



Universidad de las Ciencias Informáticas

Facultad 1

SUBSISTEMA DE APROVISIONAMIENTO DE USUARIOS PARA EL SISTEMA DE ADMINISTRACIÓN DE IDENTIDADES

Trabajo de Diploma para optar por el Título de
Ingeniero en Ciencias Informáticas.

Autores: José Carlos Correa Bautista.
Yendry Machado García.

Tutores: Ing. Yanisleivi Valdés Fernández.
Ing. Wilhem Manuel Verano Escalona.

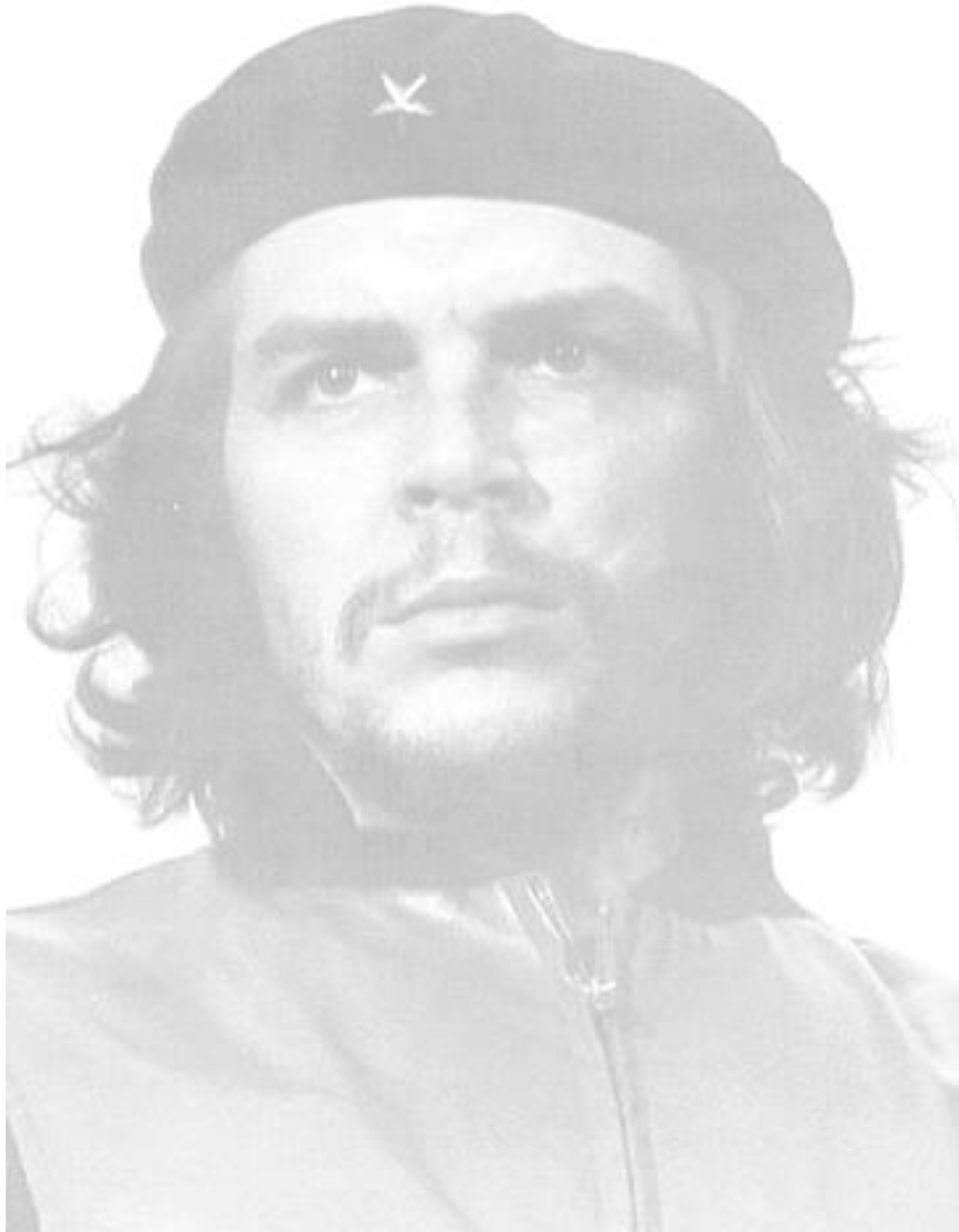
Consultor: Ing. Roberto Quiñones Bondartchuk.

Ciudad de la Habana, 14 de junio del 2012.
"Año 54 de la Revolución"





*“Subsistema de Provisiónamiento de Usuarios para el Sistema de
Administración de Identidades”*



*“El futuro de nuestra patria tiene que ser necesariamente un futuro de hombres de ciencia,
tiene que ser un futuro de hombres de pensamiento, porque precisamente es lo que más
estamos sembrando; lo que más estamos sembrando son oportunidades a la
inteligencia...”*

Se



*“Subsistema de Provisiónamiento de Usuarios para el Sistema de
Administración de Identidades”*

DECLARACIÓN DE AUTORÍA.

Declaro que somos los únicos autores de este trabajo y autorizamos a la facultad 1 de la Universidad de las Ciencias Informáticas a hacer uso del mismo en su beneficio.

Para que así conste firmamos la presente a los ____ días del mes de _____ del año _____.

José Carlos Correa Bautista

Yendry Machado García

(Autor)

(Autor)

Yanisleivi Valdés Fernández

Wilhem Manuel Verano Escalona

(Tutor)

(Tutor)



“Subsistema de Aprovechamiento de Usuarios para el Sistema de
Administración de Identidades”

AGRADECIMIENTOS.

Agradezco profundamente a mis padres que son lo más grande e importante que tengo en la vida, por sus esfuerzos, su dedicación, su cariño y su amor. Cada día le agradezco a dios por haberme dado unos padres tan excepcionales y maravillosos que siempre se han preocupado porque yo sea quien quiera ser y porque cumpla mis sueños, si volviera a nacer lo único que pediría no sería riquezas ni poder sino a los mismos padres que tengo de los cuales no tengo ya palabras para expresar el amor que siento por ellos. También quiero agradecer a mi novia linda que siempre estuvo a mi lado consintiéndome, ayudándome y aguantando los arranques de mi carácter. A mi familia que es pequeña pero unida, a mis tíos y tías, a mi primo querido que siempre está al tanto de mí, a mis primitas, a la única abuela que me queda que es la súper abuela y a Aliuci. Además a mis suegros y familia por parte de mi novia que siempre están preocupados por mí. Agradecer a mi compañero de tesis que ha trabajado duro junto conmigo sin descanso para lograr una tesis como la que hoy presentamos, a mis amigos que de una forma u otra me ayudaron a discutir un tema o perfeccionar cosas.

José Carlos.

Bueno ha llegado el momento en el que se cierran 5 años de estudio y sacrificio, es el momento en que quisiera agradecer a todos los que de una forma u otra ayudaron a que este sueño se volviera realidad. Primero y de manera especial le quiero agradecer a mi familia que siempre han sido mi apoyo para seguir adelante. A mi mamá por ser mi guía y soporte en todo momento, por todo el cariño y la confianza que he recibido de ella. A mi hermana, por ser la mejor de todas y siempre estar ahí para mí. A mis sobrinas ya que su mera existencia me ilumina el camino. A mi papá por toda su ayuda y confianza que me ha brindado durante toda mi vida. Al resto de la familia que siempre me han apoyado y creído en mí, a mis abuelos, a mis tíos, a mis primos, a mi cuñado y todos esos que aunque la sangre no nos une si lo hacen los sentimientos. A mi compañero de tesis por aguantarme todo este año y trabajar cada día para que la tesis saliera adelante. A todos mis amigos que también jugaron un papel fundamental en este logro pues convirtieron momentos tristes en alegrías, a los que tenía antes de entrar a la universidad, a todos los que encontré al entrar aquí. A la vieja guardia, el grupo de primer año que pasamos muchos buenos momentos juntos, a todos los que vinieron después y se convirtieron en parte importante en mi vida. A todos gracias.

Yendry.

Agradecemos a todos aquellos que de una forma u otra ayudaron al desarrollo de este trabajo, a nuestros tutores que fueron nuestras manos derechas en todo este período, a Roberto nuestro tutor moral como el mismo se autonombró, a todos los profes del proyecto que siempre nos tendieron su mano dándonos consejos y respondiendo preguntas a toda hora.

Autores.



*“Subsistema de Aprovisionamiento de Usuarios para el Sistema de
Administración de Identidades”*

DEDICATORIA.

Muchas personas dicen que nací con una estrella en la frente, yo digo que nací de dos estrellas que iluminan el camino por donde voy. Por eso dedico esta tesis a esas dos relucientes estrellas que son mis padres hermosos pues sin ellos no lo hubiese logrado. Dedico también mi trabajo a mi familia, a mi novia y a mis seres queridos que por leyes de la vida no están aquí para disfrutar de este momento de alegría.

José Carlos.

Quisiera dedicarles este momento muy especial a mi papá y a las 4 mujeres de mi vida; mi mamá, mi hermana y mis dos sobrinitas, que por todo su apoyo y cariño siempre tendrán un lugar especial en mi corazón. A toda mi familia por creer en mí y estar cuando los he necesitado.

Yendry.

Dedicamos esta tesis a nuestros familiares, amigos. A la revolución por darnos la oportunidad de estar aquí y al natalicio de dos grandes mártires de esta patria Ernesto Che Guevara y Antonio Maceo.

Autores.



*“Subsistema de Aprovisionamiento de Usuarios para el Sistema de
Administración de Identidades”*

RESUMEN.

La presente investigación tiene como principal objetivo desarrollar un Subsistema para el Aprovisionamiento de Usuarios en el Sistema de Administración de Identidades. Surge debido a la necesidad de mejorar la gestión del ciclo de vida de las cuentas de usuarios y la administración de los recursos asociados al Sistema de Administración de Identidades. Para lograr el cumplimiento del objetivo de la investigación, se llevó a cabo un estudio de diferentes sistemas de administración de identidades existentes, enfocándose fundamentalmente en la estructura y el funcionamiento de sus sistemas de aprovisionamiento de usuarios, para adquirir mayor visión en cuanto a las ventajas y desventajas que presentan dichos sistemas y más adelante realizar una propuesta de solución. Además se analizaron herramientas y tecnologías a utilizar, así como la definición de los artefactos generados a partir de la metodología MSF para el desarrollo de *software* ágil.

A partir de la realización de este Subsistema se obtuvieron diferentes resultados como son la gestión de plantillas para la creación de cuentas de usuarios, que permiten la creación de tipos de usuarios agrupados por roles, recursos a los que tienen acceso y atributos de identidad. También se puede apreciar la gestión de recursos a partir de conectores, donde se permite definir al sistema cuales son los recursos donde realizará el aprovisionamiento de usuarios. Además cuenta con un proceso de solicitud guiado por un flujo de trabajo que permite la flexibilidad del mismo. Con el desarrollo del Subsistema se mejora la gestión del ciclo de vida de las cuentas de usuarios y la administración de los recursos asociados al Sistema de Administración de Identidades.

Palabras clave: administración de identidades, aprovisionamiento de usuarios, cuentas, metodologías, plantillas, servicios, *workflow*.



“Subsistema de Aprovisionamiento de Usuarios para el Sistema de
Administración de Identidades”

ÍNDICE.

INTRODUCCIÓN.	1
CAPÍTULO 1 FUNDAMENTACIÓN TEÓRICA	6
1.1 Introducción.	6
1.2 Conceptos fundamentales.	6
1.2.1 Seguridad Informática.	6
1.2.2 Identidad.	7
1.2.3 Identidad Digital.	7
1.2.4 Administración.	8
1.2.5 Sistemas Gestión de Identidades.	8
1.2.6 Aprovisionamiento de Usuarios.	9
1.2.7 Motor de Aprovisionamiento.	10
1.2.8 Motor de Tareas.	11
1.3 Análisis de los Sistemas de Aprovisionamiento de Usuarios.	11
1.3.1 Gestor de Aprovisionamiento Tivoli de IBM 7.2.1.	12
1.3.2 Sistema de Administración de Identidades Oracle11g.	13
1.3.3 Sistema de Aprovisionamiento de Microsoft.	13
1.3.4 Sistema de Administración de Identidades de Novell Edición Estándar 4.	15
1.3.5 Servidor de Identidades WSO2.	16
1.3.6 Módulo de Administración para la aplicación de aprovisionamiento de usuarios del Sistema de Administración de Identidades.	17
1.3.7 Gestor de Tareas para el aprovisionamiento de usuarios.	18
1.3.8 Librería de Conectores.	18
1.4 Investigación sobre las principales herramientas, tecnologías y lenguajes a utilizar.	19
1.4.1 .Net Framework 4.	19
1.4.2 ASP.net MVC 3.	20
1.4.3 Entity Framework 4.1.	21
1.4.4 Windows Workflow Foundation.	22
1.4.5 Windows Communication Foundation.	23



*“Subsistema de Aprovisionamiento de Usuarios para el Sistema de
Administración de Identidades”*

1.4.6 Lenguaje de programación C Sharp	23
1.4.7 IDE de desarrollo Visual Studio 2010.....	24
1.4.8 Sistema Gestor de Base de Datos Oracle 11g.....	25
1.4.9 Lenguaje Unificado de Modelado (UML).....	25
1.4.10 Altova MissionKit 2011.....	26
1.4.11 PLSQL Developer 7.1.4.1390.	27
1.4.12 Metodología de Desarrollo MSF for Agile Software Development.	28
1.5 Conclusiones.	30
CAPÍTULO 2 VISIÓN Y PLANIFICACIÓN	31
2.1 Introducción.	31
2.2 Capturar visión del proyecto.....	31
2.2.1 Descripción del Subsistema.....	31
2.2.2 Definición de las Personas.	32
2.2.3 Propuesta de Solución.....	33
2.3 Planificación.....	35
2.3.1 Escenarios del Subsistema.....	35
2.3.2 Priorización de Escenarios.	37
2.3.3 Plan de Iteraciones.	38
2.3.4 Cronometrar Escenarios.	39
2.3.5 Requisitos de Calidad de Servicio.....	40
2.3.6 Descripción de los Escenarios.....	40
2.4 Conclusiones.	41
CAPÍTULO 3 CONSTRUCCIÓN	42
3.1 Introducción.	42
3.2 Especificación de la Arquitectura.	42
3.2.1 Modelo de Capas.....	42
3.2.2 Modelo Vista Controlador (MVC).	45
3.3 Patrones de Diseño.	46
3.3.1 Patrón Experto.....	46



“Subsistema de Aprovisionamiento de Usuarios para el Sistema de
Administración de Identidades”

3.3.2 Patrón Creador.....	47
3.3.3 Patrón Controlador.....	47
3.3.4 Patrón Bajo Acoplamiento.....	48
3.3.5 Patrón Alta Cohesión.....	48
3.3.6 Patrón Repositorio.....	48
3.3.7 Patrón Observador.....	49
3.3.8 Patrón Recuerdo.....	50
3.4 Diagrama de Aplicación.....	50
3.5 Diagrama Lógico de Centro de Datos.....	50
3.6 Diagrama de Clases.....	51
3.7 Uso de <i>Windows Workflow Foundation</i>	51
3.8 Modelo de Datos del Sistema.....	52
3.9 Interfaz Gráfica.....	53
3.10 Conclusiones.....	53
CAPÍTULO 4 PRUEBA.....	54
4.1 Introducción.....	54
4.2 Pruebas de Caja Blanca.....	54
4.3 Pruebas Unitarias.....	54
4.4 Pruebas de Caja Negra.....	56
4.6 Pruebas de Rendimiento.....	58
4.7 Conclusiones.....	60
CONCLUSIONES GENERALES.....	61
RECOMENDACIONES.....	62
REFERENCIAS BIBLIOGRÁFICAS.....	63
GLOSARIO DE TÉRMINOS.....	66
ANEXOS.....	71

ÍNDICE DE FIGURAS.

Figura 1: Atributos que componen una Identidad Digital.....	8
Figura 2: Procesos comprendidos dentro de la Administración de Identidades.....	9



*“Subsistema de Aprovisionamiento de Usuarios para el Sistema de
Administración de Identidades”*

Figura 3: Aprovisionamiento de Usuarios.	10
Figura 4: Propuesta de Solución.....	33
Figura 5: Estructura del Motor de Tareas.	34
Figura 6: Primera Iteración.	39
Figura 7: Segunda Iteración.	39
Figura 8: Tercera Iteración.	39
Figura 9: Vista lógica de la arquitectura del <i>software</i>	44
Figura 10: Ejemplo Patrón Experto.....	47
Figura 11: Ejemplo Patrón Creador.	47
Figura 12: Ejemplo Patrón Controlador.....	47
Figura 13: Ejemplo Patrón Bajo Acoplamiento.....	48
Figura 14: Ejemplo Patrón Alta Cohesión.	48
Figura 15: Diagrama Lógico de Centro de Datos del Subsistema de Aprovisionamiento.	51
Figura 16: Diagrama de Clases del Modelo de Datos.....	53
Figura 17: Servidor #1.	60
Figura 18: Servidor #2.	60

ÍNDICE DE TABLAS.

Tabla 1: Descripción de las personas.	33
Tabla 2: Lista de grupos de escenarios y sus prioridades.....	38
Tabla 3: Planificación de los escenarios.	39
Tabla 4: Descripción del Escenario Crear Conector.....	41
Tabla 5: Descripción de la prueba unitaria AddTemplateTest.	55
Tabla 6: Descripción de la prueba unitaria ModifyTemplateDataTest.....	56
Tabla 7: Descripción de la prueba unitaria RemoveTemplateTest.	56
Tabla 8: Caso de Prueba Gestionar Políticas de Contraseña.	57
Tabla 9: Descripción de las variables de entrada del Caso de Prueba Gestionar Políticas de Contraseña.....	58
Tabla 10: Características de los Servidores.....	59
Tabla 11: Resultados de la Prueba de Carga.	59



INTRODUCCIÓN.

El surgimiento y evolución de las tecnologías de la información y las comunicaciones han dado paso a un nuevo tiempo que sin duda marcará una pauta en la historia de la humanidad, la era de la información. Este es el período en el cual el movimiento y almacenamiento de los datos digitales se vuelve más rápido que el movimiento de datos físicos, ventaja que motivó a las grandes organizaciones a automatizar todos sus procesos. Debido al uso de información digital sensible, se hace necesaria su protección informática, donde se ha vuelto cada vez más crítico salvaguardar las bases de datos, las aplicaciones y la información personal, del uso malintencionado de los usuarios así como del espionaje y sabotaje de intrusos. Es precisamente en ese marco donde la Administración de Identidades (IDM, por sus siglas en inglés) que comprende procesos como la Autenticación, Autorización, Auditoría y Aprovisionamiento de Usuarios, juega un papel fundamental. Kevin Kampman¹ define IDM como “...un conjunto de procesos, y un soporte de infraestructura, para la creación, mantenimiento y uso de las identidades digitales...”. Para Archie Reed² la esencia de un IDM es “...proveer una combinación de procesos y tecnologías para gestionar y asegurar el acceso a la información y los recursos de una organización mientras también se protegen los perfiles de los usuarios...”. No obstante, estas definiciones tienen un elemento en común, la gestión de los datos que se relacionan con las identidades individuales, ya que pretende facilitar y controlar el acceso a los sistemas de información y a las instalaciones(MICROSOFT 2004).

Cuando la Administración de Identidades comenzó a destacarse a inicios del año 2000 las organizaciones se enfrentaron a un conjunto de desafíos, ya que en esa época, los impulsores comerciales más importantes de la Administración de Identidades eran las necesidades de aumentar la productividad a bajos costos. En la actualidad, el alto grado de productividad y el bajo costo siguen estando presentes, pero también se han agregado algunos fenómenos relativamente nuevos como la variedad de normas que gobiernan la integridad y privacidad de los datos, además de la importancia que ha cobrado el acceso a la información en la actual sociedad(SUN 2006a). Generalmente las grandes organizaciones que han automatizado sus procesos, presentan dificultades en cuanto a su completitud, así como sus diseños poco satisfactorios, especialmente para aprovisionar usuarios en los recursos disponibles en la organización que pueden ser: bases de datos, servidores de correo o servidores ftp. Dentro de este problema la

¹ Kevin Kampman: MSc. Ciencias de la Computación, Universidad de Dayton. Licenciado en Bellas Artes y Fotografía, Universidad de Ohio.

² Archie Reed: MSc. Gestión de Redes Empresariales, Universidad de Sheffield Hallam.



“Subsistema de Aprovisionamiento de Usuarios para el Sistema de Administración de Identidades”

principal variable es el tiempo de espera para que los usuarios tengan acceso a las aplicaciones necesarias en su desempeño laboral, además del costo agregado que trae consigo la realización manual de un conjunto de actividades para el aprovisionamiento de usuarios en determinado recurso. También existe la poca flexibilidad que poseen los sistemas para la gestión de sus recursos, puesto que algunas organizaciones por razones económicas o estratégicas pueden cambiar las tecnologías o los sistemas donde persisten sus informaciones, necesitando así una nueva perspectiva para la creación de cuentas de usuarios. Los sistemas de aprovisionamiento de usuarios han sido la solución a todos estos problemas de administración del ciclo de vida de las cuentas de usuarios en las organizaciones, reduciendo en tiempo y dinero, la concesión y renovación de las cuentas de usuarios debido a que este proceso se realiza de manera centralizada.

En el mundo existen numerosas herramientas para la Administración de Identidades tanto libres como privativas dentro de las cuales se pueden resaltar dentro de las soluciones privadas, las pertenecientes a Novell, IBM y Oracle que cuentan con un gran prestigio a nivel internacional, así como dentro de las soluciones libres WSO2. En nuestro país la necesidad de un IDM también se encuentra vigente debido a que en la investigación realizada no se encontró ningún sistema de tal envergadura. A partir de la creación de varios proyectos del Centro de Identificación y Seguridad Digital (CISED), perteneciente a la Universidad de las Ciencias Informáticas (UCI), se han realizado determinadas colaboraciones con el Ministerio del Interior (MININT). Dicho ministerio presenta problemáticas como son la no existencia de un marco de seguridad común para aplicaciones empresariales dentro de sus redes, la diversidad de almacenes de políticas de acceso asociadas a dichas aplicaciones, las labores de aprovisionamiento de cuentas de usuarios en los recursos realizadas de manera manual y descentralizadas, y la existencia de aplicaciones sin la utilización de estándares internacionales de seguridad. Debido a estas limitaciones, surge la necesidad de contar con un sistema que estandarice y controle la autenticación, autorización y aprovisionamiento de los usuarios en los sistemas y recursos de las redes del MININT(CISED 2012). La idea de desarrollar un IDM propio surge debido a la necesidad del ministerio, de mantener el control sobre cada fragmento de código por cuestiones de seguridad. Siguiendo de esta manera, el objetivo de alcanzar una soberanía tecnológica que permita a nuestro país elegir libremente y sin condicionamientos externos el rumbo al que prefiere orientar su desarrollo informático. Es por ello que en el Centro de Identificación y Seguridad Digital (CISED), se está desarrollando el Sistema Administración de Identidades, que se encargará de los procesos de Identificación, Autenticación, Auditoría y Aprovisionamiento de Usuarios.

INTRODUCCIÓN



“Subsistema de Aprovisionamiento de Usuarios para el Sistema de Administración de Identidades”

Dentro de este proyecto se encuentra el Subsistema de Aprovisionamiento de Usuarios, elemento clave en las soluciones de Administración de Identidades. Este encierra la gestión del ciclo de vida de las cuentas de usuarios, aportando ventajas sobre la información del usuario en la infraestructura de los directorios de las organizaciones.

Para el desarrollo del Subsistema de Aprovisionamiento de Usuarios se revisaron las investigaciones correspondientes a los módulos Portal de Administración, Motor de Tareas y la Librería de Conectores. En estos trabajos se han desarrollado soluciones para la gestión del proceso de solicitud de las cuentas de usuarios y el aprovisionamiento en OpenLdap. Las mismas poseen una arquitectura rígida con poca adaptabilidad a cambios, ya que no permiten aprovisionar en otros recursos, como por ejemplo Oracle11g, Zimbra y OpenFire, además de que no se utiliza la Librería de Conectores. En el proceso de solicitud de cuentas de usuarios se tienen en cuenta atributos de identidad estáticos, impidiendo así que el proceso sea genérico. Tampoco cuenta con una web para los usuarios estándares, revisores de solicitudes y aprobadores de solicitudes, donde estos puedan realizar acciones sobre el sistema. Además de que estas soluciones no están en correspondencia con las necesidades del proyecto, dentro de las cuales se pueden encontrar el aprovisionamiento de usuarios en más de un recurso, la gestión de atributos de identidad y el uso de la Librería de Conectores. Por estas razones se decidió que el Subsistema de Aprovisionamiento de Usuarios para el Sistema de Administración de Identidades se desarrolle nuevamente.

Problema a resolver

¿Cómo mejorar la gestión del ciclo de vida de las cuentas de usuarios y la administración de los recursos asociados al Sistema de Administración de Identidades?

Objeto de estudio

Proceso de Aprovisionamiento de Usuarios.

Campo de Acción

Proceso de Aprovisionamiento de Usuarios en el Sistema de Administración de Identidades del departamento de Seguridad Digital.

Objetivo general

Desarrollar un Subsistema para el Aprovisionamiento de Usuarios en el Sistema de Administración de Identidades, que permita la gestión dinámica de las cuentas de usuarios y los recursos asociados de forma genérica.

INTRODUCCIÓN



*“Subsistema de Aprovisionamiento de Usuarios para el Sistema de
Administración de Identidades”*

Para dar cumplimiento al objetivo planteado se organizan las **tareas de investigación** siguientes:

1. Elaboración del marco teórico.
2. Identificación de las necesidades del sistema a desarrollar a partir del análisis de diferentes Sistemas de Administración de Identidades.
3. Investigación sobre las herramientas y tecnologías a utilizar.
4. Diseño del Subsistema de Aprovisionamiento de Usuarios.
5. Implementación del módulo Portal de Usuario en la plataforma establecida.
6. Implementación del módulo Portal de Administración en la plataforma establecida.
7. Implementación del módulo Motor de Tareas en la plataforma establecida.
8. Realización de pruebas.

Métodos Científicos.

Para el desarrollo de esta investigación y el logro de su objetivo, así como sus tareas específicas, se utilizan métodos teóricos y empíricos, a través de los cuales se obtiene una idea más clara y concisa de lo que se quiere llevar a cabo:

Teóricos:

Histórico - Lógico: se investigó la manera en que se realiza el proceso de aprovisionamiento de usuarios en diferentes sistemas desde sus inicios, obteniéndose así las características y aspectos negativos y positivos del proceso en cuestión, arribando a la conclusión de la necesidad de realizar un sistema que facilite este proceso con la arquitectura seleccionada.

Análisis - Síntesis: se revisa y analiza la bibliografía encontrada referente a este tema y se sintetizan los aspectos más importantes para la investigación.

Empíricos:

Observación: se observó exhaustivamente el funcionamiento real en los procesos de aprovisionamiento de usuarios dentro de diferentes sistemas, identificando así los problemas que se deben resolver.

Tomando en consideración el valor metodológico y práctico de esta investigación se obtienen aportes como: mejoras notables en el Sistema Administración de Identidades en cuanto a su proceso de Aprovisionamiento de Usuarios.



“Subsistema de Aprovisionamiento de Usuarios para el Sistema de Administración de Identidades”

Justificación de la Investigación.

Debido a la necesidad de resolver las problemáticas planteadas para mejorar el Aprovisionamiento de Usuarios, se decidió desarrollar un Subsistema de Aprovisionamiento de Usuarios acorde con las nuevas modificaciones del proyecto y que mejore el aprovisionamiento de usuarios del mismo. La solución brindará todas las funcionalidades con que contaba anteriormente. Además permitirá el aprovisionamiento de usuarios en varios recursos utilizando la Librería de Conectores, así como la posibilidad de agregarle conectores nuevos para lograr el aprovisionamiento en recursos que no estén contenidos en la aplicación. También se podrán utilizar plantillas para la creación de cuentas de usuarios, que permiten la creación de tipos de usuarios agrupados por roles, recursos a los que tienen acceso y atributos de identidad, para darle la capacidad al sistema de adaptarse mejor a la organización donde se utilice. Otra característica es la adición del Portal de Usuarios, a través del cual los usuarios estándares, revisores de solicitud y aprobadores de solicitud puedan realizar acciones sobre el Subsistema.

Estructura del Sistema.

Capítulo 1 Fundamentación Teórica: se realizará la investigación y el estudio del estado del arte tomando en consideración las aplicaciones más importantes inmersas en el aprovisionamiento de usuarios. A partir de dicha búsqueda se mostrarán los aspectos y características fundamentales que puedan ser útiles o sirvan de cimiento para la propuesta de solución del Subsistema que se plantea. También se llevará a cabo un análisis general en cuanto a las herramientas, lenguajes de programación y tecnologías que serán usadas durante el proceso de desarrollo de la aplicación.

Capítulo 2 Visión y Planificación: en este capítulo se determinarán las actividades que serán realizadas por los usuarios que intervendrán en el Subsistema y se realizará la especificación de las funcionalidades que tendrá el mismo. Se desarrollará además, el diseño de la solución en el cual se encontrarán los elementos y aspectos necesarios para la posterior implementación del Subsistema.

Capítulo 3 Construcción: se establecerán los patrones de diseño que serán utilizados en el Subsistema, además se especificará la arquitectura del mismo. También se obtendrán elementos como el diagrama de clases y el diseño del modelo de base de datos, así como detalles de la implementación.

Capítulo 4 Prueba: en este capítulo se realizarán una serie de pruebas unitarias al Subsistema para validar y verificar la calidad del mismo, además de pruebas de estrés a los servicios y la aplicación web. Se obtendrá también un resumen acerca de la evaluación de estas pruebas.



Capítulo 1 Fundamentación Teórica

1.1 Introducción.

El desarrollo de las tecnologías informáticas ha propiciado la creación de soluciones y productos de todo tipo. Las herramientas para la gestión de identidades digitales no quedan fuera de este ámbito, teniendo un impacto importante en cuanto a tiempo y dinero en empresas u organizaciones donde se implanten. Este tipo de sistemas se corresponden a la infraestructura técnica y organizativa utilizada para la definición, selección y administración de los atributos de una entidad (MARIO, REYES DE LOS MOZOS 2007), además estos sistemas tienen implícitos procesos fundamentales como la autenticación, autorización, auditoría y aprovisionamiento de usuarios. A nivel global existen numerosas soluciones de este tipo, y para el desarrollo del Subsistema de Aprovevisionamiento de Usuarios en el Sistema de Administración de Identidades, es necesario como primer paso el estudio de las soluciones existentes, las tecnologías y herramientas a utilizar.

1.2 Conceptos fundamentales.

Para un mejor entendimiento de la investigación, se especificarán un conjunto de conceptos importantes relacionados con el tema de la gestión de identidades.

1.2.1 Seguridad Informática.

La seguridad informática es una disciplina que se encarga de proteger la integridad y la privacidad de la información almacenada en un sistema informático. La misma pretende garantizar tres aspectos fundamentales: la **confidencialidad**, que no es más que permitir el acceso a la información o activos informáticos solo a los usuarios autorizados para hacerlo, la **integridad** donde los activos o la información solo pueden ser modificados por los usuarios autorizados de la manera autorizada y la **disponibilidad** que consiste en que los activos informáticos puedan ser accedidos por los usuarios autorizados en el momento requerido (UCI 2011).



1.2.2 Identidad.

La identidad personal desde el punto de vista legal está formada, tradicionalmente, por el conjunto de datos resultantes de la unión de la información relativa a una persona presente en los registros públicos, que va a permitir identificarla de forma unívoca (SERGIO and ANA 2009). También se puede decir que está formada por un conjunto de características, datos o informaciones que son propias de una persona o un grupo y que permiten diferenciarlos del resto (ACADEMIA 2010).

La identidad de una persona está dada principalmente de manera tradicional por el nombre y los apellidos. Estos dos elementos son los más utilizados para distinguir a una persona y tienen que figurar como tales en todos los documentos, formularios y registros que puedan haber oficialmente. Sin embargo, hoy en día debido a la gran cantidad de población mundial existente y a la informatización de la sociedad, también deben tenerse en cuenta otros elementos más específicos y detallados, para poder diferenciar a personas con el mismo nombre. Elementos tales como las huellas dactilares y otras cuestiones biológicas sirven para distinguir a unas personas de otras. Todos ellos son utilizados por lo general a nivel administrativo o legal para registrar la existencia de los individuos oficialmente.

1.2.3 Identidad Digital.

De acuerdo a diferentes fuentes bibliográficas, al conjunto de rasgos que caracterizan a un individuo o colectivo en un medio de transmisión digital se le conoce como Identidad Digital. La identidad digital no existe a priori, se debe crear y vincular unívocamente al individuo o al colectivo (EVOLUCY 2012). También se puede expresar que la identidad digital de un usuario está constituida por el total de los conjuntos de atributos que constituyen las distintas cuentas de un usuario. Para una entidad dada, típicamente existirán múltiples identidades digitales que pueden ser únicas o no. Una identidad digital es, por definición, un subconjunto de la identidad y puede ser considerada como la manifestación de la entidad en la red (SERGIO and ANA 2009). Otro concepto similar es que la identidad digital es la personalidad que transmite una empresa o profesional a través de la red, en su web corporativa, blog de empresa, en su presencia en redes sociales y en la web en general. Cualquier empresa o profesional comunica, de forma consciente o inconsciente, valores, atributos y rasgos propios que lo definen y diferencian de su competencia (UZKIAGA 2012).

A partir de las anteriores definiciones, el concepto más básico de identidad digital es el conjunto de atributos personales o grupales que se realizan en Internet. Estas expresiones son fruto de la capacidad



“Subsistema de Aprovechamiento de Usuarios para el Sistema de Administración de Identidades”

de editar en diferentes espacios web, redes sociales, perfiles profesionales, blogs, páginas de empresas, comentarios que se realizan en noticias, opiniones en foros, etc. Juntando ese conjunto de diminutas piezas se tendrá al final un rompecabezas que compondrá quienes son y qué hacen, la identidad digital.

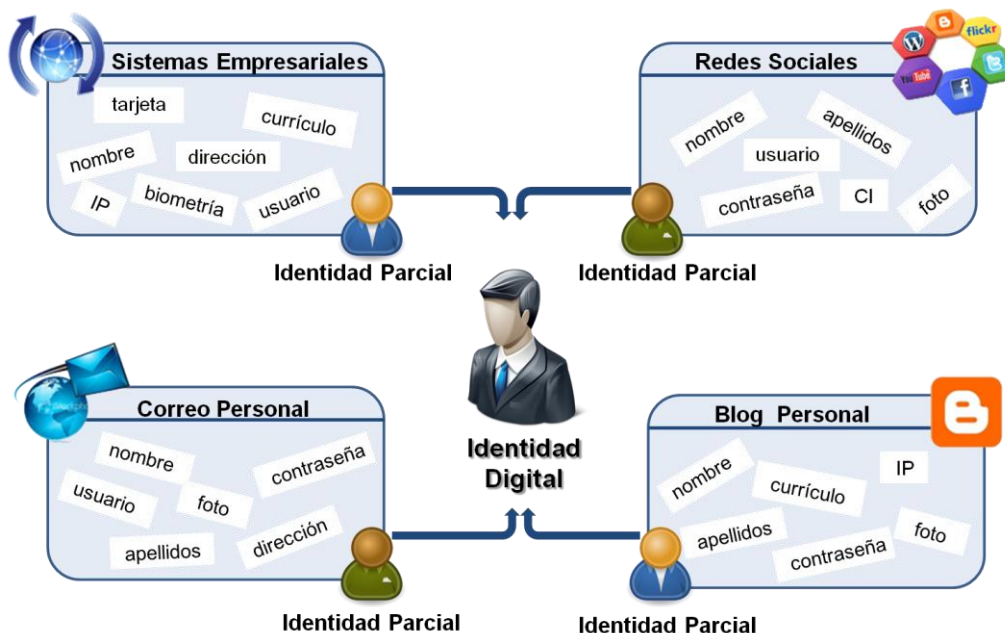


Figura 1: Atributos que componen una Identidad Digital.

1.2.4 Administración

La administración es el proceso de planear, organizar, liderar y controlar el trabajo de los miembros de la organización además de utilizar todos los recursos de la empresa para lograr objetivos organizacionales específicos (JAMES 1994). La administración en los sistemas de aprovisionamiento de usuarios hace posible establecer las políticas de contraseñas, la configuración de las conexiones a los recursos, así como la gestión de los mismos. También permite la creación, inhabilitación y eliminación de cuentas de usuarios, dándole a estos los permisos necesarios para su desempeño laboral.

1.2.5 Sistemas Gestión de Identidades.

Luego de expresar los conceptos anteriores, se puede especificar lo que se considera un Sistema de Gestión o Administración de Identidades, que no es más que la infraestructura técnica y organizativa para la definición, gestión y administración de atributos de identidad (SERGIO and ANA 2009). La gestión de la



“Subsistema de Aprovisionamiento de Usuarios para el Sistema de Administración de Identidades”

identidad y del acceso, es un área de negocio que se dedica al aprovisionamiento de cuentas de usuarios y contraseñas, mediante automatismos, de acuerdo con políticas bien definidas y aplicadas. Comprende la implantación de sistemas de identificación y autenticación única, así como la gestión centralizada de las atribuciones de los usuarios, basada en directorios. También es un modelo de autorizaciones, que concentra en un solo punto las autorizaciones de acceso (IGNACIO and XAVIER 2006).

Dentro de este tipo de sistemas existen procesos como son la Autenticación, Autorización, Auditoría y Aprovisionamiento de Usuarios.



Figura 2: Procesos comprendidos dentro de la Administración de Identidades.

1.2.6 Aprovisionamiento de Usuarios.

Como se plantea anteriormente, uno de los procesos pertenecientes a la administración de identidades es el Aprovisionamiento de Usuarios, en el cual se basa la presente investigación. Este proceso fundamental encierra la gestión del ciclo de vida de las cuentas de usuarios y se basa en crear, leer, modificar y eliminar la información de los usuarios en las cuentas de los recursos de una organización. Además facilita la gestión centralizada de los atributos de usuarios que se encuentran de manera local en cada recurso, también permite la comparación y el reporte de las cuentas que existen en los recursos específicos y que no están registradas, identificando diferencias entre las mismas. Otra de sus funcionalidades es la administración general de claves de acceso de las cuentas de usuarios, permitiendo establecer políticas de contraseñas (SUN 2006b).

Los sistemas de aprovisionamiento de usuarios generalmente están compuestos por determinados elementos como una interfaz de usuario, donde los mismos pueden chequear los contenidos de la base de datos interna, realizar solicitudes, aprobar o denegar cambios propuestos. Otro elemento es un motor



“Subsistema de Aprovisionamiento de Usuarios para el Sistema de Administración de Identidades”

de aprovisionamiento, que interpreta las solicitudes enviadas y las distribuye entre determinados recursos. Además se pueden encontrar el repositorio de datos, encargado de almacenar los perfiles de usuarios, las plantillas de cuentas, las políticas de cuentas, las definiciones del proceso de flujo de trabajo, el estado del flujo de trabajo, la información de auditoría, entre otros. También entre dichos elementos están los conectores, los cuales leen la información acerca de los usuarios de sistemas integrados y aplicaciones, además de realizar actividades de actualización como crear, modificar y eliminar cuentas de usuarios. Partiendo de lo anteriormente expresado, el aprovisionamiento de usuarios no es más que la gestión del ciclo de vida de las cuentas de usuarios y la administración de los recursos asociados a estos, valiéndose de componentes como por ejemplo un motor de aprovisionamiento, un portal web y conectores para la interacción con los recursos de la entidad en cuestión.

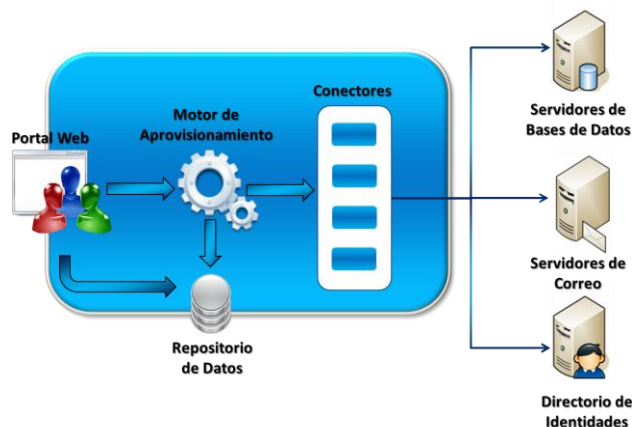


Figura 3: Aprovisionamiento de Usuarios.

1.2.7 Motor de Aprovisionamiento.

El Motor de Aprovisionamiento es uno de los componentes básicos existentes dentro de un sistema de aprovisionamiento de usuarios. Este sistema está destinado a ayudar a las organizaciones a agilizar los procesos del ciclo de vida de las cuentas de usuarios, actualizando cada cambio que ocurre respecto a los objetos relacionados con los usuarios de forma automática. Para alcanzar los objetivos, el motor de aprovisionamiento debe realizar uno o más procesos, los cuales pueden ser:

- **Auto-Aprovisionamiento:** encargado de la creación automática de usuarios en sistemas y aplicaciones cuando estos hacen su entrada en la organización.



“Subsistema de Aprovisionamiento de Usuarios para el Sistema de Administración de Identidades”

- **Auto-Desactivación:** desactiva objetos de usuarios, cuando estos ya no brindan sus servicios a la organización o cuando son marcados como inactivos. Desactiva automáticamente usuarios u objetos de usuarios, cuando el tiempo de cumplimiento de una tarea ha concluido.
- **Sincronización de Identidad:** cuando son realizados cambios en los atributos de un usuario, automáticamente este proceso replica dichos cambios en los sistemas o aplicaciones pertinentes.
- **Auto-Servicio de Cambio de Perfil:** permite a los usuarios actualizar su información del perfil.
- **Flujo de Trabajo de Autorización:** permite a los titulares del negocio revisar y aprobar o rechazar cambios propuestos a los perfiles de usuarios o a los derechos de acceso del mismo.
- **Certificación de Acceso:** pide periódicamente a los administradores que chequeen la lista de sus subordinados inmediatos, para verificar que aún son empleados de la organización o que aún son sus subordinados. Pide además a los propietarios de datos o aplicaciones que chequeen la lista de acceso de usuarios. (MARIO, MEZQUINA AZAHARES and CARLOS 2010)

1.2.8 Motor de Tareas.

A grandes rasgos un motor de tareas ejecuta determinados elementos a medida que recibe solicitudes para ello y no espera a que finalice un elemento en ejecución antes de iniciar otro (IBM 2012). Luego de definir el significado de motor de tareas, se puede ser más específico dentro de un sistema de gestión de identidades, donde está representado como motor de aprovisionamiento que interpreta las solicitudes enviadas y las distribuye entre los recursos apropiados (BHUVAN URGAONKAR 2008). Esto quiere decir que un motor de aprovisionamiento es un subconjunto de un motor de tareas, solo que se encarga específicamente de las acciones de aprovisionamiento de usuarios como son crear, eliminar o editar cuentas de usuarios en determinados recursos.

1.3 Análisis de los Sistemas de Aprovisionamiento de Usuarios.

Después de haber enunciado los conceptos fundamentales para la presente investigación, es necesario abordar en cuanto a soluciones existentes. Muchas empresas han apostado por el desarrollo de sistemas de administración de identidades, algunas de ellas han alcanzado grandes éxitos a nivel internacional como por ejemplo *Oracle*, *IBM*, *Microsoft* y *Novell*. A continuación se mencionarán diferentes funcionalidades y características de determinados productos de este tipo.



1.3.1 Gestor de Aprovisionamiento *Tivoli* de IBM 7.2.1.

El gestor de aprovisionamiento *Tivoli* de IBM forma parte del administrador de identidades *Tivoli* de IBM 7.2.1. Dicho sistema de aprovisionamiento consiste en un servidor de aprovisionamiento, una consola de operador y administrador basada en web y un entorno de desarrollo de paquetes de automatización(IBM 2011).

En el diagrama que se encuentra en el **Anexo I** se muestran los principales componentes de este sistema y además de que forma interactúan. Esos componentes se explican a continuación:

- **Servidor de Aprovisionamiento:** contiene elementos como son la base de datos de aprovisionamiento, modelo de datos, automatización, conformidad y remediación, reportes, descubrimientos e infraestructura de despliegue(IBM 2011).
- **Interfaz de Servicios Web:** mediante el uso de servicios web, incluyendo *framework* de recursos de servicios web (WSRF), se puede acceder al modelo de datos directamente sin necesidad de iniciar la interfaz web. También se pueden utilizar los servicios web para acceder, manipular o cambiar los objetos directamente en el modelo de datos(IBM 2011).
- **Consola de Operador y Administrador:** utilizando el operador basado en la web y la consola de administrador, se interactúa con el servidor de aprovisionamiento. La consola de operador y administrador proporciona una representación gráfica de los activos, incluye asistentes para simplificar la configuración y otras características tales como la presentación de informes y el seguimiento de estado de las tareas que no están disponibles desde la interfaz de línea de comandos(IBM 2011).
- **Librería de Administración de Servicios Integrados de IBM:** es una librería compartida de automatización de procesos. Se trata de un amplio catálogo en línea, que contiene más de quinientos negocios de *IBM Tivoli* y extensiones de productos asociados, incluyendo paquetes de automatización, adaptadores de integración, los agentes y la documentación(IBM 2011).
- **Directorio de Usuarios:** *Tivoli Provisioning Manager* se integra con varios servidores de directorio, lo que le permite administrar sus cuentas de usuarios y la autenticación de los mismos con un servidor de directorio de su elección(IBM 2011).



1.3.2 Sistema de Administración de Identidades Oracle11g.

El Sistema de Administración de Identidades de Oracle se basa en una arquitectura de varios niveles. A continuación se analizan cada uno de estos niveles en detalle y se puede apreciar el diagrama en el **Anexo II**.

Presentación (*Presentation Tier*): la capa de presentación representa la interfaz de usuario de la aplicación. Está implementada siguiendo el patrón MVC y tiene dos clientes de interfaz de usuario, la consola de administración y la consola de diseño(ORACLE 2010).

Servicios de Negocio (*Business Services Tier*): se implementa como una aplicación EJB, e incluye servicios y funciones como son el servicio de gestión de usuarios, los servicios de gestión de políticas y los servicios de conciliación y aprovisionamiento(ORACLE 2010).

Datos (*Data Tier*): consiste en el repositorio que gestiona y almacena datos y metadatos, compatible con la base de datos relacional y LDAP(ORACLE 2010).

Intermedia (*Middleware Tier*): capa intermedia que contiene componentes como la suite de SOA, *Scheduler* y *BIP Reports*(ORACLE 2010).

Una arquitectura de integración escalable y flexible es esencial para el éxito del despliegue de soluciones de aprovisionamiento. El Sistema de Administración de identidades de Oracle ofrece una arquitectura de integración por medio del *framework* de conectores, que proporciona una rápida integración con los sistemas, también separa el código del conector a través de librerías de integración específicas. Esta separación trae consigo que la ampliación, mantenimiento y actualización de los conectores sea un proceso manejable y sencillo. Los cambios de versión del sistema de destino, simplemente requieren del reemplazo del código del conector sin necesidad de realizar alguna modificación en el diseño en general(ORACLE 2010).

1.3.3 Sistema de Aprovisionamiento de Microsoft.

El Sistema de Aprovisionamiento (*MPS por sus siglas en inglés*) contiene varios componentes que se pueden instalar utilizando la herramienta *MPS Deployment Tool*. Seguidamente se describen las funciones de cada uno de estos componentes y además la arquitectura que ellos conforman(MICROSOFT 2011h).

MPS cuenta con una variedad de componentes que soportan el aprovisionamiento de usuarios, entre los cuales se encuentran:

- Espacios de nombres (*Namespaces*).



“Subsistema de Aprovisionamiento de Usuarios para el Sistema de Administración de Identidades”

- Procedimientos (*Procedures*).
- Motor de Aprovisionamiento (*Provisioning Engine*).
- Proveedores (*Providers*).
- Bases de Datos (*Databases*).

La interacción entre estos componentes se puede apreciar en el diagrama perteneciente al **Anexo III**. De manera general en este sistema para iniciar los servicios de aprovisionamiento, se provee una petición XML como entrada al MPS. Luego el sistema convierte el XML especificado a llamadas API hacia aplicaciones externas como son *Microsoft Active Directory*, DNS, *Internet Information Services (IIS)* y *Microsoft SQL Server*, donde las tareas apropiadas de aprovisionamiento son ejecutadas. MPS cuenta con un motor de aprovisionamiento transaccional que permite en caso de que falle alguna transacción, virar atrás la misma. Dicho sistema también posee un historial de transacciones de aprovisionamiento. Además, MPS brinda facilidades de colas de espera para aceptar peticiones que pueden ser ejecutadas más adelante, es decir, en caso de no poder atenderlas en el momento en que se producen, las mantiene en espera hasta que puedan ser atendidas(MICROSOFT 2011h).

Los componentes de MPS proporcionan un entorno robusto de aprovisionamiento con una arquitectura extensible que a su vez permite el empleo de una gran variedad de aplicaciones y servicios.

- **Espacios de Nombres y Procedimientos:** los espacios de nombres son definiciones XML que contienen colecciones de uno o más procedimientos y son almacenadas en la base de datos de configuración MPS (*MPS Configuration Database*). Los procedimientos también son definiciones XML y son usados para invocar la funcionalidad de cada uno de los proveedores u otros espacios de nombres(MICROSOFT 2011h).
- **Motor de Aprovisionamiento:** servicio que recibe peticiones XML y realiza una serie de acciones de aprovisionamiento predefinidas o creadas en el contexto de una identidad segura específica(MICROSOFT 2011h)..
- **Proveedores:** son bibliotecas de vínculos dinámicos (DLL) que contienen los procedimientos de los componentes que hacen llamadas API a aplicaciones externas y ejecutan las tareas de bajo nivel iniciadas por las solicitudes de aprovisionamiento. MPS viene con un conjunto básico de proveedores que permiten comenzar rápidamente utilizando las capacidades y beneficios de este tipo de sistema(MICROSOFT 2011h).



*“Subsistema de Aprovisionamiento de Usuarios para el Sistema de
Administración de Identidades”*

- **Bases de Datos:** la base de datos MPS incluye la base de datos de configuración, registro de transacciones, registro de auditoría y servicio de auditoría y recuperación. Además, proporciona una base de datos que permite la prestación de planes de servicio predefinidos al cliente y el seguimiento de eventos de aprovisionamiento para facilitar las funciones de facturación(MICROSOFT 2011h)..

1.3.4 Sistema de Administración de Identidades de Novell Edición Estándar 4.

Algunas de las características del Sistema de Administración de Identidades de Novell son las siguientes:

- **Provisión y Desaprovisionamiento Automatizados:** se puede automatizar completamente la provisión de usuarios. La creación de un registro en un sistema de recursos humanos puede automatizarse para la provisión de acceso al edificio, la creación de una cuenta de correo electrónico, el acceso al sistema telefónico y otras muchas posibilidades.
- **Provisión Basada en Reglas:** se realiza la provisión de acuerdo con reglas configuradas y controladas por el usuario, lo que le aporta la flexibilidad que necesita para satisfacer necesidades empresariales.
- **Arquitectura en Tiempo Real Basada en Eventos:** realiza el aprovisionamiento y desaprovisionamiento de identidades de manera instantánea y automática, en respuesta a cualquier evento en el ciclo de vida de la cuenta usuario. *Novell Identity Manager 4* está basado en eventos y está construido sobre un motor síncrono de provisión automatizada.
- **Información de Estado Actual:** permite obtener informes de todas las identidades, los privilegios de accesos a recursos y las actividades relacionadas con identidad que tienen lugar en su repositorio seguro de identidades.
- **Compatibilidad con Autenticación Única:** la interfaz de usuario se integra con diversas soluciones de autenticación única, de modo que los usuarios solamente necesitan recordar una contraseña, lo que incrementa la seguridad.
- **Gestión de Contraseñas y Autoservicio del Usuario:** permite a los usuarios gestionar sus datos de identidad y contraseñas. Los usuarios pueden modificar su información en las páginas de su empresa u obtener ayuda para recordar o restablecer contraseñas sin que tenga que intervenir el personal de informática.



“Subsistema de Aprovisionamiento de Usuarios para el Sistema de Administración de Identidades”

- **Limpieza y Normalización de Datos:** la herramienta limpia los datos de aquellas aplicaciones que añade al sistema de provisión de identidades (NOVELL 2012).

1.3.5 Servidor de Identidades WSO2.

Al igual que la utilización de servicios crece en las empresas, la gestión de las identidades de los empleados, proveedores, socios y clientes se ha convertido en un desafío cada vez mayor. WSO2 proporciona la gestión sofisticada de identidades y la seguridad de aplicaciones web y servicios empresariales (WSO2 2012).

Brinda la posibilidad de que los arquitectos y desarrolladores de la empresa mejoren la experiencia del cliente mediante la garantía de las interacciones en líneas seguras dentro y fuera de una arquitectura SOA.

Sistemas y Gestión de Identidad del usuario.

- Implementa un almacén flexible de usuarios a través de LDAP interno, LDAP externo, Directorio Activo de *Microsoft* o cualquier base de datos JDBC.
- API para la integración de la gestión de identidad a cualquier aplicación.
- Múltiples factores de autenticación a través de XMPP para OpenID.
- Single Sign-On (SSO) a través de OpenID, SAML2 y el KDC de Kerberos.
- Aprovisionamiento a través de SCIM en lugar de SPML legado.
- Implementa la seguridad REST con OAuth 2.0 y XACML.
- Gestión flexible de perfiles para los usuarios que soportan múltiples perfiles.
- XKMS para el almacenamiento y distribución de claves.

Gestión de derechos.

- Función de control de acceso basado en roles (RBAC)
- Gestión de reclamaciones del control de acceso a través de XACML y OpenID.
- Gestión de derechos de auditoría avanzada.
- Gestión de derechos para cualquier llamada REST o SOAP.

Administrar y Controlar.

- Consola web de gestión integral y monitoreo de la seguridad a nivel empresarial y SSO.
- Soporte flexible de registro con la integración de sistemas empresariales.



“Subsistema de Aprovisionamiento de Usuarios para el Sistema de Administración de Identidades”

- Gestión de la configuración centralizada a través de diferentes entornos de implementación, con ciclos de vida y control de versiones con la integración a *WSO2 Governance Registry*.

1.3.6 Módulo de Administración para la aplicación de aprovisionamiento de usuarios del Sistema de Administración de Identidades.

El módulo desarrollado posibilita gestionar de manera centralizada todo el ciclo de vida de las cuentas de usuarios de la aplicación de aprovisionamiento del sistema de administración de identidades. Las acciones a realizar sobre las cuentas de usuarios, como crear, habilitar, suspender, eliminar y bloquear, se rigen por un flujo de aprobación de solicitudes definido en el módulo motor de tareas, que cuenta con cinco estados: creada, revisada, aprobada, confirmada, revocada y completada(VIVIANA and ARMANDO 2010).

El sistema informático fue desarrollado utilizando C# como lenguaje de programación sobre la plataforma .NET propiciando seguridad, además de la utilización de *Bison framework* para la gestión de las instancias de flujos de trabajo y Oracle como sistema gestor de bases de datos por la necesidad de un continuo crecimiento del trabajo y un rápido acceso a los registros almacenados. También, el sistema desarrollado permite gestionar toda la información referente a las cuentas de usuarios en el sistema de administración de identidades y cuenta con un diseño orientado a objetos. El *software* cuenta con una arquitectura basada en un modelo de capas, utiliza el estilo arquitectónico Modelo Vista Controlador e implementa el patrón Fachada. Este tipo de arquitectura soporta el desarrollo incremental de sistemas y evita que los cambios en una de las capas afecten directamente al resto. La arquitectura se encuentra representada por cinco capas lógicas que dan un alto nivel de encapsulamiento de las responsabilidades, permitiendo reducir al máximo el acoplamiento y aumentar la reutilización entre las mismas(VIVIANA and ARMANDO 2010). El diagrama donde se muestra la vista lógica de la arquitectura se puede apreciar en el **Anexo IV**.

Esta solución es una buena herramienta para el aprovisionamiento de usuarios y se basa en las necesidades del proyecto en ese momento, pero tiene una arquitectura rígida que brinda poca flexibilidad y adaptabilidad a cambios lo que provocó que a partir de las nuevas necesidades del proyecto de hacer más flexible el aprovisionamiento y cambiar el modelo de datos, dicha solución quedara sin concordancia con el proyecto, además no utiliza ningún componente para conectarse a los recursos, lo que le aporta más rigidez al sistema y hace difícil su uso. También su funcionamiento está basado en el módulo Gestor de Tareas el cual solo aprovisiona en un solo recurso, lo que implica una desventaja adicional.



1.3.7 Gestor de Tareas para el aprovisionamiento de usuarios.

El módulo Motor de Tareas se encarga de procesar de forma rápida y segura las solicitudes de usuarios disminuyendo el tiempo de espera de estos últimos. Debido al flujo de salida de los trabajadores de una organización, ya sea de forma temporal o de forma permanente, se almacenan en los repositorios cuentas inactivas por desuso, estas cuentas se convierten en una potencial amenaza para la seguridad del sistema. El módulo Motor de Tareas se encarga de revisar temporalmente estos repositorios en busca de cuentas en desuso para su suspensión, y en caso de que lleven suspendidas un tiempo determinado se procederá a su eliminación(MARIO, MEZQUINA AZAHARES and CARLOS 2010). La vista lógica de este módulo se encuentra disponible en el **Anexo V**.

El sistema informático al igual que el módulo de administración fue desarrollado utilizando C# como lenguaje de programación sobre la plataforma .NET y Oracle como sistema gestor de bases de datos. Permite gestionar las cuentas de usuarios aprovisionadas en repositorios suscritos al sistema de administración de identidades. Este módulo tiene una arquitectura dividida en capas. Con esta arquitectura las capas inferiores proveen servicios a las capas superiores y las capas superiores delegan responsabilidades a las capas inferiores(MARIO, MEZQUINA AZAHARES and CARLOS 2010).

El módulo está desarrollado con tecnología *RabbitMQ*, un gestor de colas de mensajes que sirve como interfaz de comunicación con el motor y las aplicaciones que se sirven de él. Por otra parte el motor aprovisiona usuarios solamente en *OpenLdap*. Estas características hacen de este componente un *software* que no cumple con las necesidades actuales del proyecto, sobre todo el manejo de los recursos debido a que se necesita aprovisionar usuarios en varios de los mismos y brindar la posibilidad de que el usuario incorpore recursos nuevos a la aplicación. Esta nueva visión imposibilita la utilización de este módulo en el proyecto.

1.3.8 Librería de Conectores.

Los conectores de un Sistema de Administración de Identidades son los encargados de garantizar la interacción con los recursos que va a utilizar el sistema. Un conector está compuesto por un conjunto de clases, en las cuales quedan implementadas las funcionalidades para realizar las tareas de aprovisionamiento. Los conectores para poder realizar sus funciones tienen que conocer algunas credenciales del recurso al cual se van a conectar para de esta forma ejecutar sus tareas. Los parámetros de conexión varían en dependencia del recurso y una vez introducidos por el administrador del sistema



“Subsistema de Aprovechamiento de Usuarios para el Sistema de Administración de Identidades”

son guardados cifrados en una base de datos encargada de contener toda esta información. La Librería de Conectores está compuesta por librerías de clases que estarán integradas al servicio de aprovisionamiento, este servicio es quien a través de las librerías realizará las funciones de provisión(WILHEM and ERNESTO 2011).

1.4 Investigación sobre las principales herramientas, tecnologías y lenguajes a utilizar.

Luego de analizar la panorámica referente al estado del arte de las soluciones de gestión de identidades existentes, también es importante abordar las principales herramientas, tecnologías y lenguajes que serán utilizados para el desarrollo del Subsistema.

1.4.1 .Net Framework 4.

.NET Framework es el modelo de programación completo y coherente de Microsoft para compilar aplicaciones que ofrezcan una gran experiencia visual al usuario, comunicación perfecta y segura, además de la capacidad de modelar una amplia gama de procesos empresariales. El mismo funciona en paralelo con versiones anteriores de .NET Framework(MICROSOFT 2011f).

Microsoft .NET Framework 4 proporciona las siguientes mejoras y características nuevas:

- **Mejoras en el Lenguaje Común en Tiempo de Ejecución (CLR) y la Biblioteca de Clases Base (BCL):** mejora en el rendimiento, incluida una mayor compatibilidad con equipos multinúcleo, recolección de elementos no utilizados en segundo plano y asociación del generador de perfiles en el servidor. También tiene una depuración más fácil(MICROSOFT 2011f).
- **Mejoras en el Acceso a Datos y El Modelado:** el *Entity Framework* permite a los desarrolladores programar con bases de datos relacionales usando objetos .NET y el Lenguaje de Consultas Integrado (LINQ)(MICROSOFT 2011f).
- **Mejoras en ASP.NET:** más control sobre HTML, identificadores de elementos y hojas CSS personalizadas. Trae nuevas características de datos dinámicos, incluidos nuevos filtros de consulta, plantillas de entidad, mayor compatibilidad con *Entity Framework 4*. Además tiene incluida la compatibilidad con formularios web para nuevas mejoras de librerías de AJAX(MICROSOFT 2011f).
- **Mejoras en Windows Workflow Foundation(WWF):** estas mejoras incluyen un modelo de programación de actividades mejorado, un mejor funcionamiento del diseñador, un nuevo estilo de



modelado de diagramas de flujo, una paleta de actividades expandida e integración con reglas de flujos de trabajo(MICROSOFT 2011f).

- **Mejoras en *Windows Communication Foundation (WCF)*:** la compatibilidad con servicios de flujos de trabajo de WCF, que permiten programas con actividades de mensajería y correlación. Además, *.NET Framework 4* proporciona nuevas características de WCF como la detección de servicios, servicio de enrutamiento, diagnósticos y rendimiento(MICROSOFT 2011f).

1.4.2 *ASP.net MVC 3*.

El Modelo – Vista – Controlador (MVC) separa la arquitectura de la aplicación en tres componentes: el modelo, la vista, y el controlador. *ASP.NET MVC* proporciona una alternativa a los formularios web de *ASP.NET* para la creación de aplicaciones web(MICROSOFT 2011g).

El *framework* MVC incluye los siguientes componentes:

- **Modelo:** los objetos del modelo son las partes de la aplicación que implementan la lógica de dominio de datos de las aplicaciones(MICROSOFT 2011g).
- **Vista:** las vistas son las que muestran la interfaz de usuario de la aplicación(MICROSOFT 2011g).
- **Controlador:** los controladores son los que se encargan de la interacción del usuario, trabajar con el modelo y además seleccionar una vista para hacer que se muestre en la interfaz de usuario. En una aplicación MVC, la vista sólo muestra la información, el controlador es el encargado de responder a la entrada del usuario y la interacción. Por ejemplo, el controlador se encarga de la cadena de consulta de valores, y pasa estos valores para el modelo, que a su vez podrían ser usados para consultar a la base de datos(MICROSOFT 2011g).

Las ventajas de una aplicación web basada en MVC:

- Es más fácil dividir una aplicación en el modelo, la vista y el controlador para manejar la complejidad de las aplicaciones(MICROSOFT 2011g).
- Se procesan las solicitudes de aplicaciones web a través de un único controlador. Esto permite diseñar una aplicación que sea compatible con una amplia infraestructura de enrutamiento(MICROSOFT 2011g).
- Ofrece un mejor soporte para desarrollo basado en pruebas (TDD)(MICROSOFT 2011g).



- Beneficia las aplicaciones web que son apoyadas por grandes equipos de desarrolladores y de diseñadores web que necesitan un alto grado de control sobre el comportamiento de la aplicación(MICROSOFT 2011g).

1.4.3 *Entity Framework 4.1.*

Las aplicaciones del *Entity Framework* requieren de la creación de un modelo conceptual que defina las entidades y las relaciones, un modelo lógico que represente el modelo relacional subyacente y las asignaciones entre los dos. Las características y componentes del *Entity Framework* que a continuación se mencionan trabajan en conjunto para brindar un entorno de programación de un extremo a otro(MICROSOFT 2011a).

- **El *Entity Data Model (EDM)*:** especifica el esquema de diseño que se utiliza para generar las clases programables que usa el código de la aplicación(MICROSOFT 2011a).
- **El Componente *Object Services*:** permite a los programadores trabajar con las clases del CLR generadas a partir del modelo conceptual. También proporciona compatibilidad de infraestructura con el *Entity Framework*, a partir de servicios como administración de estados, seguimiento de cambios, resolución de identidad, relaciones de carga y navegación(MICROSOFT 2011a).
- ***LINQ para Entidades*:** proporciona compatibilidad con *LINQ* para consultar las entidades(MICROSOFT 2011a).
- ***Entity SQL*:** es un lenguaje independiente del almacenamiento que es similar a SQL y que se utiliza para la consulta y manipulación de gráficos(MICROSOFT 2011a).
- **El proveedor *EntityClient*:** extiende el modelo de proveedor de *ADO.NET* teniendo acceso a los datos en lo que respecta a las entidades conceptuales y relaciones(MICROSOFT 2011a).

ADO.NET Entity Framework 4.1 trae consigo dos características nuevas, el API de *DbContext* y *Code First*. El API de *DbContext* es una abstracción simplificada del tipo *ObjectContext*, está optimizada para las tareas comunes y los patrones de código. *Code First* es un nuevo patrón de desarrollo para el *Entity Framework* que proporciona una alternativa a los patrones *Database First* y *Model First* existentes. Además permite definir su modelo mediante clases CLR(MICROSOFT 2011c).



1.4.4 *Windows Workflow Foundation.*

Un *workflow* o flujo de trabajo es un conjunto de unidades elementales llamadas actividades que están almacenadas como un modelo que describe un proceso real. Los flujos de trabajo proporcionan una manera de describir el orden de ejecución y las relaciones de dependencia entre las partes de trabajo de ejecución corta o prolongada. Este trabajo pasa a través del modelo desde el principio hasta el final y las actividades pueden ser ejecutadas por personas o por funciones del sistema.(MICROSOFT 2007b)

El *framework .NET 4* agrega una serie de características a *Windows Workflow Foundation*. Seguidamente se describen algunas de ellas(MICROSOFT 2011e).

- **Actividades de Mensajería:** las actividades de mensajería *receive*, *sendreply*, *send* y *receiverreply* se utilizan para enviar y recibir mensajes de *Windows Communication Foundation (WCF)* desde el flujo de trabajo.
- **WorkflowServiceHost:** es el host de flujo de trabajo para uso inmediato que admite varias instancias, configuración y mensajería de WCF. También se integra con persistencia, seguimiento y control de instancias a través de un conjunto de comportamientos de servicio.
- **Diagrama de Flujo:** es un paradigma conocido para representar visualmente los problemas de dominio. Una característica básica del diagrama de flujo es que solo se ejecuta una actividad en un momento dado.
- **InvokeMethod:** la actividad *InvokeMethod* permite invocar métodos públicos en objetos o tipos dentro del ámbito. Admite la invocación de métodos estáticos y de instancia con o sin parámetros y de métodos genéricos. También permite la ejecución de métodos sincrónica y asincrónicamente.
- **Servicio de Enrutamiento de WCF:** el servicio de enrutamiento está diseñado para ser un enrutador por *software* genérico que permite controlar cómo los mensajes de WCF circulan entre los clientes y los servicios. El servicio de enrutamiento permite desacoplar los clientes de los servicios.
- **Seguimiento:** el seguimiento de flujo de trabajo proporciona una visión sobre la ejecución de una instancia de flujo de trabajo.
- **Almacén de Instancias de Flujo de Trabajo de Oracle:** *OracleWorkflowInstanceStore* es una implementación basada en Oracle 11g de un almacén de instancias que almacena el estado de una instancia en ejecución junto con todos los datos necesarios para cargar y reanudar esa



instancia. Esta característica viene incluida por defecto para SQL pero permite el trabajo con otros gestores como es el caso de Oracle 11g.

1.4.5 Windows Communication Foundation.

Windows Communication Foundation (WCF) permite un amplio control sobre las funciones de mensajería de una aplicación. Esta tecnología tiene determinadas características como son los servicios de flujo de trabajo donde se describen cómo crearlos y configurarlos, también están las direcciones, enlaces y contratos donde se describe cómo controlar varios aspectos del servicio. También se pueden encontrar las transacciones que son las operaciones de transacción que se puedan revertir de ser necesario. WCF ayuda a crear aplicaciones que posean gran confidencialidad e integridad. Existen otras características como son las conexiones de redes punto a punto, los metadatos, el hospedaje, la integración e interoperabilidad donde se utiliza WCF para extender su lógica existente. Otro aspecto es el modelo de programación web HTTP de WCF que permite a los programadores exponer las operaciones del servicio WCF. También una característica importante es la integración de AJAX y la compatibilidad con JSON(MICROSOFT 2011d).

1.4.6 Lenguaje de programación C Sharp.

C#, leído en inglés “C Sharp” es un lenguaje de propósito general diseñado por *Microsoft* para su plataforma *.NET*. Programar en dicha plataforma usando dicho lenguaje es sencillo e intuitivo. Además es un lenguaje de programación que toma las mejores características de lenguajes preexistentes como Visual Basic, Java o C++ y las combina en uno solo y el hecho de ser relativamente reciente no implica que sea inmaduro(JOSÉ 2008).

A continuación se recogen de manera resumida las principales características de C#. Alguna de las cuales no son exactamente propias del lenguaje, sino de la plataforma *.NET* en general.

- **Orientación a Objetos:** C# es un lenguaje orientado a objetos. Es más puro en tanto que no admite ni funciones ni variables globales sino que todo el código y datos han de definirse dentro de definiciones de tipos de datos, lo que reduce problemas por conflictos de nombres y facilita la legibilidad del código(JOSÉ 2008).



- **Orientación a Componentes:** la propia sintaxis de C# incluye elementos propios del diseño de componentes. Es decir, la sintaxis de C# permite definir cómodamente propiedades, eventos o atributos(JOSÉ 2008).
- **Gestión Automática de Memoria:** permite que en el lenguaje no sea necesario incluir instrucciones de destrucción de objetos. Sin embargo, C# también proporciona un mecanismo de liberación de recursos determinista a través de la instrucción *using*(JOSÉ 2008).
- **Seguridad de Tipos:** C# incluye mecanismos que permiten asegurar que los accesos a tipos de datos siempre se realicen correctamente, lo que permite evitar que se produzcan errores difíciles de detectar por acceso a memoria no perteneciente a ningún objeto y es especialmente necesario en un entorno gestionado por un recolector de basura(JOSÉ 2008).

1.4.7 IDE de desarrollo *Visual Studio 2010*.

Microsoft Visual Studio no es más que un entorno de desarrollo integrado (IDE) para *Windows*. Este entorno de desarrollo soporta diferentes lenguajes de programación entre los cuales se pueden mencionar *Visual C++*, *Visual C#*, *Visual J#*, y *Visual Basic*. *Microsoft Visual Studio 2010 Ultimate* es un potente paquete de herramientas de administración del ciclo de vida de las aplicaciones. Con este paquete se garantiza la calidad de los resultados, desde el diseño hasta la implementación(MICROSOFT 2011b).

Esta suite posee disímiles características entre las principales se encuentran las siguientes:

- **Depuración y Diagnóstico:** la suite presenta *IntelliTrace*, que no es más que una valiosa característica de depuración. Los evaluadores pueden archivar errores enriquecidos y modificables para que los desarrolladores puedan reproducir siempre el error del que se informe y el estado en el que se encontró(MICROSOFT 2011b).
- **Herramientas de Prueba:** *Visual Studio 2010 Ultimate* incorpora todas las herramientas avanzadas de *Microsoft* para pruebas(MICROSOFT 2011b).
- **Arquitectura y Modelado:** los diagramas por capas ayudan a garantizar el cumplimiento de la arquitectura y permiten validar artefactos de código con respecto a los diagramas. Además, dicha suite admite los cinco diagramas de UML más comunes que conviven junto con su código(MICROSOFT 2011b).



1.4.8 Sistema Gestor de Base de Datos **Oracle 11g**.

Oracle es un sistema gestor de bases de datos relacional (RDBMS *Relational Data Base Management System*), que permite realizar las siguientes operaciones: introducir, almacenar, recuperar y manejar datos de manera sencilla(STOCKERS 2012).

Las características más destacadas de este RDBMS son las siguientes:

- **Soporte de Transacciones:** capaz de ejecutar órdenes manteniendo la integridad de los datos, haciendo que las transacciones no puedan finalizar en un estado intermedio(STOCKERS 2012).
- **Estabilidad:** con una tasa de errores muy baja(STOCKERS 2012).
- **Escalabilidad:** capaz de adaptarse o de cambiar según las necesidades(STOCKERS 2012).
- **Soporte Multiplataforma:** compatible con diversos sistemas operativos como *Linux, Unix, Windows, Solaris, AIX* o *Mac OS X Server*(STOCKERS 2012).

Oracle 11g es la nueva suite de productos *software* de la compañía Oracle, basados en la tecnología *Grid Computing* (o computación Grid). El Grid es una arquitectura que agrupa varios servidores y recursos de almacenamiento y procesamiento en una estructura económica y flexible que atiende a todas las necesidades de la organización(STOCKERS 2012).

Los elementos sobre los que se basa el *Oracle Grid Computing* son:

- **Oracle Database 11g:** es el motor de la base de datos. Dispone de las herramientas capaces de gestionar el almacenamiento de la información, optimizar los recursos, ofrecer un nivel máximo de atención en los servicios, etc. Gestiona de forma eficaz la información relacional, correos electrónicos, multimedia y XML(STOCKERS 2012).
- **Oracle Application Server 11g:** es el servidor de aplicaciones Oracle. Nivel intermedio de la arquitectura de tres niveles que consiste en un conjunto de servicios que permite a las aplicaciones basadas en exploradores web interactuar con las bases de datos de Oracle(STOCKERS 2012).
- **Oracle Enterprise Manager 11g:** proporciona el marco adecuado para llevar a cabo la administración de la base de datos(STOCKERS 2012).

1.4.9 Lenguaje Unificado de Modelado (UML).

Lenguaje unificado para la construcción de modelos se define como un lenguaje que permite especificar, visualizar y construir los artefactos de los sistemas informáticos. Es un sistema notacional destinado a los sistemas de modelado que utilizan conceptos orientados a objetos(CRAIG 2004).



1.4.10 *Altova MissionKit 2011.*

Altova MissionKit 2011 es una suite integrada de herramientas para XML, bases de datos y UML diseñada para cubrir las diversas necesidades durante el desarrollo y gestión de datos de los arquitectos de *software* y desarrolladores XML actuales. Esta suite cuenta con herramientas para el desarrollo XML, desarrollo web y de servicios web, también posee herramientas de integración y mapeado de datos, generación y publicación de datos XML, así como de modelado UML, comparación, consulta y diseño de bases de datos (ALTOVA 2011a).

Incluye ocho herramientas las cuales se especifican a continuación:

- ***Altova StyleVision:*** se utiliza para generar informes y formularios electrónicos a partir de una gran variedad de formatos fuente. Ofrece avanzadas funciones que permiten transformar contenido XML y de bases de datos en HTML, RTF, PDF, Word 2007 o superior y formularios. Permite diseñar recursos profesionales con potentes características de visualización, como contenido dinámico, plantillas modulares y gráficos (ALTOVA 2011g).
- ***Altova DatabaseSpy:*** herramienta única de consulta, diseño y comparación para múltiples bases de datos. Se conecta a las principales bases de datos y permite editar SQL, diseñar estructuras de bases de datos (BD), editar contenido de BD y comparar esquemas y contenido de BD (ALTOVA 2011b).
- ***Altova DiffDog:*** se usa para la comparación y combinación de archivos, directorios, esquemas y tablas de bases de datos. Las funciones de comparación y edición son compatibles con contenido XML (ALTOVA 2011c).
- ***Altova MapForce:*** permite asignar, convertir e integrar cualquier combinación de datos XML, bases de datos, archivos planos, Excel 2007 o superior y servicios web. Después transforma estos datos instantáneamente o genera código de integración de datos libre de derechos de autor para utilizarlo en conversiones permanentes (ALTOVA 2011d).
- ***Altova SchemaAgent:*** herramienta de análisis y administración de relaciones entre esquemas XML, instancias XML, archivos XSLT y archivos WSDL en múltiples proyectos, en una intranet o en toda una empresa. También permite visualizar archivos utilizados en proyectos de asignación de datos de *MapForce* (ALTOVA 2011e).
- ***Altova SemanticWorks:*** editor visual de RDF y OWL para la web semántica (ALTOVA 2011f).



*“Subsistema de Aprovisionamiento de Usuarios para el Sistema de
Administración de Identidades”*

- **Altova UModel:** es usado para el diseño de modelos de aplicaciones con UML de forma visual, genera código Java, C# o Visual Basic, así como documentación. Permite aplicar técnicas de ingeniería inversa a programas ya existentes para convertirlos en diagramas UML 2. *UModel* es la herramienta UML que permite un diseño de *software* visual y práctico en cualquier tipo de proyecto(ALTOVA 2011h).
- **Altova XMLSpy:** editor y entorno de desarrollo XML con avanzadas funciones de modelado, edición, transformación y depuración de tecnologías XML. Contiene un diseñador gráfico de esquemas, generador de código, conversores de archivos, generadores de perfiles, integración total de bases de datos, compatibilidad con XSLT, XPath, XQuery, WSDL, SOAP, XBRL y documentos *Office Open XML* (OOXML), además de complementos para *Visual Studio* y *Eclipse*(ALTOVA 2011i).

1.4.11 PLSQL Developer 7.1.4.1390.

PLSQL Developer es un entorno de desarrollo integrado que está específicamente dirigido a la creación de unidades de programas almacenados para bases de datos Oracle. Se centra en la sencillez de su uso, calidad del código y la productividad de las principales ventajas en el desarrollo de aplicaciones Oracle(AUTOMATIONS 2012).

Las siguientes son las principales características de *PLSQL Developer*:

- **Potente PLSQL Editor:** el editor brinda, descripciones de objetos, asistente de código, consejos del compilador, refactorización, jerarquía de código, plegado de código, exploración de hipervínculo y Biblioteca.
- **PLSQL Beautifier:** permite formatear su código SQL y PLSQL a través de un conjunto de normas definido por el usuario. Su código de forma automática puede ser enriquecido al compilar, guardar o abrir un archivo.
- **Diagramas:** la ventana del diagrama permite crear una representación gráfica de una selección de objetos.
- **Proyectos:** para organizar el trabajo se puede utilizar en *PLSQL Developer* concepto de proyectos, que le permite trabajar en el ámbito de un conjunto específico de objetos, en lugar de una completa base de datos o esquema.



- **Pruebas de regresión:** se pueden definir y ejecutar un conjunto de pruebas, y determinar rápidamente si todas las pruebas fueron ejecutadas correctamente.

1.4.12 Metodología de Desarrollo *MSF for Agile Software Development*.

Una metodología de desarrollo es un conjunto de procedimientos, técnicas, herramientas y un soporte documental que ayuda a los desarrolladores a realizar nuevas aplicaciones. No existe una regla para la utilización de una metodología u otra, la selección de esta depende del tipo de aplicación y el equipo que lo va a desarrollar. Las metodologías se pueden clasificar como ágiles y tradicionales. Las tradicionales son aquellas que se centran fundamentalmente en el control del proceso, además son las más efectivas para proyectos de gran tamaño. Entre ellas las más populares son *OPEN*, *METRICA 3* y *RUP*. Por su parte la ingeniería de *software* ágil combina una filosofía y un conjunto de directrices de desarrollo. La filosofía busca la satisfacción del cliente y la entrega temprana de *software* incremental, además de equipos de proyectos pequeños y con alta motivación. Las directrices de desarrollo resaltan la entrega sobre el análisis y diseño y la comunicación activa y continua entre los desarrolladores y los clientes. El ambiente moderno de negocios ocasiona que los sistemas basados en computadoras y los productos de *software* estén acelerados y en cambio continuos. La ingeniería de *software* ágil representa una opción razonable a la ingeniería convencional para ciertas clases de *software* y ciertos tipos de proyectos. Ha demostrado su utilidad al entregar sistemas exitosos con rapidez. Ejemplos de metodologías de este tipo se pueden encontrar *XP*, *SCRUMP*, *MSF* y *SXP*(ROGER 2007).

Este trabajo se enfocará principalmente en las metodologías ágiles, especialmente en la metodología *Microsoft Solution Framework for Agile Software Development (MSF)*.

Esta es una metodología flexible e interrelacionada con una serie de conceptos, modelos y prácticas de uso, que controlan la planificación, el desarrollo y la gestión de proyectos tecnológicos. *MSF* se centra en los modelos de proceso y de equipo dejando en un segundo plano las elecciones tecnológicas(CARLOS, OLIVARES ROJAS JUAN 2008). Cuenta con las siguientes características:

- **Adaptable:** usada en cualquier parte.
- **Escalable:** puede organizar equipos pequeños entre tres o cuatro personas, así como también, proyectos que requieren cincuenta personas o más.
- **Flexible:** es utilizada en el ambiente de desarrollo de cualquier cliente.



“Subsistema de Aprovechamiento de Usuarios para el Sistema de
Administración de Identidades”

- **Tecnología Agnóstica:** puede ser usada para desarrollar soluciones basadas sobre cualquier tecnología.
- **Modelos:** MSF se compone de varios modelos que se encargan de cada una de las fases del desarrollo de un proyecto de los cuales los más importantes se mencionan a continuación.
- **Modelo de Proceso:** diseñado para controlar, minimizando riesgos y aumentando la calidad. Proporciona pautas para el ciclo de vida del proyecto, describiendo fases y actividades.
- **Modelo de Gestión de Proyectos:** describe el rol de la gestión del proyecto, y como permitir mayor estabilidad, desde proyectos pequeños hasta proyectos largos y complejos.
- **Modelo de Gestión de la Preparación:** describe conocimientos, aptitudes y habilidades necesarias para planificar, desarrollar y gestionar soluciones exitosas.

El proceso de desarrollo en MSF consta de cinco fases, las cuales junto a los modelos mencionados anteriormente, complementan el ciclo de desarrollo del *software*:

- **Visión y Alcance:** en esta fase el equipo y el cliente definen los objetivos que se persiguen y hasta dónde se quiere llegar con el proyecto. La fase culmina cuando el Alcance y la Visión del proyecto sean aprobados.
- **Planificación:** es en esta fase en la que el equipo prepara las especificaciones funcionales, se realiza el proceso de diseño de la solución y se preparan los planes de trabajo, estimaciones de costo y cronogramas de los diferentes entregables del proyecto. Esta fase termina cuando esté aprobado el Plan de Proyecto.
- **Desarrollo:** durante esta fase el equipo realiza la mayor parte de la construcción de los componentes tanto documentación como código y se crea una solución de la arquitectura a establecer. Como resultado se obtiene una versión de la infraestructura del producto después de aplicarle revisiones al código que se va generando.
- **Estabilización:** en esta fase se llevan a cabo las pruebas sobre la solución, que enfatizan el uso y operación bajo condiciones reales. El equipo se enfoca en detectar los errores, solucionarlos y preparar la solución para el lanzamiento.
- **Implantación o Despliegue:** en esta fase el equipo implanta la tecnología y los componentes utilizados por la solución, estabiliza la implantación, apoya el funcionamiento y la transición del proyecto y obtiene la aprobación final del cliente. La fase termina con el hito Implantación completa.



*“Subsistema de Aprovisionamiento de Usuarios para el Sistema de
Administración de Identidades”*

1.5 Conclusiones.

A partir de conceptos y características fundamentales de los productos líderes en campo de la administración de identidades enfocándose en el proceso de aprovisionamiento de usuarios, así como herramientas y tecnologías existentes para su desarrollo se logró conocer el funcionamiento de este proceso de manera general. También se realizó el estudio de las investigaciones desarrolladas en el proyecto anteriormente en este ámbito, pero a pesar de ser buenas herramientas, se evidenció que no cumplen con las necesidades actuales del proyecto. Luego del estudio realizado, la metodología que guiará el proceso de desarrollo del Subsistema de Aprovisionamiento de Usuarios será MSF para el Desarrollo de *Software* Ágil. Para el modelado de clases y la base de datos, así como el trabajo con XML se hará uso de la herramienta *Altova MissionKit 2011*. El IDE de desarrollo que se utilizará para trabajar en la plataforma seleccionada *.NET* será *Visual Studio 2010*. Para lograr que toda la información persista se utilizará *Oracle Database 11g*. También se usarán tecnologías como son *Windows Communication Foundation* para la interacción con servicios web, así como *Windows Workflow Foundation* para los flujos de trabajo.



Capítulo 2 Visión y Planificación

2.1 Introducción.

De acuerdo a la metodología MSF para el Desarrollo de *Software* Ágil, guía para el proceso de desarrollo del presente sistema informático, se deben puntualizar un conjunto de conceptos y requerimientos utilizando un lenguaje entendible por todos los involucrados en el proceso en cuestión. Dicha metodología plantea que se debe adquirir una visión clara de lo que se desea desarrollar y además propone la realización de una planificación que indique al equipo de trabajo hacia la exitosa construcción del sistema, así como una descripción de los escenarios y requisitos del Subsistema. Además se presenta la arquitectura propuesta en conjunto con la descripción de sus aspectos más importantes. En esta sección se realizan las actividades presentes dentro de los flujos de trabajo que sugiere la metodología para iniciar el desarrollo de un proyecto(MICROSOFT 2007a).

2.2 Capturar visión del proyecto.

Iniciar un proyecto requiere el establecimiento claro de la visión del mismo. Capturar y comunicar esa visión central es el elemento más importante para mantener un proyecto enfocado. Esta visión puede cambiar como resultado del ambiente de negocios o del proyecto. Si el cambio se produce, es necesario volver a alinear el proyecto en relación a la nueva visión. Desde la visión se comienza a entender quiénes serán los usuarios. También se especifica si el proyecto es impulsado por fecha o por contenido. La visión y sus actividades relacionadas crean una base sólida sobre la cual puede ser construido un proyecto(MICROSOFT 2007a).

2.2.1 Descripción del Subsistema.

El Subsistema de Aprovisionamiento de Usuarios para el Sistema de Administración de Identidades permite gestionar el ciclo de vida de las cuentas de usuarios en los diferentes servidores a los cuales preste servicio, de manera que no se tengan que crear cuentas de forma independiente, sino que se solicite crear una y cuando se cree, se tendrá acceso a todos los recursos de una vez. Este Subsistema cuenta con varios módulos los cuales son: **Portal del Usuario**, **Portal de Administración** y el **Motor de Tareas**.



“Subsistema de Aprovisionamiento de Usuarios para el Sistema de Administración de Identidades”

Las actividades principales de las que se encargará el Subsistema son:

- Creación de cuentas de usuarios a través de un sistema de solicitudes.
- Asignación de tipos de acceso a las diferentes aplicaciones, sistemas y los tipos de autenticación que necesite.
- Gestión de los recursos asociados al Subsistema.
- Gestión de plantillas de cuentas de usuarios.
- Generación de reportes de las actividades realizadas en el Subsistema.

El módulo “Portal de Usuario” estará disponible para los usuarios estándares, revisores de solicitudes y aprobadores de solicitudes, así como contiene opciones básicas de la aplicación. El módulo “Portal de Administración” estará disponible sólo a los usuarios de tipo “Administrador”. En este módulo se administrarán las cuentas de usuarios creadas previamente a través de solicitudes, se crearán y administrarán los recursos que serán provisionados y se administrarán además, las preguntas de seguridad y las políticas de contraseñas de las cuentas de usuario. El módulo “Motor de Tareas” estará funcionando siempre y se encargará de chequear las restricciones de las cuentas y los recursos.

2.2.2 Definición de las Personas.

Persona	Acciones
Administrador del sistema	Es la persona que puede realizar las acciones de: <ul style="list-style-type: none">➤ Configuración del sistema.➤ Creación de plantillas.➤ Creación de preguntas de seguridad.➤ Creación de políticas de contraseña➤ Creación de usuarios.➤ Eliminación manual de usuarios.➤ Registro de nuevos conectores.➤ Registro de nuevos recursos.
Aprobador de solicitudes	Es la persona que puede aprobar o revocar solicitudes y listar solicitudes pendientes.
Revisor de solicitudes	Es la persona que puede revisar las solicitudes que se encuentren en estado de revisión de datos y listar solicitudes pendientes de revisión.
Usuario estándar	Es la persona que puede realizar las siguiente acciones: <ul style="list-style-type: none">➤ Cancelar la solicitud cuando aún no ha sido aprobado.➤ Cambiar datos personales.



“Subsistema de Aprovisionamiento de Usuarios para el Sistema de Administración de Identidades”

	<ul style="list-style-type: none">➤ Cambiar contraseña.➤ Recuperar contraseña.
--	---

Tabla 1: Descripción de las personas.

2.2.3 Propuesta de Solución.

La propuesta de solución que se plantea, es una aplicación web desarrollada sobre *ASP.net MVC 3* que cuenta con un Portal de Usuario, el cual contiene todas las funcionalidades a las que el usuario estándar tiene permisos. El mismo interactúa directamente con el Repositorio de Datos y el Motor de Tareas, este último está basado en servicios web y servicios del sistema operativo *windows* que brindan una serie de funcionalidades para el aprovisionamiento de usuarios en los diferentes recursos utilizando la Librería de Conectores. La aplicación cuenta además con un Portal de Administración que permite la configuración y gestión del sistema en general, este portal también utiliza el Repositorio de Datos y las facilidades del Motor de Tareas.

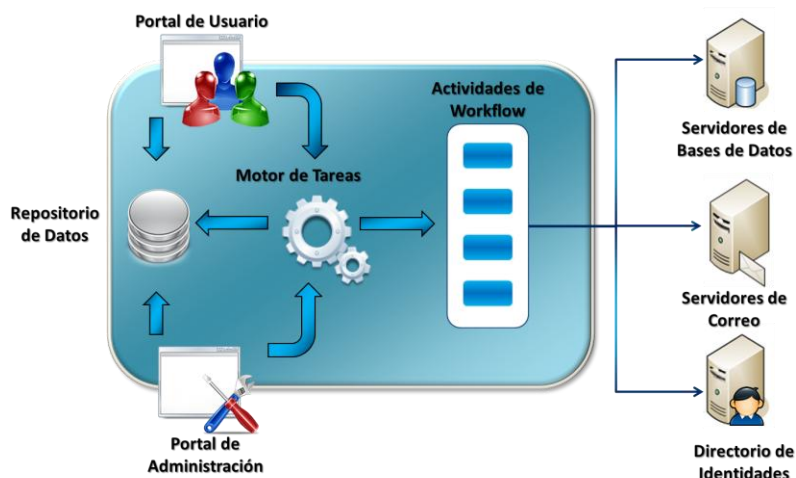


Figura 4: Propuesta de Solución.

A continuación se brinda una explicación a detalle de cada uno de los elementos mostrados en la figura, que componen la propuesta de solución:

- **Portal de Usuario:** en este portal se agrupan un conjunto de funcionalidades mediante el cual los usuarios estándares, revisores de solicitudes y aprobadores de solicitudes realizan acciones sobre el Subsistema, dentro de las que se encuentran: Cambiar Contraseña, Recuperar Contraseña y realizar la Solicitud de Creación de Cuenta de Usuario a partir de plantillas existentes.



“Subsistema de Aprovisionamiento de Usuarios para el Sistema de Administración de Identidades”

- **Repositorio de Datos:** es la base de datos donde se almacena la información de las cuentas de usuarios, los recursos asociados al Subsistema.
- **Motor de Tareas:** es el responsable de procesar una petición para convertirla en un procedimiento, el cual invoca una funcionalidad en un conector. Las entidades más importantes en el mismo son las peticiones, los procedimientos y los conectores. Las peticiones son el inicio o punto de partida de todos los servicios de aprovisionamiento, las cuelas son enviadas al motor de tareas mediante servicios web. Cada petición contiene la llamada a un procedimiento, el cual a su vez, debe invocar las funcionalidades correctas para satisfacer la petición. Los procedimientos son el núcleo de todas las transacciones de aprovisionamiento. Los conectores son la fuente de todas las funcionalidades requeridas para soportar todos los servicios de aprovisionamiento. El motor de tareas usa los conectores para ejecutar peticiones y además estos son objetos que brindan las interfaces funcionales para la comunicación con componentes externos. Otra de las características del motor de tareas es la capacidad de recuperarse ante fallos y para ello se utiliza el mecanismo de transacciones.

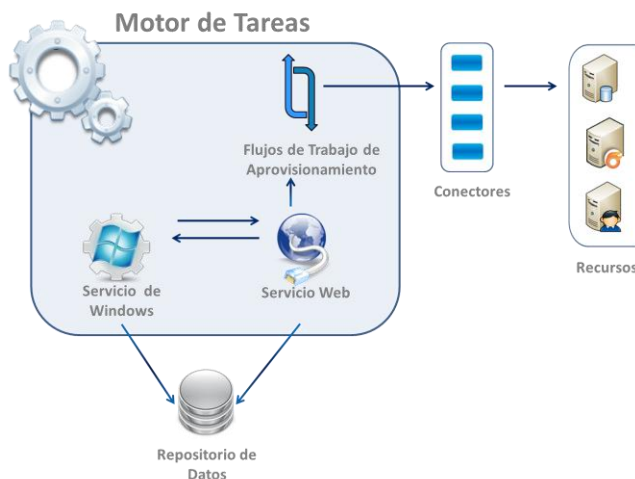


Figura 5: Estructura del Motor de Tareas.

- **Portal de Administración:** brinda las funcionalidades de administración del Subsistema. Permite la Gestión de Atributos de Identidades, lo que posibilita que el Subsistema sea flexible ante las necesidades de la organización, otra funcionalidad es la Gestión de Plantillas las cuales permiten la creación de tipos de usuarios agrupados por roles, recursos a los que tienen acceso y atributos de identidad, lo que facilita el proceso de creación de cuentas de usuario. Este portal también



“Subsistema de Aprovisionamiento de Usuarios para el Sistema de Administración de Identidades”

posibilita realizar el proceso de solicitud de creación de cuenta de usuario, donde se revisan los datos introducidos por el usuario y se aprueba o se revoca dicha solicitud. Esto se realiza a través de un flujo de trabajo que permite que en caso de existir errores en la revisión, regresar al paso anterior, ocurriendo de la misma manera hasta que la revisión sea satisfactoria. Además se puede llevar a cabo la administración de cuentas de usuarios donde se eliminan, muestran, modifican y se cambia el estado de dichas cuentas. Otra característica importante es la Gestión de Conectores, donde se cargan las librerías que permiten conectarse a los recursos donde el Subsistema realizará el aprovisionamiento. También se encuentra la Gestión de Recursos a partir de los conectores existentes, donde se hace posible habilitar o deshabilitar los recursos, así como definir horarios para aprovisionar en los mismos. Por otra parte se encuentra la Gestión de Preguntas de Seguridad, las cuales son mostradas para la recuperación de la contraseña, y la Gestión de Políticas de Contraseña para garantizar la seguridad de las mismas.

- **Actividades de *Workflow*:** son actividades que utilizan las bondades de los conectores para aprovisionar en los recursos asociados al Subsistema. Estas actividades están incluidas dentro de los flujos de trabajo de aprovisionamiento y forman parte del Motor de Tareas.

2.3 Planificación.

Luego de haber definido la Visión del Subsistema, se puede pasar a la fase de Planificación. En esta etapa del proyecto el equipo de trabajo analiza, identifica y además prioriza los requerimientos que describen la solución, conjuntamente con ello se generan algunos artefactos como son la lista de escenarios, así como la lista de requisitos de calidad de servicios, los cuales son utilizados para especificar los requisitos del *software* que sirven de guía para todo el proceso de desarrollo (MICROSOFT 2007a). Seguidamente se abordarán los artefactos generados en esta fase de acuerdo a la metodología MSF para el Desarrollo de *Software* Ágil.

2.3.1 Escenarios del Subsistema.

Los escenarios son utilizados de manera general para la definición de las prestaciones que brindará el *software* al usuario y representa una única vía de interacción entre el usuario y el sistema que se está desarrollando. Entre las actividades del ciclo de vida de un escenario se encuentra listar los mismos y para el desarrollo del Subsistema propuesto se identificaron los que se presentan a continuación:



Lista de Escenarios:

- Gestionar Conectores.
 - ✓ Crear conector.
 - ✓ Editar conector.
 - ✓ Eliminar conector.
 - ✓ Listar conectores.
- Gestionar Políticas de Contraseña.
 - ✓ Crear política de contraseña.
 - ✓ Editar política de contraseña.
 - ✓ Eliminar política de contraseña.
 - ✓ Listar políticas de contraseña.
- Gestionar Preguntas de Seguridad.
 - ✓ Crear pregunta de seguridad.
 - ✓ Editar pregunta de seguridad.
 - ✓ Eliminar pregunta de seguridad.
 - ✓ Listar preguntas de seguridad.
- Administrar Recursos.
 - ✓ Crear recurso.
 - ✓ Editar recurso.
 - ✓ Eliminar recurso.
 - ✓ Habilitar o deshabilitar recurso.
 - ✓ Listar recursos.
- Gestionar Plantillas.
 - ✓ Crear plantilla.
 - ✓ Editar plantilla.
 - ✓ Eliminar plantilla.
 - ✓ Listar plantillas.
- Gestionar Atributos de Identidad.
 - ✓ Crear atributo de identidad.
 - ✓ Editar atributo de identidad.



“Subsistema de Aprovisionamiento de Usuarios para el Sistema de Administración de Identidades”

- ✓ Eliminar atributo de identidad.
- ✓ Listar atributos de identidad.
- Administrar Solicitudes.
 - ✓ Aprobar solicitud.
 - ✓ Revocar solicitud.
 - ✓ Revisar solicitud.
 - ✓ Listar solicitudes.
- Administrar Cuentas de Usuario.
 - ✓ Cambiar de estado cuentas de usuario.
 - ✓ Eliminar cuenta de usuario.
 - ✓ Editar cuenta de usuario.
 - ✓ Listar cuentas de usuario.
- Configuraciones Generales.
 - ✓ Solicitar creación de cuenta de usuario.
 - ✓ Realizar seguimiento de solicitud de creación de cuenta.
 - ✓ Recuperar contraseña.
 - ✓ Cambiar contraseña.
 - ✓ Bloquear cuentas por desuso.
 - ✓ Eliminar cuentas bloqueadas.

2.3.2 Priorización de Escenarios.

Los escenarios se priorizan teniendo en cuenta la importancia que tienen cada uno de ellos para el desarrollo de la aplicación, en función de las dependencias existentes entre ellos. Este proceso de priorización de la lista de escenarios trae consigo identificar los escenarios más importantes en el momento de implementar el sistema, para que dicha implementación sea en las primeras iteraciones. Calificar al escenario con cinco puntos equivale a tener una prioridad “Alta”, con cuatro puntos “Media” y con tres puntos “Baja”.

No.	Escenarios	Prioridad
1	Gestionar conectores.	5
2	Gestionar políticas de contraseña.	5



*“Subsistema de Aprovisionamiento de Usuarios para el Sistema de
Administración de Identidades”*

3	Gestionar preguntas de seguridad.	5
4	Administrar recursos.	4
5	Gestionar plantillas.	4
6	Gestionar Atributos de Identidad.	4
7	Administrar solicitudes.	3
8	Administrar cuentas de usuario.	3
9	Configuraciones generales.	3

Tabla 2: Lista de grupos de escenarios y sus prioridades.

2.3.3 Plan de Iteraciones.

Una iteración es un conjunto de tareas programadas para ocurrir en un determinado período de tiempo. Determinar la longitud de una iteración incluye tener en cuenta factores claves incluyendo la fecha de entrega del proyecto, el tamaño de los escenarios y el tiempo de integración (MICROSOFT 2007a). MSF define las iteraciones como un período fijo de tiempo para programar tareas. Una iteración es generalmente un período de entre dos y seis semanas. Dichas iteraciones generalmente son numeradas consecutivamente y siguen una a otra de manera continua.

Dada la fecha de entrega orientada se ha tomado en consideración la realización de tres iteraciones, las cuales tomarían un tiempo de quince semanas aproximadamente para su total desarrollo.

- **Iteración #1:** Se implementarán los escenarios de mayor prioridad en el Subsistema, siendo necesario un total de seis semanas aproximadamente.
- **Iteración #2:** Se implementarán los escenarios de prioridad media en el Subsistema, siendo necesario un total de cinco semanas aproximadamente.
- **Iteración #3:** Se implementarán los escenarios de prioridad baja en el Subsistema, siendo necesario un total de cuatro semanas aproximadamente.

No.	Escenarios	Prioridad	Riesgo	Esfuerzo (días)	Iteración
1	Gestionar conectores.	5	Alto	9	1
2	Gestionar políticas de contraseña.	5	Medio	9	1
3	Gestionar preguntas de seguridad.	5	Medio	9	1
4	Administrar recursos.	4	Alto	10	2
5	Gestionar plantillas.	4	Alto	9	2
6	Gestionar Atributos de Identidad.	4	Medio	9	2



“Subsistema de Aprovisionamiento de Usuarios para el Sistema de Administración de Identidades”

7	Administrar solicitudes.	3	Alto	6	3
8	Administrar cuentas de usuario.	3	Alto	7	3
9	Configuraciones generales.	3	Bajo	9	3

Tabla 3: Planificación de los escenarios.

2.3.4 Cronometrar Escenarios.

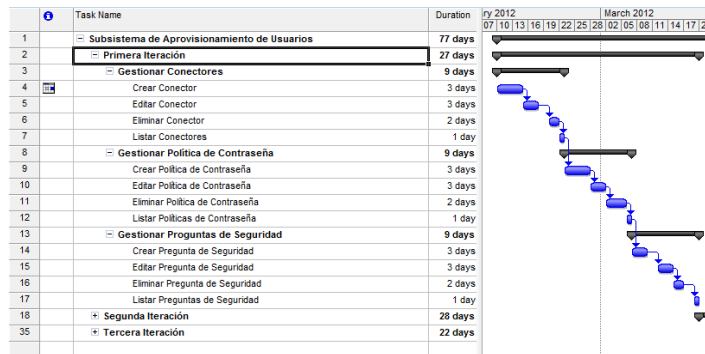


Figura 6: Primera Iteración.

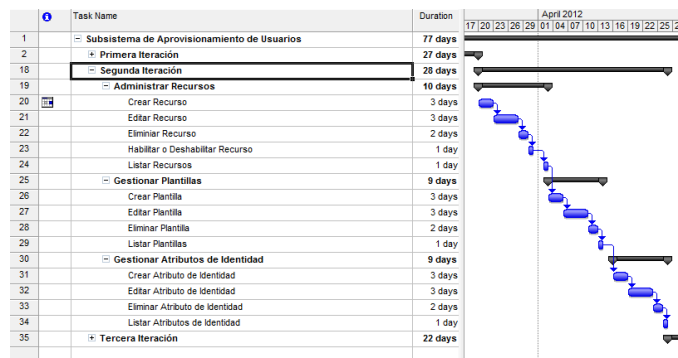


Figura 7: Segunda Iteración.

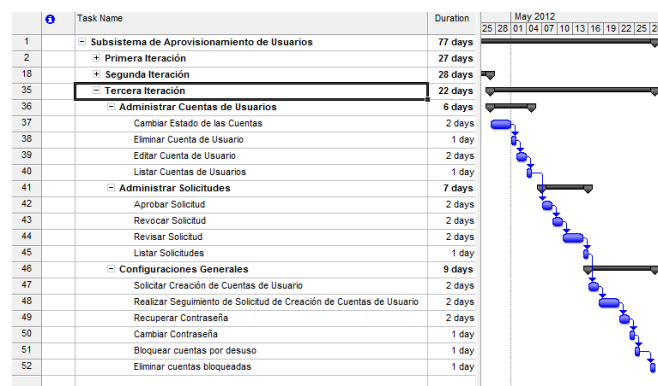


Figura 8: Tercera Iteración.

CAPÍTULO 2: VISIÓN Y PLANIFICACIÓN



“Subsistema de Aprovisionamiento de Usuarios para el Sistema de Administración de Identidades”

2.3.5 Requisitos de Calidad de Servicio.

Representan los requisitos no funcionales en las categorías de plataforma, rendimiento, seguridad y apariencia. A continuación se listan los requisitos de calidad de servicio que se identificaron para el desarrollo del Subsistema propuesto.

- Fiabilidad.
 - ✓ No se realizarán mantenimientos preventivos en horario laboral, deberán ejecutarse en un horario estipulado o los fines de semana, para no afectar la disponibilidad del Subsistema.
 - ✓ Se garantizará la consistencia de los datos, se realizarán comprobaciones y validaciones automáticas en todos los casos posibles.
- Restricciones de diseño.
 - ✓ El Subsistema debe implementarse usando el lenguaje *C Sharp*, sobre la plataforma *.NET*.
 - ✓ El sistema gestor de bases de datos, será *Oracle 11g*.
- Plataforma que soporta.
 - ✓ La aplicación debe estar disponible las 24 horas del día.
 - ✓ El Subsistema debe utilizar el *.NET framework* en su versión 4.0.
 - ✓ El servidor central debe tener sistema operativo *Windows Server 2003* o superior.
- Seguridad.
 - ✓ Solo se accederá a la BD desde la aplicación, nunca directamente desde el gestor de Base de Datos.
- Apariencia o interfaz externa.
 - ✓ Los campos obligatorios de cada ventana deben estar debidamente identificados y resaltados.
 - ✓ Permitir la paginación en los listados.
 - ✓ Contar con accesos directos a las funcionalidades más utilizadas.
 - ✓ Contar con notificaciones que informen al usuario.

2.3.6 Descripción de los Escenarios.

Una de las descripciones de escenarios más significativa que se puede apreciar dentro del Subsistema es la correspondiente a “Crear Conector”, la cual se muestra a continuación. Los demás escenarios están disponibles en el **Anexo VI**.



*“Subsistema de Aprovisionamiento de Usuarios para el Sistema de
Administración de Identidades”*

Nombre del Escenario: Crear Conector.		Identificador: Ap_Adm9.
Objetivo del Escenario: Permitir la creación de un conector en el sistema.		
Persona: Administrador.		
Iteración: 1ra.	Prioridad: 5.	Complejidad: Alta.
Descripción: El administrador selecciona la opción Gestionar Conectores. El sistema muestra en pantalla un listado con todos los conectores definidos. El administrador selecciona la opción Crear Conector y se muestra un formulario con los datos y el archivo requeridos: <ul style="list-style-type: none">➤ Nombre.➤ Descripción.➤ Archivo Zip. Luego de introducidos los datos y el archivo se selecciona la opción Crear y el sistema guarda el conector incluyendo las librerías que se encuentran dentro del archivo Zip.		
Validaciones: <ul style="list-style-type: none">➤ El nombre del conector debe coincidir con el nombre de la librería que se encuentra dentro del archivo Zip y que contiene las clases CreateResource y RelatedAttribute.		

Tabla 4: Descripción del Escenario Crear Conector.

2.4 Conclusiones.

En este capítulo se realizó la descripción del Subsistema en conjunto con la planificación del mismo a través de la realización de los elementos correspondientes a los conceptos fundamentales manejados por esta etapa dentro de la metodología utilizada, para obtener de esta forma una visión más específica sobre lo que se desea desarrollar. Además se expresó una propuesta de solución con el flujo de funcionamiento e interacción del Subsistema, también se plantearon los escenarios que resolverán las funcionalidades del mismo, se identificaron las personas que intervendrán en él y se realizó una estimación del tiempo que se empleará para implementar los escenarios, dicha implementación se llevará a cabo en tres iteraciones. El estudio realizado en este capítulo ha facilitado un entendimiento de las características del Subsistema, contribuyendo a una mejor comprensión del problema que se plantea al inicio de la investigación.



Capítulo 3 Construcción

3.1 Introducción.

En el presente capítulo se especifica la arquitectura del sistema, así como la propuesta de desarrollo del sistema utilizando buenas prácticas y patrones de diseño bien definidos con el objetivo de crear un Subsistema que cumpla con las necesidades del usuario final. También se muestran los diferentes diagramas que propone la metodología de desarrollo en la fase de construcción, para facilitar el desarrollo de la aplicación. Se describe además el uso de determinadas tecnologías y la descripción del modelo de datos.

3.2 Especificación de la Arquitectura.

El primer paso importante en la fase de construcción es la especificación de la Arquitectura de *Software*, la cual a manera de concepto es la organización fundamental de un sistema representada en sus componentes, las relaciones entre ellos, el ambiente y los principios que orientan su diseño y evolución(IEEE 1471-2000). La misma involucra un conjunto de decisiones significativas acerca de la organización del sistema, selecciona sus elementos estructurales y sus interfaces, así como su comportamiento. También involucra funcionalidad, usabilidad, tolerancia a cambios, rendimiento, reutilización y aspectos estéticos(LORENA 2010).

Los Patrones Arquitectónicos son los que definen la estructura de un sistema, los cuales a su vez se componen de subsistemas con sus responsabilidades, también tienen una serie de directivas para organizar los componentes del mismo sistema, con el objetivo de facilitar la tarea del diseño de tal sistema(WILEY JOHN 1996). El Subsistema que se desarrolla posee una arquitectura basada en el modelo de N-Capas, las cuales se especifican más adelante.

3.2.1 Modelo de Capas.

Este modelo (denominado algunas veces modelo de máquina abstracta) organiza el sistema en capas, cada una de las cuales proporciona un conjunto de servicios. Cada capa puede pensarse como una máquina abstracta cuyo lenguaje de máquina se define por los servicios proporcionados por la misma.



“Subsistema de Aprovisionamiento de Usuarios para el Sistema de Administración de Identidades”

Este tipo de arquitectura soporta el desarrollo incremental de sistemas y evita que los cambios en una de las capas afecten directamente al resto (IAN 2006).

La arquitectura del Subsistema planteado se representa a través de cinco capas lógicas que dan un alto nivel de encapsulamiento de las responsabilidades, permitiendo reducir al máximo el acoplamiento y aumentar la reutilización entre las mismas. La descripción de las capas es la siguiente:

- **Capa de Presentación:** se encuentran todas las interfaces o vistas que son mostradas al usuario utilizando el patrón arquitectónico MVC donde están presentes los elementos necesarios para su correcto funcionamiento, entre los cuales se pueden citar: los ficheros de código Java Script, que dan paso a la integración con los componentes de JQuery; así como los archivos CSS que contienen estilos para la aplicación. Es típico encontrar MVC a través de tres capas pero en este caso se encuentra solo en la capa de presentación donde los controladores representan la lógica de la presentación, las vistas representan las interfaces de usuario y el modelo representa los datos de la presentación que son el resultado del procesamiento en la capa de negocio. Esta capa de presentación se encuentra representada por el componente **Gyes.Provisioning.WebApplication** de la aplicación, y tiene interacción directa con la capa de negocio.
- **Capa de Negocio:** contiene las clases controladoras, que son las que manejan todas las acciones sobre las entidades valiéndose del acceso a datos. Se encuentra constituida por el componente **Gyes.Provisioning.Core** y tiene relación directa con las capas: Servicios, Acceso a Datos y Entidades.
- **Capa de Entidades:** contiene las clases entidades del Subsistema que se gestionan en la aplicación, persisten en la base de datos y se muestran en la presentación. Se encuentra constituida por el componente **Gyes.Provisioning.Entities**.
- **Capa de Servicios:** se encuentran los servicios del sistema operativo *windows* y los servicios web que realizan el aprovisionamiento de usuarios, los cuales gestionan la lógica de los procesos de aprovisionamiento representada por flujos de trabajo permitiendo la creación, modificación y eliminación de cuentas de usuarios en los recursos. También permite el bloqueo de las mismas por desuso y su eliminación cuando están bloqueadas. Esta capa está constituida por los componentes: **Gyes.Provisioning.Services**, **Gyes.Provisioning.WindowsServices**,



“Subsistema de Provisiónamiento de Usuarios para el Sistema de Administración de Identidades”

Gyes.Provisioning.WebService.WorkflowActivities. Además tiene relación con la capa de Entidades.

- **Capa de Acceso a Datos:** es el componente que da soporte a las funcionalidades de la capa de negocio y se encuentra relacionado con la fuente de datos. En esta capa se encuentra incluido el *Entity Framework*, herramienta utilizada para la generación del modelo *Entity Data Model (EDM)* y se encarga de la manipulación de la información en la base de datos. Las facilidades que brinda este modelo son aprovechadas por la implementación del Patrón Repositorio que a su vez se abstrae de las dependencias del *Entity Framework* de la base de datos y posibilita un mejor y fácil trabajo con los datos. La principal función de esta capa es realizar una implementación de las funcionalidades definidas en las interfaces de la capa de negocio y al mismo tiempo trabajar directamente con la fuente de datos. Esta capa está constituida por los componentes: **Gyes.Practices.Repository**, **Gyes.Provisioning.DAL**. Además tiene relación con la capa de Entidades y la Fuente de Datos.

Fuente de Datos: está constituida por las tablas que permiten el almacenamiento de la información recolectada y procesada utilizando como Sistema Gestor de Base de Datos *Oracle 11g*.

El siguiente diagrama muestra una vista detallada de la Arquitectura, donde se muestran las capas que componen el Subsistema y las interacciones entre ellas:

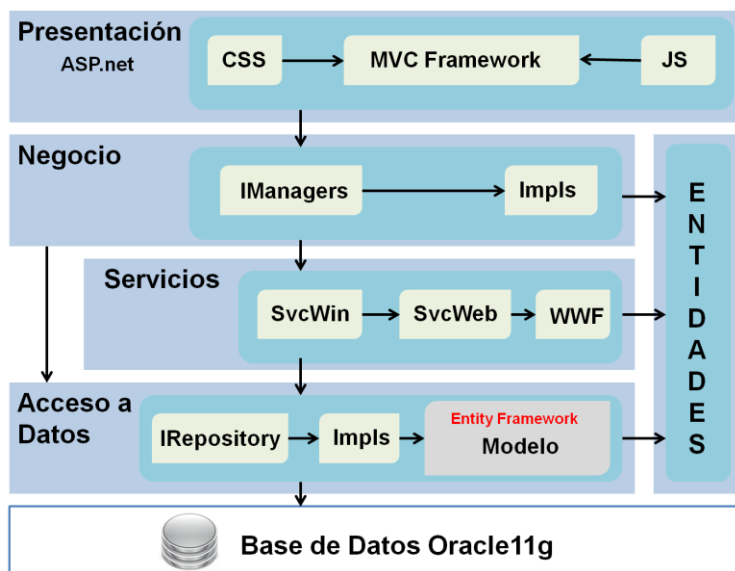


Figura 9: Vista lógica de la arquitectura del software.



3.2.2 Modelo Vista Controlador (MVC).

Modelo-Vista-Controlador (MVC) ha sido un importante patrón arquitectónico en la informática desde hace muchos años. Originalmente llamado Cosa-Modelo-Vista-Editor en 1979, fue simplificada más tarde a Modelo-Vista-Controlador. Es un medio poderoso y elegante de separación de problemas dentro de una aplicación (por ejemplo, separar la lógica de acceso a datos de la lógica de la pantalla) y se aplica muy bien en las aplicaciones web. Su separación explícita de las preocupaciones le añade una pequeña cantidad de complejidad extra al diseño de una aplicación, pero los extraordinarios beneficios compensan el esfuerzo adicional. Además se ha utilizado en decenas de estructuras desde su introducción(GALLOWAY JON 2011).

El MVC separa la interfaz de usuario de una aplicación en tres aspectos principales:

- **El Modelo:** un conjunto de clases que describen los datos con los que trabaja, así como las reglas de negocio sobre cómo los datos pueden ser cambiados y manipulados(GALLOWAY JON 2011).
- **La Vista:** define cómo la interfaz de usuario se mostrará(GALLOWAY JON 2011).
- **El Controlador:** un conjunto de clases que maneja la comunicación por parte del usuario, el flujo general de la aplicación y la lógica específica de la misma(GALLOWAY JON 2011).

En la solución propuesta se utiliza el patrón MVC, permitiendo así la definición de la estructura de la capa de presentación en la arquitectura utilizada y administrando la manera en que los datos son mostrados al usuario. En el caso de dicha solución el modelo lo representan las clases que contienen todas las validaciones necesarias para que los datos pasados a través de las vistas sean validados, de esta manera se utilizan las facilidades brindadas por *ASP MVC 3* para validaciones llamadas *DataAnnotations*, que no son más que validaciones predefinidas por el *framework* y además posibilita la inserción o implementación de anotaciones nuevas. Por otra parte las vistas son las interfaces que se muestran al usuario, dentro de las cuales se utilizan técnicas para mostrar interfaces a partir de objetos adquiridos desde librerías como por ejemplo los datos necesarios para agregar recursos nuevos utilizando la Librería de Conectores y también al cargar las interfaces en dependencia de la plantilla seleccionada, cuya plantilla contiene los atributos necesarios para crear una solicitud de su tipo además de los recursos, roles, preguntas de seguridad y política de contraseña con que contará el usuario que posea una cuenta a partir de dicha plantilla. El controlador simplemente se encarga de manejar todo el flujo de información entre el modelo, las vistas y las entidades que se explican en la arquitectura.



3.3 Patrones de Diseño.

Un Patrón de Diseño es una descripción de un problema y su solución que recibe un nombre y se puede emplear en otros contextos. En teoría indica la manera de utilizarlo en circunstancias diversas(CRAIG 2004). Dentro de los patrones de diseño se encuentran dos grupos fundamentales conocidos por Patrones Generales de Asignación de Responsabilidades de *Software* (GRASP por sus siglas en inglés) y *Banda de los Cuatro* (GOF por sus siglas en inglés). Los Patrones GRASP describen los principios fundamentales de la asignación de responsabilidades a objetos, expresados en forma de patrones(CRAIG 2004). Los Patrones GOF describen las formas comunes en que diferentes tipos de objetos pueden ser organizados para trabajar unos con otros. Tratan la relación entre clases, la combinación de clases y la formación de estructuras de mayor complejidad. Permiten crear grupos de objetos para ayudar a realizar tareas complejas(ERICH *et al.* 1995). Los patrones de diseño GOF se clasifican en tres categorías basadas en su propósito: creacionales, estructurales y de comportamiento:

- **Creacionales:** describen las formas de crear instancias de objetos. El objetivo de estos patrones es de abstraer el proceso de instanciación y ocultar los detalles de cómo los objetos son creados o inicializados(ERICH *et al.* 1995).
- **Estructurales:** describen como las clases y objetos pueden ser combinados para formar grandes estructuras y proporcionar nuevas funcionalidades. Estos objetos adicionales pueden ser incluso objetos simples u objetos compuestos(ERICH *et al.* 1995).
- **Comportamiento:** definen la comunicación e iteración entre los objetos de un sistema. El propósito de este patrón es reducir el acoplamiento entre los objetos(ERICH *et al.* 1995).

Seguidamente serán explicados los patrones GRASP y GOF que fueron utilizados para el desarrollo del Subsistema.

3.3.1 Patrón Experto.

El Patrón Experto perteneciente al grupo de patrones GRASP consiste en asignar una responsabilidad al experto en información: la clase que cuenta con la información necesaria para cumplir la responsabilidad(CRAIG 2004). Con este patrón se pretende que los objetos hagan cosas relacionadas con la información que poseen.



“Subsistema de Provisiónamiento de Usuarios para el Sistema de Administración de Identidades”

```
public class Account
{
    private AccountIdentityAttribute[] _identityAttribute;

    public Dictionary<string, object> GetIdentityAttribute()
    {
        if (_identityAttribute != null)
            return _identityAttribute.ToDictionary<AccountIdentityAttribute, string, object>
                (attr => attr.name, attr => attr.value);
        else
            throw new Exception("Do not exist identity attributes associated with the account.");
    }
}
```

Figura 10: Ejemplo Patrón Experto.

3.3.2 Patrón Creador.

El Patrón Creador perteneciente al grupo de patrones GRASP se basa en asignarle a la clase B la responsabilidad de crear una instancia de la clase A(CRAIG 2004). Este patrón guía la asignación de responsabilidades relacionadas con la creación de objetos y tiene como propósito fundamental encontrar un creador que se debe conectar con el objeto producido en cualquier evento.

```
public abstract class ManagerRepositoryImpl : IManagerRepository
{
    public ElementsRepositoryImpl _elements;
    public RelationsRepositoryImpl _relations;

    protected ManagerRepositoryImpl()
    {
        _elements = new ElementsRepositoryImpl();
        _relations = new RelationsRepositoryImpl();
    }
}
```

Figura 11: Ejemplo Patrón Creador.

3.3.3 Patrón Controlador.

El Patrón Controlador perteneciente al grupo de patrones GRASP no es más que asignar la responsabilidad del manejo de un mensaje de los eventos de un sistema a una clase. Un evento del sistema es un evento de alto nivel generado por un actor externo; es un evento de entrada externa(CRAIG 2004). El patrón propone el diseño de clases con la responsabilidad de controlar el flujo de eventos del sistema a clases específicas.

```
public class ConfigurationManagerImpl : IConfigurationManager
{
    private IConfigurationRepository _configurationRepository;

    public ConfigurationManagerImpl(...)

    public void AddDefaultConfiguration(ConfigurationTask task)...

    public void ModifyConfiguration(ConfigurationTask task, Configuration obj)...

    public Configuration GetConfiguration(ConfigurationTask task)...
}
```

Figura 12: Ejemplo Patrón Controlador.

CAPÍTULO 3: CONSTRUCCIÓN



3.3.4 Patrón Bajo Acoplamiento.

El Patrón Bajo Acoplamiento perteneciente al grupo de patrones GRASP consiste en asignar una responsabilidad para mantener bajo acoplamiento. El acoplamiento es una medida de la fuerza con que una clase está conectada a otras, con las que conoce y con que recurre a ellas (CRAIG 2004). El patrón propone el diseño de clases más independientes, lo que reduce el impacto del cambio y facilita la reutilización en otros sistemas.

```
public class AnswerManagerImpl : IAnswerManager
{
    private IAnswerRepository _answerRepository;

    public AnswerManagerImpl()
    {
        _answerRepository = new AnswerRepositoryImpl();
    }
}
```

Figura 13: Ejemplo Patrón Bajo Acoplamiento.

3.3.5 Patrón Alta Cohesión.

El Patrón Alta Cohesión perteneciente al grupo de patrones GRASP se basa en asignar una responsabilidad de modo que la cohesión siga siendo alta. La cohesión es una medida de cuán relacionadas y enfocadas están las responsabilidades de una clase. Una alta cohesión caracteriza a las clases con responsabilidades estrechamente relacionadas que realicen un trabajo enorme (CRAIG 2004). El patrón propone el diseño de clases con responsabilidades moderadas en su área funcional y que colabore con las otras para llevar a cabo una tarea.

```
public abstract class GenericRepository<C, T> : IRepository<T> where T : class where C : ObjectContext, new()
{
    private C _context = new C();

    public virtual IQueryable<T> GetAll()...

    public virtual IQueryable<T> FindBy(System.Linq.Expressions.Expression<Func<T, bool>> predicate)...

    public virtual void Add(T entity)...

    public virtual void Delete(T entity)...

    public virtual void SaveChanges()...

    public C Context...
}
```

Figura 14: Ejemplo Patrón Alta Cohesión.

3.3.6 Patrón Repositorio.

El Patrón Repositorio es un patrón estructural perteneciente al grupo de patrones GOF que permite de una forma sencilla, hacer que las capas de datos se puedan probar y trabajar de una forma más simétrica a la



“Subsistema de Aprovisionamiento de Usuarios para el Sistema de Administración de Identidades”

orientación a objetos con los modelos relacionales. Un repositorio realiza las tareas de intermediario entre las capas de modelo de dominio y mapeo de datos. Los objetos clientes construyen de forma declarativa consultas y las envían a los repositorios para que las satisfagan. Conceptualmente, un repositorio encapsula a un conjunto de objetos almacenados en la base de datos y las operaciones que sobre ellos pueden realizarse. Para cada tipo de objeto lógico que necesite acceso global, se debe crear un objeto (Repositorio) que proporcione la apariencia de una colección en memoria de todos los objetos de ese tipo. Se debe establecer el acceso mediante una interfaz bien conocida, proporcionar métodos para añadir y eliminar objetos, que realmente encapsularán la inserción o eliminación de datos en el almacén de datos. Proporcionar métodos que seleccionen objetos basándose en ciertos criterios de selección y devuelvan objetos o colecciones de objetos instanciados (entidades del dominio) con los valores de dicho criterio, de forma que encapsule el almacén real (base de datos, etc.) y la tecnología base de consulta (CÉSAR *et al.* 2011).

Aprovechando todas las ventajas expresadas anteriormente y en función de las características del Subsistema, se realizó una implementación del Patrón Repositorio donde se implementan las funcionalidades globales de una interfaz general, así como interfaces por cada una de las tablas de la base de datos, que heredan de la interfaz general y cuentan además con funcionalidades específicas. Las implementaciones de estas interfaces contienen las nuevas operaciones y heredan de la implementación general, lo que le permite utilizar las operaciones de la misma. Además existe una interfaz con su implementación que expone un grupo de operaciones que comprenden las relaciones entre los diferentes repositorios. Todo esto es posible por el Contexto que crea los repositorios y mapea las Clases *POCO*, que no son más que una representación de las tablas de la base de datos, con el modelo generado por el *Entity Framework*, al cual se le realizan las consultas y el contexto se encarga de hacer los cambios en la fuente de datos. El diagrama de clases de la implementación de este patrón se puede encontrar en el **Anexo VII**.

3.3.7 Patrón Observador.

Observador (*Observer*) es un patrón de comportamiento perteneciente al grupo de patrones GOF que define una dependencia de uno a muchos entre objetos, de forma que cuando un objeto cambie de estado se notifique y actualicen automáticamente todos los objetos que dependen de él (ERICH *et al.* 1995). En la solución este patrón se evidencia específicamente en la relación entre entidades, un ejemplo de ello es la



relación compuesta por Plantilla (*Template*) y Recurso (*Resource*), las plantillas contienen una lista de recursos y en el momento en que los recursos se modifican los elementos modificados que están contenidos en la plantilla también se modifican.

3.3.8 Patrón Recuerdo.

Recuerdo (*Memento*) es un patrón de comportamiento perteneciente al grupo de patrones GOF que posibilita volver a estados anteriores del sistema (ERICH *et al.* 1995). Este patrón es muy utilizado en el proceso de aprovisionamiento de usuarios de la solución, en el momento en que se intentan crear las cuentas de usuarios en los recursos. Si alguno de estos intentos falla el sistema revierte todo lo que se realizó y vuelve al estado anterior.

3.4 Diagrama de Aplicación.

Después de entender la arquitectura del Subsistema así como algunos de los patrones utilizados, se muestra otro de los elementos importantes que define la metodología utilizada, el diagrama de aplicación. Este se crea con el objetivo de dar una perspectiva de todos los componentes que se relacionen en la aplicación. Es una solución de ámbito y muestra los elementos de despliegue, como los servicios web, aplicaciones web, aplicaciones de *Windows* y las aplicaciones de referencia, tales como bases de datos externas y servicios web externos. El diagrama muestra las conexiones entre las aplicaciones que reflejan su configuración actual en la solución (MICROSOFT 2007a). Este artefacto se puede apreciar en el **Anexo VIII**.

3.5 Diagrama Lógico de Centro de Datos.

Otro artefacto definido dentro de la metodología planteada es el diagrama lógico de centro de datos, el cual define o documenta las configuraciones específicas de un *software* de servidor de aplicaciones, tales como *Internet Information Services*, *SQL Server* o *BizTalk Server*, que tienen un propósito específico como un servidor web seguro para el usuario. El diagrama muestra cómo estos servidores lógicos están conectados entre sí. Los mismos pueden agruparse dentro de las zonas que definen los límites lógicos de comunicación. Las zonas pueden ser configuradas para limitar los tipos de servidores lógicos que pueden contener la dirección y tipos de comunicación (MICROSOFT 2007a).



“Subsistema de Aprovisionamiento de Usuarios para el Sistema de Administración de Identidades”

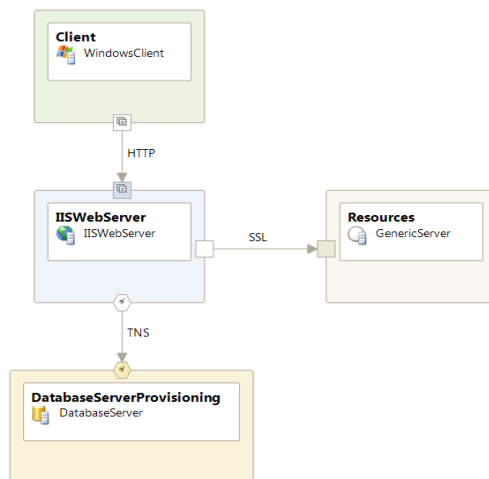


Figura 15: Diagrama Lógico de Centro de Datos del Subsistema de Aprovisionamiento.

El diagrama muestra la interacción entre los servidores lógicos del Subsistema y los protocolos de comunicación. Cuenta con clientes que a través del protocolo HTTP se comunican con el servidor web donde se encuentran hospedadas la aplicación y los servicios, es importante resaltar que estos dos pueden estar o no en servidores independientes, en función de las necesidades y características de la entidad. El servidor web interactúa con los diferentes recursos asociados al Subsistema donde se realizará el aprovisionamiento, utilizando SSL como protocolo de comunicación. Por otra parte también se comunica a través del protocolo TNS con el servidor de base de datos, donde persiste toda la información de los usuarios.

3.6 Diagrama de Clases.

Un diagrama de clases representa una agrupación lógica o física de las clases y sus relaciones(MICROSOFT 2007a). Los diagramas de clases pertenecientes al Subsistema se encuentran organizados por componentes y se pueden apreciar en el **Anexo IX**.

3.7 Uso de *Windows Workflow Foundation*.

Para entender un poco más cómo funciona la solución que se plantea, es importante especificar el uso de la tecnología *Windows Workflow Foundation*. Los flujos de trabajo son la base del aprovisionamiento del Subsistema ya que estos son los encargados de guiar este proceso, a través de actividades para conectarse a los diferentes servidores, así como permiten realizar todas las operaciones sobre los



recursos. Además se realizan salvadas de las tareas completadas satisfactoriamente para que si el Subsistema falla, se restaure el proceso y comience por donde se encontraba anteriormente. El uso de flujos de trabajo en el aprovisionamiento de usuarios posibilita que en caso de fallar la conexión de red o el recurso no esté disponible, la tarea pare y espere hasta que sea posible restablecer la comunicación con el mismo, para completar las acciones sobre él. También los flujos de trabajo constituyen una parte fundamental en el proceso de solicitud de cuentas de usuarios, es decir que dicho proceso no es más que un flujo de trabajo que recibe como entrada un objeto de tipo Solicitud con todos los datos necesarios para solicitar una cuenta de usuario y dentro del mismo se define qué interfaz gráfica debe mostrarse al solicitante o a los revisores definidos en el objeto Solicitud. Dicho flujo permite además que el proceso se rija por un conjunto de pasos necesarios a partir de los intereses de la empresa, pues en caso de que una entidad determinada, cambie la cantidad o el tipo de revisores, solo hay que adaptar el flujo de trabajo sin necesidad de modificar el código de la aplicación. Otra ventaja es que funciona a partir del estado que contiene el objeto solicitud, lo cual permite que si se interrumpe el proceso de solicitud, cuando este se recupere, se encontrará en el estado en el que se encontraba anteriormente. Se puede apreciar el flujo lógico que representa un flujo de trabajo de aprovisionamiento de usuarios del Motor de Tareas, en el **Anexo X**.

3.8 Modelo de Datos del Sistema.

Un modelo de datos es un conjunto de conceptos que permiten describir los datos, las relaciones que existen entre ellos, la semántica y las restricciones de consistencia. Además es una representación lógica y física de los datos persistentes usados por la aplicación, permite describir:

- **Las Estructuras de Datos:** los tipos de datos y la forma en que se relacionan.
- **Las Restricciones de Integridad:** condiciones que deben cumplir los datos para reflejar correctamente la realidad deseada.
- **Operaciones de Manipulación de los Datos:** operaciones sobre la información en la base de datos.

El modelo de datos del Subsistema está formado por dos tablas, Elementos y Relaciones. Los elementos están compuestos por un identificador, un tipo y los datos. Los tipos de elementos así como los datos contenidos en un XML, están definidos por un XSD. Por otra parte, la tabla Relaciones contiene las relaciones entre elementos; los tipos de relaciones están definidas en un XSD y está compuesta por un



identificador para la relación, el identificador del titular de la relación, el identificador del elemento relacionado con el titular y el tipo de relación que estos poseen.

Este modelo es una variante de un patrón de diseño conocido como EVA (*Entity Value Attribute*). El cual permite tener un dominio detallado sobre todos los atributos que se le asignan a cualquier elemento que es almacenado. Las principales características de EVA, que a su vez se convierten en sus principales ventajas son la gran flexibilidad en el almacenamiento de datos y la posibilidad de ampliar el conjunto de atributos sin cambiar la estructura de la tabla.

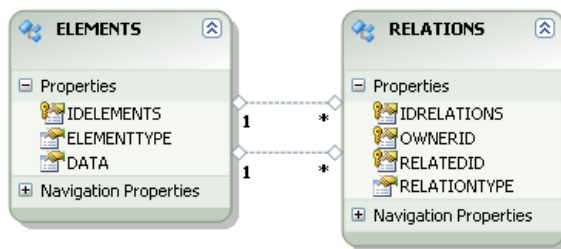


Figura 16: Diagrama de Clases del Modelo de Datos.

3.9 Interfaz Gráfica.

Las interfaces pertenecientes al Subsistema se encuentran disponibles en el [Anexo XI](#).

3.10 Conclusiones.

Este capítulo abarca una serie de conclusiones importantes en la fase de construcción del Subsistema. Entre ellas se encuentra la descripción de la arquitectura donde se definió el modelo de N-capas, especificando cuáles capas formarían parte de la solución así como la interacción entre ellas. Se utilizaron patrones de diseño para identificar los aspectos importantes de la estructura del diseño del Subsistema. Se especificaron los diagramas de aplicación y lógico de centro de datos para una mayor comprensión del funcionamiento del Subsistema en general. También se mostraron los diagramas de clases pertenecientes a cada una de las capas que conforman la arquitectura, para conocer la estructura de las mismas. Otro elemento importante en este capítulo es el uso de *Windows Workflow Foundation* para la realización del proceso de solicitud y los servicios de aprovisionamiento que se brindarán en el Subsistema.



Capítulo 4 Prueba

4.1 Introducción.

Luego de haber abordado la fundamentación teórica de la investigación y expresar las características del Subsistema a través de la visión y planificación, así como los principales elementos a tener en cuenta para la etapa de construcción del *software*; pues comienza la etapa de estabilización del Subsistema. De acuerdo a la metodología de desarrollo que se utiliza, se deben especificar los tipos de prueba que serán usados, donde se pueden destacar las pruebas unitarias a través de la técnica de caja blanca. Así como el uso de la técnica de caja negra para las funcionalidades de la aplicación. También se harán uso de las pruebas de rendimiento a los servicios web utilizados en el Subsistema.

4.2 Pruebas de Caja Blanca.

Se basa fundamentalmente en desarrollar pruebas de forma que se asegure que la operación interna se ajusta a las especificaciones, y que todos los componentes internos se hayan probado de forma adecuada (MANUEL 2012). Las pruebas de caja blanca conocidas también como pruebas de caja de cristal o pruebas estructurales se centran en los detalles procedimentales del *software*, por lo que su diseño está de gran manera ligado al código fuente. Para la realización de este tipo de pruebas, el probador escoge diferentes valores de entrada para examinar cada uno de los posibles flujos de ejecución del programa y cerciorarse de que se devuelven los valores de salida adecuados. Aunque las pruebas de caja blanca son aplicables a varios niveles como son unidad, integración y sistema, habitualmente se aplican a las unidades de *software* (LAURIE 2006).

4.3 Pruebas Unitarias.

Luego de especificar el concepto de pruebas de caja blanca, es importante abordar acerca de las pruebas de unidad. En la programación, una prueba unitaria o de unidad es una forma de probar el correcto funcionamiento de un módulo de código. Esto sirve para asegurar que cada una de las partes que integran la aplicación funcionen correctamente por separado (CARLOS, BENÍTEZ 2011). Las pruebas unitarias se realizan para controlar el funcionamiento de pequeñas porciones de código como son subprogramas (en la programación estructurada) o métodos (en la programación orientada a objeto POO). Generalmente son



*“Subsistema de Aprovisionamiento de Usuarios para el Sistema de
Administración de Identidades”*

realizadas por los mismos programadores puesto que al conocer con mayor detalle el código, se les simplifica la tarea de elaborar conjuntos de datos de prueba para probarlo. (PABLO and CARLOS 2003). El *IDE Visual Studio 2010* permite la realización de pruebas de unidad a los métodos y a continuación se muestran las pruebas unitarias realizadas a algunas de las funcionalidades más importantes dentro de la Gestión de Plantillas. También se pueden encontrar pruebas unitarias realizadas a funcionalidades pertenecientes a la Gestión de Solicitudes en el **Anexo XII**.

Prueba de Unidad				
Nombre Prueba: AddTemplateTest.				
Estado: Satisfactoria.	Tipo: Caja Blanca.	Ultima Ejecución: 17/05/2012		
Ejecutado por: José Carlos Correa B.	Verificado por: Wilhem Manuel Verano.			
Descripción: Para la ejecución de esta prueba se deben seleccionar los roles con que contarán los usuarios pertenecientes a la plantilla que se creará, los recursos a los que tendrán acceso, las preguntas de seguridad que debe responder en caso de recuperar la contraseña, así como la política de contraseña que será utilizada para la creación de cuentas utilizando dicha plantilla. También se deben crear los atributos de identidad necesarios para el aprovisionamiento de usuarios en los recursos seleccionados.				
Entrada: Template template, string path.				
Criterio de aceptación: Crea una plantilla.				
Resultado:				
✔ Test run completed Results: 1/1 passed; Item(s) checked: 0				
	Result	Test Name	Project	Error Message
<input type="checkbox"/>	✔ Passed	AddTemplateTest	WebApplicationTest	

Tabla 5: Descripción de la prueba unitaria AddTemplateTest.

Prueba de Unidad			
Nombre Prueba: ModifyTemplateDataTest.			
Estado: Satisfactoria.	Tipo: Caja Blanca.	Ultima Ejecución: 17/05/2012	
Ejecutado por: José Carlos Correa B.	Verificado por: Wilhem Manuel Verano.		
Descripción: Para la ejecución de esta prueba se debe seleccionar la plantilla de la lista de plantillas y acceder a la opción "Editar Plantilla", luego se modifican los datos de la misma, ya sean los roles, los recursos, las preguntas de seguridad, la política de contraseña o los atributos de identidad.			
Entrada: string oldName, Template template, string path.			
Criterio de aceptación: Modifica la plantilla.			



“Subsistema de Aprovisionamiento de Usuarios para el Sistema de
Administración de Identidades”

Resultado:			
Test run completed Results: 1/1 passed; Item(s) checked: 0			
Result	Test Name	Project	Error Message
Passed	ModifyTemplateDataTest	WebApplicationTest	

Tabla 6: Descripción de la prueba unitaria ModifyTemplateDataTest.

Prueba de Unidad			
Nombre Prueba: RemoveTemplateTest.			
Estado: Satisfactoria.	Tipo: Caja Blanca.	Ultima Ejecución: 17/05/2012	
Ejecutado por: José Carlos Correa B.	Verificado por: Wilhem Manuel Verano.		
Descripción: Para la realización de esta prueba se debe seleccionar la plantilla de la lista de plantillas y acceder a la opción "Eliminar Plantilla", luego se elimina la plantilla del sistema.			
Entrada: <i>string</i> name, <i>string</i> path.			
Criterio de aceptación: Elimina la plantilla.			
Resultado:			
Test run completed Results: 1/1 passed; Item(s) checked: 0			
Result	Test Name	Project	Error Message
Passed	RemoveTemplateTest	WebApplicationTest	

Tabla 7: Descripción de la prueba unitaria RemoveTemplateTest.

4.4 Pruebas de Caja Negra.

Esta técnica consiste básicamente en realizar pruebas de forma tal que se compruebe que cada función es operativa. Las pruebas de caja negra se llevan a cabo sobre la interfaz del *software*, obviando el comportamiento interno y la estructura del programa. Los casos de prueba de caja negra pretenden demostrar que las funciones del *software* son operativas, que la entrada se acepta de forma correcta, que se produce una salida correcta y que la integridad de la información externa se mantiene (MANUEL 2012).

Escenario	Descripción	Texto	Respuesta del Sistema	Flujo Central
Esc. 1.1: Crear política de contraseña.	Creación de una nueva política de contraseña en el	Nombre (Válido) Registro Máximo (Válido) Edad Mínima (Válida) Edad Máxima (Válida) Aviso de Expiración (Válido)	El Subsistema crea una nueva política de contraseña y muestra una notificación de creación exitosa.	Módulo Portal de Administración / Opción Gestionar Políticas /Selección de la Política de Contraseña/



*“Subsistema de Aprovechamiento de Usuarios para el Sistema de
Administración de Identidades”*

	Subsistema.	Política (Válida) Descripción (Válida)		Opción Crear Política de Contraseña.
		Nombre (Inválido) Registro Máximo (Inválido) Edad Mínima (Inválida) Edad Máxima (Inválida) Aviso de Expiración (Inválido) Política (Inválida) Descripción (Inválida)	El Subsistema muestra un mensaje de error informando que se introdujeron datos incorrectos.	
Esc. 1.2: Modificar política de contraseña.	Modificación de una política de contraseña en el Subsistema.	Nombre (Válido) Registro Máximo (Válido) Edad Mínima (Válida) Edad Máxima (Válida) Aviso de Expiración (Válido) Política (Válida) Descripción (Válida)	El Subsistema actualiza los datos de la política de contraseña y muestra una notificación de modificación exitosa.	Módulo Portal de Administración / Opción Gestionar Políticas / Selección de la Política de Contraseña / Opción Editar Política de Contraseña.
		Nombre (Inválido) Registro Máximo (Inválido) Edad Mínima (Inválida) Edad Máxima (Inválida) Aviso de Expiración (Inválido) Política (Inválida) Descripción (Inválida)	El Subsistema muestra un mensaje de error informando que se introdujeron datos incorrectos.	
Esc. 1.3: Eliminar política de contraseña.	Eliminación de una política de contraseña del Subsistema.	Válida	El Subsistema elimina la política de contraseña seleccionada y muestra una notificación de eliminación exitosa.	Módulo Portal de Administración / Opción Gestionar Políticas / Selección de la Política de Contraseña / Opción Eliminar Política de Contraseña.
		Inválida	El Subsistema muestra un mensaje de error informando se realizó una acción indebida.	
Esc. 1.4: Listar políticas de contraseña.	Muestra todas las políticas de contraseña.	Válido	El Subsistema lista todas las políticas de contraseña existentes en el mismo.	Módulo Portal de Administración / Opción Gestionar Políticas.

Tabla 8: Caso de Prueba Gestionar Políticas de Contraseña.



“Subsistema de Aprovechamiento de Usuarios para el Sistema de
Administración de Identidades”

#	Nombre del Campo	Clasificación	Valor Nulo	Descripción
1	Nombre	Campo de Texto	No	Puede ser cualquier combinación de caracteres alfanuméricos incluyendo el espacio.
2	Registro Máximo	Campo de Texto	No	Puede ser cualquier número entero positivo.
3	Edad Mínima	Campo de Texto	No	Puede ser cualquier número entero positivo menor que el campo edad máxima.
4	Edad Máxima	Campo de Texto	No	Puede ser cualquier número entero positivo mayor que el campo edad mínima.
5	Aviso de Expiración	Campo de Texto	No	Puede ser cualquier número entero positivo.
6	Política	Campo de Texto	No	Una expresión regular que valide la contraseña a la que se le asigne.
2	Descripción	Campo de Texto	No	Puede ser cualquier combinación de caracteres alfanuméricos incluyendo el espacio.

Tabla 9: Descripción de las variables de entrada del Caso de Prueba Gestionar Políticas de Contraseña

4.6 Pruebas de Rendimiento.

Después de realizar las pruebas anteriores, también es necesario aplicarle pruebas de rendimiento al Subsistema. Las pruebas de rendimiento son aquellas que son realizadas para determinar qué tan rápido un sistema realiza una tarea bajo ciertas condiciones pre-planificadas de trabajo. Estas pruebas también son utilizadas para validar y verificar diferentes aspectos de la calidad de *software*, como por ejemplo, escalabilidad, fiabilidad y el buen uso de los recursos. Existen diferentes tipos de pruebas de rendimiento, las cuales son muy útiles, siempre y cuando sean usadas a través de herramientas que permitan visualizar los detalles técnicos y gráficos de los resultados de las pruebas (SYBVEN 2012).

Entre los tipos de pruebas de rendimiento más importantes se encuentran:

- **Pruebas de Carga:** es el tipo más sencillo de pruebas de rendimiento. Una prueba de carga se realiza generalmente para observar el comportamiento de una aplicación bajo una cantidad de peticiones esperada. Esta carga puede ser el número esperado de usuarios concurrentes utilizando la aplicación y que realizan un número específico de transacciones durante el tiempo



*“Subsistema de Aprovisionamiento de Usuarios para el Sistema de
Administración de Identidades”*

que dura la carga. Esta prueba puede mostrar los tiempos de respuesta de todas las transacciones importantes de la aplicación(SYBVEN 2012).

- **Prueba de Estrés:** se utiliza normalmente para romper la aplicación. Se va doblando el número de usuarios que se agregan a la aplicación y se ejecuta una prueba de carga hasta que se rompe. Este tipo de prueba se realiza para determinar la solidez de la aplicación en los momentos de carga extrema y ayuda a los administradores a determinar si la aplicación rendirá lo suficiente en caso de que la carga real supere a la carga esperada(SYBVEN 2012).

Al Subsistema se le realizaron específicamente Pruebas de Carga, utilizando dos servidores con características técnicas diferentes, arrojando distintos resultados.

Indicadores	Servidor - 1	Servidor - 2
Microprocesador	Pentium Dual Core 3.00 GHz	I3 - M330 2.13 GHz
Memoria RAM	2 Gb	4 Gb

Tabla 10: Características de los Servidores.

Atendiendo a lo anteriormente explicado los resultados de las pruebas fueron los siguientes:

Indicadores	Servidor - 1	Servidor - 2
Tiempo de Inicio	23/05/2012 - 1:16:27	23/05/2012 - 0:25:02
Tiempo de Terminado	23/05/2012 - 1:21:27	23/05/2012 - 0:25:02
Duración	00:05:00	00:05:00
Numero de Hilos	10	10
Números de Pruebas	219	410
Número de Fallos	22	10
Porcentaje de Fallos	10	2,44
Promedio de Tiempos de Respuesta	6,59 milisegundos	2,17 milisegundos
Porcentaje de Microprocesador	13,9 %	11,4 %
Memoria Disponible	551 Mb	1450 Mb

Tabla 11: Resultados de la Prueba de Carga.



“Subsistema de Aprovisionamiento de Usuarios para el Sistema de Administración de Identidades”

Comportándose durante la prueba de la siguiente forma:

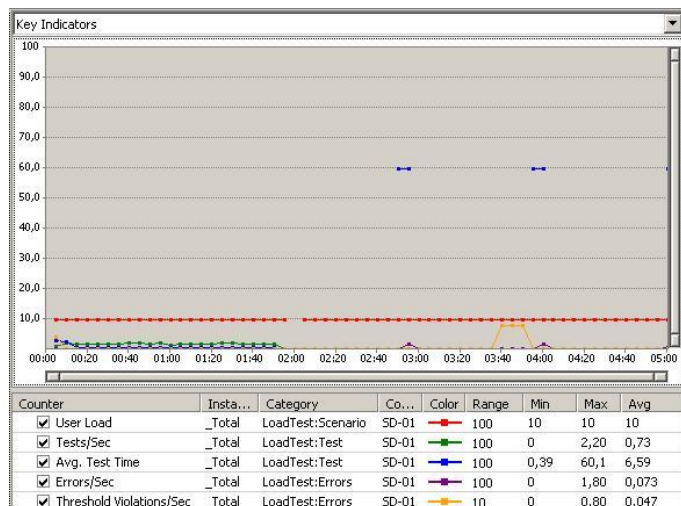


Figura 17: Servidor #1.

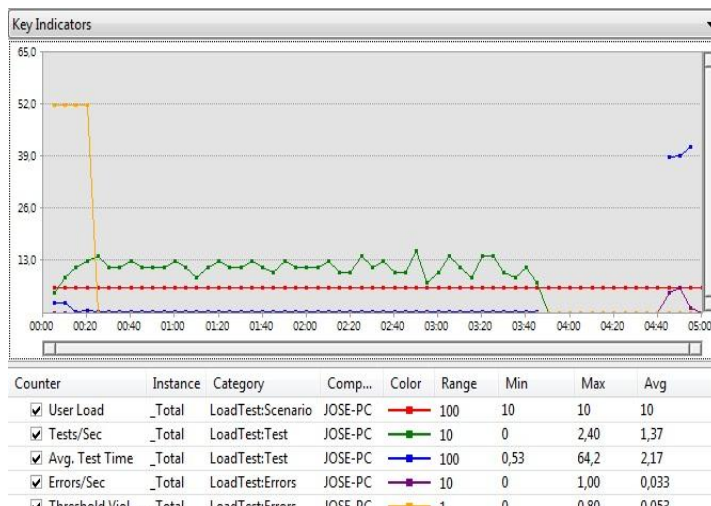


Figura 18: Servidor #2.

De manera general los resultados de las pruebas fueron satisfactorios demostrando el buen rendimiento del Subsistema. Evidenciándose además que a medida que el servidor sea más potente mejor será el comportamiento del Subsistema.

4.7 Conclusiones.

Luego de la realización de pruebas unitarias, se comprobó el correcto funcionamiento de procedimientos internos del *software*. Con el uso de casos de prueba se validó que las funciones a nivel de interfaces de usuario fueran operativas. También a través del uso de pruebas de rendimiento se verificó la rapidez del Subsistema luego de la realización de tareas bajo ciertas condiciones. A partir del empleo de estos tipos de pruebas se encontraron un conjunto de no conformidades, las cuales fueron la base fundamental para que el Subsistema mejorara su rendimiento y funcionamiento. Además las pruebas realizadas brindaron un resultado satisfactorio que le aporta al Subsistema mayor nivel de robustez y validez.



CONCLUSIONES GENERALES.

A partir del desarrollo de la presente investigación se arribaron a las siguientes conclusiones:

- Luego de realizar un estudio de los principales Sistemas de Administración de Identidades enfocándose en la estructura y el funcionamiento de sus Sistemas de Aprovisionamiento de Usuarios, se logró adquirir mayor visión y entendimiento del proceso de aprovisionamiento.
- Se realizó un estudio de las herramientas y tecnologías necesarias para el desarrollo del Subsistema, definiendo así los lenguajes, herramientas y métodos a seguir para su implementación.
- La descripción de escenarios presentes en la fase de planificación definida por la metodología MSF para el Desarrollo de *Software* Ágil, permitió el conocimiento de las funcionalidades del Subsistema.
- Se realizó el plan de iteraciones que permitió la organización del tiempo y el trabajo con el fin de culminar el desarrollo del Subsistema en el tiempo requerido.
- La arquitectura del Subsistema se representó a través de cinco capas lógicas que brindan un alto nivel de encapsulamiento de las responsabilidades, permitiendo reducir al máximo el acoplamiento y aumentar la reutilización entre las mismas.
- Las pruebas realizadas permitieron comprobar las funcionalidades del Subsistema, estas brindaron resultados satisfactorios otorgando validez a la investigación.
- Con el desarrollo del Subsistema de Aprovisionamiento de Usuarios para el Sistema de Administración de Identidades, se permite la gestión dinámica de las cuentas de usuarios y los recursos asociados de forma genérica.



*“Subsistema de Aprovisionamiento de Usuarios para el Sistema de
Administración de Identidades”*

RECOMENDACIONES.

Se recomienda a los interesados en mejorar o continuar esta investigación:

- Implementar otras versiones de este Subsistema incorporando funcionalidades nuevas que sean útiles para lograr un mejor producto, como por ejemplo la implementación de más conectores para que el Subsistema aprovisiona usuarios en más recursos por defecto.
- Realizar la integración con las demás aplicaciones que conformarán el Sistema de Administración de Identidades.
- Desarrollar el manual de usuario del Subsistema para capacitar al personal que lo utilizará.
- Presentar la investigación en eventos de cortes científicos.



REFERENCIAS BIBLIOGRÁFICAS.

- ACADEMIA, D. L. L. E. R. *Identidad*, 2010. 22.
- ALTOVA. *Altova MissionKit*, 2011a. [Disponible en: <http://www.altova.com/es/missionkit/software-development-tools.html>].
- *DatabaseSpy*, 2011b. [Disponible en: <http://www.altova.com/es/databasespy.html>].
- *DiffDog*, 2011c. [Disponible en: <http://www.altova.com/es/diffdog/diff-merge-tool.html>].
- *MapForce*, 2011d. [Disponible en: <http://www.altova.com/es/mapforce.html>].
- *SchemaAgent*, 2011e. [Disponible en: <http://www.altova.com/es/schemaagent.html>].
- *SemanticWorks*, 2011f. [Disponible en: <http://www.altova.com/es/semanticworks.html>].
- *StyleVision*, 2011g. [Disponible en: <http://www.altova.com/es/stylevision.html>].
- *UModel*, 2011h. [Disponible en: <http://www.altova.com/es/umodel.html>].
- *XMLSpy*, 2011i. [Disponible en: <http://www.altova.com/es/xmlspy.html>].
- AUTOMATIONS, A. *PLSQL Developer*, 2012. [Disponible en: <http://www.allroundautomations.com/plsqldev.html>].
- BHUVAN URGAONKAR , P. S., ABHISHEK CHANDRAY , PAWAN GOYALZ. *Agile, Dynamic Provisioning of Multitier Internet Applications*, 2008.
- CARLOS, B. *JUnit, testeando nuestras aplicaciones*, 2011.
- CARLOS, O. R. J. *Metodología de Desarrollo MSF*. Universidad Interamericana para el Desarrollo, 2008.
- CÉSAR, T. L.; J. C. NELSON, *et al. Guía de Arquitectura N-Capas orientada al Dominio con .Net 4.0*. MICROSOFT, 2011.
- CISED. *Proyecto Técnico Sistema de Administración de Identidades.*, 2012.
- CRAIG, L. *UML y Patrones Introducción al análisis y diseño orientado a objetos.*, 2004. p. 8420534382.
- ERICH, G.; H. RICHARD, *et al. Design Patterns: Elements of Reusable Object-Oriented Software*. 1995. p. 0201633612.
- EVOLUCY, T. C. S. L. *Identidad Digital Evolucy Technology Consulting S.L*, 2012.
- GALLOWAY JON , H. P., WILSON BRAD , SCOTT ALLEN K. *Professional ASP.NET MVC 3*. COPYRIGHT © 2011 BY JOHN WILEY & SONS, I. I., INDIANA, John Wiley & Sons, Inc., 2011.



*“Subsistema de Aprovisionamiento de Usuarios para el Sistema de
Administración de Identidades”*

- IAN, S. *Ingeniería del Software*, 2006. Séptima Edición.
- IBM. *Manual de consulta*, 2012. [Disponible en: http://publib.boulder.ibm.com/tividd/td/tec/GC32-0666-02/es_ES/HTML/TECref112.htm.
- *Provisioning User Guide*, 2011.
- IEEE. *Arquitectura*, 1471-2000.
- IGNACIO, A. D. and U. A. XAVIER. *La Gestión de Identidades y Capacidades por las Administraciones Públicas*, 2006.
- JAMES, S. A. F. *Administración*. México, Prentice-Hall Hispanoamericana, 1994. p. 968-880-351-0.
- JOSÉ, G. S. A. *El lenguaje de programación C#*, 2008.
- LAURIE, W. *White-Box Testing*, 2006.
- LORENA, E. *Arquitectura de un Sistema de Información*, 2010.
- MANUEL, T. G. *Técnicas de Prueba*, 2012.
- MARIO, M. A. and V. P. CARLOS. *Gestor de tareas para Aprovisionamiento de Usuarios*. Seguridad Digital. Habana, Universidad de las Ciencias Informáticas, 2010. p.
- MARIO, R. D. L. M. SIGMAPA - PASSI, plataforma para la gestión de personas *TECNIMAP*, 2007.
- MICROSOFT. *Características de Entity Framework*, 2011a. [Disponible en: <http://msdn.microsoft.com/es-es/library/bb896338%28v=vs.90%29.aspx>.
- *Características de Visual Studio 2010 Ultimate*, 2011b. [Disponible en: <http://www.microsoft.com/spain/visualstudio/products/2010-editions/ultimate>.
- Code First en ADO.Net Entity Framework 4.1 *MSDN*, 2011c.
- *Detalles de las características de WCF*, 2011d.
- *Detalles de las características de Windows Workflow Foundation*, 2011e. [Disponible en: <http://msdn.microsoft.com/es-es/library/gg427656.aspx>.
- *Detalles y Características de .Net Framework*, 2011f. [Disponible en: <http://www.microsoft.com/downloads/es-es/details.aspx?FamilyID=9cfb2d51-5ff4-4491-b0e5-b386f32c0992>.
- *Getting Started with ASP.net MVC 3*, 2011g. [Disponible en: <http://www.asp.net/mvc/tutorials/getting-started-with-aspnet-mvc3>.
- Microsoft Provisioning System Component Overview *TechNet*, 2011h.
- *MSF for Agile Software Development*, 2007a.

REFERENCIAS BIBLIOGRÁFICAS



“Subsistema de Aprovisionamiento de Usuarios para el Sistema de
Administración de Identidades”

- MICROSOFT, C. Gestión de Identidades *Microsoft TechNet*, 2004.
- *Información general de flujos de trabajo.*, 2007b.
- NOVELL. *Identity Manager Standard Edition*, 2012. [Disponible en: <http://www.novell.com/es-es/products/identitymanagerstandard/features/#provisioning>].
- ORACLE. *Oracle Identity Manager Architecture*, 2010.
- PABLO, S. and F. CARLOS Documentación y pruebas Antes del paradigma de objetos, 2003.
- ROGER, P. S. *Ingeniería de software. Un enfoque práctico*. 7ma. 2007. p.
- SERGIO, S. G. and G. O. ANA. *Gestión de Identidad en las Administraciones Públicas: Interoperabilidad pan -Europea*. España, CIBSI, 2009.
- STOCKERS, I. T. Introducción a Oracle y Oracle Database 11g Versión *Informática & Tecnología Stockers*, 2012.
- SUN, M. *La Integración del Aprovisionamiento de la Auditoría de Identidades*. Estados Unidos, 2006a.
- *Solución ELCM Gestión del Ciclo de Vida de los Empleados*, 2006b.
- SYBVEN, C. *Pruebas de Rendimiento*, 2012. [Disponible en: http://www.corporacionsybven.com/portal/index.php?option=com_content&view=article&id=246].
- UCI. *Introducción a la Seguridad Informática*. D.S.D, 2011.
- UZKIAGA Qué es la identidad digital y herramientas para crearla *UZKIAGA*, 2012.
- VIVIANA, L. M. and M. L. ARMANDO. *Módulo Administración para la Aplicación de Aprovisionamiento de Usuarios del Sistema de Administración de Identidades*. Seguridad Digital. Habana, Universidad de las Ciencias Informáticas, 2010. p.
- WILEY JOHN , S. *Pattern-Oriented Software Architecture, Buschmann, F. et. al*, 1996.
- WILHEM, M. V. and R. M. ERNESTO. *Módulo de Conectores para el Subsistema de Aprovisionamiento de Usuarios del Sistema Administración de Identidades.*: Seguridad Digital. Habana, UCI, 2011. p.
- WSO2. *WSO2 Identity Server*, 2012. [Disponible en: <http://wso2.com/products/identity-server/>]

REFERENCIAS BIBLIOGRÁFICAS



GLOSARIO DE TÉRMINOS.

A

AJAX: técnica de desarrollo web para crear aplicaciones interactivas. Estas aplicaciones se ejecutan en el cliente, mientras se mantiene la comunicación asíncrona con el servidor. De esta forma es posible realizar cambios sobre las páginas sin necesidad de recargarlas.

Aplicación: cualquier programa que corra en un sistema operativo y que haga una función específica para un usuario. Por ejemplo, procesadores de palabras, bases de datos, agendas electrónicas, etc.

API (*Application Programming Interface*): interfaz de programación de aplicaciones, conjunto de convenciones internacionales que definen cómo debe invocarse una determinada función de un programa desde una aplicación.

ASP: es un *framework* para aplicaciones web desarrollado y comercializado por *Microsoft*. Es usado por programadores para construir sitios web dinámicos, aplicaciones web y servicios web XML.

Autenticación: la autenticación es el acto de proveer una identidad a una red, aplicación o recurso. Las técnicas de autenticación van desde una simple entrada de identificador (ID) de usuario y contraseña hasta potentes mecanismos como los ficheros, certificados de clave pública y biométrica.

Autorización: es el proceso de decidir si una identidad digital es permitida para ejecutar una acción de respuesta. La autorización sucede después de la autenticación y usa atributos o derechos, asociados con la identidad digital para determinar a qué recursos puede acceder dicha identidad digital.

B

Base de Datos: conjunto exhaustivo no redundante de datos estructurados organizados independientemente de su utilización y su implementación en máquina, accesibles en tiempo real y compatibles con usuarios concurrentes con necesidad de información diferente y no predicable en tiempo.

C

Clases POCO (*Plain Old CLR Object*): También conocidos como objetos que ignoran la persistencia. Son clases de datos asignadas a entidades definidas en un modelo de datos, admiten la mayoría de los mismos comportamientos de consulta, inserción, actualización y eliminación que los tipos de entidad generados por las herramientas *Entity Data Model*.

Common Language Runtime (CLR): entorno de ejecución para los códigos de los programas que corren sobre la plataforma *Microsoft .NET*. El CLR es el encargado de compilar una forma de código intermedio



“Subsistema de Provisiónamiento de Usuarios para el Sistema de
Administración de Identidades”

llamada *Common Intermediate Language*, al código de máquina nativo, mediante un compilador en tiempo de ejecución.

Contraseñas: una contraseña o clave (en inglés *password*) es una forma de autenticación que utiliza información secreta para controlar el acceso hacia algún recurso. Pueden contener caracteres alfanuméricos e incluso otros símbolos.

Conectores: son los encargados de garantizar la interacción con los recursos que va a utilizar el Subsistema.

Cronometrar: cronometrar el desarrollo de un *software* es tener el control del tiempo de vida de este.

D

DLL (*Dynamic Link Library*): la Biblioteca de Vínculos Dinámicos es un archivo que contiene funciones que se pueden llamar desde aplicaciones u otras DLL. Los desarrolladores utilizan las DLL para poder reciclar el código y aislar las diferentes tareas.

E

EDM (*Entity Data Model*): es un conjunto de conceptos que describen la estructura de los datos, independientemente del formato en el que estén almacenados. EDM se basa en el modelo entidad-relación, pero también incorpora nuevas funciones y amplía sus usos tradicionales.

EJB (*Enterprise Java Bean*): es un programa en lenguaje java que realiza alguna gestión compleja y que aguarda a que otro programa cliente lo invoque.

Entidad: representación de un objeto individual concreto del mundo real. Cualquier tipo de objeto o concepto sobre el que se recoge información puede ser una cosa, persona, concepto abstracto o suceso.

Escalabilidad: la capacidad del sistema informático de cambiar su tamaño o configuración para adaptarse a las circunstancias cambiantes.

E

Framework: estructura de soporte definida en la cual otro proyecto de *software* puede ser organizado y desarrollado. Típicamente, puede incluir soporte de programas, bibliotecas y un lenguaje interpretado entre otros *software* para ayudar a desarrollar y unir los diferentes componentes de un proyecto.

I

IDE: *software* compuesto por un conjunto de herramientas de programación. Es un entorno de programación que ha sido empaquetado como un programa de aplicación, es decir, consiste en un editor de código, un compilador, un depurador y un constructor de interfaz gráfica (GUI).



“Subsistema de Aprovechamiento de Usuarios para el Sistema de
Administración de Identidades”

Interoperabilidad: capacidad de comunicación entre diferentes programas y máquinas de distintos fabricantes.

K

KDC (*Key Distribution Server*): centro de distribución de claves de Kerberos, este servicio permite que los usuarios inicien sesión en la red utilizando el protocolo de autenticación Kerberos.

L

LDAP: es un protocolo a nivel de aplicación que permite el acceso a un servicio de directorio ordenado y distribuido para buscar diversa información en un entorno de red.

LINQ (*Language Integrated Query*): es un conjunto de características en *Visual Studio* que agrega eficaces capacidades de consulta a la sintaxis de los lenguajes *C Sharp* y *Visual Basic*.

M

Módulo: componente auto controlado de un sistema que posee una interfaz bien definida hacia otros componentes.

O

OAuth: es un protocolo abierto que permite autenticarse a través de las API.

Openfire: sistema de mensajería instantánea hecho en java, que utiliza el protocolo XMPP.

OpenID: es un estándar de identificación digital descentralizado, con el que un usuario puede identificarse en una página web a través de una URL y puede ser verificado por cualquier servidor que soporte el protocolo.

P

Plantillas: agrupamiento de usuarios por sus características, que pueden ser roles, atributos de identidad, recursos a los que tienen acceso, preguntas de seguridad asociadas a los mismos y las políticas de contraseña.

R

RabbitMQ: *software* de negociación de mensajes de código abierto, el cual se encuentra dentro de la categoría de *middleware* de mensajería. Implementa el estándar *Advanced Message Queuing Protocol* (AMQP), es un estándar abierto en la capa de aplicaciones de un sistema de comunicación. Las características que definen al protocolo AMQP son la orientación a mensajes, encolamiento, enrutamiento, exactitud y seguridad.



REST (*Representational State Transfer*): la Transferencia de Estado Representacional es una técnica de arquitectura de *software* para sistemas hipermedia distribuidos como la *World Wide Web*.

S

SAML: es una versión de la norma OASIS SAML para la autenticación y el intercambio de datos de autorización entre dominios de seguridad.

Servidor: un servidor es una computadora que maneja peticiones de datos, correo, servicios de redes y transferencia de archivos de otras computadoras (clientes). También puede referirse a un *software* específico. Una computadora puede tener distintos *software* de servidor, proporcionando muchos servidores a clientes en la red.

Single Sign-On (SSO): es un procedimiento de autenticación que habilita al usuario para acceder a varios sistemas con una sola instancia de identificación.

SPML (*Service Provisioning Markup Language*): el lenguaje de marcado de servicios de aprovisionamiento es el estándar abierto para la integración y la interoperación de las peticiones del servicio de aprovisionamiento.

SQL: lenguaje declarativo de acceso a bases de datos (BD) relacionales que permite especificar diversos tipos de operaciones en éstas. Una de sus características es el manejo del álgebra y el cálculo relacional permitiendo efectuar consultas con el fin de recuperar información de interés de una BD.

SSH: este intérprete de órdenes es un protocolo que facilita las comunicaciones seguras entre dos sistemas usando una arquitectura cliente - servidor y que permite a los usuarios conectarse a un host remotamente. A diferencia de otros protocolos de comunicación remota, este encripta la sesión de conexión, haciendo imposible que alguien pueda obtener contraseñas no encriptadas.

SCIM (*Simple Cloud Identity Management*): la gestión única de la identidad de las nubes está diseñada para hacer la gestión de la identidad del usuario en las aplicaciones basadas en nube y servicios más fáciles.

SOAP (*Simple Object Access Protocol*): es un protocolo estándar que define cómo dos objetos en diferentes procesos pueden comunicarse por medio de intercambio de datos XML.

I

Transacciones: una transacción es un conjunto de procesos que se ejecutan uno después del otro. Si algún subproceso falla, lo realizado anteriormente debe deshacerse para que los datos no se alteren, a este comportamiento se le denomina todo o nada.



U

Usuario: persona que tiene una cuenta en una determinada computadora por medio de la cual puede acceder a los recursos y servicios que ofrece una red.

X

XAML: es un lenguaje declarativo de marcado, el archivo .XAML especifica la estructura del flujo de trabajo en términos de dominio de las actividades específicas.

XACML (*Extensible Access Control Markup Language*): el Lenguaje de Marcado Extensible de Control de Acceso es un estándar para la gestión de la política de control de acceso. Fue diseñado para convertirse en un estándar universal para la descripción de quién tiene acceso a los recursos.

XKMS (*XML Key Management Specification*): la especificación XML para manejo de claves, usa el *framework* de servicios web para facilitar a los desarrolladores asegurar la comunicación entre sus aplicaciones utilizando la infraestructura de clave pública (PKI). Es un protocolo desarrollado por el *World Wide Web Consortium* (W3C) que describe la distribución y registro de claves públicas.

XML (*Extensible Markup Language*): lenguaje de marcado extensible, es un metalenguaje extensible de etiquetas desarrollado por el *World Wide Web Consortium* (W3C).

XMPP: protocolo extensible de mensajería y comunicación de presencia, es un protocolo abierto y extensible basado en XML, originalmente ideado para mensajería instantánea. Con el protocolo XMPP queda establecida una plataforma para el intercambio de datos XML que puede ser usada en aplicaciones de mensajería instantánea.

XSD: *XML Schema* es un lenguaje de esquema utilizado para describir la estructura y las restricciones de los contenidos de los documentos XML de una forma muy precisa, más allá de las normas sintácticas impuestas por el propio lenguaje XML.

Z

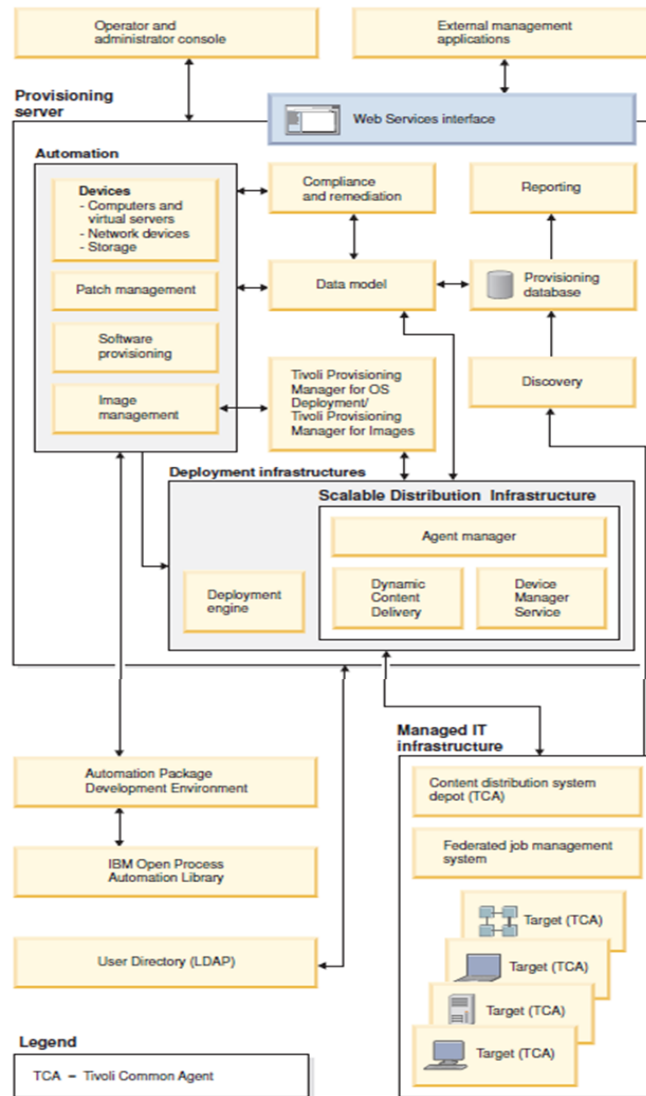
ZCS (*Zimbra Collaboration Suite*): es un *software* de correo electrónico que permite a los empleados de oficina enviar, recibir, guardar y buscar mensajes. Posee tanto el componente de servidor como su respectivo cliente. Hace uso de tecnologías de código abierto existentes tales como *Postfix*, *MySQL* y *OpenLdap*.



“Subsistema de Aprovisionamiento de Usuarios para el Sistema de Administración de Identidades”

ANEXOS.

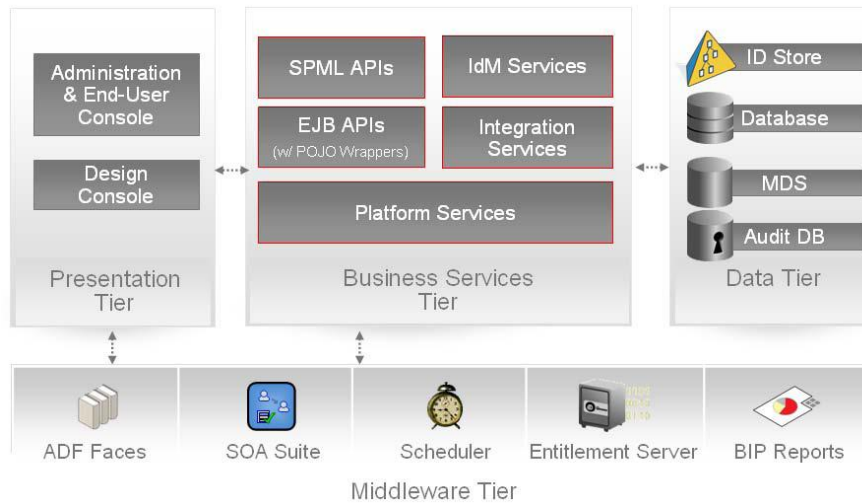
Anexo I: Arquitectura del Sistema de Aprovisionamiento IBM Tivoli(IBM 2011).



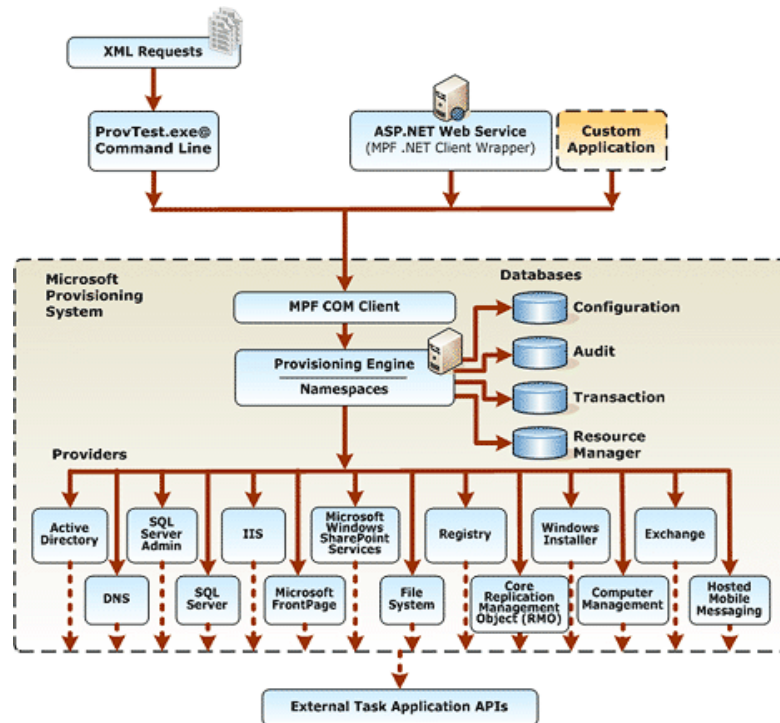


"Subsistema de Aprovisionamiento de Usuarios para el Sistema de Administración de Identidades"

Anexo II: Arquitectura de Oracle Identity Manager 11g(ORACLE 2010).



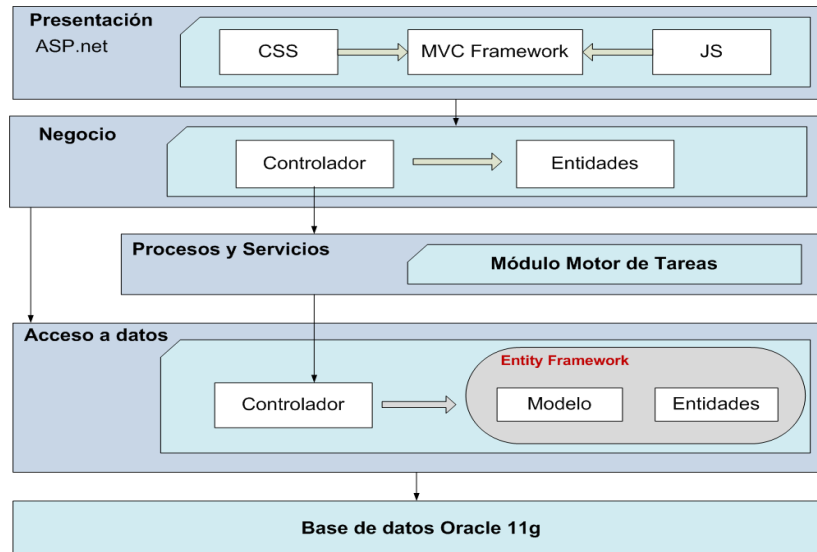
Anexo III: Arquitectura del Sistema de Aprovisionamiento de Microsoft(MICROSOFT 2011h).





“Subsistema de Aprovisionamiento de Usuarios para el Sistema de Administración de Identidades”

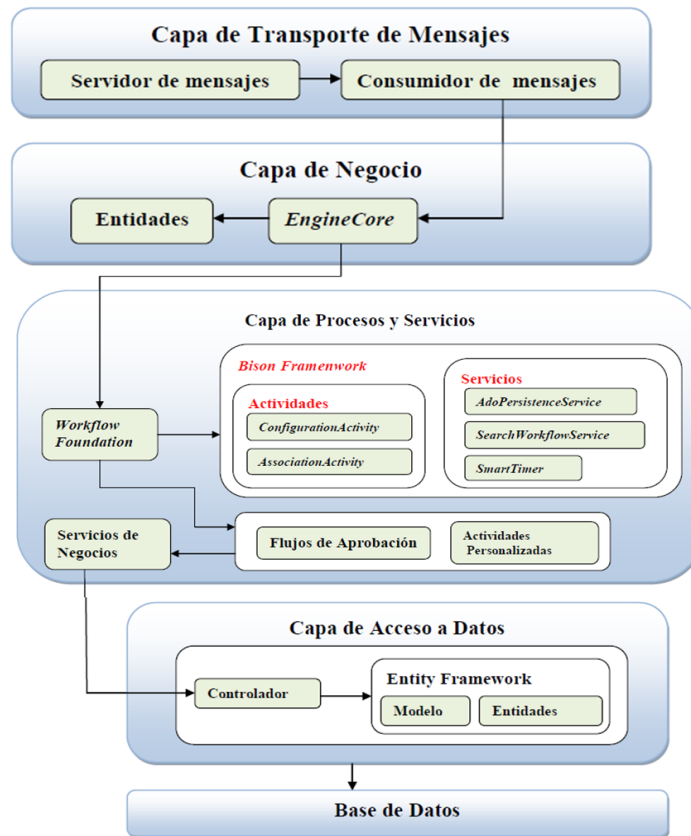
Anexo IV: Vista lógica de la arquitectura de *software* del Módulo de Administración para la Aplicación de Aprovisionamiento de Usuarios del Sistema de Administración de Identidades(VIVIANA and ARMANDO 2010).





“Subsistema de Aprovisionamiento de Usuarios para el Sistema de Administración de Identidades”

Anexo V: Vista lógica del módulo Gestor de Tareas para el Aprovisionamiento de Usuarios(MARIO, MEZQUINA AZAHARES and CARLOS 2010).





*“Subsistema de Aprovisionamiento de Usuarios para el Sistema de
Administración de Identidades”*

Anexo VI: Descripción de Escenarios.

Módulo Portal de Usuario:

Nombre del Escenario: Aprobar solicitud.		Identificador: Ap_PU1.
Objetivo del Escenario: Permitir aprobar una solicitud de creación de cuenta de usuario.		
Persona: Aprobador de Solicitudes.		
Iteración: 3ra.	Prioridad: 3.	Complejidad: Alta.
Descripción: El aprobador de solicitudes accede al sistema y selecciona la opción Administrar Solicitudes, el sistema muestra un listado con todas las solicitudes. El aprobador de solicitudes selecciona una solicitud que se encuentre en estado revisada y selecciona la opción Cargar Solicitud. El sistema muestra la solicitud con sus datos. El aprobador de solicitudes selecciona la opción Aprobar y el sistema crea la cuenta de usuario en los recursos contenidos en la solicitud, en caso de que no sea posible crearla, la solicitud pasa a estado aprobada y más tarde se puede reintentar el aprovisionamiento.		
Validaciones: -		

Nombre del Escenario: Revocar solicitud.		Identificador: Ap_PU2.
Objetivo del Escenario: Permitir revocar una solicitud de creación de cuenta de usuario.		
Persona: Aprobador de Solicitudes.		
Iteración: 3ra.	Prioridad: 3.	Complejidad: Alta.
Descripción: El aprobador de solicitudes accede al sistema y selecciona la opción Administrar Solicitudes, el sistema muestra un listado con todas las solicitudes. El aprobador de solicitudes selecciona una solicitud que se encuentre en estado revisada y selecciona la opción Cargar Solicitud. El sistema muestra la solicitud con sus datos. El aprobador de solicitudes selecciona la opción Revocar y el sistema pasa la solicitud a estado revocada.		
Validaciones: -		

Nombre del Escenario: Revisar solicitud.		Identificador: Ap_PU3.
Objetivo del Escenario: Permitir revisar una solicitud de creación de cuenta de usuario.		
Persona: Revisor de Solicitudes.		
Iteración: 3ra.	Prioridad: 3.	Complejidad: Alta.
Descripción: El revisor de solicitudes accede al sistema y selecciona la opción Administrar Solicitudes, el sistema muestra un listado con todas las solicitudes. El revisor de solicitudes selecciona una solicitud que se encuentre en estado pendiente o desacuerdo y selecciona la opción Cargar Solicitud. El sistema muestra la solicitud con sus datos para ser revisados además del campo “No Conformidades” para que escriba lo que no está correcto en la solicitud. El revisor de solicitudes escribe la no conformidad y		



*“Subsistema de Aprovisionamiento de Usuarios para el Sistema de
Administración de Identidades”*

selecciona la opción Revisar.
Validaciones: -

Nombre del Escenario: Listar solicitudes.	Identificador: Ap_PU4.	
Objetivo del Escenario: Permitir listar las solicitudes de creación de cuenta de usuario existentes en el sistema.		
Persona: Revisor de Solicitudes, Aprobador de Solicitudes.		
Iteración: 3ra.	Prioridad: 3.	Complejidad: Alta.
Descripción: El revisor de solicitudes o aprobador de solicitudes accede al sistema y selecciona la opción Administrar Solicitudes, el sistema muestra un listado con todas las solicitudes.		
Validaciones: -		

Nombre del Escenario: Solicitar creación de cuenta de usuario.	Identificador: Ap_PU5.	
Objetivo del Escenario: Permitir solicitar la creación de una cuenta de usuario en el sistema.		
Persona: Usuario Estándar.		
Iteración: 3ra.	Prioridad: 3.	Complejidad: Baja.
Descripción: El usuario accede a la aplicación con la intención de solicitar la creación de una cuenta de usuario. Luego selecciona la opción Solicitar Cuenta y selecciona la plantilla de cuenta de usuario. Se muestran a partir de dicha plantilla una serie de campos a ser llenados de manera obligatoria. El usuario introduce los datos correspondientes y selecciona el botón Crear. Finalmente, el sistema realiza las validaciones necesarias y si procede la información entonces se crea la solicitud de cuenta de usuario con sus datos correspondientes.		
Validaciones: El sistema debe validar que: <ul style="list-style-type: none">➤ El usuario no se encuentre registrado en la Base de Datos.➤ La contraseña introducida coincida con la política de contraseña perteneciente a la plantilla seleccionada.		

Nombre del Escenario: Realizar seguimiento de solicitud de creación de cuenta.	Identificador: Ap_PU6.	
Objetivo del Escenario: Permitir el seguimiento de solicitud de creación de una cuenta de usuario.		
Persona: Usuario Estándar.		
Iteración: 3ra.	Prioridad: 3.	Complejidad: Baja.
Descripción: El usuario accede a la aplicación y selecciona la opción Seguimiento Solicitud, el sistema muestra entonces un formulario para		



*“Subsistema de Aprovisionamiento de Usuarios para el Sistema de
Administración de Identidades”*

ser rellenado con el usuario del solicitante de la cuenta. Finalmente el sistema muestra en pantalla la siguiente información de:

- Usuario solicitante.
- Usuario de la cuenta.
- Estado actual de la solicitud.

Validaciones:

El sistema debe validar que:

- El usuario solicitante existe en la base de datos.

Nombre del Escenario: Recuperar contraseña.		Identificador: Ap_PU7.
Objetivo del Escenario: Permitir la recuperación de la contraseña del usuario.		
Persona: Usuario Estándar		
Iteración: 3ra.	Prioridad: 3.	Complejidad: Baja.
Descripción: El usuario accede al sistema y selecciona la opción Recuperar Contraseña, el sistema muestra entonces un formulario a ser llenado con los datos requeridos, entre los cuales se encuentra el campo Usuario que debe introducirse de manera obligatoria. Después de introducidos los datos por el usuario, el sistema muestra una serie de preguntas de seguridad que deberán ser respondidas. El sistema realiza verificaciones en la Base de Datos y es generada una contraseña aleatoria para ser enviada por correo electrónico o alguna vía alternativa.		
Validaciones: <ul style="list-style-type: none">➤ Se valida el usuario contra la Base de Datos para mostrar las preguntas necesarias para la recuperación.➤ Se valida que las respuestas correspondan con las respuestas almacenadas del usuario en cuestión.		

Nombre del Escenario: Cambiar contraseña.		Identificador: Ap_PU8.
Objetivo del Escenario: Permitir que el usuario pueda cambiar su contraseña.		
Persona: Usuario Estándar.		
Iteración: 3ra.	Prioridad: 3.	Complejidad: Baja.
Descripción: El usuario accede al sistema y selecciona la opción Cambiar contraseña, el sistema muestra un formulario a ser rellenado con los datos requeridos para poder cambiar la contraseña: <ul style="list-style-type: none">➤ Usuario.➤ Contraseña Actual.➤ Contraseña Nueva.➤ Confirmar Contraseña Nueva. Después del usuario introducir estos datos se selecciona la opción Cambiar Contraseña, el sistema realiza las validaciones y		



*“Subsistema de Aprovisionamiento de Usuarios para el Sistema de
Administración de Identidades”*

finalmente es cambiada la contraseña registrándose en la Base de Datos.

Validaciones:

- Se valida que la contraseña actual coincida con la contraseña guardada en la base de datos.
- Se valida la contraseña nueva y su confirmación.
- Se valida que la nueva contraseña presente los requisitos mínimos de acceso que requiere la plantilla.

Módulo Portal de Administración:

Nombre del Escenario: Crear Conector.		Identificador: Ap_Adm9.
Objetivo del Escenario: Permitir la creación de un conector en el sistema.		
Persona: Administrador.		
Iteración: 1ra.	Prioridad: 5.	Complejidad: Alta.
Descripción: El administrador selecciona la opción Gestionar Conectores. El sistema muestra en pantalla un listado con todos los conectores definidos. El administrador selecciona la opción Crear Conector y se muestra un formulario con los datos y el archivo requeridos: <ul style="list-style-type: none">➤ Nombre.➤ Descripción.➤ Archivo Zip. Luego de introducidos los datos y el archivo se selecciona la opción Crear y el sistema guarda el conector incluyendo las librerías que se encuentran dentro del archivo Zip.		
Validaciones: <ul style="list-style-type: none">➤ El nombre del conector debe coincidir con el nombre de la librería que se encuentra dentro del archivo Zip y que contiene las clases CreateResource y RelatedAttribute.		

Nombre del Escenario: Editar Conector.		Identificador: Ap_Adm10.
Objetivo del Escenario: Permitir la modificación de un conector en el sistema.		
Persona: Administrador.		
Iteración: 1ra.	Prioridad: 5.	Complejidad: Alta.
Descripción: El administrador selecciona la opción Gestionar Conectores. El sistema muestra en pantalla un listado con todos los conectores definidos. El administrador selecciona la opción Editar Conector y se muestra un formulario con los datos a modificar y el archivo es requerido nuevamente: <ul style="list-style-type: none">➤ Nombre.➤ Descripción.		



*“Subsistema de Aprovisionamiento de Usuarios para el Sistema de
Administración de Identidades”*

- Archivo Zip.

Luego de modificados los datos e introducido el archivo se selecciona la opción Crear y el sistema modifica el conector.

Validaciones: -

Nombre del Escenario: Eliminar Conector.		Identificador: Ap_Adm11.
Objetivo del Escenario: Permitir la eliminación de un conector del sistema.		
Persona: Administrador.		
Iteración: 1ra.	Prioridad: 5.	Complejidad: Alta.
Descripción: El administrador selecciona la opción Gestionar Conectores. El sistema muestra en pantalla un listado con todos los conectores definidos. El administrador selecciona el conector que desea eliminar, selecciona la opción Eliminar Conector y el sistema elimina el conector.		
Validaciones: -		

Nombre del Escenario: Listar Conectores.		Identificador: Ap_Adm12.
Objetivo del Escenario: Permitir listar los conectores existentes en el sistema.		
Persona: Administrador.		
Iteración: 1ra.	Prioridad: 5.	Complejidad: Alta.
Descripción: El administrador selecciona la opción Gestionar Conectores. El sistema muestra en pantalla un listado con todos los conectores definidos.		
Validaciones: -		

Nombre del Escenario: Crear política de contraseña.		Identificador: Ap_Adm13.
Objetivo del Escenario: Permitir la creación de una política de contraseña de acceso al sistema.		
Persona: Administrador.		
Iteración: 1ra.	Prioridad: 5.	Complejidad: Media.
Descripción: El administrador accede a la aplicación y selecciona la opción Gestionar Políticas. El sistema muestra en pantalla un listado con todas las políticas de contraseña definidas, selecciona la opción Crear Política de Contraseña y se muestra un formulario con los datos requeridos: <ul style="list-style-type: none">➤ Nombre.➤ Registro máximo de la contraseña.➤ Edad máxima de la contraseña.➤ Edad mínima de la contraseña.		



*“Subsistema de Aprovisionamiento de Usuarios para el Sistema de
Administración de Identidades”*

- Aviso de expiración.
- Política de la contraseña.
- Descripción.

Luego de introducidos los datos se selecciona la opción Crear y se guarda la política de contraseña en el sistema.

Validaciones: -

Nombre del Escenario: Editar política de contraseña.		Identificador: Ap_Adm14.
Objetivo del Escenario: Permitir la modificación de una política de contraseña de acceso al sistema.		
Persona: Administrador.		
Iteración: 1ra.	Prioridad: 5.	Complejidad: Media.
Descripción: El administrador accede a la aplicación y selecciona la opción Gestionar Políticas. El sistema muestra en pantalla un listado con todas las políticas de contraseña definidas, selecciona la opción Editar Política de Contraseña y se muestra un formulario con los datos a modificar: <ul style="list-style-type: none">➤ Nombre.➤ Registro máximo de la contraseña.➤ Edad máxima de la contraseña.➤ Edad mínima de la contraseña.➤ Aviso de expiración.➤ Política de la contraseña.➤ Descripción. Luego de modificados los datos se selecciona la opción Editar y se guarda la política de contraseña modificada en el sistema.		
Validaciones: -		

Nombre del Escenario: Eliminar política de contraseña.		Identificador: Ap_Adm15.
Objetivo del Escenario: Permitir la eliminación de una política de contraseña de acceso al sistema.		
Persona: Administrador.		
Iteración: 1ra.	Prioridad: 5.	Complejidad: Media.
Descripción: El administrador accede a la aplicación y selecciona la opción Gestionar Políticas. El sistema muestra en pantalla un listado con todas las políticas de contraseña definidas, selecciona la política de contraseña que desea eliminar y luego la opción Eliminar Política de Contraseña. El sistema elimina la política de contraseña del sistema.		
Validaciones: -		



*“Subsistema de Aprovisionamiento de Usuarios para el Sistema de
Administración de Identidades”*

Nombre del Escenario: Listar políticas de contraseña.		Identificador: Ap_Adm16.
Objetivo del Escenario: Permitir listar las políticas de contraseña de acceso al sistema.		
Persona: Administrador.		
Iteración: 1ra.	Prioridad: 5.	Complejidad: Media.
Descripción: El administrador accede a la aplicación y selecciona la opción Gestionar Políticas. El sistema muestra en pantalla un listado con todas las políticas de contraseña definidas en el sistema.		
Validaciones: -		

Nombre del Escenario: Crear pregunta de seguridad.		Identificador: Ap_Adm17.
Objetivo del Escenario: Permitir la creación de una pregunta de seguridad en el sistema.		
Persona: Administrador.		
Iteración: 1ra.	Prioridad: 5.	Complejidad: Media.
Descripción: El administrador accede al sistema y selecciona la opción Gestionar Preguntas, el sistema muestra un listado con todas las preguntas de seguridad. El administrador selecciona la opción Crear Pregunta de Seguridad y se muestra un formulario con los datos requeridos: <ul style="list-style-type: none">➤ Pregunta de Seguridad.➤ Descripción. Luego de introducir los datos se selecciona la opción Crear y se guarda la pregunta de seguridad en el sistema.		
Validaciones: -		

Nombre del Escenario: Editar pregunta de seguridad.		Identificador: Ap_Adm18.
Objetivo del Escenario: Permitir la modificación de una pregunta de seguridad en el sistema.		
Persona: Administrador.		
Iteración: 1ra.	Prioridad: 5.	Complejidad: Media.
Descripción: El administrador accede al sistema y selecciona la opción Gestionar Preguntas, el sistema muestra un listado con todas las preguntas de seguridad. El administrador selecciona una pregunta de seguridad de la lista y luego la opción Editar Pregunta de Seguridad y se muestra un formulario con los datos a modificar: <ul style="list-style-type: none">➤ Pregunta de Seguridad.➤ Descripción. Luego de introducir los datos se selecciona la opción Editar y se guarda la pregunta de seguridad editada en el sistema.		
Validaciones: -		



*“Subsistema de Aprovisionamiento de Usuarios para el Sistema de
Administración de Identidades”*

Nombre del Escenario: Eliminar pregunta de seguridad.		Identificador: Ap_Adm19.
Objetivo del Escenario: Permitir la eliminación de una pregunta de seguridad en el sistema.		
Persona: Administrador.		
Iteración: 1ra.	Prioridad: 5.	Complejidad: Media.
Descripción: El administrador accede al sistema y selecciona la opción Gestionar Preguntas, el sistema muestra un listado con todas las preguntas de seguridad. El administrador selecciona una pregunta de seguridad de la lista, luego la opción Eliminar Pregunta de Seguridad y se elimina la pregunta de seguridad del sistema.		
Validaciones: -		

Nombre del Escenario: Listar preguntas de seguridad.		Identificador: Ap_Adm20.
Objetivo del Escenario: Permitir listar las preguntas de seguridad existentes en el sistema.		
Persona: Administrador.		
Iteración: 1ra.	Prioridad: 5.	Complejidad: Media.
Descripción: El administrador accede al sistema y selecciona la opción Gestionar Preguntas, el sistema muestra un listado con todas las preguntas de seguridad.		
Validaciones: -		

Nombre del Escenario: Crear recurso.		Identificador: Ap_Adm21.
Objetivo del Escenario: Permitir la creación de un recurso en el sistema.		
Persona: Administrador.		
Iteración: 2da.	Prioridad: 4.	Complejidad: Alta.
Descripción: El administrador accede al sistema y selecciona la opción Administrar Recursos, el sistema muestra en pantalla un listado de los recursos de la organización. El administrador selecciona la opción Crear Recurso, luego selecciona el conector necesario para crear el recurso y se muestra un formulario con los datos necesarios para conectarse desde el sistema al recurso. Estos datos vienen definidos dentro de una de las librerías existentes en el conector seleccionado. Algunos de ellos pueden ser: <ul style="list-style-type: none">➤ Nombre del recurso.➤ Descripción.➤ IP del Servidor.➤ Usuario.➤ Contraseña.		



*“Subsistema de Aprovisionamiento de Usuarios para el Sistema de
Administración de Identidades”*

Validaciones: -

Nombre del Escenario: Editar recurso.		Identificador: Ap_Adm22.
Objetivo del Escenario: Permitir la modificación de los datos de un recursos en el sistema.		
Persona: Administrador.		
Iteración: 2da.	Prioridad: 4.	Complejidad: Alta.
Descripción: El administrador accede al sistema y selecciona la opción Administrar Recursos, el sistema muestra en pantalla un listado de los recursos de la organización. El administrador selecciona la opción Editar Recurso y se muestra un formulario con los datos a ser editados. Estos datos dependen del recurso, ya que son creados a partir de conectores que tienen definidos dentro de una de sus librerías los datos necesarios para crearlos. Algunos de los datos pueden ser: <ul style="list-style-type: none">➤ Nombre del recurso.➤ Descripción.➤ IP del Servidor.➤ Usuario.➤ Contraseña. Luego de editar los datos se selecciona la opción editar y el sistema edita los datos.		
Validaciones: -		

Nombre del Escenario: Eliminar recurso.		Identificador: Ap_Adm23.
Objetivo del Escenario: Permitir la eliminación de un recurso en el sistema.		
Persona: Administrador.		
Iteración: 2da.	Prioridad: 4.	Complejidad: Alta.
Descripción: El administrador accede al sistema y selecciona la opción Administrar Recursos, el sistema muestra en pantalla un listado de los recursos de la organización. El administrador selecciona la opción Eliminar Recurso y se elimina el recurso del sistema.		
Validaciones: -		

Nombre del Escenario: Habilitar o deshabilitar recurso.		Identificador: Ap_Adm24.
Objetivo del Escenario: Permitir la habilitación o no de un recurso en el sistema para el aprovisionamiento de usuarios.		
Persona: Administrador.		
Iteración: 2da.	Prioridad: 4.	Complejidad: Alta.
Descripción:		



*“Subsistema de Aprovisionamiento de Usuarios para el Sistema de
Administración de Identidades”*

El administrador accede al sistema y selecciona la opción Administrar Recursos, el sistema muestra en pantalla un listado de los recursos de la organización. El administrador selecciona la opción Habilitar o Deshabilitar Recurso, en caso de estar habilitado se deshabilita y en caso de estar deshabilitado se habilita para el aprovisionamiento de usuarios.

Validaciones: -

Nombre del Escenario: Listar recursos.		Identificador: Ap_Adm25.
Objetivo del Escenario: Permitir listar los recursos existentes en el sistema.		
Persona: Administrador.		
Iteración: 2da.	Prioridad: 4.	Complejidad: Alta.
Descripción: El administrador accede al sistema y selecciona la opción Administrar Recursos, el sistema muestra en pantalla un listado de los recursos de la organización.		
Validaciones: -		

Nombre del Escenario: Crear plantilla.		Identificador: Ap_Adm26.
Objetivo del Escenario: Permitir la creación de una plantilla para la solicitud de creación de cuentas de usuarios.		
Persona: Administrador.		
Iteración: 2da.	Prioridad: 4.	Complejidad: Alta.
Descripción: El administrador accede al sistema y selecciona la opción Gestionar Plantillas, el sistema muestra un listado con todas las plantillas. El administrador selecciona la opción Crear Plantilla y se muestra un formulario con los datos requeridos: <ul style="list-style-type: none">➤ Nombre.➤ Descripción.➤ Roles (Marcar los roles que tendrán los usuarios creados a partir de la plantilla).➤ Recursos (Marcar los recursos a los que tendrán acceso los usuarios creados a partir de la plantilla).➤ Preguntas de Seguridad (Marcar las preguntas de seguridad que se les mostrarán cuando se necesite recuperar la contraseña de algún usuario creado a partir de la plantilla).➤ Política de Contraseña (Marcar la política de contraseña para validar la contraseña de acceso al sistema). Posteriormente el administrador selecciona la opción Siguiente, el sistema muestra una interfaz para gestionar los atributos de identidad necesarios para la plantilla. Luego el administrador selecciona la opción Crear y el sistema guarda la plantilla.		
Validaciones: -		

Nombre del Escenario: Editar plantilla.	Identificador: Ap_Adm27.
--	---------------------------------



*“Subsistema de Aprovisionamiento de Usuarios para el Sistema de
Administración de Identidades”*

Objetivo del Escenario: Permitir la modificación de una plantilla para la solicitud de creación de cuentas de usuarios.		
Persona: Administrador.		
Iteración: 2da.	Prioridad: 4.	Complejidad: Alta.
Descripción: El administrador accede al sistema y selecciona la opción Gestionar Plantillas, el sistema muestra un listado con todas las plantillas. El administrador selecciona la opción Editar Plantilla y se muestra un formulario con los datos a modificar: <ul style="list-style-type: none">➤ Nombre.➤ Descripción.➤ Roles (Marcar los roles que tendrán los usuarios creados a partir de la plantilla).➤ Recursos (Marcar los recursos a los que tendrán acceso los usuarios creados a partir de la plantilla).➤ Preguntas de Seguridad (Marcar las preguntas de seguridad que se les mostrarán cuando se necesite recuperar la contraseña de algún usuario creado a partir de la plantilla).➤ Política de Contraseña (Marcar la política de contraseña para validar las contraseña de acceso al sistema). Luego el administrador selecciona la opción Siguiente, el sistema muestra una interfaz para gestionar los atributos de identidad necesarios para la plantilla. Posteriormente el administrador selecciona la opción Editar y el sistema guarda la plantilla.		
Validaciones: -		

Nombre del Escenario: Eliminar plantilla.		Identificador: Ap_Adm28.
Objetivo del Escenario: Permitir la eliminación de una plantilla para la solicitud de creación de cuentas de usuarios.		
Persona: Administrador.		
Iteración: 2da.	Prioridad: 4.	Complejidad: Alta.
Descripción: El administrador accede al sistema y selecciona la opción Gestionar Plantillas, el sistema muestra un listado con todas las plantillas. El administrador selecciona la plantilla que desea eliminar, la opción Eliminar Plantilla y el sistema elimina la plantilla.		
Validaciones: -		

Nombre del Escenario: Listar plantillas.		Identificador: Ap_Adm29.
Objetivo del Escenario: Permitir listar las plantillas para la solicitud de creación de cuentas de usuarios.		
Persona: Administrador.		
Iteración: 2da.	Prioridad: 4.	Complejidad: Alta.
Descripción: El administrador accede al sistema y selecciona la opción Gestionar Plantillas, el sistema muestra un listado con todas las		



*“Subsistema de Aprovisionamiento de Usuarios para el Sistema de
Administración de Identidades”*

plantillas.
Validaciones: -

Nombre del Escenario: Crear atributo de identidad.		Identificador: Ap_Adm30.
Objetivo del Escenario: Permitir la creación de un atributo de identidad.		
Persona: Administrador.		
Iteración: 2da.	Prioridad: 4.	Complejidad: Media.
Descripción: El administrador accede al sistema y selecciona la opción Gestionar Plantillas, el sistema muestra un listado con todas las plantillas. El administrador selecciona la opción Crear Plantilla o la opción Editar Plantilla, se introducen los datos necesarios, se selecciona la opción Siguiente, el sistema muestra un listado con los atributos de identidad, el administrador selecciona la opción Crear Atributo de Identidad y el sistema muestra un formulario con los datos requeridos: <ul style="list-style-type: none">➤ Nombre.➤ Descripción.➤ Validaciones (pueden ser validaciones clásicas o una expresión regular). Luego de introducidos los datos se selecciona la opción Crear. El sistema muestra los atributos correspondientes a los conectores utilizados por los recursos seleccionados en la plantilla. El administrador relaciona el atributo de identidad creado con los atributos de los conectores y selecciona la opción guardar. El sistema guarda el atributo de identidad incluyendo su relación con los atributos de los conectores.		
Validaciones: -		

Nombre del Escenario: Editar atributo de identidad.		Identificador: Ap_Adm31.
Objetivo del Escenario: Permitir la modificación de un atributo de identidad.		
Persona: Administrador.		
Iteración: 2da.	Prioridad: 4.	Complejidad: Media.
Descripción: El administrador accede al sistema y selecciona la opción Gestionar Plantillas, el sistema muestra un listado con todas las plantillas. El administrador selecciona la opción Editar Plantilla o Crear Plantilla, se introducen los datos necesarios, se selecciona la opción Siguiente, el sistema muestra un listado con los atributos de identidad, el administrador selecciona la opción Editar Atributo de Identidad y el sistema muestra un formulario con los datos a editar: <ul style="list-style-type: none">➤ Nombre.➤ Descripción.➤ Validaciones (pueden ser validaciones clásicas o una expresión regular). Luego de modificados los datos se selecciona la opción Editar. El sistema muestra los atributos correspondientes a los		



*“Subsistema de Aprovisionamiento de Usuarios para el Sistema de
Administración de Identidades”*

conectores utilizados por los recursos seleccionados en la plantilla. El administrador relaciona el atributo de identidad editado con los mismos atributos de los conectores o con otros y selecciona la opción guardar. El sistema guarda el atributo de identidad incluyendo su relación con los atributos de los conectores.

Validaciones: -

Nombre del Escenario: Eliminar atributo de identidad.		Identificador: Ap_Adm32.
Objetivo del Escenario: Permitir la eliminación de un atributo de identidad.		
Persona: Administrador.		
Iteración: 2da.	Prioridad: 4.	Complejidad: Media.
Descripción: El administrador accede al sistema y selecciona la opción Gestionar Plantillas, el sistema muestra un listado con todas las plantillas. El administrador selecciona la opción Editar Plantilla o Crear Plantilla, se introducen los datos necesarios, se selecciona la opción Siguiente, el sistema muestra un listado con los atributos de identidad, el administrador selecciona la opción Eliminar Atributo de Identidad perteneciente al atributo de identidad que desea eliminar. El sistema elimina el atributo de identidad y la relación con los atributos de los conectores.		
Validaciones: -		

Nombre del Escenario: Listar atributos de identidad.		Identificador: Ap_Adm33.
Objetivo del Escenario: Permitir listar los atributos de identidad.		
Persona: Administrador.		
Iteración: 2da.	Prioridad: 4.	Complejidad: Media.
Descripción: El administrador accede al sistema y selecciona la opción Gestionar Plantillas, el sistema muestra un listado con todas las plantillas. El administrador selecciona la opción Editar Plantilla o Crear Plantilla, se introducen los datos necesarios, se selecciona la opción Siguiente, el sistema muestra un listado con los atributos de identidad correspondientes a la plantilla.		
Validaciones: -		

Nombre del Escenario: Cambiar de estado de una cuenta de usuario.		Identificador: Ap_Adm34.
Objetivo del Escenario: Permitir el cambio del estado de un cuenta de usuario creada en el sistema.		
Persona: Administrador.		
Iteración: 3ra.	Prioridad: 3.	Complejidad: Alta.
Descripción: El administrador accede al sistema y selecciona la opción Administrar Cuentas, el sistema muestra en pantalla un listado de		



*“Subsistema de Aprovisionamiento de Usuarios para el Sistema de
Administración de Identidades”*

cuentas de usuarios y además se muestran los estados a los que pueden cambiar dichas cuentas, estos estados son:

- Activa.
- Suspendida.
- Bloqueada.
- Eliminada.

El administrador selecciona la cuenta a la que le desea cambiar el estado y selecciona el estado. En caso de que el estado seleccionado sea Activa, Bloqueada o Suspendida, se le cambia el estado a todas las cuentas existentes en los recursos y la existente en el sistema. Por otra parte en caso de que el estado sea Eliminada, se eliminan las cuentas existentes en todos los recursos donde existía pero en el sistema solo pasa al estado Eliminada.

Validaciones:

- Del estado “Activa” la cuenta puede pasar a cualquier estado.
- Del estado “Bloqueada” la cuenta solo puede pasar al estado “Activa” o “Eliminada”.
- Del estado “Suspendida” la cuenta solo puede pasar al estado “Activa”.
- Del estado “Eliminada” la cuenta no puede pasar a ningún otro estado.

Nombre del Escenario: Editar cuenta de usuario.		Identificador: Ap_Adm35.
Objetivo del Escenario: Permitir editar una cuenta de usuario creada en el sistema.		
Persona: Administrador.		
Iteración: 3ra.	Prioridad: 3.	Complejidad: Alta.
Descripción: El administrador accede al sistema y selecciona la opción Administrar Cuentas, el sistema muestra en pantalla un listado de cuentas de usuarios. El administrador selecciona la cuenta a la que desea editar y selecciona la opción Editar Cuenta y se muestra un formulario con los datos de la cuenta que pueden ser editados. Los campos dependen de la plantilla a partir de la cual la cuenta fue creada, es decir pueden variar.		
Validaciones: -		

Nombre del Escenario: Listar cuentas de usuario.		Identificador: Ap_Adm36.
Objetivo del Escenario: Permitir listar las cuentas de usuario creadas en el sistema.		
Persona: Administrador.		
Iteración: 3ra.	Prioridad: 3.	Complejidad: Alta.
Descripción: El administrador accede al sistema y selecciona la opción Administrar Cuentas, el sistema muestra en pantalla un listado de cuentas de usuarios.		
Validaciones: -		



*“Subsistema de Aprovisionamiento de Usuarios para el Sistema de
Administración de Identidades”*

Módulo Motor de Tareas:

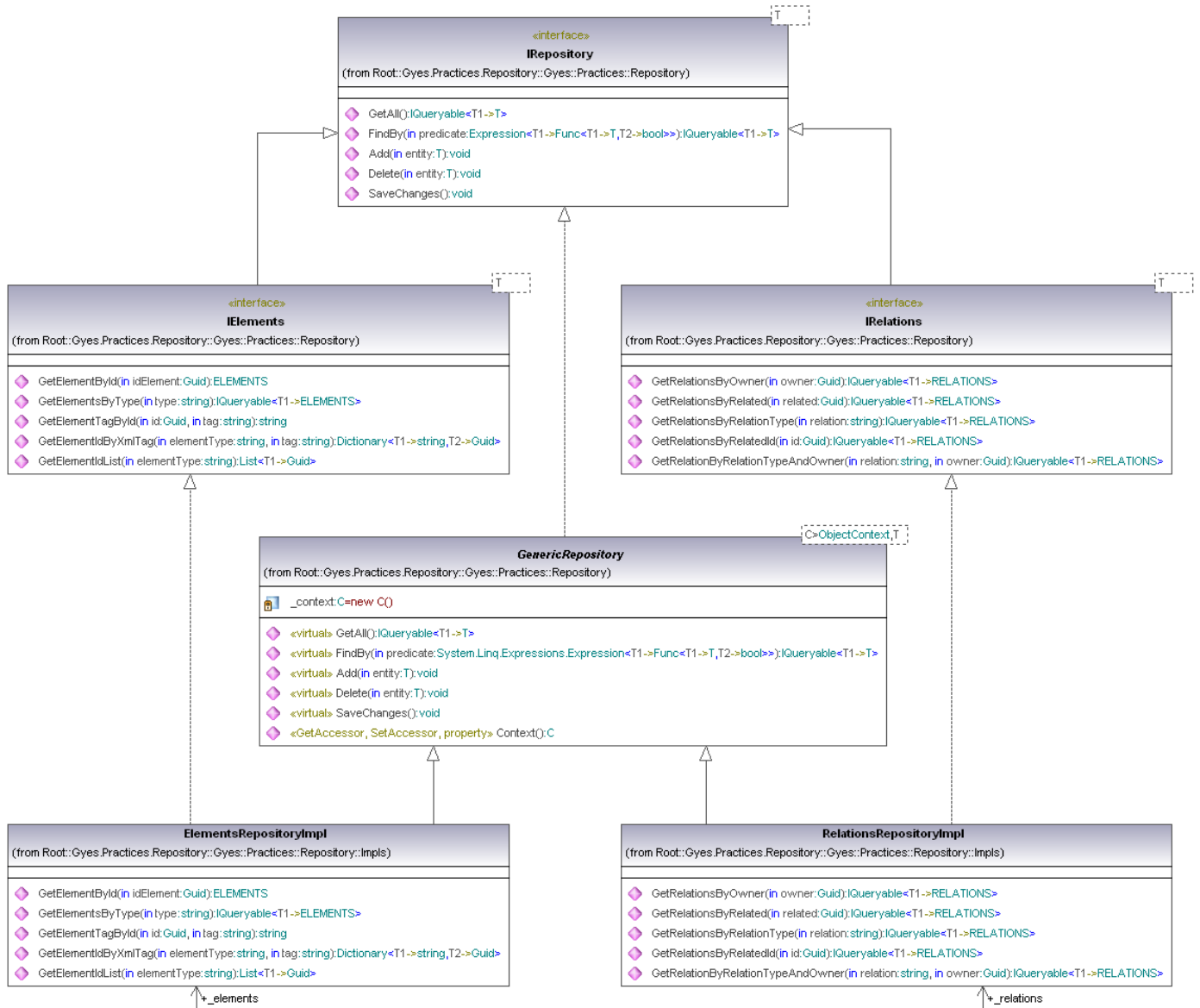
Nombre del Escenario: Bloquear cuentas por desuso.		Identificador: Ap_Mt37.
Objetivo del Escenario: Permitir al motor de tareas bloquear las cuentas que se encuentren en desuso en el sistema.		
Persona: Motor de tareas.		
Iteración: 3ra.	Prioridad: 3.	Complejidad: Baja.
Descripción: El sistema pide los datos necesarios para la configuración del análisis de bloqueo de cuentas por desuso. Los datos necesarios para esta acción son: <ul style="list-style-type: none">➤ Tipo de análisis (Manual o Automático).➤ Tiempo límite de desuso. En caso de que seleccione el tipo de análisis automático, se piden además al usuario los siguientes datos: <ul style="list-style-type: none">➤ Frecuencia (Diario o Semanal).➤ Hora de inicio del análisis. El administrador introduce los datos y selecciona la opción Guardar y el sistema guarda la configuración.		
Validaciones: -		

Nombre del Escenario: Eliminar cuentas bloqueadas.		Identificador: Ap_Mt38.
Objetivo del Escenario: Permitir al motor de tareas eliminar las cuentas bloqueadas existentes en el sistema.		
Persona: Motor de tareas.		
Iteración: 3ra.	Prioridad: 3.	Complejidad: Baja.
Descripción: El sistema pide los datos necesarios para la configuración del análisis de eliminación de cuentas bloqueadas. Los datos necesarios para esta acción son: <ul style="list-style-type: none">➤ Tipo de análisis (Manual o Automático).➤ Tiempo límite de estar bloqueadas. En caso de que seleccione el tipo de análisis automático, se piden además al usuario los siguientes datos: <ul style="list-style-type: none">➤ Frecuencia (Diario o Semanal).➤ Hora del inicio del análisis. El administrador introduce los datos y selecciona la opción Guardar y el sistema guarda la configuración.		
Validaciones: -		



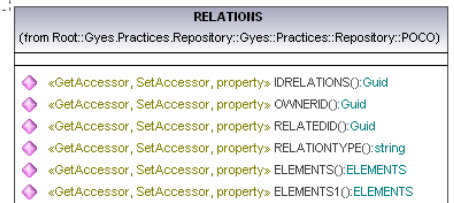
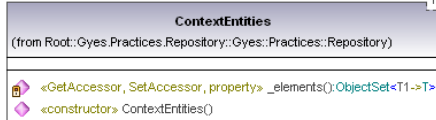
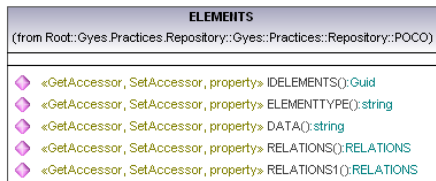
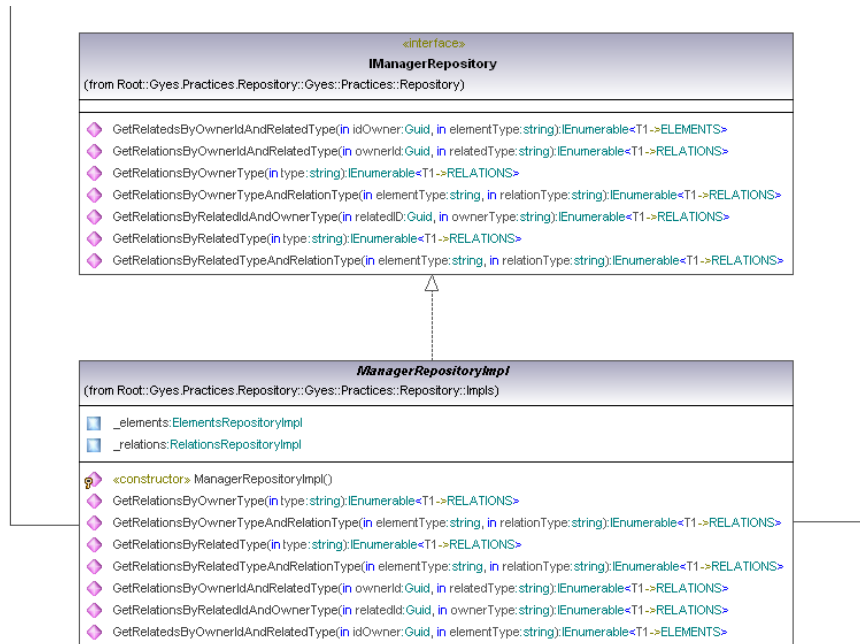
"Subsistema de Aprovisionamiento de Usuarios para el Sistema de Administración de Identidades"

Anexo VII: Diagrama de Clases Patrón Repositorio.





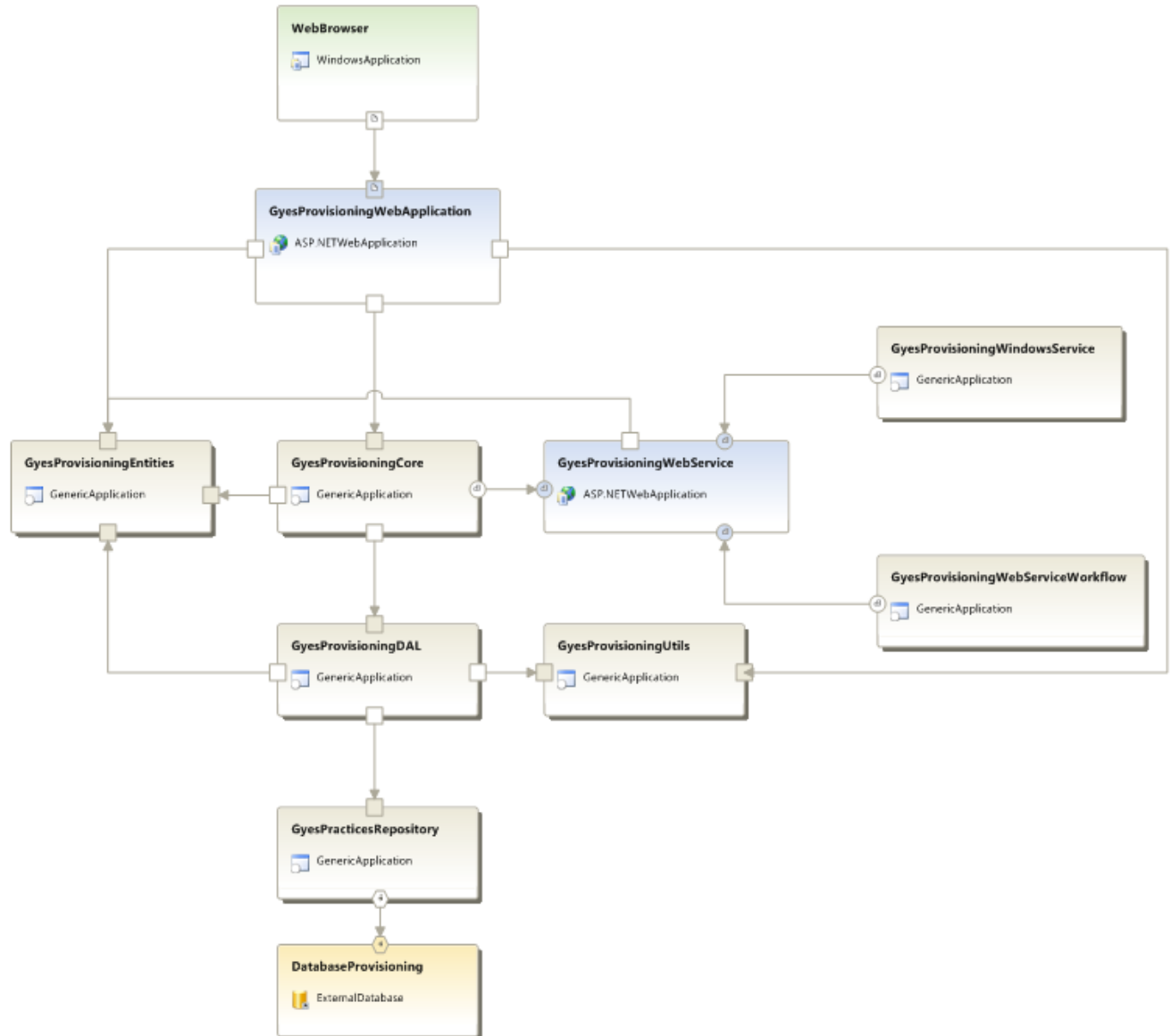
“Subsistema de Aprovisionamiento de Usuarios para el Sistema de Administración de Identidades”





“Subsistema de Aproveccionamiento de Usuarios para el Sistema de Administracion de Identidades”

Anexo VIII: Diagrama de Aplicacion del Subsistema de Aproveccionamiento de Usuarios.





“Subsistema de Aprovisionamiento de Usuarios para el Sistema de Administración de Identidades”

Anexo IX: Diagramas de Clases por Capas.

Presentación:

MailSender
(from Root::Gyes.Provisioning.WebApplication::Gyes::Provisioning::WebApplication::Code::Correo)
<ul style="list-style-type: none">_cliente: SmtplibClient_recuperarContraseña: stringasunto: string[*]
<ul style="list-style-type: none">«constructor» MailSender()«constructor» MailSender(in enableSSL: bool, in message: string)«GetAccessor, SetAccessor, property» RecuperarContraseña(): string«GetAccessor, SetAccessor, property» Asunto(): string[*]SendMail(in from: string, in to: string, in subject: string, in m: string): bool

MvcApplication
(from Root::Gyes.Provisioning.WebApplication::Gyes::Provisioning)
<ul style="list-style-type: none">RegisterGlobalFilters(in filters: GlobalFilterCollection): voidRegisterRoutes(in routes: RouteCollection): voidApplication_Start(): void

PasswordGenerator
(from Root::Gyes.Provisioning.WebApplication::Gyes::Provisioning::WebApplication::Code::Correo)
<ul style="list-style-type: none">porcentajeMayusculas: intporcentajeSimbolos: intporcentajeNumeros: intsemilla: Randomcaracteres: string="abcdefghijklmnopqrstuvwxyz"numeros: string="0123456789"simbolos: string="%\$#@+~=&"password: StringBuilder
<ul style="list-style-type: none">«GetAccessor, SetAccessor, property» LongitudPassword(): int«GetAccessor, SetAccessor, property» PorcentajeMayusculas(): int«GetAccessor, SetAccessor, property» PorcentajeSimbolos(): int«GetAccessor, SetAccessor, property» PorcentajeNumeros(): int«constructor» PasswordGenerator()«constructor» PasswordGenerator(in longitudCaracteres: int)«constructor» PasswordGenerator(in longitudCaracteres: int, in porcentajeMayusculas: int, in porcentajeSimbolos: int, in porcentajeNumeros: int)GetNewPassword(): stringSetCaracteresEspeciales(in numeroCaracteresMayuscula: int, in numeroCaracteresNumericos: int, in numeroCaracteresSimbolos: int): voidGetPassword(): stringGeneraPassword(): voidReemplazaCaracteresEspeciales(in posiciones: int[*], in posicionInicial: int, in posicionFinal: int, in tipoCaracter: TipoCaracterEnum): voidGetPosicionesCaracteresEspeciales(in numeroPosiciones: int): int[*]GetCaracterAleatorio(in tipoCaracter: TipoCaracterEnum): char
<ul style="list-style-type: none">TipoCaracterEnum


















“Subsistema de Aprovisionamiento de Usuarios para el Sistema de Administración de Identidades”


































AccountController
(from Root::Gyes.Provisioning.WebApplication::Gyes::Provisioning::WebApplication::Controllers)
<pre> I _accountManager:IAccountManager=new AccountManagerImpl() I _resourceManager:IResourceManager=new ResourceManagerImpl() I _securityQuestionManager:ISecurityQuestionManager=new SecurityQuestionManagerImpl() I _templateManager:ITemplateManager=new TemplateManagerImpl() I _answerManager:IAnswerManager=new AnswerManagerImpl() I _configurationManager:IConfigurationManager=new ConfigurationManagerImpl() serviceClient:ProvisioningServiceClient=new ProvisioningServiceClient("WSHttpBinding_IProvisioningService") I _accountGlobal:Account=new Account() I notifications:List<T1->string=new List<string>() </pre>
<pre> AddNotification(in notification: string): JsonResult GetNotification(): JsonResult CheckOldPassword(in _oldPassword: string, in _user: string): JsonResult CheckUser(in _user: string): JsonResult CheckPolicy(in _newPassword: string, in _user: string): JsonResult <<attributes>> GetAccountData(in id: string): JsonResult LoadAccount(in id: string): ActionResult <<attributes>> LoadAccount(in collection: FormCollection): ActionResult ChangePassword(): ActionResult <<attributes>> ChangePassword(in model: ChangePasswordModel): ActionResult ChangePasswordSuccess(): ActionResult Logout(): ActionResult <<attributes>> Logout(in model: AccountLockoutModel): ActionResult DeleteAccount(): ActionResult <<attributes>> DeleteAccount(in model: AccountLockoutModel): ActionResult StartLogout(): ActionResult StartLogoutNow(): ActionResult StartDelete(): ActionResult StartDeleteNow(): ActionResult LogoutDefaultValues(): ActionResult DeleteAccountDefaultValues(): ActionResult RecoverPassword(): ActionResult <<attributes>> RecoverPassword(in model: RecoverPasswordModel): ActionResult RecoverPasswordSecurityQuestions(in model: string): ActionResult <<attributes>> RecoverPasswordSecurityQuestions(in collection: FormCollection, in model: string): ActionResult Delete(in id: string): ActionResult Enable(in id: string): ActionResult Suspend(in id: string): ActionResult Block(in id: string): ActionResult List(in id: string): ActionResult FilterAccount(in searchingName: string, in searchingOperator: JqGridSearchOperators, in searchingValue: string): List<T1->Account> <<attributes>> GridAccount(in request: JqGridRequest): ActionResult </pre>

ConnectorsController
(from Root::Gyes.Provisioning.WebApplication::Gyes::Provisioning::WebApplication::Controllers)
<pre> I _connectorManager:IConnectorManager=new ConnectorManagerImpl() I notifications:List<T1->string=new List<string>() </pre>
<pre> AddNotification(in notification: string): JsonResult GetNotification(): JsonResult CheckConnectorName(in _name: string): JsonResult Index(): ActionResult <<attributes>> Index(in collection: FormCollection): ActionResult Create(): ActionResult <<attributes>> Create(in connectorModel: ConnectorModel, in file: HttpPostedFileBase): ActionResult Edit(in id: string): ActionResult <<attributes>> Edit(in connectorModel: ConnectorModel, in file: HttpPostedFileBase): ActionResult Delete(in id: string): ActionResult List(): ActionResult FilterConnector(in searchingName: string, in searchingOperator: JqGridSearchOperators, in searchingValue: string): List<T1->Connector> <<attributes>> GridConnector(in request: JqGridRequest): ActionResult </pre>



“Subsistema de Aprovisionamiento de Usuarios para el Sistema de Administración de Identidades”

PasswordPolicyController	
(from Root::Gyes.Provisioning.WebApplication::Gyes::Provisioning::WebApplication::Controllers)	
 <code>_passwordPolicyManager:IPasswordPolicyManager=new PasswordPolicyManagerImpl()</code>	
 <code>_templateManager:ITemplateManager=new TemplateManagerImpl()</code>	
 <code>_notifications:List<T1->string=new List<string>()</code>	
 <code>AddNotification(in notification:string):JsonResult</code>	
 <code>GetNotification():JsonResult</code>	
 <code>Compare(in _maxAge:int, in _minAge:int):JsonResult</code>	
 <code>Index():ActionResult</code>	
 <code>List():ActionResult</code>	
 <code>Create():ActionResult</code>	
 <code><<attributes>> Create(in passwordPolicyModel:PasswordPolicyModel, in collection:FormCollection):ActionResult</code>	
 <code>Edit(in id:string):ActionResult</code>	
 <code><<attributes>> Edit(in passwordPolicyModel:PasswordPolicyModel, in collection:FormCollection):ActionResult</code>	
 <code>Delete(in id:string):ActionResult</code>	
 <code>FilterPasswordPolicy(in searchingName:string, in searchingOperator:JqGridSearchOperators, in searchingValue:string):List<T1->PasswordPolicy></code>	
 <code><<attributes>> GridPasswordPolicy(in request:JqGridRequest):ActionResult</code>	

RequestController	
(from Root::Gyes.Provisioning.WebApplication::Gyes::Provisioning::WebApplication::Controllers)	
 <code>_requestManager:IRequestManager=new RequestManagerImpl()</code>	
 <code>_resourceManager:IResourceManager=new ResourceManagerImpl()</code>	
 <code>_accountManager:IAccountManager=new AccountManagerImpl()</code>	
 <code>_connectorManager:IContectorManager=new ConnectorManagerImpl()</code>	
 <code>_templateManager:ITemplateManager=new TemplateManagerImpl()</code>	
 <code>serviceClient:ProvisioningServiceClient=new ProvisioningServiceClient("WSHttpBinding_IProvisioningService")</code>	
 <code>_key:int=1</code>	
 <code>_requestingUser:string=""</code>	
 <code>_notifications:List<T1->string=new List<string>()</code>	
 <code>GotoNextPage(in page:string, in requestMode:Request):string</code>	
 <code>AddNotification(in notification:string):JsonResult</code>	
 <code>GetNotification():JsonResult</code>	
 <code>CheckUser(in Usuario:string):JsonResult</code>	
 <code>Index():ActionResult</code>	
 <code>LoadRequest(in id:string):ActionResult</code>	
 <code>Pending(in id:string):ActionResult</code>	
 <code><<attributes>> Pending(in collection:FormCollection):ActionResult</code>	
 <code>Nonconforming(in id:string):ActionResult</code>	
 <code><<attributes>> Nonconforming(in collection:FormCollection):ActionResult</code>	
 <code>Review(in id:string):ActionResult</code>	
 <code><<attributes>> Review(in collection:FormCollection):ActionResult</code>	
 <code>Reviewed(in id:string):ActionResult</code>	
 <code><<attributes>> Reviewed(in collection:FormCollection):ActionResult</code>	
 <code>Revoked():ActionResult</code>	
 <code>Approved():ActionResult</code>	
 <code>Delete(in id:string):ActionResult</code>	
 <code>Tracking():ActionResult</code>	
 <code><<attributes>> Tracking(in collection:FormCollection):ActionResult</code>	
 <code>List(in id:string):ActionResult</code>	
 <code>FilterRequest(in searchingName:string, in searchingOperator:JqGridSearchOperators, in searchingValue:string):List<T1->Request></code>	
 <code><<attributes>> GridRequest(in requestGrid:JqGridRequest):ActionResult</code>	
 <code>FilterTrackingRequest(in searchingName:string, in searchingOperator:JqGridSearchOperators, in searchingValue:string):List<T1->Request></code>	
 <code><<attributes>> GridTrackingRequest(in requestGrid:JqGridRequest):ActionResult</code>	



“Subsistema de Aprovisionamiento de Usuarios para el Sistema de Administración de Identidades”

```
ResourcesController  
(from Root:Gyes.Provisioning.WebApplication:Gyes:Provisioning.WebApplication:Controllers)  
  
_resourceManager:IResourceManager=new ResourceManagerImpl()  
_templateManager:ITemplateManager=new TemplateManagerImpl()  
_notifications:List<T1>->string=new List<string>()  
  
AddNotification(n notification:string):JsonResult  
GetNotification():JsonResult  
ClassData(in item:string):ViewResultBase  
JsonData(in resource:string):JsonResult  
Index():ActionResult  
List():ActionResult  
Create():ActionResult  
<attributes> Create(in collection:FormCollection):ActionResult  
Edit(in id:string):ActionResult  
<attributes> Edit(in id:string, in collection:FormCollection):ActionResult  
Delete(in id:string):ActionResult  
EnableDisable(in id:string):ActionResult  
FilterResource(in searchingName:string, in searchingOperator:JqGridSearchOperators, in searchingValue:string):List<T1->Resource>  
<attributes> GridResource(in request:JqGridRequest):ActionResult
```

```
SecurityQuestionController  
(from Root:Gyes.Provisioning.WebApplication:Gyes:Provisioning.WebApplication:Controllers)  
  
_securityQuestionManager:ISecurityQuestionManager=new SecurityQuestionManagerImpl()  
_templateManager:ITemplateManager=new TemplateManagerImpl()  
_notifications:List<T1>->string=new List<string>()  
  
AddNotification(n notification:string):JsonResult  
GetNotification():JsonResult  
SecurityQuestionsList():ActionResult  
CreateSecurityQuestion():ActionResult  
<attributes> CreateSecurityQuestion(in model:SecurityQuestionModel):ActionResult  
EditSecurityQuestion(in id:string):ActionResult  
<attributes> EditSecurityQuestion(in id:string, in model:SecurityQuestionModel):ActionResult  
DeleteSecurityQuestion(in id:string):ActionResult  
FilterSecurityQ(in searchingName:string, in searchingOperator:JqGridSearchOperators, in searchingValue:string):List<T1->SecurityQuestion>  
<attributes> GridSecurityQ(in request:JqGridRequest):ActionResult
```

```
TemplateController  
(from Root:Gyes.Provisioning.WebApplication:Gyes:Provisioning.WebApplication:Controllers)  
  
_requestModel:RequestModel=new RequestModel()  
_idTemplateGlobal:string  
_key:int=1  
_auxiliar:string=""  
_oldName:string=""  
_templateAuxiliar:Template  
_notifications:List<T1>->string=new List<string>()  
_templateManager:ITemplateManager=new TemplateManagerImpl()  
_resourceManager:IResourceManager=new ResourceManagerImpl()  
_securityQuestionManager:ISecurityQuestionManager=new SecurityQuestionManagerImpl()  
_passwordPolicyManager:IPasswordPolicyManager=new PasswordPolicyManagerImpl()  
_identityAttributeManager:IdentityAttributeManager=new IdentityAttributeManagerImpl()  
_connectorManager:IConnectorManager=new ConnectorManagerImpl()  
  
GoToNextPage(in page:string):string  
LoadTemplate(in idTemplate:string):ActionResult  
<attributes> LoadTemplate(in collection:FormCollection):ActionResult  
AddNotification(n notification:string):JsonResult  
GetNotification():JsonResult  
CheckTemplateName(in _templateName:string):JsonResult  
CheckAttrName(in attrName:string):JsonResult  
<attributes> GetAttribute(in attribute:JsonObjectAttribute):JsonResult  
<attributes> AddAttribute(in attribute:JsonObjectAttribute):ActionResult  
LoadPage():ActionResult  
<attributes> GetRelatedAttributes(in name:string):ActionResult  
Index():ActionResult  
<attributes> ModifyAttribute(in attribute:JsonObjectAttribute):JsonResult  
Create():ActionResult  
<attributes> Create(in template:TemplateModel, in collection:FormCollection):ActionResult  
Create1(in formCollection:FormCollection):ActionResult  
Guardar():ActionResult  
Edit(in id:string):ActionResult  
<attributes> Edit(in template:TemplateModel, in collection:FormCollection):ActionResult  
Edit1(in formCollection:FormCollection):ActionResult  
Save1():ActionResult  
DeleteAttrCreate(in id:string):ActionResult  
DeleteAttrEdit(in id:string):ActionResult  
Delete(in id:string):ActionResult  
List():ActionResult  
FilterTemplate(in searchingName:string, in searchingOperator:JqGridSearchOperators, in searchingValue:string):List<T1->Template>  
<attributes> GridTemplate(in request:JqGridRequest):ActionResult
```



“Subsistema de Aprovisionamiento de Usuarios para el Sistema de Administración de Identidades”

AccountLockoutModel
(from Root::Gyes.Provisioning.WebApplication::Gyes::Provisioning::WebApplication::Models)
<ul style="list-style-type: none"> ◆ «constructor» AccountLockoutModel() ◆ «GetAccessor, SetAccessor, attributes, property» _automatic():bool ◆ «GetAccessor, SetAccessor, attributes, property» _weekly():bool ◆ «GetAccessor, SetAccessor, attributes, property» _weekDay():string ◆ «GetAccessor, SetAccessor, attributes, property» _daily():bool ◆ «GetAccessor, SetAccessor, attributes, property» _exactTime():string ◆ «GetAccessor, SetAccessor, attributes, property» _limitTime():int ◆ «GetAccessor, SetAccessor, attributes, property» _limitTimeManual():int

ConnectorModel
(from Root::Gyes.Provisioning.WebApplication::Gyes::Provisioning::WebApplication::Models)
<ul style="list-style-type: none"> ◆ «GetAccessor, SetAccessor, attributes, property» _idConnector():Guid ◆ «GetAccessor, SetAccessor, attributes, property» _name():string ◆ «GetAccessor, SetAccessor, attributes, property» _description():string

RecoverPasswordModel
(from Root::Gyes.Provisioning.WebApplication::Gyes::Provisioning::WebApplication::Models)
<ul style="list-style-type: none"> ◆ «GetAccessor, SetAccessor, attributes, property» _user():string

ResourceModel
(from Root::Gyes.Provisioning.WebApplication::Gyes::Provisioning::WebApplication::Models)
<ul style="list-style-type: none"> ◆ «GetAccessor, SetAccessor, attributes, property» _name():string ◆ «GetAccessor, SetAccessor, attributes, property» _ip():string ◆ «GetAccessor, SetAccessor, attributes, property» _user():string ◆ «GetAccessor, SetAccessor, attributes, property» _password():string ◆ «GetAccessor, SetAccessor, attributes, property» _port():int ◆ «GetAccessor, SetAccessor, attributes, property» _otrosCampos():Dictionary<T1->object,T2->object» ◆ «constructor» ResourceModel()

SecurityQuestionModel
(from Root::Gyes.Provisioning.WebApplication::Gyes::Provisioning::WebApplication::Models)
<ul style="list-style-type: none"> ◆ «GetAccessor, SetAccessor, attributes, property» id():Guid ◆ «GetAccessor, SetAccessor, attributes, property» _securityQuestion():string ◆ «GetAccessor, SetAccessor, attributes, property» _description():string

TemplatesRepository
(from Root::Gyes.Provisioning.WebApplication::Gyes::Provisioning::WebApplication::Models)
<ul style="list-style-type: none"> ◆ «GetAccessor, SetAccessor, property» template():TemplateModel ◆ «GetAccessor, SetAccessor, property» TemplateReturn():TemplateModel

ChangePasswordModel
(from Root::Gyes.Provisioning.WebApplication::Gyes::Provisioning::WebApplication::Models)
<ul style="list-style-type: none"> ◆ «GetAccessor, SetAccessor, attributes, property» _user():string ◆ «GetAccessor, SetAccessor, attributes, property» _oldPassword():string ◆ «GetAccessor, SetAccessor, attributes, property» _newPassword():string ◆ «GetAccessor, SetAccessor, attributes, property» _confirmPassword():string

PasswordPolicyModel
(from Root::Gyes.Provisioning.WebApplication::Gyes::Provisioning::WebApplication::Models)
<ul style="list-style-type: none"> ◆ «GetAccessor, SetAccessor, attributes, property» _idPolicy():Guid ◆ «GetAccessor, SetAccessor, attributes, property» _name():string ◆ «GetAccessor, SetAccessor, attributes, property» _maxPasswordRecord():int ◆ «GetAccessor, SetAccessor, attributes, property» _description():string ◆ «GetAccessor, SetAccessor, attributes, property» _minAge():int ◆ «GetAccessor, SetAccessor, attributes, property» _maxAge():int ◆ «GetAccessor, SetAccessor, attributes, property» _expirationWarning():int ◆ «GetAccessor, SetAccessor, attributes, property» _policy():string

RequestModel
(from Root::Gyes.Provisioning.WebApplication::Gyes::Provisioning::WebApplication::Models)
<ul style="list-style-type: none"> ◆ «GetAccessor, SetAccessor, attributes, property» _idRequest():Guid ◆ «GetAccessor, SetAccessor, attributes, property» _state():RequestState ◆ «GetAccessor, SetAccessor, property» _requestingUser():string ◆ «GetAccessor, SetAccessor, property» _accountType():string ◆ «GetAccessor, SetAccessor, property» _users():Dictionary<T1->string,T2->bool» ◆ «GetAccessor, SetAccessor, property» _data():Dictionary<T1->string,T2->object» ◆ «GetAccessor, SetAccessor, property» _attributes():List<T1->IdentityAttribute» ◆ «GetAccessor, SetAccessor, property» _nonConformities():List<T1->NonConformity» ◆ «GetAccessor, SetAccessor, property» _roleList():List<T1->Role» ◆ «GetAccessor, SetAccessor, property» _resourceList():List<T1->Resource» ◆ «GetAccessor, SetAccessor, property» _securityQuestions():List<T1->SecurityQuestion» ◆ «GetAccessor, SetAccessor, property» _passwordPolicy():PasswordPolicy ◆ «constructor» RequestModel()

TemplateModel
(from Root::Gyes.Provisioning.WebApplication::Gyes::Provisioning::WebApplication::Models)
<ul style="list-style-type: none"> ◆ «GetAccessor, SetAccessor, attributes, property» _templateName():string ◆ «GetAccessor, SetAccessor, attributes, property» _templateDescription():string ◆ «GetAccessor, SetAccessor, property» _identityAttributes():List<T1->IdentityAttribute» ◆ «GetAccessor, SetAccessor, property» _resourcesList():List<T1->Resource» ◆ «GetAccessor, SetAccessor, property» _securityQuestionsList():List<T1->SecurityQuestion» ◆ «GetAccessor, SetAccessor, property» _passwordPolicyList():List<T1->PasswordPolicy» ◆ «GetAccessor, SetAccessor, attributes, property» _role1():bool ◆ «GetAccessor, SetAccessor, attributes, property» _role2():bool ◆ «GetAccessor, SetAccessor, property» error():bool ◆ «GetAccessor, SetAccessor, property» _descriptionRoles():Dictionary<T1->string,T2->string» ◆ «constructor» TemplateModel()



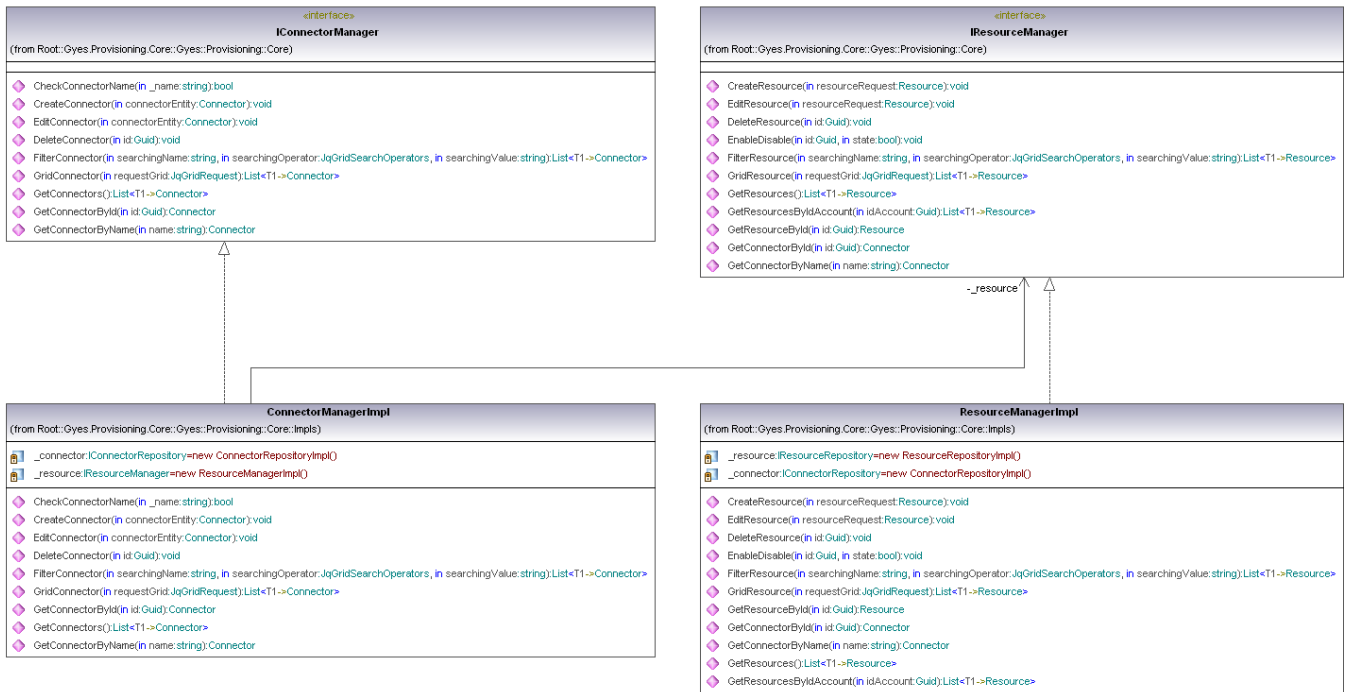
“Subsistema de Aprovisionamiento de Usuarios para el Sistema de Administración de Identidades”

Negocio:



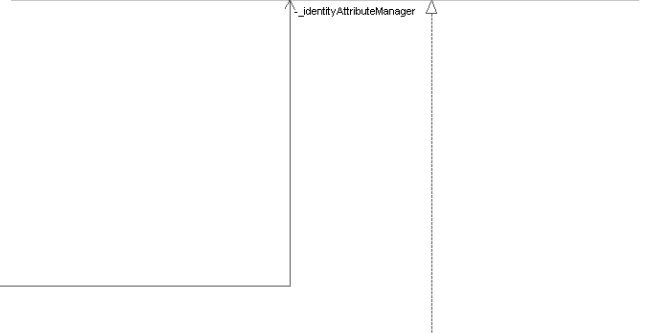
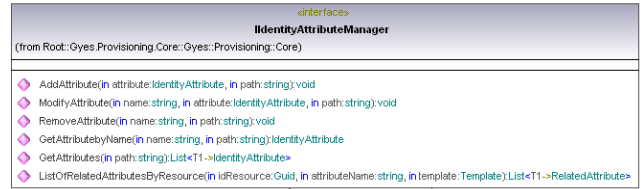
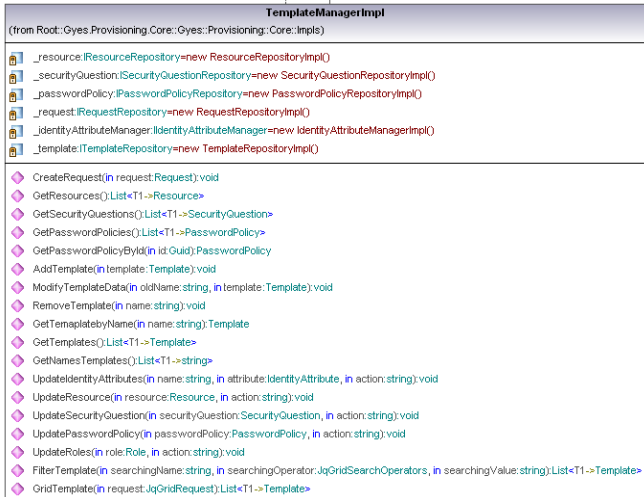


“Subsistema de Aprovisionamiento de Usuarios para el Sistema de Administración de Identidades”



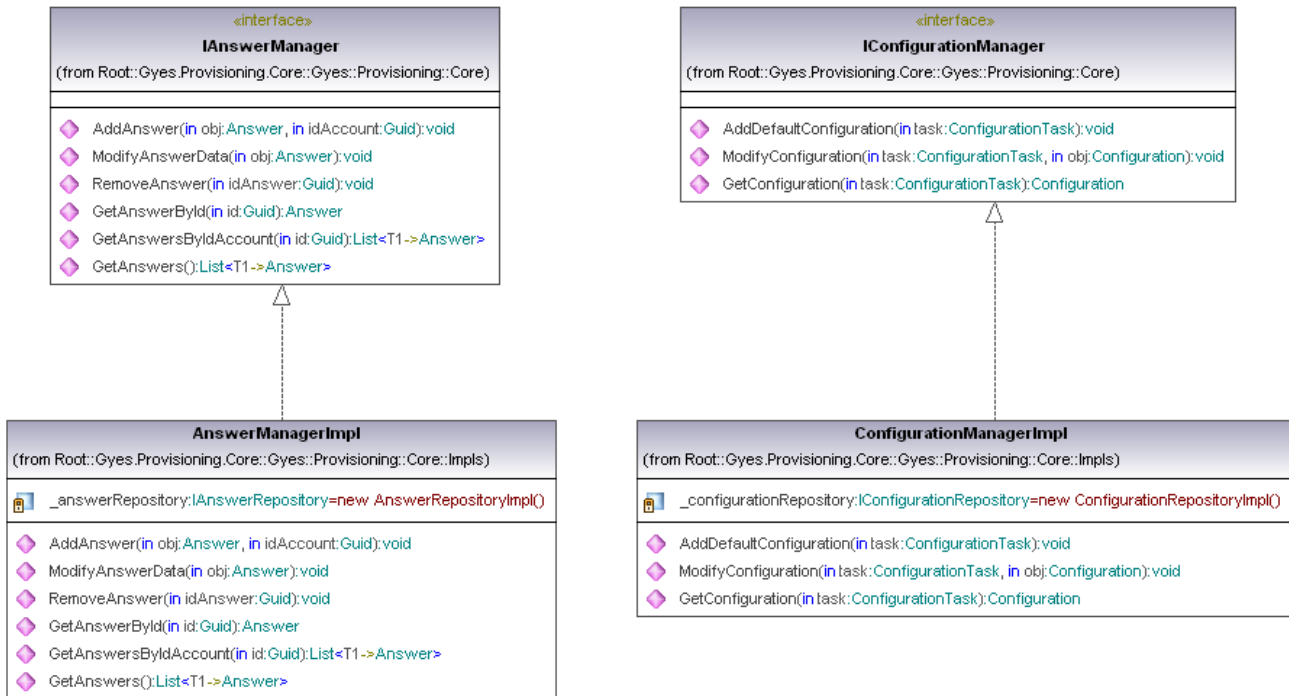


“Subsistema de Aprovisionamiento de Usuarios para el Sistema de Administración de Identidades”





“Subsistema de Aprovisionamiento de Usuarios para el Sistema de Administración de Identidades”

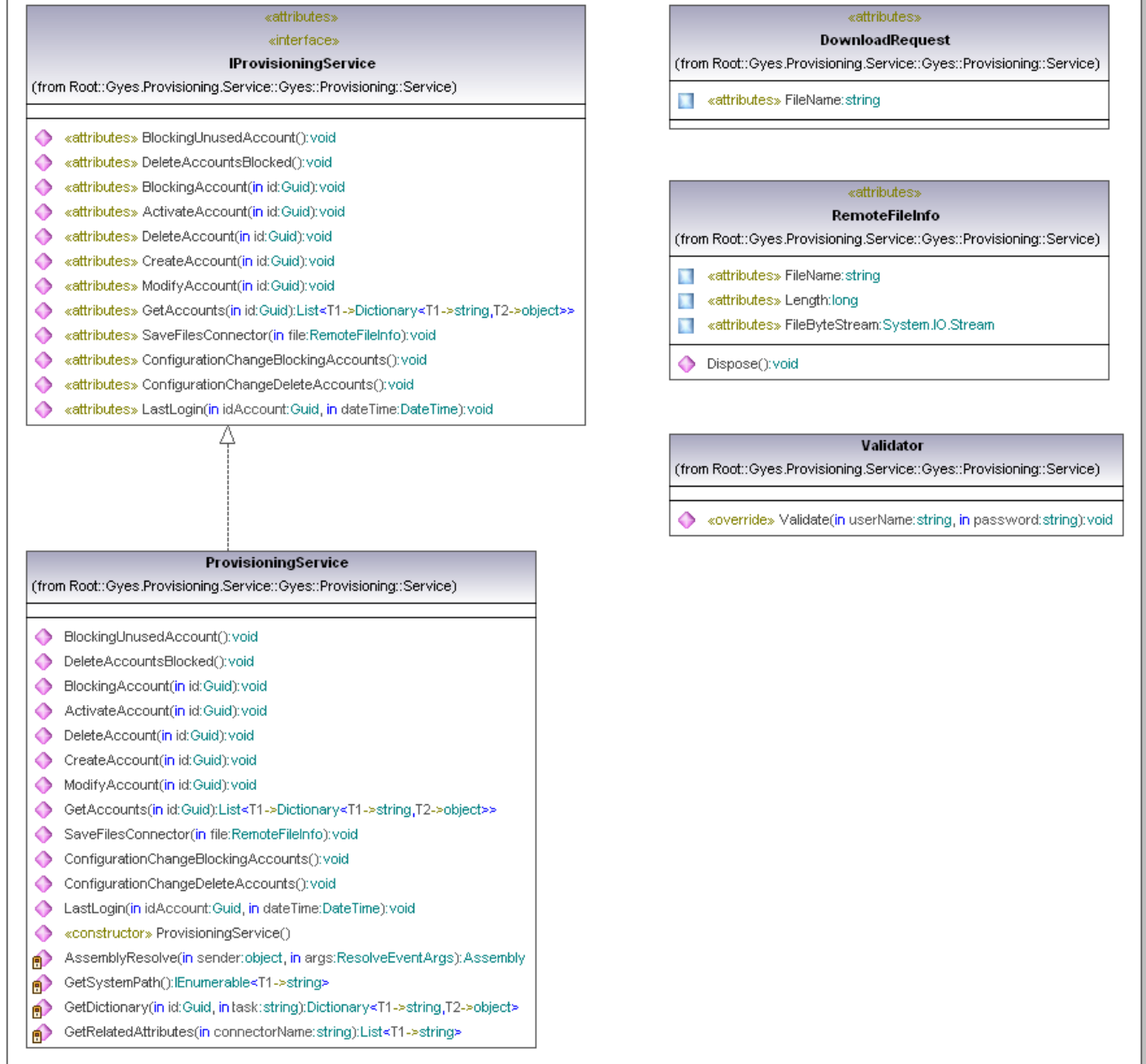




“Subsistema de Aprovechamiento de Usuarios para el Sistema de Administración de Identidades”

Servicios:

pkg Component View





















Generated by UModel










www.altova.com



“Subsistema de Aprovisionamiento de Usuarios para el Sistema de Administración de Identidades”










ActivateAccountActivity	
(from Root::Gyes.Provisioning.WebService.Workflow::Gyes::Provisioning::WebService::Workflow::Activities)	
 <code>_objectConnector:InArgument<T1->IConnectorFunctionality></code>	
 <code>_objectType:InArgument<T1->Type></code>	
 <code>_data:InArgument<T1->Dictionary<T1->string,T2->object>></code>	
 <code>_resultTask:OutArgument<T1->bool></code>	
 <code>«override» Execute(in context:NativeActivityContext):void</code>	
 <code>«GetAccessor, SetAccessor, property» ObjectConnector():InArgument<T1->IConnectorFunctionality></code>	
 <code>«GetAccessor, SetAccessor, property» ObjectType():InArgument<T1->Type></code>	
 <code>«GetAccessor, SetAccessor, property» Data():InArgument<T1->Dictionary<T1->string,T2->object>></code>	
 <code>«GetAccessor, SetAccessor, property» ResultTask():OutArgument<T1->bool></code>	










BlockAccountActivity	
(from Root::Gyes.Provisioning.WebService.Workflow::Gyes::Provisioning::WebService::Workflow::Activities)	
 <code>_objectConnector:InArgument<T1->IConnectorFunctionality></code>	
 <code>_objectType:InArgument<T1->Type></code>	
 <code>_data:InArgument<T1->Dictionary<T1->string,T2->object>></code>	
 <code>_resultTask:OutArgument<T1->bool></code>	
 <code>«override» Execute(in context:NativeActivityContext):void</code>	
 <code>«GetAccessor, SetAccessor, property» ObjectConnector():InArgument<T1->IConnectorFunctionality></code>	
 <code>«GetAccessor, SetAccessor, property» ObjectType():InArgument<T1->Type></code>	
 <code>«GetAccessor, SetAccessor, property» Data():InArgument<T1->Dictionary<T1->string,T2->object>></code>	
 <code>«GetAccessor, SetAccessor, property» ResultTask():OutArgument<T1->bool></code>	




















CreateAccountActivity	
(from Root::Gyes.Provisioning.WebService.Workflow::Gyes::Provisioning::WebService::Workflow::Activities)	
 <code>_objectConnector:InArgument<T1->IConnectorFunctionality></code>	
 <code>_objectType:InArgument<T1->Type></code>	
 <code>_data:InArgument<T1->Dictionary<T1->string,T2->object>></code>	
 <code>_resultTask:OutArgument<T1->bool></code>	
 <code>«override» Execute(in context:NativeActivityContext):void</code>	
 <code>«GetAccessor, SetAccessor, property» ObjectConnector():InArgument<T1->IConnectorFunctionality></code>	
 <code>«GetAccessor, SetAccessor, property» ObjectType():InArgument<T1->Type></code>	
 <code>«GetAccessor, SetAccessor, property» Data():InArgument<T1->Dictionary<T1->string,T2->object>></code>	
 <code>«GetAccessor, SetAccessor, property» ResultTask():OutArgument<T1->bool></code>	



“Subsistema de Aprovisionamiento de Usuarios para el Sistema de Administración de Identidades”

DeleteAccountActivity	
(from Root::Gyes.Provisioning.WebService.Workflow::Gyes::Provisioning::WebService::Workflow::Activities)	
 <code>_objectConnector:InArgument<T1->IConnectorFunctionality></code>	
 <code>_objectType:InArgument<T1->Type></code>	
 <code>_data:InArgument<T1->Dictionary<T1->string,T2->object>></code>	
 <code>_resultTask:OutArgument<T1->bool></code>	
 <code>«override» Execute(in context:NativeActivityContext):void</code>	
 <code>«GetAccessor, SetAccessor, property» ObjectConnector():InArgument<T1->IConnectorFunctionality></code>	
 <code>«GetAccessor, SetAccessor, property» ObjectType():InArgument<T1->Type></code>	
 <code>«GetAccessor, SetAccessor, property» Data():InArgument<T1->Dictionary<T1->string,T2->object>></code>	
 <code>«GetAccessor, SetAccessor, property» ResultTask():OutArgument<T1->bool></code>	

GetObjectConnector	
(from Root::Gyes.Provisioning.WebService.Workflow::Gyes::Provisioning::WebService::Workflow::Activities)	
 <code>_className:InArgument<T1->string></code>	
 <code>_url:InArgument<T1->string></code>	
 <code>_objectConnector:OutArgument<T1->IConnectorFunctionality></code>	
 <code>_objectType:OutArgument<T1->Type></code>	
 <code>«override» Execute(in context:NativeActivityContext):void</code>	
 <code>«GetAccessor, SetAccessor, property» ClassName():InArgument<T1->string></code>	
 <code>«GetAccessor, SetAccessor, property» Url():InArgument<T1->string></code>	
 <code>«GetAccessor, SetAccessor, property» ObjectConnector():OutArgument<T1->IConnectorFunctionality></code>	
 <code>«GetAccessor, SetAccessor, property» ObjectType():OutArgument<T1->Type></code>	

LoadConfiguration	
(from Root::Gyes.Provisioning.WebService.Workflow::Gyes::Provisioning::WebService::Workflow::Activities)	
 <code>_urlFileConfiguration:InArgument<T1->string></code>	
 <code>_connectorName:InArgument<T1->string></code>	
 <code>_className:OutArgument<T1->string></code>	
 <code>_url:OutArgument<T1->string></code>	
 <code>_firstAttempt:OutArgument<T1->string></code>	
 <code>_secondAttempt:OutArgument<T1->string></code>	
 <code>_thirdAttempt:OutArgument<T1->string></code>	
 <code>_moreAttempt:OutArgument<T1->string></code>	
 <code>_maxAttempt:OutArgument<T1->int></code>	
 <code>«override» Execute(in context:NativeActivityContext):void</code>	
 <code>«GetAccessor, SetAccessor, property» UrlFileConfiguration():InArgument<T1->string></code>	
 <code>«GetAccessor, SetAccessor, property» ConnectorName():InArgument<T1->string></code>	
 <code>«GetAccessor, SetAccessor, property» ClassName():OutArgument<T1->string></code>	
 <code>«GetAccessor, SetAccessor, property» Url():OutArgument<T1->string></code>	
 <code>«GetAccessor, SetAccessor, property» FirstAttempt():OutArgument<T1->string></code>	
 <code>«GetAccessor, SetAccessor, property» SecondAttempt():OutArgument<T1->string></code>	
 <code>«GetAccessor, SetAccessor, property» ThirdAttempt():OutArgument<T1->string></code>	
 <code>«GetAccessor, SetAccessor, property» MoreAttempt():OutArgument<T1->string></code>	
 <code>«GetAccessor, SetAccessor, property» MaxAttempt():OutArgument<T1->int></code>	



“Subsistema de Aprovisionamiento de Usuarios para el Sistema de Administración de Identidades”

ModifyAccountActivity
(from Root::Gyes.Provisioning.WebService.Workflow::Gyes::Provisioning::WebService::Workflow::Activities)
<ul style="list-style-type: none"> _objectConnector:InArgument<T1->IConnectorFunctionality> _objectType:InArgument<T1->Type> _data:InArgument<T1->Dictionary<T1->string,T2->object>> _resultTask:OutArgument<T1->bool>
<ul style="list-style-type: none"> «override» Execute(in context:NativeActivityContext):void «GetAccessor, SetAccessor, property» ObjectConnector():InArgument<T1->IConnectorFunctionality> «GetAccessor, SetAccessor, property» ObjectType():InArgument<T1->Type> «GetAccessor, SetAccessor, property» Data():InArgument<T1->Dictionary<T1->string,T2->object>> «GetAccessor, SetAccessor, property» ResultTask():OutArgument<T1->bool>

RollbackActivity
(from Root::Gyes.Provisioning.WebService.Workflow::Gyes::Provisioning::WebService::Workflow::Activities)
<ul style="list-style-type: none"> _objectConnector:InArgument<T1->IConnectorFunctionality> _objectType:InArgument<T1->Type>
<ul style="list-style-type: none"> «override» Execute(in context:NativeActivityContext):void «GetAccessor, SetAccessor, property» ObjectConnector():InArgument<T1->IConnectorFunctionality> «GetAccessor, SetAccessor, property» ObjectType():InArgument<T1->Type>

ContainerObject
(from Root::Gyes.Provisioning.WebService.Workflow::Gyes::Provisioning::WebService::Workflow)
<ul style="list-style-type: none"> «GetAccessor, SetAccessor, property» _objectConnector():IConnectorFunctionality «GetAccessor, SetAccessor, property» _objectType():Type



“Subsistema de Aprovisionamiento de Usuarios para el Sistema de
Administración de Identidades”

pkg Component View

<<partial>>
AccountStatusCheckerSvc
(from Root::Gyes.Provisioning.WindowsService::Gyes::Provisioning::WindowsService)

- CheckAnalysis_BlockingAccounts:Timer
- DailyAnalysis_BlockingAccounts:Timer
- WeeklyAnalysis_BlockingAccounts:Timer
- automaticAnalysis_BlockingAccounts:bool
- dailyAnalysis_BlockingAccounts:bool
- day_BlockingAccounts:string
- timeStartAnalysis_BlockingAccounts:TimeSpan
- CheckAnalysis_DeleteAccounts:Timer
- DailyAnalysis_DeleteAccounts:Timer
- WeeklyAnalysis_DeleteAccounts:Timer
- automaticAnalysis_DeleteAccounts:bool
- dailyAnalysis_DeleteAccounts:bool
- day_DeleteAccounts:string
- timeStartAnalysis_DeleteAccounts:TimeSpan
- components:System.ComponentModel.IContainer=null
- EventLogService:System.Diagnostics.EventLog

<<constructor>> AccountStatusCheckerSvc()
CheckAnalysis_BlockingAccounts_Elapsed(in sender:object, in e:ElapsedEventArgs):void
CheckAnalysis_DeleteAccounts_Elapsed(in sender:object, in e:ElapsedEventArgs):void
DailyAnalysis_BlockingAccounts_Elapsed(in sender:object, in e:ElapsedEventArgs):void
DailyAnalysis_DeleteAccounts_Elapsed(in sender:object, in e:ElapsedEventArgs):void
WeeklyAnalysis_BlockingAccounts_Elapsed(in sender:object, in e:ElapsedEventArgs):void
WeeklyAnalysis_DeleteAccounts_Elapsed(in sender:object, in e:ElapsedEventArgs):void
<<override>> OnStart(in args:string[*]):void
<<override>> OnStop():void
<<override>> OnCustomCommand(in command:int):void
VerifySettingsBlockingAccounts():void
VerifySettingsDeleteAccounts():void
GetConfiguration_BlockingAccounts():void
GetConfiguration_DeleteAccounts():void
GetDayOfWeek_BlockingAccounts():int
GetDayOfWeek_DeleteAccounts():int
StartAnalysis_BlockingAccounts():void
StartAnalysis_DeleteAccounts():void
<<override>> Dispose(in disposing:bool):void
InitializeComponent():void

<<static>>
Program
(from Root::Gyes.Provisioning.WindowsService::Gyes::Provisioning::WindowsService)

- Main():void

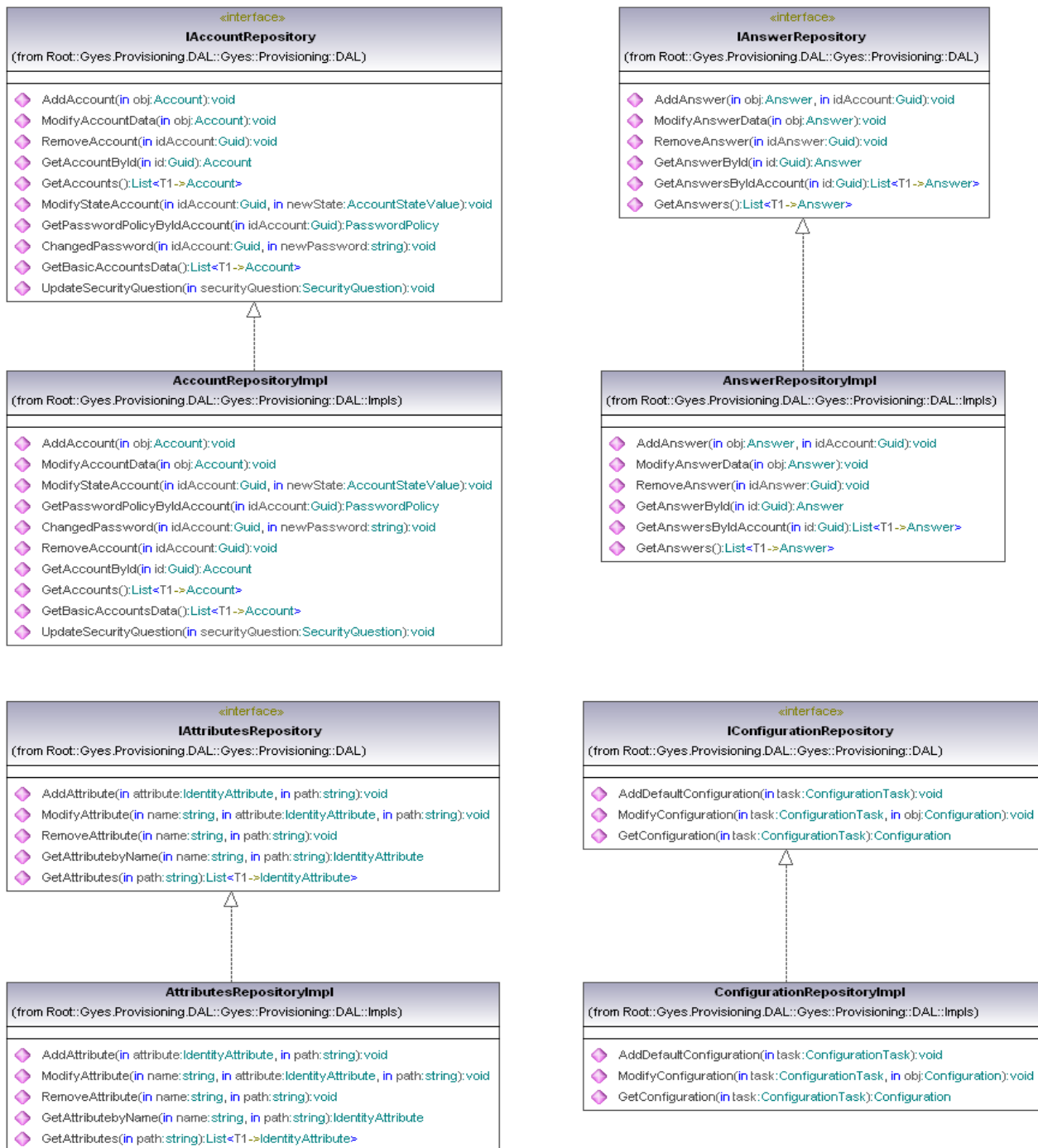
Generated by UModel

www.altova.com



“Subsistema de Aprovisionamiento de Usuarios para el Sistema de Administración de Identidades”

Acceso a Datos:



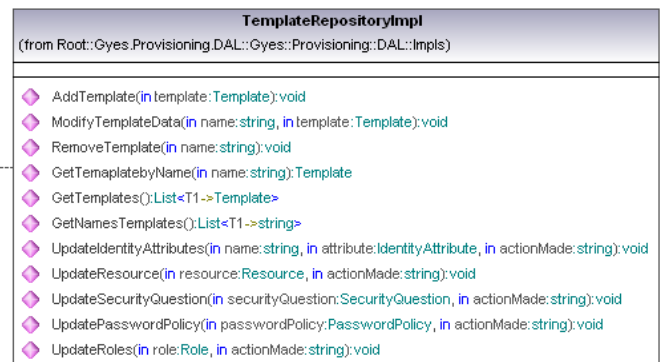
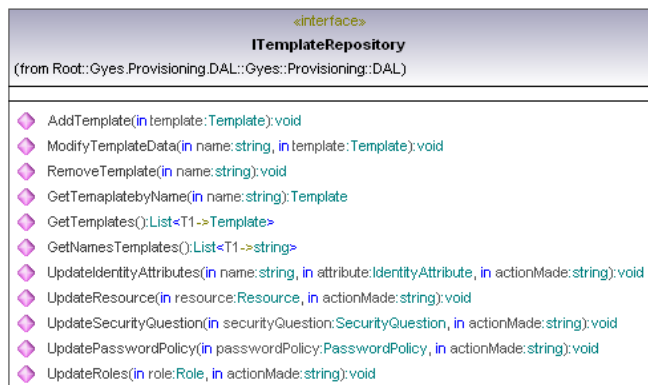
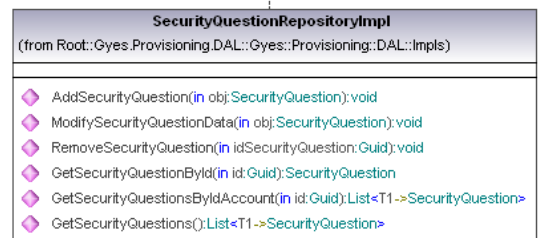
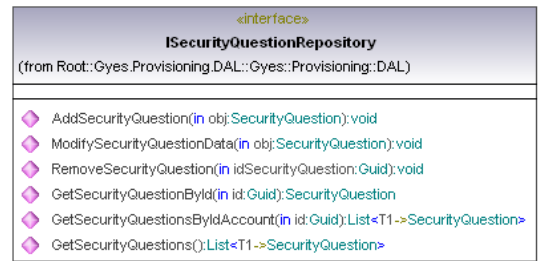
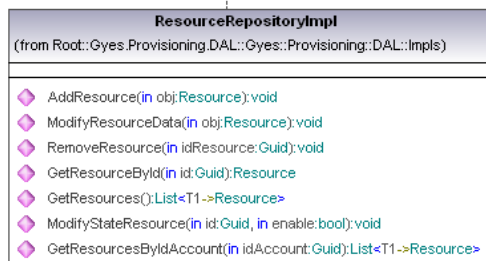
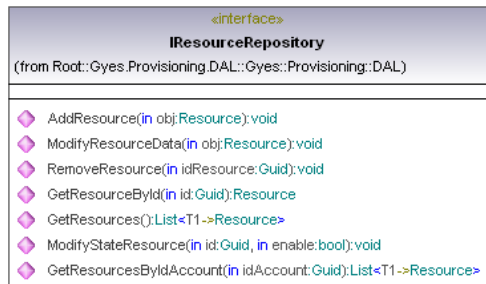


“Subsistema de Aprovisionamiento de Usuarios para el Sistema de Administración de Identidades”





“Subsistema de Aprovisionamiento de Usuarios para el Sistema de Administración de Identidades”





“Subsistema de Aprovisionamiento de Usuarios para el Sistema de Administración de Identidades”

Entidades:

«attributes» Account
(from Root::Gyes.Provisioning.Entities::Gyes::Provisioning::Entities)
<ul style="list-style-type: none"> «attributes»_idAccount:Guid _typeAccount:string _userName:string _password:string _state:State _lastLogin:DateTime _identityAttribute:AccountIdentityAttribute[*]
<ul style="list-style-type: none"> «GetAccessor, SetAccessor, attributes, property»_passwordPolicy():Guid «GetAccessor, SetAccessor, attributes, property»_passwordHistories():Guid «GetAccessor, SetAccessor, attributes, property»_answer():List<T1->Answer> «GetAccessor, SetAccessor, attributes, property»_resources():List<T1->Guid> «GetAccessor, SetAccessor, attributes, property»Id():string «GetAccessor, SetAccessor, attributes, property»TypeAccount():string «GetAccessor, SetAccessor, property»UserName():string «GetAccessor, SetAccessor, property»Password():string «GetAccessor, SetAccessor, property»State():State «GetAccessor, SetAccessor, property»LastLogin():DateTime «GetAccessor, SetAccessor, attributes, property»IdentityAttribute():AccountIdentityAttribute[*] GetIdentityAttribute():Dictionary<T1->string,T2->object> SetIdentityAttribute(in attributes:Dictionary<T1->string,T2->object>):void

«attributes» Request
(from Root::Gyes.Provisioning.Entities::Gyes::Provisioning::Entities)
<ul style="list-style-type: none"> _state:RequestState _requestingUser:string _accountType:string _users:RequestUser[*] _data:RequestValue[*] _disagreements:NonConformity[*] _roles:Role[*] _resources:RequestResource[*] _answers:Answer[*] _passwordPolicy:PasswordPolicy _attributes:IdentityAttribute[*]
<ul style="list-style-type: none"> «GetAccessor, SetAccessor, attributes, property»_idRequest():Guid «GetAccessor, SetAccessor, attributes, property»Id():string «GetAccessor, SetAccessor, property»RequestingUser():string «GetAccessor, SetAccessor, property»State():RequestState «GetAccessor, SetAccessor, property»AccountType():string «GetAccessor, SetAccessor, attributes, property»Users():RequestUser[*] «GetAccessor, SetAccessor, attributes, property»Data():RequestValue[*] «GetAccessor, SetAccessor, attributes, property»Disagreements():NonConformity[*] «GetAccessor, SetAccessor, attributes, property»Roles():Role[*] «GetAccessor, SetAccessor, attributes, property»Resources():RequestResource[*] «GetAccessor, SetAccessor, attributes, property»Answers():Answer[*] «GetAccessor, SetAccessor, property»PasswordPolicy():PasswordPolicy «GetAccessor, SetAccessor, attributes, property»IdentityAttribute():IdentityAttribute[*] GetUsers():Dictionary<T1->string,T2->bool> SetUsers(in users:Dictionary<T1->string,T2->bool>):void GetData():Dictionary<T1->string,T2->object> SetData(in data:Dictionary<T1->string,T2->object>):void GetNonConformity():List<T1->NonConformity> GetRoles():List<T1->Role> GetRequestResource():List<T1->RequestResource> GetSecurityQuestion():List<T1->Answer> GetIdentityAttribute():List<T1->IdentityAttribute>

«attributes, partial» Configuration
(from Root::Gyes.Provisioning.Entities::Gyes::Provisioning::Entities)
<ul style="list-style-type: none"> _item:object _type:ConfigurationType _task:ConfigurationTask
<ul style="list-style-type: none"> «GetAccessor, SetAccessor, attributes, property»Item():object «GetAccessor, SetAccessor, attributes, property»type():ConfigurationType «GetAccessor, SetAccessor, attributes, property»task():ConfigurationTask

«attributes» Connector
(from Root::Gyes.Provisioning.Entities::Gyes::Provisioning::Entities)
<ul style="list-style-type: none"> «attributes»_idConnector:Guid
<ul style="list-style-type: none"> «GetAccessor, SetAccessor, property»_name():string «GetAccessor, SetAccessor, property»_description():string «GetAccessor, SetAccessor, attributes, property»IdConnector():string «GetAccessor, SetAccessor, property»Name():string «GetAccessor, SetAccessor, property»Description():string

«attributes» Answer
(from Root::Gyes.Provisioning.Entities::Gyes::Provisioning::Entities)
<ul style="list-style-type: none"> «attributes»_idAnswer:Guid «attributes»_idSecurityQuestion:Guid _valueAnswer:string
<ul style="list-style-type: none"> «GetAccessor, SetAccessor, attributes, property»Id():string «GetAccessor, SetAccessor, attributes, property»IdSecurityQuestion():string «GetAccessor, SetAccessor, attributes, property»Value():string



“Subsistema de Aprovechamiento de Usuarios para el Sistema de Administración de Identidades”

«attributes» PasswordPolicy
(from Root::Gyes.Provisioning.Entities::Gyes::Provisioning::Entities)
<ul style="list-style-type: none"> 📄 «attributes»_idPolicy:Guid 📄 _name:string 📄 _maxPasswordRecord:int 📄 _description:string 📄 _policyAge:PasswordPolicyAge 📄 _expirationWarning:int 📄 _policy:string
<ul style="list-style-type: none"> 🔹 «GetAccessor, SetAccessor, attributes, property» IdPolicy():string 🔹 «GetAccessor, SetAccessor, property» Name():string 🔹 «GetAccessor, SetAccessor, attributes, property» MaxPasswordRecord():string 🔹 «GetAccessor, SetAccessor, property» Description():string 🔹 «GetAccessor, SetAccessor, property» Age():PasswordPolicyAge 🔹 «GetAccessor, SetAccessor, attributes, property» ExpirationWarning():string 🔹 «GetAccessor, SetAccessor, property» Policy():string

«attributes, partial» ConfigurationAutomaticType
(from Root::Gyes.Provisioning.Entities::Gyes::Provisioning::Entities)
<ul style="list-style-type: none"> 📄 _value:AutomaticTypeValue
<ul style="list-style-type: none"> 🔹 «GetAccessor, SetAccessor, attributes, property» value():AutomaticTypeValue

«attributes» PasswordHistory
(from Root::Gyes.Provisioning.Entities::Gyes::Provisioning::Entities)
<ul style="list-style-type: none"> 📄 «attributes»_idHistory:Guid 📄 «attributes»_idAccount:Guid 📄 _passwordsChanged:History[*]
<ul style="list-style-type: none"> 🔹 «GetAccessor, SetAccessor, attributes, property» IdHistory():string 🔹 «GetAccessor, SetAccessor, attributes, property» IdAccount():string 🔹 «GetAccessor, SetAccessor, attributes, property» PasswordsChanged():History[*] 🔹 GetHistory():List<T1->History> 🔹 AddHistory(in history:History):void

«attributes» AccountIdentityAttribute
(from Root::Gyes.Provisioning.Entities::Gyes::Provisioning::Entities)
<ul style="list-style-type: none"> 📄 _name:string 📄 _value:string
<ul style="list-style-type: none"> 🔹 «GetAccessor, SetAccessor, attributes, property» name():string 🔹 «GetAccessor, SetAccessor, attributes, property» value():string

«attributes» Template
(from Root::Gyes.Provisioning.Entities::Gyes::Provisioning::Entities)
<ul style="list-style-type: none"> 📄 _name:string 📄 _description:string 📄 _passwordPolicy:PasswordPolicy 📄 _securityQuestions:SecurityQuestion[*] 📄 _attributes:IdentityAttribute[*] 📄 _roles:Role[*] 📄 _resources:RequestResource[*]
<ul style="list-style-type: none"> 🔹 «GetAccessor, SetAccessor, property» Name():string 🔹 «GetAccessor, SetAccessor, property» Description():string 🔹 «GetAccessor, SetAccessor, property» PasswordPolicy():PasswordPolicy 🔹 «GetAccessor, SetAccessor, property» SecurityQuestions():SecurityQuestion[*] 🔹 «GetAccessor, SetAccessor, attributes, property» Attributes():IdentityAttribute[*] 🔹 «GetAccessor, SetAccessor, property» Roles():Role[*] 🔹 «GetAccessor, SetAccessor, attributes, property» Resources():RequestResource[*] 🔹 GetSecurityQuestions():List<T1->SecurityQuestion> 🔹 GetAttributes():List<T1->IdentityAttribute> 🔹 GetRoles():List<T1->Role> 🔹 GetResources():List<T1->RequestResource>

«attributes» Resource
(from Root::Gyes.Provisioning.Entities::Gyes::Provisioning::Entities)
<ul style="list-style-type: none"> 📄 «attributes»_idResource:Guid 📄 _name:string 📄 _description:string 📄 _state:ResourceState 📄 _schedules:Schedule[*] 📄 «attributes»_connector:Guid 📄 _connectionData.ResourceConnectionData[*]
<ul style="list-style-type: none"> 🔹 «GetAccessor, SetAccessor, attributes, property» Id():string 🔹 «GetAccessor, SetAccessor, property» Name():string 🔹 «GetAccessor, SetAccessor, property» Description():string 🔹 «GetAccessor, SetAccessor, property» State():ResourceState 🔹 «GetAccessor, SetAccessor, property» Connector():string 🔹 «GetAccessor, SetAccessor, attributes, property» Schedule():Schedule[*] 🔹 «GetAccessor, SetAccessor, attributes, property» ConnectionData():ResourceConnectionData[*] 🔹 GetConnectionData():Dictionary<T1->string,T2->object> 🔹 SetConnectionData(in connectionData:Dictionary<T1->string,T2->object>):void 🔹 GetSchedules():List<T1->Schedule>



“Subsistema de Aprovechamiento de Usuarios para el Sistema de Administración de Identidades”

«attributes, partial» ConfigurationAutomatic
(from Root::Gyes.Provisioning.Entities::Gyes::Provisioning::Entities)
<ul style="list-style-type: none"> _limitTime:string _time:DateTime _type:ConfigurationAutomaticType _weekDay:AutomaticWeekDay
<ul style="list-style-type: none"> «GetAccessor, SetAccessor, attributes, property» LimitTime():string «GetAccessor, SetAccessor, property» Time():DateTime «GetAccessor, SetAccessor, property» Type():ConfigurationAutomaticType «GetAccessor, SetAccessor, property» WeekDay():AutomaticWeekDay

«attributes, partial» PasswordPolicyAge
(from Root::Gyes.Provisioning.Entities::Gyes::Provisioning::Entities)
<ul style="list-style-type: none"> _minAge:string _maxAge:string
<ul style="list-style-type: none"> «GetAccessor, SetAccessor, attributes, property» minAge():string «GetAccessor, SetAccessor, attributes, property» maxAge():string

«attributes, partial» ResourceConnectionData
(from Root::Gyes.Provisioning.Entities::Gyes::Provisioning::Entities)
<ul style="list-style-type: none"> _name:string _value:string
<ul style="list-style-type: none"> «GetAccessor, SetAccessor, attributes, property» name():string «GetAccessor, SetAccessor, attributes, property» value():string

«attributes, partial» Element
(from Root::Gyes.Provisioning.Entities::Gyes::Provisioning::Entities)
<ul style="list-style-type: none"> _item:object _type:ElementType
<ul style="list-style-type: none"> «GetAccessor, SetAccessor, attributes, property» Item():object «GetAccessor, SetAccessor, attributes, property» type():ElementType

«attributes, partial» ConfigurationManual
(from Root::Gyes.Provisioning.Entities::Gyes::Provisioning::Entities)
<ul style="list-style-type: none"> _limitTime:string
<ul style="list-style-type: none"> «GetAccessor, SetAccessor, attributes, property» LimitTime():string

«attributes» Attributes
(from Root::Gyes.Provisioning.Entities::Gyes::Provisioning::Entities)
<ul style="list-style-type: none"> _identityAttributes:IdentityAttribute[*]
<ul style="list-style-type: none"> «GetAccessor, SetAccessor, attributes, property» IdentityAttributes():IdentityAttribute[*] GetIdentityAttributes():List<T1->IdentityAttribute

«attributes» History
(from Root::Gyes.Provisioning.Entities::Gyes::Provisioning::Entities)
<ul style="list-style-type: none"> _password:string _time>PasswordHistoryDateChange
<ul style="list-style-type: none"> «GetAccessor, SetAccessor, property» Password():string «GetAccessor, SetAccessor, property» DateChange():PasswordHistoryDateChange

«attributes» IdentityAttribute
(from Root::Gyes.Provisioning.Entities::Gyes::Provisioning::Entities)
<ul style="list-style-type: none"> _name:string _typeData:object _description:string _relatedAttribute:RelatedAttribute[*] _validations:RequestAttributeValidation[*]
<ul style="list-style-type: none"> «GetAccessor, SetAccessor, property» Name():string «GetAccessor, SetAccessor, property» TypeData():string «GetAccessor, SetAccessor, property» Description():string «GetAccessor, SetAccessor, attributes, property» RelatedAttribute():RelatedAttribute[*] «GetAccessor, SetAccessor, attributes, property» Validations():RequestAttributeValidation[*] GetValidations():Dictionary<T1->string,T2->object> SetValidations(in validationAttr:Dictionary<T1->string,T2->object>):void AddValidations(in validationName:string, in validationValue:object):void

«attributes, partial» RequestResource
(from Root::Gyes.Provisioning.Entities::Gyes::Provisioning::Entities)
<ul style="list-style-type: none"> _id:string _name:string _description:string
<ul style="list-style-type: none"> «GetAccessor, SetAccessor, attributes, property» id():string «GetAccessor, SetAccessor, attributes, property» name():string «GetAccessor, SetAccessor, attributes, property» description():string



"Subsistema de Aprovisionamiento de Usuarios para el Sistema de
Administración de Identidades"

«attributes, partial» RequestUser
(from Root::Gyes.Provisioning.Entities::Gyes::Provisioning::Entities)
[icon] _user:string [icon] _value:bool
[icon] «GetAccessor, SetAccessor, attributes, property» user():string [icon] «GetAccessor, SetAccessor, attributes, property» value():bool

«attributes, partial» RequestAttributeValidation
(from Root::Gyes.Provisioning.Entities::Gyes::Provisioning::Entities)
[icon] _name:string [icon] _value:string
[icon] «GetAccessor, SetAccessor, attributes, property» name():string [icon] «GetAccessor, SetAccessor, attributes, property» value():string

«attributes, partial» RequestValue
(from Root::Gyes.Provisioning.Entities::Gyes::Provisioning::Entities)
[icon] _name:string [icon] _value:string
[icon] «GetAccessor, SetAccessor, attributes, property» name():string [icon] «GetAccessor, SetAccessor, attributes, property» value():string

«attributes, partial» ResourceState
(from Root::Gyes.Provisioning.Entities::Gyes::Provisioning::Entities)
[icon] _enable:bool
[icon] «GetAccessor, SetAccessor, attributes, property» enable():bool

«attributes» Schedule
(from Root::Gyes.Provisioning.Entities::Gyes::Provisioning::Entities)
[icon] _startTime:DateTime [icon] _endTime:DateTime
[icon] «GetAccessor, SetAccessor, attributes, property» StartTime():System.DateTime [icon] «GetAccessor, SetAccessor, attributes, property» EndTime():System.DateTime

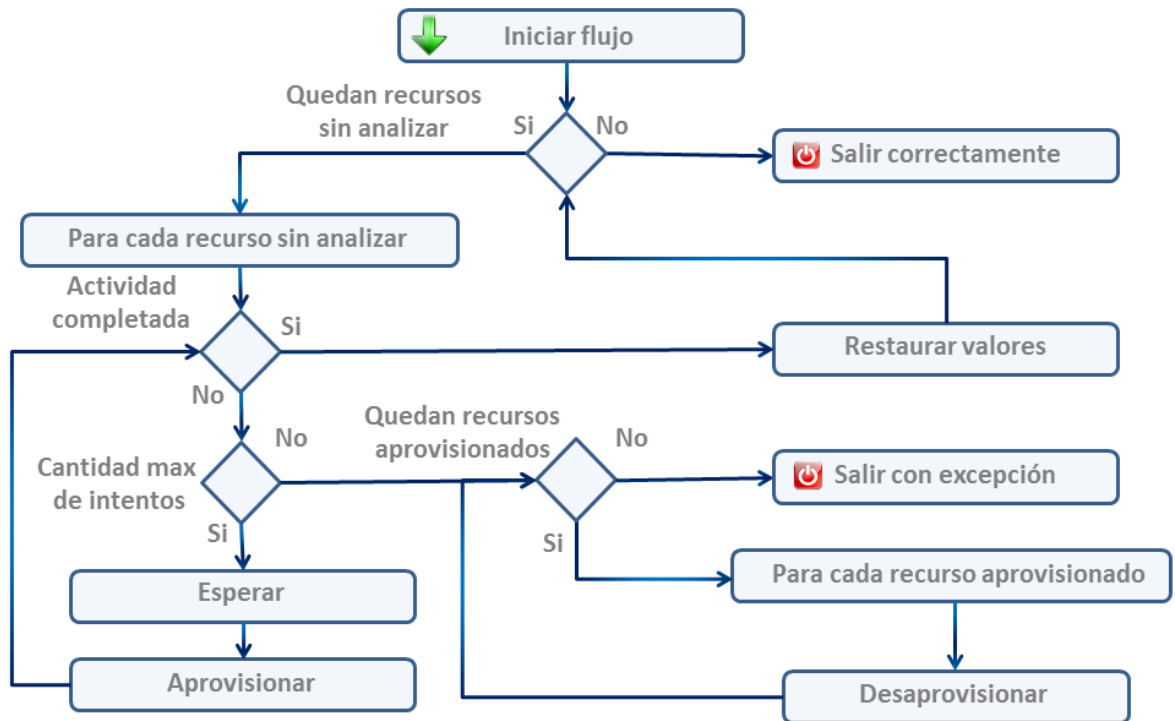
«attributes» Role
(from Root::Gyes.Provisioning.Entities::Gyes::Provisioning::Entities)
[icon] «attributes»_id:Guid [icon] _name:string [icon] _description:string
[icon] «GetAccessor, SetAccessor, attributes, property» Id():string [icon] «GetAccessor, SetAccessor, attributes, property» Name():string [icon] «GetAccessor, SetAccessor, attributes, property» Description():string

«attributes» SecurityQuestion
(from Root::Gyes.Provisioning.Entities::Gyes::Provisioning::Entities)
[icon] «attributes»_idQuestion:Guid [icon] _question:string [icon] _description:string
[icon] «GetAccessor, SetAccessor, property» IdQuestion():string [icon] «GetAccessor, SetAccessor, property» Value():string [icon] «GetAccessor, SetAccessor, property» Description():string

«attributes, partial» State
(from Root::Gyes.Provisioning.Entities::Gyes::Provisioning::Entities)
[icon] _value:AccountStateValue [icon] _description:string [icon] _date:DateTime
[icon] «GetAccessor, SetAccessor, attributes, property» value():AccountStateValue [icon] «GetAccessor, SetAccessor, attributes, property» description():string [icon] «GetAccessor, SetAccessor, attributes, property» date():DateTime



Anexo X: Flujo lógico que representa un flujo de trabajo de aprovisionamiento de usuarios del Motor de Tareas.





“Subsistema de Aprovisionamiento de Usuarios para el Sistema de Administración de Identidades”

Anexo XI: Interfaces del Subsistema.

Gestionar Conectores:

The screenshot shows the 'Gestionar Conectores' interface. The top navigation bar includes the Gyes logo, the text 'Administración de Identidades', and a breadcrumb trail: 'Sistema de Aprovisionamiento | Administrador de Acceso | Administrador de Contrato | Administrador de Sesiones'. The left sidebar is titled 'Aprovisionamiento' and contains a menu with 'Gestionar Conectores' selected. The main content area displays a table titled 'Conectores' with the following data:

	Nombre	Descripción
1	ZimbraConnector	Para servidores de Zimbra
2	Oracle11gConnector	Para servidores de oracle

At the bottom of the table, there is a pagination control showing 'Página 1 de 1' and 'Mostrando 1 - 2 de 2'.

Crear Conector:

The screenshot shows the 'Crear Conector' interface. The top navigation bar is identical to the previous screenshot. The left sidebar is also identical. The main content area displays a form titled 'Conectores' with the following fields and controls:

- Nombre:** A text input field.
- Descripción:** A larger text area for entering details.
- Cargar Archivo Zip:** A text input field with an 'Examinar...' button next to it.
- Buttons:** 'Crear' and 'Cancelar' buttons at the bottom.



“Subsistema de Aprovisionamiento de Usuarios para el Sistema de Administración de Identidades”

Gestionar Políticas de Contraseña:

Bienvenido Nombre Usuario

Sistema de Aprovisionamiento | Administrador de Acceso | Administrador de Contrato | Administrador de Sesiones

Aprovisionamiento

- Gestionar Conectores
- Gestionar Plantillas
- Gestionar Políticas**
- Gestionar Preguntas
- Administrar Recursos
- Administrar Solicitudes
- Administrar Cuentas
- Solicitar Cuentas
- Configuración

Políticas de Contraseña

	Nombre ↕	Descripción	Edad Max	Edad Min	Expiración	Registro
1	Política por defecto	La contraseña tiene que tener co	90	3	5	25
2	Política de prueba	letras mayusculas minusculas, ce	90	3	3	24

Mostrando 1 - 2 de 2

Crear Política de Contraseña:

Sistema de Aprovisionamiento | Administrador de Acceso | Administrador de Contrato | Administrador de Sesiones

Aprovisionamiento

- Gestionar Conectores
- Gestionar Plantillas
- Gestionar Políticas**
- Gestionar Preguntas
- Administrar Recursos
- Administrar Solicitudes
- Administrar Cuentas
- Solicitar Cuentas
- Configuración

Política de Contraseña

Nombre

Registro Máximo

Edad Mínima

Edad Máxima

Aviso de Expiración

Política

Descripción



“Subsistema de Aprovisionamiento de Usuarios para el Sistema de Administración de Identidades”

Gestionar Preguntas de Seguridad:

BIENVENIDO Nombre Usuario

Sistema de Aprovisionamiento | Administrador de Acceso | Administrador de Contrato | Administrador de Sesiones

Aprovisionamiento

- Gestionar Conectores
- Gestionar Plantillas
- Gestionar Políticas
- Gestionar Preguntas**
- Administrar Recursos
- Administrar Solicitudes
- Administrar Cuentas
- Solicitar Cuentas
- Configuración

	Pregunta	Descripción
1	¿Cuál es el nombre de su madre ?	Por solicitud del usuario
2	¿Cual fue su primer trabajo ?	Pregunta de seguridad por defecto del IDM
3	¿Cual es su ocupación actualmente ?	Por solicitud del usuario

Página 1 de 1 | Mostrando 1 - 3 de 3

Crear Pregunta de Seguridad:

BIENVENIDO Nombre Usuario

Sistema de Aprovisionamiento | Administrador de Acceso | Administrador de Contrato | Administrador de Sesiones

Aprovisionamiento

- Gestionar Conectores
- Gestionar Plantillas
- Gestionar Políticas
- Gestionar Preguntas**
- Administrar Recursos
- Administrar Solicitudes
- Administrar Cuentas
- Solicitar Cuentas
- Configuración

Preguntas de Seguridad

Pregunta de Seguridad

Descripción

Crear Cancelar



“Subsistema de Aprovisionamiento de Usuarios para el Sistema de Administración de Identidades”

Administrar Recursos:

Bienvenido Nombre Usuario

Gyes
Administración de Identidades

Sistema de Aprovisionamiento | Administrador de Acceso | Administrador de Contrato | Administrador de Sesiones

Aprovisionamiento

- Gestionar Conectores
- Gestionar Plantillas
- Gestionar Políticas
- Gestionar Preguntas
- Administrar Recursos**
- Administrar Solicitudes
- Administrar Cuentas
- Solicitar Cuentas
- Configuración

	Nombre	Descripción	Habilitado
1	Comedores	BD de Comedores	Sí
2	Correo	Correo UCI	Sí

Página 1 de 1 Mostrando 1 - 2 de 2

Crear Recurso:

Recursos

Oracle11gConnector

Oracle11gConnector

Name

Description

Host

Port

User

Password

DBRole

Horarios

Inicio Fin



“Subsistema de Aprovisionamiento de Usuarios para el Sistema de Administración de Identidades”

Gestionar Plantillas:

Plantillas

	Nombre	Descripción
1	Profesores	Plantilla para crear usuarios de tipo Profesores
2	Trabajadores	Plantilla para la creacion de cuentas de tipo Trabajadores
3	Estudiantes	Plantilla para la creacion de cuentas de tipo Estudiantes

Página 1 de 1 Mostrando 1 - 3 de 3

Crear Plantilla:

Plantilla

Nombre:

Descripción:

Roles

	Nombre	Descripción
<input type="checkbox"/>	Rol 1	Aprobador de solicitudes.
<input type="checkbox"/>	Rol 2	Administrar el sistema.

Recursos

	Nombre	Descripción
<input type="checkbox"/>	Comedores	BD de Comedores
<input type="checkbox"/>	Correo	Correo UCI



“Subsistema de Aprovisionamiento de Usuarios para el Sistema de Administración de Identidades”

Preguntas de Seguridad

Pregunta	Descripción
<input type="checkbox"/> ¿Cuál es el nombre de su madre ?	Por solicitud del usuario
<input type="checkbox"/> ¿Cual fue su primer trabajo ?	Pregunta de seguridad por defecto del IDM
<input type="checkbox"/> ¿Cual es su ocupación actualmente ?	Por solicitud del usuario

Políticas de Contraseña

Nombre	Descripción
<input type="radio"/> Política por defecto	La contraseña tiene que tener como minimo 8 caracteres al menos una letra mayuscula y una minuscula un caracter especial o un numero
<input type="radio"/> Política de prueba	letras mayusculas minusculas, caracteres extraños y numeros

Gestionar Atributos de Identidad:

Aprovisionamiento

- Gestionar Conectores
- Gestionar Plantillas**
- Gestionar Políticas
- Gestionar Preguntas
- Administrar Recursos
- Administrar Solicitudes
- Administrar Cuentas
- Solicitar Cuentas
- Configuración

Gestion de Atributos

Atributos

Campo	Descripción		
<input type="radio"/> Usuario	Usuario de la persona		
<input type="radio"/> Contraseña	Contraseña del usuario		

Atributos Relacionados

Conector	Atributo
<input type="checkbox"/> Oracle11gConnector	USER
<input type="checkbox"/> Oracle11gConnector	USERPASS
<input type="checkbox"/> Oracle11gConnector	default_tablespace
<input type="checkbox"/> Oracle11gConnector	temporary_tablespace



“Subsistema de Aprovisionamiento de Usuarios para el Sistema de Administración de Identidades”

Crear Atributo de Identidad:

The screenshot shows the 'Gyes' system interface with a modal dialog titled 'Atributo'. The dialog is divided into several sections:

- Datos Básicos:** Includes a 'Nombre' (Name) text input field and a 'Descripción' (Description) text area.
- Selección de Opción:** Contains two radio buttons: 'Expresiones Regulares' (unselected) and 'Validaciones Clásicas' (selected).
- Validaciones:** Includes several checkboxes and input fields:
 - Requerido
 - Longitud
 - Longitud Mínima
 - Longitud Máxima
 - Tipo Contraseña

At the bottom of the dialog are 'Add' and 'Cancelar' buttons. The background shows the main system menu with options like 'Gestionar Plantillas', 'Gestionar Políticas', and 'Solicitar Cuentas'.

Solicitar Creación de Cuenta de Usuario:

The screenshot shows the 'Gyes' system interface with a form titled 'Trabajadores'. The form contains the following fields:

- Usuario
- Contraseña
- Correo
- Grupo
- Nombre Completo
- Iniciales
- Cuota
- SN

Below these fields is a section for 'Preguntas de Seguridad' with a question: '¿Cuál es el nombre de su madre?' and a corresponding text input field.

At the bottom of the form are 'Crear' and 'Cancelar' buttons. The left sidebar shows the 'Solicitar Cuentas' menu with sub-options for 'Profesores', 'Trabajadores', and 'Estudiantes'.



“Subsistema de Aprovisionamiento de Usuarios para el Sistema de Administración de Identidades”

Administrar Solicitudes:

Bienvenido Nombre Usuario

Gyes Administración de Identidades

Sistema de Aprovisionamiento | Administrador de Acceso | Administrador de Contrato | Administrador de Sesiones

Aprovisionamiento

- Gestionar Conectores
- Gestionar Plantillas
- Gestionar Políticas
- Gestionar Preguntas
- Administrar Recursos
- Administrar Solicitudes
- Administrar Cuentas
- Solicitar Cuentas
- Configuración

	Tipo de Cuenta	Usuario Solicitante	Usuario de la Cuenta	Estado
1	Estudiantes	Secretaria	reger	Pendiente

Página 1 de 1 Mostrando 1 - 1 de 1

Revisión de Solicitudes:

Bienvenido Nombre Usuario

Gyes Administración de Identidades

Sistema de Aprovisionamiento | Administrador de Acceso | Administrador de Contrato | Administrador de Sesiones

Aprovisionamiento

- Gestionar Conectores
- Gestionar Plantillas
- Gestionar Políticas
- Gestionar Preguntas
- Administrar Recursos
- Administrar Solicitudes
- Administrar Cuentas
- Solicitar Cuentas
- Configuración

Trabajadores - Revisar

Usuario
jcbautista

Contraseña
●●●●●●●●

Correo
jcbautista@mail.lab.uci.cu

Grupo
default

Nombre Completo
José Carlos Correa Bautista

Iniciales
JCCB

Cuota
5000

SN
sn

No Conformidades

No Conformidades	
1 Cuota muy pequeña	06/06/2012 10:35:24

Aceptar Revisar Cancelar



“Subsistema de Aprovisionamiento de Usuarios para el Sistema de Administración de Identidades”

Administrar Cuentas de Usuarios:

The screenshot shows the Gyes web application interface. The top navigation bar includes the Gyes logo and the text "Administración de Identidades". Below this, there are links for "Sistema de Aprovisionamiento", "Administrador de Acceso", "Administrador de Contrato", and "Administrador de Sesiones". The left sidebar contains a menu with options like "Gestionar Conectores", "Gestionar Plantillas", "Gestionar Políticas", "Gestionar Preguntas", "Administrar Recursos", "Administrar Solicitudes", "Administrar Cuentas" (highlighted), "Solicitar Cuentas", and "Configuración". The main content area displays a table titled "Cuentas" with the following data:

	Usuario	Tipo de Cuenta	Última Entrada	Estado
1	ymachadog	Profesores	30/05/2012 21:17	Activa
2	yngarcia	Trabajadores	30/05/2012 21:18	Eliminada
3	jccorrea	Estudiantes	30/05/2012 21:20	Activa
4	jose	Trabajadores	30/05/2012 21:40	Activa
5	joseintymachadog	Estudiantes	31/05/2012 12:10	Activa
6	agpina	Profesores	31/05/2012 8:25	Eliminada

At the bottom of the table, there is a pagination control showing "Página 1 de 1" and "Mostrando 1 - 6 de 6".

Mostrar Datos de la Cuenta:

The screenshot shows the Gyes web application interface with a modal window titled "Información" open. The modal contains the following sections:

- Datos de la Cuenta:**
 - Tipo de Cuenta: Estudiantes
 - Usuario: jccorrea
 - Política de Contraseña: Política por defecto
 - Estado de la Cuenta: Activa
- Recursos:** (Empty field)
- Preguntas de Seguridad:**
 - ¿ Cual es su ocupación actualmente ?
 - ¿ Cual fue su primer trabajo ?
 - ¿ Cuál es el nombre de su madre ?
- Atributos de Identidad:**
 - Tabla por Defecto: TS_USERS
 - Tabla Temporal: TEMP

An "Aceptar" button is located at the bottom right of the modal.



“Subsistema de Aprovisionamiento de Usuarios para el Sistema de Administración de Identidades”

Cambiar Contraseña:

The screenshot shows the 'Cambiar Contraseña' form within the Gyes system. The interface includes a top navigation bar with the Gyes logo and 'Administración de Identidades', a user greeting 'Bienvenido Nombre Usuario', and a secondary navigation bar with links to 'Sistema de Aprovisionamiento', 'Administrador de Acceso', 'Administrador de Contrato', and 'Administrador de Sesiones'. A left sidebar contains a menu with 'Aprovisionamiento' (containing links for 'Gestionar Conectores', 'Gestionar Plantillas', 'Gestionar Políticas', 'Gestionar Preguntas', 'Administrar Recursos', 'Administrar Solicitudes', 'Administrar Cuentas', and 'Solicitar Cuentas') and 'Configuración' (containing 'Cambiar Contraseña', 'Recuperar Contraseña', and 'Seguimiento Solicitud'). The main content area is titled 'Cambiar Contraseña' and contains a form with the following fields: 'Usuario' (text input), 'Contraseña Actual' (password input), 'Contraseña Nueva' (password input), and 'Confirmar Contraseña Nueva' (password input). At the bottom of the form are two buttons: 'Cambiar Contraseña' and 'Cancelar'.

Recuperar Contraseña:

The screenshot shows the 'Recuperar Contraseña' form within the Gyes system. The interface is identical to the previous screenshot, including the top navigation bar, user greeting, secondary navigation bar, and left sidebar menu. The main content area is titled 'Recuperar Contraseña' and contains a form with the following fields: 'Usuario' (text input). At the bottom of the form are two buttons: 'Siguiete' and 'Cancelar'.



“Subsistema de Aprovisionamiento de Usuarios para el Sistema de Administración de Identidades”

Seguimiento de Solicitudes:

The screenshot displays the Gyes web application interface. The top navigation bar includes the Gyes logo, the text 'Administración de Identidades', and a user greeting 'Bienvenido Nombre Usuario'. Below this, a secondary navigation bar lists 'Sistema de Aprovisionamiento', 'Administrador de Acceso', 'Administrador de Contrato', and 'Administrador de Sesiones'. The left sidebar menu is divided into two sections: 'Aprovisionamiento' (with options like Gestionar Conectores, Gestionar Plantillas, Gestionar Políticas, Gestionar Preguntas, Administrar Recursos, Administrar Solicitudes, Administrar Cuentas, and Solicitar Cuentas) and 'Configuración' (with options like Cambiar Contraseña, Recuperar Contraseña, and Seguimiento Solicitud). The main content area is titled 'Seguimiento de solicitudes' and features a search field for 'Usuario Solicitante' with 'Aceptar' and 'Cancelar' buttons. Below this is a table with columns for 'Tipo de Cuenta', 'Usuario Solicitante', 'Usuario de la Cuenta', and 'Estado'. The table is currently empty, and a message at the bottom right states 'No existen Solicitudes'. A pagination bar at the bottom indicates 'Página 1 de 1'.

Configuración del Bloqueo de Cuentas por Desuso:

The screenshot displays the Gyes web application interface for the 'Bloqueo de Cuentas' configuration. The top navigation bar and secondary navigation bar are identical to the previous screenshot. The left sidebar menu is also identical, but the 'Configuración' section is expanded to show 'Administración de Cuentas' and 'Configuración'. The main content area is titled 'Bloqueo de Cuentas' and contains the following configuration options: 'Tipo de Configuración' with radio buttons for 'Manual' and 'Automática' (selected); 'Configuración Automática' with a 'Fecha' section containing a dropdown for 'Semanal' and a dropdown for 'Miércoles'; and an 'Horas' section with input fields for 'Hora del Análisis' (set to 23:31) and 'Tiempo Máximo de Desuso en Días' (set to 3). At the bottom, there are buttons for 'Guardar', 'Cancelar', and 'Valores por Defecto'.



“Subsistema de Aprovisionamiento de Usuarios para el Sistema de Administración de Identidades”

Configuración de Eliminación de Cuentas Bloqueadas:

Aprovisionamiento

- Gestionar Conectores
- Gestionar Plantillas
- Gestionar Políticas
- Gestionar Preguntas
- Administrar Recursos
- Administrar Solicitudes
- Administrar Cuentas**
- Solicitar Cuentas
- Configuración

Eliminación de Cuentas

Tipo de Configuración

Manual

Automática

Configuración Automática

Fecha

Semanal

Miércoles

Horas

Hora del Análisis

1:00

Tiempo Máximo de Desuso en Días

90

Guardar Cancelar Valores por Defecto



*“Subsistema de Aprovisionamiento de Usuarios para el Sistema de
Administración de Identidades”*

Anexo XII: Pruebas Unitarias.

Prueba de Unidad				
Nombre Prueba: CreateRequestTest				
Estado: Satisfactoria	Tipo: Caja Blanca	Ultima Ejecución: 17/05/2012		
Ejecutado por: Yendry Machado	Verificado por: Yanisleivi Valdés			
Descripción: Para la ejecución de esta prueba se debe seleccionar la plantilla del menú "Solicitar Cuenta", luego se introducen los datos que pide la plantilla seleccionada para la creación de cuentas de usuario en los recursos contenidos en dicha plantilla y se accede a la opción "Crear".				
Entrada: Request request				
Criterio de aceptación: Crea una solicitud.				
Resultado:				
✔ Test run completed Results: 1/1 passed; Item(s) checked: 0				
	Result	Test Name	Project	Error Message
<input type="checkbox"/>	✔ Passed	CreateRequestTest	WebApplicationTest	

Prueba de Unidad				
Nombre Prueba: DeleteRequestTest				
Estado: Satisfactoria	Tipo: Caja Blanca	Ultima Ejecución: 17/05/2012		
Ejecutado por: Yendry Machado	Verificado por: Yanisleivi Valdés			
Descripción: Para la ejecución de esta prueba se debe seleccionar la solicitud del listado de solicitudes y acceder a la opción "Eliminar Solicitud", luego se elimina la solicitud del sistema.				
Entrada: Guid id				
Criterio de aceptación: Elimina una solicitud.				
Resultado:				
✔ Test run completed Results: 1/1 passed; Item(s) checked: 0				
	Result	Test Name	Project	Error Message
<input type="checkbox"/>	✔ Passed	DeleteRequestTest	WebApplicationTest	



*“Subsistema de Aprovisionamiento de Usuarios para el Sistema de
Administración de Identidades”*

Prueba de Unidad				
Nombre Prueba: GetRequestsTest				
Estado: Satisfactoria	Tipo: Caja Blanca	Ultima Ejecución: 17/05/2012		
Ejecutado por: Yendry Machado	Verificado por: Yanisleivi Valdés			
Descripción: Para la ejecución de esta prueba se debe seleccionar la opción "Administrar Solicitudes", luego el sistema devuelve una lista con todas las solicitudes existentes.				
Entrada:				
Criterio de aceptación: Devuelve una lista con las solicitudes.				
Resultado:				
✓ Test run completed Results: 1/1 passed; Item(s) checked: 0				
	Result	Test Name	Project	Error Message
<input type="checkbox"/>	Passed	GetRequestsTest	WebApplicationTest	

Anexo XIII: Casos de Prueba.

- **Caso de prueba Administrar Recursos.**
- **Caso de prueba Gestionar Plantillas.**
- **Caso de prueba Gestionar Conectores.**
- **Caso de prueba Administrar Cuentas de Usuario.**
- **Caso de prueba Administrar Solicitudes.**
- **Caso de prueba Configuraciones Generales.**
- **Caso de prueba Gestionar Políticas de Contraseña.**
- **Caso de prueba Gestionar Preguntas de Seguridad**