder construir peticiones XACML bien formadas a partir de una petición del sistema CERTILOC. No se implementa como una clase del modelo de datos ya que las peticiones no se guardarán en la base de datos (sólo se guardan peticiones de autorización en su formato XACML para poder enriquecer los registros de actividad del sistema de políticas de privacidad).

Esta clase nos permite construir una determinada petición de autorización contra la que se aplicarán las políticas de privacidad.

Tal y como define el estándar XACML, una petición puede contener un conjunto de valores de **sujeto**, un conjunto de valores de **recurso**, un conjunto de valores de **acción** y un conjunto de valores de **entorno**.

- Como ya se ha comentado anteriormente en el presente documento, el **sujeto** representa la entidad (persona, entidad jurídica, etc.) que realiza la petición de autorización.

- El **recurso** representa el recurso al que se desea acceder cuando se realiza la petición de autorización (un dispositivo GPS, un dispositivo RFID, un conjunto de información, etc.).
- La **acción** representa la acción que se quiere llevar a cabo sobre el recurso (descargar, localizar, etc.)
- El entorno representa condiciones concretas sobre el entorno bajo el cual se realiza la petición (datos de localización que puedan influir, hora de la petición, hora de la petición original en caso de ser una petición diferida, etc.).

4.4 EL MODELO DE LA BASE DE DATOS

La base de datos que soportará el sistema de políticas de privacidad de CERTILOC estará hecha a imagen y semejanza del modelo de datos de políticas de privacidad.

Cabe remarcar que éste es uno de los puntos más ambiciosos del proyecto ya que se pretende mantener en una base de datos, lo que originalmente se guardaba en archivos XML (las políticas XACML).

A continuación se presenta un diagrama entidad-relación del diseño de la base de datos.

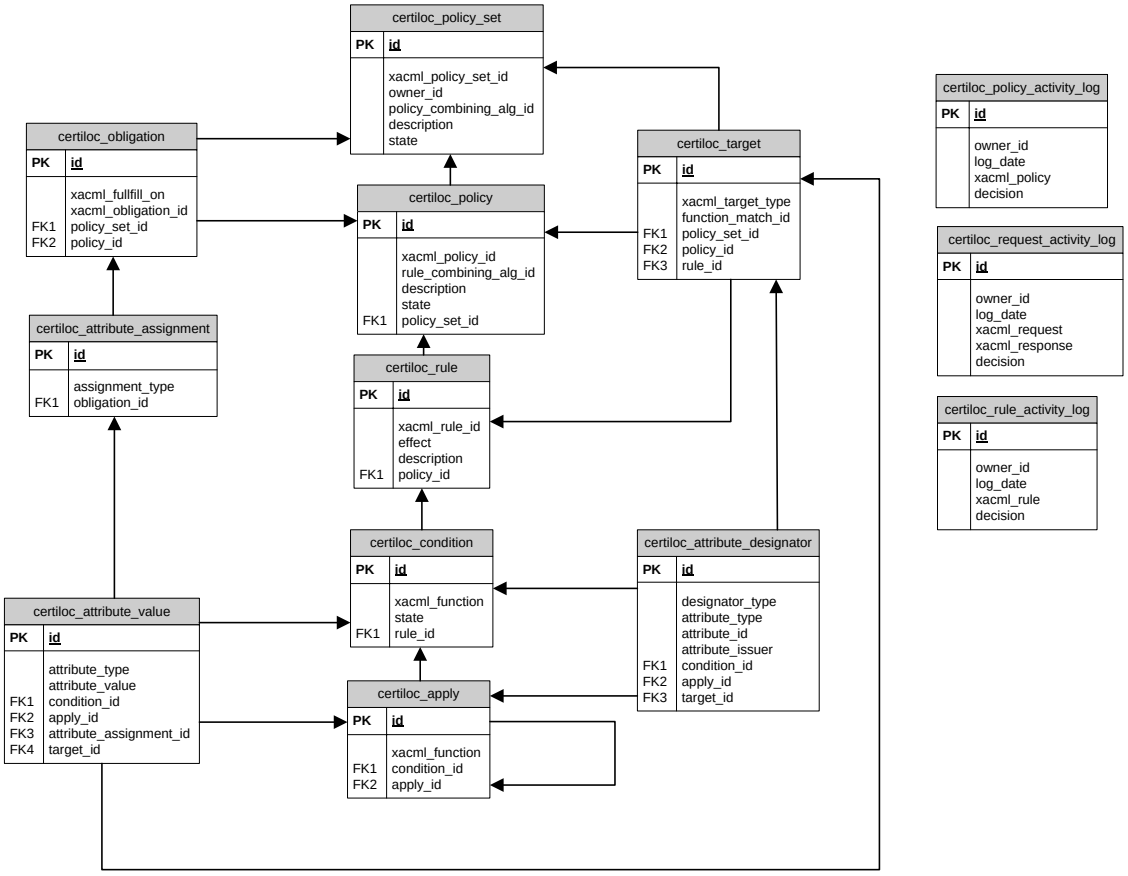


Figura 22. *Diagrama E/R Modelo de base de datos*

La figura anterior muestra un diagrama completo de la base de datos. Podemos comprobar que es igual que el modelo de datos ya que, cada entidad de la base de datos representa una clase del modelo de datos. Utilizando una capa de acceso a datos intermedia (DAO), podremos acceder al sistema de persistencia de datos mediante instancias de clases. Las instancias de las clases serán tuplas de cada una de las entidades de la base de datos.

4.5 INTERFAZ DE USUARIO WEB

Se describe en este apartado el diseño general de la interfaz de usuario web diseñada para la gestión de las políticas de privacidad de los usuarios.

Para el marco general de la aplicación se integrarán los gráficos desarrollados para el núcleo de CERTILOC en anteriores proyectos orientados al demostrador de CERTILOC.

Dichos gráficos, están organizados mediante el uso de hojas de estilo en cascadas y código HTML que se reutilizará directamente tal cual nos lo han traspasado desde los otros proyectos. La mayor parte del trabajo de diseño del marco inicial de la interfaz web y del

diseño de gráficos de la aplicación fueron contribuciones realizadas por el primer PFC que se desarrolló orientado al desarrollo de CERTILOC (Memoria PFC - Calvo Martínez 2007).

La siguiente figura nos muestra un detalle aproximado del marco general de la aplicación indicando sus distintos elementos generales.

Figura 23. *Marco general interfaz de usuario Web CERTILOC*

Para el sistema de gestión de políticas y de seguimiento de actividad de las mismas (módulo SGP), tendremos que utilizar el marco general presentado en la Figura 23.

La gestión de políticas de privacidad puede ser una tarea tediosa para los usuarios ya que el modelo de las mismas es complejo. Recordemos que hemos decidido adaptar el modelo de datos de XACML que es bastante completo pero que contempla varias exigencias.

Para el desarrollo del SGP, debemos intentar facilitar, en la medida de lo posible, la tarea de gestión de políticas (creación, modificación, borrado, activación y desactivación) a los usuarios finales.

Dado que los datos persistentes del modelo de datos de políticas de privacidad se va a albergar en una base de datos, podremos construir un menú de navegación que permita a los usuarios navegar por sus políticas. El menú será un menú en forma de árbol donde la raíz del mismo serán los conjuntos de políticas y se irá expandiendo hasta sus hojas (especificadores de atributos, valores de atributos, etc.).

Además, dado que el estándar XACML contempla un lenguaje concreto para las políticas de seguridad, debemos proporcionar un mecanismo a los usuarios para comprobar la corrección y exactitud de las distintas políticas y elementos que va creando.

Para crear el árbol de conjuntos de políticas de cada usuario, lo cargaremos dinámicamente sobre el menú en forma de árbol, cuando el usuario entre al SGP. Es decir, cargaremos en forma de árbol todos los conjuntos de políticas que pertenezcan al usuario para que pueda navegar por ellos y por sus elementos.

Cada vez que el usuario pinche directamente sobre un elemento del árbol, se le mostrarán los detalles concretos de ese elemento, permitiéndole modificarlo, borrarlo o añadirle nuevos elementos o hijos.

La siguiente figura muestra un ejemplo real de la forma que podrán tomar los árboles de políticas de privacidad en el SGP de CERTILOC:

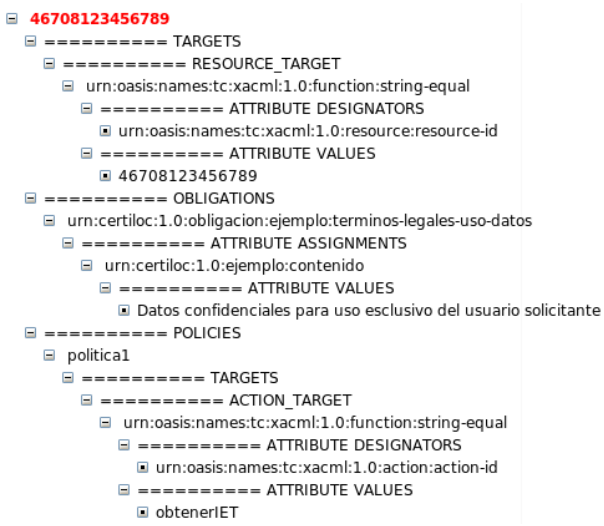


Figura 24. *Ejemplo de menú de conjuntos de políticas de usuario*

Como vemos, la raíz (el conjunto de políticas) de está remarcada en rojo. De ella cuelgan el resto de elementos. Al pinchar sobre un elemento, el usuario debe ser dirigido a los detalles de ese elemento.

Para ayudar a crear este tipo de menús se utilizará la herramienta o API **struts-menu** (Sourceforge 2009) y más en concreto el menú “List Menu”. Esta herramienta es una librería orientada para su uso en aplicaciones diseñadas con STRUTS que permite la creación de distintos tipos de menús que serán cargados dinámica o estáticamente en el sistema.

En el caso del SGP, necesitaremos una clase intermedia (tal y como vemos en el apartado 4.7.10.1.1 del presente documento) que nos ayudará a hacer la carga dinámica de los menús que vamos a mostrar al usuario en la interfaz web final.

Por otro lado, para que el usuario pueda comprobar en todo momento la corrección de los datos e hijos de cierto elemento contra la definición de la especificación XACML, crearemos una ayuda intermedia para el usuario que permita convertir determinado elemento del menú de sus políticas de privacidad al formato XACML.

La siguiente figura nos muestra un ejemplo de cómo se puede integrar la visión XACML de un determinado elemento de un conjunto de políticas de privacidad.

Figura 25. *Inspección XACML de un elemento concreto en el SGP*

Dado que el módulo MARPP contendrá funciones (apartado 4.7.9.2.2 del presente documento) para convertir elementos de políticas y conjuntos de políticas al formato de XACML, la función se delegará en el mismo. El SGP sólo mostrará los datos devueltos por la conversión de cierto elemento desde el módulo MARPP.

La Figura 26 muestra un resumen de la navegación que se llevará a cabo por el usuario dentro del SGP (sistema de gestión de políticas de privacidad). El diagrama muestra las distintas páginas JSP que conformarán la aplicación y las distintas acciones que implementará el módulo y que llevarán al usuario de una página a otra.

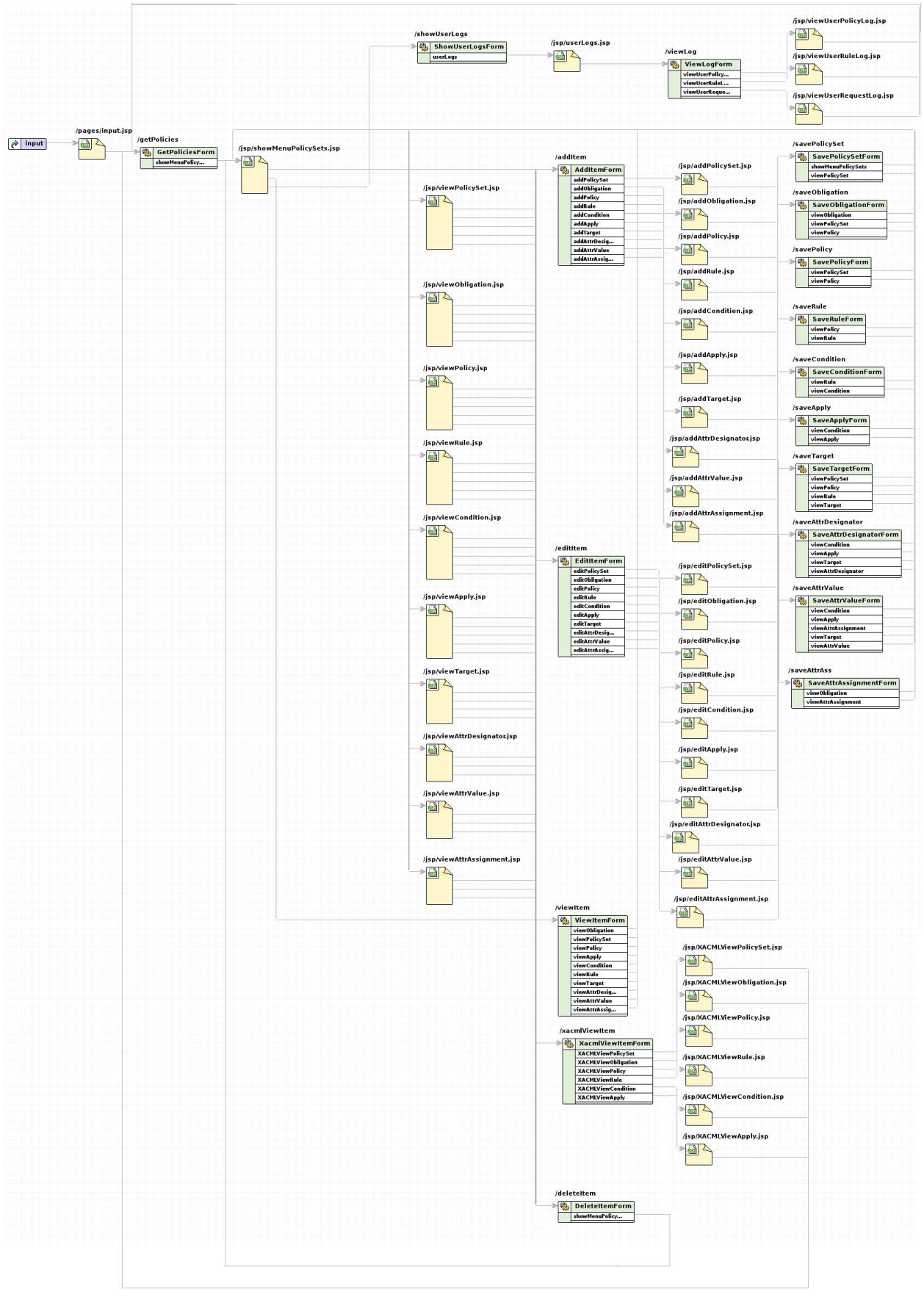


Figura 26. *Diagrama de navegación del módulo SGP*

4.6 TECNOLOGÍAS A UTILIZAR

Se presentan a continuación las distintas tecnologías escogidas por el autor del presente PFC para el desarrollo del proyecto y las razones para su elección.

4.6.1 UBUNTU DESKTOP 6.06 LTS

Para el montaje del entorno de desarrollo de la aplicación, se debe proveer un sistema operativo básico.

En el caso del desarrollo del sistema de políticas de privacidad contamos con 2 alternativas generales: sistema operativo Linux o sistema operativo Microsoft.

Dado que el entorno de producción del demostrador de CERTILOC estará basado en LINUX, nos decantamos por una distribución Linux.

Ahora sólo resta escoger la distribución adecuada para el desarrollo. En este caso, se ha escogido Ubuntu Desktop Edition 6.06 LTS (Canonical Ltd. Ubuntu 2009).

4.6.1.1 Descripción

Ubuntu es una distribución GNU/Linux que ofrece un sistema operativo predominantemente enfocado a ordenadores de escritorio, aunque también proporciona soporte para servidores (Portal Web doc.ubuntu-es.org 2009).

Basada en Debian GNU/Linux, Ubuntu concentra su objetivo en la facilidad de uso, la libertad de uso, los lanzamientos regulares (cada 6 meses) y la facilidad en la instalación. Ubuntu es patrocinado por Canonical Ltd., una empresa privada fundada y financiada por el empresario sudafricano Mark Shuttleworth.

El nombre de la distribución proviene del concepto zulú y xhosa de ubuntu, que significa "humanidad hacia otros" o "yo soy porque nosotros somos". Ubuntu es un movimiento sudafricano encabezado por el obispo Desmond Tutu, quien ganó el Premio Nobel de la Paz en 1984 por sus luchas en contra del Apartheid en Sudáfrica.

4.6.1.2 Razones para su elección

Tal y como se ha descrito en la introducción de este apartado, dado que el sistema operativo del entorno de producción del demostrador de CERTILOC será Ubuntu Server (requisito CERTILOC-PP-RNFD-001) utilizaremos la distribución Ubuntu Desktop Edition 6.06 LTS (Canonical Ltd. Ubuntu 2009).

Se ha escogido esta versión específica ya que aporta una garantía de servicio por parte del fabricante de un mínimo de 3 años a partir del 2007, tal y como nos indican las iniciales LTS ("Long Term Service").

4.6.2 EL API "SUN'S XACML IMPLEMENTATION"

Para la conversión de contexto entre las peticiones y las políticas en el formato nativo de CERTILOC al formato de XACML y viceversa, tenemos que servirnos de un API intermedio que provea un modelo de datos orientado específicamente a XACML.

Desgraciadamente, no existen más alternativas en el mercado del software, para conseguir este objetivo, más que la biblioteca que se ha escogido para este propósito en el presente proyecto. Esta tecnología es la implementación de XACML de Sun Microsystems (XACML - Sun Microsystems , Inc. 2009).

4.6.2.1 Descripción

Sun´s XACML Implementation (XACML - Sun Microsystems , Inc. 2009) es un proyecto de desarrollo de código abierto que persigue implementar el estándar XACML, definido por Sun. Está escrito en el lenguaje de programación JavaTM (Java - Sun Microsystems , Inc. 2009).

Sun's XACML Implementation proporciona un soporte completo para todas las funciones obligatorias de XACML, así como una serie de funciones opcionales. Concretamente, ofrece un soporte completo para el modelo de datos de XACML, para analizar tanto la solicitud de petición como la respuesta en formato XACML, para evaluar la aplicabilidad de las políticas y para evaluar las peticiones de decisión contra conjuntos de políticas XACML.

Además, este API proporciona todas las definiciones de tipos de atributos, funciones y algoritmos de combinación incluidos en el estándar XACML. Por otro lado, proporciona una interfaz para extender toda su funcionalidad y para crear nuevos mecanismos para encontrar elementos tales como políticas o atributos.

Es un proyecto que fue desarrollado en los laboratorios de Sun Microsystems (Sun Microsystems , Inc. 2009) y forma parte de un proyecto de investigación de la seguridad en internet.

4.6.2.2 Razones para su elección

Se ha decidido escoger esta API de desarrollo ya que no se ha encontrado ninguna otra para orientada al desarrollo utilizando el estándar XACML.

La única posible alternativa a esta tecnología sería desarrollar nuestro propio API para la utilización de XACML. Esta posibilidad no es viable por los límites de tiempo y de recursos que caracterizan el presente proyecto.

4.6.2.3 Adaptaciones necesarias

En el caso del API utilizado para el manejo de datos en formato XACML, se incluye un apartado sobre adaptaciones necesarias de la herramienta para adaptarla a los requisitos impuestos en el análisis de CERTILOC.

Para cada una de las adaptaciones del API, se presentará una tabla con los siguientes datos:

Identificador único de la adaptación	
Módulo XACML implicado	Indica el módulo o paquete del API que se amplia
Paquete o Clase creada	Indica el nombre del paquete o clase creada para suplir la necesidad
Necesidad de CERTILOC	Indica las necesidades concretas de CERTILOC que motivan la ampliación
Ampliaciones al API	Muestra un resumen de las ampliaciones necesarias para cumplir las necesidades de CERTILOC

Tabla 116. *Tabla para la especificación de ampliaciones al API Sun's XACML Implementation*

API-XACML-AD-001	
Módulo XACML implicado	Ninguno
Paquete o Clase creada	certiloc.base.xacml.CertilocXACMLConfiguration
Necesidad de CERTILOC	Necesidad de poder especificar atributos propios (localización, rol de usuario, etc.) no contemplados en el contexto de XACML

Ampliaciones al API	Es necesario introducir atributos de entorno concretos para el Rol de usuario CERTILOC, la información de Localización IET y la Fecha y Hora IET y la Fecha y Hora de ejecución de la petición. Se crean una serie de constantes comunes para poder introducirlas de la misma manera en peticiones y políticas

Tabla 117. *Ampliación al API Sun's XACML Implementation API-XACML-AD-001*

API-XACML-AD-002	
Módulo XACML implicado	com.sun.xacml.attr
Paquete o Clase creada	certiloc.base.xacml.attr
Necesidad de CERTILOC	Necesidad de poder especificar atributos de Coordenada (X, Y y Sistema de Referencia)
Ampliaciones al API	Se incluye un nuevo tipo de atributo XACMl, la Coordenada

Tabla 118. *Ampliación al API Sun's XACML Implementation API-XACML-AD-002*

API-XACML-AD-003	
Módulo XACML implicado	com.sun.xacml.combine
Paquete o Clase creada	certiloc.base.xacml.combine
Necesidad de CERTILOC	Necesidad de poder hacer un seguimiento de la actividad de las políticas de privacidad. Para ello necesitamos ejecutar nuestros propios algoritmos de combinación de políticas y reglas.
Ampliaciones al API	Se incluyen dos nuevos algoritmos de combinación de reglas y políticas que generan registros de actividad de políticas en el repositorio de políticas de privacidad o RPP

Tabla 119. *Ampliación al API Sun's XACML Implementation API-XACML-AD-003*

API-XACML-AD-004	
Módulo XACML implicado	com.sun.xacml.cond
Paquete o Clase creada	certiloc.base.xacml.cond
Necesidad de CERTILOC	Necesidad de poder crear nuestras propias funciones de combinación de información.
Ampliaciones al API	Se incluyen dos nuevas funciones. La primera para calcular si una Coordenada determinada se encuentra en el área designada por un rectángulo.

Tabla 120. *Ampliación al API Sun's XACML Implementation API-XACML-AD-004*

API-XACML-AD-005	
Módulo XACML implicado	com.sun.xacml.finder
Paquete o Clase creada	certiloc.base.xacml.finder
Necesidad de CERTILOC	Necesidad de poder cargar un módulo de políticas XACML desde una base de datos MySQL
Ampliaciones al API	Dado que el API originalmente sólo considera un módulo para la carga de políticas desde ficheros de texto planos en formato XML, se ha tenido que crear un módulo de carga de políticas de privacidad entero para poder hacer la carga de las políticas de privacidad mediante el sistema MARPP

Tabla 121. *Ampliación al API Sun's XACML Implementation API-XACML-AD-005*

4.6.3 MYSQL SERVER

Para la persistencia de datos en el sistema necesitaremos contar con un motor de bases de datos. Esta será la manera de guardar políticas de privacidad (en contexto CERTILOC) frente a la alternativa de guardar los datos en ficheros de texto.

Para este objetivo encontramos distintas alternativas como Oracle, Microsoft SQL Server o MySQL. En nuestro caso, dado que el entorno de producción del demostrador de CERTILOC se ejecuta bajo Linux (requisitos CERTILOC-**PP-RNFD-001,** CERTILOC-**PP-RRec-001**) y que el motor de bases de datos será MySQL (requisito CERTILOC-**PP-RNFD-004**), nos decantamos claramente por MySQL.

4.6.3.1 Descripción

MySQL es un sistema de bases de datos que nos permite montar un sistema completo de persistencia de datos.

Será el sistema elegido para la persistencia de los datos de las políticas de privacidad del demostrador de CERTILOC.

Actualmente, es el sistema de base de datos relacionales cuyo uso es el más extendido del mundo (MySQL AB 2009). Cuenta con más de seis millones de instalaciones alrededor de todo el mundo y su utilización sigue creciendo.

MySQL se ofrece como una aplicación de código abierto y gratuita para el desarrollo, sin embargo, para su uso empresarial exige la compra de una licencia.

4.6.3.2 Razones para su elección

Se ha decidido escoger MySQL como sistema de bases de datos ya que, tal y como se ha descrito anteriormente, su licencia es gratuita para entornos de desarrollo y su uso es, actualmente, el más extendido del mundo.

Otra de las principales para su elección ha sido por que va a ser el sistema de base de datos que utilizará el demostrador de CERTILOC en el entorno de producción. En caso de escoger otro sistema de bases de datos, nos arriesgamos a afrontar una implantación, larga y tediosa, del módulo del sistema de políticas de privacidad con el entorno de producción del demostrador de CERTILOC.

4.6.4 APACHE TOMCAT WEB SERVER

Para dar soporte al módulo de gestión de políticas de privacidad (SGP) debemos utilizar un servidor web que soporte las tecnologías a utilizar (requisito CERTILOC-PP-RNFD-005). Recordamos que el sistema debe accesible por los usuarios mediante tecnologías web (requisito CERTILOC-PP-RNFI-001).

En el mercado encontramos una amplia gama de servidores web. Los más utilizados, son Apache Tomcat Server (The Apache Software Foundation 2009) e Internet Information Server (Microsoft. 2009) donde, Apache Server gana actualmente a Internet Information Server en un 20% en cuanto a número de instalaciones (Netcraft 2009).

Una vez más, dado que el entorno de producción del demostrador de CERTILOC se ejecuta bajo Linux (requisitos CERTILOC-**PP-RNFD-001,** CERTILOC-**PP-RRec-001**) y que el servidor web del entorno de producción será Apache Tomcat (requisito CERTILOC-**PP-RNFD-003**), nos decantamos por él. Además, Apache Tomcat está mucho mejor integrado que su rival para el desarrollo utilizando tecnologías JSP y STRUTS tal y como manda la especificación de la aplicación (requisito CERTILOC-PP-RNFD-005).

4.6.4.1 Descripción

Apache Tomcat es un servidor web compatible con varias tecnologías para el desarrollo web con tecnologías del lado del servidor (PHP, JSP, ASP, ASP .NET, etc.).

Es una implementación de las tecnologías Java Servlet y JavaServer Pages (JSP) y está, actualmente, desarrollado por el proyecto Apache Tomcat (The Apache Software Foundation 2009).

Dado que Apache Tomcat se puede integrar con la tecnología JSP y con Struts, será un buen candidato para el desarrollo del actual proyecto. Seguidamente podemos observar las razones para su elección.

4.6.4.2 Razones para su elección

Se ha escogido Apache Tomcat como servidor web por varias razones. La principal es porque va a ser el servidor web que utilizará el demostrador de CERTILOC en su entorno de producción.

Por otro lado, es un servidor web que está muy integrado con la herramienta Eclipse que, como veremos en apartados posteriores, será la herramienta de entorno de desarrollo escogida para el proyecto actual.

4.6.5 ECLIPSE IDE

Para la implementación de todos los módulos de desarrollo, haremos uso de un entorno de desarrollo unificado (IDE).

En este caso, dado que la aplicación se va a desarrollar utilizando Linux como sistema operativo base, podemos escoger dos entornos de desarrollo distintos soportados directamente por Sun para el desarrollo de aplicaciones Java: Netbeans o Eclipse.

La decisión entre uno u otro no es sencilla ya que ambos tienen características buenas y otras malas.

Por un lado, Netbeans nos aporta un IDE de desarrollo bastante bien integrado, por defecto, con las tecnologías web y la persistencia de datos, pero consume más recursos que Eclipse.

Por otro lado, Eclipse está menos integrado con tecnologías concretas y su configuración es más complicada pero consume menos recursos.

Podríamos decir que en el mundo del desarrollo Java, son competencia directa y están bastante igualadas en sus características, siendo Eclipse más “ligero”, en cuanto al uso de recursos del sistema en tiempo de ejecución.

Dado que la máquina donde se debe desarrollar la aplicación no es muy potente (aptdo. 3.3.3 - MR-001) en cuanto a memoria se refiere, nos decantamos por eclipse para el entorno de desarrollo.

4.6.5.1 Descripción

Eclipse es principalmente una plataforma de programación, usada para crear entornos integrados de desarrollo (Fundación Eclipse 2009).

Eclipse fue desarrollado originalmente por IBM como el sucesor de su familia de herramientas para VisualAge. Eclipse es ahora desarrollado por la Fundación Eclipse (“Eclipse foundation), una organización independiente sin ánimo de lucro que fomenta una comunidad de código abierto. "Eclipse Foundation", está compuesta por las compañías: Borland, Computer Associates, Ericsson, Fujitsu, Hitachi, HP, IBM, Oracle, Rational Software, Red Hat, Serena, SAP, SuSE, Sybase, Versant, WebGain.

Esta herramienta, permite el desarrollo de proyectos en distintos lenguajes de programación, entre los que podemos encontrar java y jsp por lo que podrá valer para el desarrollo del sistema de políticas de privacidad.

4.6.5.2 Razones para su elección

Se ha escogido Eclipse para la plataforma de desarrollo ya que es una herramienta muy extendida en el entorno de la programación en java por lo que tiene una amplia comunidad de soporte.

Además, Eclipse es una herramienta que nos va a permitir integrar las distintas tecnologías a utilizar (MySQL, Apache Tomcat, Struts y Sun´s XACML Implementation, etc.) para el desarrollo.

Por último, dado el poco consumo de recursos de sistema en tiempo de ejecución de Eclipse, nos permitirá tener un entorno de desarrollo más rápido y fluido.

4.6.6 OMONDO ECLIPSE UML

Para poder extraer diagramas UML de bajo nivel durante la implementación de la aplicación, y para poder definir clases partiendo de este tipo de diagramas, necesitaremos un componente para Eclipse que nos facilite esta tarea.

En el mercado existen varios componentes o “plugins” para Eclipse que cumplen este objetivo como Omondo UML (Omondo - The Live UML Company 2009) o Smart Development Environment Community Edition for Eclipse (Visual Paradigm for UML 2009), siendo Omondo UML uno de los más extendidos.

En oposición de Omondo UML frente a Smart Development, debemos indicar que Omondo UML no se ofrece como un software libre en su versión profesional o “Studio”. No obstante, se permite utilizar una versión reducida, de carácter libre o gratuito que es completamente integrable con el entorno de desarrollo Eclipse.

Se elige Omondo UML como tecnología a utilizar ya que aparentemente ofrece un mejor soporte que Smart Development.

4.6.6.1 Descripción

Omondo EclipseUML (Omondo - The Live UML Company 2009) es un componente para el entorno de desarrollo Eclipse que nos va a permitir llevar a cabo un desarrollo en cascada donde podremos integrar fácilmente el diseño de la aplicación con el desarrollo de la misma.

Es un software desarrollado por la compañía Omondo y ofrece dos versiones (Free y Pro). La versión “Free Edition”, como su propio nombre indica, es completamente gratuita y ha sido desarrollada bajo los estándares del software libre. Esta versión nos ofrece todas las

funciones que vamos a necesitar para el desarrollo de los módulos del sistema de políticas de privacidad:

- Diseño de diagramas de paquetes y componentes
- Diseño de diagramas de clases
- Diseño de diagramas de secuencia de bajo nivel
- Diseño de diagramas de colaboración

A partir de estos diagramas, EclipseUML nos va a permitir generar parte del código de nuestra aplicación. Además otra función que posiblemente necesitemos es la función de ingeniería inversa. Es decir, extraer diagramas UML a partir de porciones de código Java. Esta función nos será útil para el desarrollo de diagramas de secuencia UML complejos, sobre todo en funciones donde existan bucles o saltos condicionales.

Veamos el esquema global de desarrollo que propone Omondo para el desarrollo de aplicaciones utilizando sus herramientas:

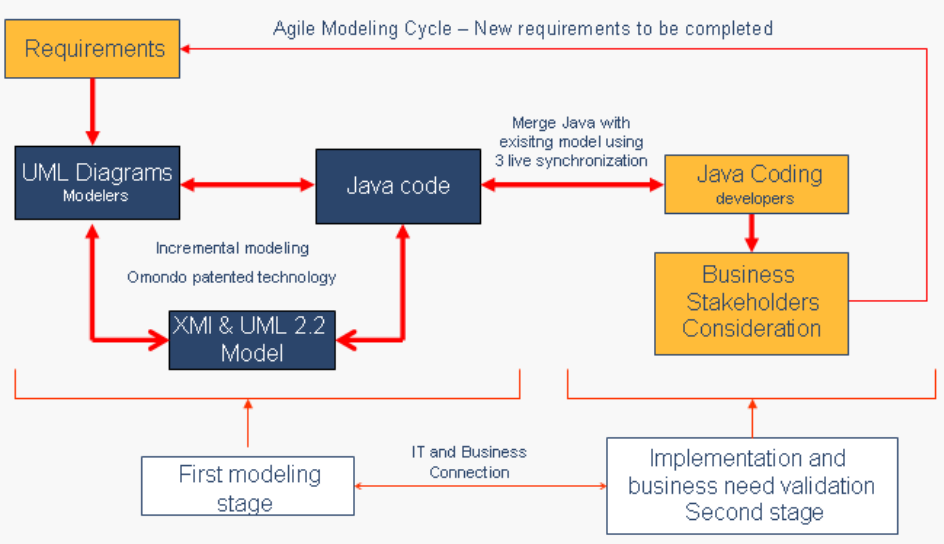


Figura 27. *Esquema de desarrollo propuesto por Omondo (Omondo - The Live UML Company 2009)*

4.6.6.2 Razones para su elección

Dado que el IDE de desarrollo será Eclipse, se ha decidido escoger este componente para el desarrollo de los distintos diagramas UML de bajo y alto nivel.

Omondo EclipseUML es una herramienta ampliamente extendida y utilizada en el mundo del desarrollo con Eclipse.

Va a aportar cierta capacidad para poder ampliar el modelo de datos del sistema de políticas de privacidad con facilidad. Los futuros programadores que puedan tener que acceder al código de nuestros módulos para modificarlo o entenderlo, lo podrán plantear desde el punto de vista de diseño y sin realizar grandes esfuerzos para comprender el código fuente y su función en cada módulo desarrollado.

Además, dado que el uso de Omondo UML está ampliamente extendido, podemos encontrar bastante soporte para el mismo.

4.6.7 HIBERNATE

Para el acceso a datos de la aplicación, utilizaremos una capa intermedia de acceso a datos (DAO) valiéndonos para ello de algún componente o plugin para Eclipse.

En el mercado encontramos muchas herramientas, en forma de API, para implementar capas de acceso a datos. En este caso, hemos comparado Hibernate (Hibernate - Red Hat, Inc. 2009) frente a JDBaccess (JDBaccess.com 2008).

En este caso la elección ha sido sencilla. Dado que el autor del presente PFC ya conocía de antemano el funcionamiento de Hibernate, la tecnología escogida ha sido ésta. Además, destaca su carácter de software de libre distribución y su amplio soporte, derivado de un amplio uso en el desarrollo de aplicaciones Java.

4.6.7.1 Descripción

Hibernate es una herramienta de alto rendimiento para la implementación de persistencia relacional de objetos (Hibernate - Red Hat, Inc. 2009).

Hibernate tiene su propio lenguaje para la especificación de consultas SQL: HQL (Hibernate Query Language). Lo que se pretende con esto es desacoplar las consultas que se realizan a la base de datos con la propia tecnología de la base de datos. No obstante, también nos va permitir realizar consultas en lenguaje nativo SQL o con criterios de la programación orientada a objetos.

El proyecto Hibernate ha sido desarrollado bajo los estándares del software libre y su uso, en aplicaciones bajo la plataforma J2EE, está ampliamente extendido. Una vez más, esto va a permitir que nuestra aplicación sea fácilmente extensible y comprensible para otros programadores.

A continuación presentamos un esquema de los distintos usos de esta herramienta con otras tecnologías:

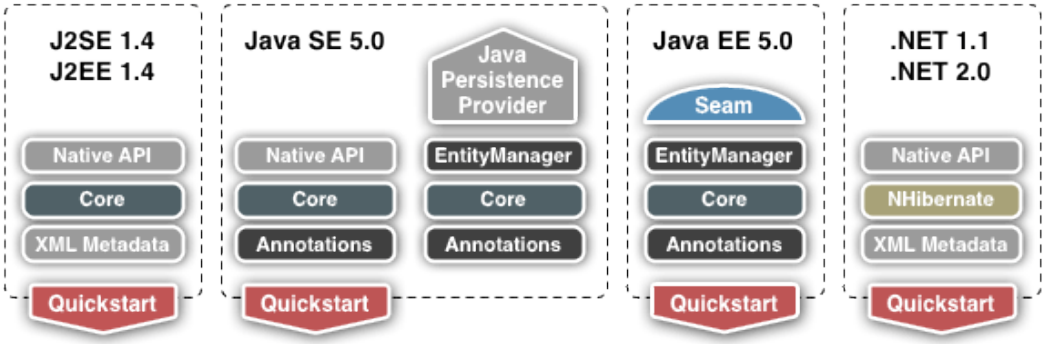


Figura 28. *Esquema del uso del proyecto Hibernate (Hibernate - Red Hat, Inc. 2009)*

4.6.7.2 Razones para su elección

Mediante el uso de Hibernate, nuestra aplicación podrá cambiar de tecnología de base de datos con facilidad. Además, no tendremos que tocar una sola línea de código para adaptarla a la nueva tecnología. Bastará con un cambio en un fichero de propiedades de Hibernate y listo.

Nos va a permitir desarrollar clases persistentes, con el paradigma de la programación orientada a objetos, haciendo uso de la herencia, el polimorfismo, la composición, las colecciones y la asociación. En general, Hibernate va a permitir acceder a los distintos datos de la aplicación de una manera cómoda (mediante instancias a clases concretas) y, sobre todo nos va a facilitar desarrollar un código bien estructurado, ordenado, fácil de entender para futuros desarrolladores.

4.6.8 LOG4J LOGGER

Para implementar el sistema de registros a nivel de sistema (requisito de software CERTILOC-PP-RFO-012) se ha decidido guardar los registros de cada subsistema en un archivo de texto concreto guardado en el servidor. Para conseguir este objetivo, nos ayudaremos de una clase intermedia que nos permitirá definir distintos formatos y destinos para cada evento registrado.

Existen varias herramientas, gratuitas en forma de API, que nos facilitan esta tarea. Entre ellas Log4J (Log4J - Apache Software Foundation 2007) o el paquete java.util.logging (Logging - Sun Microsystems, Inc. 2004) de Java 1.5.

En este caso nos decantamos por Log4J por su facilidad de uso frente al paquete java.util.logging.

4.6.8.1 Descripción

Log4j es una biblioteca de código abierto, desarrollada por Apache (Log4J - Apache Software Foundation 2007), que permite a los desarrolladores de software elegir la salida y el nivel de granularidad de los mensajes o registros generados por una aplicación en a tiempo de ejecución.

La configuración de salida y granularidad de los mensajes es realizada en tiempo de ejecución mediante uno o varios archivos de configuración externos.

Su un uso está ampliamente extendido y ofrece mucha información de soporte.

4.6.8.2 Razones para su elección

Se ha decidido seleccionar esta tecnología ya que su uso está muy extendido y existe gran variedad de información de soporte de la misma.

Por otro lado, Log4J nos ofrece una manera sencilla de configurar realizada mediante un archivo de configuración. Este fichero nos va a permitir tener centralizada toda la configuración de los registros del sistema, pudiendo modificar tanto el destino como el formato de los registros de una manera ordenada y sencilla.

4.6.9 EXADEL STUDIO FOR ECLIPSE

Para integrar nuestro entorno de desarrollo (Eclipse + Apache Tomcat + MySQL + Hibernate) con la tecnología STRUTS, nos ayudamos de un herramienta intermedia.

En este caso sólo se ha barajado el uso de un componente para Eclipse: Exadel Studio (Exadel, Inc 2009).

4.6.9.1 Descripción

Exadel Studio es una herramienta que nos ayuda a integrar distintas tecnologías orientadas, en mayor medida, al desarrollo web dentro en un mismo IDE.

Con Exadel podremos desarrollar los distintos componentes web, del sistema de políticas de privacidad mediante STRUTS, de una manera visual (mediante un diagrama de navegación separado por acciones, formularios y páginas JSP). Por otro lado, Exadel nos ofrece varias herramientas para configurar y utilizar hibernate de una manera integrada con el entorno de desarrollo.

4.6.9.2 Razones para su elección

Se ha decidido seleccionar esta tecnología ya que es la única que se ha encontrado para

integrar Eclipse con STRUTS e Hibernate.

4.7 DISEÑO DETALLADO

Veamos a continuación una especificación detallada de los paquetes que conforman nuestro sistema así como los diagramas de secuencia de las funciones más destacadas.

4.7.1 EL SISTEMA ACP

Se presenta a continuación una especificación detallada del sistema ACP. Recordamos que el sistema ACP es el encargado de devolver respuestas de autorización al recibir las peticiones de autorización en formato CERTILOC y evaluarlas, mediante el formato XACML, contra las políticas de privacidad activas en el sistema.

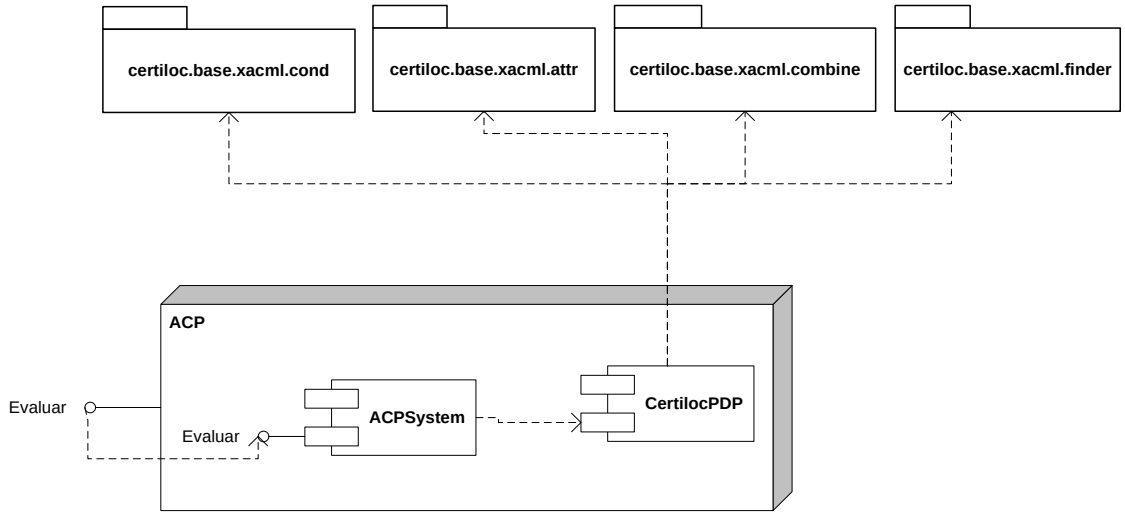


Figura 29. *Diagrama de componentes del Subsistema ACP*

En los siguientes apartados veremos en detalle cada uno de los paquetes y clases de este módulo.

4.7.1.1 Paquete Certiloc.acp

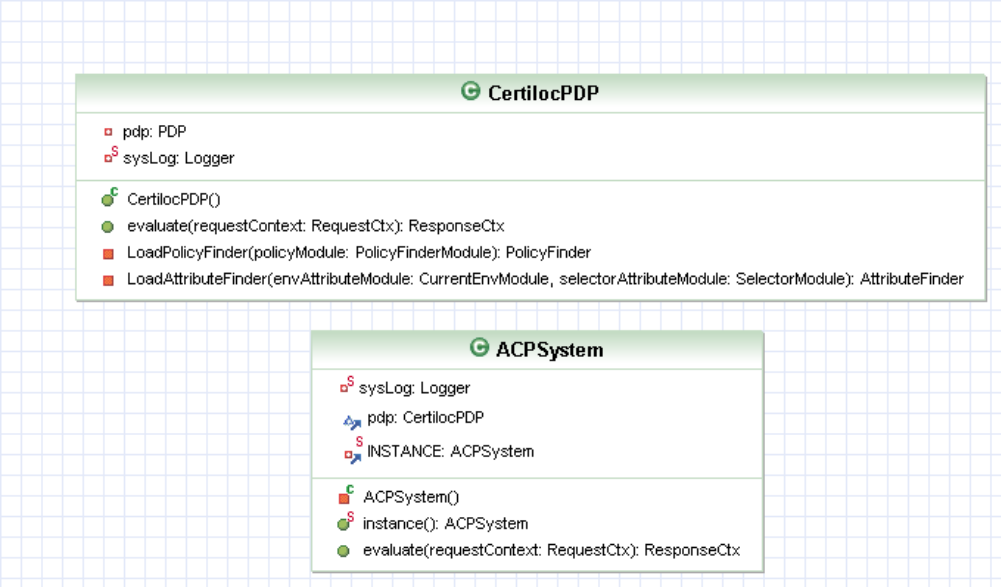


Figura 30. *Paquete Certiloc.acp*

4.7.1.1.1 Clase ACPSystem

4.7.1.1.1.1 Descripción

Clase que proporciona a CERTILOC la implementación de la funcionalidad de evaluación de peticiones en formato XACML.

Esta clase es la fachada (“***Facade pattern”***) para el uso del subsistema ACP.

Esta clase se basa en el patrón de diseño Singleton para asegurar que sólo existe una instancia del objeto en cada ejecución.

4.7.1.1.1.2 Atributos

`INSTANCE:ACPSystem` – La única instancia del ACPSystem creado

`pdp:CertilocPDP` – Instancia de un PDP específico para CERTILOC

`sysLog:Logger` – Instancia para registrar eventos del sistema

4.7.1.1.1.3 Funciones más destacadas

4.7.1.1.1.3.1 La función ACPSystem.evaluate()

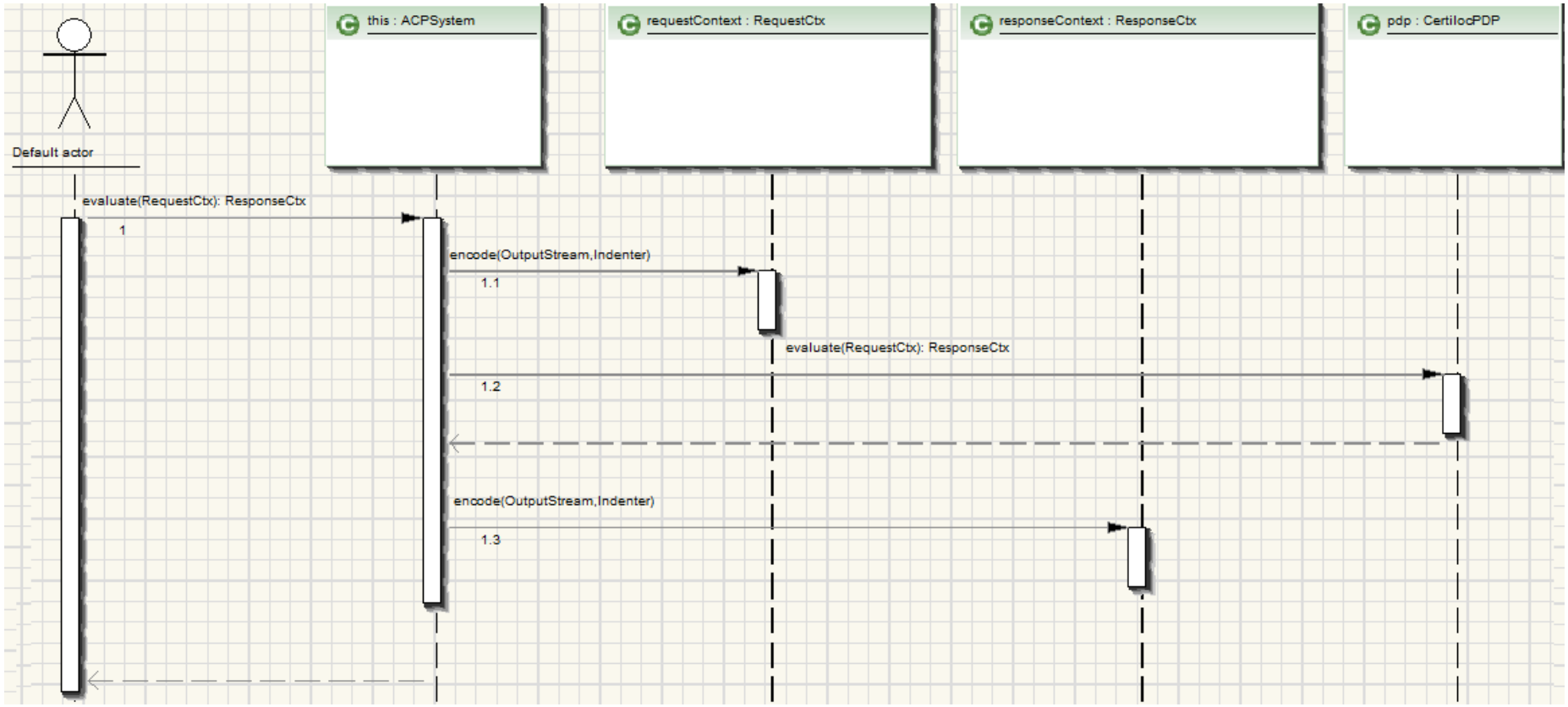


Figura 31. *Secuencia de la función ACPSystem.evaluate()*

4.7.1.1.1.3.1.1 Descripción

Método para evaluar una determinada petición. El módulo **ACP** recibe una petición de autorización en formato XACML y la pasa al **PDP** (Punto de Decisión de Políticas – Definición de XACML apartado 4 del presente documento) para su posterior evaluación contra las políticas activas en el sistema. Este método se repite en la clase **CertilocPDP** ya que, como recordará el lector, la clase a **ACPSystem**, es una simple fachada del sistema que facilita el uso del subsistema **ACP** (la fachada no realiza la evaluación sino que obtiene una respuesta de autorización mediante un **PDP**).

4.7.1.1.1.3.1.2 Parámetros

`requestContext` - La petición en formato XACML

4.7.1.1.1.3.2 Devuelve

Una respuesta de evaluación de la petición en formato XACML

4.7.1.1.2 Clase CertilocPDP

4.7.1.1.2.1 Descripción

Clase que implementa un **PDP** (Punto de Decisión de Políticas – Definición de XACML apartado 4 del presente documento) personalizado para CERTILOC.

Este **PDP** está configurado para funcionar específicamente para CERTILOC. Es decir, está preparado para utilizar los atributos, algoritmos de combinación, las funciones, y módulos de políticas de privacidad propias de CERTILOC (Véanse los apartados 4.7.4, 4.7.6, 4.7.7, 4.7.8 para obtener los detalles de los distintos elementos).

El objetivo de esta clase es evaluar peticiones en **formato XACML** frente a las políticas de privacidad activas, también **en formato XACML**, definidas por los usuarios del sistema CERTILOC.

4.7.1.1.2.2 Atributos

`pdp: PDP` – Instancia de un PDP genérico de XACML

`sysLog:Logger` – Instancia para registrar eventos del sistema

4.7.1.1.2.3 Funciones más destacadas

4.7.1.1.2.3.1 La función CertilocPDP.evaluate()

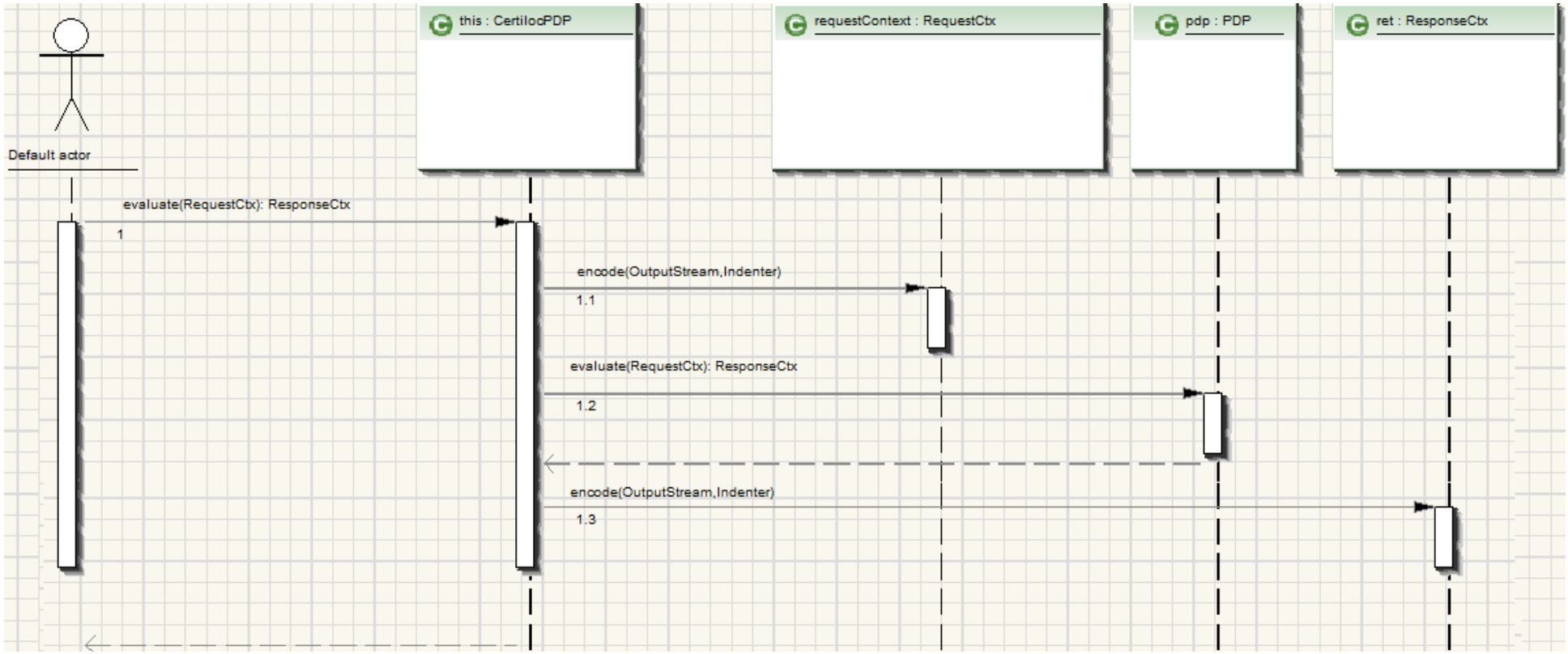


Figura 32. *Secuencia de la función CertilocPDP.evaluate()*

4.7.1.1.2.3.1.1 Descripción

Método para evaluar una determinada petición con un **PDP** (Punto de Decisión de Políticas).

Cada vez que se instancia la clase **ACPSystem**, ésta crea un nuevo **CertilocPDP**. A su vez, cada vez que se crea un **CertilocPDP**, éste crea una instancia de un **PDP,** utilizando la implementación de Sun´s XACML implementation (apartado 5.5 del presente documento) y lo configura para utilizar las políticas activas en el sistema (políticas de privacidad de CERTILOC), los algoritmos de combinación de políticas propios de CERTILOC y sus funciones (**TimeInRangeFunction** y **LocationInRectangleArea** como veremos en capítulos posteriores). El método descrito en el presente apartado, simplemente lanza la evaluación de la petición de autorización contra el **PDP** configurado previamente (al crear un nuevo **CertilocPDP**). Las codificaciones ("encode") mostradas en la Figura 32 nos ayudan a obtener una visualización XML de la petición entrante y la respuesta devuelta. Estas visualizaciones se pueden introducir en los registros de actividad del sistema para ver posibles causas de error y el seguimiento del sistema (para poder hacer esto, el Logger debe haber sido configurado previamente en el nivel DEBUG).

4.7.1.1.2.3.1.2 Parámetros

`requestContext` - La petición en formato XACML

4.7.1.1.2.3.1.3 Devuelve

Una respuesta de evaluación de la petición en formato XACML

4.7.2 EL SISTEMA AGPA

Se presenta a continuación una especificación detallada del sistema AGPA. Recordamos que el sistema AGPA es el encargado de convertir las peticiones en formato nativo del contexto de CERTILOC al formato XACML y lo mismo, pero a la inversa, para las respuestas obtenidas.

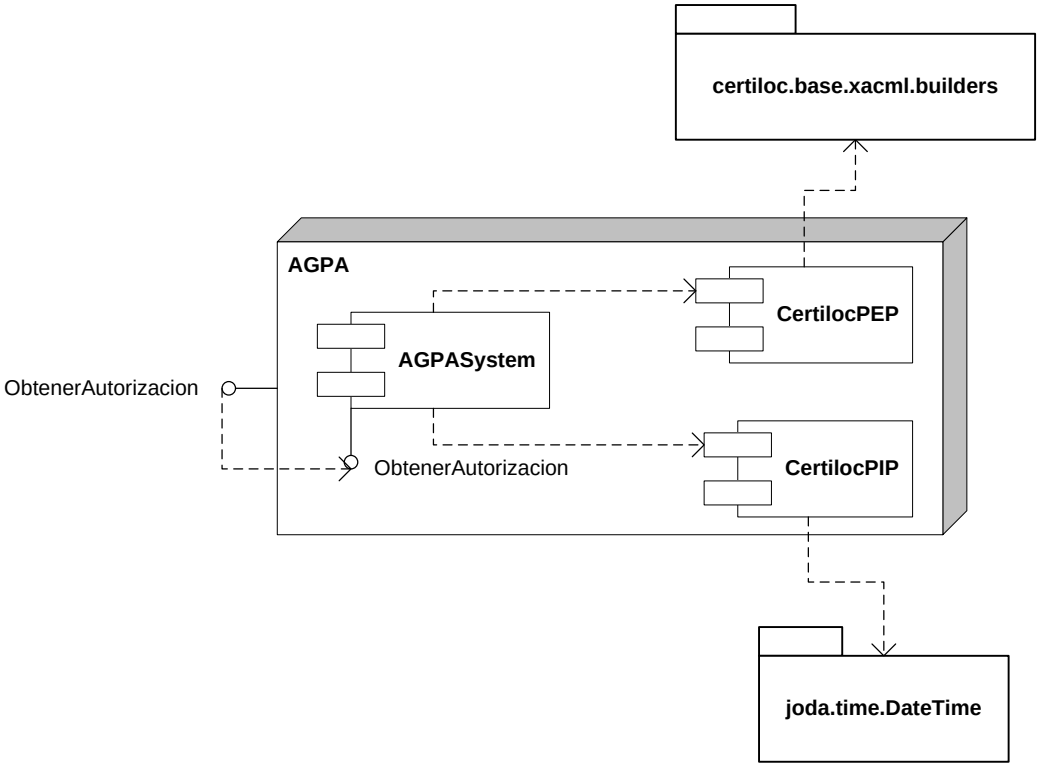


Figura 33. *Diagrama de componentes delSubsistema AGPA*

En los siguientes apartados veremos en detalle cada uno de los paquetes y clases de este módulo.

4.7.2.1 Paquete Certiloc.AGPA

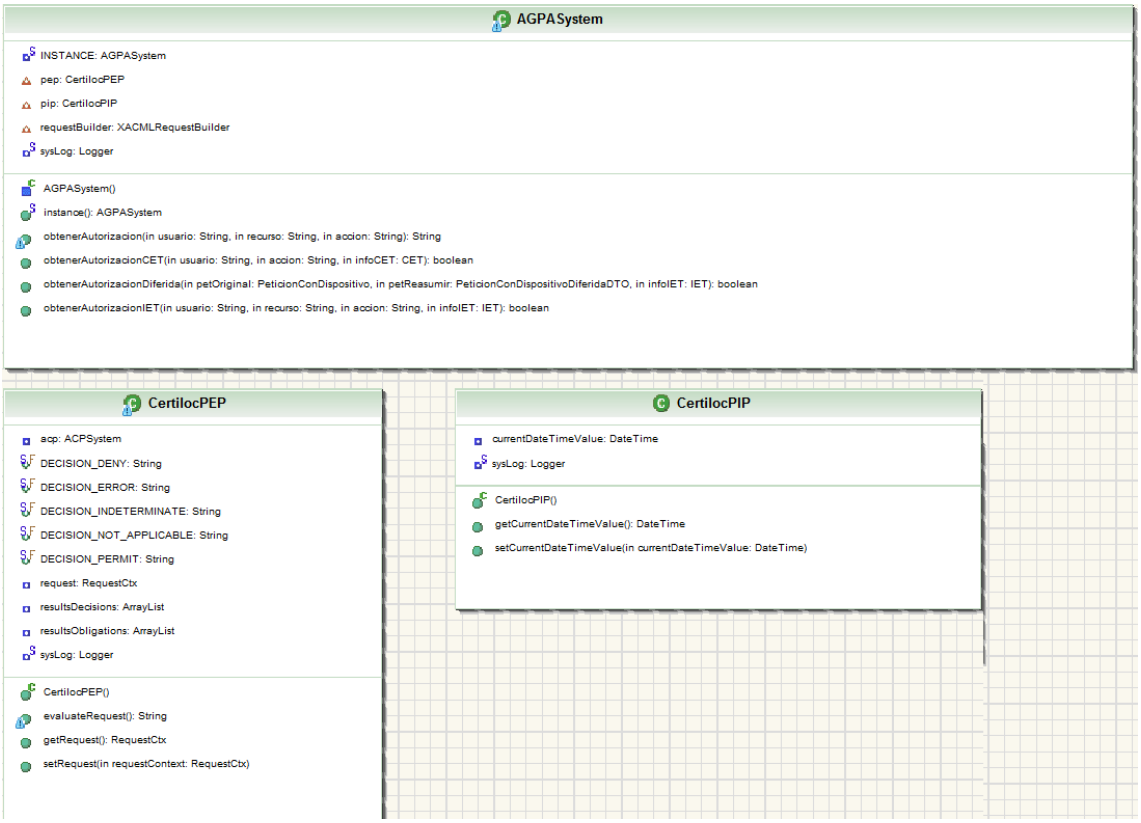


Figura 34. *Paquete Certiloc.agpa*

4.7.2.1.1 La clase AGPASystem

4.7.2.1.1.1 Descripción

Clase que proporciona a CERTILOC la implementación de la funcionalidad de obtención de permisos para realizar acciones sobre recursos, según las políticas de privacidad definidas por los usuarios responsables de cada recurso.

Esta clase recibe las peticiones en el formato nativo de CERTILOC y las convierte al contexto de XACML para pasarlas al sistema ACP.

Esta clase es la fachada (***"Facade pattern"***) para el uso del subsistema AGPA.

Esta clase se basa en el patrón de diseño Singleton para asegurar que sólo existe una instancia del objeto.

4.7.2.1.1.2 Atributos

`INSTANCE:AGPASystem` – Instancia del único AGPASystem existente

`pip: CertilocPIP` – Instancia de un PIP específico para CERTILOC

`pep: CertilocPEP` – Instancia de un PEP específico para CERTILOC

`sysLog:Logger` – Instancia para registrar eventos del sistema

4.7.2.1.1.3 Funciones más destacadas

4.7.2.1.1.3.1 La función AGPASystem.obtenerAutorizacion

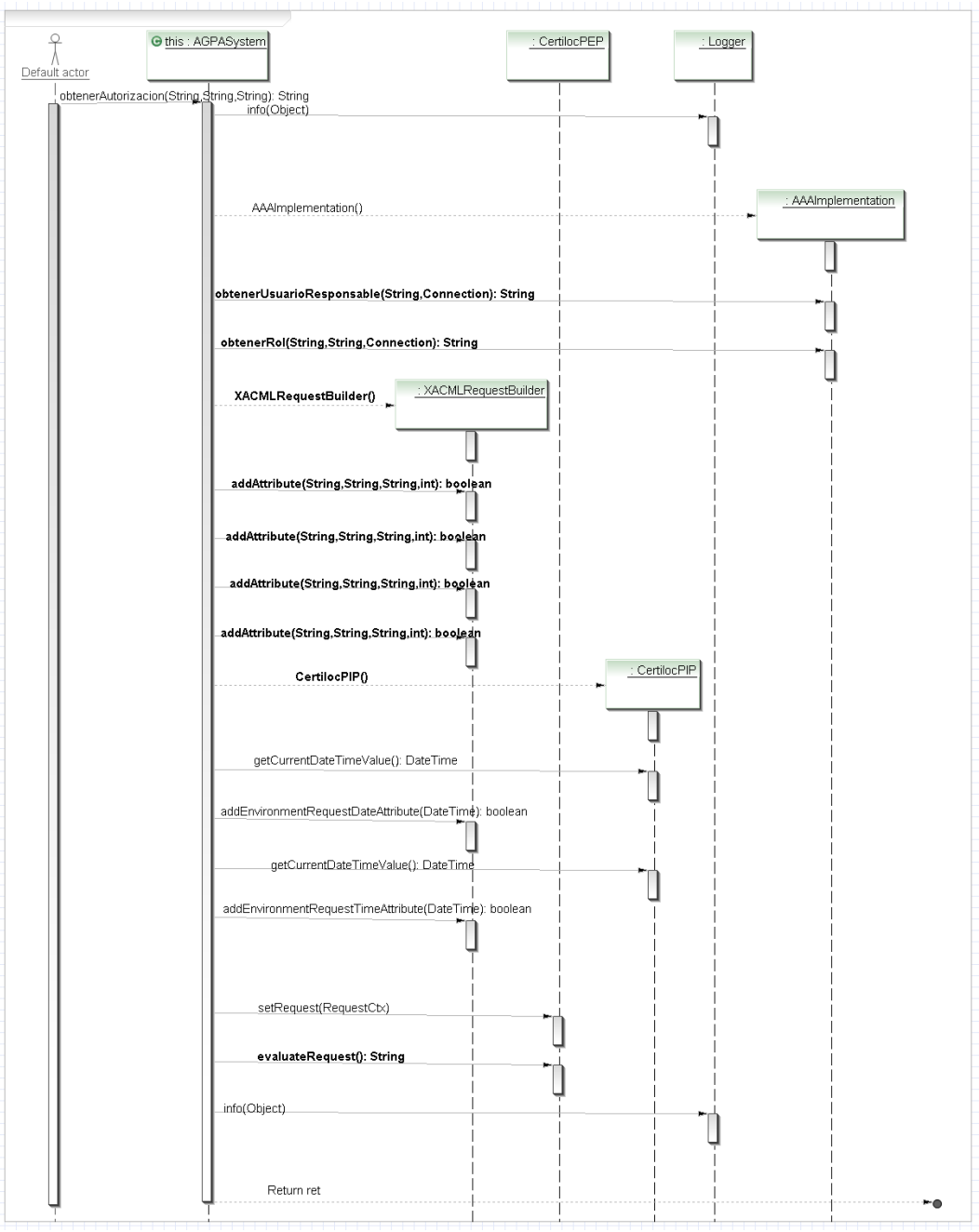


Figura 35. *Secuencia de la función AGPASystem.obtenerAutorizacion*

4.7.2.1.1.3.1.1 Descripción

Método para obtener autorización para realizar un tipo concreto de operación.

Esta función se encarga de convertir una petición de autorización en **formato** CERTILOC (en el contexto de CERTILOC) al **formato XACML** y pasarla al **PEP** (Punto de Ejecución de Políticas - para ver una definición detallada de este elemento ver sección 3.5.1.3.3 del presente documento) para evaluarla con las políticas de privacidad activas en el sistema mediante un **PDP** (Punto de Decisión de Políticas, aptdo. 3.5.1.3.3).

Para convertir la petición de autorización del **formato** CERTILOC al **formato XACML** (antes de pasarla al **PEP**) se ayuda de un constructor de peticiones XACML (**XACMLRequestBuilder**) y de un **PIP** (Punto de Información de Políticas, aptdo. 3.5.1.3.3).

Construye la petición de autorización en formato XACML incluyendo información sobre el **usuario** que solicita la petición (**sujeto** en XACML tal y como se describe en el aptdo. 3.5 del presente documento), el **dispositivo** para el que se solicita la operación (**recurso** en XACML, aptdo. 3.5 del presente documento) y la **acción** que se quiere realizar (**acción** en XACML, aptdo. 3.5 del presente documento). También incluye la información sobre la **hora** y **fecha** de la creación de la petición (dentro del **entorno** de la petición en XACML, aptdo. 3.5 del presente documento). La información sobre la hora y la fecha actuales la obtiene ayudándose del **PIP.**

Tal y como muestra el diagrama, para poder obtener la información del **propietario del dispositivo** para el que se hace la solicitud de autorización y el **rol del usuario solicitante** frente al usuario responsable del dispositivo, este método hace uso de la clase **AAAImplementation** que nos devuelve esta información.

Una vez está creada la **petición en formato XACML**, la pasa al **PEP** para su posterior evaluación contra las políticas de privacidad activas en el sistema.

4.7.2.1.1.3.1.2 Parámetros

`usuario` - El usuario que solicita la autorización.

`recurso` - El recurso sobre el que se quiere realizar la operación.

`accion` - la acción que se quiere realizar.

4.7.2.1.1.3.1.3 Devuelve

`Verdadero` – En el caso en el que la operación es autorizada.

`Falso` – En el caso en el que la operación no es autorizada

4.7.2.1.1.3.2 La función AGPASystem.obtenerAutorizacionDiferida

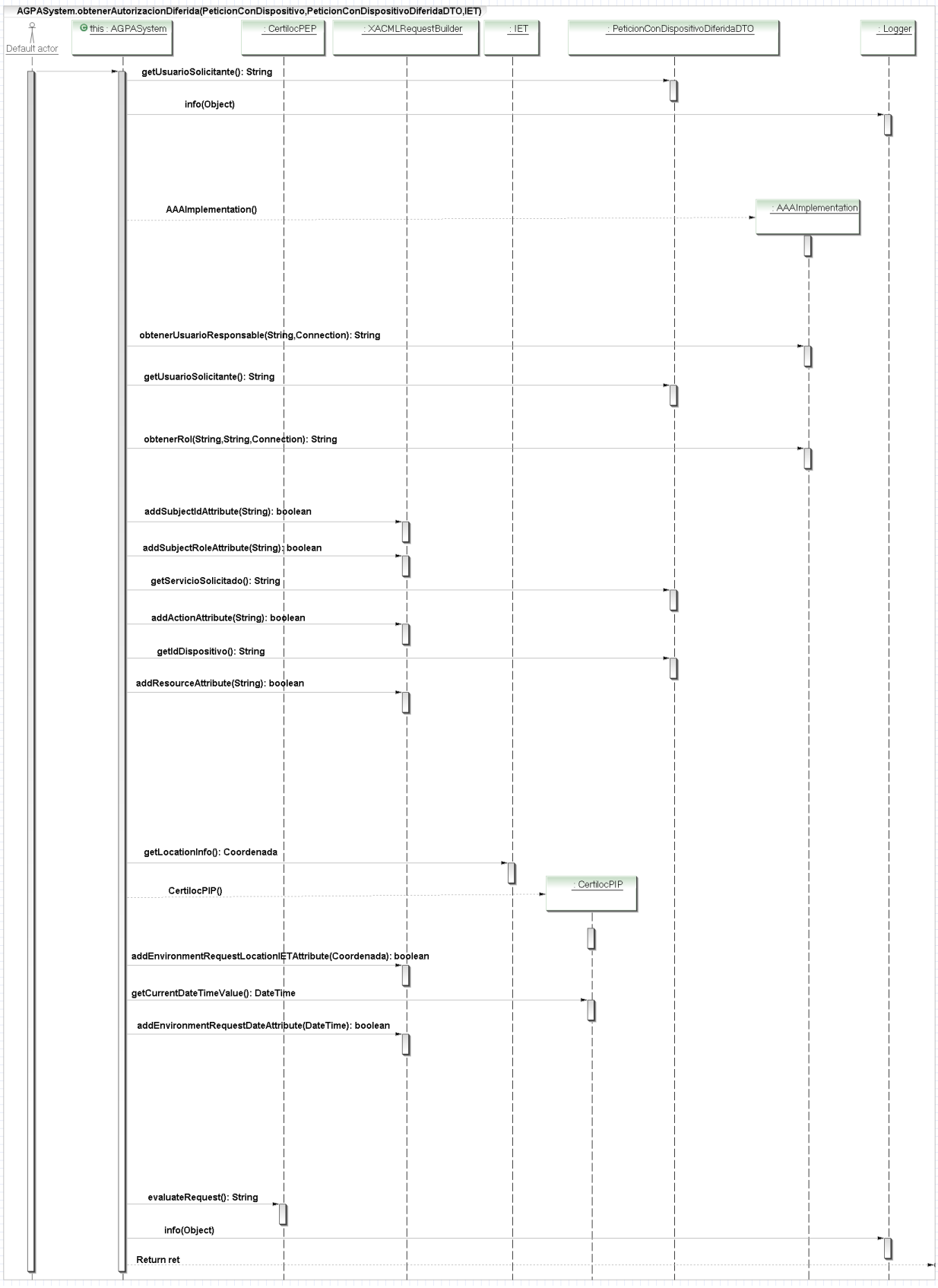


Figura 36. *Secuencia de la función AGPASystem.obtenerAutorizacionDiferida*

4.7.2.1.1.3.2.1 Descripción

Método para obtener autorización para realizar una operación diferida.

El funcionamiento de este método es **análogo al método descrito en el apartado 4.7.2.1.1.3.1.1** pero al contrario que éste, obtiene los distintos parámetros para la conversión de la petición (del **formato nativo de** CERTILOC al **formato de XACML)**, directamente de los datos contenidos en la **PeticionConDispositivoDiferida** que entra por parámetro. Estos datos son: El **id del usuario solicitante** de la petición (el id del **sujeto** en **XACML**), el **servicio solicitado** en la petición (la **acción** en **XACML**), la **información espacio temporal de la solicitud** o petición de autorización original (IET, englobada dentro del **entorno** de la petición en **XACML**) y finalmente el **dispositivo** para el que se hace la solicitud o petición de autorización (el **recurso** en **XACML**).

4.7.2.1.1.3.2.2 Parámetros

`Peticion` – Petición actual que se quiere realizar.

`PeticionDiferida` - Petición inicial que se quiere realizar.

`infoIET` – Información IET actual del dispositivo sobre el que se realiza la petición.

4.7.2.1.1.3.2.3 Devuelve

`Verdadero` – En el caso en el que la operación es autorizada.

`Falso` – En el caso en el que la operación no es autorizada

4.7.2.1.1.3.3 La función AGPASystem.obtenerAutorizacionCET

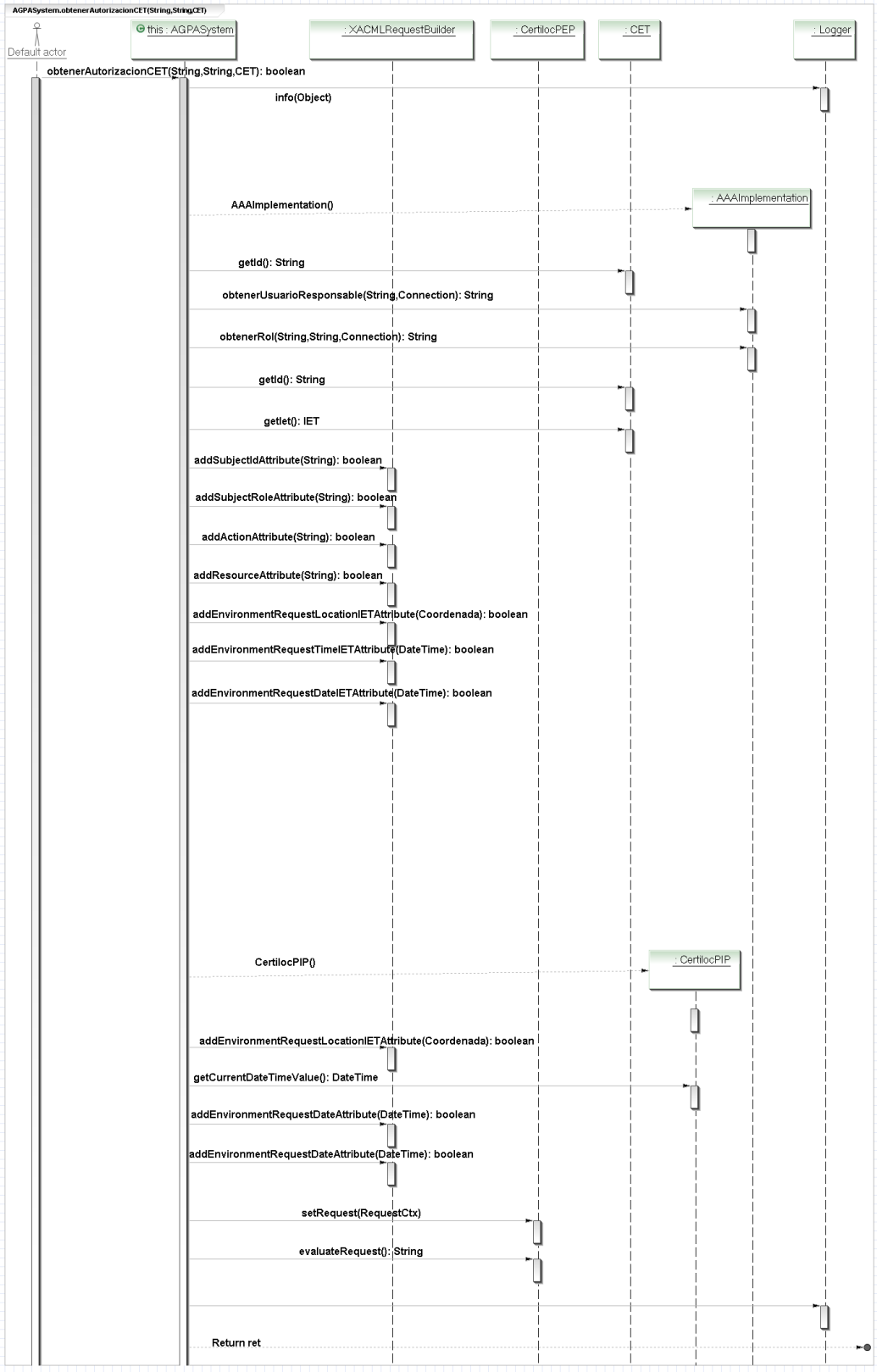


Figura 37. *Secuencia de la función AGPASystem.obtenerAutorizacionCET*

4.7.2.1.1.3.3.1 Descripción

Método para obtener autorización para realizar una acción sobre un certificado CET.

El funcionamiento de este método es **análogo al método descrito en el apartado 4.7.2.1.1.3.1.1** pero al contrario que éste, obtiene los distintos parámetros para la conversión de la petición (del **formato nativo de** CERTILOC al **formato de XACML)**, directamente de los datos contenidos en el **CET** (Certificado Espacio Temporal) que recibe por parámetro. Estos datos son: El **id del certificado** para el que se solicita la petición (el id del **recurso** en **XACML**) y la **información espacio temporal** del certificado (englobada dentro del **entorno** de la petición en **XACML**). La información sobre el **id del usuario** solicitante de servicios (**sujeto** en **XACML**) entra por parámetro y la **acción** a realizar sobre el certificado, se reciben por parámetro.

4.7.2.1.1.3.3.2 Parámetros

`usuario` - El usuario que solicita la autorización

`accion` - Acción que se quiere realizar sobre el recurso

`infoCET` - El certificado sobre el que se realiza la petición

4.7.2.1.1.3.3.3 Devuelve

`Verdadero` – En el caso en el que la operación es autorizada.

`Falso` – En el caso en el que la operación no es autorizada

4.7.2.1.1.3.4 La función AGPASystem.obtenerAutorizacionIET

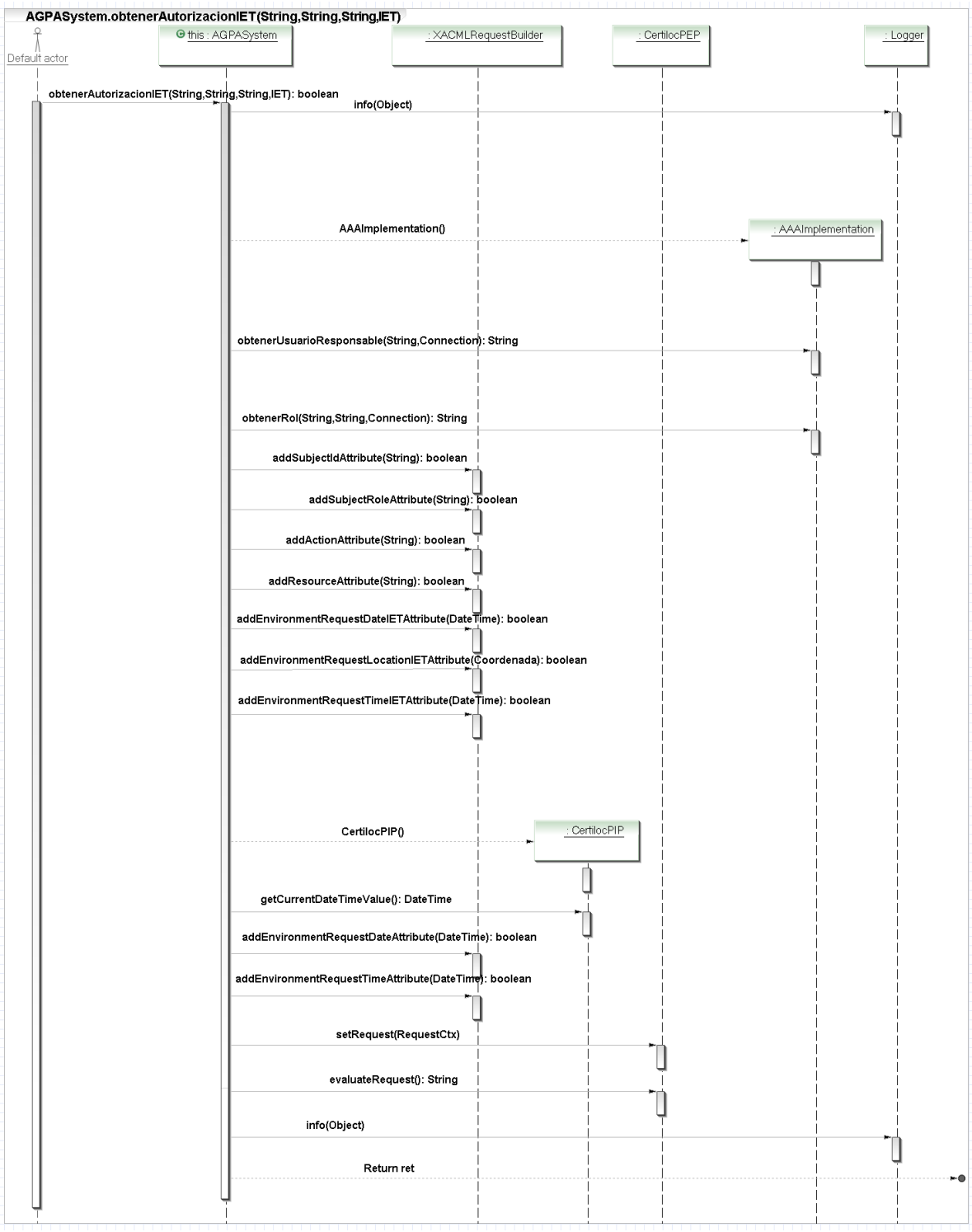


Figura 38. *Secuencia de la función AGPASystem.obtenerAutorizacionIET*

4.7.2.1.1.3.4.1 Descripción

Método para obtener autorización para realizar una acción, teniendo en cuenta la información IET, referente al dispositivo pasado por parámetro.

El funcionamiento de este método es **análogo al método descrito en el apartado 4.7.2.1.1.3.1.1**. Obtiene los distintos parámetros para la conversión de la petición (del **formato nativo de** CERTILOC al **formato de XACML)**, directamente de los datos que recibe por parámetro. Estos datos son: El **id del usuario solicitante** de la petición (el id del **sujeto** en **XACML**), el **servicio solicitado** en la petición (la **acción** en **XACML**), la **información espacio temporal del dispositivo para el que se realiza la petición** (**IET**, englobada dentro del **entorno** de la petición en **XACML**) y finalmente el **dispositivo** para el que se hace la solicitud o petición de autorización (el **recurso** en **XACML**).

4.7.2.1.1.3.4.2 Parámetros

`usuario` - El usuario que solicita la autorización

`recurso` - El recurso sobre el que se quiere realizar la acción

`accion` - Acción que se quiere realizar sobre el recurso

`infoIET` - Informacion espacio-temporal sobre la localización de recurso y la hora en que se hace la petición.

4.7.2.1.1.3.4.3 Devuelve

`Verdadero` – En el caso en el que la operación es autorizada.

`Falso` – En el caso en el que la operación no es autorizada

4.7.2.1.2 La clase CertilocPEP

4.7.2.1.2.1 Descripción

Clase que implementa un XACML PEP (Policy Enforcement Point) específico para CERTILOC.

El objetivo de esta clase es pasar un contexto de petición al sistema ACP para su evaluación y extraer el resultado de la respuesta obtenida.

4.7.2.1.2.2 Atributos

`acp: ACPSystem` – Instancia hacia el módulo ACP de CERTILOC

`request:RequestCtx` – Contexto de petición de en formato XACML

`resultsDecisions:ArryList` – Array con las decisiones obtenidas en formato XACML

`resultsObligations:ArrayList` – Array con las obligaciones asociadas a las decisiones obtenidas en formato XACML

`sysLog:Logger` – Instancia para registrar eventos del sistema

4.7.2.1.2.3 Funciones más destacadas

4.7.2.1.2.3.1 La función CertilocPEP.evaluateRequest

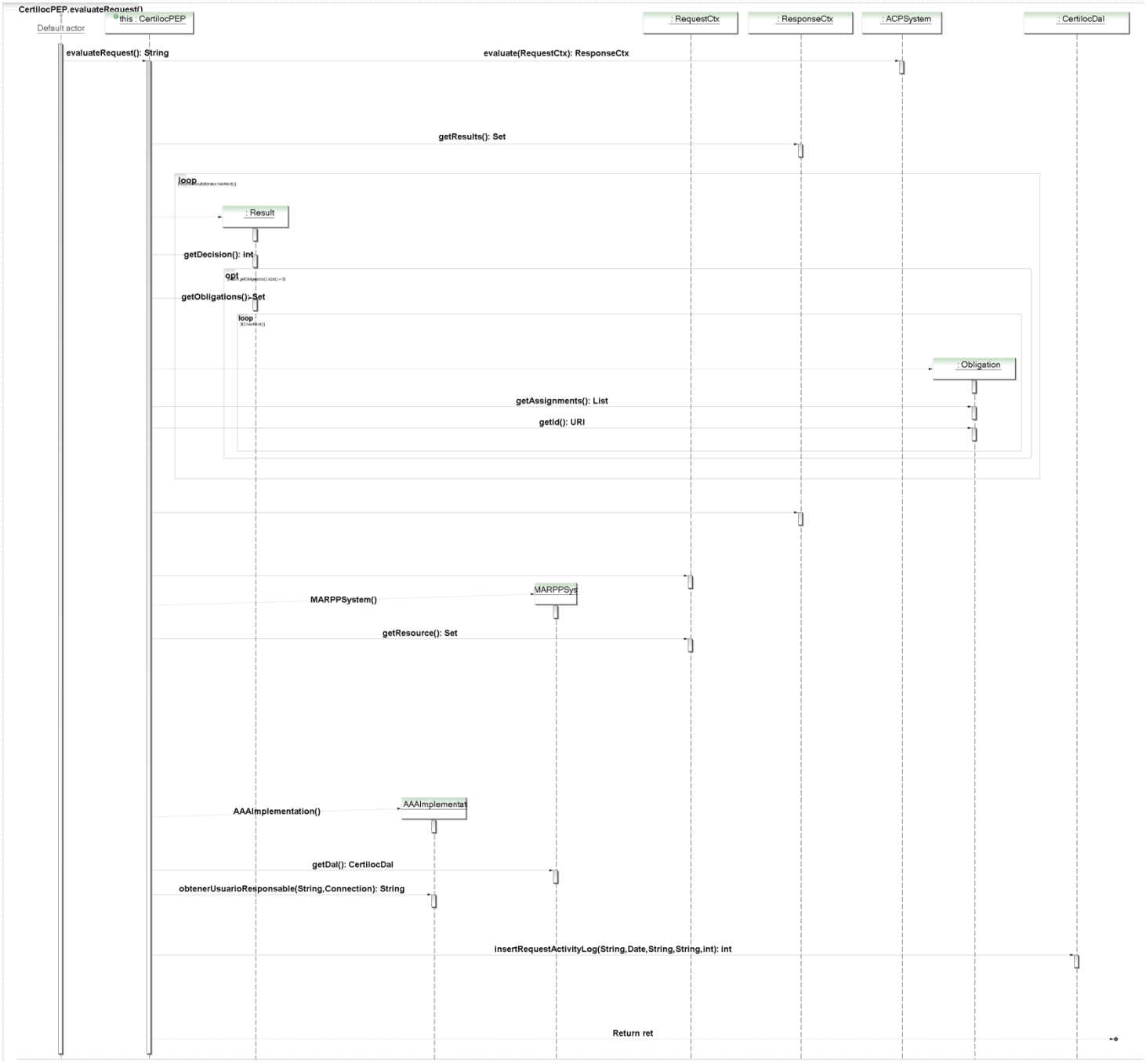


Figura 39. *Secuencia de la función CertilocPEP.evaluateRequest*

4.7.2.1.2.3.1.1 Descripción

Método evaluador de peticiones en formato XACML.

Utiliza una instancia del subsistema **ACPSystem** (aptdo. 4.7.1) para evaluar, mediante un **CertilocPDP**, una petición previamente convertida al formato **XACML**.

Al obtener la respuesta en **formato XACML** la examina en busca de obligaciones. Si la respuesta contiene obligaciones, las extrae y pasa al formato de CERTILOC. En esta versión del demostrador de CERTILOC, la respuesta que se devuelve al sistema **AAA** (aptdo. 4.2) únicamente devuelve verdadero o falso (permitir o denegar) con lo que las obligaciones no se propagan en la respuesta sino que quedan en este punto de la aplicación. Sería interesante poder devolver las obligaciones al sistema **AAA** para que éste las muestre al dar la respuesta al usuario solicitante de servicios inicial.

Una vez ha extraídas las distintas obligaciones, genera un registro de actividad de petición (aptdo. 4.7.9.3.12 del presente documento), asociado al propietario del dispositivo para el que se recibe la petición. Para extraer el responsable del dispositivo, hace una consulta al módulo **AAA** (custodia la información sobre los propietarios de cada dispositivo).

El registro de actividad de petición es lo último que se genera en la función ya que, llegados a este punto, ya sabemos la respuesta que se habrá devuelto y la podremos incluir en el registro de actividad.

4.7.2.1.2.3.1.2 Devuelve

Una cadena con el resultado de la evaluación (Tal y como se indica en la Tabla 98 puede ser Permitir, Denegar, Indeterminado o No Aplicable).

4.7.2.1.3 La clase CertilocPIP

4.7.2.1.3.1 Descripción

Clase que implementa un XACML PIP (Punto de Información de Políticas) específico para CERTILOC.

El objetivo de esta clase es obtener atributos de entorno relacionados con el momento de la creación de una petición de autorización. Básicamente, al crearse una instancia de esta clase, se registran los valores del tiempo y la fecha actuales. Estos valores ser podrán devolver y actualizar, en un futuro, para proveer información de entorno sobre la creación de una petición.

4.7.2.1.3.2 *Atributos*

`currentDateTimeValue: DateTime` – Atributo para la hora y la fecha actuales

`sysLog:Logger` – Instancia para registrar eventos del sistema

4.7.2.1.3.3 *Funciones más destacadas*

4.7.3 PAQUETE CERTILOC.BASE.XACML

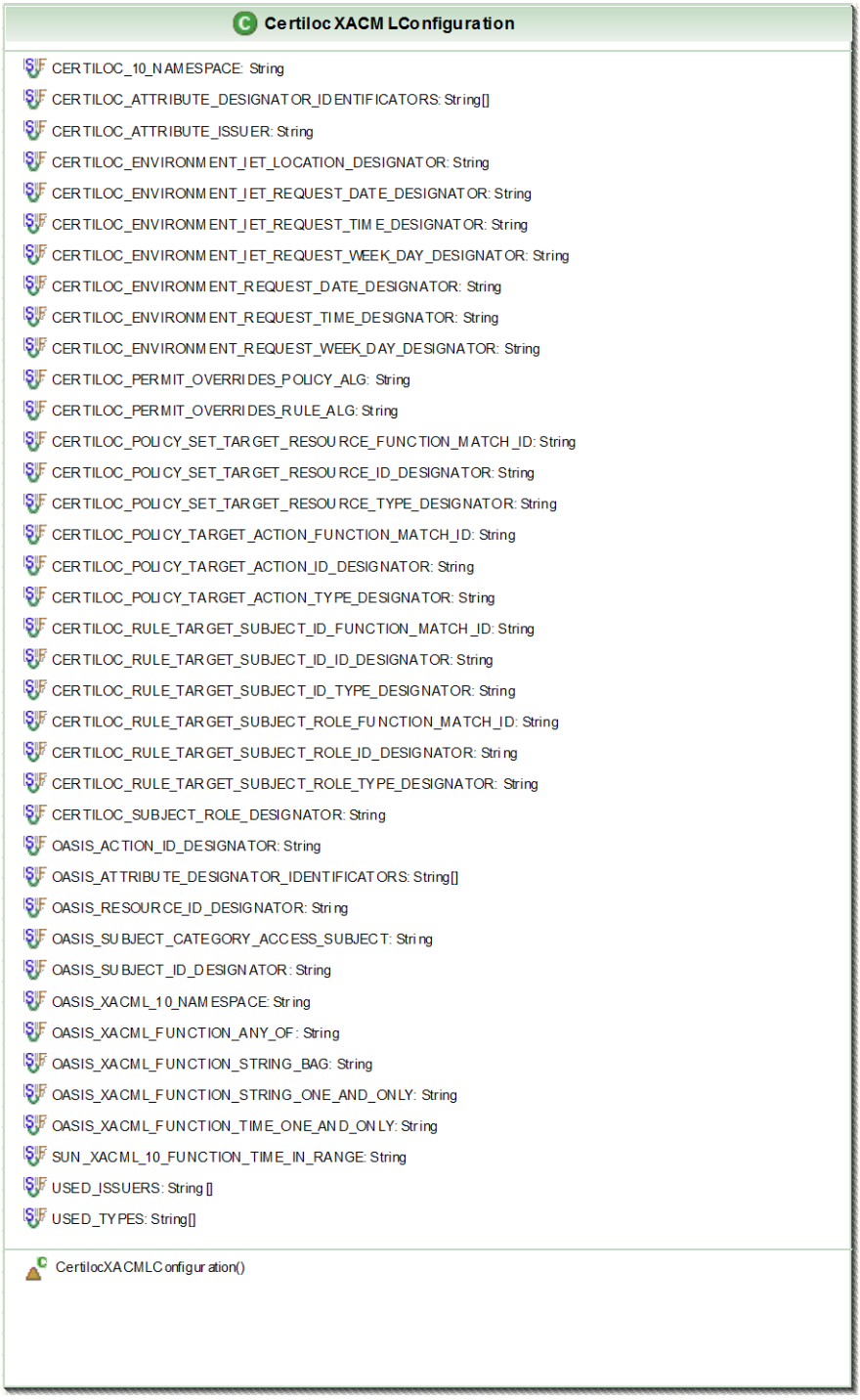


Figura 40. *Paquete Certiloc.base.xacml*

4.7.3.1 La clase CertilocXACMLConfiguration

4.7.3.1.1 Descripción

Esta clase contiene todas las constantes que va a utilizar el sistema XACML de CERTILOC.

Estas constantes no están definidas en el API de "Sun's XACML Implementation" o son exclusivas del sistema de políticas de privacidad de CERTILOC.

4.7.3.1.2 Atributos

Son todos constantes. Dirigirse al código fuente si se quiere saber su valor exacto.

4.7.4 PAQUETE CERTILOC.BASE.XACML.ATTR

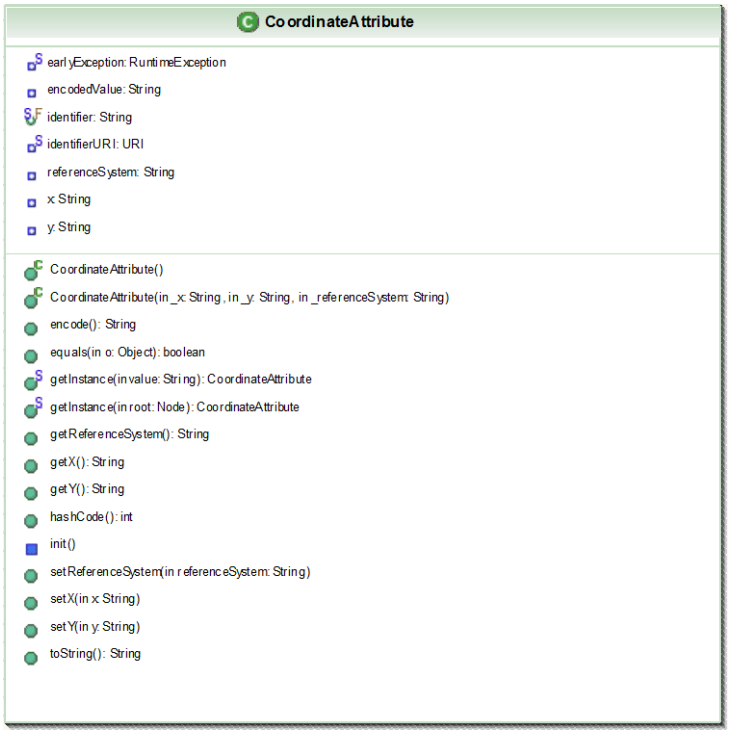


Figura 41. *Paquete Certiloc.base.xacml.attr*

4.7.4.1 La clase CoordinateAttribute

4.7.4.1.1 Descripción

Implementará el tipo de atributo "Coordenada", propio de CERTILOC, para el sistema de políticas de privacidad XACML.

Esta clase contiene métodos para crear una coordenada a partir de una cadena expresada de la siguiente manera “X,Y”.

4.7.4.1.2 Atributos

`earlyException:RuntimeException` – Definición estática hacia una instancia de una excepción en tiempo de ejecución.

`encodedValue: String` – Cadena que define el valor codificado en XACML de la instancia del atributo de coordenada

`indentifier: String` – Constante que define el identificador XACML de los atributos de coordenada

`indentifierURI: String` – Constante que define el identificador XACML, con morfología URI (The Internet Society 1998), de todos los atributos de coordenada

`referenceSystem: String` – Sistema de referencia que estamos utilizando para definir el espacio en el que se encuentra la coordenada

`x: String` – Coordenada X

`y: String` – Coordenada Y

`sysLog:Logger` – Instancia para registrar eventos del sistema

4.7.4.1.3 Funciones más destacadas

Esta clase debe implementar funciones para obtener un valor de atributo de coordenada a partir de una cadena y una raíz XML.

También debe implementar una función para imprimir la instancia en formato XACML.

4.7.5 PAQUETE CERTILOC.XACML.BUILDERS

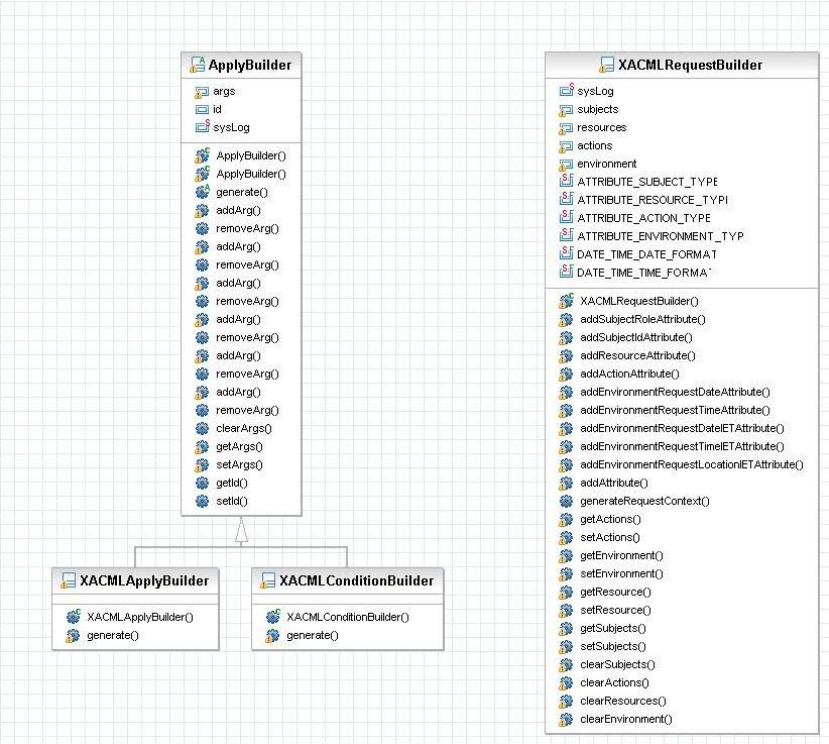


Figura 42. *Paquete Certiloc.xacml.builders*

4.7.5.1 La clase XACMLRequestBuilder

4.7.5.1.1 Descripción

Clase que implementa un constructor de contextos de petición en formato XACML.

4.7.5.1.2 Atributos

`ATTRIBUTE_ACTION_TYPE:int` – Constante que define los atributos de **acción**.

`ATTRIBUTE_ENVIRONMENT_TYPE:int` – Constante que define los atributos de **entorno**.

`ATTRIBUTE_RESOURCE_TYPE:int` – Constante que define los atributos de **recurso**.

`ATTRIBUTE_SUBJECT_TYPE:int` – Constante que define los atributos del **sujeto**.

`DATE_TIME_DATE_FORMAT:String` – Cadena que define el formato para las fechas.

`DATE_TIME_TIME_FORMAT:String` – Cadena que define el formato para las horas.

`Actions` – Conjunto de atributos de la acción que se va a llevar a cabo.

`Environment` – Conjunto de atributos del entorno en el que se lleva a cabo la petición.

`Reources` – Conjunto de atributos sobre el recurso del que se realiza la petición.

`Subjects` – Conjunto de atributos sobre el sujeto que solicita la petición.

`sysLog:Logger` – Instancia para registrar eventos del sistema

4.7.5.1.3 Funciones más destacadas

4.7.5.1.3.1 La función XACMLRequestBuilder.addAttribute

4.7.5.1.3.1.1 Descripción

Método para añadir un atributo concreto a la petición que vamos a construir. El atributo se añade al conjunto de parámetros de sujeto, de acción, de entorno o de recurso según el **tipo de atributo** que se especifica por parámetro.

4.7.5.1.3.1.2 Parámetros

`value` - valor del atributo.

`id` - identificador del atributo con morfología URI (The Internet Society 1998).

`attributeIssuer` - el emisor del atributo.

`attributeType` - tipo de atributo.

4.7.5.1.3.2 La función XACMLRequestBuilder.generateRequestContext

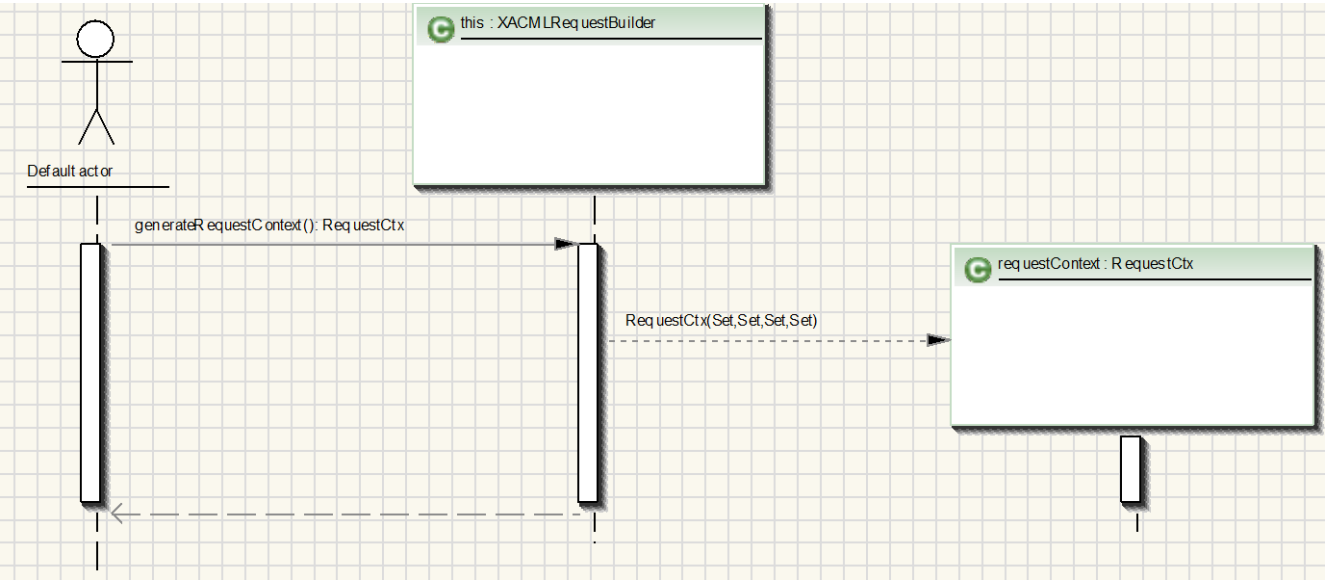


Figura 43. *Secuencia de la función XACMLRequestBuilder.generateRequestContext*

4.7.5.1.3.2.1 Descripción

Método para convertir una petición de autorización del **formato de** CERTILOC al **formato XACML**.

Genera una petición en formato XACML, basándose en los sujetos, recursos, acciones y entorno de la petición CERTILOC.

4.7.5.1.3.2.2 Devuelve

Un contexto de petición en formato XACML.

4.7.5.2 La clase ApplyBuilder

4.7.5.2.1 Descripción

Clase abstracta para un constructor de aplicativos y condiciones de XACML.

4.7.5.2.2 Atributos

`args:List` – Lista de argumentos para realizar aplicar a la función del aplicativo o condición

`id:String` – identificador, con morfología de tipo URI (The Internet Society 1998),de la función XACML que ejecutamos en el aplicativo o la condición

`sysLog:Logger` – Instancia para registrar eventos del sistema

4.7.5.2.3 Funciones más destacadas

4.7.5.2.3.1 ApplyBuilder.generate()

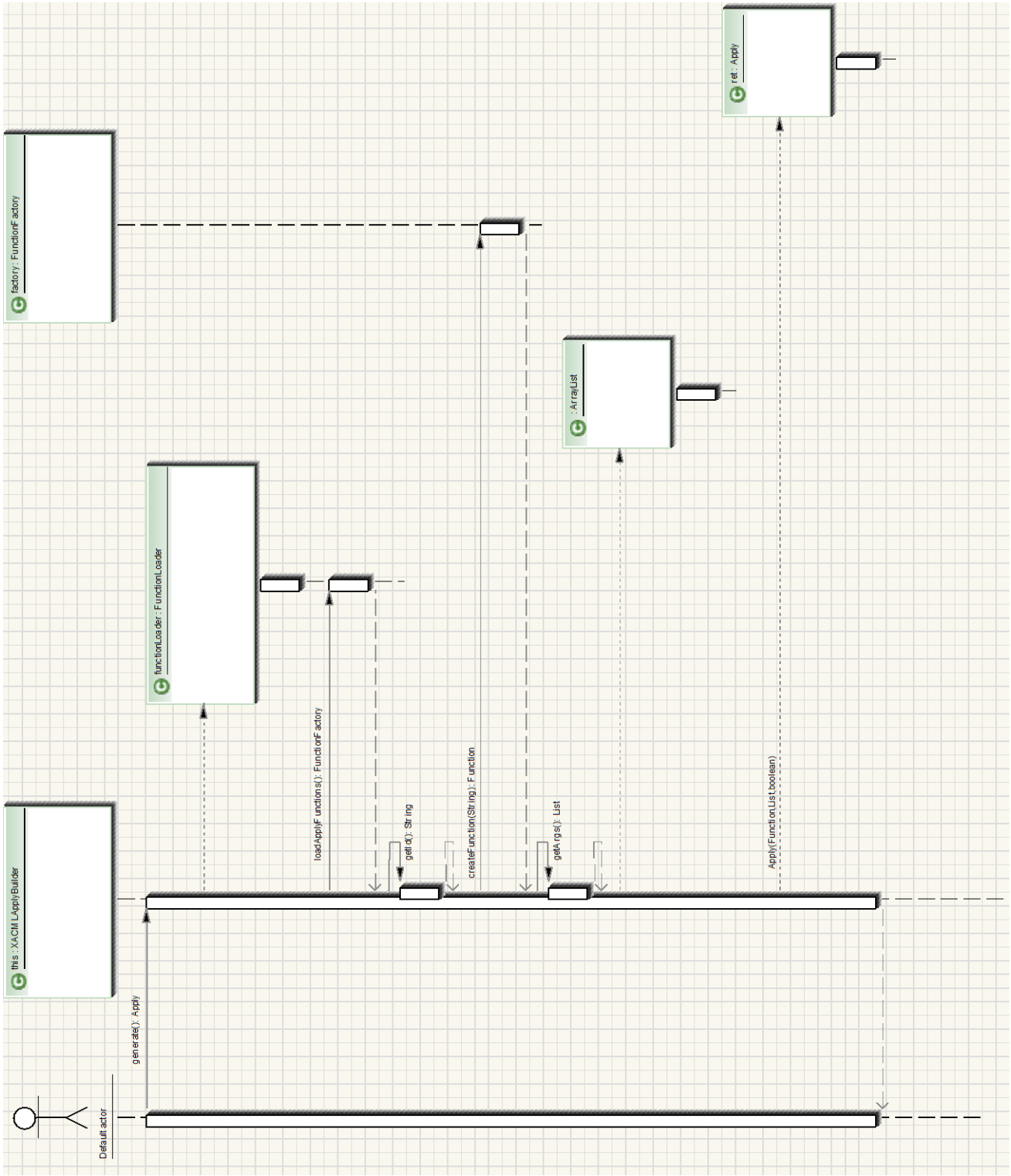


Figura 44. *La función ApplyBuilder.generate()*

4.7.5.2.3.1.1 Descripción

Función abstracta que deben definir las clases hijas de ésta. El objetivo es convertir el aplicativo o la condición del **formato nativo de** CERTILOC al **formato XACML**. Este método, carga todas las funciones definidas en el sistema (aptdo. 4.7.7.3) y genera un aplicativo o una condición, con **formato XACML,** a partir de los argumentos del mismo.

4.7.5.3 La clase XACMLApplyBuilder

4.7.5.3.1 Descripción

Clase para un constructor de aplicativos XACML.

4.7.5.3.2 Atributos

Heredados de la clase padre.

4.7.5.3.3 Funciones más destacadas

4.7.5.3.3.1 XACMLConditionBuilder.generate()

4.7.5.3.3.1.1 Descripción

Función que devuelve un aplicativo con formato XACML a partir de los atributos del aplicativo CERTILOC (sus argumentos y su identificador XACML).

4.7.5.3.3.1.2 Devuelve

Aplicativo en formato XACML

4.7.5.4 La clase XACMLConditionBuilder

4.7.5.4.1 Descripción

Clase para un constructor de condiciones XACML.

4.7.5.4.2 Atributos

Heredados de la clase padre.

4.7.5.4.3 Funciones más destacadas

4.7.5.4.3.1 XACMLConditionBuilder.generate()

4.7.5.4.3.1.1 Descripción

Función que devuelve una condición XACML a partir de los atributos de la condición CERTILOC (definida por sus atributos, sus argumentos y su id).

4.7.5.4.3.1.2 Devuelve

Condición en formato XACML

4.7.6 PAQUETE CERTILOC.BASE.XACML.COMBINE

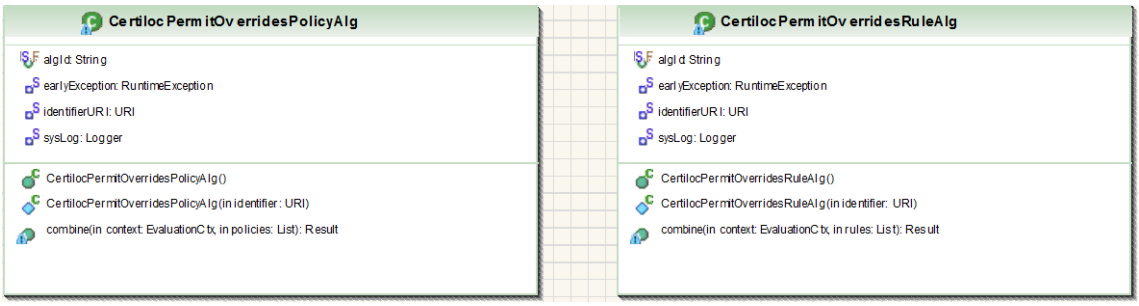


Figura 45. *Paquete Certiloc.base.xacml.combine*

4.7.6.1 La clase CertilocPermitOverridesPolicyAlg

4.7.6.1.1 Descripción

Esta clase implementa el algoritmo estándar "Permit Overrides" (Preferencia a Permitir) de combinación de políticas creado para CERTILOC. Con esta clase podemos saber cuáles son las políticas responsables de la respuesta de autorización que se evalúa contra una petición determinada. Necesitamos implementarla para crear registros sobre la actividad de las políticas.

4.7.6.1.2 Atributos

`algId:String` – Constante que define el identificador XACML del algoritmo.

`earlyException:RuntimeException` – Definición estática hacia una instancia de una excepción en tiempo de ejecución.

`identifierURI:URI` – identificador XACML con morfología URI (The Internet Society 1998).

`sysLog:Logger` – Instancia para registrar eventos del sistema

4.7.6.1.3 Funciones más destacadas

4.7.6.1.3.1 CertilocPermitOverridesPolicyAlg.combine()

4.7.6.1.3.1.1 Descripción

Busca políticas coincidentes, que devuelvan una respuesta a la petición recibida.

4.7.6.1.3.1.2 Parámetros

`context:EvaluationContext` – Contexto de evaluación

`policies:List` – Lista de políticas activas del sistema (deben estar en formato XACML)

4.7.6.1.3.1.3 Devuelve

El resultado en formato XACML.

4.7.6.2 La clase CertilocPermitOverridesRuleAlg

4.7.6.2.1 Descripción

Este es el algoritmo estándar "Permit Overrides" (Preferencia a Permitir) de combinación de reglas. Con esta clase podemos saber cuáles son las reglas responsables de la respuesta de autorización que se evalúa contra una petición determinada. Necesitamos implementarla para crear registros sobre la actividad de las reglas del sistema y su influencia en la respuesta devuelta al usuario.

4.7.6.2.2 Atributos

`algId:String` – Constante que define el identificador XACML del algoritmo.

`earlyException:RuntimeException` – Definición estática hacia una instancia de una excepción en tiempo de ejecución.

`identifierURI:URI` – identificador XACML con morfología URI (The Internet Society 1998).

`sysLog:Logger` – Instancia para registrar eventos del sistema

4.7.6.2.3 Funciones más destacadas

4.7.6.2.3.1 CertilocPermitOverridesRuleAlg.combine()

4.7.6.2.3.1.1 Descripción

Busca reglas coincidentes con la petición y, que devuelvan una respuesta a la petición recibida.

4.7.6.2.3.1.2 Parámetros

`context:EvaluationContext` – Contexto de evaluación

`rules:List` – Lista de reglas activas de las políticas definidas en el sistema (deben estar en formato XACML)

4.7.6.2.3.1.3 Devuelve

El resultado en formato XACML.

4.7.7 PAQUETE CERTILOC.BASE.XACML.COND

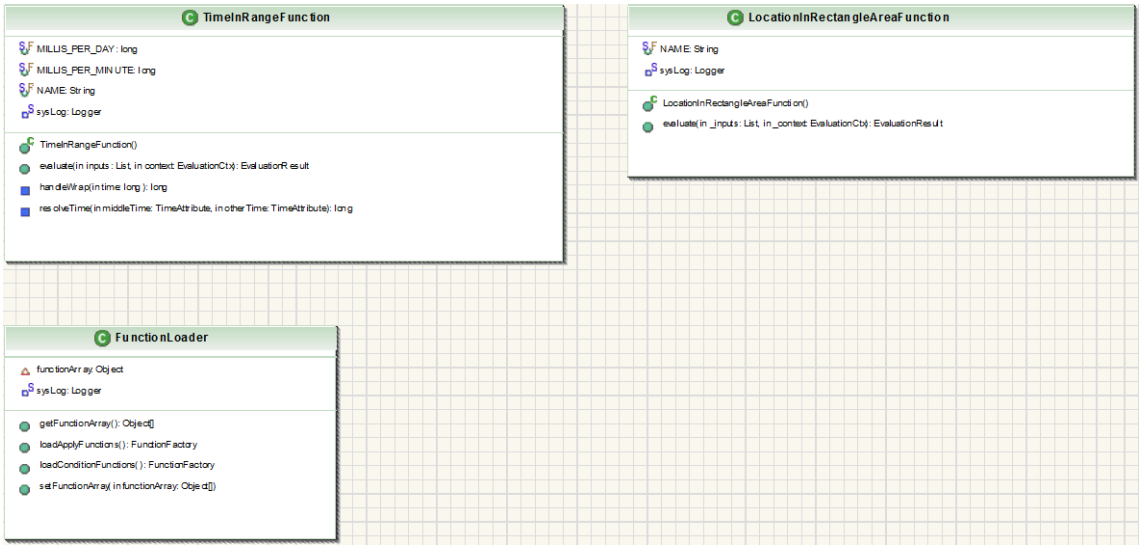


Figura 46. *El paquete Certiloc.base.xacml.cond*

4.7.7.1 La clase TimeInRangeFunction

4.7.7.1.1 Descripción

Esta clase implementa la función "Time In Range", es decir hora dentro del rango. Recibe tres parámetros y devuelve verdadero en caso que el primer valor pasado por parámetro sea una hora contenida entre las otras dos que se pasan.

Esta no es una función estándar en XACML 1.0 pero se ha propuesto para introducirla en XACML 2.0 (muy probablemente se incluirá).

Esta función sólo acepta rangos de tiempo de, como máximo, 24 horas.

4.7.7.1.2 Atributos

`MILLIS_PER_DAY:long` – Constante que define la cantidad de milisegundos en un día

`MILLIS_PER_MINUTE:long` – Constante que define la cantidad de milisegundos en un minuto

`NAME:String` – identificador XACML de la función.

`sysLog:Logger` – Instancia para registrar eventos del sistema

4.7.7.1.3 Funciones más destacadas

4.7.7.1.3.1 La función TimeInRangeFunction.evaluate()

4.7.7.1.3.1.1 Descripción

Evalúa la función "Time in range" (tiempo dentro del rango).

Recibe tres parámetros de tipo TimeAttribute (atributo de tiempo u hora).

Devuelve verdadero en caso que el primer valor pasado por parámetro esté dentro del rango de los dos valores restantes.

Si no se especifica región de hora para el segundo y/o el tercer parámetro se utiliza la región de hora del primer valor. Esto nos permite utilizar la función de la siguiente manera: time-in-range(hora-actual, 9am, 5pm) y se obtiene la evaluación en la zona horaria local.

4.7.7.1.3.1.2 Parámetros

`inputs` - lista de objetos que representan los tres parámetros pasados a la función
`context` - representa el contexto de la petición

4.7.7.1.3.1.3 Devuelve

Un "EvaluationResult" (resultado de evaluación) con el resultado de aplicar la función sobre los parámetros pasados.

4.7.7.2 La clase LocationInRectangleAreaFunction

4.7.7.2.1 Descripción

Esta clase implementa la función "Location in rectangle area", es decir, devuelve verdadero en caso que una localización (en coordenadas del eje cartesiano) esté dentro de un área rectangular delimitada por otras dos coordenadas (representan la esquina izquierda inferior y la esquina derecha superior del rectángulo).

4.7.7.2.2 Atributos

`NAME:String` – identificador XACML de la función.

`sysLog:Logger` – Instancia para registrar eventos del sistema

4.7.7.2.3 Funciones destacadas

4.7.7.2.3.1 La función LocationInRectangleAreaFunction.evaluate()

4.7.7.2.3.1.1 Descripción

Evalúa la función "Location In Rectangle Area" (localización dentro del área de un rectángulo).

Recibe tres parámetros de tipo StringAtributte (atributo de tipo cadena de caracteres).

Devuelve verdadero en caso que el primer valor pasado por parámetro esté dentro del área delimitada por el segundo y el tercer parámetro, que a su vez representan la coordenada de la esquina inferior izquierda y la superior derecha respectivamente.

4.7.7.2.3.1.2 Recibe

`inputs` - lista de objetos que representan los tres parámetros pasados a la función
`context` - representa el contexto de la petición

4.7.7.2.3.1.3 Devuelve

Un "EvaluationResult" (resultado de evaluación) con el resultado de aplicar la función sobre los parámetros pasados.

4.7.7.3 La clase FunctionLoader

4.7.7.3.1 Descripción

Esta clase implementa una utilidad que nos permite cargar en memoria las distintas funciones, del estándar XACML y las implementadas por nosotros, que van a contener las políticas definidas en el sistema.

4.7.7.3.2 Atributos

`functionArray:Object` – Objeto que contiene un array a los distintos tipos de funciones que vamos a cargar.

`sysLog:Logger` – Instancia para registrar eventos del sistema

4.7.7.3.3 Funciones destacadas

4.7.7.3.3.1 La función FunctionLoader.loadConditionFunctions()

4.7.7.3.3.1.1 Descripción

Esta función nos va a permitir cargar en memoria las funciones para las condiciones.

4.7.7.3.3.1.2 Devuelve

Un "Function Factory" (factoría de funciones) con las funciones que vamos a cargar.

4.7.7.3.3.2 La función FunctionLoader.loadApplyFunctions()

4.7.7.3.3.2.1 Descripción

Esta función nos va a permitir cargar las funciones para los aplicativos.

4.7.7.3.3.2.2 Devuelve

Un "Function Factory" (factoría de funciones) con las funciones que vamos a cargar.

4.7.8 PAQUETE CERTILOC.BASE.XACML.FINDER

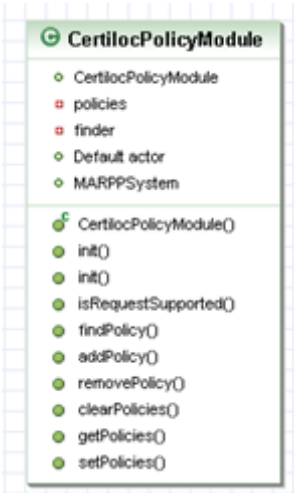


Figura 47. *El paquete Certiloc.base.xacml.finder*

4.7.8.1 La clase CertilocPolicyModule

4.7.8.1.1 Descripción

Esta clase será el módulo de políticas XACML (las que van a evaluar las peticiones de autorización) propias de CERTILOC (las activas en la base de datos de políticas de privacidad).

4.7.8.1.2 Atributos

`finder:PolicyFinder` – Objeto que nos permite buscar políticas XACML, coincidentes con las peticiones de autorización.

`policies:Set` – El conjunto de políticas activas XACML (cargadas desde la base de datos de CERTILOC y convertidas a XACML).

4.7.8.1.3 Funciones destacadas

4.7.8.1.3.1 La función CertilocPolicyModule.findPolicy()

4.7.8.1.3.1.1 Descripción

Método que nos permite obtener un resultado de evaluación de la petición pasada por parámetro contra las políticas del sistema.

Dado que la clase CertilocPolicyModule extiende **com.sun.xacml.finder.PolicyFinderModule**, esta función debe ser implementada.

4.7.8.1.3.1.2 Recibe

`context:com.sun.xacml.EvaluationCtx` – El contexto de evaluación (contiene la petición de autorización).

4.7.8.1.3.1.3 Devuelve

Un "com.sun.xacml.finder.PolicyFinderResult"; el resultado de evaluar el contexto de evaluación contra las políticas.

4.7.8.1.3.2 La función CertilocPolicyModule.init()

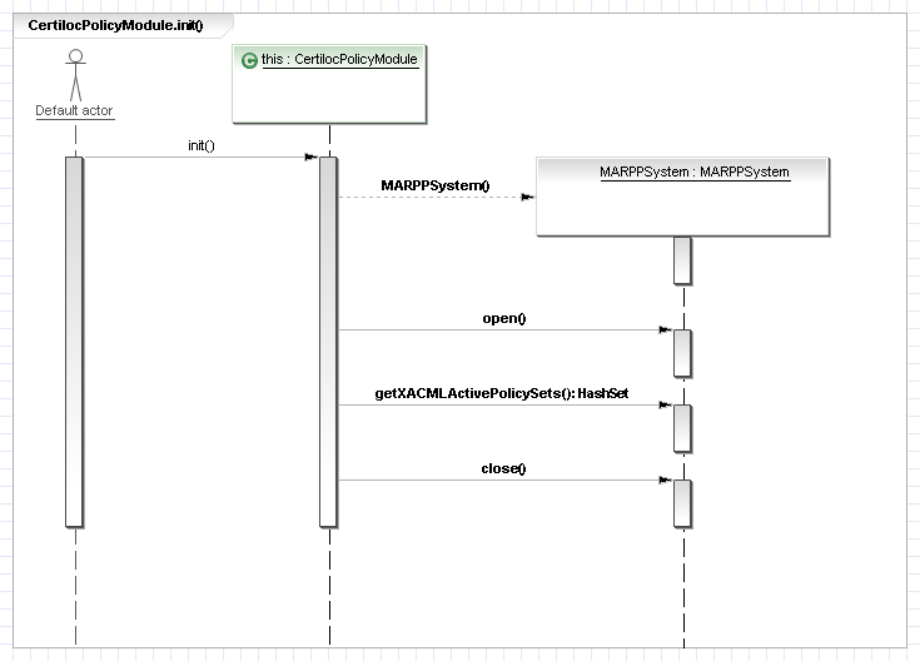


Figura 48. *Secuencia de la función CertilocPolicyModule.init()*

4.7.8.1.3.2.1 Descripción

Este método vale para inicializar el módulo de políticas. Carga todas las políticas activas en el sistema.

4.7.9 EL SISTEMA MARPP

Se presenta a continuación una especificación detallada del sistema MARPP. Recordamos al lector que MARPP era el módulo de acceso a repositorio de políticas de privacidad. Provee una interfaz unificada para acceder a los distintos datos de la base de datos.

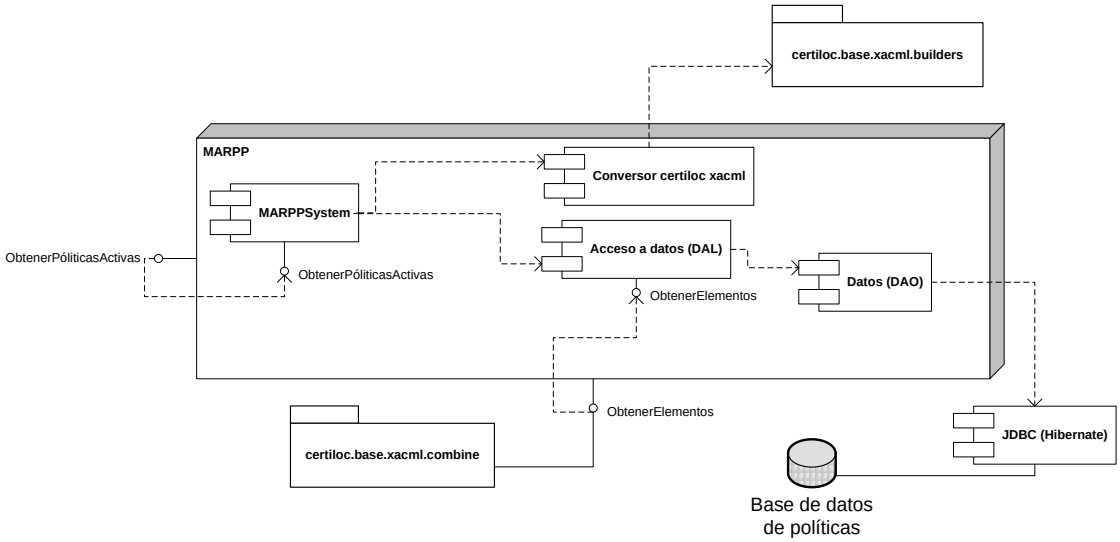


Figura 49. *Diagrama de componentes del Subsistema MARPP*

En los siguientes apartados veremos en detalle cada uno de los paquetes y clases de este módulo.

4.7.9.1 Paquete Certiloc.marpp

Figura 50. *El paquete Certiloc.marpp*

4.7.9.1.1 La clase MARPPSystem

4.7.9.1.1.1 Descripción

Clase que proporciona a CERTILOC la implementación de un módulo de acceso al repositorio de políticas de privacidad.

Esta clase es la fachada (Fecade pattern) para el uso del sistema **MARPP**.

4.7.9.1.1.2 Atributos

`converter:CertilocToXACMLConverter` – clase para convertir elementos del sistema de políticas de CERTILOC a XACML.

`dal:Certiloc.marpp.CertilocDal` – El contexto de evaluación (contiene la petición de autorización).

`sysLog:Logger` – Instancia para registrar eventos del sistema.

4.7.9.1.1.3 Funciones más destacadas

4.7.9.1.1.3.1 La función MARPPSystem.getXACMLActivePolicySets()

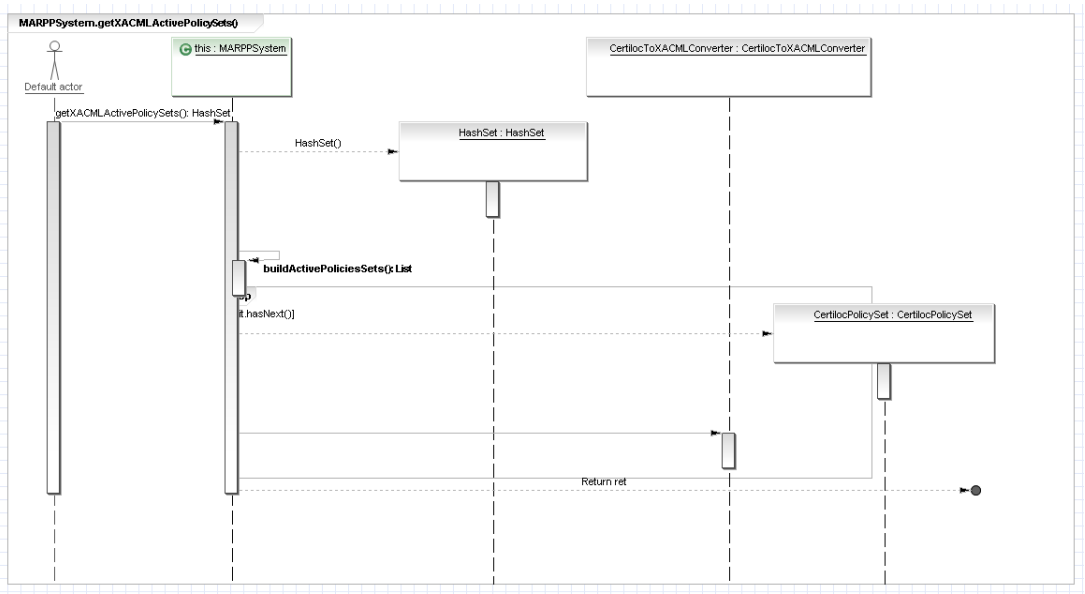


Figura 51. *La función MARPPSystem.getXACMLActivePolicySets()*

4.7.9.1.1.3.1.1 Descripción

Esta función carga todas las funciones activas, del sistema de políticas de privacidad de CERTILOC, convertidas a XACML.

Para ello, recorre todas las políticas activas del sistema y las va convirtiendo al formato XACML (ayudándose de la clase CertiloToXACMLConverter – aptdo 4.7.9.2.2) y las introduce en un conjunto para devolverlo a quien lo haya solicitado (suele solicitarlo un CertilocPDP al crearse).

4.7.9.1.1.3.1.2 Devuelve

Un conjunto Hash equivalente al conjunto de políticas activas en el sistema de CERTILOC, convertidas al formato de XACML.

4.7.9.2 Paquete Certiloc.marpp.dal

Figura 52. *El paquete Certiloc.marpp.dal*

4.7.9.2.1 La clase Certiloc.marpp.dal.CertilocDAL

4.7.9.2.1.1 Descripción

Clase que actúa como capa de acceso a datos (“Data Access Layer”) de la base de datos del sistema.

4.7.9.2.1.2 Atributos

`DISABLED_POLICY:int` – entero para definir el carácter deshabilitado.

`ENABLED_POLICY:int` – entero para definir el carácter habilitado.

`em:EntityManager` – Instancia al gestor de entidades (necesario para utilizar los métodos de acceso a datos de Hibernate).

`emf:EntityManagerFactory` – Factoría de creación de gestores de entidades.

`log:Logger` – Instancia para registrar eventos del sistema

`States:String[]` – Array con los estados habilitado y deshabilitado en formato texto

4.7.9.2.1.3 Funciones más destacadas

Debe implementar funciones para **insertar, modificar, borrar y recuperar** datos de todas las tablas que conforman el sistema de políticas de privacidad de CERTILOC (datos albergados en la base de datos).

La manera de llevar a cabo las funciones, va a ser crear instancias a todas las clases del paquete **Certiloc.marpp.dao**.

Dado que se manejan los datos mediante instancias que representan cada registro de cada tabla de la base de datos de políticas de privacidad, esta clase también debe implementar funciones para **refrescar y guardar** (hacer persitentes) las instancias que manejamos.

4.7.9.2.2 La clase Certiloc.marpp.dal.CertilocToXACMLConverter

4.7.9.2.2.1 Descripción

Esta clase nos permite convertir los distintos elementos de las políticas de privacidad de CERTILOC a su equivalente XACML.

4.7.9.2.2.2 Atributos

`sysLog:Logger` – Instancia para registrar eventos del sistema

4.7.9.2.2.3 Funciones más destacadas

Debe contener funciones para convertir todas las instancias de las clases del modelo de políticas de CERTILOC a su representación en XACML.

4.7.9.2.2.3.1 La función CertilocToXACMLConverter.CertilocPolicySetToXACMLPolicySet

4.7.9.2.2.3.1.1 Descripción

Método para construir un conjunto de políticas XACML, a partir de un conjunto de políticas de CERTILOC.

4.7.9.2.2.3.1.2 Parámetros

`CertilocPolicySet` – Un conjunto de políticas del sistema de políticas de privacidad de CERTILOC.

4.7.9.2.2.3.1.3 Devuelve

Un “com.sun.xacml.PolicySet”; conjunto de políticas en formato XACML.

4.7.9.2.2.3.2 La función CertilocToXACMLConverter.CertilocPolicyToXACMLPolicy

4.7.9.2.2.3.2.1 Descripción

Método para construir una política XACML, a partir de una política CERTILOC.

4.7.9.2.2.3.2.2 Parámetros

`CertilocPolicy` – Una política del sistema de políticas de privacidad de CERTILOC.

4.7.9.2.2.3.2.3 Devuelve

Un “com.sun.xacml.Policy”; política en formato XACML.

4.7.9.2.2.3.3 La función CertilocToXACMLConverter.CertilocObligationToXACMLObligation

4.7.9.2.2.3.3.1 Descripción

Método para construir una obligación XACML, a partir de una obligación CERTILOC.

4.7.9.2.2.3.3.2 Parámetros

`CertilocObligation` – Una obligación del sistema de políticas de privacidad de CERTILOC.

4.7.9.2.2.3.3.3 Devuelve

Un “com.sun.xacml.Obligation”; obligación en formato XACML.

4.7.9.2.2.3.4 La función CertilocAttributeAssignmentToXACMLAttributeAssignment

4.7.9.2.2.3.4.1 Descripción

Método para construir una asignación de atributo XACML, a partir de una asignación de atributo CERTILOC.

4.7.9.2.2.3.4.2 Parámetros

`caa` – Una asignación de atributo del sistema de políticas de privacidad de CERTILOC.

4.7.9.2.2.3.4.3 Devuelve

Un "com.sun.xacml.ctx.AttributeAssignment"; Asignador de atributo en formato XACML.

4.7.9.2.2.3.5 La función CertilocRuleToXACMLRule

4.7.9.2.2.3.5.1 Descripción

Método para construir una regla XACML, a partir de una regla CERTILOC.

4.7.9.2.2.3.5.2 Parámetros

`CertilocRule` – Una regla del sistema de políticas de privacidad de CERTILOC.

4.7.9.2.2.3.5.3 Devuelve

Un "com.sun.xacml.Rule"; Regla en formato XACML.

4.7.9.2.2.3.6 La función CertilocConditionToXACMLCondition

4.7.9.2.2.3.6.1 Descripción

Método para construir una condición XACML, a partir de una condición CERTILOC.

4.7.9.2.2.3.6.2 Parámetros

`CertilocCondition` – Una condición del sistema de políticas de privacidad de CERTILOC.

4.7.9.2.2.3.6.3 Devuelve

Un "com.sun.xacml.cond.Apply"; Condición en formato XACML.

4.7.9.2.2.3.7 La función CertilocApplyToXACMLApply

4.7.9.2.2.3.7.1 Descripción

Método para construir un aplicativo XACML, a partir de un aplicativo CERTILOC.

4.7.9.2.2.3.7.2 Parámetros

`CertilocApply` – Un aplicativo del sistema de políticas de privacidad de CERTILOC.

4.7.9.2.2.3.7.3 Devuelve

Un “com.sun.xacml.cond.Apply”; Aplicativo en formato XACML.

4.7.9.2.2.3.8 La función CertilocAttributeValueToXACMLAttributeValue

4.7.9.2.2.3.8.1 Descripción

Método para construir un valor de atributo XACML, a partir de un valor de atributo CERTILOC.

4.7.9.2.2.3.8.2 Parámetros

`CertilocAttributeValue` – Un valor de atributo del sistema de políticas de privacidad de CERTILOC.

4.7.9.2.2.3.8.3 Devuelve

Un “com.sun.xacml.attr.AttributeValue”; Valor de atributo en formato XACML.

4.7.9.2.2.3.9 La función CertilocAttributeDesignatorToXACMLAttributeDesignator

4.7.9.2.2.3.9.1 Descripción

Método para construir un denominador de atributo XACML, a partir de un denominador de atributo CERTILOC.

4.7.9.2.2.3.9.2 Parámetros

`CertilocAttributeDesignator` – Un denominador de atributo del sistema de políticas de privacidad de CERTILOC.

4.7.9.2.2.3.9.3 Devuelve

Un “com.sun.xacml.attr.AttributeDesignato”; Denominador de atributo en formato XACML.

4.7.9.3 Paquete Certiloc.marpp.dao

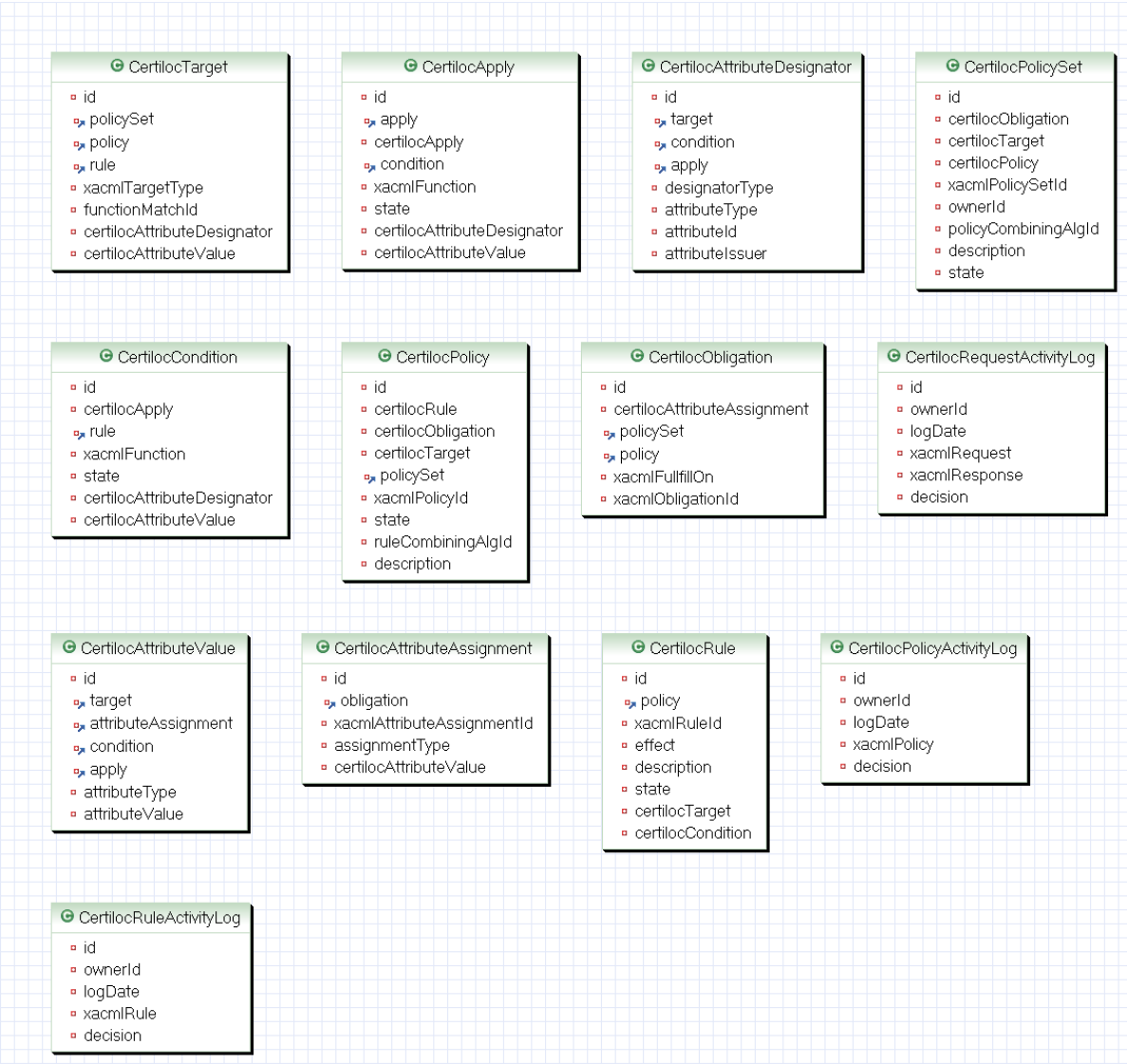


Figura 53. *El paquete Certiloc.marpp.dao*

4.7.9.3.1 La clase Certiloc.marpp.dao.CertilocPolicySet

4.7.9.3.1.1 Descripción

Esta clase representa un conjunto de políticas personalizado para el modelo de CERTILOC.

Es un objeto de acceso a datos, es decir, la base de datos tendrá una tabla que coincida con el nombre de la clase. Cada fila de la tabla representará la instancia de un objeto concreto. Para acceder y modificar los datos de la instancia en la base de datos, lo haremos directamente desde la instancia al objeto.

4.7.9.3.1.2 Atributos

`id:int` – Identificador único numérico

`CertilocObligation:collection` – Colección de obligaciones del conjunto de políticas

`CertilocTarget:collection` – Colección de objetivos del conjunto de políticas

`CertilocPolicy:collection` – Colección de políticas del conjunto de políticas

`xacmlPolicySetId:string` – Cadena con el identificador del conjunto de políticas en formato XACML (al igual que otros identificadores XACML, debe tener la morfología de una URI)

`ownerId:string` – Identificador del propietario del conjunto de políticas (un usuario de Certiloc)

`policyCombiningAlgId:string` – Identificador del algoritmo de combinación de políticas que utiliza para combinar las distintas políticas que contiene

`description:string` – Descripción corta

`state:int` – Estado, habilitado o deshabilitado

4.7.9.3.2 La clase Certiloc.marpp.dao.CertilocPolicy

4.7.9.3.2.1 Descripción

Esta clase representa una política personalizada para el modelo de CERTILOC.

Es un objeto de acceso a datos, es decir, la base de datos tendrá una tabla que coincida con el nombre de la clase. Cada fila de la tabla representará la instancia de un objeto concreto. Para acceder y modificar los datos de la instancia en la base de datos, lo haremos directamente desde la instancia al objeto.

4.7.9.3.2.2 Atributos

`id:int` – Identificador único numérico

`CertilocObligation:collection` – Colección de obligaciones la política

`policySet:CertilocPolicySet` – instancia del conjunto de políticas al que pertenece

`CertilocTarget:collection` – Colección de objetivos de la política

`CertilocRule:collection` – Colección de reglas de la política

`xacmlPolicyId:string` – Cadena con el identificador de la política en formato XACML (al igual que otros identificadores XACML, debe tener la morfología de una URI)

`ruleCombiningAlgId:string` – Identificador del algoritmo de combinación de reglas que utiliza para combinar las distintas reglas que contiene

`description:string` – Descripción corta

`state:int` – Estado, habilitado o deshabilitado

4.7.9.3.3 La clase Certiloc.marpp.dao.CertilocRule

4.7.9.3.3.1 Descripción

Esta clase representa una regla personalizada para el modelo de CERTILOC.

Es un objeto de acceso a datos, es decir, la base de datos tendrá una tabla que coincida con el nombre de la clase. Cada fila de la tabla representará la instancia de un objeto concreto. Para acceder y modificar los datos de la instancia en la base de datos, lo haremos directamente desde la instancia al objeto.

4.7.9.3.3.2 Atributos

`id:int` – Identificador único numérico

`policy:CertilocPolicy` – instancia de la política a la que pertenece

`CertilocCondition:collection` – Colección de condiciones de la regla. Siguiendo el estándar que propone XACML, sólo una de estas condiciones debería estar habilitada para cada regla (una regla XACML sólo puede tener una condición). En el caso de CERTILOC, al poder habilitar y deshabilitar elementos de una política de seguridad, una regla podría contener varias condiciones, todas deshabilitadas o con una habilitada.

`CertilocTarget:collection` – Colección de objetivos de la regla

`xacmlRuleId:string` – Cadena con el identificador de la regla en formato XACML (al igual que otros identificadores XACML, debe tener la morfología de una URI)

`description:string` – Descripción corta

`state:int` – Estado, habilitado o deshabilitado

`effect:int` – Efecto de la regla (“Permit” o “Deny”, permitir o denegar respectivamente)

4.7.9.3.4 La clase Certiloc.marpp.dao.CertilocCondition

4.7.9.3.4.1 Descripción

Esta clase representa una condición personalizada para el modelo de CERTILOC.

Es un objeto de acceso a datos, es decir, la base de datos tendrá una tabla que coincida con el nombre de la clase. Cada fila de la tabla representará la instancia de un objeto concreto. Para acceder y modificar los datos de la instancia en la base de datos, lo haremos directamente desde la instancia al objeto.

4.7.9.3.4.2 Atributos

`id:int` – Identificador único numérico

`rule:CertilocRule` – instancia de la regla a la que pertenece

`CertilocApply:collection` – Colección de aplicativos de la condición

`CertilocAttributeDesignator:collection` – Colección de los especificadores de atributo de la condición

`CertilocAttributeValue:collection` – Colección de los valores de atributo de la condición.

`CertilocApply:collection` – Colección de aplicativos de la condición.

`xacmlFunction:string` – Cadena con el identificador de la función que se lleva a cabo en la condición. Debe estar tener la morfología de una URI (The Internet Society 1998), según la especificación de XACML.

`state:int` – Estado: habilitado o deshabilitado.

4.7.9.3.5 La clase Certiloc.marpp.dao.CertilocApply

4.7.9.3.5.1 Descripción

Esta clase representa un aplicativo personalizado para el modelo de CERTILOC.

Es un objeto de acceso a datos, es decir, la base de datos tendrá una tabla que coincida con el nombre de la clase. Cada fila de la tabla representará la instancia de un objeto concreto.

Para acceder y modificar los datos de la instancia en la base de datos, lo haremos directamente desde la instancia al objeto.

4.7.9.3.5.2 Atributos

`id:int` – Identificador único numérico

`condition:CertilocCondition` – instancia de la condición al que pertenece el aplicativo (en caso de no pertenecer a otro aplicativo)

`apply:CertilocApply` – instancia del aplicativo al que pertenece este aplicativo (en caso de no pertenecer a una condición)

`CertilocApply:collection` – Colección de aplicativos de la condición

`CertilocAttributeDesignator:collection` – Colección de los especificadores de atributo de este aplicativo

`CertilocAttributeValue:collection` – Colección de los valores de atributo de este aplicativo

`CertilocApply:collection` – Colección de aplicativos de éste aplicativo

`xacmlFunction:string` – Cadena con el identificador de la función que se lleva a cabo en el aplicativo. Debe estar tener la morfología de una URI (The Internet Society 1998) según la especificación de XACML

`state:int` – Estado, habilitado o deshabilitado

4.7.9.3.6 La clase Certiloc.marpp.dao.CertilocTarget

4.7.9.3.6.1 Descripción

Esta clase representa un objetivo para una regla, una política o un conjunto de políticas.

Es un objeto de acceso a datos, es decir, la base de datos tendrá una tabla que coincida con el nombre de la clase. Cada fila de la tabla representará la instancia de un objeto concreto. Para acceder y modificar los datos de la instancia en la base de datos, lo haremos directamente desde la instancia al objeto.

4.7.9.3.6.2 Atributos

`id:int` – Identificador único numérico

`policySet:CertilocPolicySet` – instancia del conjunto de políticas al que pertenece el objetivo (en caso de no pertenecer a una regla o una política)

`policy:CertilocPolicy` – instancia de la política a la que pertenece el objetivo (en caso de no pertenecer a una regla o un conjunto de políticas)

`rule:CertilocRule` – instancia de la regla a la que pertenece el objetivo (en caso de no pertenecer a una política o un conjunto de políticas)

`xacmlTargetType:int` – Entero que designa el tipo de objetivo (de recurso, de sujeto o de acción)

`functionMatchId:string` – Cadena que indica la función que se ejecuta para evaluar si existen coincidencias entre el objetivo en cuestión y los atributos presentados en una petición (si se da la coincidencia, indica que la regla, política o conjuntos de políticas a la que pertenece el objetivo se puede aplicar contra la petición recibida)

`CertilocAttributeDesignator:collection` – Colección de los especificadores de atributo de este objetivo

`CertilocAttributeValue:collection` – Colección de los valores de atributo de este objetivo

4.7.9.3.6.3 Funciones más destacadas

Debe contener funciones para acceder y modificar todos los atributos y varios métodos constructores.

4.7.9.3.7 La clase Certiloc.marpp.dao.CertilocAttributeValue

4.7.9.3.7.1 Descripción

Esta clase representa un valor de atributo.

Es un objeto de acceso a datos, es decir, la base de datos tendrá una tabla que coincida con el nombre de la clase. Cada fila de la tabla representará la instancia de un objeto concreto. Para acceder y modificar los datos de la instancia en la base de datos, lo haremos directamente desde la instancia al objeto.

4.7.9.3.7.2 Atributos

`id:int` – Identificador único numérico

`target:CertilocTarget` – instancia del conjunto de políticas al que pertenece el objetivo (en caso de no pertenecer a una regla o una política)

`attributeAssignment:CertilocAttributeAssignment` – instancia de la asignación de atributo a la que pertenece el valor (en caso de no pertenecer a un objetivo, a una condición o a un aplicativo)

`target:CertilocTarget` – instancia del objetivo al que pertenece el valor (en caso de no pertenecer a una asignación de atributo, a una condición o a un aplicativo)

`condition:CertilocCondition` – instancia de la condición a la que pertenece el valor (en caso de no pertenecer a un objetivo, a una asignación de atributo o a un aplicativo)

`apply:CertilocApply` – instancia del aplicativo al que pertenece el valor (en caso de no pertenecer a una asignación de atributo, a una condición o a un objetivo)

`attributeType:string` – tipo del valor que contiene este valor de atributo

`attributeValue:string` – valor que contiene este valor de atributo

4.7.9.3.8 La clase Certiloc.marpp.dao.CertilocObligation

4.7.9.3.8.1 Descripción

Esta clase representa una obligación para una política o un conjunto de políticas.

Es un objeto de acceso a datos, es decir, la base de datos tendrá una tabla que coincida con el nombre de la clase. Cada fila de la tabla representará la instancia de un objeto concreto. Para acceder y modificar los datos de la instancia en la base de datos, lo haremos directamente desde la instancia al objeto.

4.7.9.3.8.2 Atributos

`id:int` – Identificador único numérico

`policySet:CertilocPolicySet` – instancia del conjunto de políticas al que pertenece la obligación (en caso de no pertenecer a una política)

`policy:CertilocPolicy` – instancia de la política a la que pertenece la obligación (en caso de no pertenecer a un conjunto de políticas)

`xacmlFullfillOn:int` – Entero que representa cuándo se debe incluir la obligación en la respuesta (puede ser cuando se evalúe a permitir o “Permit” o cuando se evalúe a denegar o “Deny”)

`xacmlObligationId:string` – Cadena que indica el identificador XACML de la obligación

4.7.9.3.9 La clase Certiloc.marpp.dao.CertilocAttributeAssignment

4.7.9.3.9.1 Descripción

Esta clase representa una asignación de atributo para una política o un conjunto de políticas.

Es un objeto de acceso a datos, es decir, la base de datos tendrá una tabla que coincida con el nombre de la clase. Cada fila de la tabla representará la instancia de un objeto concreto. Para acceder y modificar los datos de la instancia en la base de datos, lo haremos directamente desde la instancia al objeto.

4.7.9.3.9.2 Atributos

`id:int` – Identificador único numérico

`obligation:CertilocObligation` – instancia de la obligación a la que pertenece la asignación de atributo

`xacmlAttributeAssignmentId:string` – Identificador XACML de la asignación de atributo

`assignmentType:string` – Cadena que representa el tipo de asignación de atributo

`CertilocAttributeValue:collection` – Colección de valores de la asignación de atributo

4.7.9.3.10 La clase Certiloc.marpp.dao.CertilocAttributeDesignator

4.7.9.3.10.1 Descripción

Esta clase representa una especificación de atributo para un objetivo, una condición o un aplicativo.

Es un objeto de acceso a datos, es decir, la base de datos tendrá una tabla que coincida con el nombre de la clase. Cada fila de la tabla representará la instancia de un objeto concreto. Para acceder y modificar los datos de la instancia en la base de datos, lo haremos directamente desde la instancia al objeto.

4.7.9.3.10.2 Atributos

`id:int` – Identificador único numérico

`target:CertilocTarget` – instancia del objetivo al que pertenece la especificación de atributo (en caso de no pertenecer a una condición o un aplicativo)

`condition:CertilocCondition` – instancia de la condición a la que pertenece la especificación de atributo (en caso de no pertenecer a un objetivo o un aplicativo)

`apply:CertilocApply` – instancia del aplicativo al que pertenece la especificación de atributo (en caso de no pertenecer a un objetivo o una condición)

`designatorType:int` – entero que representa el tipo de especificación de atributo (de sujeto, de acción, de recurso o de entorno)

`attributeType:string` – tipo del valor de atributo que especifica el especificador de atributo

`attributeId:string` – Identificador XACMl del atributo

`attributeIssuer:string` – Emisor del atributo (en caso de no ser un atributo contemplado en el estándar XACML, se debe indicar quién es su emisor)

4.7.9.3.11 La clase Certiloc.marpp.dao.CertilocAttributeValue

4.7.9.3.11.1 Descripción

Esta clase representa un valor de atributo para un objetivo, una condición, un aplicativo o una asignación de atributo.

Es un objeto de acceso a datos, es decir, la base de datos tendrá una tabla que coincida con el nombre de la clase. Cada fila de la tabla representará la instancia de un objeto concreto. Para acceder y modificar los datos de la instancia en la base de datos, lo haremos directamente desde la instancia al objeto.

4.7.9.3.11.2 Atributos

`id:int` – Identificador único numérico

`target:CertilocTarget` – instancia del objetivo al que pertenece el valor del atributo (en caso de no pertenecer a una condición, un aplicativo o a una asignación de atributo)

`condition:CertilocCondition` – instancia de la condición a la que pertenece la especificación de atributo (en caso de no pertenecer a un objetivo, un aplicativo o a una asignación de atributo)

`apply:CertilocApply` – instancia del aplicativo al que pertenece el valor del atributo (en caso de no pertenecer a un objetivo, una condición o a una asignación de atributo)

`attributeAssignment:CertilocAttributeAssignment` – instancia de la asignación de atributo a la que pertenece el valor del atributo (en caso de no pertenecer a un objetivo, una condición o a un aplicativo)

`attributeType:string` – tipo del valor de atributo que especifica el especificador de atributo

`attributeValue:string` – valor de atributo

4.7.9.3.12 La clase Certiloc.marpp.dao.RequestActivityLog

4.7.9.3.12.1 Descripción

Esta clase representa un registro con la información de una petición de autorización.

Es un objeto de acceso a datos, es decir, la base de datos tendrá una tabla que coincida con el nombre de la clase. Cada fila de la tabla representará la instancia de un objeto concreto. Para acceder y modificar los datos de la instancia en la base de datos, lo haremos directamente desde la instancia al objeto.

4.7.9.3.12.2 Atributos

`id:int` – Identificador único numérico

`ownerId:String` –nombre de usuario del usuario al que pertenece el dispositivo o recurso objetivo de la petición recibida

`logDate:Date` – Fecha y hora de recepción de la petición de autorización que se registra

`xacmlRequest:String` – Petición recibida en formato XACML

`xacmlResponse:String` – Respuesta devuelta a la petición de autorización, en formato XACML

`decision:Integer` – Entero que designa la decisión ante la petición de autorización registrada

4.7.9.3.13 La clase Certiloc.marpp.dao.PolicyActivityLog

4.7.9.3.13.1 Descripción

Esta clase representa un registro con información de una actividad en las políticas del sistema. Consideramos que hay actividad en las políticas del sistema cuando recibimos una petición de autorización que genera una respuesta. En esta primera implementación del sistema de políticas, sólo se ha creado un algoritmo de combinación de políticas donde prevalece la permisividad con lo que sólo se crea un registro en caso que la respuesta devuelta sea **permitir**. El registro guarda la información de la política responsable de dar una respuesta permisiva ante una petición de autorización.

Es un objeto de acceso a datos, es decir, la base de datos tendrá una tabla que coincida con el nombre de la clase. Cada fila de la tabla representará la instancia de un objeto concreto. Para acceder y modificar los datos de la instancia en la base de datos, lo haremos directamente desde la instancia al objeto.

4.7.9.3.13.2 Atributos

`id:int` – Identificador único numérico

`ownerId:String` –nombre de usuario del usuario al que pertenece el dispositivo o recurso objetivo de la petición que ha generado la actividad

`logDate:Date` – Fecha y hora de recepción de la petición de autorización que se registra

`xacmlPolicy:String` – Política responsable de la respuesta devuelta, en formato XACML

`decision:Integer` – Entero que designa la decisión devuelta por la política registrada

4.7.9.3.14 La clase Certiloc.marpp.dao.RuleActivityLog

4.7.9.3.14.1 Descripción

Esta clase representa un registro con información de una actividad en las reglas del sistema. Consideramos que hay actividad en las reglas del sistema cuando recibimos una

petición de autorización que genera una respuesta. En esta primera implementación del sistema de políticas, sólo se ha creado un algoritmo de combinación de reglas, donde prevalece la permisividad con lo que sólo se crea un registro en caso que la respuesta devuelta sea **permitir**. El registro guarda la información de la regla responsable de dar una respuesta permisiva ante una petición de autorización.

Es un objeto de acceso a datos, es decir, la base de datos tendrá una tabla que coincida con el nombre de la clase. Cada fila de la tabla representará la instancia de un objeto concreto. Para acceder y modificar los datos de la instancia en la base de datos, lo haremos directamente desde la instancia al objeto.

4.7.9.3.14.2 Atributos

`id:int` – Identificador único numérico

`ownerId:String` –nombre de usuario del usuario al que pertenece el dispositivo o recurso objetivo de la petición que ha generado la actividad

`logDate:Date` – Fecha y hora de recepción de la petición de autorización que se registra

`xacmlRule:String` – Regla, en formato XACML, responsable de la respuesta devuelta

`decision:Integer` – Entero que designa la decisión devuelta por la regla registrada

4.7.10 EL SISTEMA SGP

Se presenta a continuación un diagrama conceptual del sistema SGP. Recordamos al lector que éste sistema era el encargado de ofrecer, a los usuarios de CERTILOC, la capacidad de gestionar sus políticas de privacidad mediante una interfaz de usuario Web.

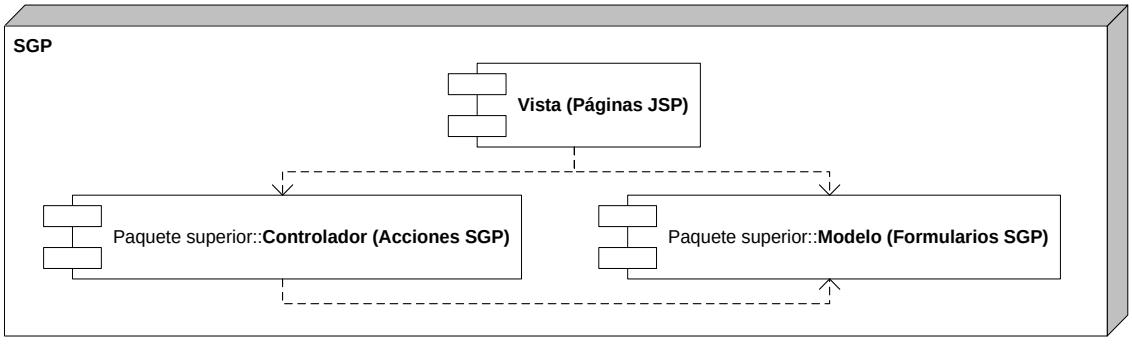


Figura 54. *El Subsistema SGP*

En los siguientes apartados se describe en detalle este sistema.

4.7.10.1 Paquete Certiloc.sgp

Figura 55. *El paquete Certiloc.sgp*

Este paquete contiene las clases necesarias para implementar el "Servicio de Gestión de la Privacidad" o **SGP**. El **SGP**, es el encargado de proporcionar una interfaz de usuario web a los distintos usuarios de CERTILOC para la gestión de las políticas de privacidad relacionadas con sus dispositivos.

Este paquete ha sido programado utilizando la lógica que propone "Apache STRUTS" con lo que se ha omitido la definición detallada de cada clase y solamente se describe la función que cumple. Recordemos que para desarrollar aplicaciones siguiendo la especificación de STRUTS, debemos generar acciones, formularios y vistas que conformarán el conjunto de la aplicación web. Las acciones definen los controladores que se disparan ante cada evento del

sistema; Los formularios definen las entradas de datos del sistema; La vista proporciona una interfaz de usuario para la introducción de datos en los formularios, para disparar ciertos eventos y para mostrar el comportamiento general de la aplicación. De aquí deducimos que estamos utilizando el patrón **MVC – Modelo Vista Controlador**.

Las clases de este paquete que no se utilizan directamente para definir el funcionamiento de la aplicación siguiendo STRUTS, son clases de apoyo que nos ayudarán a crear controles para la vista de la aplicación y para mantener el código fuente mejor ordenada y escalable.

4.7.10.1.1 La clase Certiloc.sgp.CertilocPoliciesMenu

Esta clase implementa un control para navegar visualmente través de un árbol de políticas de privacidad de CERTILOC.

Nos va a permitir crear un menú desplegable, a partir de las políticas de privacidad pertenecientes a un usuario para que éste pueda navegar a través de ellas. El usuario podrá elegir y desplegar los nodos de sus políticas de privacidad desde la raíz hasta sus hojas.

Esta clase va a simplificar notablemente la manera de consultar las políticas de privacidad.

De ahora en adelante, nos referiremos a “los elementos” como los nodos del árbol de políticas de privacidad que se muestran al usuario.

4.7.10.2 Paquete Certiloc.sgp.actions

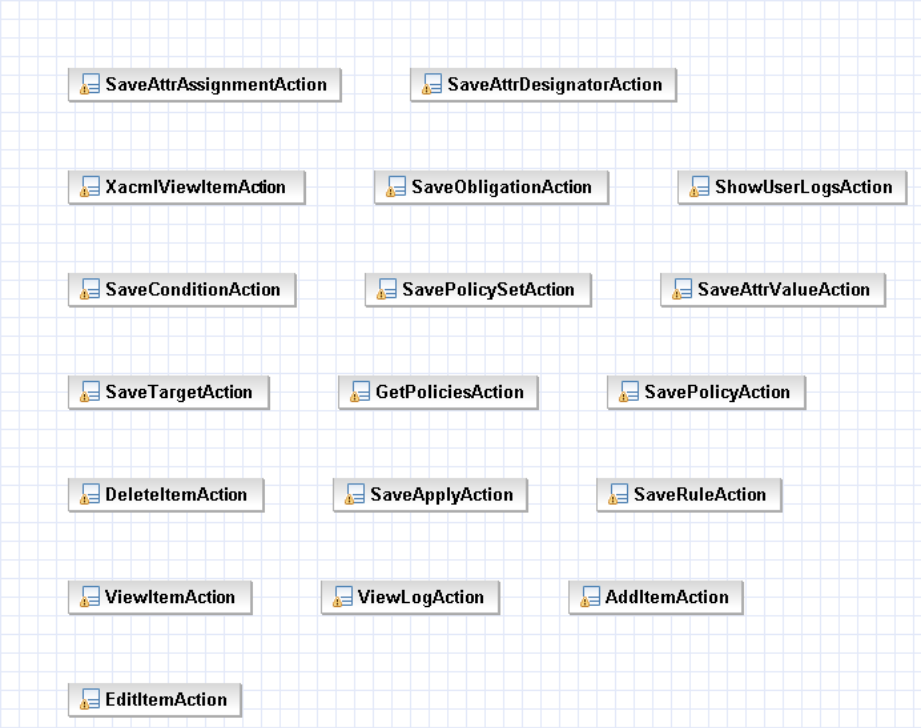


Figura 56. *El paquete Certiloc.sgp.actions*

Este paquete contiene las clases necesarias para controlar las distintas acciones de la interfaz de usuario web.

Recordemos que estamos siguiendo la especificación propuesta por STRUTS con lo que cada acción a realizar en la aplicación debe tener una implementación distinta.

4.7.10.2.1 La clase Certiloc.sgp.actions.GetPoliciesAction

Obtiene las todas las políticas de privacidad pertenecientes a un usuario. Para ello accede a la base de datos mediante el sistema MARPP.

4.7.10.2.2 La clase Certiloc.sgp.actions.ViewItemAction

Cuando un usuario pincha sobre un elemento del árbol de políticas de privacidad mostrado en la interfaz web, pasa a ver los detalles de ese nodo. Se crea una variable de sesión con la información del elemento seleccionado. Esta clase permite centrar la atención en alguno de los nodos del menú de políticas de privacidad presentadas al usuario en la interfaz web. De esta manera, el nodo se podrá editar, ver en formato XACML, borrar, etc. Además nos permitirá añadir nuevos elementos hijos al nodo seleccionado.

4.7.10.2.3 La clase Certiloc.sgp.actions.EditItemAction

Esta clase permite editar los detalles de un nodo del árbol de políticas de privacidad presentadas al usuario.

4.7.10.2.4 La clase Certiloc.sgp.actions.AddItemAction

Esta clase permite añadir un nuevo nodo hijo a alguno de los nodos del árbol de políticas de privacidad presentadas al usuario en la interfaz web.

4.7.10.2.5 La clase Certiloc.sgp.actions.DeleteItemAction

Esta clase permite eliminar un nodo del árbol de políticas de privacidad presentadas al usuario en la interfaz web.

4.7.10.2.6 La clase Certiloc.sgp.actions.XacmlViewItemAction

Esta clase permite ver, en formato XML, un nodo del árbol de políticas de privacidad presentadas al usuario en la interfaz web.

4.7.10.2.7 La clase Certiloc.sgp.actions.SavePolicySetAction

Esta clase permite guardar posibles modificaciones realizadas a través de la interfaz web, sobre un determinado conjunto de políticas.

4.7.10.2.8 La clase Certiloc.sgp.actions.SavePolicyAction

Esta clase permite guardar posibles modificaciones realizadas a través de la interfaz web, sobre una determinada política.

4.7.10.2.9 La clase Certiloc.sgp.actions.SaveRuleAction

Esta clase permite guardar posibles modificaciones realizadas a través de la interfaz web, sobre una determinada regla.

4.7.10.2.10 La clase Certiloc.sgp.actions.SaveTargetAction

Esta clase permite guardar posibles modificaciones realizadas a través de la interfaz web, sobre un determinado objetivo.

4.7.10.2.11 La clase Certiloc.sgp.actions.SaveConditionAction

Esta clase permite guardar posibles modificaciones realizadas a través de la interfaz web, sobre una determinada condición.

4.7.10.2.12 La clase Certiloc.sgp.actions.SaveApplyAction

Esta clase permite guardar posibles modificaciones realizadas a través de la interfaz web, sobre un determinado aplicativo.

4.7.10.2.13 La clase Certiloc.sgp.actions.SaveAttValueAction

Esta clase permite guardar posibles modificaciones realizadas a través de la interfaz web, sobre un determinado valor de atributo.

4.7.10.2.14 La clase Certiloc.sgp.actions.SaveObligationAction

Esta clase permite guardar posibles modificaciones realizadas a través de la interfaz web, sobre una determinada obligación.

4.7.10.2.15 La clase Certiloc.sgp.actions.SaveAttrAssignmentAction

Esta clase permite guardar posibles modificaciones realizadas a través de la interfaz web, sobre una determinada asignación de atributo.

4.7.10.2.16 La clase Certiloc.sgp.actions.SaveAttrDesignatorAction

Esta clase permite guardar posibles modificaciones realizadas a través de la interfaz web, sobre un determinado especificador de atributo.

4.7.10.2.17 La clase Certiloc.sgp.actions.ShowUserLogsAction

Esta clase permite obtener todos los registros de actividad asociados a un determinado usuario para mostrarlos por la interfaz web.

4.7.10.2.18 La clase Certiloc.sgp.actions.ViewLogAction

Esta clase permite obtener un determinado registro de actividad de la base de datos para mostrarlo por la interfaz web.

4.7.10.3 Paquete Certiloc.sgp.FORMS

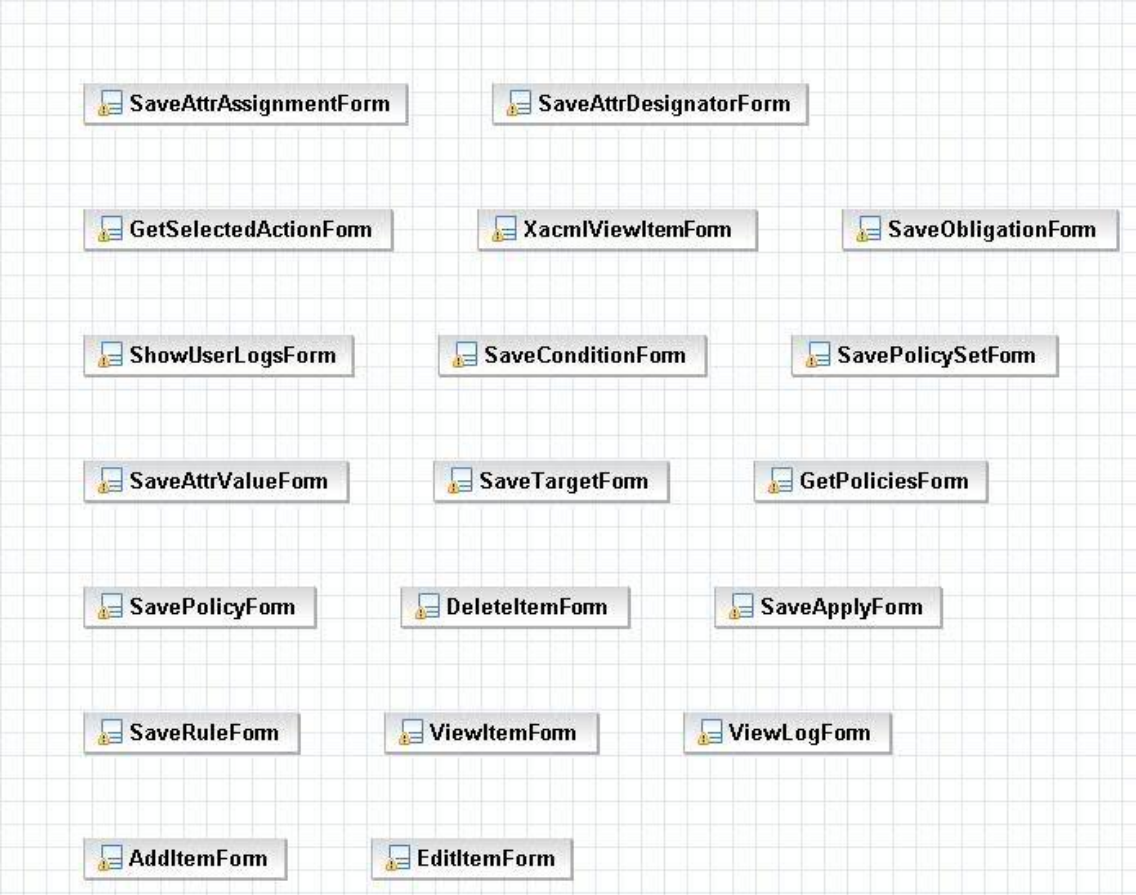


Figura 57. *El paquete Certiloc.sgp.forms*

Este paquete contiene las clases que implementan los distintos formularios para el intercambio de información entre el usuario y el sistema a través de la interfaz web.

Cada formulario está asociado a una acción de las presentadas en la clase anterior.

4.7.10.3.1 La clase Certiloc.sgp.forms.GetPoliciesForm

Pasa la información del id del usuario para el que se recuperan todas las políticas. Además contiene un listado con todas las políticas asociadas a ese usuario.

4.7.10.3.2 La clase Certiloc.sgp.forms.ViewItemForm

Contiene información sobre el id y el tipo del elemento que se quiere ver o seleccionar.

4.7.10.3.3 La clase Certiloc.sgp.forms.EditItemForm

Contiene información sobre el tipo del elemento a editar. No contiene información sobre el id del elemento que se va a editar ya que se edita el elemento seleccionado

actualmente. El elemento seleccionado se guarda en una variable de sesión cuando se ejecuta la acción ViewItemAction.

4.7.10.3.4 La clase Certiloc.sgp.forms.AddItemForm

Contiene información sobre el tipo del elemento a añadir y el origen de donde se recibe la acción de añadir. El origen de la acción está relacionado con el elemento seleccionado que el usuario está viendo actualmente. Como ya se ha comentado, el elemento seleccionado actualmente se guarda en una variable de sesión. De esta manera, sabemos a qué elemento se le está añadiendo el tipo de elemento que se va a añadir.

4.7.10.3.5 La clase Certiloc.sgp.forms.DeleteItemForm

Esta clase contiene información sobre el tipo del elemento que se va a eliminar.

4.7.10.3.6 La clase Certiloc.sgp.forms.XacmlViewItemForm

Esta clase contiene información sobre el tipo del elemento que se va a ver en formato XACML.

4.7.10.3.7 La clase Certiloc.sgp.forms.SavePolicySetForm

Contiene información sobre los atributos del conjunto de políticas que se va a guardar: el identificador XACML, una descripción, su estado (activo o inactivo), el algoritmo de combinación de políticas que utiliza y si se trata de guardar un nuevo elemento o un elemento editado.

4.7.10.3.8 La clase Certiloc.sgp.forms.SavePolicyForm

Contiene información sobre los atributos de la política que se va a guardar: el identificador XACML, una descripción, su estado (activo o inactivo), el algoritmo de combinación de reglas que utiliza y si se trata de guardar un nuevo elemento o un elemento editado.

4.7.10.3.9 La clase Certiloc.sgp.forms.SaveRuleForm

Contiene información sobre los atributos de la regla que se va a guardar: el identificador XACML, una descripción, su estado (activo o inactivo), el efecto que provoca su cumplimiento (permitir o denegar) y si se trata de guardar un nuevo elemento o un elemento editado.

4.7.10.3.10 La clase Certiloc.sgp.forms.SaveTargetForm

Contiene información sobre los atributos de la regla que se va a guardar: el tipo de función que comprueba si hay coincidencia, el tipo de objetivo (de recurso, entorno, acción o sujeto) y si se trata de guardar un nuevo elemento o un elemento editado.

4.7.10.3.11 La clase Certiloc.sgp.forms.SaveConditionForm

Contiene información sobre los atributos de la condición que se va a guardar: la función que utiliza la condición, el estado (activo o inactivo) y si se trata de guardar un nuevo elemento o un elemento editado.

4.7.10.3.12 La clase Certiloc.sgp.forms.SaveApplyForm

Contiene información sobre los atributos del aplicativo que se va a guardar: la función que utiliza la condición, el estado (activo o inactivo) y si se trata de guardar un nuevo elemento o un elemento editado.

4.7.10.3.13 La clase Certiloc.sgp.forms.SaveAttrValueForm

Contiene información sobre los atributos del valor de atributo que se va a guardar: el valor del atributo, el tipo del atributo y si se trata de guardar un nuevo elemento o un elemento editado.

4.7.10.3.14 La clase Certiloc.sgp.forms.SaveObligationForm

Contiene información sobre los atributos de la obligación que se va a guardar: el caso en el que se deba cumplimentar (al permitir o denegar), el id en formato XACML y si se trata de guardar un nuevo elemento o un elemento editado.

4.7.10.3.15 La clase Certiloc.sgp.forms.SaveAttrAssignmentForm

Contiene información sobre los atributos de la asignación de atributo que se va a guardar: el tipo de la asignación de atributo, el id en formato XACML y si se trata de guardar un nuevo elemento o un elemento editado.

4.7.10.3.16 La clase Certiloc.sgp.forms.SaveAttrDesignatorForm

Contiene información sobre los atributos del especificador de atributo que se va a guardar: el id en formato XACML, el tipo del atributo que se especifica, el emisor del especificador de atributo, el tipo de especificador y si se trata de guardar un nuevo elemento o un elemento editado.

4.7.10.3.17 La clase Certiloc.sgp.forms.ShowUserLogsForm

No contiene ninguna información. Sabemos qué registros de actividad se deben a mostrar obteniendo el id de usuario desde una variable de sesión.

4.7.10.3.18 La clase Certiloc.sgp.forms.ViewLogForm

Contiene información sobre el registro de actividad que se va a mostrar: el id del registro de actividad y el tipo del mismo.

4.7.10.4 Paquete Certiloc.sgp.constants

Figura 58. *El paquete Certiloc.sgp.constants*

Este paquete contiene varias clases que sólo contienen constantes utilizadas a lo largo de la aplicación web.

4.7.10.4.1 La clase Certiloc.sgp.forms.WebAppLocations

Constantes relacionadas con las distintas localizaciones donde se pueda encontrar un usuario al navegar por el árbol de políticas de privacidad.

4.7.10.4.2 La clase Certiloc.sgp.forms.WebAppActions

Constantes relacionadas con las acciones que pueda ejecutar un usuario al navegar por la aplicación web.

4.7.10.4.3 La clase Certiloc.sgp.forms.WebAppItems

Constantes relacionadas con los elementos que se encuentran en un árbol de políticas de privacidad.

5 IMPLEMENTACIÓN, PRUEBAS E IMPLANTACIÓN

En el presente apartado se presentan las particularidades de la implementación del sistema de políticas de privacidad diseñado en el apartado anterior.

Tal y como se ha visto en el apartado anterior, nuestro sistema consta de tres partes bien diferenciadas:

- ACP y AGPA – Conversión y Evaluación de peticiones de autorización: con la ayuda del API de SUN "Sun's XACML Implementation", convertimos las peticiones que nos llegan en formato XACML y las evalúamos contra todas las políticas activas del sistema de seguridad, previamente convertidas al contexto de XACML.
- MARPP –Lectura y escritura de datos persistentes: las políticas de privacidad de CERTILOC se almacenan en una base de datos, así como los registros de actividad de la aplicación. Hibernate nos ayuda a crear un sistema mediante el cual hacemos el acceso a los datos persistentes, con una capa intermedia para tratarlas como clases.
- SGP – Gestión de políticas de privacidad: STRUTS nos proporciona un marco de desarrollo para aplicaciones web que nos ayudará a integrar el sistema de políticas de privacidad con el servidor web de CERTILOC.

En este apartado se presentan los aspectos particulares de la implementación, el entorno de desarrollo utilizado y el proceso de integración e implantación de los módulos desarrollados con el resto de módulos de CERTILOC y con el servidor de producción.

Por último se presentan los resultados de la aplicación de las pruebas de aceptación del sistema, presentadas en el apartado 4.6 del presente documento.

5.1 ASPECTOS PARTICULARES DE LA IMPLEMENTACIÓN

A continuación se presentan los aspectos particulares de la implementación del sistema de políticas de privacidad.

El sistema a implementar implica la creación de varios subsistemas que interactúan entre sí, como vimos en apartados anteriores. Todos los subsistemas se comunican, pero a la vez son independientes y podrían ejecutarse de manera distribuida. Por lo tanto, debemos crear un sistema de eventos que ponga a todos en común y nos permita evaluar errores en tiempo de ejecución. Por otro lado, el modelo de datos a manejar es un modelo jerárquico complicado y, para poder ponerlo a prueba el sistema, necesitamos crear una batería de datos ficticios.

Para poder hacer un seguimiento de la actividad del sistema y evaluar con facilidad los errores generados por los subsistemas en tiempo de ejecución, se utiliza un sistema de eventos común. Todas las clases del sistema, contienen un miembro llamado sysLog, de tipo Logger, que escribe eventos del sistema a archivos de texto. Todas las clases, guardan los eventos del sistema en un lugar común que se encuentra en la raíz de la aplicación, pero que podría contenerse en cualquier otro lugar. Dado que todos los subsistemas utilizan un miembro Logger, toda la configuración de registros de cada sistema se guarda en el mismo archivo, "*log4j.properties*". En esta primera implementación del sistema de políticas de privacidad, todos los registros generados se guardan en la raíz de la aplicación, en la carpeta "*logs/system/*". Cada subsistema tiene sus propios ficheros de registros, que se guardan en esta carpeta con el nombre del subsistema en un archivo con extensión *.log.

Por otro lado, para la creación de una batería de datos de prueba se ha implementado un archivo de comandos por lotes que ejecuta comandos de SQL contra la base de datos, mediante la aplicación cliente de MySQL. Genera datos de políticas de privacidad donde se contemplan distintas informaciones para su aplicación: información de sujeto, de acción a realizar, y del recurso al que se desea acceder. Por otro lado, las reglas de estas políticas incluyen condiciones que permiten evaluar casos concretos donde se tiene en cuenta distintos parámetros de sujeto, de acción, de recurso y del entorno en el que se hace la solicitud. Algunas condiciones implementan funciones lógicas como el operador OR para combinar otras funciones.

5.2 ENTORNO DE DESARROLLO

Todos los componentes del sistema de políticas de privacidad se ejecutan del lado de un servidor. Es decir, no existen distintos entornos donde la aplicación pueda ser ejecutada, por lo que el desarrollo del sistema entero se ha hecho utilizando el mismo entorno de desarrollo.

Para el entorno de desarrollo, se ha elegido el IDE Java Eclipse 3.2, integrado con MySQL 5.5 y Apache Tomcat v6.0. Este entorno de desarrollo, ha permitido que el sistema de políticas de privacidad pueda ser desarrollado de una manera totalmente autónoma al entorno de producción de CERTILOC.

Para completar la funcionalidad para desarrollo de aplicaciones en STRUTS, se ha utilizado Exadel Studio como componente de Eclipse. Exadel Studio ha permitido tener un punto de vista centralizado de la aplicación web. Entre otras funcionalidades, esta herramienta

permite obtener distintas maneras de acceder y modificar el archivo de configuración de STRUTS, *struts-config.xml*.

Además, se ha utilizado el componente Omondo EclipseUML para el modelado de las distintas clases y paquetes de los sistemas que conforman el proyecto. Esta herramienta nos permite crear diagramas UML a partir de código desarrollado, así como crear código partiendo de una modelación visual mediante diagramas UML.

Por otro lado, Java Eclipse ofrece herramientas y mecanismos para exportar proyectos de manera que se puedan desplegar, en muy poco tiempo, en un entorno de producción real. Esto ha simplificado, en gran medida, la integración con el servidor de CERTILOC y la puesta en marcha del sistema completo.

5.3 IMPLANTACIÓN

Una vez finalizada la implementación del proyecto, se ha implantado en el servidor de producción de CERTILOC con el resto de módulos desarrollados.

El proceso de implantación de la aplicación ha supuesto modificar parte del código, tanto existente como desarrollado, hacer una copia de seguridad general del estado del servidor de producción antes de la implantación del sistema de políticas y crear una batería de datos de prueba para probar el sistema en producción con el resto de módulos.

La modificación de código ha supuesto cambios en el módulo SGP. Se ha modificado toda la vista de la interfaz de usuario, para coincidir con la estética, en cuanto a diseño gráfico, del resto de módulos para la aplicación Web.

Además, se han introducido nuevas referencias en los ficheros de recursos de idiomas de la aplicación y se han referenciado en el código del sistema SGP, para poder dar un soporte al cambio de idioma en tiempo de ejecución..

También se han modificado varias clases de los sistemas AGPA y ACP, por un lado, para poder recuperar la asociación de dispositivos a usuarios de CERTILOC, y por otro, para obtener el la relación de Rol entre dos usuarios de CERTILOC. Durante la implementación del sistema de políticas de privacidad, esta información no era accesible, ya que se guardaba en una base de datos perteneciente a otro módulo de CERTILOC. Dado que esta información no existía, se utilizaron datos inventados. La asociación de dispositivos a usuarios concretos, era crucial para la generación de los registros de actividad de aplicación, para poder asociar registros de actividad a usuarios concretos. La obtención de la relación de rol entre dos usuarios de

CERTILOC, era necesaria para la generación de peticiones de autorización del sistema AGPA. Para poder acceder a la lectura de estos datos, ha sido necesaria la creación de la clase *CERTILOC.base.utils.CERTILOCDBConnection*, que crea una conexión a la base de datos general de CERTILOC. Esta clase nos permite llamar a los distintos métodos de la clase *CERTILOC.aaa.impl.AAAImplementation*, que siempre solicitan como parámetro, una conexión activa a la base de datos general de CERTILOC.

Por otro lado, se ha modificado parte del código que ya existía en el sistema. Por un lado, para poder acceder al sistema SGP mediante la interfaz de usuario Web y, por otro, para que las peticiones de autorización fuesen evaluadas por las políticas de privacidad activas en el sistema. Para alcanzar estos objetivos, se ha modificado el fichero *struts-config.xml* para que, al ejecutarse la acción *GestorOpcionesClienteForm*, si se ha elegido la opción *serviciosGestionPrivacidad*, el usuario sea redirigido al inicio del sistema de gestión de políticas de privacidad – la página */jsp/policyManagerOptions.jsp*. Por otro lado, se ha modificado la clase *CERTILOC.acp.impl.ACPImplementation*, para llamar a los métodos correspondientes de *CERTILOC.agpa.AGPASystem* dependiendo del tipo de autorización que se quiera evaluar:

- `compruebaPrivilegiosServicioInmediatoIET`
- `compruebaPrivilegiosServicioInmediatoCET`
- `compruebaPrivilegiosServicioReasumido`

Antes de integrar todo el código modificado en el servidor de producción, se ha realizado una copia de seguridad completa del sistema copiando la raíz de datos del servidor a una ubicación segura.

Una vez hecha la copia de seguridad, se han copiado los nuevos archivos con las modificaciones del código, y se ha creado la base de datos de políticas de privacidad.

Por último se han creado nuevas políticas de privacidad reales para interactuar con el resto de datos del sistema. Esto nos ha permitido poner a prueba la integración del sistema de políticas de privacidad con el resto de CERTILOC.

5.4 RESULTADOS DE PRUEBAS DE ACEPTACIÓN DEL SISTEMA

Se presentan a continuación los resultados obtenidos al aplicar las pruebas de aceptación del sistema sobre el entorno de la aplicación una vez desplegada en el servidor de producción.

Identificador de la prueba	Resultado
CERTILOC-**PP-PA-001**	Prueba superada
CERTILOC-**PP-PA-002**	Prueba superada
CERTILOC-**PP-PA-003**	Prueba superada
CERTILOC-**PP-PA-004**	Prueba superada
CERTILOC-**PP-PA-005**	Prueba superada
CERTILOC-**PP-PA-006**	Prueba superada
CERTILOC-**PP-PA-007**	Prueba superada
CERTILOC-**PP-PA-008**	Prueba superada
CERTILOC-**PP-PA-009**	Prueba superada
CERTILOC-**PP-PA-010**	Prueba superada
CERTILOC-**PP-PA-011**	Prueba superada
CERTILOC-**PP-PA-012**	Prueba superada
CERTILOC-**PP-PA-013**	Prueba superada
CERTILOC-**PP-PA-014**	Prueba superada
CERTILOC-**PP-PA-015**	Prueba no ejecutada (Prueba relacionada con un requisito de baja prioridad que no se ha podido comprobar por falta de coordinación entre los miembros de desarrollo del demostrador de CERTILOC)

Tabla 122. *Resultados de la aplicación de las Pruebas de aceptación del sistema*

A pesar de haber pasado todas las pruebas con éxito el caso del sistema de políticas de privacidad no es distinto del de otros productos software. Todas las aplicaciones informáticas, excepto las que desaparecen o caen en el olvido, en sus primeras versiones, no siempre han contemplado todos los posibles casos de ejecución durante las pruebas de sistema con lo que se tienden a mejorar o, cuanto menos, a mantener. Si algún día se plantease poner en producción el sistema de políticas de privacidad, en un entorno real, es recomendable que se complete el plan de pruebas de sistema para considerar todos los casos posibles.

6 CONCLUSIONES DEL PROYECTO

En el presente apartado se presentan las conclusiones generales derivadas de la implementación del proyecto.

En primer lugar, se presentan las contribuciones concretas del presente PFC al conjunto del proyecto CERTILOC.

A continuación, se presentan distintas evaluaciones sobre los distintos artefactos software obtenidos tras la implementación.

Además, se presentan unas conclusiones generales sobre el proceso de desarrollo en general y de las dificultades encontradas y superadas durante dicho proceso.

Por último, se hará una conclusión general de este proyecto en particular y del proyecto global en el que se enmarca, CERTILOC.

6.1 CONTRIBUCIONES APORTADAS POR EL PRESENTE PFC AL CONJUNTO DE CERTILOC

Presentamos a continuación y en orden de dificultad las distintas contribuciones importantes del presente PFC al conjunto del demostrador de CERTILOC y al marco del proyecto CERTILOC en general.

En primer lugar, cabe destacar la adaptación del sistema de políticas de privacidad al estándar XACML (XACML - OASIS 2009). Esta aportación nos permite afirmar que el sistema de políticas de privacidad (SPP) del demostrador o prototipo de CERTILOC es **fiable y de calidad,** ya que está desarrollado siguiendo las directrices de verdaderos expertos en el universo de los sistemas de autenticación y la autorización.

Además la adaptación al estándar XACML, permite una futura interacción de CERTILOC con otros sistemas que también lo utilicen.

La complicación derivada de adaptar el sistema de políticas de privacidad al estándar implica varios aspectos a tener en cuenta:

- Una comprensión completa del propio estándar teniendo en cuenta la poca documentación al respecto y su uso limitado en la actualidad.
- La búsqueda de utilidades, herramientas o librerías que permitan facilitar la adaptación al estándar, una vez más teniendo en cuenta su poca extensión de uso en la actualidad. En el caso del actual PFC, se ha tenido que estudiar en profundidad el API Sun's

XACML Implementation (XACML - Sun Microsystems , Inc. 2009) para comprender dónde se debía modificar o ampliar para su uso con CERTILOC. Esto ha supuesto el llegar incluso a depurar dicho API para comprender completamente su funcionamiento. La herramienta se ha ampliado para incluir las siguientes aportaciones:

- Creando **nuevos atributos XACML**, para el tipo de datos Coordenada.
- Añadiendo **nuevas funciones XACML**, para la evaluación de información espacial y temporal.
- **Ampliando la manera en que se combinan determinadas políticas o reglas** XACML mediante algoritmos de combinación, para provocar una respuesta de autorización. En este sentido, se ha ampliado el estándar para poder registrar la actividad de reglas, políticas y peticiones.
- Creando un **módulo de carga de políticas de privacidad desde una base de datos MySQL**. Originalmente, el API desarrollado por Sun Microsystems está concebido para trabajar únicamente con ficheros de texto planos y no con datos albergados en una base de datos.

- Ampliaciones al propio estándar XACML para cumplir las necesidades de concretas de CERTILOC. En este sentido, se ha ampliado el estándar para poder tener en cuenta distintos parámetros, como son el rol de usuario o la información espacio temporal de determinado dispositivo, en las políticas de privacidad.

- Amoldar el flujo de datos entre actores que propone XACML al modelo de flujo de datos de CERTILOC. El estándar XACML propone una traducción de datos entre el formato nativo de los datos del sistema donde se implanta el sistema de políticas y el propio lenguaje XACML (traducción de peticiones de autorización, de políticas de privacidad y de repuestas de autorización). El presente PFC ha integrado el modelo de flujo de datos propuesto por XACML, con la arquitectura propuesta para el sistema de políticas de privacidad (SPP) de CERTILOC. Esto facilita en gran medida la futura ampliación del SPP y la introducción de nuevos agentes en el modelo de flujo de datos.

En segundo lugar, otra de las aportaciones claves del presente PFC al conjunto de CERTILOC, ha sido abstraer todo el modelo de datos del estándar XACML para poder plasmarlo en una vista de entidades y relaciones que posteriormente se han llevado a una base de datos MySQL. Cabe remarcar que el modelo de datos de políticas de XACML es complejo y poco trivial con lo que ha supuesto un incremento grande en el esfuerzo invertido para el desarrollo. Esta aportación permite que se traten los datos de las políticas de privacidad de manera

dinámica y que se les apliquen reglas de negocio tanto de CERTILOC como de XACML en tiempo de ejecución.

Además para aportar escalabilidad al modelo de datos del sistema de políticas de privacidad, se ha utilizado una capa intermedia de acceso a datos. De esta manera, los datos de la base de datos no son accedidos directamente mediante conjuntos de registros y sentencias SQL sino que se pueden tratar los datos como instancias concretas de clases. Si en un futuro se necesitase ampliar el modelo de datos de políticas de privacidad con nuevas entidades o relaciones, se podría hacer con cierta facilidad y reutilizando en gran medida el código implementado en el presente PFC.

En tercer y último lugar, este PFC ha aportado una interfaz de usuario Web avanzada e intuitiva para la gestión de políticas de privacidad por parte de los usuarios de CERTILOC. El hecho de haber escogido una base de datos para la persistencia de las políticas de privacidad, ha permitido el poder crear una interfaz de usuario intuitiva para la creación y edición de los elementos de las políticas de privacidad. Una vez más, cabe remarcar que el modelo de datos que propone el estándar XACML para la definición de políticas de privacidad es complejo y muy específico. Antes de evaluar determinada petición de autorización, todas las políticas se traducen al lenguaje XACML y por consiguiente al lenguaje XML. En caso que los usuarios de CERTILOC tuviesen que escribir a mano sus políticas de privacidad en formato XACML (XML), supondría que un usuario sólo podría definir una política de privacidad en caso de conocer muy a fondo el lenguaje XACML. Conociendo de primera mano el citado estándar, puedo afirmar rotundamente que esta es una tarea tediosa y muy complicada y que la detección de errores se hace increíblemente difícil y, en casos de políticas muy grandes, prácticamente imposible. La interfaz web para la gestión de políticas de privacidad, aporta una disminución enorme en cuanto a la dificultad para la definición de las distintas políticas de privacidad por parte de los usuarios de CERTILOC.

6.2 EVALUACIÓN DE LOS PRODUCTOS SOFTWARE OBTENIDOS

Tal y como hemos visto en apartados anteriores, la implementación del presente proyecto ha supuesto la generación de varios artefactos software.

Sería fácil y cómodo no evaluar la calidad general de los distintos productos generados, pero es muy necesario tener un punto de vista crítico para poder plantear mejoras generales y líneas de desarrollo futuras. Para poder cuantificar la calidad de los productos, haremos una pequeña evaluación de cada uno de ellos por separado:

- Sistemas ACP y AGPA: Los sistemas ACP y AGPA, son los encargados de la evaluación de las distintas peticiones de autorización de CERTILOC, contra las políticas de privacidad activas en el sistema y de la traducción del contexto de CERTILOC al contexto o lenguaje de XACML y viceversa. Los productos finales obtenidos, son bastante completos y cumplen todos los requisitos planteados para el proyecto. Estos sub-sistemas podrían ser utilizados, sin ser modificados, para su puesta en marcha en un entorno de producción real. Estos módulos, podrían ser ampliados introduciendo nuevas funciones para evaluación de condiciones y aplicativos y creando nuevos algoritmos de combinación (vemos un detalle de las futuras líneas de desarrollo posibles en el apartado 7 del presente documento), pero su estado actual es totalmente válido para un entorno de producción real.
- Sistema SGP: El producto final derivado del desarrollo del sistema SGP cumple todas las especificaciones de los requisitos de usuario, incluso algunos más, pero podría ser mejorado, sobre todo en términos de validación de formularios. En un principio, el sistema SGP estaba pensado para permitir a los usuarios simplemente, guardar, borrar y modificar archivos con formato XACML que contenían políticas de privacidad. El haber elegido una base de datos para guardar los datos de las políticas de privacidad, nos ha permitido que los usuarios del gestor de políticas de privacidad puedan visualizar un menú, en forma de árbol, donde pueden ir navegando y editando los distintos elementos contenidos en sus conjuntos de políticas de privacidad. Además, el sistema SGP se ha ampliado para poder dar un soporte multi-lenguaje a los usuarios.
- Sistema MARPP: El desarrollo del sistema MARPP, encargado de custodiar la base de datos de políticas de privacidad, ha derivado en un producto de calidad fácilmente escalable. El producto final es bastante completo y cumple fielmente todos los requisitos planteados (los referentes a la custodia de los datos de la aplicación). Mientras no cambie el modelo de datos de políticas de privacidad de CERTILOC, las clases incluidas en los paquetes CERTILOC.marpp.dao y CERTILOC.marpp.dal, no deberían sufrir modificación alguna. Sin embargo, la fachada del sistema, es decir, la clase CERTILOC.marpp.MARPPSystem, podría ampliarse implementando nuevas funciones para obtener distintas informaciones del sistema de políticas, como por ejemplo, indicar a qué usuario pertenece un determinado elemento de los datos (políticas, conjuntos de políticas, reglas, etc.) del sistema de políticas de privacidad. De esta manera, se podría aumentar la seguridad del sistema SGP para, antes de devolver la información de un determinado elemento, comprobar si éste pertenece al usuario CERTILOC que está intentando acceder a él.

En conjunto, el producto software derivado del desarrollo del SPP y por consiguiente, del presente PFC, es un producto totalmente capaz de cumplir las expectativas del demostrador real de CERTILOC. Hay que tener en cuenta que esta es la primera implementación del demostrador de CERTILOC y sienta una base bastante firme para futuras ampliaciones y mejoras del sistema de políticas de privacidad.

Dado que este PFC concluye la creación del primer demostrador de CERTILOC, cabe hacer una pequeña mención y evaluación del conjunto del proyecto y de la unión de los distintos PFCs que han permitido su creación.

La suma del esfuerzo de los distintos desarrolladores de esta primera implementación del demostrador y la capacidad de coordinación del equipo de dirección ha desembocado en la implementación de un primer prototipo de CERTILOC completamente funcional.

Como ya se ha comentado a lo largo del documento, el prototipo incluye gran número de funciones completamente implementadas de las que destacan las siguientes:

- La obtención de IET proveniente incluso de dispositivos reales como son los GPS conectados a PDAs (Memoria PFC - de Fuentes García-Romero de Tejada 2007)
- La Certificación digital de la IET a parte de su almacenamiento y gestión (Memoria PFC - Calvo Martínez 2007)
- Gestión de usuarios, dispositivos, autenticación y autorización en cuanto al manejo de información IET (Memoria PFC - Gallo Martínez 2008).
- El respeto a la privacidad de los usuarios mediante el seguimiento y el uso de un sistema de políticas de privacidad diseñado en base a un lenguaje estándar de seguridad de calidad y muy fiable (Memoria PFC – John Pater 2009).

Dado su carácter de prototipo, posiblemente el sistema no esté capacitado para entrar en producción dando servicios a usuarios relacionados con dispositivos de forma masiva. Este prototipo sin embargo, demuestra que la creación de un sistema de certificación de la localización espacio-temporal, respetuoso con la privacidad de los usuarios es posible.

El primer demostrador de CERTILOC sienta unas bases bastante buenas, amplias y completas a la posible creación de un sistema de certificación de la localización a gran escala.

6.3 PROCESO DE DESARROLLO

Veremos a continuación una serie de conclusiones sobre el proceso de desarrollo en general y, por otro lado, una descripción concreta de las distintas dificultades encontradas y superadas durante dicho proceso.

6.3.1 CONCLUSIONES GENERALES SOBRE EL DESARROLLO GLOBAL

Desde un principio, los objetivos que debía cumplir el sistema de políticas de privacidad de CERTILOC eran claros y concisos. Esto ha permitido desarrollar el sistema, de una manera independiente al resto de módulos que componen el proyecto.

El proceso de análisis y de elección de tecnologías para la implementación ha sido relativamente sencillo ya que el marco del proyecto CERTILOC estaba definido previamente. Esto ha permitido que se conozca de antemano la información que manejaban el resto de módulos que componen CERTILOC. Además, las funciones de cada módulo estaban perfectamente delimitadas lo que ha permitido que, aún desarrollando un proyecto dividido en cuatro PFCs, todos estos se integrasen de una manera gradual y organizada. De esta manera, se ha conseguido un demostrador completo de CERTILOC que funciona prácticamente tal y como se exigió en un principio.

Por otro lado, el proceso de diseño y de implementación del presente PFC, se han complicado bastante al tener que utilizar XACML como hilo conductor. Ha sido complicado implementar los distintos requisitos necesarios para CERTILOC utilizando el sistema XACML y se han tenido que realizar muchas pruebas para completar el conocimiento del estándar. Las dificultades durante el proceso de implementación, tal y como se comenta en el siguiente apartado, no derivaban de una mala especificación de requisitos y objetivos, sino de la precaria documentación existente para el uso de XACML.

Todas las fases de desarrollo se han documentado de manera independiente, lo que ha simplificado en cierta medida el proceso completo de desarrollo. El mayor problema al documentar las distintas fases, ha sido encontrar definiciones claras de los distintos términos utilizados en la definición del estándar XACML. Muchos de estos términos son técnicos y sólo existen dentro del universo de XACML, por lo que, encontrar traducciones y definiciones ha sido complicado. Este problema no se ha encontrado sólo al crear la documentación sino también al tener que entender la documentación ofrecida por los autores del estándar.

Por otro lado, el entorno de desarrollo utilizado era sencillo y con un amplio soporte. Los problemas derivados de dicho entorno, se han resuelto con rapidez. El uso de Eclipse como herramienta de desarrollo está muy extendido y se integra muy bien tanto con MySQL (Bases de datos) como con Apache Tomcat (Servidores Web) como con Exadel Studio (Entorno de Desarrollo para aplicaciones con STRUTS). Además, el hecho de ser una herramienta tan extendida, nos ha permitido encontrar varios componentes para completar su funcionalidad y mejorar la calidad del proceso de desarrollo.

El uso de herramientas de modelado UML integradas con el entorno de desarrollo de la aplicación (Omondo - The Live UML Company 2009), nos ha permitido definir, de manera sencilla, varias partes del código y también a crear una documentación amplia sobre el diseño del sistema.

Los distintos patrones de diseño utilizados han simplificado la resolución, de una manera rápida y simple, de varios de los problemas del actual proyecto. Además, han ayudado a generar un código ampliable y reutilizable y a mejorar la comprensión que puedan hacer posibles futuros desarrolladores del proyecto.

6.3.2 DIFICULTADES DEL PROCESO DE DESARROLLO

Se describen a continuación las dificultades encontradas durante el proceso de desarrollo así como la manera de tratar cada una de ellas.

La principal dificultad encontrada durante el proceso de desarrollo ha sido la implementación siguiendo el estándar de políticas de privacidad de XACML. Hay que decir su uso no está muy extendido y existe muy poco soporte documentado. La propia documentación ofrecida por los autores de dicho estándar, es una documentación densa y difícil de entender. Existen incluso partes del sistema de políticas de privacidad que no vienen explicadas específicamente en el estándar por lo que muchas veces se ha tenido que intuir su funcionamiento.

Por otro lado, la implementación de Sun del estándar XACML (XACML - Sun Microsystems , Inc. 2009), es un API que tampoco ofrece a penas documentación específica. Esto ha supuesto un problema desde el comienzo del desarrollo. En múltiples ocasiones se ha tenido que averiguar su funcionamiento a base de depurar el código ofrecido en el propio API. Dado que la evaluación de peticiones de autorización contra conjuntos de políticas, en formato XACML, se hace enteramente utilizando este API, ha sido necesario interceder en el proceso de evaluación para poder registrar la actividad de las políticas y reglas en la aplicación. Esto nos

permite mostrar, a los distintos usuarios, la actividad de sus políticas y sus reglas. Al no existir documentación sobre cómo se realiza el proceso de evaluación de peticiones dentro del API se ha tenido que intuir su funcionamiento para la correcta configuración del sistema de políticas de privacidad de CERTILOC.

En muchas ocasiones, para comprender a fondo el estándar XACML y su implementación por parte de Sun, no ha habido más remedio que aplicar el método de prueba y error.

Además, la implementación de XACML de Sun utilizada en el desarrollo del actual proyecto, no ofrece métodos alternativos para albergar políticas de privacidad más que en ficheros. Dado que el sistema de políticas de privacidad de CERTILOC pretende utilizar una base de datos para albergar las distintas políticas, se ha tenido que crear un diseño de datos completo para poder alcanzar este objetivo. En un principio, se evaluó si merecería la pena transferir el origen de datos de ficheros a una base de datos, y se decidió que así era ya que iba a simplificar enormemente la gestión de las políticas de privacidad por parte de los usuarios. Una vez más, nos enfrentábamos a lo desconocido ya que en ningún lugar de la documentación de las librerías utilizadas, se indicaba cómo se podría alcanzar este objetivo.

Otra de las principales dificultades afrontadas ha sido el decidir cómo mostrar la información de las políticas de privacidad de CERTILOC a sus usuarios. El estándar XACML, no es un estándar trivial que cualquier usuario pueda entender, por lo que se ha hecho un gran esfuerzo para plantear una manera cómoda para crear, modificar, borrar, activar y desactivar políticas de privacidad. Una vez que se decidió cómo mostrar los conjuntos de políticas de privacidad a los distintos usuarios, hubo que encontrar componentes compatibles con J2EE y con STRUTS para poder crear árboles dinámicos de políticas de privacidad. El proyecto *struts-menu* (Sourceforge 2009) nos ha ayudado a conseguir nuestro objetivo, de una manera razonablemente cómoda aunque su integración con el sistema de políticas de privacidad tampoco ha sido sencilla. Durante el desarrollo del módulo SGP, se pensó que merecía la pena invertir un esfuerzo extra para conseguir desarrollar una interfaz de usuario sencilla, intuitiva y, sobre todo, cómoda. Para el desarrollo de este módulo, se ha seguido una pequeña guía de usabilidad (Usabilidad Web 2009) y se ha prestado especial atención a los siguientes puntos:

- La organización de las páginas utilizando encabezados, listas y una estructura consistente, además de utilizar CSS para la maquetación donde ha sido posible.
- La estructuración de los datos mediante tablas para facilitar la lectura línea a línea.

- Los enlaces de hipertexto utilizando texto con sentido leído dentro del contexto.

Por último, el entorno de desarrollo ha generado ciertos problemas interrumpiendo en diversas ocasiones el proceso de desarrollo. El hecho de integrar Eclipse con los distintos servidores y componentes, en ocasiones provocaba cierres inesperados de la aplicación y un funcionamiento lento. No han sido pocas las veces a lo largo del proceso de desarrollo en que se ha tenido que purgar la información del espacio de trabajo de Eclipse para restablecer su funcionamiento.

6.4 CONCLUSIONES PERSONALES

El desarrollo del sistema de políticas de privacidad de CERTILOC, ha sido largo y, en ocasiones, bastante complicado, pero por otro lado, ha sido enormemente enriquecedor.

No sólo ha permitido ampliar de manera considerable el conocimiento de las distintas tecnologías de las que depende, sino que además ha permitido conocer el funcionamiento de un posible sistema de seguridad.

El hecho de haber elegido CERTILOC como marco de desarrollo para mi proyecto de fin de carrera, no estaba derivado de la mera casualidad, sino que partía del interés por el conocimiento de los distintos mecanismos y sistemas de seguridad existentes en las tecnologías de la información hoy en día. Puedo decir que mis objetivos se han cumplido completamente.

Los conocimientos adquiridos durante la realización de la carrera (Ingeniería Informática), han sido el principal motor del desarrollo, y no ha habido ninguna parte del proceso de desarrollo donde me estuviese enfrentando a áreas desconocidas de las tecnologías de la información.

Por otro lado, el enfrentarse solo al desarrollo del proyecto, ha derivado en una madurez en cuanto a la capacidad de dirigir proyectos software de gran envergadura. Cabe remarcar que todos los procesos de cada fase de desarrollo han sido planeados, desarrollados y comprobados, lo que ha permitido adoptar y comprender todos los puntos de vista o roles de desarrollo que suelen existir en este tipo de proyectos.

La utilización de XACML ha supuesto ampliar los conocimientos de la tecnología XML así como a aumentar la visión de los sistemas de seguridad para las tecnologías de la información, existentes hoy en día. XACML es un proyecto ambicioso que pretende unificar los

distintos sistemas de políticas de privacidad existentes en el universo de la informática. A pesar que este estándar no está muy extendido en la actualidad, la idea básica es muy buena y podría extenderse en cualquier momento. Ha sido un privilegio conocer esta tecnología de primera mano. Mis convicciones me llevan a pensar que, si en un futuro surgen nuevas ideas para la unificación de sistemas de seguridad, a buen seguro tendrán una relación estrecha con esta tecnología, o en su defecto, con una arquitectura parecida a la planteada en el estándar.

Por último, cabe comentar que el desarrollo del proyecto ha permitido poner a prueba todos los conocimientos adquiridos durante la realización de mis estudios en informática, y más en concreto, de los adquiridos en los cursos 4º y 5º. Hay que decir que la especialidad escogida durante estos cursos han sido precisamente los sistemas informáticos distribuidos. Esto ha permitido conocer a fondo las distintas tecnologías utilizadas, incluso antes de comenzar el desarrollo del proyecto actual.

Los resultados obtenidos en la generación de los distintos artefactos software, son completamente funcionales y totalmente ampliables y reutilizables.

Por último comentar que, si se tuviese la oportunidad de replantear el proyecto desde cero, creo que se adoptarían las mismas soluciones aplicadas en el presente proyecto para los problemas que se resuelven, o en su defecto, algunas muy similares. Lógicamente, algunas de las soluciones adoptadas podrían ser replanteadas pero, en líneas generales, la arquitectura de la aplicación en conjunto es sólida y fiable, y sienta unas buenas bases para una futura implementación real y puesta en producción de CERTILOC.

7 FUTURAS LÍNEAS DE DESARROLLO

En el presente apartado se presentan posibles futuras líneas de desarrollo para mejorar y ampliar el sistema de políticas de privacidad de CERTILOC. A continuación se listan todas las mejoras propuestas:

- **Nuevas funciones para parámetros de localización**: El estándar XACML incluye muchas funciones aplicables para las políticas de privacidad XACML pero, cuando se necesitan funciones concretas, éstas deben ser implementadas de manera manual. Como vimos en el diseño del sistema (apartado 4 del presente documento), para CERTILOC se han desarrollado dos funciones a medida que se encuentran en las distintas clases del paquete CERTILOC.base.xacml.cond. Actualmente hay sólo dos funciones definidas: TimeInRangeFuncion y LocationInRectangleAreaFunction. La primera nos indica si una hora está dentro de un rango definido por dos horas y la segunda nos indica si una coordenada está dentro del área de un rectángulo definido por su coordenada superior derecha (cota superior) y su coordenada inferior izquierda (cota inferior). Lo ideal para CERTILOC sería tener distintos tipos de funciones para evaluar parámetros de localización con lo que se podrían crear funciones similares a LocationInRectangelAreaFunction que tuviesen en cuenta otras formas geométricas como polígonos o incluso formas personalizadas que definan áreas o zonas geográficas.

- **Nuevos algoritmos de combinación**: Tal y como se comentó en el diseño del sistema, se han tenido que crear clases para implementar algoritmos de combinación para poder registrar la actividad de las políticas de privacidad de la aplicación. Según los requisitos iniciales, necesitábamos que los usuarios de CERTILOC pudiesen visualizar las distintas actividades que hayan tenido sus reglas y políticas de privacidad ante peticiones de autorización. En tiempo de ejecución, sólo podemos afirmar que una determinada regla o política de seguridad influye en la respuesta ante una petición de autorización si durante la ejecución del algoritmo de combinación de su nodo padre (conjunto de políticas para políticas o política concreta para reglas), se puede encontrar una respuesta concreta de autorización o denegación de la petición. Los algoritmos implementados, utilizan el modelo del algoritmo XACML PermitOverrides, es decir, “Permitir prevalece”. Convendría implementar nuevos algoritmos de combinación en el paquete CERTILOC.base.xacml.combine. Otros algoritmos futuros, pueden utilizar otros modelos de algoritmos definidos en el estándar XACML, por ejemplo “DenyOverrides” o “FirstApplicable” – “Denegar prevalece” y “Primer resultado aplicable” respectivamente.

- **Mejoras sobre el sistema SGP:**
  - Actualmente, los distintos formularios presentados en la interfaz web de usuario de gestión de políticas (módulo SGP), no realizan validación de campos. En un futuro, convendría implementar la validación de campos de los distintos formularios. Cabe remarcar que lo que aquí se propone, no es solo la simple validación de tipos de datos o campos vacíos de formularios. Sería ideal crear una validación dinámica de corrección en cuanto al formato XACML que estamos dando a nuestras políticas. Esta validación pasaría por evaluar si los atributos de cada uno de los elementos de un conjunto de políticas, está bien formado. Por ejemplo para atributos que deben tener la morfología de una URI (The Internet Society 1998), que dicha URI esté bien formada y cumpla las especificaciones concretas del formato de cadenas URI. Por otro lado, debe evaluar si el número de argumentos, pasados a las funciones de aplicativos y condiciones, es correcto y tienen tipos correctos para esa función. Actualmente, el usuario final dispone de funciones para convertir sus políticas y reglas de seguridad al formato XACML. En caso que haya algún error en la conversión de la política, regla o conjunto de políticas, se indicará al usuario que existen errores en el formato del elemento pero no se indica dónde. Indicar exactamente dónde se encuentra el error, es complicado, ya que la conversión se hace desde la librería "Sun's XACML Implementation" y nosotros perdemos, en cierto sentido, el control de la conversión. Sin embargo, un sistema de validación de formularios Web previo, que indique si un campo, de cierto elemento, es apto para su conversión a XACML, podría evitar que se generasen problemas en la conversión.
  - Por otro lado, el sistema SGP actual, no comprueba si un elemento de una política de privacidad, definido por su "Id", pertenece al mismo usuario que lo está gestionando. Si nos encontráramos ante un usuario mal intencionado, con algo de perspicacia y conocimientos avanzados en tecnologías web, éste podría llegar a seleccionar elementos de políticas de privacidad que no le pertenecen. Para poder resolver este problema, sería necesario implementar, en el sistema MARPP, una función que indicase si un elemento determinado pertenece a un determinado usuario CERTILOC. Por otro lado, habría que modificar la clase *CERTILOC.sgp.actions.GetSelectedAction* para que comprobase la pertenencia del elemento seleccionado al usuario que ejecuta

la sesión. Éste objetivo parece más sencillo de lo que en realidad es, ya que, en la base de datos de políticas de privacidad, sólo se asocian Conjuntos de políticas a usuarios de CERTILOC. Para poder evaluar, por ejemplo, si cierta condición pertenece a un usuario concreto, deberíamos ir subiendo en el árbol desde la condición, hasta encontrar el conjunto de políticas al que pertenece, para poder evaluar si éste pertenece al usuario que se va a comprobar – si el conjunto de políticas pertenece al usuario, entonces la condición también le pertenecerá.

- Por otro lado, existe un último tipo de comprobación que debería ser implementada. Al definir una política de seguridad, si el usuario está definiendo una política de seguridad para un dispositivo que no le pertenece, no debería dejarle crear la política. Al modificar o guardar un nuevo especificador de atributo, si tiene relación con urn:oasis:names:tc:xacml:1.0:resource:resource-id, se debería comprobar que el **valor de atributo** asociado a este especificador de atributo, pertenece al usuario que ha definido el especificador. Una vez más esta no es una contribución sencilla ya que en XACML los valores de atributos y los especificadores de atributos se definen de manera independiente y no es tan sencillo llevar un control de los valores de atributos y especificadores asociados definidos por un usuario.

- **Distribución de módulos ACP, AGPA, SGP y MARPP en distintos servidores**: Para poder alcanzar este objetivo habría que modificar cierta parte del proyecto. En particular, habría que crear todo un sistema de paso de mensajes entre los módulos para poder transferir la información de unos a otros. En un principio, todas las fachadas de cada sistema – ACPSystem, AGPASystem y MARPPSystem – se han diseñado e implementado siguiendo el patrón de diseño Singleton. Esto permite tratar cada instancia de cada Sistema como un elemento único con lo que se deja el código preparado para una posible implementación distribuida. Una implementación distribuida del sistema de políticas de privacidad, es un objetivo ambicioso y pasaría por modificar buena parte del código desarrollado.

8 MANUAL DE USUARIO

Se presenta en el siguiente anexo un pequeño manual de usuario del Sistema de Gestión de Políticas de privacidad – SGP – para ayudar a los potenciales usuarios a gestionar sus políticas.

8.1 ACCEDER AL SGP

- **Paso 1**: El usuario accede a la aplicación.
- **Paso 2**: El usuario introduce su nombre de usuario, contraseña y elige un rol y presiona el botón “Login con contraseña” (en caso de estar accediendo con usuario y contraseña).
- **Paso 3:** De la lista de servicios disponibles, elige la última opción, “Servicios Gestión Privacidad ”

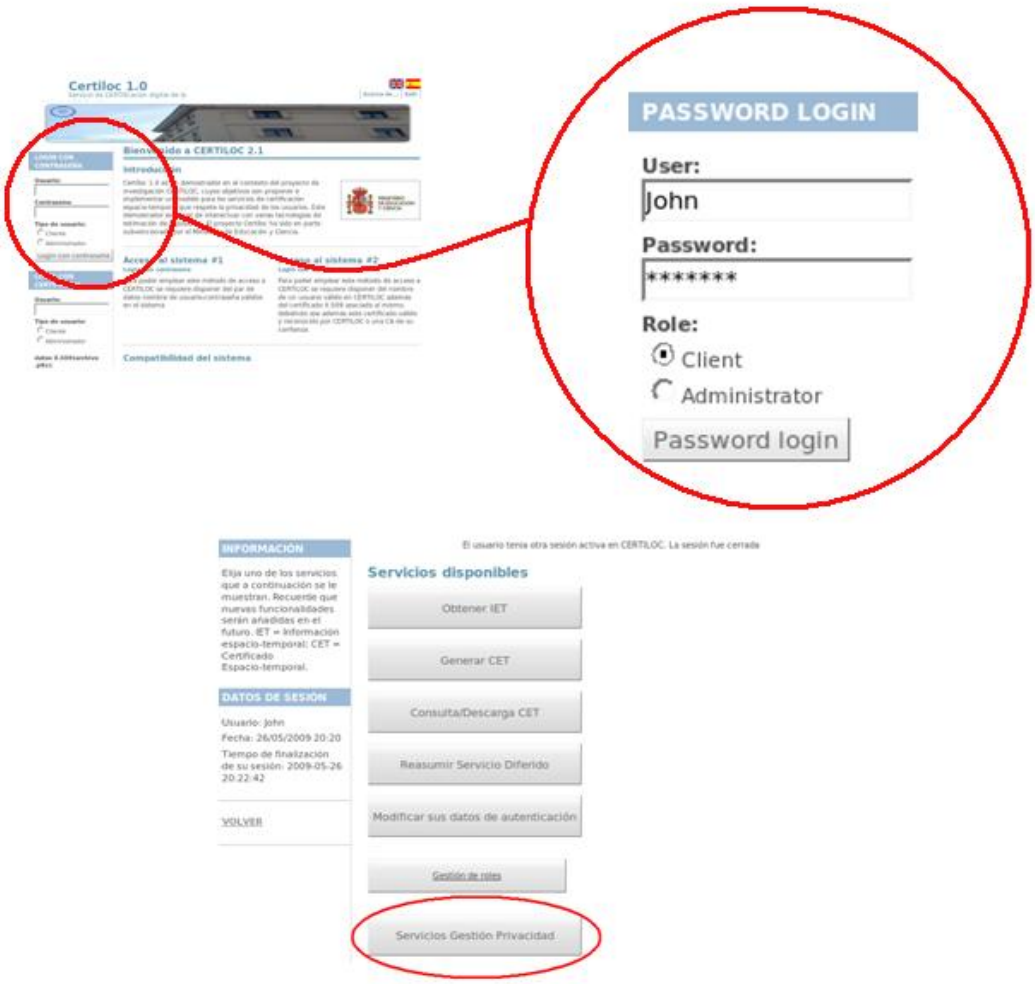


Figura 59. *Manual de usuario: Acceder al SGP*

8.2 INICIO DEL SGP

Una vez dentro del sistema de gestión de políticas, el usuario tiene dos opciones:

- **Ver sus políticas de privacidad:** Para pasar a la gestión de políticas de privacidad del usuario.
- **Registros de actividad de políticas:** Para acceder a los registros de actividad de la aplicación relacionados con el usuario.

Figura 60. *Manual de usuario: Inicio del sistema de Gestión de políticas.*

8.3 GESTIÓN DE POLÍTICAS DE PRIVACIDAD

Cuando el usuario accede a gestionar sus políticas de privacidad se presentará una pantalla como la siguiente:

Figura 61. *Manual de usuario: Raíz de la gestión de políticas de privacidad.*

En este punto, el usuario puede elegir un nodo del árbol para ver sus detalles o crear un nuevo conjunto de políticas de privacidad.

8.3.1 FICHAS DE ELEMENTOS DEL ÁRBOL DE POLÍTICAS

Al seleccionar un nodo en el árbol de políticas de privacidad, el usuario accede a los detalles del elemento.

La ficha de todos los elementos ofrece la siguiente información y funciones:

Detalles del elemento	Identificador: politica1 Algoritmo de combinación: urn:certiloc:1.0:rule-combining-algorithm:permit-overrides Descripición: Esta es una política de ejemplo Estado: habilitado
Raíz del árbol a partir elemento seleccionado	⊞ **conjunto_politicas:John:46708123456789**
Botones para realizar acciones	Botones para **Editar**, **Borrar**, **Ver en formato XACML**, **Añadir otro elemento**, etc.

8.3.2 FORMULARIOS PARA AGREGAR NUEVOS ELEMENTOS

Para crear un elemento, el usuario debe presionar sobre cualquier botón que indique “Añadir <ELEMENTO A AÑADIR>”. Cuando se ha accedido a la ficha de algún elemento,

podremos presionar sobre el botón Añadir para añadir nuevos elementos hijos al nodo en el que nos encontramos.

Detalles a tener en cuenta al crear un nuevo elemento:

- Todos los **identificadores** deben tener la morfología de una **URI** (The Internet Society 1998)
- En los **conjuntos de políticas y las políticas**, para que éstas generen registros de actividad de políticas, **se deben elegir sólo algoritmos de combinación** creados expresamente para CERTILOC
- En los **conjuntos de políticas** se deben **elegir** sólo algoritmos de combinación de **políticas**
- En las **políticas** se deben **elegir** sólo algoritmos de combinación de **reglas**
- **Descripción**: En los elementos que se pide una descripción, se debe introducir una **pequeña descripción** del elemento creado pero es **opcional**.
- El **estado** se marcará como **habilitado o deshabilitado** según se desee que el elemento **actúe contra posibles peticiones de autorización**
- Se debe seleccionar el emisor **urn:**CERTILOC**:1.0:issuer:admin@**CERTILOC**.com** al crear especificadores de atributo sólo si el atributo ha sido creado expresamente para CERTILOC. En caso de especificadores de atributo que vengan por **defecto en XACML**, no se debe introducir **ningún emisor**.
- Se debe presionar el botón **guardar** si se quiere terminar de agregar el nuevo elemento.

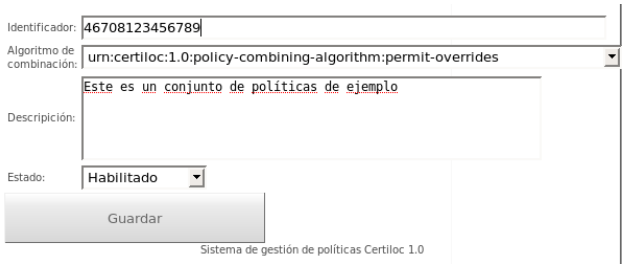


Figura 62. *Manual de usuario: Formularios para agregar nuevos Elemento*

8.3.3 FORMULARIOS PARA EDITAR ELEMENTOS

Desde la ficha de un nodo, podemos presionar el botón Editar cuando deseemos editar los atributos concretos de ese nodo.

Se debe presionar el botón **guardar** si se quieren guardar los cambios realizados sobre el elemento.

8.3.4 BORRAR ELEMENTOS DEL ÁRBOL DE POLÍTICAS

Desde la ficha de un nodo, podemos presionar el botón Borrar cuando deseemos borrar por completo el elemento y sus nodos hijos.

Una vez borrado, el usuario es dirigido a la raíz de sus conjuntos de políticas de privacidad.

8.3.5 VER ELEMENTO EN FORMATO XACML

Desde la ficha de un nodo, podemos presionar el botón ***"Ver en XACML"*** cuando deseemos obtener una representación con formato XACML del elemento que estamos visualizando.

Conviene utilizar esta función con frecuencia para evaluar posibles errores en conjuntos de políticas, políticas y reglas de privacidad.

Figura 63. *Manual de usuario: Vista XACML de un elemento del árbol de políticas*

8.4 VISTA DE REGISTROS DE ACTIVIDAD DE LA APLICACIÓN

Desde el inicio del SGP, el usuario puede elegir ver los registros de actividad de la aplicación relacionados con peticiones de autorización para dispositivos de los que él es responsable.

En la pantalla inicial de esta sección el usuario verá un panel como el siguiente:

Figura 64. *Manual de usuario: Panel de registros de actividad de la aplicación*

El usuario podrá visualizar los detalles de los registros de actividad de generados por:

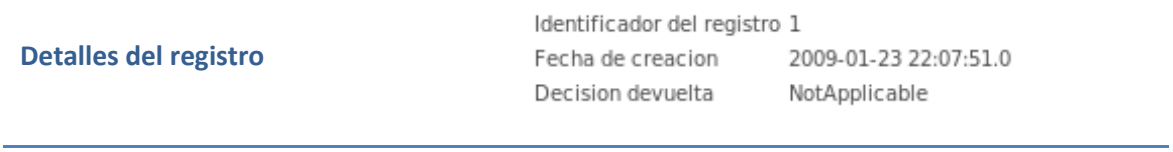
- Peticiones de autorización realizadas sobre dispositivos de los que es responsable
- Actividad de sus Políticas (se considera que hay actividad en una política cuando es ella la que determina la respuesta de permitir o denegar)
- Actividad de sus Reglas (se considera que hay actividad en una regla cuando es ella la que determina la respuesta de permitir o denegar) – Es una manera de ver en detalle una actividad concreta. Una política puede tener infinitas reglas con lo que, si sólo se mostrasen los registros de actividad de políticas, el usuario habría veces que no sabría porqué se ha devuelto una respuesta concreta. Es por esto que se ha decidido separar la actividad de las políticas de la actividad de las reglas.

Todos los registros se presentan de la siguiente manera:

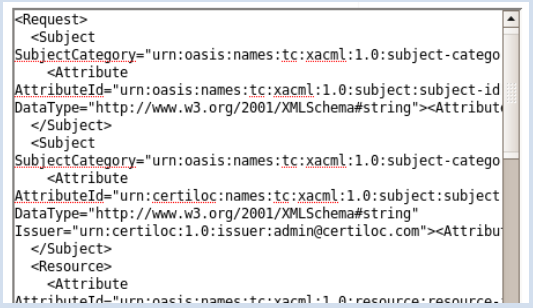
Se indica el número de registro, la fecha y hora de creación del mismo y un botón para ver sus detalles.

8.4.1 DETALLE DE REGISTROS DE ACTIVIDAD DE PETICIÓN

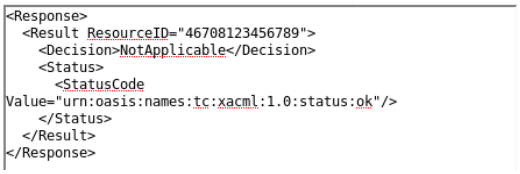
Los registros de actividad de petición muestran la siguiente información:



Petición de autorización recibida



Respuesta devuelta



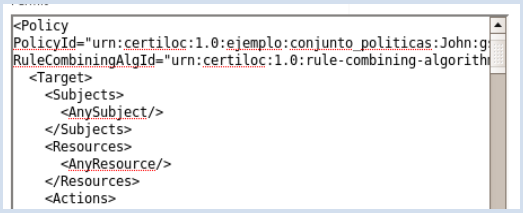
8.4.2 DETALLE DE REGISTROS DE ACTIVIDAD DE POLÍTICAS

Los registros de actividad de políticas muestran la siguiente información:

Detalles del registro

Identificador del registro	1
Fecha de creacion	2009-01-23 22:17:46.0
Decision devuelta	Permit

Política implicada en la respuesta

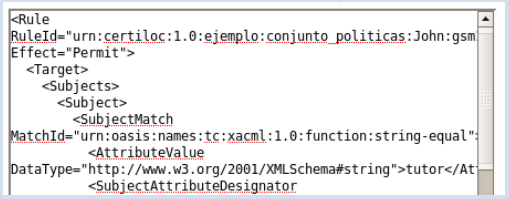


8.4.3 DETALLE DE REGISTROS DE ACTIVIDAD DE REGLAS

Los registros de actividad de reglas muestran la siguiente información:

Detalles del registro

Identificador del registro	1
Fecha de creacion	2009-01-23 22:17:46.0
Decision devuelta	Permit

Política implicada en la respuesta	

8.5 ELEMENTOS COMUNES EN LA APLICACIÓN

Se presentan a continuación los elementos básicos de funcionamiento que se repiten a lo largo de todo el SGP.

8.5.1 CABECERA DE LA GESTIÓN DE POLÍTICAS

A lo largo del sistema de gestión de políticas se podrá ver una cabecera que contiene:

- Un enlace a opciones del gestor de políticas de privacidad – Lleva al usuario al Inicio del SGP
- Un enlace a raíz de conjuntos de políticas de privacidad – Lleva al usuario a la raíz de sus políticas de privacidad.
- Un título – Indica la ubicación del usuario dentro del SGP

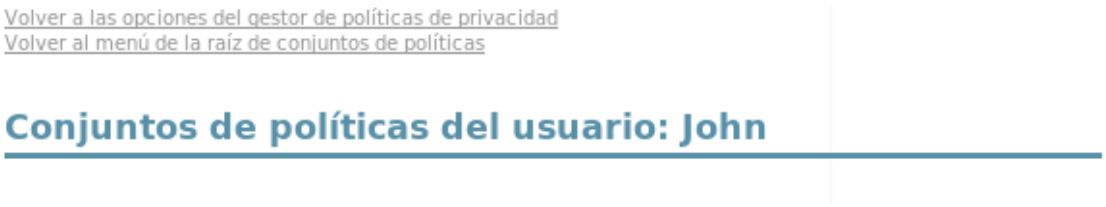


Figura 65. *Manual de usurio: Cabecera del SGP*

8.5.2 MENÚ DE NAVEGACIÓN DE POLÍTICAS

Las políticas de privacidad, se presentan a los usuarios como un menú con forma de árbol.

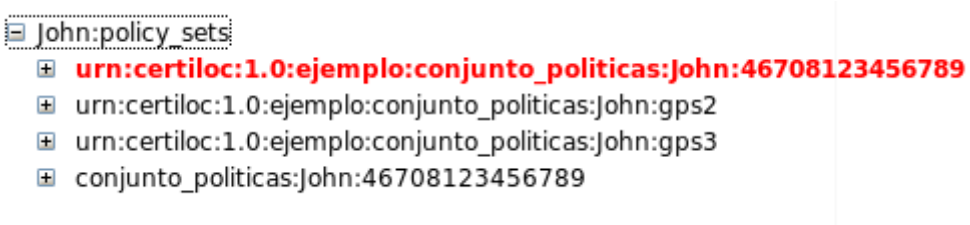


Figura 66. *Manual de usuario: Menú de navegación de políticas de privacidad*

Para desplazarse por el menú el usuario puede realizar las siguientes acciones:

- Desplegar un nodo presionando sobre el b n
- Contraer un nodo presionando sobre el bo
- Acceder a un nodo presionando sobre su 1 politica1

8.5.3 BOTONES

A lo largo del SGP podemos encontrar varios botones para realizar distintas acciones. Todos los botones comparten el mismo formato e indican la acción que realizan con su nombre.

Figura 67. *Botón*

9 TUTORIAL DE CREACIÓN DE POLÍTICAS DE PRIVACIDAD

Se presenta en el siguiente apartado un pequeño tutorial para la creación de un conjunto de políticas de privacidad.

Vamos a crear una política que permita realizar la **ACCIÓN ObtenerIET** (obtener información espacio temporal) de uno de nuestros dispositivos, sólo en caso que el **SUJETO** que esté intentando llevar a cabo la acción lo haga con el ROL de **tutor**, el **RECURSO** a localizar sea el dispositivo **46708123456789**, y con la condición de que la **hora de la petición** (hora de petición actual, no la original en caso de ser una petición diferida) sea entre las 08:00:00 y las 23:00:00.

Paso 1 - Accedemos a la aplicación CERTILOC

Paso 2 – Introducimos nuestro nombre de usuario, contraseña y elegimos un rol y presionamos el botón “Login con contraseña”

Paso 3 - De la lista de servicios disponibles, elegimos la última opción “Servicios Gestión Privacidad”

Paso 4 - Dentro del sistema de “Servicios Gestión Privacidad”, elegimos la opción “Ver sus políticas de privacidad”

Paso 5 - Presionamos el botón “Nuevo conjunto de políticas”.

Rellenamos el formulario de nuevo elemento indicando:

- identificador (con morfología URI): ***46708123456789***
- un algoritmo de combinación de políticas: ***urn:***CERTILOC***:1.0:policy-combining-algorithm:permit-overrides***
- descripción: ***Este es un conjunto de políticas de ejemplo***
- estado: ***habilitado***
- Presionamos el botón “Guardar”

Paso 7 – Podemos comprobar que la nueva política se ha creado satisfactoriamente. Pinchamos sobre la nueva política para seguir añadiendo elementos.

Paso 8 - Entramos en el nuevo conjunto de políticas, y presionamos el botón “Ver en XACML” para ver si todos los datos introducidos son correctos y no hay problemas. Si conseguimos ver la nueva política en formato XACML todo irá bien, en caso contrario se mostrará una ventana de error.

Paso 9 - Agregar nueva obligación: Seleccionamos el nuevo conjunto de políticas y elegimos la opción "Agregar Obligación".

Rellenamos el formulario indicando:

- identificador con la morfología propia de una URI: ***urn:***CERTILOC***:1.0:obligacion:ejemplo:terminos-legales-uso-datos***
- ¿Cuándo se devuelve la obligación?: ***Permit***
- Presionamos el **botón "Guardar"**

Paso 10 - Agregar asignación de atributo a la obligación: Seleccionamos la nueva obligación y presionamos el botón "Agregar asignador de atributo"

Rellenamos el formulario indicando:

- identificador con morfología de una URI: ***urn:***CERTILOC***:1.0:ejemplo:contenido***
- el tipo de atributo que contiene: ***http://www.w3.org/2001/XMLSchema#string***
- Presionamos el **botón "Guardar"**

Paso 11 - Agregar un valor a la asignación de atributo de la obligación: Seleccionamos la nueva asignación de atributo y presionamos el botón "Agregar valor de atributo"

Rellenamos el formulario indicando:

- valor: "***Datos confidenciales para uso exclusivo del usuario solicitante***"
- el tipo de atributo que contiene: ***http://www.w3.org/2001/XMLSchema#string***
- Presiona el **botón "Guardar"**

Paso 12 - Agregar un objetivo al conjunto de políticas creado: Seleccionamos el nuevo conjunto de políticas y elegimos la opción "Agregar Objetivo".

Rellenamos el formulario indicando:

- tipo de objetivo: ***Resource***
- una función de evaluación de coincidencia: ***urn:oasis:names:tc:xacml:1.0:function:string-equal***
- Presiona el **botón "Guardar"**

Paso 13 - Agregar un especificador de atributo al nuevo objetivo: Seleccionamos el nuevo objetivo creado y elegimos la opción “Agregar Designador de Atributo”.

Rellena el formulario indicando:

- identificador: ***urn:oasis:names:tc:xacml:1.0:resource:resource-id***
- tipo de especificador: ***Resource Target***
- tipo de atributo: ***http://www.w3.org/2001/XMLSchema#string***
- emisor del atributo: ***vacío***
- Presiona el **botón “Guardar”**

Paso 14 - Agregar un valor de atributo al nuevo objetivo: Seleccionamos el nuevo objetivo creado y elegimos la opción “Agregar Valor de Atributo”.

Rellenamos el formulario indicando:

- valor: ***46708123456789***
- tipo de atributo que contiene: ***http://www.w3.org/2001/XMLSchema#string***
- Presiona el botón “Guardar”

Paso 15 - Agregar una nueva política al conjunto de políticas creado: Seleccionamos el nuevo conjunto de políticas y elegimos la opción “Agregar Política”.

Rellenamos el formulario indicando:

- identificador con morfología URI: ***politica1***
- un algoritmo de combinación de reglas: ***urn:***CERTILOC***:1.0:rule-combining-algorithm:permit-overrides***
- descripción: ***Esta es una política de ejemplo***
- estado: ***habilitado***
- Presiona el botón “Guardar”

Paso 16 - Agregar un objetivo a la política creada: Seleccionamos la política creada y elegimos la opción “Agregar Objetivo”.

Rellenamos el formulario indicando:

- tipo de objetivo: ***Action***
- función evaluación coin.: ***urn:oasis:names:tc:xacml:1.0:function:string-equal***
- Presiona el ***botón “Guardar”***

Paso 17 - Agregar un especificador de atributo al nuevo objetivo: Seleccionamos el nuevo objetivo creado y elegimos la opción “Agregar Designador de Atributo”.

Rellenamos el formulario indicando:

- identificador con morfología de una URI: ***urn:oasis:names:tc:xacml:1.0:action:action-id***
- tipo de especificador: ***Action Target***
- tipo de atributo: ***http://www.w3.org/2001/XMLSchema#string***
- emisor del atributo: ***vacío***
- Presiona el **botón “Guardar”**

Paso 18 - Agregar un valor de atributo al nuevo objetivo: Seleccionamos el nuevo objetivo creado y elegimos la opción “Agregar Valor de Atributo”.

Rellenamos el formulario indicando:

- valor: ***obtenerIET***
- tipo de atributo que contiene: ***http://www.w3.org/2001/XMLSchema#string***
- Presiona el **botón “Guardar”**

Paso 19 - Agregar una nueva regla a la política creada: Seleccionamos la política creada y elegimos la opción “Agregar Regla”.

Rellenamos el formulario indicando:

- identificador con morfología de una URI: ***regla1***
- Efecto: ***Permitir***
- Descripción: ***Esto es una regla de ejemplo***
- Estado: ***Habilitado***
- Presiona el botón “Guardar”

Paso 20 - Agregar un objetivo a la regla creada: Seleccionamos la regla creada y elegimos la opción “Agregar Objetivo”.

Rellenamos el formulario indicando:

- tipo de objetivo: ***Subject***
- función evaluación coin.: ***urn:oasis:names:tc:xacml:1.0:function:string-equal***
- Presiona el **botón “Guardar”**

Paso 21 - Agregar un especificador de atributo al nuevo objetivo: Seleccionamos el nuevo objetivo creado y elegimos la opción “Agregar Designador de Atributo”.

Rellenamos el formulario indicando:

- identificador: ***urn:***CERTILOC***:names:tc:xacml:1.0:subject:subject-role***
- tipo de especificador: ***Subject Target***
- tipo de atributo: ***http://www.w3.org/2001/XMLSchema#string***
- emisor del atributo: ***urn:***CERTILOC***:1.0:issuer:admin@***CERTILOC***.com***
- Presiona el **botón “Guardar”**

Paso 22 - Agregar un valor de atributo al nuevo objetivo: Seleccionamos el nuevo objetivo creado y elegimos la opción “Agregar Valor de Atributo”.

Rellenamos el formulario indicando:

- valor: ***tutor***
- tipo de atributo que contiene: ***http://www.w3.org/2001/XMLSchema#string***
- Presiona el **botón “Guardar”**

Paso 23 - Agregar una condición a la nueva regla: Seleccionamos la nueva regla creada y elegimos la opción “Agregar Condición”.

Rellenamos el formulario indicando:

- identificador: ***http://research.sun.com/projects/xacml/names/function#time-in-range***
- el estado de la condición: ***habilitado***
- Presiona el **botón “Guardar”**

Paso 24 - Agregar Valores de atributo a la nueva condición: Seleccionamos la nueva condición creada y elegimos la opción “Agregar Valor de atributo” para indicar desde qué hora se puede obtener la IET.

Rellenamos el formulario indicando:

- valor: ***08:00:00***
- el tipo de atributo que contiene: ***http://www.w3.org/2001/XMLSchema#time***
- Presiona el botón “Guardar”

Volvemos a la condición y agregamos un nuevo atributo para indicar hasta qué hora se puede obtener la IET. Seleccionamos la opción “Agregar Valor de atributo”.

Rellena el formulario indicando:

- valor: ***23:00:00***
- el tipo de atributo que contiene: ***http://www.w3.org/2001/XMLSchema#time***
- Presiona el **botón “Guardar”**

Paso 25 - Agregar un aplicativo a la condición creada: Seleccionamos la nueva condición creada y elegimos la opción “Agregar Aplicativo”.

Rellenamos el formulario indicando:

- identificador: ***urn:oasis:names:tc:xacml:1.0:function:time-one-and-only***
- el estado de la condición: ***habilitado***
- Presiona el **botón “Guardar”**

Paso 26: Agregar un especificador de atributo al aplicativo creado: Seleccionamos el nuevo aplicativo creado y elegimos la opción “Agregar Designador de Atributo”.

Rellenamos el formulario indicando:

- Identificador: ***urn:***CERTILOC***:1.0:environment:hora-peticion***
- Emisor del atributo: ***urn:***CERTILOC***:1.0:issuer:admin@***CERTILOC***.com***
- Attribute Type: ***http://www.w3.org/2001/XMLSchema#time***
- Tipo de Designador: ***Environment Target***
- Presiona el **botón “Guardar”**

Paso 27 - Comprobar que todo ha ido bien: Seleccionamos el conjunto de políticas creado desde la raíz de conjuntos de políticas y, si no se han cometido errores, cuando seleccionemos el botón “Ver en XACML”, debería aparecer la descripción XACML del conjunto de políticas creado.

10 BIBLIOGRAFÍA Y REFERENCIAS DOCUMENTALES

- Agencia Española de Protección de Datos. *Portal Web Agencia Española de Protección de Datos.* 2009. https://www.agpd.es/portalweb/index-ides-idphp.php.
- BOE 04-04-2009. «XVI Convenio Colectivo empresas de CONSULTORÍA y ESTUDIOS DE MERCADO y de OPINIÓN PÚBLICA (2007-09).» 2009.
- BOE núm. 298. «Ley Orgánica 15/1999, de 13 de diciembre, de Protección de Datos de Carácter Personal.» 1999.
- Canonical Ltd. Ubuntu . *About Ubuntu.* 2009. http://www.ubuntu.com/.
- Consejo Superior de Administración Electrónica. *Sitio web corporativo.* 2009. http://www.csae.map.es/.
- Diccionario - Real Academia de la Lengua Española. *Diccionario de la Real Academia de la Lengua Española.* 2009. http://www.rae.es/rae.html.
- Exadel, Inc. *Exadel Studio Pro.* 2009. http://www.exadel.com/web/portal/products/ExadelStudioPro.
- Freeman, Eric Freeman & Elisabeth. «Head First: Design Patterns.» En *Head First: Design Patterns*, de Eric Freeman & Elisabeth Freeman. O'REILLY, 2004.
- Fundación Eclipse. *Fundación Eclipse.* 2009. http://plataformaeclipse.com/.
- Gonzalez-Tablas Ferreres - Tesis, Ana Isabel. 2005.
- Gonzalez-Tablas, A.I., J.M. Fuentes, J.C. Calvo, A Orfila, J Gallo, y J Pater. «CERTILOC: Análisis y diseño de un servicio de certificación espacio-temporal respetuoso con la privacidad.» 2007.
- Gonzalez-Tablas, Ana Isabel, John Pater, Jose Maria Fuentes, Johanna Gallo, y Jose Carlos Calvo. «ESPECIFICACIÓN DE LOS REQUISITOS Y ANÁLISIS PRELIMINAR DEL SISTEMA CERTILOC.» 2007.
- Hibernate - Red Hat, Inc. *Relational Persistence for Java and .NET.* 2009. https://www.hibernate.org/.
- Java - Sun Microsystems , Inc. *Java.* 2009. http://java.sun.com/.
- JDBaccess.com. *What is JDBaccess?* 2008. http://jdbaccess.com/.
- Log4J - Apache Software Foundation. *Apache Logging Services - Log4J.* 2007. http://logging.apache.org/log4j/1.2/index.html.
- Logging - Sun Microsystems, Inc. *Package java.util.logging.* 2004. http://java.sun.com/j2se/1.5.0/docs/api/java/util/logging/package-summary.html.

- Memoria PFC - Calvo Martínez, Jose Carlos. «Diseño e implementación de la plataforma base de CERTILOC y del servicio de certificación espacio-temporal.» 2007.
- Memoria PFC - de Fuentes García-Romero de Tejada, José María. «Memoria PFC - Diseño e implementación del sistema de localización de dispositivos móviles con conectividad limitada dotados de receptor GPS para CERTILOC.» 2007.
- Memoria PFC - Gallo Martínez, Johanna. «Memoria Proyecto de fin de carrera.» 2008.
- Métrica V.3 - Consejo Superior de Administración Electrónica. *Métrica V.3.* 2009. http://www.csi.map.es/csi/metrica3/index.html.
- Microsoft. *The Official Microsoft Information Server Site.* 2009. http://www.iis.net.
- MVC - Sun Microsystems , Inc. *MVC.* 2009. http://java.sun.com/blueprints/patterns/MVC-detailed.html.
- MySQL AB. *Why MySQL?* 2009. http://www.mysql.com/why-mysql/.
- Netcraft. 2009. http://news.netcraft.com/archives/2008/01/28/january_2008_web_server_survey.html.
- OASIS. *OASIS Security Services.* 2009. http://www.oasis-open.org/committees/tc_home.php?wg_abbrev=security.
- Omondo - The Live UML Company. *Omondo Eclipse UML.* 2009. http://www.eclipsedownload.com/.
- Portal Web doc.ubuntu-es.org. *Sitio web de doc.ubuntu-es.org.* 2009. http://doc.ubuntu-es.org/Sobre_Ubuntu.
- Sourceforge. *Struts-menu.* 2009. http://struts-menu.sourceforge.net/.
- Struts - Apache Software Foundation. *Struts.* 2009. http://struts.apache.org/.
- Struts - Apache Software Foundation. «Struts.» http://struts.apache.org/, 2009.
- Sun Microsystems , Inc. *Sitio Web Corporativo.* 2009. http://es.sun.com/.
- The Apache Software Foundation. *Apache Tomcat.* 2009. http://tomcat.apache.org/.
- The Internet Society. «Uniform Resource Identifiers (URI): Generic Syntax.» 1998. http://www.ietf.org/rfc/rfc2396.txt.
- Usabilidad Web. «GUÍA BREVE PARA CREAR SITIOS WEBS ACCESIBLES .» 2009. http://www.usabilidad-web.com/articulos-usabilidad/accesibilidad/guia-rapida-usabilidad.html.

- Visual Paradigm for UML. *Smart Development Environment Community Edition for Eclipse.* 2009. http://www.visual-paradigm.com/product/sde/ec/communityedition.jsp.
- XACML - OASIS. *eXtensible Access Control Markup Language (XACML).* 2009. http://www.oasis-open.org/committees/tc_home.php?wg_abbrev=xacml.
- XACML - Sun Microsystems , Inc. *Sun's XACML Implementation.* 2009. http://sunxacml.sourceforge.net/.