Implementación de un sistema unificado de autenticación de usuarios aplicado a los diferentes servicios de la Universidad Tecnológica de Bolívar

SEBASTIÁN GONZÁLEZ DÍAZ

UNIVERSIDAD TECNOLÓGICA DE BOLÍVAR

FACULTAD DE INGENIERÍA

PROGRAMA DE INGENIERÍA DE SISTEMAS

CARTAGENA D.T.H. Y C.

2010

Implementación de un sistema unificado de autenticación de usuarios aplicado a los diferentes servicios de la Universidad Tecnológica de Bolívar

SEBASTIÁN GONZÁLEZ DÍAZ

Director:

JAIRO ENRIQUE SERRANO CASTAÑEDA

Ingeniero de sistemas

Magister en software libre

UNIVERSIDAD TECNOLÓGICA DE BOLÍVAR

FACULTAD DE INGENIERÍA

PROGRAMA DE INGENIERÍA DE SISTEMAS

CARTAGENA D.T.H. Y C.

2010

CONTENIDO

GRÁFICOS

TABLAS

Agradezco a Dios por ser mi eterno acompañante y guía. A mi querida madre Ana Díaz,que me ha acompañado toda la vida y quien es mi inspiración para seguir adelante. A mi padre Cesar Augusto González y a mi abuelo Cesar Tulio González, por ser quienes me han patrocinado en todo este camino llamado vida. A mi hermano por ser un motivo más para seguir adelante. A mi tutor Jairo Serrano por ser mi consejero en este y muchos otros trabajos. Y por último pero no menos importante, a mi amigos Giordano Amir Nobles Escandón, David Julián Tete Mieles, Jeisson José Guevara Mendivil, Julián Manuel Ramírez Chávez y Diego Germán Navarro Tesillos; excelentes personas que conocí gracias a mi paso por la UTB.

Atte

Sebastián González Díaz

INTRODUCCIÓN

Actualmente, las computadoras han penetrado en nuestra sociedad de una forma tan veloz e incontrolada, que hemos llegado a un punto en que muchas personas y familias de recursos medios, y algunas de escasos recursos (gracias a proyectos como CPE: Computadores para educar), tienen acceso a un computador personal de escritorio o portatil.

En la Universidad Tecnológica de Bolívar (la cual se nombrará en adelante como UTB) esto no es la excepción, ya que la mayoría de las dependencias (sino son todas) cuentan con al menos un computador personal, y los estudiantes tienen acceso a varios salones de sistemas y una amplia sala de internet en las bibliotecas tanto de la sede de Ternera como en la sede de Manga, además de poder conectarse a internet inalámbrico en diferentes puntos de acceso dispuestos en ambos campus universitarios.

Además de los equipos (hardware) con los que cuenta la UTB, también cuenta con una infraestructura de servicios web, incluyendo dentro de esta un sistema para la gestión de la información académica (SIRIUS), una plataforma de aprendizaje virtual (SAVIO), una página web institucional, etc.

Todos estos servicios web tienen alguna forma de autenticación, ya sea para entrar únicamente a partes restringidas para realizar labores administrativas dentro del sitio, o a un espacio personal con información específica donde la persona va hacer uso del servicio como tal, y que tiene su perfil de usuario previamente establecido.

De entre todos estos servicios, dos (2) tienen una sincronización en cuanto a lo que se refiere a credenciales de acceso por medio de una interfaz que hace que uno se alimente de la información del otro.

Adicional a esto, en una tesis de pregrado anterior se había propuesto el diseño e implementación de un sistema de autenticación normalizado para la unificación de una red de servicios informáticos, sobre la cual se basa este proyecto.

Por lo tanto, el presente trabajo tiene como objetivo extender esa producción previa, adaptándola a los servicios institucionales actuales y seguir las recomendaciones hechas por las personas que la realizaron.

Este trabajo está organizado en tres grandes secciones que son: la identificación y explicación de la problemática que se está tratando en la UTB (capítulo 1), los métodos existentes para mitigar estos problemas (capítulos 2, 3 y 4) y cual de esos métodos se eligió y se implementó con el fin de que fuera la solución más efectiva (capítulos 5 y 6).

GLOSARIO

Autenticación: en sistemas es el proceso mediante el cual un usuario que intenta acceder es quien dice ser al proveer una serie de datos como entrada al mismo.

Autorización: es el proceso mediante el cual se le otorga a un usuario el permiso de acceder a determinados servicios los cuales estén previamente establecidos. Esto proceso ocurre generalmente como consecuencia de la autenticación en caso de haber resultado exitosa.

Back-end: hace referencia a lo que se encuentra "detrás" de la parte visual de un sistema que se muestra al público, generalmente el sitio donde se almacena la información del mismo.

CMS: son las siglas en inglés de *Content Management System* o sistema de gestión de contenidos en español. Es un software que facilita la administración de contenidos, principalmente web, separando la programación del diseño y ahorrando tiempo en el proceso desarrollo.

Drupal: es un CMS modular y muy configurable, que tiene a su disposición una amplia gama de módulos que se usan para añadirle más funcionalidad. Es

software libre bajo la licencia GNU/GPL, desarrollado en PHP y mantenido por una comunidad de usuarios.

Kerberos: es un protocolo, desarrollado inicialmente por el MIT, usado para la autenticación en redes de computadores donde dos equipos de cómputo pueden verificar su identidad mutuamente de manera segura, aunque se encuentren en una red insegura.

Moodle: es un CMS enfocado en la gestión de contenidos educativos basados en internet. Es software libre bajo la licencia GNU/GPL, desarrollado en PHP y soporta varios tipos de bases de datos.

OSI: son las siglas en inglés de *Open System Interconnection* y es un modelo de red creado por la Organización Internacional para la Estandarización (ISO) el cual define una arquitectura para la interconexión de diferentes sistemas de comunicaciones. Este está basado en siete capas que abstraen todo el proceso de comunicación entre dos aplicaciones en red.

Servicio: en el ámbito de sistemas, este término hace referencia a la(s) funcionalidad(es) que ofrece un software para su utilización por medio de unas interfaces que provee el mismo.

SSL: es un protocolo criptográfico a nivel de aplicación en la pila de protocolos TCP/IP que asegura en la capa de transporte una comunicación segura entre los

dos extremos. Provee una autenticación entre ambos extremos de la comunicación usando criptografía y el envío de certificados. Este protocolo fue reemplazado posteriormente por TLS.

TCP/IP: es el nombre con el que se denomina comúnmente a una familia de protocolos de red que permiten la transmisión de datos entre computadores. Es una arquitectura basada en capas al igual que el modelo OSI, pero con la diferencia de que solo la componen cuatro capas y no siete.

Ticket-granting ticket: es un archivo de identificación encriptado y con un tiempo límite de validez el cual es otorgado a un usuario después de un proceso de autenticación exitoso.

TLS: es un protocolo criptográfico sucesor de SSL y que surgió como mejora a este. Aunque tiene la misma funcionalidad y objetivo de SSL no es capaz de interoperar con este.

Panorama de la integración en los sistemas de la UTB

Historia

Desde sus inicios, la UTB ha visto la importancia de poseer una infraestructura tecnológica basada en los sistemas de información. Aproximadamente desde el año 1993 existe en la universidad un software llamado SIFAD que soporta los procesos en el área de contabilidad, finanzas y administración. Este software es un desarrollo de la misma universidad.

Posteriormente, en el año 2001, la UTB le apostó a la virtualidad como apoyo al proceso educativo con el Sistema de Aprendizaje Virtual Interactivo (SAVIO), el cual es un sistema web que sirve como apoyo virtual en la educación a los cursos de pregrado de las carreras ofrecidas por esta universidad y al cual pueden acceder los estudiantes a través de internet con su código de estudiante y una contraseña.

Además de SAVIO, también existía el sistema de bibliotecas, el cual permitía a los estudiantes la reserva de libros por internet para poder reclamarlos después en la sede donde debía ir a buscar dicho texto. Este usaba el nombre de usuario y la

contraseña del estudiante para poder acceder a la cuenta del usuario y poder realizar la reserva.

Más recientemente, en el año 2007, se adquirió en la UTB un sistema nuevo para que los estudiantes pudieran realizar la matrícula académica por medio de internet. El Sistema Integrado de Recursos de Información Universitaria para el Servicio (SIRIUS), ayuda a llevar un control sobre todo el historial académico de los estudiantes, además de permitir a los docentes la publicación de las notas durante el semestre por medio de esta plataforma y de esta forma se lleva una comunicación más ágil entre estudiantes y profesores.

Aproximadamente para la misma época en que se adquirió y se puso en funcionamiento SIRIUS, se hizo un cambio en la plataforma SAVIO. Antes había sido un desarrollo propio realizado en la UTB, pero en el 2008 se decidió hacer una migración a la plataforma libre virtual de aprendizaje Moodle, la cual es la que se usa en la actualidad y el sistema recibe el nombre de SAVIO-Moodle.

En el año 2009, la UTB adquirió dos sistemas: Janium.net y Microsoft Exchange 2003. El primero es un sistema de bibliotecas el cual fue adquirido para reemplazar el sistema anterior y suplir algunas necesidades que no se estaban cubriendo.

El segundo es un servicio de correo electrónico que reemplazó al que se usaba previamente. Este se encuentra disponible solamente para los docentes y

directivos de la UTB, y proporciona mayor capacidad de almacenamiento y otras características que el sistema anterior no tenía.

Por último, también en el 2009, en la dependencia de la UTBVirtual se realizó un desarrollo interno llamado vForge. Este es un sistema administrador de proyectos, el cual fue creado para llevar un control de los diferentes proyectos de educación virtual llevados a cabo por la UTBVirtual. En el 2010, se desarrolló la segunda versión de este sistema.

Estado actual

Entre los diferentes servicios de la UTB, se encuentran los siguientes: SIRIUS, SAVIO-Moodle, vForge, sitios web institucionales, servicio de correo electrónico.

SIRIUS es el Sistema Integrado de Recursos de Información Universitaria para el Servicio, y es el que administra toda la información académica de los estudiantes de la Universidad Tecnológica de Bolívar. Este es un sistema web al cual se puede acceder desde internet. Es usado por estudiantes, docentes y las personas que lo administran. Los estudiantes y docentes pueden ingresar al sistema con su ID de usuario y su NIP, el cual está compuesto por 6 caracteres.

Otro de los sistemas es SAVIO-Moodle, que es la plataforma virtual educativa de la UTB, basada en la plataforma de aprendizaje virtual libre, Moodle. Esta se

encarga de ofrecer a estudiantes y docentes un apoyo en la educación, facilitando de esta forma la comunicación y los procesos educativos por medio de ambientes virtuales de aprendizaje.

En el caso de los sitios web institucionales, estos son páginas web públicas que son principalmente para mostrar información universitaria de lo que se está haciendo actualmente, eventos, oferta académica, proyectos, etc. Estos sitios están basados en Drupal, el cual es un administrador de contenidos, y que está disponible para su uso como software libre. Al ser estas webs de dominio público y estar más orientadas a la proyección de la universidad hacia el exterior, tienen menos usuarios registrados que SAVIO-Moodle y SIRIUS ya que solo serían necesarios los que realizan labores administrativas dentro del sitio.

vForge es una aplicación desarrollada completamente en la Dirección de Educación virtual de la UTB, usando como base el framework de desarrollo Symfony. Este es un administrador de proyectos que usan en esta dependencia para realizarle un mejor seguimiento a la producción de cursos virtuales para las carreras que pertenecen a la facultad de Educación Formal para el Trabajo. Esta aplicación es usada por el personal de esta dependencia solamente, así que el número de usuarios es reducido en comparación con SIRIUS o SAVIO-Moodle.

El servicio de correo electrónico institucional está disponible para el personal de la UTB, de tal forma que docentes y directivos pueden tener una cuenta personal de correo electrónico. Este servicio tiene como base Microsoft Exchange y tiene una capacidad de espacio de almacenamiento de hasta 100 MB.

Al tener toda esta variedad de servicios disponibles en el campus universitario, en su mayoría para el uso de la comunidad, y al mismo tiempo que cada uno tenga su propia información almacenada independientemente, hace que haya muchos datos redundantes entre todos los sistemas. Ejemplo: en SAVIO-Moodle y SIRIUS se usan las mismas credenciales de acceso para ambos sistema, lo cual hace que tanto el código de usuario como la contraseña estén almacenados en las bases de datos de ambos sistemas y de esta forma duplicando el espacio de almacenamiento ocupado por esta información.

Aparte del espacio de almacenamiento usado, si un usuario hace uso de más de un servicio, entonces en cada uno podría tener unas credenciales de acceso diferente. Esto puede llegar a crear inconformidad entre los usuarios ya que estos necesitan recordar la clave de cada uno de los sistemas o llevarlas escritas en alguna parte.

Algunos usuarios, para evitar el tener que recordar muchas claves, usan la misma en todos los servicios. Esto es una practica muy insegura, ya que si a un usuario

le roban la clave, la persona que se la robó puede acceder a cada uno de los servicios de los que hace uso la víctima del robo, y este para remediarlo tendría que cambiar la clave en cada sistema.

Como se puede apreciar, al igual que lo mencionó en su entrevista el Director de Servicios Informáticos de la UTB (ver Anexo A), uno de los mayores problemas que se presenta actualmente en la infraestructura tecnológica de los sistemas de información de la universidad es la falta de integración entre los diferentes servicios. Los otros dos problemas que se mencionan en dicha entrevista están por fuera del alcance de este trabajo, así que por lo tanto en este se va a enfocar principalmente en el problema de la integración de los servicios y más específicamente en lo que concierne a la unificación en los datos de los usuarios.

En una tesis de pregrado anterior se propuso la implementación de un sistema para la normalización en el proceso de autenticación de usuarios de la UTB. Dicha tesis, trataba de cubrir las necesidades de ese entonces, las cuales eran el acceso a SAVIO, al webmail institucional, y a un sistema de mensajería instantánea. Esta tesis no llegó a ser implementada en la UTB, así que el problema no se solucionó, sino que por el contrario, con el tiempo se amplió debido a la inclusión de los sistemas más recientes (SIRIUS y biblioteca).

Como medida de urgencia, para tratar de mitigar un poco esta problemática, se realizó una integración entre dos de estos sistemas: SAVIO-Moodle y SIRIUS. Con este mecanismo, se sincronizaron las credenciales de usuario de estos dos sistemas, haciendo que en ambos se utilicen los mismos datos de acceso. Pero esta no es una solución definitiva, ya que solo funciona para estos dos sistemas y no es extendible a otros sistemas que se encuentren actualmente en funcionamiento o a futuros sistemas que se piensen implementar.

Este último mencionado, es el único mecanismo de unificación de credenciales de acceso implementado en la UTB. Antes de ese no se había utilizado ninguno y es el que está en funcionamiento actualmente.

Antecedentes del proyecto

En la UTB, no existe ni ha existido un sistema que unifique o centralice de alguna forma los datos de autenticación de los usuarios. El primer intento de realizar este proyecto fue evidenciado en el año 2006, cuando un equipo de estudiantes, compuesto por Yuranis Henriquez Nuñez, Jairo Enrique Serrano Castañeda y Gilberto Orozco Linero, propuso como trabajo de grado el "*Análisis, Diseño e Implementación de un sistema de autenticación normalizado para la unificación de una red de servicios informáticos, aplicado a la Universidad Tecnológica de Bolívar.*"

En dicho proyecto, se hizo un estudio de los diferentes servicios de los que disponía la UTB en el momento de su realización, los cuales eran: servicio de correo electrónico, mensajería instantánea, plataforma de aprendizaje SAVIO y página web unitecnologica. Dicho estudio fue realizado y propuesto, pero no llego a implementarse.

Con el tiempo, la plataforma de servicios de la UTB creció con la adquisición de nuevos sistemas para brindarle una mejor comodidad a la comunidad académica en el aspecto de acceso y uso de dichos servicios. Primero fue la adquisición e implementación del Sistema Integrado de Recursos de Información Universitaria para el Servicio (SIRIUS) y más recientemente Janium.net como nueva plataforma de administración de la biblioteca. Además de esto, también se hizo un cambio en la plataforma de aprendizaje SAVIO para usar Moodle con unas adaptaciones internas a las necesidades de la universidad.

Con la creciente oferta de servicios por parte de la UTB, se van presentando problemas que antes no se evidenciaban, como por ejemplo, la redundancia en los datos de los usuarios almacenados en cada sistema, la inconformidad por parte de los usuarios al tener diferentes claves de acceso por cada servicio y el incremento en solicitudes para el reclamo de contraseñas extraviadas.

Desde hace un tiempo, está funcionando un sistema de sincronización entre los servicios SAVIO-Moodle y SIRIUS, el cual alimenta las bases de datos de SAVIO-Moodle con la información que necesita de SIRIUS (estudiantes y sus materias).

Este sistema está limitado a los dos servicios mencionados, no es en tiempo real, crea redundancia de datos y es unilateral, es decir, los datos solo viajan en un solo sentido y no en ambos (en este caso, de SIRIUS a SAVIO-Moodle).

Teniendo como base el trabajo de grado "*Análisis, Diseño e Implementación de un sistema de autenticación normalizado para la unificación de una red de servicios informáticos, aplicado a la Universidad Tecnológica de Bolívar*", se puede realizar la implementación de un sistema que tenga unas características semejantes al que se propuso en dicho proyecto pero que esté enfocado a darle solución a los problemas que se han originado más recientemente en la UTB.

Requerimientos del sistema

Tomando como base la información recogida de la entrevista realizada al Director de Servicios Informáticos de la UTB, Juan Carlos Mantilla (ver Anexo A), se puede extraer la siguiente lista de requerimientos que debe cumplir el sistema final que va a ser implementado.

Requerimientos no funcionales

A continuación se encuentra una tabla con la lista de requerimientos no funcionales con los cuales deberá cumplir el sistema a implementar.

Nro	Descripción	Prioridad
1	Que los datos más críticos del sistema (contraseñas) estén protegidos por algún algoritmo de cifrado.	Esencial
2	Que personas no autorizadas no sean capaces de visualizar la información almacenada en el sistema.	Esencial
3	Que el sistema provea las interfaces necesarias para que los usuarios u otros sistemas puedan hacer uso de este fácilmente.	Esencial
4	Que los usuarios del sistema puedan visualizar la información en un formato que puedan entender.	Esencial
5	Que el sistema arroje siempre el resultado correspondiente con la entrada o un mensaje de error en caso de un mal ingreso de datos o de no existir una salida que corresponda.	Esencial
6	Que la información disponible este siempre actualizada o en su defecto que se actualice en un marco de tiempo no mayor a 30 minutos.	Esencial
7	Que el tiempo de respuesta de la aplicación no supere los tres (3) segundos.	Esencial

Tabla No 1: Requerimientos no funcionales del sistema.

Estos requerimientos no funcionales corresponden a las cualidades mínimas enunciadas por el Director de Servicios Informáticos de la UTB, Juan Carlos Mantilla, con las que debe cumplir el sistema a implementar.

Requerimientos funcionales

Para los requerimientos funcionales, el sistema se ha dividido en tres subsistemas, los cuales son: usuarios, sistemas clientes y administración. El subsistema de usuarios se refiere a las operaciones que se realizarían sobre la información que pertenece al personal de la universidad. El de sistemas clientes corresponde a la interacción con otros sistemas de información. Por último, el subsistema de administración hace referencia a la asignación de las labores de los administradores del sistema.

A continuación se encuentran los requerimientos funcionales del sistema divididos por subsistemas:

Subsistema de usuarios

ID	Descripción	Prioridad	Actor(es)
RF101	El sistema debe permitir que se agregue información perteneciente a los diferentes usuarios del sistema (estudiantes, docentes, etc.).	Esencial	Administrador
RF102	El sistema debe permitir que se realicen modificaciones parciales o totales de la información que tiene almacenada de los usuarios.	Esencial	Administrador

RF103	El sistema debe permitir la eliminación de la información de algún usuario del sistema en caso de ser necesario.	Esencial	Administrador
RF104	El sistema debe permitir la consulta de la información perteneciente a los usuarios del sistema.	Esencial	Administrador, sistemas clientes

Tabla No 2: Requerimientos funcionales del subsistema de usuarios

Subsistema de sistemas clientes

ID	**Descripción**	**Prioridad**	**Actor(es)**
RF201	El sistema debe aceptar las conexiones por parte de otros sistemas clientes que estén permitidos.	Esencial	Sistemas clientes
RF202	El sistema debe restringir el acceso a sistemas clientes que no estén autorizados a usar sus servicios.	Esencial	Sistemas clientes
RF203	El sistema debe mantener un listado de los sistemas clientes permitidos.	Esencial	Administrador
RF204	El sistema debe permitir el agregar sistemas clientes nuevos al listado de los permitidos.	Esencial	Administrador
RF205	El sistema debe permitir la consulta del listado de los sistemas clientes	Esencial	Administrador

	permitidos.		
RF206	El sistema debe permitir la eliminación de sistemas clientes del listado de los permitidos.	Esencial	Administrador

Tabla No 3: Requerimientos funcionales del subsistema de sistemas clientes

Subsistema de administración

ID	**Descripción**	**Prioridad**	**Actor(es)**
RF301	El sistema debe permitir añadir al sistema administradores que puedan realizar operaciones administrativas.	Esencial	Administrador
RF302	El sistema debe permitir la limitación de operaciones que puede realizar un administrador en el sistema, ya sea individualmente o por grupos de administradores.	Esencial	Administrador
RF303	El sistema debe permitir la eliminación de administradores del sistema.	Esencial	Administrador

Tabla No 4: Requerimientos funcionales del subsistema de administración

Se han determinado estos como los requerimientos funcionales mínimos con los que debe cumplir la aplicación final para que se puedan satisfacer plenamente las necesidades que origina este problema en la UTB.

Casos de uso

A partir de los requerimientos especificados previamente, se pueden describir los siguientes casos de uso. En algunos casos, en la sección del flujo de eventos se puede llegar a referenciar otro caso de uso debido a que pueden tener flujos de eventos similares.

ID	RF101
Nombre	Agregar usuario al sistema
Descripción	En este proceso se guarda la información de usuarios nuevos dentro del sistema.
Actor principal	Administrador
Actor(es) secundario(s)	(ninguno)
Precondiciones	(ninguna)
Postcondiciones	● Que quede almacenado un nuevo usuario en el sistema.
Flujo de eventos	1. El administrador ingresa la información perteneciente al usuario (ID, nombre, apellidos, etc.). 2. El sistema verifica que el usuario no exista en el sistema. 3. El administrador confirma el registro. 4. El sistema guarda la información del usuario.
Flujos alternativos	2. Si el usuario existe en el sistema ○ Se muestra un mensaje de error

	○ Regresar al paso 1 3. Si el administrador cancela el registro ○ Regresar al paso 1
Prioridad	Esencial

Tabla No 5: Caso de uso RF101

ID	RF102
Nombre	Modificar información de usuario
Descripción	En este proceso se realiza la modificación de la información que se tiene almacenada de un usuario.
Actor principal	Administrador
Actor(es) secundario(s)	(ninguno)
Precondiciones	● Que haya usuarios en el sistema.
Postcondiciones	● Que se cambie la vieja información del usuario, por la nueva que se ingrese.
Flujo de eventos	1. El sistema carga el listado de los usuarios. 2. El sistema muestra el listado que cargó previamente. 3. El administrador selecciona el estudiante al que desea modificar la información. 4. El administrador modifica la información. 5. El administrador confirma la modificación. 6. El sistema guarda las modificaciones.
Flujos alternativos	5. Si el administrador cancela la modificación ○ Regresa al paso 2
Prioridad	Esencial

Tabla No 6: Caso de uso RF102

ID	RF103
Nombre	Eliminar usuario del sistema
Descripción	En este proceso se lleva a cabo la eliminación de la información perteneciente a un usuario.
Actor principal	Administrador
Actor(es) secundario(s)	(ninguno)
Precondiciones	● Que haya usuarios en el sistema.
Postcondiciones	● Que se borre del sistema la información del usuario.
Flujo de eventos	1. El sistema carga el listado de usuarios. 2. El sistema muestra la lista cargada previamente. 3. El administrador selecciona el usuario que va a eliminar. 4. El administrador confirma la operación. 5. El sistema elimina el usuario del sistema.
Flujos alternativos	4. Si el administrador cancela la eliminación ○ Regresa al paso 2
Prioridad	Esencial

Tabla No 7: Caso de uso RF103

ID	RF104
Nombre	Consultar usuarios del sistema
Descripción	En este proceso se muestra la información asociada a un usuario que ya se encuentre registrado.
Actor principal	Administrador
Actor(es) secundario(s)	Sistemas clientes

Precondiciones	● Que haya usuarios en el sistema.
Postcondiciones	● Que se muestren los usuarios que hay.
Flujo de eventos	1. El sistema carga el listado de los usuarios. 2. El sistema muestra el listado de los usuarios almacenados. 3. El administrador selecciona el usuario que desea. 4. El sistema muestra la información detallada del usuario seleccionado.
Flujos alternativos	(ninguno)
Prioridad	Esencial

Tabla No 8: Caso de uso RF104

ID	RF201
Nombre	Gestión de conexiones de sistemas clientes
Descripción	En este proceso se realiza la aceptación de conexiones por parte de sistemas clientes para que hagan uso del servicio.
Actor principal	Sistemas clientes
Actor(es) secundario(s)	(ninguno)
Precondiciones	● Que haya un listado de sistemas clientes permitidos.
Postcondiciones	● Que se acepte o rechace la conexión.
Flujo de eventos	1. El sistema cliente realiza un intento de conexión. 2. El sistema carga el listado de sistemas clientes permitidos.

	3. El sistema comprueba si el sistema cliente que intenta conectarse está dentro del listado cargado previamente. 4. Se acepta la petición de conexión.
Flujos alternativos	3. Si el sistema cliente no está en el listado ○ Se rechaza la petición de conexión
Prioridad	Esencial

Tabla No 9: Caso de uso RF201

ID	RF204
Nombre	Agregar sistemas clientes permitidos
Descripción	En este proceso se lleva a cabo el almacenamiento de nuevos sistemas clientes al sistema de los permitidos.
Actor principal	Administrador
Actor(es) secundario(s)	(ninguno)
Precondiciones	● Que exista el listado de los sistemas clientes permitidos.
Postcondiciones	● Que esté agregado el nuevo sistema cliente al listado de los permitidos.
Flujo de eventos	El flujo de eventos de este caso de uso es similar al del RF101.
Flujos alternativos	Los flujos alternativos de este caso de uso son similares a los del RF101.
Prioridad	Esencial

Tabla No 10: Caso de uso RF204

ID	RF205
Nombre	Consulta de los sistemas clientes permitidos

Descripción	En este proceso se lleva a cabo la consulta del listado de los sistemas clientes permitidos.
Actor principal	Administrador
Actor(es) secundario(s)	(ninguno)
Precondiciones	● Que exista el listado de sistemas clientes permitidos.
Postcondiciones	● Que se muestre el listado de los sistemas clientes permitidos.
Flujo de eventos	1. El sistema carga el listado de los sistemas clientes permitidos. 2. El sistema muestra el listado cargado en el paso anterior.
Flujos alternativos	(ninguno)
Prioridad	Esencial

Tabla No 11: Caso de uso RF205

ID	RF206
Nombre	Eliminar sistemas clientes permitidos
Descripción	En este proceso se lleva a cabo la eliminación de sistemas clientes del listado de permitidos.
Actor principal	Administrador
Actor(es) secundario(s)	(ninguno)
Precondiciones	● Que exista el listado de sistemas clientes permitidos.
Postcondiciones	● Que sea eliminado el sistema cliente de la lista de permitidos.
Flujo de eventos	El flujo de eventos de este caso de uso es similar al del RF103.

Flujos alternativos	Los flujos alternativos de este caso de uso son similares a los del RF103.
Prioridad	Esencial

Tabla No 12: Caso de uso RF206

ID	RF301
Nombre	Agregar administradores al sistema
Descripción	En este proceso se lleva a cabo la inclusión de un nuevo administrador dentro del sistema.
Actor principal	Administrador
Actor(es) secundario(s)	(ninguno)
Precondiciones	(ninguna)
Postcondiciones	● Que se agregué al sistema el nuevo administrador.
Flujo de eventos	El flujo de eventos de este caso de uso es similar al del RF101.
Flujos alternativos	Los flujos alternativos de este caso de uso son similares a los del RF101.
Prioridad	Esencial

Tabla No 13: Caso de uso RF301

ID	RF302
Nombre	Establecer permisos a administradores
Descripción	En este proceso se lleva a cabo la limitación de las operaciones que puede realizar un administrador en el sistema, ya sea para restringir como para permitir.
Actor principal	Administrador
Actor(es) secundario(s)	(ninguno)
Precondiciones	● Que exista algún administrador en el sistema.

Postcondiciones	● Que se actualice la lista de las operaciones permitidas al administrador seleccionado.
Flujo de eventos	1. El sistema carga el listado de administradores. 2. El sistema muestra el listado cargado previamente. 3. El administrador selecciona del listado al administrador o grupo de estos a los que desea establecerle las operaciones permitidas. 4. El sistema carga el listado de operaciones permitidas sobre la información. 5. El sistema muestra el listado que cargó el en paso anterior. 6. El administrador selecciona o deselecciona las operaciones que desea. 7. El administrador confirma la operación. 8. El sistema guarda la nueva información.
Flujos alternativos	7. Si el administrador cancela la operación ○ Regresar al paso 2
Prioridad	Esencial

Tabla No 14: Caso de uso RF302

ID	RF303
Nombre	Eliminar administradores del sistema
Descripción	En este proceso se lleva a cabo el borrado de la información de un administrador del sistema.
Actor principal	Administrador

Actor(es) secundario(s)	(ninguno)
Precondiciones	● Que exista más de un (1) administrador en el sistema.
Postcondiciones	● Que se elimine la información del administrador elegido.
Flujo de eventos	El flujo de eventos de este caso de uso es similar al del RF103.
Flujos alternativos	Los flujos alternativos de este caso de uso son similares a los del RF103.
Prioridad	Esencial

Tabla No 15: Caso de uso RF303

Adicional a los casos de uso previamente desarrollados, se presenta a continuación el diagrama de casos de uso del sistema.

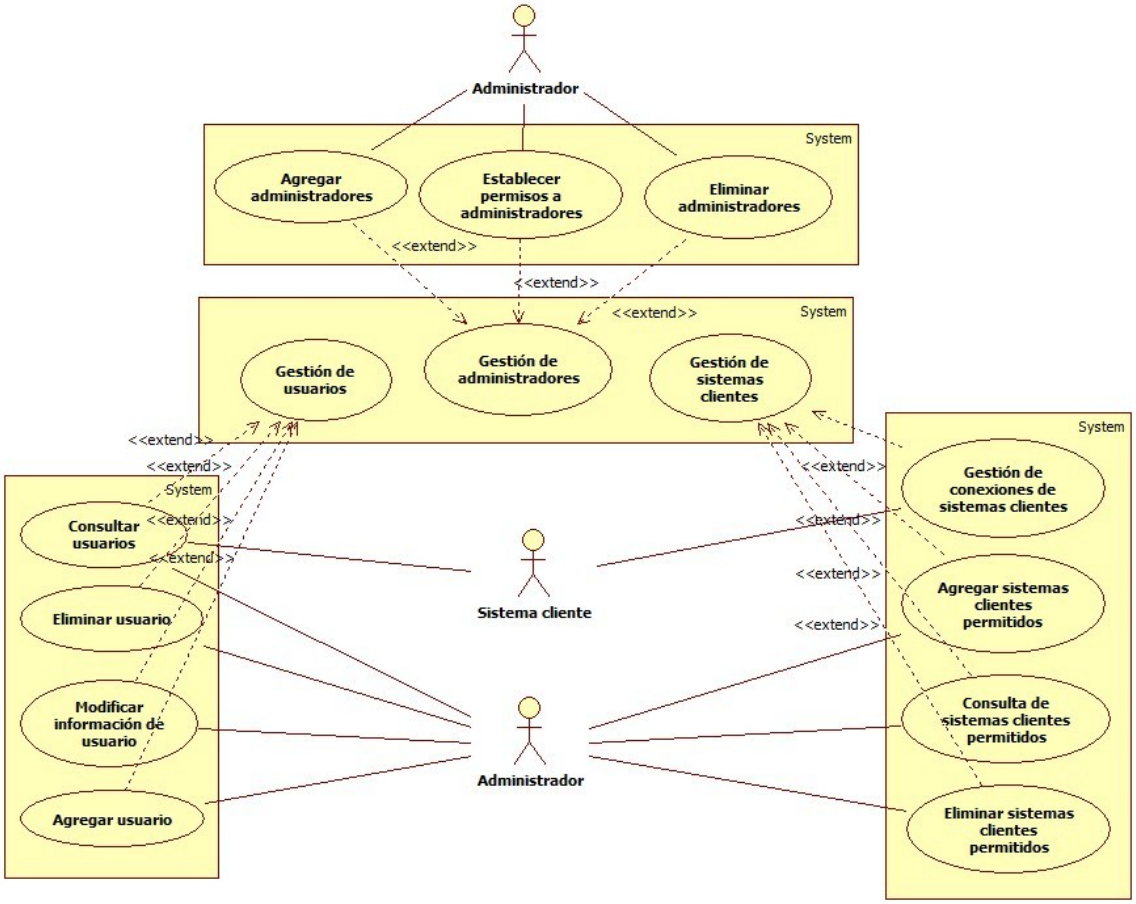

Gráfico No 1: Diagrama de casos de uso del sistema

Teniendo claro que se quiere llegar a conseguir al finalizar con la implementación del sistema se procederá a explorar las diferentes alternativas de solución que existen para esta problemática.

Alternativas de solución

Atendiendo a toda la problemática descrita hasta este punto, existen dos alternativas de solución que pueden cubrir las necesidades actuales. La primera es la adopción de un sistema de autenticación centralizada y la segunda es la implementación de un sistema SSO [3].

Un sistema de autenticación centralizada es aquel que mantiene toda la información de los usuarios (datos personales y claves de acceso) unificada en un solo servidor, independizando de esta labor a los diferentes servicios dentro de una organización y proporcionando más uniformidad e integridad en la información.

En el momento de realizarse la autenticación en algún servicio, este se conecta al sistema de autenticación centralizada y este provee temporalmente los datos de acceso como respuesta a la solicitud del servicio. Estos datos solo son usados para realizar el proceso de autenticación, son desechados después de haber realizado esto y no se almacenan localmente en el sistema que use el servicio para la persistencia de datos.

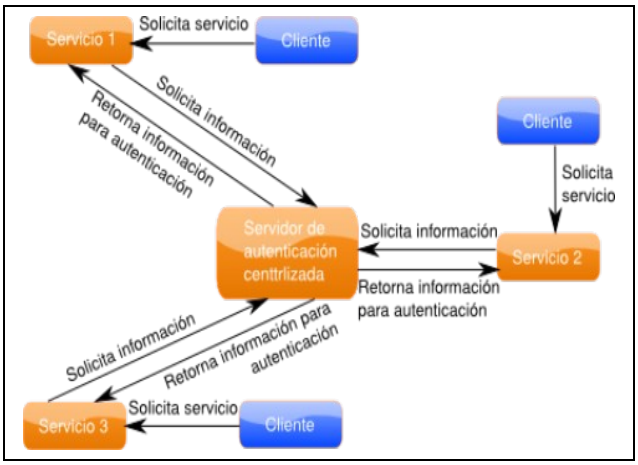

Gráfico No 2: Funcionamiento de un servicio de autenticación centralizada.

Un SSO (siglas de Single Sign-On) es un sistema que permite el acceso a una variedad de servicios realizando una sola vez el proceso de autenticación. En otras palabras, ofrecen las ventajas de los sistemas de autenticación centralizada pero avanzan más en el concepto ya que no se hace necesario pasar por un proceso de autenticación en cada servicio al que quiera acceder el usuario sino que solo se lleva a cabo una vez.

Para realizar esto, los sistemas SSO proveen una interfaz al usuario en donde este ingresa las credenciales de acceso y el sistema registra el inicio de sesión. Cuando el usuario intente acceder a un servicio en específico, este no le pedirá que escriba de nuevo los datos de acceso, sino que comprueba si la persona ha iniciado sesión o no y procede según sea el caso.

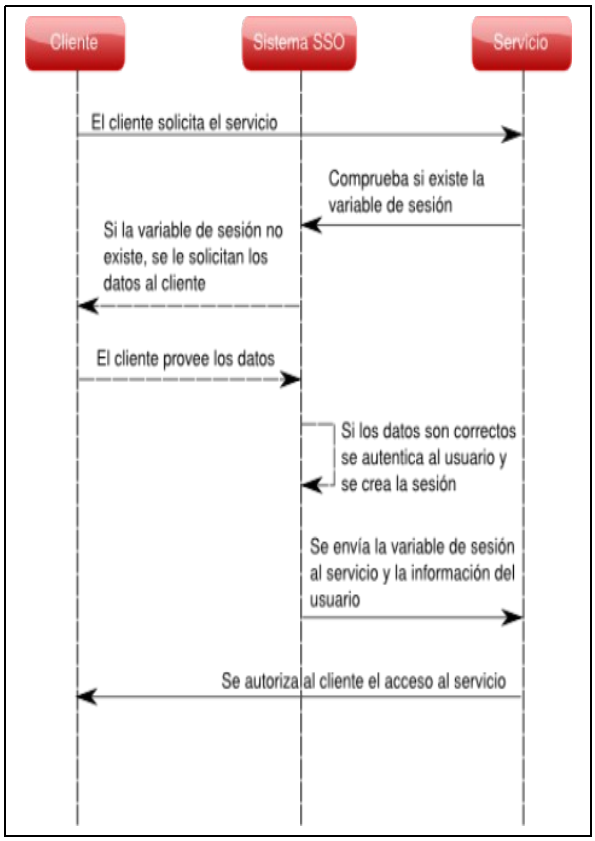

Gráfico No 3: Funcionamiento de un sistema SSO.

Un ejemplo de una unificación en la autenticación para el ingreso a los servicios es el que tiene Google. Esta unificación es del tipo SSO ya que con una sola cuenta de Google (nombre de usuario y contraseña) y realizando una única vez el proceso de autenticación se puede acceder a Gmail, Google Documents, Google Calendar, Google Reader, Google Wave, Blogger y muchos otros servicios ofrecidos por esta compañía. Además, cuando Google lanza un nuevo servicio, la misma cuenta existente funciona sin necesidad de tener que realizar un nuevo proceso de registro.

En el mercado hay diversas herramientas y mecanismos para implementar tanto un servidor de autenticación centralizada como un sistema SSO. Existen varios

métodos, tanto libres como privativos, para ser desplegados en diversas plataformas y con arquitecturas y niveles de seguridad diferentes.

Implementación de un sistema de autenticación centralizada

Una forma sencilla de realizar un sistema de autenticación centralizada, es implementando una base de datos centralizada en la que se almacene toda la información de los usuarios y sus respectivas contraseñas, las cuales serán usadas en todos los servicios. Este sistema es fácil de implementar pero al mismo tiempo es muy simple y por lo tanto no ofrece tantas ventajas.

Otra forma de realizar una autenticación centralizada, es implementando un **servicio de directorio** [4]. Este tipo de servicio puede verse como una analogía a lo que vendría siendo en la realidad los directorios telefónicos. Es un mecanismo que se puede utilizar para almacenar, de manera centralizada, un conjunto de información que tiene una estructura definida. Este servicio también recibe el nombre de “directorio electrónico” [4].

Implementación de un sistema SSO

Para la implementación de un sistema SSO, se puede desarrollar un software que utilice un sistema de autenticación centralizada como base y construirlo específicamente para suplir las necesidades actuales. La realización de este

tendría la ventaja de estar 100% integrado con los servicios de la UTB, pero se presentaría una gran desventaja y es el tiempo de desarrollo que demandaría un proyecto de esta magnitud.

Una solución más sencilla para la implantación de un sistema SSO es la utilización de algún protocolo o herramienta ya existente que ayude al despliegue de un servicio que provea esta funcionalidad. Estos sistemas proporcionarían todas las ventajas de la autenticación centralizada, además de permitir el acceso de los usuarios a los servicios que necesita sin la necesidad de escribir sus credenciales de acceso más de una vez [3].

Ventajas y desventajas

A continuación se muestra un cuadro que nombra las ventajas y desventajas que ocasionaría la adopción de cada una de estas dos alternativas.

Sistema	**Ventajas**	**Desventajas**
Sistema de autenticación centralizada	● Unifica en un solo servicio las credenciales de acceso de los usuarios. ● Facilidad en su implementación e inclusión de nuevos sistemas.	● Punto único de vulnerabilidad. ● Cada servicio que se conecte a este sistema tiene su propio nivel de seguridad.

	● Posibilidad de evolucionar a un sistema SSO.	
SSO	● Permite el acceso a varios servicios con un solo inicio de sesión. ● Incremento en la productividad de los usuarios. ● Simplificación en la administración.	● Punto único de vulnerabilidad. ● Aumento en la vulnerabilidad cuando el usuario abandona temporalmente el escritorio.

Tabla No 16: Comparación entre el sistema de autenticación centralizada y el SSO.

Como se puede observar en el cuadro, en términos generales el sistema de autenticación centralizada se enfoca más en su simplicidad de implementación y flexibilidad, mientras que el SSO se concentra más en la experiencia de usuario y su facilidad de uso.

Por otro lado, en ambos casos, se realiza una unificación de credenciales de acceso. Esto es una ventaja y una desventaja al mismo tiempo. Es una ventaja desde el punto de vista del usuario ya que este no necesita memorizar más que un solo par de datos (usuario y contraseña), pero se convierte en desventaja debido a que alguna persona puede enterarse de la clave de acceso de otra y así puede ingresar no a uno sino a todos los servicios de los que hace uso esta última.

La desventaja más crítica que presentan ambos sistemas es la de ser un punto único de vulnerabilidad. Como ambos sistemas plantean arquitecturas centralizadas, entonces necesitan de la implementación de mecanismos de tolerancia a fallos y de seguridad para prevenir posibles ataques de negación de servicios.

Todos estos inconvenientes pueden ser resueltos si se toman las medidas necesarias para evitarlos o en su defecto reducir su impacto. Y en contraposición, las ventajas que ofrecen son capaces de solucionar los problemas actuales de la universidad.

Sistemas candidatos

Tomando como base las dos alternativas de solución se pueden determinar las diferentes posibilidades para proceder en la implementación de una solución. Estas opciones son:

- Base de datos central.
- Servicio de directorio.
- Desarrollo de un sistema SSO teniendo como sistema de persistencia una de las alternativas de autenticación centralizada.
- Despliegue de un sistema SSO existente.

Debido a la gran cantidad de tiempo y esfuerzo que demandaría un proyecto de tan alta dimensión como sería el desarrollo propio de un sistema SSO, esta opción se descarta.

Por otro lado, el uso que se le daría al sistema final después de implementado demandaría una alta tasa de consultas permanentemente y solo un aumento en la tasa de modificaciones en los datos al iniciar cada semestre académico. Esto conlleva a que resulte más efectiva la implementación de un servicio de directorio que una base de datos central, debido a que el directorio está optimizado para soportar grandes cantidades de operaciones de lecturas y volúmenes modestos de modificaciones (aprox. 1000 datos en una transacción) sin incurrir en reducciones del rendimiento normal del sistema[1].

Además de lo expuesto previamente, el servicio de directorio no usa un modelo de datos relacional lo cual aumenta un poco su seguridad en el ambiente web evitando ataques del tipo SQL Injection. También, está soportado por una serie de protocolos que definen el funcionamiento estándar que debe tener un software de este tipo, por lo cual no debe existir ningún problema en caso de realizar una migración de un sistema a otro.

---

1 Datos encontrados en el capítulo 2 de la guía en línea *LDAP for Rocket Scientists,* licensiada bajo CC-BY-NC y escrita por el webmaster de zytrax.com. Se puede encontrar directamente en el siguiente enlace: http://www.zytrax.com/books/ldap/ch2/.

Por las razones expuestas, se descarta la opción de la base de datos central ya que un servicio de directorio se adapta más a las necesidades de la UTB, a las características del proyecto y provee ciertas ventajas adicionales que no proporciona un sistema tradicional de base de datos relacional.

Esto nos deja con dos opciones posibles de solución: servicio de directorio o sistema SSO. Como se ha dicho, ambas opciones son válidas para solucionar los problemas actuales de la UTB, pero hay que determinar cual de las dos se elije. Para esto, se procederá a determinar los posibles sistemas candidatos a usarse como solución de software, que pertenezcan a las dos alternativas mencionadas, y posteriormente realizarles pruebas para de esa forma elegir el sistema a implementar.

Los criterios iniciales para elegir el sistema a implementar son los siguientes:

- Buena documentación
- Plataforma para la que esté disponible
- Facilidad de despliegue
- Que sea software libre
- Que sea gratuito
- Facilidad de administración
- Beneficios que ofrezca en comparación con otras opciones
- Herramientas que ofrezca para facilitar su uso (adicional)

Servicio de directorio

En las páginas siguientes se van a explicar algunos conceptos básicos sobre los servicios de directorio en general. Al finalizar el capítulo se procederá a escoger las opciones candidatas a ser implementadas.

¿Qué es?

El término servicio de directorio se ha usado de manera ambigua para referirse a diversos elementos independientes: el sitio donde se almacena la información, los elementos hardware y software que ayudan a la administración de dicha información, las aplicaciones que usan la información (del lado del cliente o del servidor), etc. [4]

En este trabajo, el término servicio de directorio hace referencia a todo ese conjunto de elementos previamente mencionados, ya que no solo es necesaria la información contenida como tal, sino que esta pueda ser administrada, accedida por algún método y que esté disponible para su uso por otros sistemas en una misma red, porque como tal es un servicio.

Breve historia sobre los servicios de directorio

A partir del concepto de servicio de directorio se han creado varios protocolos para lograr una correcta implementación de un directorio. El primero que fue definido para la implementación de un servidor de directorio fue el X.500 el cual es un conjunto de protocolos basados en el modelo OSI y fue ideado inicialmente para soportar ciertos requisitos del protocolo X.400 [8].

Los protocolos que hacen parte de la familia de X.500 son DAP (*Directory Access Protocol*), DSP (*Directory System Protocol*), DISP (*Directory Information Shadowing Protocol*) y DOP (*Directory Operational Bindings Management Protocol*) [8].

Este protocolo (el X.500), debido a que fue basado en OSI, no llegó a ser completamente implementado. El protocolo perteneciente a esta familia que fue más afectado desde ese punto de vista fue DAP, debido a que era el que hacía las comunicaciones entre los clientes y el servicio de directorio. A raíz de esto, y de la popularidad que estaba ganando en ese tiempo TCP/IP, se creó una versión más ligera de DAP llamada LDAP (*Lightweight Directory Access Protocol*) [8].

Cuando fue creado, LDAP estaba pensado solo para ser una versión más ligera de DAP, que pudiera proveer toda la funcionalidad (o la mayoría) que ofrecía este,

y que se pudiera conectar al directorio por medio de TCP/IP sin necesidad de realizar cambios en este. Así se creo entonces LDAP v1.

Poco tiempo después de su creación, se decidió formalizar el protocolo LDAP como una alternativa ligera que proveía la funcionalidad de DAP. Se redactó el documento RFC 1777 con las especificaciones que debía cumplir una implementación de LDAP. Esta, aunque fue la primera versión formal, se llamo LDAP v2.

Actualmente, la familia de protocolos X.500 ha dado paso a LDAP como solución completa de software, y no solo para la conexión entre los clientes y el servidor como se hizo originalmente. Este, ha llegado a ser un protocolo que incluye todo el potencial de X.500 como servicio de directorio, pero que está basado en TCP/IP. A esto se le conoce como LDAP v3 y sus especificaciones están descritas en el documento RFC 4510[2].

Protocolo X.500

Es todo un conjunto de protocolos desarrollados con el objetivo de soportar los requisitos del protocolo X.400. Este hace uso de la pila de protocolos OSI, así que

2 Este documento es la versión más reciente y dejo obsoletos a otros escritos anteriormente. Una lista completa de todos los documentos de especificación de LDAP puede ser encontrada en el enlace bibliográfico [9]

su implementación se hace muy complicada y necesitaría una gran potencia computacional.

La especificación de este conjunto de protocolos define los siguientes elementos [10]:

- Un modelo de información para la estructura de esta en el directorio.
- Un espacio de nombres (*namespace*) que permite que la organización pueda ser organizada y referenciada.
- Un modelo funcional que determina las operaciones que pueden realizarse sobre la información.
- Un sistema de autenticación que agrega un poco de seguridad a la información del directorio.
- Un modelo de operaciones distribuidas que determina como es distribuida la información y como son llevadas a cabo las operaciones.

La unidad elemental de información son los registros, los cuales están compuestos de uno o más atributos. Cada atributo está compuesto por un tipo y uno o más valores (atributos multivalorados). El tipo determina la forma o sintaxis del valor o valores que componen el atributo [10].

Los registros se organizan jerárquicamente en forma de árbol. Cada registro tiene asignado un RDN o *Relative Distinguished Name*, el cual es un dato con la forma “nombre = valor”. A su vez, un registro también tiene asociado un DN o

Distinguished Name, el cual está compuesto por el RDN del mismo mas los RDN de los ancestros del registro referenciado, todos separados por coma (,). El DN identifica unívocamente a un elemento en la jerarquía del árbol de registros [10].

DAP

Para realizar operaciones sobre la información contenida en el directorio se definió un protocolo llamado DAP, también dentro de la especificación de X.500 (más precisamente en la X.511).

DAP son las siglas de *Directory Access Protocol* que traducen a Protocolo de Acceso a Directorio. En este protocolo se definen las siguientes operaciones [11]:

- Leer, la cual extrae los atributos de un registro cuyo nombre es conocido.
- Buscar, el cual selecciona un grupo de registros por medio de un filtro de búsqueda y retorna un grupo de atributos por cada registro que cumpla los criterios.
- Listar, que enumera los hijos de un registro.
- Comparar, que toma un DN, un tipo y un valor de un atributo, y comprueba si el registro con ese DN tiene esos datos en el atributo.
- Crear, la cual agrega nuevos registros a un directorio.
- Modificar, que permite modificar registros, y también añadir y eliminar atributos a estos.

- Borrar, la cual elimina registros del directorio.
- Modificar RDN, que permite modificar el RDN de un registro.
- Enlazar, que permite que un cliente inicie sesión.
- Desenlazar, que termina una sesión de directorio.
- Abandonar, que permite que una operación en proceso sea cancelada.

LDAP

Son las siglas de *Lightweight Directory Access Protocol*. Fue originalmente diseñado como un protocolo de red que permitía una forma alternativa para acceder a un servidor de directorio. En otras palabras, fue ideado como una alternativa a DAP, pero más ligera.

En la actualidad, la idea original ha evolucionado y ya es tratada como una extensión del conjunto de protocolos X.500 ya que acepta el mismo modelo de información y *namespaces*, pero este trabaja con la pila de protocolos TCP/IP.

En cuanto a la funcionalidad, conserva la mayoría de las funciones de DAP. Estas funcionalidades son: buscar, crear, borrar, modificar, modificar RDN, enlazar, desenlazar, abandonar y comparar. Leer y listar no están incluidas pero se emulan por medio de la operación buscar.

Desde la creación de LDAP hasta el día de hoy, este ha sufrido algunas variaciones, lo cual ha terminado en la creación de varias versiones del estándar, siendo la más reciente LDAPv3. A continuación se muestra un cuadro comparativo de las tres versiones de este protocolo.

Características	LDAPv1	LDAPv2	LDAPv3
Fecha de lanzamiento	Marzo de 1994.	Marzo de 1995.	Diciembre de 1997.
Documento de estandarización	No tuvo publicación.	RFC 1777.	RFC 4510.
Modelo de datos	Igual que el protocolo X.500.	Igual que el protocolo X.500.	Igual que el protocolo X.500.
Estructura de organización de los datos	Igual que el protocolo X.500.	Igual que el protocolo X.500.	Igual que el protocolo X.500.
Seguridad	Autenticación contra el directorio con usuario y contraseña de texto simple o con Kerberosv4.	Autenticación contra el directorio con usuario y contraseña de texto simple o con Kerberosv4.	Autenticación contra el directorio con usuario y contraseña de texto simple, autenticación por SASL o usando el protocolo SSL.
Tipos de respuestas del	Solo información.	Solo información.	Información o referencias a otros

servidor			servidores.

Tabla No 17: Versiones del protocolo LDAP.

Opciones candidatas

A partir de este protocolo (LDAP) se han desarrollado diversas implementaciones por parte de algunas empresas o fundaciones. El cuadro siguiente lista algunas de las implementaciones de este protocolo, así como la empresa o fundación detrás de esta y el tipo de licencia con que es distribuido [6]:

Software	**Empresa que lo desarrolla**	**Tipo de licencia**
Novell eDirectory	Novell, Inc.	Privativa
Red Hat Directory Server	Red Hat, Inc.	Libre (GPL)
Active Directory	Microsoft Corporation	Privativa
Open Directory	Apple Inc.	Privativa
Apache Directory Server	Apache Software Foundation	Libre (Apache License)
Oracle Internet Directory	Oracle Corporation	Privativa.
OpenDS	Sun Microsystems	Libre (CDDL)
OpenLDAP	OpenLDAP Foundation	Libre (OpenLDAP Public License)
IBM Tivoli Directory Server	IBM	Privativa

Tabla No 18: Diferentes implementaciones del protocolo LDAP.

Teniendo en cuenta la tabla previa, se puede extraer una serie de opciones candidatas de servicios de directorios que tienen la posibilidad de ser

implementados, tomando como base solo dos criterios de elección seleccionados: que sea software libre y que sea gratuito. Realizando la clasificación bajo estos lineamientos, las opciones elegibles son:

- Apache Directory Server
- OpenDS
- OpenLDAP

Estas tres son las candidatas preliminares de una de las alternativas de solución mencionadas: servicio de directorio. Ahora hay que analizar la otra alternativa, elegir otras opciones candidatas para confrontarlas con estas que recién se escogieron y finalmente elegir una que sirva como solución a la problemática que se está tratando.

SSO

Al igual que en el capítulo anterior, se procederá primeramente a exponer un poco la teoría sobre los sistemas SSO y al final de capítulo se escogerán las opciones candidatas más viables para ser implementadas.

¿Qué es?

SSO es una característica que pueden tener los sistemas de autenticación de usuarios, la cual permite otorgar autorización a los usuarios para que puedan hacer uso de múltiples, relacionados, pero independientes sistemas de software [3].

En otras palabras, permite que un usuario al momento de realizar el proceso de autenticación, solo ingrese al sistema la clave de acceso una sola vez y este le otorgue la respectiva autorización para hacer uso de los diferentes sistemas de software a los que este tenga permiso de acceder dependiendo de los alcances de su perfil.

Configuraciones comunes de SSO

Existen diferentes métodos para realizar un sistema SSO, entre los cuales se encuentran unos muy comunes los cuales son [3]:

1. Basados en Kerberos: estos sistemas están basados en el protocolo de autenticación en red, Kerberos. Esta configuración funciona solicitando al usuario que ingrese su datos de acceso al sistema y si los datos son correctos, se genera un *ticket-granting ticket* (TGT o tiquete generador de tiquetes). Este tiquete almacena la información del usuario y es el que dice si el usuario es realmente el que se identificó. Después cuando el usuario intenta acceder a una aplicación, esta no le solicita el reingreso de los datos sino que usa el *ticket-granting ticket* para solicitar un tiquete de servicio y comprobar la identidad del usuario.
2. Basados en tarjetas inteligentes: esta es una configuración que usa un sistema electrónico similar al que se encuentra en las *simcard* de los celulares, donde, si se tiene configurado, al iniciar el sistema se le solicita al usuario el ingreso de una clave la cual corresponde con la que está almacenada en la tarjeta inteligente. Si la clave ingresada coincide con la de la tarjeta, se le otorga al usuario el acceso al sistema y las aplicaciones posteriormente extraen la información de identidad de la tarjeta inteligente y no la solicitan nuevamente al usuario.
3. *OTP token*: son las siglas de one-time password (clave de una sola vez o clave de un solo uso), y es una configuración donde se usa un sistema de autenticación doble similar a la configuración basada en tarjetas inteligentes, pero en esta, el usuario no dispone de una clave estática sino

que usa una generada por algún método y que funciona para una sola sesión o transacción, dependiendo del caso.

Opciones candidatas

A continuación se muestra una tabla que contiene diferentes implementaciones de sistemas SSO con su respectiva empresa o fundación desarrolladora y el tipo de licencia bajo el que es distribuido [12]:

Software	**Empresa que lo desarrolla**	**Tipo de licencia**
CAS (Central Authentication Service)	JASIG (Java Architectures Special Interest Group)	Libre
OpenSSO	Sun Microsystems	Libre (CDDL)
JOSSO (Java Open Single-Sign On)	Atricore	Libre (LGPL)
CoSign	National Science Foundation Middleware Initiative	Libre

Tabla No 19: Implementaciones de sistemas SSO.

Al observar al contenido del cuadro, se puede concluir que no hay muchas implementaciones de sistemas SSO disponibles y no se puede realizar una selección a priori como se realizó con el servicio de directorio ya que todas las

opciones cumplen con algunas característica básicas iniciales, las cuales son: ser software libre, ser gratuitas y ser multiplataforma.

Por lo tanto, se van a elegir las cuatro opciones y se procederá a realizarles pruebas junto con las que se seleccionaron previamente de servicio de directorio para determinar el software a implementar que pueda darle solución a la problemática de la UTB.

Pruebas realizadas

Teniendo como base las opciones que se seleccionaron de las dos alternativas de solución previamente tratadas, se procedió a realizarles pruebas para evaluar su comportamiento inicial. Las pruebas consistieron en lo siguiente:

- Facilidad de instalación del sistema en diferentes sistemas operativos
- Facilidad de integración con los sistemas de información de la UTB

Para esto, se dispuso de un entorno de pruebas con las siguientes características:

- Hardware
- Procesador: Intel Pentium Dual-Core E5200 2,50 Ghz
- Memoria: 3GB
- Disco duro: 250 GB
- Teclado: Si
- Ratón: Si
- Pantalla: Si
- Software
- Sistema operativo: Windows XP Professional Edition y Ubuntu 9,10 Desktop Edition (arranque dual)
- Servidor web: Apache

- Motor de base de datos: MySQL
- Lenguaje de programación: PHP
- Servidor de *servlets*: Apache Tomcat 6.0
- Aplicaciones: Moodle 1.9, Drupal 6.16 y Apache Tomcat Manager

Las aplicaciones de Moodle y Drupal se ejecutan en el servidor web Apache, mientras que el Apache Tomcat Manager funciona en el servidor de *servlets* Apache Tomcat.

Pruebas a sistemas SSO

Inicialmente se procedió a realizarles pruebas a los sistemas SSO por ser las alternativas más cómodas a nivel de usuario final y por ofrecer lo necesario para solucionar el problema, más unas cuantas características adicionales.

La primera prueba realizada fue la de facilidad de instalación del sistema en diferentes sistemas operativos. Para esto, se procedió a realizar una instalación básica del sistema en los dos sistemas operativos usados y siguiendo los pasos indicados en la documentación oficial ofrecida en la página de cada uno. Para los cuatro sistemas SSO, los resultados arrojados fueron los siguientes:

- CAS (versión 3.3.5): se logró hacer una instalación exitosa y de manera sencilla en ambos sistemas operativos. La instalación de este software se llevó a cabo en el servidor Apache Tomcat.
- OpenSSO (versión 8.0): no se logró realizar una instalación de este sistema en ninguno de los dos sistemas operativos. El proceso de instalación iniciaba bien pero se interrumpía por un error desconocido. Esto se llevo a cabo en el servidor Apache Tomcat.
- JOSSO (versión 1.8.1): la instalación de este sistema se logró realizar bien en Windows XP, pero después de la instalación la aplicación de Apache Tomcat Manager entraba en conflicto con este sistema así que fue necesario realizar configuraciones adicionales para que volviera a funcionar. En el sistema operativo Ubuntu no se pudo realizar la instalación de este sistema. Este proceso también se llevo a cabo en el servidor Apache Tomcat.
- CoSign (versión 3.1.1): este sistema no se logró instalar ni en Windows XP ni en Ubuntu. La instalación de este se intentó llevar a cabo en el servidor web Apache.

Además de las pruebas realizadas, existen inconvenientes con dos de las opciones candidatas: OpenSSO y CoSign. OpenSSO era un proyecto amparado por Sun Microsystems, pero al haber sido adquirida por Oracle Corporation, esta última decidió interrumpir el financiamiento de este proyecto, por lo cual este sistema no va a lanzar nuevas versiones en un futuro.

El problema con CoSign es que cuenta con una documentación desactualizada. Al momento de redactar este documento, la última actualización realizada a la página principal de la wiki de CoSign es del 6 de abril de 2009. Además, la documentación también es muy escasa y se concentra principalmente en instalación del sistema y solución de problemas, más no entra en detalles más específicos de lo que se puede hacer con el software.

Teniendo en cuenta los resultados de esta primera prueba, es evidente que solo CAS ha logrado ser instalado sin ningún inconveniente en ambos sistemas operativos. Por lo tanto a este sistema se le realizó una segunda prueba para medir si cumple con los requisitos mínimos necesarios para ser considerado como una opción viable a ser implementada.

La segunda prueba consiste en como se integra este sistema con los software usados en la UTB, tomando como base Drupal y Moodle, siendo estos los más usados por la comunidad. Drupal por ser el CMS usado en la construcción de la página web institucional y la página web de UTBVirtual, y Moodle por ser la plataforma virtual de aprendizaje utilizada como núcleo de SAVIO.

Para realizar esta integración, CAS actúa como servidor y tanto Moodle como Drupal actúan como clientes. Estos dos sistemas disponen cada uno de un módulo para conectarse a CAS. Ambos módulos están basados en un cliente genérico de PHP para CAS llamado phpCAS.

Primero se procedió a hacer la configuración de CAS en Moodle. Primero hay que especificarle a Moodle que use como método de autenticación a CAS. Para esto, se realiza lo siguiente:

1. Como usuario administrativo de Moodle, se hace clic en el menú “Usuarios”, luego en “Autenticación” y después en “Gestionar autenticación”.
2. En la página que se muestra, se hace clic en el botón “Habilitar” que se encuentra a la derecha de la opción “Usar un servidor CAS (SSO)”.

Luego de realizar esto, hay que proceder con la configuración de Moodle para que pueda reconocer el servidor CAS y usarlo como método de autenticación. La siguiente tabla muestra los campos que componen el formulario de configuración de CAS en Moodle, un ejemplo del dato que puede contener y la configuración realizada:

Campo	**Breve descripción**	**Ejemplo de configuración**	**Configuración realizada**
Nombre del host	Aquí se especifíca la URL del servidor CAS.	https://localhost	https://localhost
Base URI	URI del servidor, o en blanco si el servidor no tiene URI.	En caso de encontrarse el servidor en https://localhost/CAS/, entonces	(en blanco)

		la URI sería “CAS”.	
Puerto	Este es un número que indica por que puerto se puede comunicar con el servidor.	1234	8080
Versión	Un número que indica la versión del servidor CAS.	3.0	3.3.5
Idioma	Aquí se especifica el idioma. Este cuadro de despliegue con varias opciones.	Inglés	Inglés
Modo proxy	Este es un campo de selección “Si” o “No”. Se coloca “Si” en caso de usar CAS en modo proxy.	No	No
Salir del CAS	Otro campo de opción “Si” o “No”. Este indica si al cerrar sesión se cierra solo la de Moodle o también la de CAS.	Si	No
Multi-autenticación	Campo con opciones “Si” o “No”. Se elige esta opción en “Si” en caso de querer usar CAS más otro método de autenticación.	Si	No

Tabla No 20: Configuración de CAS en Moodle.

En el mismo formulario hay más opciones de configuración, pero estas hacen parte de la configuración de un servidor LDAP, en caso de usarse alguno como *back-end* para CAS.

Después de haberse realizado la configuración respectiva en Moodle para que funcionara con CAS, Moodle avisó de un error de incompatibilidad entre versiones. El cliente de PHP que está integrado a Moodle soporta conexiones con servidores CAS hasta versiones 2.0.x y se está usando para la prueba la versión 3.3.5.

Se prosiguió a realizar la prueba con Drupal. En este caso, fue necesario descargar e instalarle el módulo a Drupal ya que este no viene instalado de manera predeterminada[3]. Con este módulo también se pretende realizar la autenticación por medio de CAS y no por el formulario normal de Drupal, tal como se hizo con Moodle.

Después de instalado el módulo en Drupal, se procedió a realizar la configuración para que Drupal usara un servidor LDAP como método de autenticación. Para ello, se siguieron los siguientes pasos:

1. Como usuario administrador de Drupal, se hace clic en "*Administer*", luego en "Site building" y por último en "*Modules*". Todo esto hace parte del menú de navegación del administrador.

3 Este módulo puede ser descargado de http://drupal.org/project/cas.

2. Ahora aparece una página con una lista de funcionalidades. Se busca la que diga “cas”, se activa (se hace clic sobre el checkbox que tiene) y después se hace clic en el botón “*Save configuration*” para guardar los cambios.

Además de la descarga del módulo para Drupal, también fue necesaria la descarga de la librería que maneja CAS en PHP llamada phpCAS, ya que esta no viene incluida dentro del paquete. Esta librería debe colocarse dentro de la carpeta que contiene el módulo.

Inmediatamente después de realizada esta instalación, se procedió a la configuración del módulo para que pudiera conectarse al servidor CAS. A continuación se muestra una tabla que indica los campos que contiene el formulario de configuración de CAS en Drupal, así como los posibles datos que este puede contener y la configuración aplicada:

Campo	**Breve descripción**	**Ejemplo de configuración**	**Configuración realizada**
CAS version	La versión del servidor CAS al cual se va a conectar. Este campo solo tiene las opciones “1.0” y “2.0 or higher”.	1.0	2.0 or higher
CAS server	URL del servidor donde se	http://192.168.0.15	localhost

	aloja CAS.		
CAS port	Número del puerto por el que escucha el servidor CAS.	443	8080
CAS URI	URI para acceder al servidor CAS en caso de no encontrarse en la raíz. Puede dejarse vacío.	CAS	(Vacío)
CAS PEM certificate verification	Campo con tres opciones para la verificación del certificado del servidor. Las opciones son: Do not verify the certificate, Verify the server using PEM certificate, Verify the Certificate Authority using PEM certificate.	Verify the Certificate Authority using PEM certificate	Do not verify the certificate
CAS PEM Certificate (phpCAS 0.6 or greater)	En este campo se especifica el certificado en caso de querer validarse. Puede dejarse vacío.	(Vacío)	(Vacío)
CAS PGT storage file format	Campo con dos opciones para indicar el formato del archivo del almacenamiento PGT. Las opciones son: Plain text, XML.	XML	Plain text
CAS PGT	URL para indicar el sitio del	(Vacío)	(Vacío)

storage path	almacenamiento PGT. También se puede dejar vacío y solo se coloca algo en caso de usar CAS como proxy.		
CAS debugging output filename	Nombre del archivo que guardará las salidas de depuración. Puede dejarse vacío en caso de no querer almacenar esa información.	logs_cas.txt	(Vacío)

Tabla No 21: Configuración de CAS en Drupal.

Habiendo realizado la configuración respectiva, se procedió a probar si esta funcionaba, pero arrojó el mismo resultado de Moodle: incompatibilidad con la versión 3.3.5 de CAS.

Como resultado de estas pruebas, quedan solo dos opciones para proceder: usar una versión más antigua de CAS que pueda ser soportada por Moodle y Drupal o descartarla como solución candidata a ser implementada. Teniendo en cuenta las necesidades del sistema, se optó por descartarla como solución candidata por las siguientes razones:

- De la versión 2.0.6 (última versión estable de la rama 2.0.x) a la versión 3.0 (primera versión estable de la rama 3.x) hubo muchos cambios en la

seguridad y en la arquitectura del sistema. Por lo tanto, la versión 2.0.6 puede ser considerada insegura y obsoleta, y uno de los factores críticos de este sistema es la seguridad.

- La documentación que existe actualmente sobre CAS es para versiones de la rama 3.x así que no es útil para versiones anteriores a la 3.0.
- El desarrollo de un cliente en PHP de CAS que soporte las versiones más recientes implicaría la realización de un desarrollo que necesitaría de un proceso de software para llevarse a cabo y eso está por fuera del alcance de este trabajo.

Además de estas razones, también hay que tener en cuenta que hay otras soluciones candidatas por evaluar que no pertenecen a la alternativa de SSO sino que son de autenticación centralizada. Las soluciones candidatas son: Apache Directory Server, OpenLDAP y OpenDS. Estas, aunque no ofrezcan tantas prestaciones como las soluciones SSO, siguen siendo alternativas de solución válidas para la problemática de la UTB.

Pruebas a servicios de directorio

Teniendo en cuenta las tres opciones de solución que se plantearon de servicio de directorio se procedió a realizarles las mismas pruebas que se le realizaron a las opciones de SSO: instalación, integración con las aplicaciones (Moodle y Drupal) y facilidad de administración que no se alcanzó a realizar previamente.

Al igual que con las opciones de SSO, se procedió a realizar una instalación de cada uno de los servicios de directorio tanto en Windows XP como en Ubuntu. Los resultados de las instalaciones de cada sistema son las siguientes:

- Apache Directory Server (versión 1.5.6): se logró realizar una instalación de este servicio de directorio en ambos sistemas operativos sin ningún problema.
- OpenLDAP (versión 2.4.21): se logró realizar una instalación exitosa de este servicio de directorio en los dos sistemas operativos. Cabe anotar que el instalador de OpenLDAP para Windows no está disponible para su descarga en la página oficial del proyecto sino que se encuentra de manera no oficial para su descarga en el siguiente sitio web: http://www.userbooster.de/en/download/openldap-for-windows.aspx.
- OpenDS (versión 2.2.0): la instalación de este servicio de directorio se pudo llevar a cabo exitosamente en los dos sistemas operativos.

Debido a que el resultado de la primera prueba fue exitoso para dos de los tres servicios de directorio, se procedió a realizar la segunda prueba a los sistemas restantes. Para esta segunda prueba también se utilizaron las aplicaciones Moodle y Drupal. Se procedió a realizar la configuración de ambos sistemas para que no realizaran la autenticación como la hacen normalmente sino que autenticaran usando un servicio de directorio.

Primero se procedió a realizar la prueba con Moodle. Esta aplicación trae integrado un módulo de autenticación usando LDAP pero dicho módulo no viene activado de manera predeterminada. Para activar el módulo se siguen los siguientes pasos:

1. Como usuario administrativo de Moodle, se hace clic en el menú “Usuarios”, luego en “Autenticación” y después en “Gestionar autenticación”.
2. En la página que se muestra, se hace clic en el botón “Habilitar” que se encuentra a la derecha de la opción “Usar un servidor LDAP”.

Después de realizada la activación de LDAP en Moodle, se procedió a configurar el módulo para que usara cada uno de los sistemas de directorio que se estaban sometiendo a pruebas, uno a la vez. Para ver una descripción detallada del significado de cada campo del formulario de configuración de LDAP en Moodle, puede revisar el anexo B.

La configuración de los dos servicios de directorio se llevó a cabo con los siguientes valores para cada uno de los campos:

Campo	ApacheDS	OpenDS
Ajustes del servidor LDAP		
URL del host	ldap://localhost:10389	ldap://localhost:1389
Versión	3	3

Codificación LDAP	utf-8	utf-8
Fijar ajustes		
Ocultar contraseñas	Si	
Nombre distinguido	uid=admin,ou=system	cn=Manager
Contraseña		
Ajustes de búsqueda de usuario		
Tipo de usuario	Por defecto	Por defecto
Contextos	ou=users,o=Moodle	ou=users,o=Moodle
Buscar subcontextos	No	No
Alias de referencia	No	No
Atributo de usuario	uid	cn
Atributo de miembro	(Vacío)	(Vacío)
Clase de objetos	(Vacío)	(Vacío)
Forzar cambio de contraseña		
Forzar cambio de contraseña	No	No
Utilizar página de cambio de contraseña estándar	Si	Si
Formato de contraseña	SHA-1 hash	Texto plano
URL para cambio de contraseña	(Vacío)	(Vacío)
Ajustes de caducidad de la contraseña LDAP		
Expiración	No	No
Advertencia de expiración	(Vacío)	(Vacío)
Atributo de expiración	(Vacío)	(Vacío)
Entradas libres	No	No
Atributo de entrada libre	(Vacío)	(Vacío)
Habilitar creación por parte del usuario		
Crear usuarios externamente	No	No
Contexto para usuarios nuevos	(Vacío)	(Vacío)
Creador de curso		
Creadores	(Vacío)	(Vacío)
Script de sincronización del Cron		
Usuario externo eliminado	Mantener interna	Mantener interna
NTLM SSO		

Habilitar	No	No
Sub-red	(Vacío)	(Vacío)
MS IE fast path?	No	No
Mapeado de datos		
Nombre	givenName	cn
Apellido	sn	sn
Dirección de correo	mail	mail
Ciudad	(Vacío)	(Vacío)
País	(Vacío)	(Vacío)
Idioma	(Vacío)	(Vacío)
Descripción	(Vacío)	(Vacío)
Página web	(Vacío)	(Vacío)
Número de ID	(Vacío)	(Vacío)
Institución	(Vacío)	(Vacío)
Departamento	(Vacío)	(Vacío)
Teléfono 1	(Vacío)	(Vacío)
Teléfono 2	(Vacío)	(Vacío)
Dirección	(Vacío)	(Vacío)

Tabla No 22: Configuraciones realizadas de ApacheDS y OpenDS en Moodle.

Para los dos servicios de directorio la configuración realizada en Moodle funcionó sin problemas. Los usuarios de prueba registrados en ambos servicios de directorio realizaron el proceso de autenticación exitosamente en Moodle con la información que tenían registrada en el LDAP.

Ahora, se procedió a realizar la prueba con Drupal para que usara un servicio de directorio como método de autenticación. Para ello, Drupal dispone de un módulo que se puede configurar para realizar la autenticación por medio de LDAP. Dicho

módulo no viene instalado por defecto en Drupal así que fue necesario descargarlo para poder realizar la prueba[4].

Después de descargado, se instaló el módulo de LDAP en Drupal y se procedió a realizar la activación de este. Para la activación de este módulo se realizó el siguiente procedimiento:

1. Como usuario administrativo de Drupal, se hace clic en "*Administer*", luego en "*Site building*" y por último en "*Modules*". Todas estas opciones están en el menú de navegación del administrador.
2. En la página que aparece con la lista de módulos disponibles para Drupal, en una sección llamada "*LDAP integration*" se activa la opción llamada "*Authentication*" y se guardan los cambios.

Teniendo el módulo activado, se procedió a realizar la configuración para que usara los servicios de directorio que se están evaluando, uno a la vez. La configuración de este módulo se hace en dos partes: autenticación y servidor. La primera parte está en la sección "*Settings*". Puede encontrarse una descripción detallada de cada campo del formulario de configuración general en el anexo C.

La segunda parte de la configuración se realiza en la sección "*Add server*". Se pueden configurar varios servidores LDAP para que funcionen al mismo tiempo, pero en las pruebas se trabajo con cada uno independientemente. Si se desea,

4 Este módulo puede ser descargado en la página http://drupal.org/project/ldap_integration.

puede encontrarse una descripción detallada de los campos de este formulario en el anexo D.

Tomando como base a estos formularios se procedió a realizar la configuración. El primer formulario solo lleva una configuración que es común a los dos servidores. El segundo formulario si se configuro independientemente para cada servidor. La configuración del primer formulario es la siguiente:

Campo	Valor
Authentication mode	
Choose authentication mode	LDAP directory only
Choose user conflict resolve procedure	Disallow login and log the conflict
Security options	
Do not store users passwords during sessions	Activo (marcado)
Sync LDAP password with the Drupal password	Inactivo (sin marcar)
LDAP UI options	
Remove password change fields from user edit form	Activo (marcado)
Alter email field on user edit form	Disable email field on form

Tabla No 23: Configuraciones generales de LDAP en Drupal.

Como se explicó previamente, esta configuración funciona para los dos servicios de directorio. Para el segundo formulario, se realizaron configuraciones independientes. Los valores de configuración usados son:

Campo	ApacheDS	OpenDS
Server settings		
Name	apacheds	opends
LDAP server	localhost	localhost
LDAP port	10389	1389
Use Start-TLS	Inactivo (sin marcar)	Inactivo (sin marcar)
Store passwords in encrypted form	Inactivo (sin marcar)	Inactivo (sin marcar)
Login procedure		
Base DNs	ou=users,o=Drupal	ou=users,o=Drupal
UserName attribute	uid	cn
Email attribute	mail	mail
PHP to transform login name	(Vacío)	(Vacío)
PHP to filter users based on their LDAP data	(Vacío)	(Vacío)
Advanced configuration		
DN for non-anonymous search	uid=admin,ou=system	cn=Manager
Password for non-anonymous search		

Tabla No 24: Configuraciones realizadas de ApacheDS y OpenDS en Drupal.

Como resultado de cada configuración aplicada, la prueba de autenticación en Drupal usando los dos servidores LDAP fue exitosa en cada caso.

Resultado de las pruebas y elección de la solución

Como resultado de las pruebas se puede concluir que solo dos de los sistemas candidatos han podido satisfacer las condiciones especificadas al principio del capítulo para ser consideradas opciones viables a ser implementadas. Estos sistemas fueron: ApacheDS y OpenDS.

Para seleccionar una solución de entre estas dos opciones, se tomaron en cuenta los criterios dichos al final del capítulo 2. Dos de estos criterios se habían tenido en cuenta para evaluar para la selección de los sistemas candidatos iniciales por cada alternativa: que la solución sea software libre y que se pueda adquirir de manera gratuita. Además, se comprobó un tercer criterio al realizar las pruebas a las opciones candidatas, el cual fue la capacidad de despliegue de dicha solución en diferentes sistemas operativos. Ahora hay que evaluar las aplicaciones candidatas finales que se tienen para elegir una que se implemente como solución.

Tomando el criterio de la buena documentación, ambos sistemas cuentan con una página web oficial del proyecto. En la página oficial de ApacheDS la documentación está dividida en tres categorías: usuarios básicos, usuarios avanzados y desarrolladores. En cada una se abarcan las tareas que haría cada tipo de usuario y ejemplos de como realizarlas. La página de OpenDS cuenta con una Wiki muy completa sobre los aspectos importantes del sistema y también está dividida en una sección para usuarios y otra para desarrolladores. Están explicadas paso a paso las tareas que realizaría cualquier usuario. Por lo tanto ambas herramientas proveen la documentación necesaria para su uso.

Desde el punto de vista de la facilidad de despliegue**,** como se dijo previamente, ambos sistemas son multiplataforma. ApacheDS cuenta con instaladores fáciles de usar para diversos sistemas operativos, entre los cuales se encuentran Linux,

Solaris, Mac OS X y Windows; además de proveer el código fuente en caso que sea necesaria su compilación. En el caso de OpenDS, existen dos opciones de instalación: una es por medio de Internet usando un instalador que se ejecuta gracias a *Java Web Start* e instala el sistema localmente en el escritorio, y la otra es descargarse un archivo comprimido y seguir las instrucciones de instalación. En este caso, ApacheDS ofrece más alternativas de instalación y mucho más fáciles de usar en comparación con OpenDS que ofrece principalmente el instalador web que tiene el inconveniente de necesitar una conexión a internet para realizar debidamente el proceso de instalación.

Analizando el criterio de la facilidad de administración, vemos que cada aplicación ofrece una herramienta administrativa. ApacheDS cuenta con una herramienta bastante completa para la gestión del directorio llamada A*pache Directory Studio* la cual está basada en el entorno de desarrollo Eclipse. Por su parte, OpenDS cuenta con una serie de herramientas propias que solo funcionan con su sistema. Por ser *Apache Directory Studio* también un software libre, permite más libertad e incluso puede usarse esta misma herramienta para administrar OpenDS, mientras que no se puede el proceso inverso con las herramientas de OpenDS para configurar ApacheDS. Por lo tanto, en este criterio y también en el de la disponibilidad de herramientas para la facilidad de uso, ofrece más ventajas la opción de ApacheDS.

Para terminar, tomando el criterio faltante que son los beneficios que ofrezca una aplicación que no tengan las demás, ambas aplicaciones tienen casi las mismas

prestaciones. Ambos tienen herramientas administrativas, trabajan bajo protocolo LDAP y pueden usar encriptación SSL por medio del protocolo LDAPS; disponen de múltiples *schemas* y se pueden crear personalizados; soportan referencias y replicación; diversos métodos de autenticación, organización jerárquica de la información, etc.

Hay un último detalle en cuando OpenDS. Este es un proyecto patrocinado originalmente por Sun Microsystems, empresa que fue recientemente adquirida por Oracle. Por lo tanto, este es un proyecto que en cualquier momento puede perder el soporte y quedar abandonado al igual que puede ocurrir con OpenSSO. Esto ocasiona que no sea una opción muy segura ya que se podría quedar implementado en la UTB un sistema que no va a mejorar con los años, sino que se va a volver obsoleto con el tiempo.

Como resultado de este análisis se puede concluir que la mejor opción a implementar que puede resolver la problemática que se está tratando de solucionar es ApacheDS.

Implementación del sistema

Para la implementación final del sistema, como ya se mencionó, se va a utilizar el servidor de directorio ApacheDS. Para la instalación de este software, se necesitan cumplir los siguientes requisitos:

- JRE (*Java Runtime Environment*) 5.0 o superior
- En caso de usar sistema operativo Linux, tener una interfaz gráfica X11
- 384 MB para la máquina virtual de Java

A continuación se describirá la estructura general de todo el sistema así como las configuraciones realizadas al servicio de directorio.

Arquitectura del sistema

El sistema cuenta con una arquitectura centralizada, teniendo como nodo central de todo, al servicio de directorio. A continuación se ilustra una imagen de la arquitectura general del sistema con sus respectivos servicios.

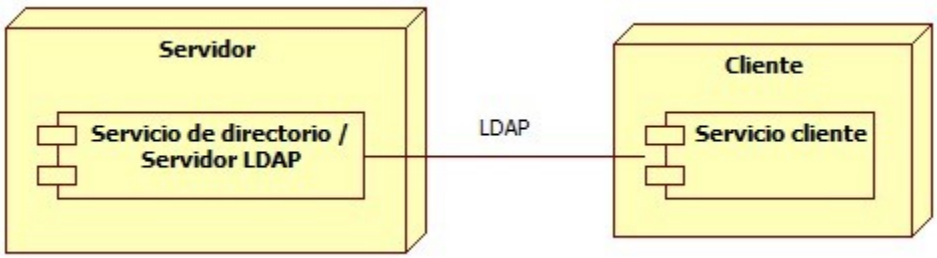

Gráfico No 4: Arquitectura general del sistema.

Como se puede apreciar en la imagen, los sistemas de información se comunican con el directorio realizando peticiones. Dichas peticiones se llevan a cabo solo en el momento en que un usuario intenta acceder a dicho sistema y después de recibida una respuesta se cierra la conexión con el directorio.

La comunicación con el directorio se puede realizar por medio del protocolo LDAP o LDAPS, pero por motivos de seguridad, para la implementación se va a utilizar el segundo por proveer una capa de encriptación para los datos durante la transmisión de estos.

El servicio de directorio se estará ejecutando en un servidor con sistema operativo Linux, el cual tendrá abiertos solo los puertos necesarios para el uso y la administración del mismo. Más específicamente el puerto 22 para el servicio de SSH de transmisión de archivos y el puerto 10636 para el protocolo LDAPS.

Estructura de la información

Para la implementación de este proyecto, se usó una estructura general para la información, tratando de no usar funcionalidades específicas de la herramienta, para que, en caso de cambiar de herramienta en un proyecto futuro, no existan muchos inconvenientes al migrar de un sistema a otro.

La estructura que se muestra a continuación es una propuesta que realiza el autor del libro *Mastering OpenLDAP* [20], la cual propone un modelo de división de los sistemas de información en tres subramas diferentes: *system* (sistemas), *groups* (grupos de usuarios) y *users* (usuarios).

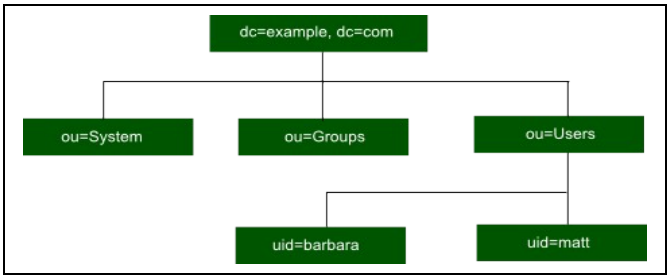

Gráfico No 5: Estructura recomendada de la información.

Esta es una estructura que se aplica más a entornos empresariales donde están debidamente delimitados unos grupos o roles bajo los cuales puedan clasificarse los usuarios. El problema que surge al tratar de aplicar esta estructura a la problemática de la UTB es que no existen unos grupos definidos en la UTB ya que la plataforma de servicios es muy heterogénea: docentes, estudiantes, administradores, directores de programa, administrador de proyectos, creador de cursos, etc.

Dado que no se puede aplicar en su totalidad la estructura recomendada en la literatura, entonces se plantea otra alternativa de solución la cual se amolda mejor a la situación actual de los sistemas de la UTB. La siguiente imagen ilustra como se estructuraría la información, basado en la estructura recomendada que se mostró previamente.

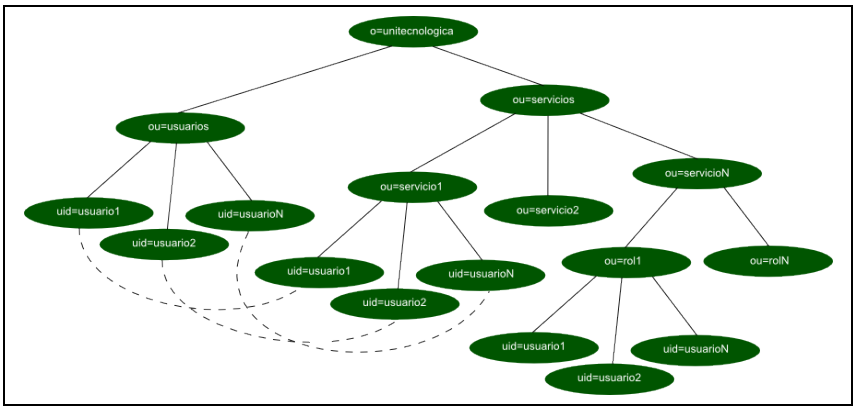

Gráfico No 6: Estructura de la información a implementar.

En esta estructura se observa una división entre los usuarios y los servicios. En la rama de los usuarios, se almacena la información perteneciente a todos los usuarios independientemente del servicio del que hagan uso, o del rol que puedan ejercer en dicho servicio.

En el subárbol de los servicios, se subdivide en una rama por cada servicio, donde estos pueden almacenar referencias a usuarios de la rama de usuarios directamente debajo en la jerarquía, o pueden subdividirse por roles los cuales son los que tendrían debajo las referencias almacenadas, dependiendo del rol al que pertenezca el usuario referenciado en dicho servicio.

Debido a que aquí en la UTB, en los sistemas de información en los que se quiere aplicar el servicio de directorio, cada uno tiene sus propios roles de usuario definidos o incluso tienen un solo tipo de usuario, entonces no se puede establecer una serie de roles de usuarios específicos. Por lo tanto, no se puede realizar una rama específica para almacenar los roles.

En el modelo propuesto, debajo de cada servicio, en caso de existir roles de usuario, se crean ramas con estos para diferenciar los tipos de usuarios. Si el servicio solo tiene un tipo de usuario, entonces no se crea ningún rol debajo de este sino que únicamente almacena los usuarios.

Utilizando este esquema se obtienen dos ventajas principales: el mantener siempre información completa y actualizada; y la independencia de la información delimitada por servicios, aumentando la seguridad sobre esta. Siempre se tendrá información actualizada porque como cada rama almacenaría un alias de la rama principal de usuarios, entonces solo se hace necesaria la actualización de esa información y se vería automáticamente reflejada sobre los demás servicios.

También se tiene la independencia de la información por servicios, ya que cada uno de estos accedería solo a la información que tiene permitida en su subrama y no tiene en cuenta que datos pueden estar almacenados para los otros.

Políticas de uso y administración

Con la creación o adquisición, y la consecuente implantación de un sistema de información en una empresa, se hace necesario establecer unas reglas o políticas que entrarán a regular el uso de dicha aplicación para que no se incurran en problemas de mala segregación de funciones, abuso de autoridad y demás.

La lista siguiente especifica las políticas que regirán el uso del sistema de autenticación centralizado a implementarse en la Universidad Tecnológica de Bolívar. Estas políticas hacen referencia a la administración, seguridad e integridad del sistema.

1. Solo existirá un usuario como *administrador del sistema*. Este tiene permisos para realizar configuraciones al servidor, consultar y modificar las entradas del directorio, crear nuevos usuarios y otorgarle los respectivos permisos a estos.
2. El *administrador del sistema* usará un software llamado Apache Directory Studio para realizar tareas administrativas.
3. Existirá uno o más usuarios *consultores,* los cuales solo pueden realizar labores de búsqueda de registros almacenados. No pueden realizar ninguna modificación.

4. Los usuarios consultores pueden usar el mismo software que el *administrador del sistema* o cualquier otra herramienta que les permita conectarse al servidor y hacer consultas.
5. El servidor de manera predeterminada permite el acceso anónimo por parte de cualquier cliente. Dicho acceso debe estar permanentemente desactivado para evitar posibles intrusiones al sistema.
6. Las conexiones con el directorio van a ser cifradas usando SSL.
7. Las claves de todos los usuarios estarán almacenadas usando la función hash SHA.
8. Todos los usuarios van a estar registrados bajo el grupo *usuarios*. Los registros de este grupo no van a ser borrados.
9. Cada sistema que se conecte al servidor, tiene asignada una rama en la jerarquía debajo del nodo *servicios* en la cual están almacenados unos *alias* que hacen referencia a un registro de un usuario que se encuentra en *usuarios*.
10. Para eliminar un usuario de un sistema, no se borra el registro en *users* sino que se borra el alias en el grupo que tiene asignado el sistema.

Estas son unas reglas básicas que se establecen para asegurar unos buenos niveles de seguridad, una correcta segregación de funciones e integridad de la información.

Configuraciones finales

La puesta en funcionamiento del servidor requirió de la instalación del software que va a funcionar como servicio de directorio que se seleccionó previamente: ApacheDS. Para esto, se descargó de la página oficial de ApacheDS el paquete que cumplía con las especificaciones de procesador y sistema operativo. Después, se procedió a copiar al servidor el paquete por medio de conexión SSH.

Teniendo el paquete en el servidor se inició el proceso de instalación del servicio. Para esto, usando la consola de comandos se realizó una conexión con el servidor también por medio de SSH y estando ubicado dentro de la carpeta en la que se encuentra el paquete se ejecutó el siguiente comando:

sudo dpkg -i apacheds.deb

Donde apacheds.deb es el nombre que tiene el archivo del paquete. Hay que agregar que si no se ejecutaba el comando desde la carpeta en la que estaba el archivo, había que anteponer al nombre de este la ruta (absoluta o relativa) hasta la carpeta donde se encontraba. Ejemplo: /home/sebastian/Descargas/apacheds.deb.

Al ejecutar dicha línea de comando, se realizó la instalación de ApacheDS. Ya con esto se tiene el servicio funcionando y publicado por el puerto que establece por

defecto desde el inicio. Ahora hay que hacerle las configuraciones necesarias para conseguir que funcione como se quiere.

Las configuraciones que se quieren realizar son para cumplir los siguientes objetivos:

1. Crear el nodo base o raíz de la estructura que se explicó previamente en este mismo capítulo para el almacenamiento de los datos.
2. Habilitar la comunicación con el servidor por medio del protocolo LDAPS.
3. Desactivar el acceso anónimo al servicio.
4. Cambiar la clave que trae el software por defecto para el administrador del sistema por una clave más segura.
5. Crear un usuario consultor y otorgarle solo los permisos de acceso más no modificación o eliminación.

Para alcanzar los tres primeros objetivos se deben realizar unas modificaciones a un archivo de configuración que se instala junto con el servidor el cual se llama *server.xml* y que al instalarse en Ubuntu se ubica por defecto en la carpeta /var/lib/ apacheds-1.5.6/default/conf. Las secciones que se necesitaron modificar de dicho archivo son las siguientes:

```
...
<defaultDirectoryService id="directoryService" instanceId="default" replicaId="1" workingDirectory="example.com" allowAnonymousAccess="false" accessControlEnabled="true" denormalizeOpAttrsEnabled="false" syncPeriodMillis="15000" maxPDUSize="2000000">
        <systemPartition>
...
```

```
        <partitions>
        <!-- NOTE: when specifying new partitions you need not include those -->
        <!-- attributes below with OID's which are the system indices, if left -->
        <!-- out they will be automatically configured for you with defaults. -->
                <jdbmPartition id="example" cacheSize="100" suffix="o=unitecnologica"
optimizerEnabled="true" syncOnWrite="true">
                <indexedAttributes>
...
        <ldapServer id="ldapServer" allowAnonymousAccess="false" saslHost="ldap.example.com"
saslPrincipal="ldap/ldap.example.com@EXAMPLE.COM" searchBaseDn="ou=users,ou=system"
maxTimeLimit="15000" maxSizeLimit="1000">
                <transports>
                        <tcpTransport address="0.0.0.0" port="10389" nbThreads="8"
backLog="50" enableSSL="false"/>
                        <tcpTransport address="0.0.0.0" port="10636" nbThreads="8"
backLog="50" enableSSL="true"/>
                </transports>

...
```

Se puede consultar el contenido completo del archivo después de las configuraciones realizadas en el anexo E. Con estas modificaciones se desactivó el acceso anónimo al servicio, se activó la comunicación por medio de SSL (LDAPS), se creó el nodo raíz para almacenar la información de los usuarios y además se preparó el servicio para restringirle los permisos de modificación a los usuarios consultores (segunda línea en rojo).

Ahora falta la modificación de la clave predeterminada del administrador y la creación de un usuario consultor. Para esto hay que usar la herramienta administrativa, que también hace parte del *Apache Directory Project*, llamada *Apache Directory Studio*.

Primero hay que realizar la configuración de la conexión al servidor indicando la URL y el puerto por el que se va a conectar, además del nombre y la clave de

algún usuario registrado en el directorio que tenga permisos de creación de registros, en este caso el administrador. Al colocar el nombre del usuario se debe escribir el DN completo. Los datos de conexión que trae por defecto el administrador del sistema son:

- Usuario: uid=admin,ou=system
- Contraseña: secret

Después de configurada dicha conexión y lograr conectarse con el servidor se va a realizar el cambio de contraseña del administrador. Esto se hace de la siguiente forma:

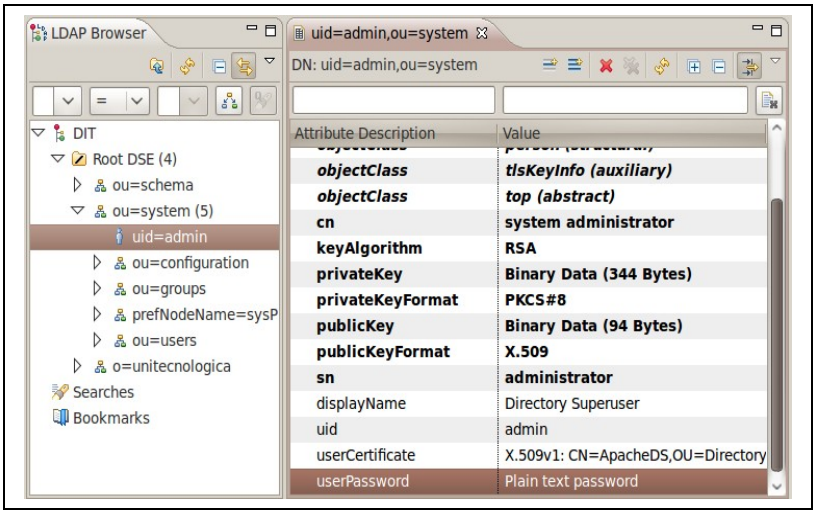

Gráfico No 7: Entrada del administrador en Apache Directory Studio.

1. Se escoge el registro del administrador (panel de la izquierda, gráfico 7) y se hace doble clic sobre el atributo *userPassword* (panel de la derecha, gráfico 7).

2. En la ventana modal que aparece (gráfico 8), se hace clic en la pestaña "*New Password*" y se ingresa una nueva clave en el cuadro de texto "*Enter New Password*". Se le puede cambiar el algoritmo de encriptado de la clave seleccionando otra opción en el campo "*Select Hash Method*". Al finalizar se presiona OK.

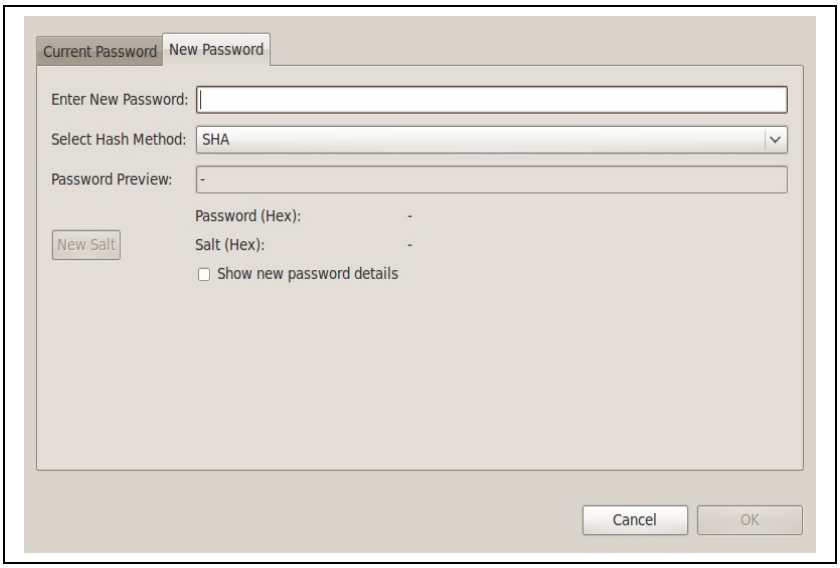

Gráfico No 8: Ventana de cambio de clave Apache Directory Studio.

Siguiendo los pasos anteriores se ha llevado a cabo el cambio de clave predeterminada del usuario administrador. El único objetivo faltante es la creación del usuario consultor y la asignación de sus respectivos permisos. Primero, hay que crear el usuario en el directorio. Para crearlo se siguen los siguientes pasos:

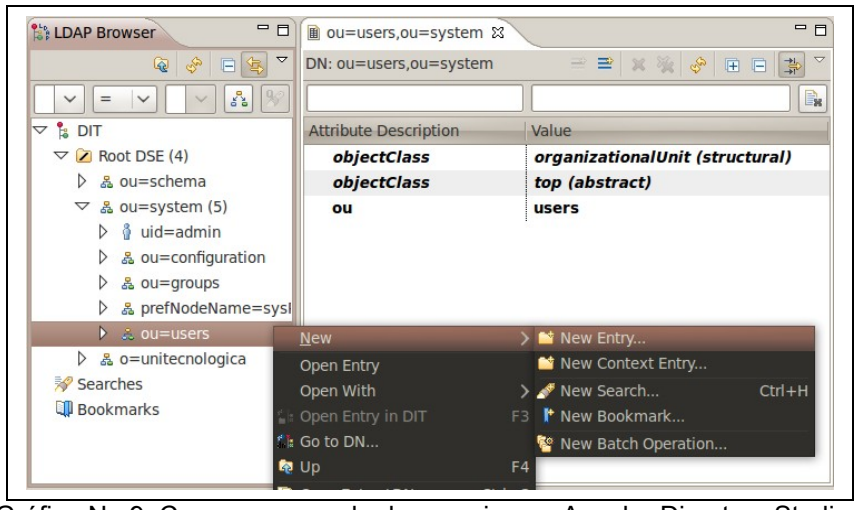

Gráfico No 9: Crear nuevo nodo de usuario con Apache Directory Studio.

1. Se hace clic derecho sobre la entrada que va a contener el registro del usuario, en este caso ou=users,ou=system. De la lista de opciones que despliega, se selecciona "*New*", luego "*New Entry...*" y se hace clic sobre esta última (ver gráfico 9).
2. En el cuadro de diálogo que aparece hay dos opciones: "*Create entry from scratch*" y "*Use existing entry as template*". La primera opción es para crear registros seleccionando cada uno de los *schemas* que se van a usar, mientras que la segunda permite seleccionar cualquier otra entrada en el directorio y se toman los *schemas* que se usen en esa. Para esta ocasión se va a seleccionar "*Create entry from scratch*" y se hace clic en "*Next*" (ver gráfico 10).

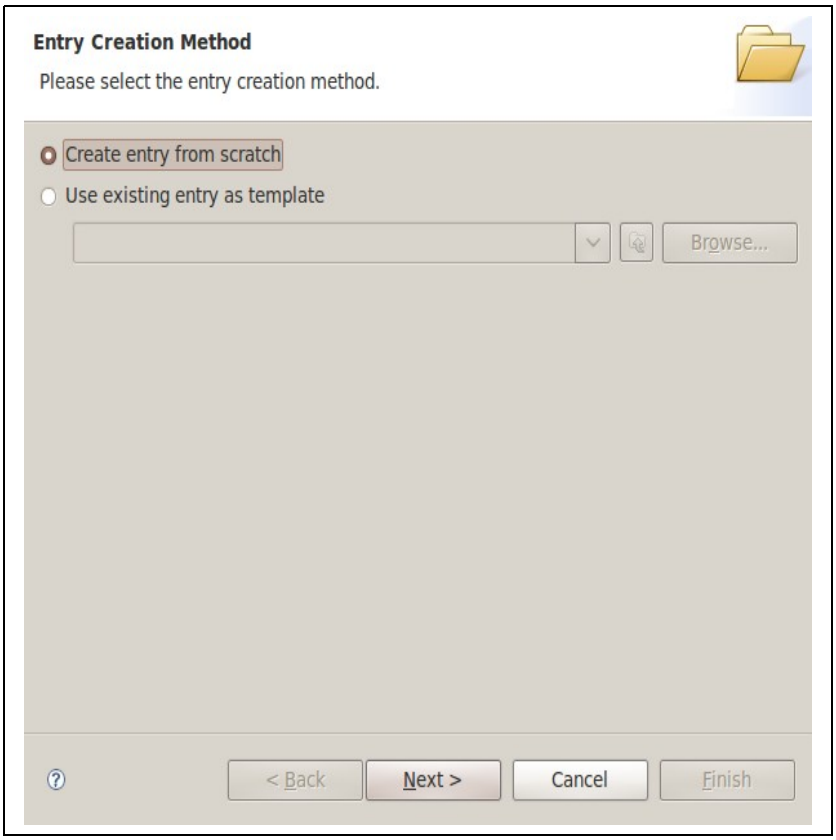

Gráfico No 10: Modo de creación de nodo en Apache Directory Studio.

3. Ahora aparece en el cuadro de diálogo una lista a la izquierda y un cuadro en blanco a la derecha. En la lista de la izquierda aparecen los *schemas* que se pueden usar y a medida que se vayan agregando con el botón "*Add*", van apareciendo en el cuadro de la derecha. Los que se hayan agregado, se pueden remover seleccionándolos en el cuadro de la derecha y haciendo clic en el botón "*Remove*". Para crear un usuario, se agrega el *schema inetOrgPerson,* el cual incluye también los *schemas organizationalPerson* y *person*, debido a que hereda de estos (ver gráfico 11). Se hace clic en "*Next*".

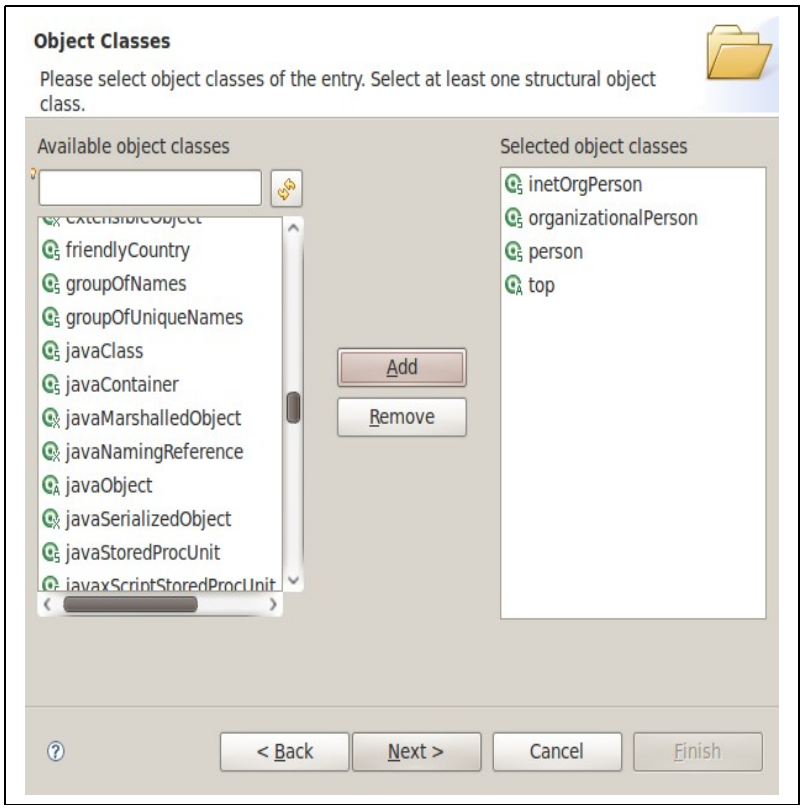

Gráfico No 11: Selección de schema inetOrgPerson en Apache Directory Studio.

4. En el siguiente cuadro se agregan el padre y el RDN del registro. El padre se asigna automáticamente al registro sobre el que se hizo clic derecho en el primer paso, pero puede ser cambiado colocando el DN completo. Para el RDN, se selecciona un atributo y se le coloca un valor. Por convención, para los usuarios se suele usar el atributo uid. Se selecciona dicho atributo y se le asigna un valor en el cuadro de texto a la derecha. Para este caso se le dio el valor de "consultor1" (ver gráfico 12). Al finalizar se hace clic en "*Next*".

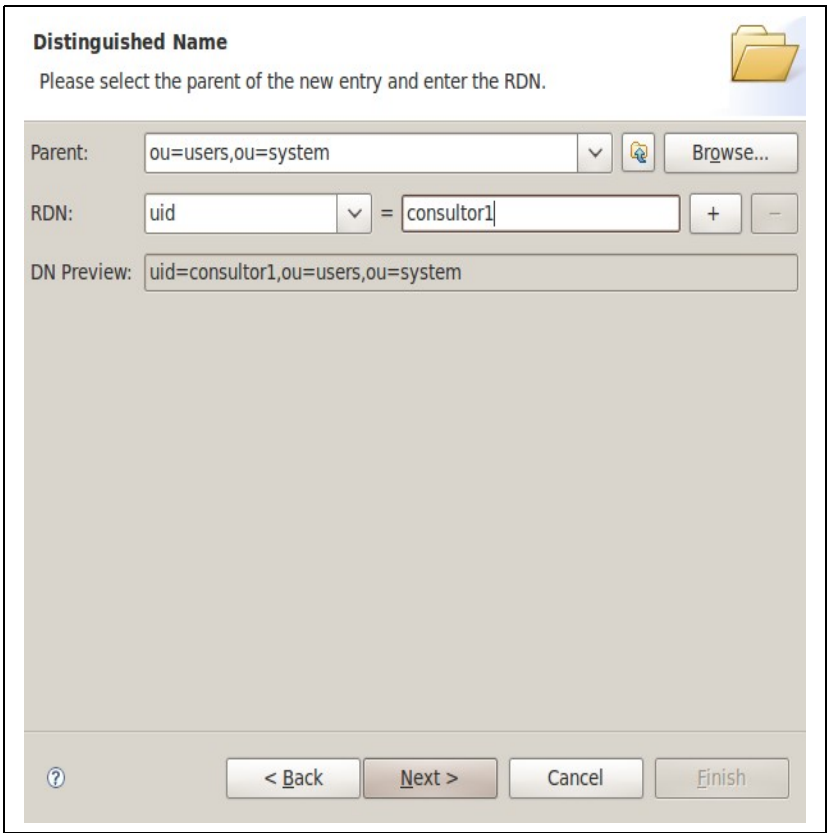

Gráfico No 12: Asignación de RDN a un nodo inetOrgPerson en Apache Directory Studio.

5. Por último, hay *schemas* que tienen algunos atributos obligatorios además del usado en el RDN, como en este caso, el atributo *cn* y el *sn*. Estos dos atributos no son obligatorios para el *schema inetOrgPerson*, pero si para uno de sus padres (en este caso, *person*). Cuando hay un *schema* que tiene algún atributo obligatorio, el *Apache Directory Studio* indica de alguna forma que hay que asignarle algún valor. En este cuadro también se puede agregar al registro cualquier otro atributo que sea necesario. Para este caso se le va a asignar una contraseña al usuario, y eso se hace con el atributo *userPassword*. Después de haber asignado los respectivos valores a los atributos, se hace clic en "*Finish*" (ver gráfico 13).

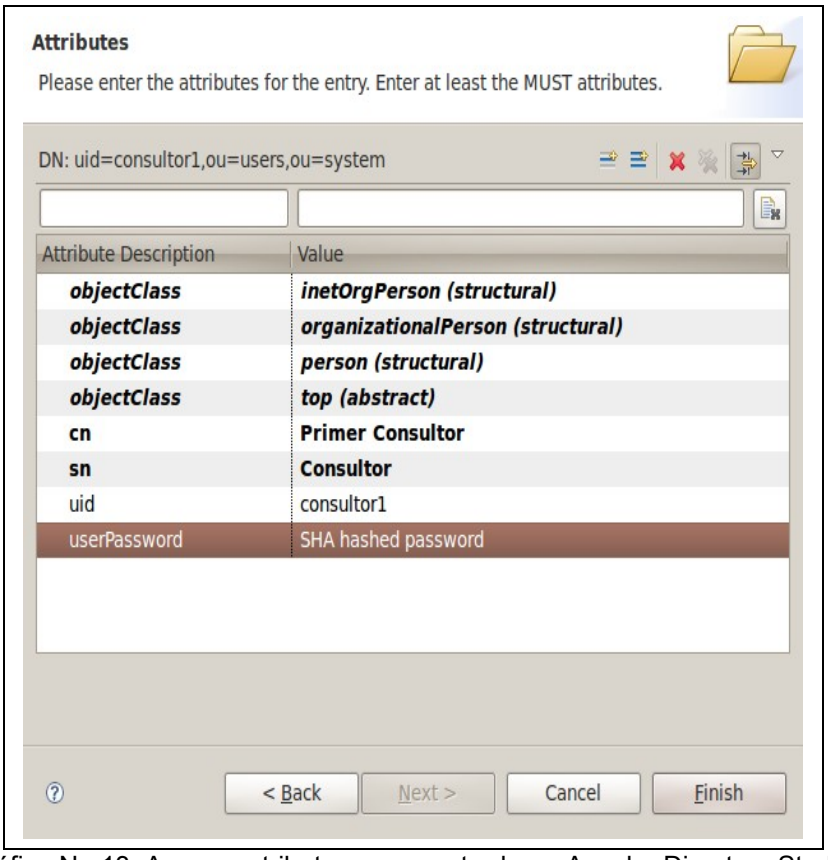

Gráfico No 13: Agregar atributos a una entrada en Apache Directory Studio.

Después de haber creado el usuario, se procede a la creación del grupo al cual va a pertenecer este. Para la creación del grupo se puede seguir una secuencia de pasos análoga a los que se siguieron para crear el usuario. Los pasos son los siguientes:

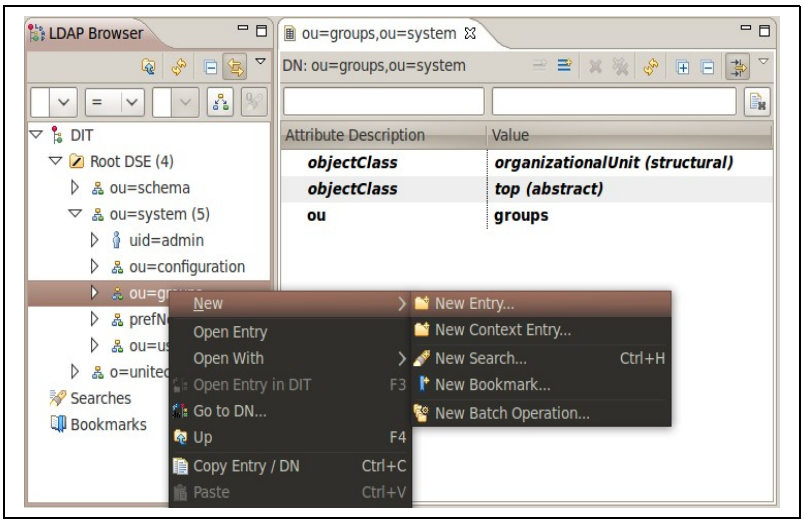

Gráfico No 14: Crear nuevo nodo de grupo con Apache Directory Studio.

1. Se hace clic derecho sobre la entrada que va a contener el registro del grupo, en este caso ou=groups,ou=system. De la lista de opciones que despliega, se selecciona "*New*", luego "*New Entry...*" y se hace clic sobre esta última (ver gráfico 14).
2. El cuadro de diálogo que aparece se selecciona la opción "*Create entry from scratch*" y se hace clic en "*Next*" (ver gráfico 10). Este cuadro de diálogo se explicó en el proceso de creación del usuario en el segundo paso.
3. En las siguientes opciones, se selecciona el *schema groupOfNames* y se hace clic en "*Next*" (ver gráfico 15). Esto también se explicó en el proceso de creación de usuario en el tercer paso.

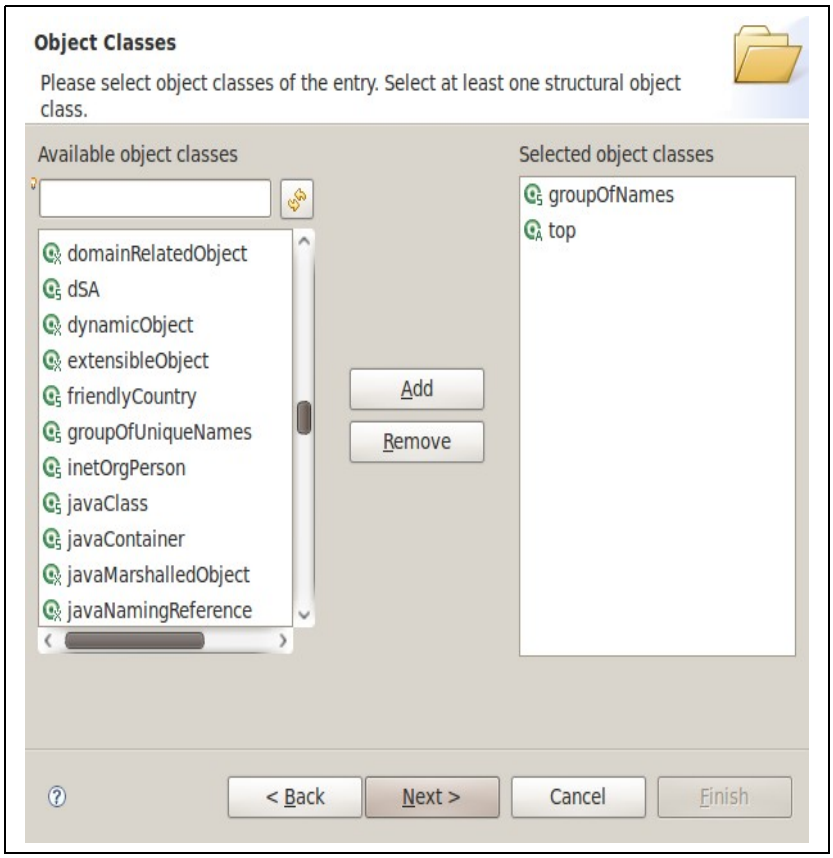

Gráfico No 15: Selección de schema groupOfNames en Apache Directory Studio.

4. Para la asignación del RDN, se escoge el atributo *cn* y se le asigna un valor. En este caso se le dio el valor de "consultores" (ver gráfico 16). Esto se explicó en el cuarto paso del proceso para crear un usuario. Se hace clic en "*Next*".

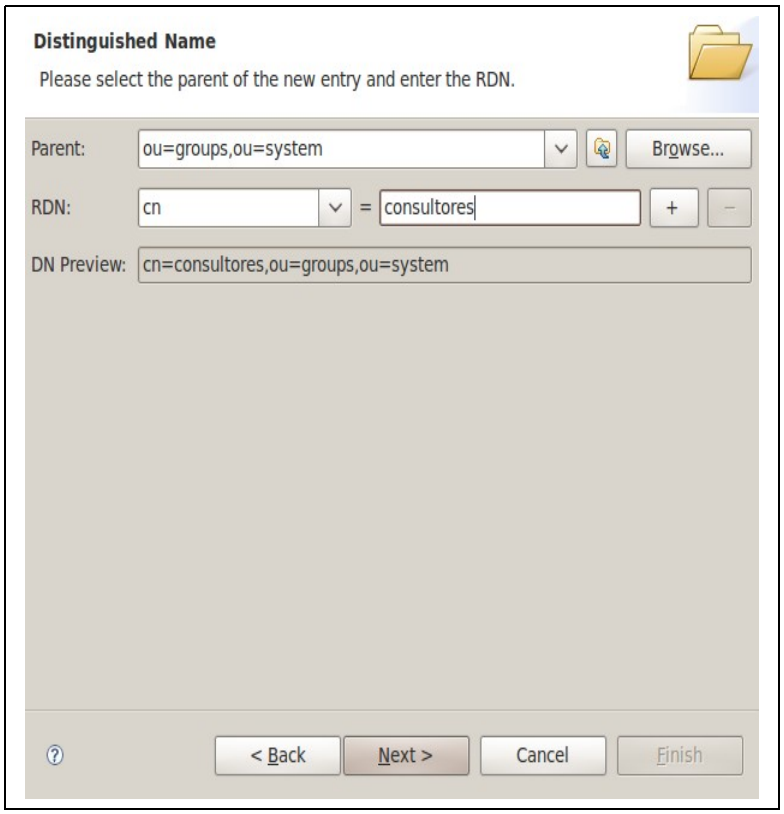

Gráfico No 16: Asignación de RDN a un nodo groupOfName en Apache Directory Studio.

5. Para finalizar se le agregan los valores respectivos a los atributos obligatorios, en caso de haberlos. En este caso el atributo *member* debe recibir los DNs de los usuarios que pertenecen al grupo. Los demás atributos son opcionales. Después de haber asignado los respectivos valores se hace clic en "*Finish*" (ver gráfico 17).

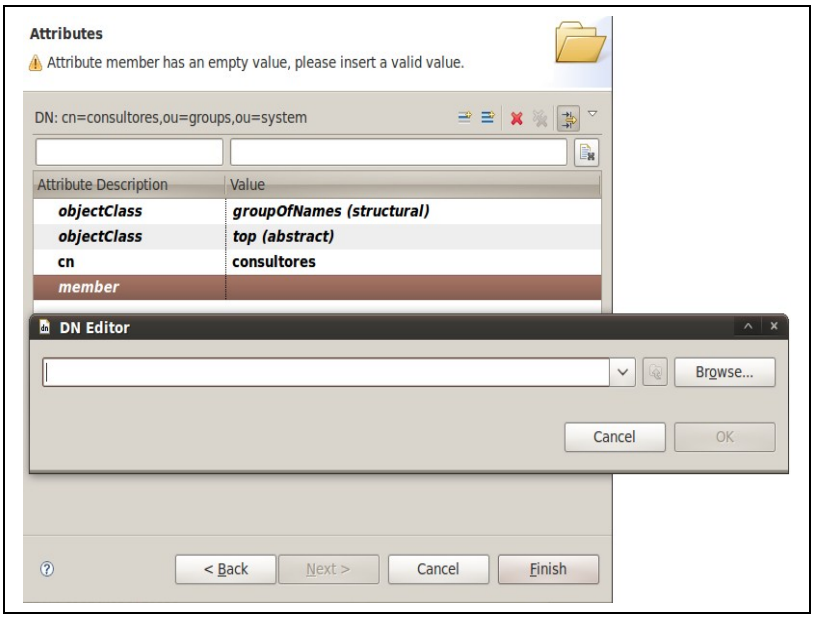

Gráfico No 17: Asignación de un valor al atributo member en Apache Directory Studio.

Lo siguiente a realizar para la puesta en marcha del servicio es la creación de la estructura básica del árbol de información con la forma que se expuso previamente en este capítulo. La creación de estos nodos no se explicará detalladamente, pero sigue los mismos pasos de la creación de usuario y la creación de grupos, con la diferencia que se selecciona en el tercer paso como schema el *organizationalUnit* y como atributo *ou* para el RDN en el cuarto paso.

Por último, hay que restringir los permisos para el acceso a la información que se va a almacenar. Esto se lleva a cabo creando un tipo especial de nodo llamado *subentry*, donde se colocan las reglas que van a aplicarse a la información contenida en esa rama del directorio. El nodo se coloca inmediatamente debajo de la raíz que va a contener la información. La creación de este nodo es de la siguiente forma:

1. Se siguen los cuatro primeros pasos básicos para la creación de un nodo, explicados previamente. Para crear un nodo del tipo *subentry* se elegiría en el segundo paso "*Create entry from scratch*", en el tercero el *schema subentry* y en el cuarto el atributo *cn* para el RDN.
2. En el quinto paso, Apache Directory Studio exige que el atributo *subtreeSpecification* contenga un valor. Este atributo se usa para indicar cuales son los nodos a los que se le aplicarían las restricciones de seguridad. Se puede indicar la raíz del árbol, cuantos niveles en la jerarquía (mínimo y máximo), excluir nodos o aplicar filtros. Apache Directory Studio ofrece una interfaz gráfica para llenar la información de este campo (ver gráfico 18). Al terminar de aplicar las reglas se hace clic en "OK".

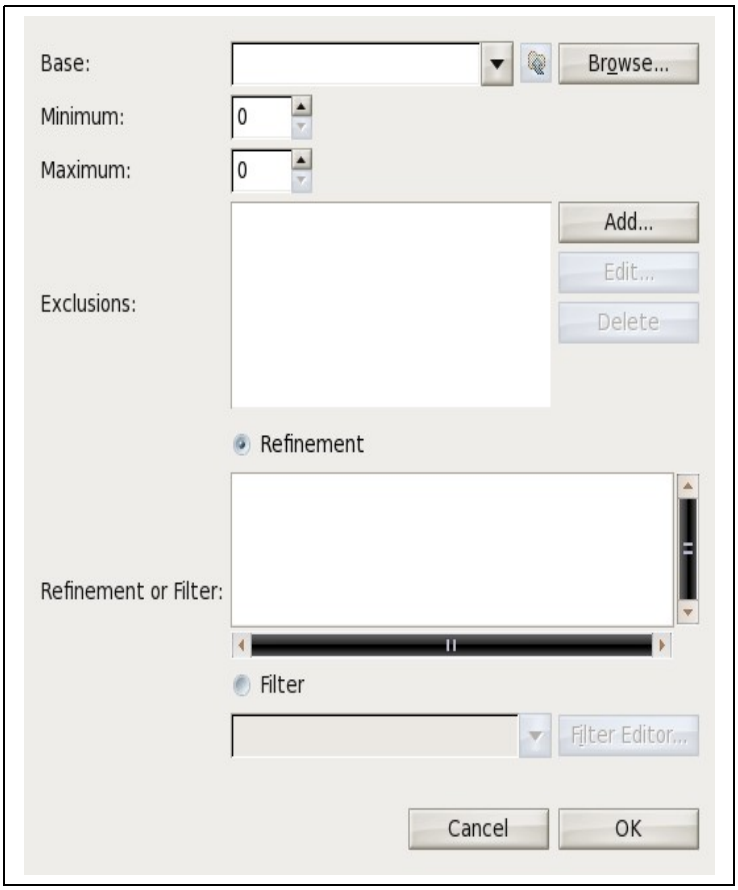

Gráfico No 18: Asignación de un valor al atributo subtreeSpecification en Apache Directory Studio.

3. Después de haber asignado un valor al atributo *subtreeSpecification*, se procede a agregar otro atributo llamado *prescriptiveACI*. Para poder agregar este atributo se debe desactivar la opción "*Show subschema attributes only*".
4. Inmediatamente después de agregar el atributo, *Apache Directory Studio* muestra una advertencia diciendo que no se puede modificar el valor del atributo *prescriptiveACI*. Se ignora dicha advertencia haciendo clic en"OK".
5. Apache Directory Studio también provee una interfaz gráfica para completar este atributo. Se tiene que indicar un ID para nombrar las restricciones de seguridad. También se le asigna una precedencia para que el servidor

pueda decidir a cual darle prioridad en caso de definir varias reglas de seguridad a lo largo del directorio y que estas puedan entrar en conflicto. Se especifica el nivel de autenticación (mínimo) para que un usuario pueda tener acceso a la sección del árbol que se está limitando. Se elige si se especifican las normas de seguridad primero por usuario o por item, aunque esto no afecta en nada la aplicación de las normas de seguridad sino la forma en que se definen estas. En este caso se seleccionó "*User first*" (ver gráfico 19).

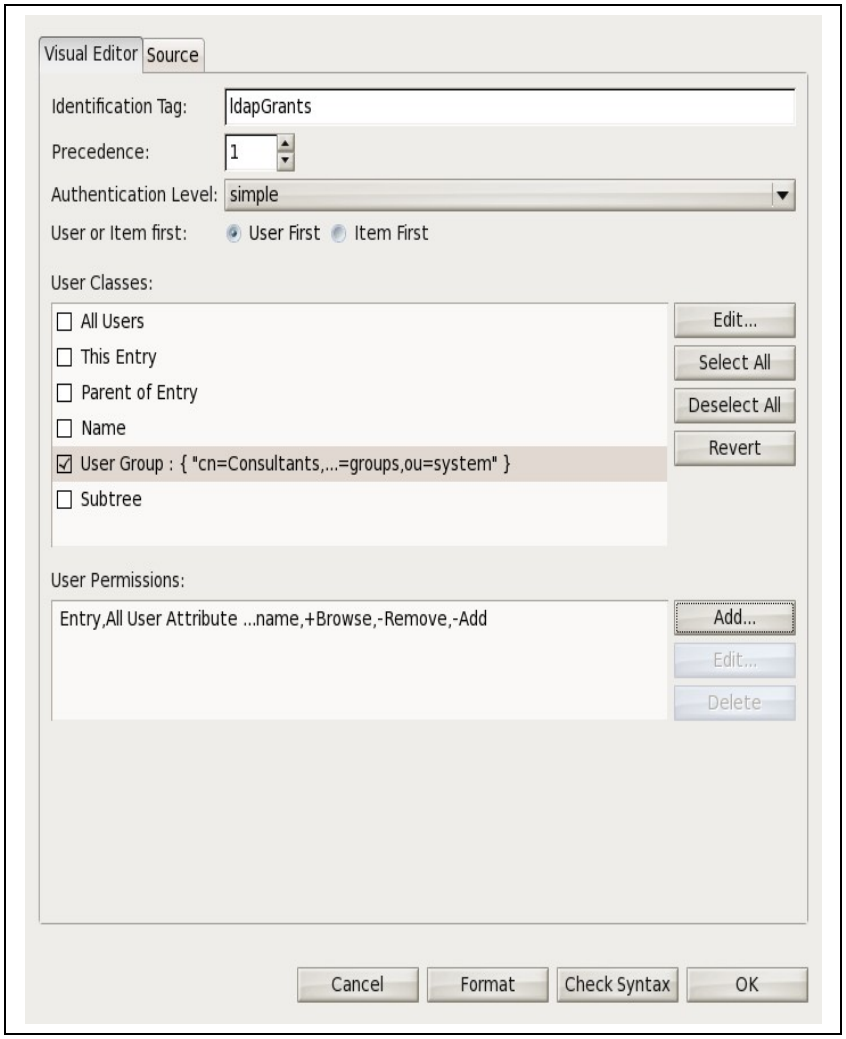

Gráfico No 19: Asignación de un valor al atributo prescriptiveACI en Apache Directory Studio.

6. Al seleccionar "*User first*" aparecen los cuadros *User classes* y *User Permissions*. En *User Classes* se especifíca a que usuarios se le aplican los permisos. Se puede aplicar a usuarios específicos, todos o algunos definidos por grupos. Al hacerlo por grupos se debe especificar el DN del grupo al que pertenecen los usuarios. En *User Permissions* se especifica a que tiene acceso o no los usuarios especificados y que puede hacer o no sobre los elementos a los que puede acceder. Se puede ver la estructura de todo el valor que se le va a asignar al atributo haciendo clic en la pestaña "*Source*" (ver gráfico 19). Después de realizar las especificaciones necesarias, se hace clic en "OK" y luego se finaliza la creación del nodo.

En la página oficial de ApacheDS se puede encontrar información completa sobre el modelo administrativo y como se configura la seguridad en el directorio, en la sección de la documentación *Advanced User's Guide*.

Con estas configuraciones finales se tiene el servicio de directorio funcionando, publicado y con los mecanismos de seguridad necesarios para evitar intrusiones no deseadas o fugas accidentales de información.

RECOMENDACIONES A FUTURO

Como primera recomendación, al haberse creado un nuevo servicio como tal que va a servir de soporte para otros, sería bueno que en la universidad se contratara personal que sirviera para darle mantenimento a dicho servicio, hacerle backups cada cierto tiempo y monitorear su funcionamiento para asegurar su continuidad operacional.

También es recomendable que en la universidad se tuviera en cuenta este servicio de directorio al momento de adquirir nuevas soluciones de software. Esto con el fin de que evaluen si la herramienta que se vaya a adquirir y/o a implementar en la U, tenga compatibilidad con el protocolo LDAP para que pueda ser integrada con este servicio.

Sería recomendable que en la universidad se hiciera un seguimiento al *Apache Directory Project* para que se pueda conservar un sistema actualizado y que aprovechara correctamente cualquiera funcionalidad nueva que sea incluida en el ApacheDS que sea para beneficio del servicio ya implementado.

Otra recomendación para que aumente la seguridad en los sistemas de información de la UTB, es que debería evaluarse periódica y constantemente los mecanismos de seguridad implementados en cada servicio. Esto, debido a que

aunque en el ApacheDS hayan unas fuertes políticas de seguridad establecidas la integración con otros servicios puede comprometer la seguridad de la información contenida en este por la falta de buenos mecanismos del lado del servicio. La cadena es tan fuerte como su eslabón más débil.

Por último, sería bueno que en un futuro se continuara esta investigación, ampliando la tesis con la implementación de un sistema SSO que tomara como base el servicio de directorio que se implementó en este trabajo.

CONCLUSIONES

Del presente trabajo desarrollado, se puede concluir que el tema de la integración de servicios es complejo y que se debe tener en cuenta siempre en el proceso de implementación de tecnologías en una empresa. No es un asunto que se evalua al final, debido a siempre se deben tomar en consideración los sistemas que existan actualmente para que se puedan comunicar fácilmente con los nuevos, aunque estos sean desarrollos propios o adquisiciones de software.

También se puede concluir que en la UTB el tema de la integración de los servicios es algo que no se ha tenido en cuenta hasta el momento, pero que con el paso del tiempo esto se ha vuelto una necesidad. Por lo tanto, la universidad actualmente es que está dándose cuenta de las ventajas de tener una mayor integración en los servicios y tener una plataforma tecnológica más unificada.

Otra conclusión que puede extraerse, es la diversidad de métodos que existen para la solución del problema de la integración de servicios. Desde el método más simple como es la implementación de una base de datos centralizada, hasta el más complejo y demorado como es el desarrollo de un sistema SSO propio; todos tenían la capacidad de servir como solución a la problemática presentada para este trabajo, aunque al final se haya escogido solo la más conveniente que fue el servicio de directorio.

Por último, puede concluirse que es de suma importancia y necesidad tener siempre en cuenta el aspecto de la seguridad en cada una de las etapas de la implementación de un sistema de información y de las posibles integraciones subsecuentes que tenga este. Esto se debe hacer con el fin de siempre asegurar unos niveles mínimos de seguridad y que no se comprometa información importante de los sistemas al realizar integraciones con otros nuevos. La seguridad es un requerimiento esencial en todo sistema de información.

BIBLIOGRAFÍA

1. Colaboradores de Wikipedia. *Autenticación* [en línea]. Wikipedia, La enciclopedia libre, 2010 [fecha de consulta: 5 de febrero del 2010]. Disponible en <http://es.wikipedia.org/w/index.php?title=Autenticaci%C3%B3n&oldid=33679678>.
2. Colaboradores de Wikipedia. *Autorización* [en línea]. Wikipedia, La enciclopedia libre, 2009 [fecha de consulta: 28 de julio del 2009]. Disponible en <http://es.wikipedia.org/w/index.php?title=Autorizaci%C3%B3n&oldid=28450594>.
3. Colaboradores de Wikipedia. *Single sign-on* [en línea]. Wikipedia, the free encyclopedia, 2010 [fecha de consulta: 20 de enero de 2010]. Disponible en <http://en.wikipedia.org/w/index.php?title=Single_sign-on&oldid=336003846>.
4. Calzada Pradas, Rafael. *Introducción al servicio de directorio* [en línea]. RedIRIS, 2010 [fecha de consulta: 12 de octubre de 2009]. Disponible en <https://www.rediris.es/ldap/doc/ldap-intro.pdf>.
5. Dunne, Chris. *Build and implement a single sign-on solution* [en línea]. IBM, 2003 [fecha de consulta: 1 de marzo de 2010]. Disponible en <http://www.ibm.com/developerworks/web/library/wa-singlesign/>.
6. Colaboradores de Wikipedia. *Directory service* [en línea]. Wikipedia, the free encyclopedia, 2010 [fecha de consulta: 12 de enero del 2010]. Disponible

en <http://en.wikipedia.org/w/index.php?title=Directory_service&oldid=337443453>.

7. Webmaster de Zytrax.com. *LDAP concepts & overview* [en línea]. Zytrax Communications, 2010 [fecha de consulta: 6 de marzo del 2010]. Disponible en <http://www.zytrax.com/books/ldap/ch2/#database>.
8. Colaboradores de Wikipedia. *X.500* [en línea]. Wikipedia, the free encyclopedia, 2009 [fecha de consulta: 11 de octubre del 2009]. Disponible en <http://en.wikipedia.org/w/index.php?title=X.500&oldid=322815685>.
9. Colaboradores de Wikipedia. *Lightweight Directory Access Protocol* [en línea]. Wikipedia, the free encyclopedia, 2010 [fecha de consulta: 9 de enero del 2010]. Disponible en <http://en.wikipedia.org/w/index.php?title=Lightweight_Directory_Access_Protocol&oldid=336332767>.
10. A. Howes, Timothy. *The Lightweight Directory Access Protocol: X.500 Lite* [en línea]. OpenLDAP, 2009 [fecha de consulta: 11 de octubre del 2009]. Disponible en <www.openldap.org/pub/umich/ldap.pdf>.
11. Colaboradores de Wikipedia. *Directory Access Protocol* [en línea]. Wikipedia, La enciclopedia libre, 2009 [fecha de consulta: 11 de octubre del 2009]. Disponible en <http://es.wikipedia.org/w/index.php?title=Directory_Access_Protocol&oldid=30506118>.
12. Colaboradores de Wikipedia. *List of single sign-on implementations* [en línea]. Wikipedia, La enciclopedia libre, 2010 [fecha de consulta: 24 de febrero del 2010]. Disponible en <http://en.wikipedia.org/w/index.php?title=List_of_single_sign-on_implementations&oldid=345642386>.

13. Comunidad JASIG. *CAS User Manual* [en línea]. Wiki de JASIG, 2010 [fecha de consulta: 25 de febrero del 2010]. Disponible en <http://www.ja-sig.org/wiki/display/CASUM/Home>.
14. Comunidad de OpenSSO. *Página oficial de OpenSSO* [en línea]. OpenSSO, 2010 [fecha de consulta: 25 de febrero del 2010]. Disponible en <https://opensso.dev.java.net/>.
15. Atricore. *JOSSO - Java Open Single Sign-On Project Home* [en línea]. Wiki de JOSSO, 2010 [fecha de consulta: 25 de febrero del 2010]. Disponible en <http://www.josso.org/confluence/display/JOSSO1/JOSSO+-+Java+Open+Single+Sign-On+Project+Home>.
16. Universidad de Michigan. *Página oficial de CoSign* [en línea]. CoSign, Collaborative Single Sign-On, 2010 [fecha de consulta: 26 de febrero del 2010]. Disponible en <http://cosign.sourceforge.net/>.
17. Apache software foundation. *The Apache Directory Project* [en línea]. Página oficial de Apache Directory Server, 2010 [fecha de consulta: 12 de marzo del 2010]. Disponible en <http://directory.apache.org/>.
18. OpenLDAP Foundation. *OpenLDAP* [en línea]. Página oficial de OpenLDAP, 2010 [fecha de consulta: 12 de marzo del 2010]. Disponible en <http://www.openldap.org/>.
19. Comunidad de OpenDS. *OpenDS* [en línea]. Página oficial de OpenDS, 2010 [fecha de consulta: 12 de marzo del 2010]. Disponible en <http://www.opends.org/>.
20. Butcher, Matt. *Mastering OpenLDAP: Configuring, Securing, and Integrating Directory Services*. Reino Unido. Packt Publishing, 2007.

21. Colaboradores de Wikipedia. *Front-end y back-end* [en línea]. Wikipedia, La enciclopedia libre, 2010 [fecha de consulta: 21 de marzo del 2010]. Disponible en <http://es.wikipedia.org/w/index.php?title=Front-end_y_back-end&oldid=35337115>.
22. Colaboradores de Wikipedia. *Sistema de gestión de contenidos* [en línea]. Wikipedia, La enciclopedia libre, 2010 [fecha de consulta: 11 de junio del 2010]. Disponible en <http://es.wikipedia.org/w/index.php?title=Sistema_de_gesti%C3%B3n_de_contenidos&oldid=37940270>.
23. Comunidad de Drupal. *Sobre Drupal* [en línea]. Drupal Hispano, Comunidad de usuarios de Drupal, 2010 [fecha de consulta: 11 de junio del 2010]. Disponible en <http://drupal.org.es/drupal>.
24. Colaboradores de Wikipedia. *Kerberos* [en línea]. Wikipedia, La enciclopedia libre, 2010 [fecha de consulta: 5 de junio del 2010]. Disponible en <http://es.wikipedia.org/w/index.php?title=Kerberos&oldid=37782376>.
25. Comunidad de Moodle. *Acerca de Moodle* [en línea]. Moodle en español, 2010 [fecha de consulta: 11 de junio del 2010]. Disponible en <http://docs.moodle.org/es/Acerca_de_Moodle>.
26. Colaboradores de Wikipedia. *Modelo OSI* [en línea]. Wikipedia, La enciclopedia libre, 2010 [fecha de consulta: 11 de junio del 2010]. Disponible en <http://es.wikipedia.org/w/index.php?title=Modelo_OSI&oldid=38130820>.
27. Colaboradores de Wikipedia. *Service (systems architecture)* [en línea]. Wikipedia, La enciclopedia libre, 2010 [fecha de consulta: 11 de junio del

2010]. Disponible en <http://en.wikipedia.org/w/index.php?title=Service_(systems_architecture)&oldid=345936924>.

28. Colaboradores de Wikipedia. *Transport Layer Security* [en línea]. Wikipedia, La enciclopedia libre, 2010 [fecha de consulta: 12 de junio del 2010]. Disponible en <http://en.wikipedia.org/w/index.php?title=Transport_Layer_Security&oldid=368405219>.
29. Colaboradores de Wikipedia. *Ticket Granting Ticket* [en línea]. Wikipedia, La enciclopedia libre, 2010 [fecha de consulta: 12 de junio del 2010]. Disponible en <http://en.wikipedia.org/w/index.php?title=Ticket_Granting_Ticket&oldid=255445634>.
30. Colaboradores de Wikipedia. *Familia de protocolos de Internet* [en línea]. Wikipedia, La enciclopedia libre, 2010 [fecha de consulta: 16 de junio del 2010]. Disponible en <http://es.wikipedia.org/w/index.php?title=Familia_de_protocolos_de_Internet&oldid=38091273>.

ANEXOS

Anexo A

Entrevista a Juan a Carlos Mantilla - Director de Servicios Informáticos de la UTB

Entrevistador: ¿Cuáles son los sistemas de información con los que cuenta la UTB?

Director: La Utb Cuenta con los siguientes sistemas de información:

SISTEMA	SERVICIOS	USUARIOS
SIRIUS	Soporta los procesos de la administración académica (admisiones, matrículas, cursos, notas, horarios, docentes, certificados, proceso de graduación, etc.)	Docentes, estudiantes, administrativos
SAVIO	Plataforma de apoyo y aprendizaje virtual.	Docentes, estudiantes
SIFAD	Soporte a los procesos de naturaleza administrativa , contable y financiera:	Administrativos

	Contabilidad, cuentas por pagar, por cobrar, cartera, inventarios, activos, nómina de personal.	
Sistema de información bibliográfica JANIUN	Servicios de biblioteca (catálogo, préstamos, acceso a bancos de datos)	Docentes, estudiantes, egresados, usuares externos.
Sistema de gestión documental	Digitalización y archivo digital de correspondencia	Administrativos

Entrevistador: ¿Qué software usan esos sistemas?

Director:

SISTEMA	**UTILIZA**
SIRIUS	ORACLE 10g, sobre Solaris 10. Oracle internet Application Server.
SAVIO	MOODLE.
SIFAD	INFORMIX DYNAMIC SERVER, INFORMIZ 4GL sobre UNIX SCO
Sistema de información bibliográfica	Aplicacion Web, que se ejcuta en ORACLE sobre UNIX en màquinas del proveedor

JANIUN	
Sistema de gestión documental	DOCUWARE sobre Unix

Entrevistador: ¿Cuál o cuáles son los problemas más frecuentes que presentan los usuarios de los sistemas de información de la UTB?

Director: No hay un suficiente nivel de integración entre los sistemas de información. Algunos sistemas no están diseñados para la operación actual. En consecuencia no cubren todos los procesos haciéndose necesario el trabajo manual. Baja inducción de los empleados a los cargos, que ocasionan tasas de error altas por desconocimiento del proceso o del manejo del sistema.

Entrevistador: ¿Cómo se resuelven actualmente dichos problemas?

Director: Se implementan varias estrategias:

- Mayor mano de obra dedicada.
- Modificación al diseño del sistema para adicionarle nuevas funcionalidades.
- Se evalúan alternativas para adquisición de una nueva solución.

Entrevistador: ¿Se ha implementado previamente alguna solución para este problema?

Director: En el plano de los procesos académicos se adquirió una solución nueva (actual SIRIUS) que sustituyo una solución antigua que presentaba exactamente esos mismos problemas.

Entrevistador: Si se hiciera una propuesta para la solución de este problema, ¿estaría dispuesto a colaborar en su ejecución?
Director: Si. Totalmente.

Entrevistador: ¿Qué requisitos esperaría que cumpliera dicha solución?
Director: Dependiendo de cual solución se trate, es decir, de cuáles procesos va a atender, los requerimientos específicos también serían distintos. Pero en términos generales toda solución debe cumplir con las siguientes características:

- Ofrecer información en tiempo real, con una velocidad de respuesta razonable.
- Garantizar la calidad de la información que entrega.
- Seguridad para los datos.
- Accesibilidad para los diferentes tipos de usuario.

Anexo B

Descripción de los campos del formulario de configuración de LDAP en Moodle.

Campo	Breve descripción	Ejemplo de configuración
Ajustes del servidor LDAP		
URL del host	La URL del servicio del LDAP al que se va a conectar.	ldap://localhost:389
Versión	La versión del protocolo LDAP que usa el servidor al cual se va a conectar. Campo con dos opciones posibles: 2 y 3.	3
Codificación LDAP	La codificación de caracteres usada por el servidor LDAP.	utf-8
Fijar ajustes		
Ocultar contraseñas	Campo de opciones “Si” o “No”. Se escoge “Si” en caso de querer evitar el almacenamiento de las contraseñas localmente en Moodle.	Si
Nombre distinguido	DN del usuario con el que se va a enlazar al servidor LDAP para realizar las consultas.	uid=admin,ou=system
Contraseña	Contraseña del usuario para enlazar al directorio (puede ser vacía).	(vacío)
Ajustes de búsqueda de usuario		
Tipo de usuario	Campo de múltiples opciones, para seleccionar el tipo de usuario almacenado en el LDAP. Los posibles valores son: Novell Edirectory, posixAccount (rfc	MS ActiveDirectory

	2307), posixAccount (rfc 2307bis), sambaSamAccount (v.3.0.7), MS ActiveDirectory, Por defecto.	
Contextos	DN del LDAP donde se van a buscar los usuarios. Se pueden especificar varios separándolos por ";".	ou=users,o=Moodle
Buscar subcontextos	Campo con opciones "Si" o "No". Se escoge "Si", en caso de ser necesaria la búsqueda del usuario en subramas de los contextos previamente especificados.	Si
Alias de referencia	Campo de "Si" o "No". Se selecciona "Si" en caso de necesitar buscar también en los alias en los contextos especificados.	Si
Atributo de usuario	Nombre del atributo en el LDAP que almacena el nombre del usuario.	cn
Atributo de miembro	Nombre del atributo en el LDAP que almacena el nombre del grupo al que pertenece el usuario. Puede dejarse vacío.	organizationName
Clase de objetos	Filtro usado para la búsqueda de usuarios. Se escribe objectClass=(clase). La clase corresponde a un esquema LDAP. Se puede dejar vacío.	(vacío)
Forzar cambio de contraseña		
Forzar cambio de contraseña	Campo de opciones "Si" o "No". Se escoge "Si" en caso de querer forzar al	Si

	usuario que cambie su contraseña al ingresar por primera vez a Moodle.	
Utilizar página de cambio de contraseña estándar	Campo de opciones “Si” o “No”. Funciona si se escogió “Si” en el campo anterior y se selecciona “Si” en caso de querer usar la página de Moodle para cambiar contraseñas.	No
Formato de contraseña	Campo de múltiples opciones. Se escoge la opción del algoritmo de cifrado usado para almacenar las contraseñas en el LDAP. Las opciones son: Texto plano, Encriptación MD5, SHA-1 hash.	Encriptación MD5
URL para cambio de contraseña	Se llena este campo en caso de haber seleccionado “Si” en el campo “Forzar cambio de contraseña” y “No” en “Utilizar página de cambio de contraseña estándar”. Se escribe la URL de la página web a usar para cambiar la contraseña del usuario. Se puede dejar en (vacío).	(vacío)
Ajustes de caducidad de la contraseña LDAP		
Expiración	Campo con dos opciones: “No” y “LDAP”. Se selecciona “No” en caso que la contraseña no expire o “LDAP” en caso de leer del directorio el tiempo de caducidad de la contraseña.	No

Advertencia de expiración	Número de días antes de la expiración de la contraseña para avisar al usuario. Funciona en caso de haber seleccionado “LDAP” en el campo anterior.	30
Atributo de expiración	Nombre del atributo que indica el tiempo de expiración de la contraseña para el usuario. Funciona si se escogió “LDAP” en el campo “Expiración”. Puede dejarse vacío.	(vacío)
Entradas libres	Campo de opciones “Si” o “No”. Si se escoge “Si”, se le da acceso al usuario por un determinado tiempo después de expirada la contraseña.	Si
Atributo de entrada libre	Atributo usado para la cuenta de entrada libre. Este campo es opcional.	(vacío)
Habilitar creación por parte del usuario		
Crear usuarios externamente	Campo con opciones “Si” o “No”. Al seleccionar “Si” se está permitiendo la creación de usuarios en el directorio desde Moodle.	No
Contexto para usuarios nuevos	Funciona en caso de haber seleccionado “Si” en el campo anterior. Contiene el DN del contexto donde se almacenarán los usuarios nuevos.	ou=Moodle,dc=example,dc=com
Creador de curso		
Creadores	DN del grupo donde están almacenados	

	los creadores de curso.	
Script de sincronización del Cron		
Usuario externo eliminado	Campo con múltiples opciones. Dependiendo de la opción seleccionada, se decide que se hace con los usuarios al momento de sincronizar con una fuente externa. Las opciones son: Mantener interna, Suspender interna, Borrado completo. Dependiendo de la opción seleccionada, si el usuario es borrado en la fuente externa, este es mantenido localmente, desactivado o eliminado, respectivamente.	Borrado completo
NTLM SSO		
Habilitar	Campo con opciones “Si” o “No”. Es para habilitar SSO en Moodle.	Si
Sub-red	Campo que recibe una IP y una máscara de red para que se realice el SSO solo con los clientes de la subred especificada.	176.16.8.1/24
MS IE fast path?	Campo con opciones “Si” o “No”. Es para habilitar ciertas características que solo funcionan en MS Internet Explorer.	No
Mapeado de datos Todos estos son datos que se pueden almacenar localmente en Moodle. Cada uno recibe un atributo LDAP y están acompañados de tres campos de opción		

múltiple los cuales son: Actualizar datos locales, Actualizar datos externos, Bloquear valor. Estas opciones se usan para que se actualicen los datos de Moodle con la información del directorio, que se actualicen en LDAP los datos modificados en Moodle y si es posible la edición del valor de dicho campo o no, respectivamente. “Actualizar datos locales” tiene las siguientes opciones: Al crearse, En cada acceso. “Actualizar datos externos” tiene las opciones: Nunca, Al actualizar. Las opciones de “Bloquear valor” son: Desbloqueado, Desbloqueado si está vacío, Bloqueado. Todos estos campos pueden ser dejados en blanco.		
Nombre	Atributo que contiene el nombre del usuario.	gn
Apellido	Atributo que contiene el apellido del usuario.	surname
Dirección de correo	Atributo que contiene la dirección de correo del usuario.	mail
Ciudad	Atributo que contiene la ciudad de residencia del usuario.	city
País	Atributo que contiene el país del usuario.	countryName
Idioma	Atributo que contiene el idioma a usar por el usuario.	lang
Descripción	Atributo que contiene una descripción del usuario.	(vacío)
Página web	Atributo que contiene la página web del usuario.	(vacío)
Número de ID	Atributo que contiene el número del documento de identificación del usuario.	(vacío)

Institución	Atributo que contiene la institución donde trabaja la persona.	(vacío)
Departamento	Atributo que contiene el departamento del usuario.	departmentNumber
Teléfono 1	Atributo que contiene un número de teléfono del usuario.	telephoneNumber
Teléfono 2	Atributo que contiene un segundo número de teléfono del usuario.	mobile
Dirección	Atributo que contiene la dirección del usuario.	registeredAddress

Anexo C

Descripción de los campos del formulario de configuración general de LDAP en Drupal.

Campo	**Breve descripción**	**Ejemplo de configuración**
Authentication mode		
Choose authentication mode	Campo con dos opciones para elegir como se realiza la autenticación. Si se elige “Mixed mode. The LDAP authentication is performed only if Drupal authentication fails”, se realiza usando la información de Drupal, y en caso	LDAP directory only

	que esta falle, se usa el LDAP. Si por el contrario, se elige “LDAP directory only”, la autenticación solo se realiza con la información del directorio.	
Choose user conflict resolve procedure	Campo con dos opciones para determinar que se realiza en caso que exista localmente el mismo nombre de usuario. Las opciones son: Disallow login and log the conflict, Associate local account with the LDAP entry. La primera no permite el acceso en caso de coincidencia y la segunda asocia el nombre local con el del LDAP.	Disallow login and log the conflict
Security options		
Do not store users passwords during sessions	Un campo con opciones de activo e inactivo. En caso de estar activo, se almacenará temporalmente la contraseña del usuario durante la sesión.	Activo (marcado)
Sync LDAP password with the Drupal password	Otro campo con opciones de activo e inactivo. Si se activa, algún cambio de contraseña en Drupal se verá reflejado en el directorio. Solo funciona si se eligió “Mixed mode” en el primer campo.	Inactivo (sin marcar)
LDAP UI options		
Remove password	Campo con opciones de activo e inactivo. En caso de activarse, se remueven del formulario	Activo (marcado)

change fields from user edit form	de edición de perfil del usuario el campo de modificación de la contraseña.	
Alter email field on user edit form	Campo con múltiples opciones. Las opciones son: Do nothing, Remove email field from form, Disable email field on form. Dependiendo de la opción seleccionada, el campo del correo electrónico en el formulario de edición de perfil del usuario, se deja quieto, se quita por completo o se deja pero no se puede editar, respectivamente.	Do nothing

Anexo D

Descripción de los campos de configuración del servidor LDAP en Drupal.

Campo	**Breve descripción**	**Ejemplo de configuración**
Server settings		
Name	Un nombre para la conexión con el servidor. Este debe ser único.	servidor
LDAP server	URL donde está alojado el servidor LDAP.	localhost
LDAP port	Puerto de conexión por el que escucha el servidor LDAP.	389

Use Start-TLS	Campo con opciones de activo e inactivo. En caso de activarse, se inicia una sesión TLS con el servidor LDAP.	Inactivo (sin marcar)
Store passwords in encrypted form	Campo que tiene las opciones de activo e inactivo. Activa el cifrado en MD5 para el almacenamiento de las claves en el LDAP.	Activo (marcado)
Login procedure		
Base DNs	Un cuadro de texto en el que se pueden escribir uno o varios DNs para realizar las búsquedadas en el directorio.	ou=users,o=Drupal
UserName attribute	Atributo que almacena el nombre del usuario.	cn
Email attribute	Atributo que almacena el correo electrónico del usuario.	mail
PHP to transform login name	Cuadro de texto que recibe un código PHP que modifica el nombre de usuario antes de ser enviado al LDAP para la autenticación. Puede ser vacío.	(vacío)
PHP to filter users based on their LDAP data	Cuadro de texto que recibe un código PHP para restringir el acceso de algunos usuarios basado en la información que se encuentre almacenada en el directorio. Puede dejarse en blanco.	(vacío)
Advanced configuration		
DN for non-anonymous	DN de un usuario que se pueda usar para realizar las búsquedas en el directorio de	cn=admin,ou=users

search	forma no anónima.	
Password for non-anonymous search	Contraseña del usuario especificado en el campo anterior.	

Anexo E

El siguiente es el contenido completo del archivo de configuración de ApacheDS usado en el montaje del servidor.

```
<?xml version="1.0" encoding="UTF-8"?>
<!--
Licensed to the Apache Software Foundation (ASF) under one or more contributor license agreements. See the NOTICE file distributed with this work for additional information regarding copyright ownership. The ASF licenses this file to you under the Apache License, Version 2.0 (the "License"); you may not use this file except in compliance with the License. You may obtain a copy of the License at
http://www.apache.org/licenses/LICENSE-2.0
Unless required by applicable law or agreed to in writing, software distributed under the License is distributed on an"AS IS" BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied. See the License for the specific language governing permissions and limitationsunder the License.
-->

<spring:beans xmlns="http://apacheds.org/config/1.5.7" xmlns:spring="http://xbean.apache.org/schemas/spring/1.0"
xmlns:s="http://www.springframework.org/schema/beans">

<defaultDirectoryService id="directoryService" instanceId="default" replicaId="1" workingDirectory="example.com" allowAnonymousAccess="false" accessControlEnabled="true" denormalizeOpAttrsEnabled="false" syncPeriodMillis="15000" maxPDUSize="2000000">

<systemPartition>
<!-- use the following partitionConfiguration to override defaults for -->
<!-- the system partition -->

<jdbmPartition id="system" cacheSize="100" suffix="ou=system" optimizerEnabled="true" syncOnWrite="true">
<indexedAttributes>
<jdbmIndex attributeId="1.3.6.1.4.1.18060.0.4.1.2.1" cacheSize="100"/>
<jdbmIndex attributeId="1.3.6.1.4.1.18060.0.4.1.2.2" cacheSize="100"/>
```

```
<jdbmIndex attributeId="1.3.6.1.4.1.18060.0.4.1.2.3" cacheSize="100"/>
<jdbmIndex attributeId="1.3.6.1.4.1.18060.0.4.1.2.4" cacheSize="100"/>
<jdbmIndex attributeId="1.3.6.1.4.1.18060.0.4.1.2.5" cacheSize="10"/>
<jdbmIndex attributeId="1.3.6.1.4.1.18060.0.4.1.2.6" cacheSize="10"/>
<jdbmIndex attributeId="1.3.6.1.4.1.18060.0.4.1.2.7" cacheSize="10"/>
<jdbmIndex attributeId="ou" cacheSize="100"/>
<jdbmIndex attributeId="uid" cacheSize="100"/>
<jdbmIndex attributeId="objectClass" cacheSize="100"/>
</indexedAttributes>
</jdbmPartition>
</systemPartition>
<partitions>
<!-- NOTE: when specifying new partitions you need not include those -->
<!-- attributes below with OID's which are the system indices, if left -->
<!-- out they will be automatically configured for you with defaults. -->
<jdbmPartition id="utb" cacheSize="100" suffix="o=unitecnologica" optimizerEnabled="true"
syncOnWrite="true">
<indexedAttributes>
<jdbmIndex attributeId="1.3.6.1.4.1.18060.0.4.1.2.1" cacheSize="100"/>
<jdbmIndex attributeId="1.3.6.1.4.1.18060.0.4.1.2.2" cacheSize="100"/>
<jdbmIndex attributeId="1.3.6.1.4.1.18060.0.4.1.2.3" cacheSize="100"/>
<jdbmIndex attributeId="1.3.6.1.4.1.18060.0.4.1.2.4" cacheSize="100"/>
<jdbmIndex attributeId="1.3.6.1.4.1.18060.0.4.1.2.5" cacheSize="10"/>
<jdbmIndex attributeId="1.3.6.1.4.1.18060.0.4.1.2.6" cacheSize="10"/>
<jdbmIndex attributeId="1.3.6.1.4.1.18060.0.4.1.2.7" cacheSize="10"/>
<jdbmIndex attributeId="dc" cacheSize="100"/>
<jdbmIndex attributeId="ou" cacheSize="100"/>
<jdbmIndex attributeId="krb5PrincipalName" cacheSize="100"/>
<jdbmIndex attributeId="uid" cacheSize="100"/>
<jdbmIndex attributeId="objectClass" cacheSize="100"/>
</indexedAttributes>
</jdbmPartition>
</partitions>
<interceptors>
<normalizationInterceptor/>
<authenticationInterceptor/>
<referralInterceptor/>
<aciAuthorizationInterceptor/>
<defaultAuthorizationInterceptor/>
<exceptionInterceptor/>
<operationalAttributeInterceptor/>

<!-- Uncomment to enable the password policy interceptor
<passwordPolicyInterceptor/>
<keyDerivationInterceptor/>
-->

<schemaInterceptor/>
<subentryInterceptor/>
<collectiveAttributeInterceptor/>
<eventInterceptor/>
<triggerInterceptor/>

<!-- Uncomment to enable replication interceptor
<replicationInterceptor>
<configuration>
```

```
<replicationConfiguration serverPort="10390" peerReplicas="instance_b@localhost:10392">
<replicaId>
<replicaId id="instance_a"/>
</replicaId>
</replicationConfiguration>
</configuration>
</replicationInterceptor>
-->
</interceptors>

<!-- Uncomment to enable replication configuration -->
<!--replicationConfiguration>
<providers>
<provider id="1 type="refreshAndPersist" timeLimit="1000" sizeLimit="1000">
<url>ldap://ldap1.acme.com:10389/ou=data,dc=acme,dc=com?*, +?sub?(objectClass=*)</url>
<connection bindMethod="simple">
<principal>uid=admin,ou=system</principal>
<credentials>secret</credentials>
</bind>
</provider>
<provider id="2 type="refreshAndPersist" timeLimit="1000" sizeLimit="1000">
<url>ldaps://ldap2.acme.com:10389/ou=data,dc=acme,dc=com?*, +?sub?(objectClass=*)</url>
<connection bindMethod="simple">
<principal>uid=admin,ou=system</principal>
<credentials>secret</credentials>
</bind>
</provider>
</providers>
</replicationConfiguration-->

</defaultDirectoryService>

<!--
+============================================================+
| ChangePassword server configuration |
+============================================================+
-->
<!-- missing atou=users,dc=example,dc=com
<changePasswordServer id="changePasswordServer">
<transports>
<tcpTransport port="60464" nbThreads="2" backLog="50"/>
<udpTransport port="60464" nbThreads="2" backLog="50"/>
</transports>
<directoryService>#directoryService</directoryService>
</changePasswordServer>
-->

<!--
+============================================================+
| Kerberos server configuration |
+============================================================+
-->
<!-- missing atou=users,dc=example,dc=com
<kdcServer id="kdcServer">
<transports>
<tcpTransport port="60088" nbThreads="4" backLog="50"/>
```

```
<udpTransport port="60088" nbThreads="4" backLog="50"/>
</transports>
<directoryService>#directoryService</directoryService>
</kdcServer>
-->

<!--
+============================================================+
| NtpServer configuration |
+============================================================+
-->
<!--ntpServer>
<transports>
<tcpTransport port="60123"/>
<udpTransport port="60123" nbThreads="1"/>
</transports>
</ntpServer-->

<!--
+============================================================+
| DnsServer configuration |
+============================================================+
-->
<!-- missing atou=users,dc=example,dc=com
<dnsServer>
<transports>
<tcpTransport port="8053"/>
<udpTransport port="8053"/>
</transports>
<directoryService>#directoryService</directoryService>
</dnsServer>
-->

<!--
+============================================================+
| LDAP Service configuration |
+============================================================+
-->

<ldapServer id="ldapServer" allowAnonymousAccess="false" saslHost="ldap.example.com"
saslPrincipal="ldap/ldap.example.com@EXAMPLE.COM" searchBaseDn="ou=users,ou=system"
maxTimeLimit="15000" maxSizeLimit="1000">
<transports>
<tcpTransport address="0.0.0.0" port="10389" nbThreads="8" backLog="50" enableSSL="false"/>
<tcpTransport address="0.0.0.0" port="10636" nbThreads="8" backLog="50" enableSSL="true"/>
</transports>
<directoryService>#directoryService</directoryService>

<!-- The list of supported authentication mechanisms. -->
<saslMechanismHandlers>
<simpleMechanismHandler mech-name="SIMPLE"/>
<cramMd5MechanismHandler mech-name="CRAM-MD5" />
<digestMd5MechanismHandler mech-name="DIGEST-MD5" />
<gssapiMechanismHandler mech-name="GSSAPI" />
<ntlmMechanismHandler mech-name="NTLM" ntlmProviderFqcn="com.foo.Bar"/>
<ntlmMechanismHandler mech-name="GSS-SPNEGO" ntlmProviderFqcn="com.foo.Bar"/>
```

```
</saslMechanismHandlers>

<!-- The realms serviced by this SASL host, used by DIGEST-MD5 and GSSAPI. -->
<saslRealms>
<s:value>example.com</s:value>
<s:value>apache.org</s:value>
</saslRealms>

<!-- the collection of extended operation handlers to install -->
<extendedOperationHandlers>
<startTlsHandler/>
<gracefulShutdownHandler/>
<launchDiagnosticUiHandler/>
<!-- The Stored Procedure Extended Operation is not stable yet and it may cause security risks.-->
<!--storedProcedureExtendedOperationHandler/-->
</extendedOperationHandlers>
</ldapServer>

<apacheDS id="apacheDS">
<ldapServer>#ldapServer</ldapServer>
</apacheDS>

<!-- uncomment the below line to start the jetty(v6.1.14) http server This can be used to provide access to the data present in DIT via http using a web application -->
<!--
<httpServer id="httpServer" port="7009" >
<webApps>
<webApp warFile="/path/to/war/file" contextPath="/myApp"/>
</webApps>
</httpServer>
-->

</spring:beans>
```