Universidad de las Ciencias Informáticas

Facultad 6

Trabajo de Diploma para optar por el título de Ingeniero en Ciencias Informáticas

SIGE-Droid: Herramienta de apoyo al trabajo con el Sistema Integrado de Gestión Estadística

Autor:

Carlos A. Martínez Gil.

Tutores:

Ing. Ailyn Romero Alonso.

Ing. Alejandro González Sánchez.

La Habana, Julio de 2016

"Año 58 de la Revolución"

"Un sueño que sueñas solo es sólo un sueño. Un sueño que sueñas con alguien es una realidad."

John Lennon.

Declaración de autoría:

Declaro ser autor de la presente tesis y reconozco a la Universidad de las Ciencias Informáticas los derechos patrimoniales de la misma, con carácter exclusivo.

Para que así conste firmo la presente a los ____ días del mes de _____ del año 2016.

Carlos Alberto Martínez Gil

_____
Firma de Autor

Ing. Ailyn Romero Alonso

_____
Firma de Tutora

Ing. Alejandro González Sánchez

_____
Firma de Tutor

Datos del contacto

Tutora: Ing. Ailyn Romero Alonso

Universidad de las Ciencias Informáticas, La Habana, Cuba

Especialidad de graduación: Ingeniería en Ciencias Informáticas

Correo Electrónico: aralonso@uci.cu

Tutor: Ing. Alejandro González Sánchez

Universidad de las Ciencias Informáticas, La Habana, Cuba

Especialidad de graduación: Ingeniería en Ciencias Informáticas

Correo Electrónico: alejandrogs@uci.cu

Autor: Carlos Alberto Martínez Gil

Universidad de las Ciencias Informáticas, La Habana, Cuba

Correo Electrónico: camartinez@estudiantes.uci.cu

A mis padres Idania y Carlos, por formarme y educarme, a ellos les debo todo lo que soy. Gracias por sus consejos y su apoyo incondicional.

A mi abuela por su cariño y dedicación, a mi familia, a mis tíos, primos y primas.

A mis tutores Ailyn, Alejandro, por su paciencia, comprensión y por estar siempre dispuestos a ayudarme. Al tribunal y al oponente por su ayuda y sus consejos.

A mis amistades de la UCI que han sido una familia para mí: Leo, Cuzo, Cesar, Migue, Fortun, Leosdani, Micha, Edel, Rafa, Yanitza, Mayi, Indira, Mónica, José Alejandro, Alejandro, Micha, Raúl, Lokiño, Marlon, Eduardo, Henry, Lija, Elián, Jorgito, Yamil, Víctor. Les doy las gracias por los buenos momentos que hemos pasado juntos.

A mis padres y a mi abuela por su educación y su entrega incondicional.

Resumen

Debido a la actualización del modelo económico empresarial que se acomete a nivel nacional, cada vez son más las entidades que utilizan el Sistema Integrado de Gestión Estadística (SIGE) para la gestión de su Sistema de Información. Actualmente, para la consulta de los datos almacenados en SIGE, es necesario acceder desde una computadora que está conectada a la red mediante el uso de la conexión cableada, limitando el área de acceso a la información. El objetivo de la investigación es desarrollar una aplicación que permita visualizar reportes y la gestión de seguridad en SIGE desde dispositivos móviles con sistema operativo Android. Para ello se realizó un análisis de SIGE identificando las características fundamentales que debía tener la herramienta. También se estudiaron aplicaciones Android que se conectan a bases de datos externas identificando la necesidad de usar servicios web en el desarrollo de la herramienta. Se identificaron y describieron los requisitos funcionales y no funcionales, proporcionando el punto de partida para las actividades de implementación. Posteriormente se implementó la herramienta SIGE-Droid posibilitando la gestión de seguridad y visualización de reportes en SIGE desde dispositivos móviles con SO Android. Se realizaron pruebas para validar el correcto funcionamiento de la herramienta corrigiendo los errores detectados. Como resultado se obtuvo la aplicación SIGE-Droid que permite visualizar los reportes almacenados en SIGE desde cualquier espacio de la institución mediante el uso de la red inalámbrica. De igual modo permite la gestión de usuarios y roles por parte del equipo de administración del sistema.

Palabras claves: Android, información, reportes, SIGE-Droid, Sistema de información, Sistema Integrado de Gestión Estadística.

Abstract

Because updating the business economic model that is undertaken at national level, they are increasingly entities using the Integrated Statistical Management System for managing its Information System. Currently, for querying data stored in SIGE, you must access from a computer that is connected to the network using a wired connection, limiting the area of access to information. The aim of the research is to develop an application that can view reports and safety management SIGE from mobile devices with Android operating system. To do an analysis of SIGE identifying the fundamental characteristics that must have made the tool. Android applications that connect to external databases identifying the need for web services in the development of the tool were also studied. They were identified and described the functional and non-functional requirements, providing the starting point for implementation activities. Later the SIGE-Droid tool enabling security management and viewing reports in SIGE from mobile devices with Android OS was implemented. Tests were conducted to validate the proper operation of the tool correcting the detected errors. As a result the SIGE-Droid application that displays reports stored in SIGE from any area of the institution by using the wireless network is obtained. Similarly allows the management of users and roles by the system administration team.

Keyword: *Android, information, reports, SIGE-Droid, Integrated Management System Statistics, Information System.*

Índice de Contenido

Resumen VI

Abstract VII

Introducción 1

Capítulo 1: Fundamentación Teórica de SIGE-Droid 5

1.1 Análisis de SIGE 5

1.2 Aplicaciones Android que se conectan a bases de datos externas 10

1.2.1 Facebook 11

1.2.2 Google Play 11

1.2.3 Gmail 11

1.3 Servicios web 12

1.4 ¿Por qué Android? 13

1.5 Metodologías de desarrollo de software 14

1.6 Lenguaje de modelado 15

1.7 Herramienta CASE 16

1.8 Lenguaje de programación 16

1.9 Entorno de desarrollo integrado 17

1.10 Sistema Gestor de Base de Datos 17

1.11 Conclusiones del capítulo 18

Capítulo 2: Análisis y diseño de SIGE-Droid 19

2.1 Modelo de Dominio 19

2.2 Propuesta de solución 20

2.3 Especificación de los requisitos del sistema 20

2.3.1. Requisitos funcionales 21

2.3.2. Requisitos no funcionales 24

2.4 Modelo de Caso de Uso del sistema 25
2.4.1 Diagrama de CU de la herramienta 26
2.4.2 Descripción textual del CU Gestionar usuario 27
2.5 Patrones arquitectónicos 33
2.6 Modelo del Diseño 34
2.6.1. Diagrama de clases del diseño 34
2.6.2 Diagrama de secuencia 36
2.7 Patrones empleados en la solución 37
2.7.1 Patrones de diseño GRASP 37
2.7.2 Patrones de diseño GoF 38
2.8 Modelo de datos 38
2.9 Modelo de despliegue 39
2.10 Conclusiones del capítulo 39
Capítulo 3: Implementación y pruebas de SIGE-Droid 41
3.1 Modelo de implementación 41
3.1.1 Diagrama de componentes 41
3.2 Código fuente 42
3.2.1 Estándar de codificación 42
3.3 Pruebas de software 44
3.3.1 Pruebas de integración 45
3.3.2 Pruebas de aceptación 47
3.4 Interfaces de la aplicación SIGE-Droid 48
3.5 Conclusiones del capítulo 50
Conclusiones generales 52
Referencias Bibliográficas 54

Índice de Tablas

Tabla 1: Comparación entre los principales sistemas operativos para móviles 13
Tabla 2: Descripción de los actores del sistema. 26
Tabla 3: Descripción del CU "Gestionar usuario." 27
Tabla 4 Secciones de prueba para el CU Gestionar usuario.....45
Tabla 5 Descripción de las variables del caso de prueba del CU Gestionar usuario 45
Tabla 6 Escenarios del caso de prueba Adicionar usuario del CU Gestionar usuario 46

Índice de Figuras

Fig. 1: Modelo de dominio de SIGE-Droid. 19
Fig. 2: Diagrama de CU de SIGE-Droid 26
Fig. 3: DCD del CU Gestionar usuario. 35
Fig. 4: Diagrama de secuencia del escenario Adicionar usuario. 36
Fig. 5: Modelo de datos de SIGE-Droid. 38
Fig. 6: Diagrama de Despliegue de SIGE-Droid. 39
Fig. 7: Diagrama de Componentes del CU Gestionar usuario. 41
Fig. 8: Fragmento de código fuente de la clase GestionarUsuarioRolActivity. 44
Fig. 9: Gráfica de iteraciones de pruebas funcionales. 47
Fig. 10: Gráfica de iteraciones de pruebas Alfa. 48
Fig. 11: Interfaz principal para la gestión de seguridad en SIGE-Droid. 49
Fig. 12: Interfaz para visualizar reporte en SIGE-Droid 50

Introducción

En los comienzos de la civilización se manifestaron formas sencillas de estadística en paredes de cuevas para contar el número de personas, animales u objetos utilizados para la supervivencia. Es por eso que en sus inicios el término estadística designaba la recolección sistemática de datos demográficos y económicos asociados a los intereses de los Estados. Posteriormente se convirtió en una disciplina matemática utilizada no solo en la ciencia sino en la manufactura y la política.

En la actualidad la estadística se ha convertido en un elemento a incorporar en sistemas informáticos, pues incide en el entendimiento efectivo de los datos; y permite el apoyo a la toma de decisiones ante disímiles situaciones a las cuales se enfrentan las empresas a diario. Ante esta tendencia estadística, el uso de aplicaciones que gestionen Sistemas de Información[1] es cada vez más utilizado por entidades y empresas.

Para dichas instituciones es de gran importancia tener concentrada en una plataforma informática toda la información relevante para la entidad. Esto les permitirá una gestión única de los datos y las bases para el análisis, las tendencias y estilos que puede tomar el negocio, teniendo en cuenta la recopilación o gestión de la información estadística.

La Universidad de las Ciencias Informáticas (UCI), para la gestión de los procesos mencionados con anterioridad, propone la aplicación informática Sistema Integrado de Gestión Estadística (SIGE); la cual soporta conceptos genéricos sobre bases metodológicas similares a los establecidos por la Oficina Nacional de Estadísticas e Información (ONEI). Esta última es el órgano del Consejo de Ministros encargado de dirigir metodológicamente la gestión de la información relevante para el Gobierno y la aplicación de la política estatal en materia de estadística.

SIGE realiza tres procesos fundamentales: la captura de datos estadísticos utilizando plantillas (de formularios o encuestas) diseñadas previamente; la validación automática de la información captada haciendo uso de reglas de validación definidas por el usuario; y el apoyo a la toma de decisiones mediante reportes que permitan evaluar el comportamiento de determinada actividad. Los reportes son una forma de organizar y exhibir la información contenida en una base de datos. Su función es aplicar un formato

[1] Un Sistema de Información es un conjunto organizado de personas, procesos y recursos, incluyendo la información y sus tecnologías asociadas, que interactúan de forma dinámica, para satisfacer las necesidades informativas que posibilitan alcanzar los objetivos de una o varias organizaciones.

determinado a los datos para mostrarlos por medio de un diseño facilitando su interpretación por parte de los usuarios.

Para la consulta de los datos almacenados en SIGE es necesario acceder a una computadora que esté conectada a una red física, limitando el área de acceso de los usuarios al sistema. Esta limitante conlleva a un conjunto de irregularidades:

- Es necesaria la impresión de los reportes almacenados en SIGE, los cuales serán analizados en el proceso de toma de decisiones, ya que no es posible acceder a ellos desde cualquier parte de la organización.
- Si se necesita consultar nuevos datos, es preciso que se imprima nuevamente la información a analizar. Esto imposibilita la toma de decisiones inmediata por parte de los directivos de la empresa, trayendo consecuencias negativas en la organización.
- El equipo encargado de la administración del sistema está limitado a la gestión de los usuarios, roles y permisos desde sus puestos de trabajo; ya que solo pueden acceder al sistema desde una computadora que esté conectada a través de la red física, lo cual incide directamente en la ejecución de estas actividades y se ve afectada la realización de las tareas de administración.

La situación anteriormente descrita permite identificar el siguiente **problema de investigación:** ¿Cómo visualizar reportes y gestionar la seguridad para apoyar el trabajo con SIGE desde dispositivos móviles?

La investigación tiene como **objeto de estudio** la visualización de reportes y gestión de seguridad en SIGE y enmarca su **campo de acción** en la visualización de reportes y gestión de seguridad en SIGE desde dispositivos móviles con sistema operativo Android.

El **objetivo general** de la investigación es: Desarrollar una aplicación que permita la visualización de reportes y la gestión de seguridad en SIGE desde dispositivos móviles con sistema operativo Android.

Del objetivo general se desglosan los siguientes **objetivos específicos**:

- Analizar los conceptos, metodología, tecnologías y herramientas necesarias para el desarrollo de SIGE-Droid.
- Realizar el análisis y diseño de la herramienta para guiar la implementación de SIGE-Droid.
- Realizar la implementación y pruebas de SIGE-Droid para apoyar el trabajo con SIGE.

Para dar cumplimiento a los objetivos planteados, se deben realizar las siguientes **tareas de investigación:**

- Análisis de los conceptos básicos y técnicos implicados en el desarrollo de SIGE-Droid para elaborar el marco teórico de la investigación.
- Selección de la metodología, tecnologías y herramientas a utilizar para guiar y desarrollar la implementación de SIGE-Droid.
- Identificación de las necesidades funcionales y tecnológicas de SIGE-Droid para su correcto funcionamiento.
- Identificación de los principios arquitectónicos de SIGE-Droid para definir su estructura global.
- Identificación de los principios de diseño y funcionamiento de SIGE-Droid para guiar el proceso de implementación.
- Implementación de los principios de diseño y funcionamiento para satisfacer las necesidades funcionales y tecnológicas de SIGE-Droid.
- Diseño de los casos de pruebas para determinar el correcto funcionamiento de los requisitos.
- Realización de las pruebas de software para comprobar el correcto funcionamiento de SIGE-Droid.

Las tareas de la investigación se derivaron de las siguientes **preguntas científicas:**

1. ¿Cuáles son los fundamentos teóricos para visualizar reportes y administrar la seguridad en SIGE mediante el uso de dispositivos móviles?
2. ¿Qué funcionalidades debe tener SIGE-Droid para visualizar reportes y administrar la seguridad en SIGE haciendo uso de dispositivos móviles?
3. ¿Cómo se deben desarrollar las funcionalidades identificadas?
4. ¿La herramienta desarrollada cumple con los requisitos identificados?

Para el cumplimiento del objetivo general planteado se utilizarán los siguientes **métodos de investigación**:

- Histórico - Lógico: Se utiliza para analizar los principales conceptos de SIGE, así como sus características y funcionamiento.
- Analítico - Sintético: Se utiliza para el estudio de la metodología y diferentes herramientas que se emplean, además para profundizar en determinados conceptos asociados al problema.
- Modelación: Se utiliza para realizar los diagramas correspondientes a los modelos de análisis, diseño e implementación de SIGE-Droid.
- Tormenta de ideas: Se utiliza para obtener las necesidades del cliente y realizar el levantamiento de

los requisitos con los que debe cumplir SIGE-Droid.

El Trabajo de Diploma está estructurado de la siguiente manera: introducción, tres capítulos, conclusiones, recomendaciones, referencias bibliográficas y anexos.

Capítulo 1: Fundamentación Teórica de SIGE-Droid

En este capítulo se presenta la definición del marco teórico de la investigación. Se realiza el estudio de las funcionalidades y características de SIGE a tener en cuenta para la realización de SIGE-Droid. Además se analizan aplicaciones desarrolladas en Android para determinar la forma en la que estas se conectan a bases de datos externas. Se aborda el estudio de la metodología, herramientas, tecnologías y lenguajes a utilizar en el desarrollo de SIGE-Droid.

Capítulo 2: Análisis y Diseño de SIGE-Droid

En este capítulo se define el modelo de dominio y se identifican, especifican y describen los requisitos funcionales y no funcionales de la aplicación. Contiene además los actores y casos de uso representados en el diagrama de casos de uso del sistema. En el proceso de diseño se brinda un acercamiento a la implementación de la aplicación. Se construye para ello los diagramas de interacción y el modelo de clases del diseño siguiendo el estilo arquitectónico y los patrones de diseño seleccionados. Esto proporcionará la base para proceder a la etapa de implementación.

Capítulo 3: Implementación y Pruebas de SIGE-Droid

En este capítulo se elaboran los artefactos correspondientes a la implementación tomando como entrada los resultados obtenidos en la etapa de diseño. Se representa el diagrama de componentes que detalla la forma en que estará estructurada la aplicación, reflejando la transformación de los elementos del modelo del diseño en términos de componentes, así como las dependencias entre ellos. Además se diseña y aplican las pruebas para comprobar el correcto funcionamiento de la aplicación.

Capítulo 1: Fundamentación Teórica de SIGE-Droid

En este capítulo se presenta la definición del marco teórico de la investigación. Se realiza el estudio de las funcionalidades y características de SIGE a tener en cuenta para la realización de SIGE-Droid. Además, se analizan aplicaciones desarrolladas en Android para determinar la forma en la que estas se conectan a bases de datos externas. Se aborda el estudio de la metodología, herramientas, tecnologías y lenguajes a utilizar en el desarrollo de SIGE-Droid.

1.1 Análisis de SIGE

El Sistema Integrado de Gestión Estadística (SIGE) es una aplicación web enriquecida basada en tecnologías libres. Surge con el objetivo de satisfacer las necesidades de automatización de los procesos estadísticos de la ONEI. Como sistema, posibilita la gestión de los metadatos que caracterizan a las empresas. Permite diseñar formularios y encuestas necesarios para la captación de la información estadística de las unidades de observación[2]. Además valida la información que ha sido recopilada en cada formulario o encuesta.

SIGE consta de una arquitectura modular formada por 6 módulos: Gestión de la Configuración (MGC), Gestión de Plantillas (MGP), Entrada de Datos (MED), Administración (MA), Herramientas (MH) y Seguridad (MSE). Además tiene integrado el Generador Dinámico de Reportes (GDR) el cual brinda un conjunto de herramientas para el diseño, generación y visualización de reportes.

MGC: Es el encargado de la gestión de las empresas y de los conceptos o metadatos necesarios para su caracterización y organización a nivel de país. En este sentido, los metadatos gestionados constituyen la base para el diseño de los formularios y las encuestas.

MGP: Es el encargado de automatizar el proceso de diseño de los formularios y encuestas necesarios para la captación de la información estadística en las unidades de observación; lo que constituye entrada para el procesamiento y obtención de informes y reportes estadísticos para la toma de decisiones estratégicas en cada uno de los ministerios.

[2] Las unidades de observación son las empresas, organismos, organizaciones e instituciones poseedoras de información de interés para la toma de decisiones.

MED: Su función se desarrolla sobre los productos finales del MGP, es decir, plantillas de formularios o encuestas en las que se definen el subconjunto de datos a captar de acuerdo a la naturaleza económica o social de la unidad de observación en cuestión. Este módulo es el responsable de la validación de la información que ha sido recopilada en cada formulario.

MA: Es el módulo que se encarga de la administración de las tareas de validación y la configuración del sistema.

MH: Tiene como objetivo fundamental la administración de los maestros, que no son más que construcciones matriciales en las que se relacionan clasificadores y clasificaciones. Permiten resumir las características de las unidades de observación a partir de una fecha determinada. Desde el punto de vista estadístico, su rol es clave dentro del procesamiento y obtención de la información, ya que permiten acotar el rango de búsqueda de la información captada.

MSE: Se encarga de la seguridad de la aplicación permitiendo restringir el acceso a la ejecución de las acciones del sistema.

GDR: Su objetivo principal es la realización de los reportes ejecutivos. Cubre el ciclo de vida de un reporte desde su creación, hasta su consulta y distribución.

Teniendo en cuenta que SIGE-Droid solo incluirá la gestión de seguridad y la visualización de reportes en SIGE, se estudia en profundidad el MSE y GDR.

Módulo de Seguridad

El área de trabajo del MSE está estructurada por las opciones que brinda el propio módulo, las cuales son: Gestión de Usuarios, Gestión de Roles y Perfiles, Autenticación LDAP y Trazas. De este módulo se estudiará en profundidad la gestión de usuarios y la gestión de roles, ya que estas son las funcionalidades que se tendrán en cuenta para la realización de SIGE-Droid.

Para la gestión de usuarios se muestra la interfaz principal en la cual se podrá adicionar usuario, modificar usuario, eliminar usuario, listar usuarios, así como asignar roles a los usuarios. A continuación se describe a detalle el proceso de gestión de usuarios:

- **Adicionar usuario**

Para la creación de un nuevo usuario se debe presionar el botón Adicionar en la interfaz para la gestión de usuarios. Seguidamente se mostrará una nueva interfaz para introducir los datos del usuario. Los datos se explican a continuación:

- Usuario: identificativo del usuario.
- Nombre: nombre del usuario.
- Apellidos: apellidos del usuario.
- Correo electrónico: correo electrónico del usuario.
- Contraseña: contraseña con la cual se autenticará el usuario
- Dirección: dirección ip desde la cual el usuario accederá al sistema.

- **Modificar usuario**

Para modificar un usuario previamente creado se debe buscar en el listado de usuarios y una vez encontrado hacer clic sobre el botón Editar. Se mostrará la interfaz correspondiente para actualizar la información del usuario. Ya introducida la nueva información del usuario, se deberá presionar el botón Aceptar para que dichos cambios sean correctamente actualizados en la aplicación y puedan ser vistos en el listado de los usuarios existentes.

- **Eliminar usuario**

Para eliminar un usuario de la aplicación se debe seleccionar el botón Eliminar correspondiente a dicho usuario. Una vez presionado este botón se muestra una ventana de advertencia donde se le notifica al usuario sobre la operación que está a punto de realizar. En caso de estar seguro de dicha operación se debe presionar el botón “Sí”, eliminándose el usuario del sistema.

- **Listar usuarios**

El sistema muestra el listado de usuarios adicionados automáticamente una vez mostrada la interfaz para la gestión de usuarios.

- **Asignar roles a usuarios**

Para poder asignar los roles a un usuario creado previamente se debe presionar el botón Perfil del usuario listado. El sistema debe mostrar la ventana Perfil del usuario en la Ventana Perfil del Usuario. En dicha

ventana se muestran los roles disponibles del lado izquierdo y los roles asignados del derecho, por lo que se deben seleccionar los roles disponibles y presionar el botón Pasar a roles seleccionados y/o viceversa. Finalmente debe presionar el botón Aceptar para que la asignación de roles al usuario sea guardada o presionar el botón Cancelar para que la misma sea cancelada.

Este módulo también permite la gestión de roles a cada uno de los usuarios del sistema. Incluye las funcionalidades crear rol, modificar rol, eliminar rol y listar roles.

- **Adicionar rol**

Para la creación de un nuevo rol se debe presionar el botón Adicionar en la interfaz para la gestión de roles. Seguidamente se mostrará una nueva interfaz para introducir el nombre y una posible descripción del rol que se desea crear. Una vez introducidos los datos necesarios para crear el nuevo rol se procede a presionar el botón Aceptar. Posteriormente se adicionará en la lista de roles el nuevo rol creado. De igual manera si desea cancelar la acción deberá presionar el botón Cancelar. Los datos necesarios para adicionar un rol se explican a continuación:

- Rol: nuevo rol.
- Descripción: breve descripción del rol que se desea crear.

- **Modificar rol**

Para modificar un rol previamente creado se debe presionar el botón Editar y automáticamente se visualizará una ventana que permite modificar la información de un rol existente. Ya introducida la nueva información del rol se deberá presionar el botón Aceptar para que dichos cambios sean correctamente actualizados en la aplicación, y pueden ser vistos en el listado de los roles existentes. De igual manera si desea cancelar la acción deberá presionar el botón Cancelar.

- **Eliminar rol**

Para eliminar un rol de la aplicación se debe seleccionar el botón Eliminar correspondiente a dicho rol. Luego se mostrará una ventana de advertencia donde se le notifica al usuario sobre la operación que está a punto de realizar. En caso de estar seguro de dicha operación se debe presionar el botón “Sí”. Esta acción eliminará el rol previamente seleccionado de la aplicación. De igual manera si desea cancelar la acción deberá presionar el botón Cancelar.

- **Listar roles**

El sistema muestra el listado de roles automáticamente una vez mostrada la interfaz para la gestión de roles.

Generador Dinámico de Reportes

GDR es una herramienta desarrollada para la realización de los reportes ejecutivos. En el contexto de SIGE es usada para la consulta de la información estadística. Cubre el ciclo de vida de un reporte desde su creación, hasta su consulta y distribución. Para cumplir con su objetivo presenta cinco módulos: Diseñador de reportes, Diseñador de consultas, Administrador de reportes, Diseñador de modelos y Visor de reportes.

Diseñador de reportes: Permite realizar el diseño de reportes personalizados desde la web. Estos reportes pueden ser salvados hacia la base de datos de SIGE para ser abiertos y utilizados en el momento que se necesite. Una vez confeccionados los reportes pueden ser visualizados los datos del mismo.

Diseñador de consultas: Permite la creación, edición y ejecución de consultas SQL, sin necesidad de conocer el gestor de base de datos ni el lenguaje de consulta estructurado (SQL).

Administrador de reportes: Muestra los reportes agrupados por categorías y las plantillas predefinidas en el módulo Diseñador de reportes. Es el encargado de mover, copiar, eliminar o renombrar los reportes, además de asignar o no permisos de visualización de reportes a los usuarios disponibles.

Diseñador de modelos: Es el encargado de conectarse a un origen de datos, ya sea a un gestor de base de datos SQLite, MySQL, Oracle, SQL Server o PostgreSQL para luego diseñar los modelos, que serán utilizados en la confección de los reportes.

Visor de reportes: Muestra un listado con todos los reportes agrupados por categorías, estas categorías son predefinidas por el usuario en el módulo Administrador de reportes.

Teniendo en cuenta que SIGE-Droid solo incluirá el proceso de visualización de reportes se estudia en profundidad el módulo Visor de Reportes.

El Visor de reportes es el encargado de mostrar la información almacenada en la base de datos mediante el uso de los reportes. Permite además la exportación de los mismos en los formatos: HTML, PDF, EXCEL y CSV, y filtrar los resultados de los reportes por medio de los campos que intervienen en el mismo. El área

de trabajo del Visor de reportes está estructurada por cinco funcionalidades, las cuales se describen a continuación:

- Vista previa del reporte: se visualiza una vista preliminar del reporte seleccionado, con una carga de datos inicial limitada a 10 filas.
- Filtro de reportes: se pueden realizar filtros a los datos del reporte para visualizar una información más exacta.
- Buscar reportes: permite buscar un reporte de la lista de reportes a partir de un criterio de búsqueda seleccionado por el usuario.
- Exportar reporte: permite exportar el reporte a un formato determinado (PDF, EXCEL, CSV y HTML). Además, se puede establecer el rango de filas que se desea tenga el reporte.
- Imprimir a PDF: permite exportar de manera rápida el reporte a formato PDF.

A partir del estudio de SIGE se realizó un análisis detallado de los procesos de gestión de usuario (Adicionar usuario, modificar usuario, eliminar usuario, listar usuarios y asignar rol) y gestión de roles (Adicionar rol, modificar rol, eliminar rol, listar roles) del MSE, funcionalidades que deberá contener SIGE-Droid. Esto permitirá que el equipo encargado de la administración del sistema pueda realizar su trabajo sin la necesidad de estar en sus puestos de trabajo. Partiendo del estudio de GDR se determinó que es necesario incluir la visualización de reportes almacenados en SIGE, así como su exportación. Esto permitirá que los directivos puedan acceder a la información deseada en cualquier momento y espacio sin la necesidad de imprimir cada reporte.

Solo se permitirá la exportación a formato PDF debido a que este formato mantiene la apariencia exacta del reporte obtenido en SIGE con las mismas fuentes e imágenes. También ofrece facilidades para su envío y recepción a través de la web pues es ligero. Otra de las ventajas de este formato es que el texto es tratado como una imagen haciendo más difícil su edición por terceros. Las acciones previamente descritas se realizarán directamente sobre la base de datos de SIGE en tiempo real y se llevarán a cabo desde dispositivos móviles con Sistema Operativo (SO) Android a través de la conexión inalámbrica.

1.2 Aplicaciones Android que se conectan a bases de datos externas

Para lograr una mayor comprensión sobre cómo SIGE-Droid podría interactuar con la base de datos (BD) de SIGE, se realiza un estudio sobre aplicaciones Android que se conectan a aplicaciones web y trabajan sobre su misma BD.

1.2.1 Facebook

La aplicación web Facebook es una red social la cual facilita la comunicación entre los usuarios, ya sea mensajes, videos e imágenes, entre otros. Sin embargo se puede acceder a dicha red social a través de un dispositivo móvil mediante el uso de una aplicación Android. Facebook cuenta con un API[3] (del inglés Application Programming Interface, en español Interfaz de Programación de Aplicaciones) que ofrece servicios web de tipo REST. Dichos servicios cuentan con las funcionalidades que ofrece el sistema y permite realizar operaciones sobre una BD alojada en los servidores de Facebook. Tanto la aplicación web de Facebook como su homóloga en Android, utilizan como puente la web para acceder a un solo repositorio de datos mediante el uso de servicios web (Facebook Developers, 2012).

1.2.2 Google Play

Google Play es una herramienta web que tiene como principal función proveer aplicaciones para móviles. Sin embargo existe una herramienta similar basada en sistema operativo Android que permite conectarse a los servidores de Google Play y descargar programas para dispositivos Android; además de brindar acceso a la información contenida en dicha aplicación web. Esta conexión entre ambas aplicaciones se realiza mediante un servicio web el cual provee las funcionalidades necesarias para realizar dichas actividades (Google Developers, 2013).

1.2.3 Gmail

Gmail es un servicio de correo electrónico al cual se puede acceder a través de una aplicación Android que lleva su mismo nombre. Estas acciones se realizan a través del uso de servicios web de tipo REST que facilitan la comunicación entre estas dos aplicaciones; ambas comparten una única BD (Google Developers, 2013).

A partir del estudio de las aplicaciones anteriores se identificó que estas utilizan los servicios web para conectarse a las BD externas y realizar acciones sobre ellas. Es por ello que la herramienta a desarrollar también deberá hacer uso de los servicios web para realizar las operaciones determinadas sobre la BD de SIGE.

[3] Un API es una interfaz de comunicación entre componentes de software. Su principal función consiste en proporcionar un conjunto de funciones de uso general.

1.3 Servicios web

Un servicio web es un aplicativo que facilita la interoperabilidad entre varios sistemas independientemente del lenguaje de programación o plataforma en que fueron desarrollados. Este debe tener una interfaz basada en un formato estándar como XML[4] (del inglés eXtensible Markup Language, en español Lenguaje de Marcas Extensible) o JSON[5] (del inglés JavaScript Object Notation, en español Objeto de Notación JavaScript) (Barry & Associates, 2015).Los servicios web pueden ser de dos tipos, SOAP O REST.

SOAP (del inglés Simple Object Access Protocol, en español Protocolo de Simple Acceso a Objetos) es un protocolo de mensajería XML extensible que forma la base de los servicios web proporcionando un mecanismo simple que permite el envío de mensajes entre aplicaciones. Un mensaje SOAP es una transmisión de una vía desde un emisor SOAP a un receptor SOAP; esto propicia que cualquier aplicación pueda participar en este intercambio como emisor o receptor, además de ser completamente independiente del protocolo de transporte subyacente (SCHAFFNER, 2015).

REST (del inglés Representational State Transfer, en español Trasferencia de Estado Representacional) es un estilo de arquitectura de software para sistemas hipermedias distribuido. REST se refiere estrictamente a una colección de principios para el diseño de arquitecturas en red. Estos principios resumen cómo los recursos son definidos y diseccionados. El término frecuentemente es utilizado en el sentido de describir a cualquier interfaz que transmite datos específicos de un domino sobre HTTP[6] (del inglés Hypertext Transfer Protocol, en español Protocolo de Transferencia de Hipertexto) sin una capa adicional (KIM, 2015).

SIGE cuenta con una capa de servicios web que ofrece las funcionalidades necesarias para la gestión de seguridad en el sistema mediante el uso de servicios de tipo SOAP. Tiene incluido además el GDR, el cual cuenta con su propia capa de servicios web de tipo REST, ofreciendo funcionalidades para la visualización de reportes. Las capas de servicios mencionadas anteriormente serán empleadas por SIGE-Droid.

De la capa de servicios de SIGE se consumirán los servicios que permiten la gestión de usuarios (adicionar usuario, modificar usuario, eliminar usuario, listar usuarios y asignar rol) y roles (adicionar rol, modificar rol, eliminar rol, listar roles). Mientras que de la capa de servicios de GDR se consumirán los referentes a la

[4] XML es un lenguaje de marcas utilizado para almacenar datos en forma legible.

[5] JSON es un formato de texto ligero usado para el intercambio de datos.

[6] HTTP es el protocolo de comunicación que permite las transferencias de información en internet.

obtención del catálogo de reportes, los reportes en sí, así como la exportación de los mismos a formato PDF.

1.4 ¿Por qué Android?

La herramienta será realizada sobre SO Android debido a que este utiliza licencia Apache, que según la FSF[7] (del inglés Free Software Fundation, en español Fundación de Software Libre) se encuentra entre las licencias consideradas software libre y consta de una amplia documentación. Además es uno de los SO para móviles que más aceptación tiene a nivel mundial y soporta gran variedad de dispositivos, aumentando considerablemente su campo de uso. Por otra parte se tuvo en cuenta que el 100 por ciento de los clientes de SIGE utilizan dispositivos móviles con SO Android. Para un mayor entendimiento de la selección se muestra la siguiente tabla comparativa entre distintos SO existentes:

Tabla 1: Comparación entre los principales sistemas operativos para móviles

Comparación					 Symbian
Licencia de software	Propietario	Software libre y abierto	Propietaria	Propietaria	Software libre
Plataforma de desarrollo	Mac	Windows, Mac y Linux	Windows	Windows y Mac	Windows, Mac y Linux
Interfaz personalizable	No	Si	Si	Si	Si
Fabricante único	Si	No	No	Si	No
Variedad de dispositivos	Modelo único	Muy alta	Baja	Baja	Muy alta
Usuarios a nivel mundial	42,59 %	47,45 %	2,25 %	0,98 %	3,22 %

[7] FSF es una organización no lucrativa dedicada a promover y defender el uso y desarrollo del software libre.

Usuarios de SIGE	0 %	100 %	0 %	0 %	0 %

1.5 Metodologías de desarrollo de software

Una metodología de desarrollo de software se refiere al entorno que se usa para estructurar, planificar y controlar el proceso de desarrollo de un sistema informático. Suele estar documentada y promovida por algún tipo de organización ya sea esta pública o privada (Boehm, 2012).

OpenUP

La metodología OpenUP (del inglés Open Unified Process, en español Proceso Unificado Abierto) es una metodología que aplica enfoques iterativos e incrementales dentro de un ciclo de vida estructurado. Utiliza una filosofía ágil que se enfoca en la naturaleza de colaboración para el desarrollo de software, basada en RUP (del inglés Rational Unified Process, en español Proceso Unificado Racional), que contiene el conjunto mínimo de prácticas que ayudan a un equipo de desarrollo de software a realizar un producto de alta calidad. El ciclo de vida del proyecto brinda a los interesados un mecanismo de supervisión y dirección para controlar los fundamentos del mismo, su ámbito, la exposición a los riesgos, el aumento de valor, entre otros aspectos. OpenUP estructura el proyecto en cuatro fases: inicio, elaboración, construcción y transición (Eclipse Foundation, 2013).

- Fase de Inicio: En esta fase las necesidades de cada participante del proyecto son tomadas en cuenta y plasmadas en objetivos del proyecto. Se definen para el proyecto: el ámbito, los limites, el criterio de aceptación, los casos de uso críticos, una estimación inicial del coste y un boceto de la planeación (Eclipse Foundation, 2013).
- Fase de Elaboración: En esta fase se realizan tareas de análisis del dominio y definición de la arquitectura del sistema. Se debe elaborar un plan de proyecto, estableciendo requisitos y arquitectura estables. Al final de la fase se debe tener una definición clara y precisa de los casos de uso, actores, la arquitectura del sistema y un prototipo ejecutable (Eclipse Foundation, 2013).
- Fase de Construcción: En esta fase todos los componentes y funcionalidades del sistema que falten por implementar son realizados, probados e integrados. Los resultados obtenidos en forma de incrementos ejecutables deben ser desarrollados de la forma más rápida posible sin dejar de lado la calidad de lo desarrollado (Eclipse Foundation, 2013).

- Fase de Transición: Esta fase corresponde a la introducción del producto en la comunidad de usuarios, cuando el producto está lo suficiente maduro. Posteriormente se presenta el producto a los usuarios finales donde se realizan las pruebas determinadas por el cliente. En función a la respuesta obtenida de parte de los clientes puede ser necesario realizar cambios en las entregas finales o implementar alguna funcionalidad más solicitada por la mayoría (Eclipse Foundation, 2013).

Se escoge esta metodología debido a que es apropiada para proyectos pequeños permitiendo disminuir las probabilidades de fracaso. También permite detectar errores tempranos a través de un ciclo iterativo, pues tiene un enfoque centrado al cliente y con iteraciones cortas. De igual manera proporciona una comprensión detallada del proyecto, beneficiando a clientes y desarrolladores sobre productos a entregar y su formalidad.

1.6 Lenguaje de modelado

El modelado es el diseño de las aplicaciones de software antes de codificar. Es una parte esencial de los grandes proyectos de software, y útil para proyectos medianos e incluso pequeños. Un modelo juega el papel análogo en el desarrollo de software (ROMERO, 2015).

UML 2.0

El UML (del inglés Unified Modeling Language, en español Lenguaje de Modelado Unificado) es un lenguaje de modelado visual estándar destinado a ser utilizado para el modelado de procesos de negocio; y para el análisis, diseño e implementación de sistemas basados en software. UML es un lenguaje común para los analistas, arquitectos y desarrolladores de software, utilizado para describir, especificar, diseñar nuevos procesos de negocio. UML proporciona valiosa ayuda a los profesionales visualizando, comunicando y aplicando sus diseños (ARLOW, 2016).

Se utiliza UML 2.0 ya que este proporciona ventajas en la representación del ciclo de vida de un software y de los artefactos específicos de OpenUP. Además, posibilita la comunicación sencilla y rápida entre el programador y los clientes del software que se desarrolla; brindando la posibilidad de que estos últimos puedan expresar su conformidad con el producto o las nuevas mejoras que desean ver introducidas.

1.7 Herramienta CASE

Son un conjunto de programas y ayudas a los analistas, ingenieros de software y desarrolladores, durante todos los pasos del ciclo de vida de desarrollo de un software, permitiendo la creación de todo tipo de diagramas (Jacobson, 2011).

Visual Paradigm for UML 8.0

Visual Paradigm for UML Standard Edition (VP-UML SE) es una plataforma de modelado diseñada para apoyar a los arquitectos de sistemas, desarrolladores y diseñadores de software. Además acelera el proceso de análisis y diseño de aplicaciones complejas facilitando la construcción y visualización de diferentes tipos de diagramas UML dentro de los que se incluyen diagramas de casos de uso, de clases, entidad relación y otros (Rumbaugh, 2009).

Se escoge la herramienta Visual Paradigm For UML 8.0 por soportar el ciclo de vida completo del proceso de desarrollo del software a través de la representación de todo tipo de diagramas. De igual manera ofrece un paquete completo necesario para la captura de requisitos, la planificación del software, modelado de clases y modelado de datos. Proporciona a los desarrolladores una plataforma que les permite diseñar un producto de forma rápida y con la calidad requerida. Además consta de una licencia gratuita y comercial.

1.8 Lenguaje de programación

Un lenguaje de programación es un lenguaje artificial que puede ser usado para controlar el comportamiento de una máquina, especialmente una computadora. Estos se componen de un conjunto de reglas sintácticas y semánticas que permiten expresar instrucciones que luego serán interpretadas (ALEGSA, 2009).

Java 7

Java es un lenguaje de programación multiplataforma orientado a objetos. Además, cuenta con una máquina virtual que permite ejecutar cualquier código compilado en Java. Actualmente puede ser usado de forma gratuita (Holgate, 2012).

Se eligió Java 7 como lenguaje de programación debido a que es el lenguaje oficial para la programación en Android. Además, presenta una amplia variedad de documentación, lo cual facilita el trabajo de los desarrolladores.

1.9 Entorno de desarrollo integrado

Un IDE (del inglés Integrated Development Environment, en español Entorno de Desarrollo Integrado) es un programa informático compuesto por un conjunto de herramientas de programación. Puede dedicarse en exclusiva a un sólo lenguaje de programación o bien, poder utilizarse para varios (InformaticaSONs, 2011).

Android Studio 1.5

Android Studio es un IDE, basado en IntelliJ IDEA[8](del inglés Integrated Development Environment for Android, en español Entorno de Desarrollo Integrado para Android) de la compañía JetBrains. Es un entorno dedicado exclusivamente a la programación de aplicaciones para dispositivos Android. Utiliza una licencia de software libre Apache 2.0, está programado en Java y es multiplataforma (academiaandroid, 2014).

Se eligió Android Studio 1.5 como IDE de desarrollo debido a que es el IDE oficial de la plataforma Android, además es de código abierto y gratuito. Por lo cual es la herramienta idónea para el desarrollo de la aplicación.

1.10 Sistema Gestor de Base de Datos

Un Sistema Gestor de Bases de Datos (SGBD) es una colección de programas cuyo objetivo es servir de interfaz entre la BD, el usuario y la aplicación. Se compone de un lenguaje de definición de datos, de un lenguaje de manipulación de datos y de un lenguaje de consulta. Un SGBD permite definir los datos a distintos niveles de abstracción y manipular dichos datos, garantizando la seguridad e integridad de los mismos (Editorial McGraw-Hill, 2012).

SQLite 3.7.4

Es una biblioteca escrita en leguaje C que implementa un SGBD. El mismo soporta tablas, índices y vistas que no necesitan de un servidor para su utilización. Su código es de dominio público y de libre uso. Es multiplataforma y sus BD pueden ser fácilmente portadas sin ninguna configuración o administración (Rómmel, 2012).

[8] IntelliJ IDEA es un IDE para el desarrollo de programas informáticos.

Se seleccionó SQLite como SGBD debido a que es de código abierto y multiplataforma. Además, SQLite es una tecnología cómoda para los dispositivos móviles debido a su simplicidad, rapidez y usabilidad. Las características citadas anteriormente hacen a SQLite el SGBD idóneo para emplear en el desarrollo de la aplicación.

1.11 Conclusiones del capítulo

Luego del análisis realizado en profundidad al MSE y GDR se llegó a la conclusión de que SIGE-Droid debe permitir la gestión de usuarios y roles para facilitar el trabajo de los administradores de SIGE; así como la visualización y exportación de reportes para facilitar el acceso a los mismos desde cualquier espacio de la institución, en el momento y espacio deseado, sin la necesidad de imprimirlos. A partir del estudio de aplicaciones en Android que operan sobre BD externas se identificó como característica fundamental que dichas operaciones deben ser realizadas mediante el uso de servicios web. Es por ello que se utiliza la capa de servicios de SIGE, la cual posee los servicios de tipo SOAP necesarios para la gestión de usuarios y roles. Por su parte el uso de la capa de servicios de GDR permite la visualización de reportes en SIGE mediante los servicios de tipo REST que ofrece la misma. Se seleccionó como metodología de desarrollo de software OpenUP, la cual guiará todo el proceso de creación de la herramienta propuesta. Se hizo un estudio y selección de las herramientas y tecnologías que serán usadas durante el proceso de desarrollo de la aplicación. Se seleccionó UML 2.0 como lenguaje de modelado y Visual Paradigm 8.0 como herramienta CASE. Como lenguaje de programación Java 7, haciendo uso del IDE de desarrollo Android Studio 1.5 y utilizando SQLite 3.7.4 como SGBD. Cada una de estas herramientas y tecnologías posee características que las hacen idóneas para el trabajo con ellas durante el proceso de desarrollo de la aplicación.

Capítulo 2: Análisis y diseño de SIGE-Droid

En este capítulo se define el modelo de dominio y se identifican, especifican y describen los requisitos funcionales y no funcionales de la aplicación. Además se describen los actores y casos de uso representados en el diagrama de casos de uso del sistema. En el proceso de diseño se brinda un acercamiento a la implementación de la aplicación. Se construye para ello los diagramas de interacción y el modelo de clases del diseño siguiendo el estilo arquitectónico y los patrones de diseño seleccionados. Esto proporcionará la base para proceder a la etapa de implementación.

2.1 Modelo de Dominio

El modelo de dominio puede utilizarse para capturar y expresar el entendimiento ganado en un área bajo análisis como paso previo al diseño de un sistema, ya sea de software o de otro tipo. El mismo no contiene conceptos propios de un sistema de software, sino de la propia realidad física. Es utilizado por el analista como un medio para comprender el negocio al cual el sistema va a servir (ESTRADA, 2015). A continuación se muestra el modelo de dominio de SIGE-Droid.

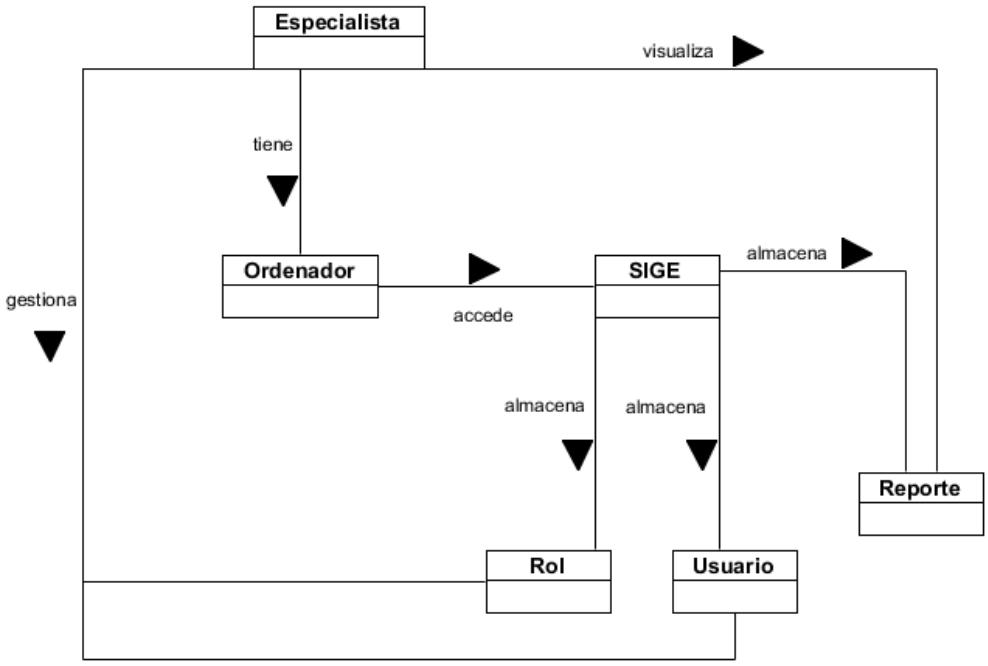


Fig. 1: Modelo de dominio de SIGE-Droid.

Descripción de las clases del dominio

Especialista: Es el encargado de interactuar con el sistema, ya sea el gestionar los usuarios del sistema y los roles que desempeñan. También tiene acceso a visualizar los reportes del sistema.
Ordenador: Es el ordenador o computadora a través del cual el especialista accede al sistema para interactuar con el mismo.
SIGE: Es el sistema informático a través del cual el especialista realiza las acciones previamente descritas.
Usuario: Constituye la especificación de los usuarios del sistema.
Rol: Constituye la especificación de los roles que desempeñan los usuarios en el sistema.
Reporte: Representación física de los reportes en el sistema.

2.2 Propuesta de solución

La herramienta a desarrollar va a permitir realizar la gestión de seguridad y la administración de reportes en SIGE haciendo uso de dispositivos móviles con SO Android, llevando por nombre SIGE-Droid. Para acceder a las funcionalidades brindadas por dicha herramienta el usuario debe estar previamente autenticado. Para ello el usuario debe especificar con anterioridad la dirección del servidor al cual debe conectarse mediante la gestión de las conexiones. Existirán dos tipos de usuario: el administrador (que poseerá permisos para la gestión de usuarios, roles y asignación de roles a usuarios) y el visor de reportes (que poseerá permisos para listar, visualizar y exportar a PDF los reportes almacenados en SIGE). Los usuarios del sistema podrán hacer uso de las funcionalidades asociadas a la gestión de seguridad o a la visualización de reportes, nunca ambas. Las funcionalidades previamente descritas son fundamentales para el cumplimiento de la problemática y los objetivos planteados. Para dar cumplimiento a dichas acciones se hará uso de las funcionalidades que brindan la capa de servicios de SIGE y GDR respectivamente.

2.3 Especificación de los requisitos del sistema

Los requisitos para un sistema de software determinan lo que hará el sistema y definen las restricciones de su operación e implementación. Se pueden clasificar en: funcionales y no funcionales. Partiendo de la descripción de las clases representadas en el modelo de dominio y las necesidades del cliente, fueron determinados los requisitos funcionales (RF) y no funcionales (RNF) del sistema (ZAPATA, 2014).

2.3.1. Requisitos funcionales

Son declaraciones de las acciones que debe realizar el sistema, de la manera en que éste debe reaccionar a entradas particulares y de cómo se debe comportar en situaciones particulares (JARAMILLO, 2013). Se identificaron los siguientes RF:

RF1: Adicionar conexión

Descripción: La aplicación debe permitir adicionar una nueva conexión.
Entrada: Nombre (Alfanumérico de 0 a 25 caracteres), Dirección SIGE (Alfanumérico de 0 a 20 caracteres), Dirección GDR (Alfanumérico de 0 a 20 caracteres), Puerto (Alfanumérico de 0 a 4 caracteres).
Salida: No Aplica (NA).

RF2: Modificar conexión

Descripción: La aplicación debe permitir editar una conexión previamente seleccionada.
Entrada: Nombre (Alfanumérico de 0 a 25 caracteres), Dirección SIGE (Alfanumérico de 0 a 20 caracteres), Dirección GDR (Alfanumérico de 0 a 20 caracteres), Puerto (Alfanumérico de 0 a 4 caracteres).
Salida: (NA).

RF3: Eliminar conexión

Descripción: La aplicación debe permitir eliminar una conexión previamente seleccionada.
Entrada: Id_conexion (Numérico de 0 a 9 caracteres).
Salida: (NA).

RF4: Listar conexión

Descripción: La aplicación debe mostrar todas las conexiones existentes en la aplicación.
Entrada: NA.
Salida: Listado de conexiones existentes en la aplicación.

RF5: Autenticar usuario

Descripción: La aplicación requiere de autenticación para acceder a los servicios que ofrece.
Entrada: Usuario (Alfanumérico de 0 a 25 caracteres), Contraseña (Alfanumérico de 0 a 30 caracteres).

Salida: (NA).

RF6: Cerrar sesión

Descripción: La aplicación debe permitir cerrar la sesión una vez el usuario lo determine.

Entada: (NA).
Salida: (NA).

RF7: Adicionar usuario

Descripción: La aplicación debe permitir adicionar un nuevo usuario.
Entrada: Nombre (Alfanumérico de 0 a 25 caracteres), Apellido (Alfanumérico de 0 a 25 caracteres), Usuario (Alfanumérico de 0 a 25 caracteres), Correo (Alfanumérico de 0 a 25 caracteres), Contraseña (Alfanumérico de 0 a 30 caracteres), Dirección IP (Alfanumérico de 0 a 20 caracteres).
Salida: (NA).

RF8: Modificar usuario

Descripción: La aplicación debe permitir editar un usuario previamente seleccionado.
Entrada: Nombre (Alfanumérico de 0 a 25 caracteres), Apellido (Alfanumérico de 0 a 25 caracteres), Usuario (Alfanumérico de 0 a 25 caracteres), Correo (Alfanumérico de 0 a 25 caracteres), Contraseña (Alfanumérico de 0 a 30 caracteres), Dirección IP (Alfanumérico de 0 a 20 caracteres).
Salida: (NA).

RF9: Eliminar usuario

Descripción: La aplicación debe permitir eliminar un usuario previamente seleccionado.
Entrada: Id_usuario (Numérico de 0 a 9 caracteres).
Salida: (NA).

RF10: Listar usuario

Descripción: La aplicación debe mostrar todos los usuarios existentes.
Entrada: (NA).
Salida: Listado de usuarios del sistema.

RF11: Adicionar rol

Descripción: La aplicación debe permitir adicionar un nuevo rol.
Entrada: Rol (Numérico de 0 a 9 caracteres), Descripción (Numérico de 0 a 25 caracteres).
Salida: (NA).

RF12: Modificar rol

Descripción: La aplicación debe permitir editar un rol previamente seleccionado.
Entrada: Rol (Numérico de 0 a 9 caracteres), Descripción (Numérico de 0 a 25 caracteres).
Salida: (NA).

RF13: Eliminar rol

Descripción: La aplicación debe permitir eliminar un rol previamente seleccionado.
Entrada: Id_rol (Numérico de 0 a 9 caracteres).
Salida: (NA).

RF14: Listar rol

Descripción: La aplicación debe mostrar todos los roles existentes.
Entada: (NA).
Salida: Listado de roles del sistema.

RF15: Asignar rol

Descripción: La aplicación debe permitir asignar rol a determinado usuario.
Entrada: Id_usuario (Numérico de 0 a 9 caracteres), Id_rol (Numérico de 0 a 9 caracteres).
Salida: (NA).

RF16: Listar reporte

Descripción: La aplicación debe brindar la posibilidad de mostrar el listado de reportes del sistema.
Entada: (NA).
Salida: Listado de reportes existentes en la aplicación.

RF17: Mostrar Reportes

Descripción: La aplicación debe brindar la posibilidad de acceder a los reportes detalladamente.
Entrada: Id_reporte (Numérico de 0 a 9 caracteres).
Salida: Visualización de los datos del reporte.

RF18: Exportar a PDF

Descripción: La aplicación debe brindar la posibilidad de exportar el reporte seleccionado a formato PDF.
Entrada: Id_reporte (Numérico de 0 a 9 caracteres).
Salida: Almacenamiento del reporte en formato PDF en la memoria interna del dispositivo.

2.3.2. Requisitos no funcionales

Son restricciones de los servicios o funciones ofrecidos por el sistema. Incluyen restricciones de tiempo, sobre el proceso de desarrollo y estándares. Los RNF a menudo se aplican al sistema en su totalidad (scribd, 2015). Se identificaron los siguientes RNF:

RNF1: Usabilidad

La aplicación debe ajustarse al entorno donde será aplicada y utilizar el castellano como lenguaje base. Debe ser capaz de aprovechar la pequeña infraestructura de red inalámbrica creada por la institución para su uso. La sencillez y claridad de sus acciones debe permitir su utilización por cualquier usuario con conocimientos básicos en la tecnología móvil, brindándole la capacidad de trabajar de manera eficaz.

RNF2: Seguridad

La información estará protegida contra accesos no autorizados utilizando mecanismos de validación que puedan garantizar el cumplimiento de esto: usuario, contraseña y nivel de acceso, de manera que cada usuario pueda tener disponible solamente las opciones relacionadas con su actividad garantizando así la confidencialidad.

RNF3: Diseño e Implementación

- El sistema deberá ser implementado en el lenguaje de programación Java.
- Se utilizará el marco de trabajo de desarrollo Android en su versión 4.0.

- Se empleará la herramienta de desarrollo Android Studio 1.5 y el SGDB SQLite 3.7.4.

RNF4: Software

Requisitos mínimos para el dispositivo móvil:

- Soporte para conexiones WIFI.
- Sistema operativo Android 4.0 o mayor.

RNF5: Hardware

Requisitos mínimos para el dispositivo móvil:

- Capacidad disponible de 20 MB en la memoria del teléfono para el almacenamiento, instalación y ejecución de la herramienta.

2.4 Modelo de Caso de Uso del sistema

El modelo de Casos de Uso (CU) describe las funcionalidades propuestas del nuevo sistema, sirven para especificar la comunicación y el comportamiento de un sistema mediante su interacción con los usuarios (AMPUERO, 2015). El modelo de CU está compuesto por actores, CU y la relación que existe entre ellos.

Casos de Uso

Los CU son artefactos narrativos que describen, bajo la forma de acciones y reacciones, el comportamiento de un sistema desde el punto de vista del usuario. Por lo tanto, establece un acuerdo entre clientes y desarrolladores sobre las condiciones y posibilidades (requisitos) que debe cumplir el sistema (Software, 2014).

Actores del sistema

Cada trabajador que tiene actividades a automatizar es un candidato a actor del sistema. Si algún actor va a interactuar con el sistema, entonces también será un actor del sistema (Software, 2014).

Tabla 2: Descripción de los actores del sistema.

Actor	***Objetivos***
Usuario	Es el encargado de autenticarse en la herramienta y gestionar la conexión a la BD.
Administrador	Es el encargado de gestionar usuarios y roles en el sistema.
Visor de reportes	Es el encargado de administrar reportes en el sistema.

2.4.1 Diagrama de CU de la herramienta

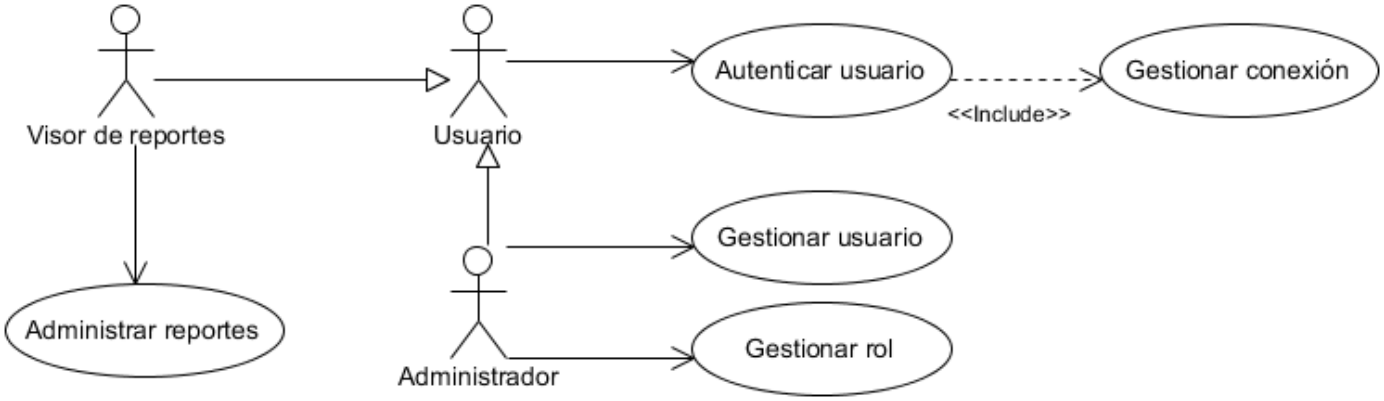


Fig. 2: Diagrama de CU de SIGE-Droid

Patrones de caso de uso

Para la implementación de la herramienta propuesta se identificaron 18 RF agrupados en los 5 CU que se muestran en la Fig. 2 teniendo en cuenta los siguientes patrones de CU:

CRUD Completo

CRUD (del inglés Create, Read, Update, Delete; en español Crear, Leer, Actualizar, Modificar). El patrón CRUD Completo consiste en un CU que permite modelar las diferentes operaciones para gestionar una entidad de información, tales como crear, leer, cambiar y dar de baja. Este patrón es usado cuando todas

las operaciones contribuyen al mismo valor de negocio y todas son cortas y simples. Ejemplo de ello se evidencia en los CU Gestionar usuario y Gestionar rol representados en el diagrama de CU del sistema.

CRUD Parcial

Este patrón es usado cuando alguna de las alternativas del CU es más significativo, muy largo, o mucho más complejo que el patrón completo. Ejemplo de ello se evidencia en el CU Administrar reportes representado en el diagrama de CU del sistema.

Inclusión concreta

Se incluye una relación del CU base al de inclusión. Es imprescindible que se ejecute el último para la ejecución del primero, puesto que existe una dependencia entre ambos. Ejemplo de ello se evidencia en los CU Autenticar usuario y Gestionar conexión, debido a que para autenticarse en el sistema es necesario configurar previamente la conexión a la BD.

Múltiples actores rol común

A un CU ingresan más de dos actores y estos desempeñan un rol común en el sistema. Ejemplo de ello se evidencia en los actores Administrador y Visor de reportes, los cuales tienen en común el rol de Usuario para realizar la gestión de conexiones y autenticación en el sistema.

2.4.2 Descripción textual del CU Gestionar usuario

Tabla 3: Descripción del CU "Gestionar usuario."

Objetivo	Gestionar los usuarios
Actores	Administrador (Inicia)
Resumen	El CU se inicia cuando el Administrador va a realizar alguna de las siguientes operaciones: Crear un nuevo usuario, Modificar usuario, Eliminar usuario, Listar usuarios existentes. El CU finaliza cuando el

	Administrador realiza alguna de las acciones mencionadas anteriormente.
Complejidad	Media
Prioridad	Crítico
Precondiciones	La aplicación SIGE-Droid debe estar previamente instalada en el dispositivo móvil y el Administrador debe estar previamente autenticado en el sistema.
Postcondiciones	En dependencia de la acción del Administrador: Se adiciona un nuevo usuario. Se modifica un usuario existente. Se elimina un usuario existente. Se muestra un listado de los usuarios existentes.

Flujo de eventos		
	Actor	**Sistema**
1.	Selecciona la opción **"Gestionar usuario".**	
2.		Muestra el listado de los usuarios existentes y se activan las opciones correspondientes a la gestión de usuarios. El Usuario puede: a. Adicionar un nuevo usuario: ir a la sección "Adicionar usuario" b. Modificar datos de un usuario: ir a la sección "Modificar usuario" c. Eliminar usuario: ir a la sección "Eliminar usuario"

		d. Listar usuarios: Los usuarios se listan automáticamente al iniciarse el CU.
Sección 1: "Adicionar Usuario**"**		
Flujo básico < Adicionar Usuario **>**		
	Actor	**Sistema**
1.	Selecciona el botón con el signo **"+"**.	
2.		Muestra la interfaz para crear un nuevo usuario, solicitando los siguientes datos: Nombre Apellido Usuario Correo Contraseña Dirección IP
3.	Introduce los datos correspondientes y selecciona la opción **"Aceptar"**.	
4.		Valida los datos introducidos.
5.		Registra el nuevo usuario, recarga el listado de usuarios y muestra el mensaje "Usuario adicionado correctamente", finalizando así el CU.
Flujos alternos		

Nº 4.a <Validación>		
	Actor	**Sistema**
1.		Muestra los campos con errores de validación subrayados en rojo.
2.	Se posiciona sobre el campo que tiene errores en los datos.	
3.		Muestra el mensaje de error de acuerdo al campo donde se posicione el Administrador.
4.	Corrige los datos y presiona la opción **"Aceptar"**.	
Flujos alternos		
Nº 5.a <Cancelar**>**		
1.	Decide no adicionar un usuario y presiona la opción **"Cancelar"**	
2.		Cancela la operación y cierra la interfaz correspondiente finalizando así el CU.
Sección 2: "Modificar Usuario**"**		
Flujo básico < Modificar Usuario **>**		
	Actor	**Sistema**

➢	Selecciona con un clic sostenido un usuario del listado de usuarios que se muestra y selecciona la opción Editar.	
➢		Muestra una interfaz con los datos del usuario seleccionado previamente.
➢	Modifica los datos del usuario que desea actualizar de la interfaz mostrada previamente por el sistema y selecciona la opción **"Aceptar"**.	
➢		Valida los campos editados.
➢		Actualiza el usuario, recarga el listado de usuarios para mostrar la actualización realizada y muestra el mensaje "Usuario actualizado correctamente", finalizando así el CU.
Flujos alternos		
Nº 4.a <Validación**>**		
➢		Muestra los campos con errores de validación.
➢	Se posiciona sobre el campo que tiene errores en los datos.	
➢	Corrige los datos y presiona la opción **"Aceptar"**.	

Flujos alternos		
Nº 5.a < Cancelar**>**		
➢	El Administrador decide no modificar el usuario y presiona la opción **"Cancelar".**	
➢		La aplicación cancela la operación y cierra la interfaz correspondiente finalizando así el CU.
Sección 3: "Eliminar Usuario**"**		
Flujo básico < Eliminar Usuario **>**		
	Actor	**Sistema**
➢	El administrador selecciona con un clic sostenido un usuario del listado de usuarios que se muestra y selecciona la opción **"Eliminar".**	
➢		La aplicación brinda una alerta: "¿Está seguro que desea eliminar el usuario seleccionado?".
➢	El Administrador selecciona la opción **"Aceptar".**	
➢		El sistema elimina el usuario, recarga el listado de usuarios y muestra el mensaje "Usuario eliminado correctamente", finalizando así el CU.

Flujos alternos		
Nº 3.a <Selección Cancelar**>**		
1.	El Administrador selecciona la opción **"Cancelar"**.	
2.		El sistema cierra la alerta finalizando así el CU.

2.5 Patrones arquitectónicos

Los patrones arquitectónicos proporcionan un marco de referencia necesario para guiar la construcción de un software, permitiendo a los programadores, analistas y todo el conjunto de desarrolladores del software compartir una misma línea de trabajo y cubrir todos los objetivos y restricciones de la aplicación. Dichos patrones son considerados el nivel más alto en el diseño de la arquitectura de un sistema puesto que establecen la estructura, funcionamiento e interacción entre las partes del software (ESCALANTE, 2016).

Se utiliza como patrón arquitectónico el MVP (del inglés Model View Presenter, en español Modelo Vista Presentador), derivado del conocido MVC (del inglés Model View Controller, en español Modelo Vista Controlador). En Android, las clases bases que se utilizan para implementar aplicaciones son Fragment[9] (en español Fragmento) y Activity[10] (en español Actividad). A nivel de desarrollo, se podría considerar que estos conforman la Vista, ya que tienen los métodos que se encargaran de diseñar los elementos que posteriormente serán mostrados al usuario. Sin embargo, una Activity o un Fragment, si no se gestionan bien, pueden acabar teniendo toda la responsabilidad, Vista y lógica de negocio. A nivel estructural, esto no sería una buena práctica, ya que, al no separar las responsabilidades, el acoplamiento, la legibilidad y otros principios de calidad del software no serían respetados. Al usar MVP, se consigue que las Activities y Fragments queden completamente desacoplados de la parte de negocio, permitiendo que las mismas

[9] Un fragmento es una sección de interfaz de usuario embebida dentro de una actividad, el cual permite optimizar el diseño de la interfaz. Los fragmentos son vistos como mini actividades contenidas dentro de una actividad anfitriona, manejando su propio diseño.

[10] Las Actividades son las pantallas que conforman una aplicación, las cuales están conformadas por dos partes: la parte lógica y la parte gráfica. La parte lógica es una archivo .java que es la clase que se crea para poder manipular, interactuar y colocar el código de dicha pantalla. La parte gráfica es un XML que tiene todos los elementos visuales que se observan en dicha pantalla.

puedan ser reutilizadas con gran facilidad. Además de dotar de una mayor legibilidad del código, el patrón MVP independiza la vista de la forma de conseguir esos datos. Divide la aplicación en al menos tres capas distintas, permitiendo probar cada una de ellas de forma independiente:

- **Capa Vista:** Incluye las interfaces de usuario (Actividades, recursos, layouts, entre otros), y aquellos artefactos de Android sin interfaz gráfica (utilidades visuales). Las clases contenidas en esta capa son las responsables de la gestión de los componentes visuales del sistema. Un ejemplo de ello es la clase AdicionarUsuarioActivity.java. En los formularios (layouts), se incluyen los archivos XML, ejemplos de ello son las clases Activity_adicionar_usuario.xml y Activity_gestionar_usuario.xml.
- **Capa Presentador:** Incluye las clases del negocio, que hacen de puente entre la vista y el modelo, incluyendo clases con métodos que alberguen la lógica de la herramienta. Siempre los artefactos de la capa Vista invocarán a las clases albergadas en esta capa, nunca accederán al modelo directamente. Las clases presentadoras contenidas en esta capa de negocio son las que albergan el control y gestión sobre todo el contenido referente a su CU, como es el caso de la clase GestionarUsuarioActivity.
- **Capa Modelo:** Incluye la definición lógica del modelo de datos a utilizar. Contiene la clase entidad Usuario y las clases encargadas de invocar a los servicios web, como es el caso de las clases AdicionarUsuarioWSTask, ModificarUsuarioWSTask, EleminarUsuarioWSTask y ListarUsuarioWSTask.

2.6 Modelo del Diseño

El modelo de diseño es encargado de determinar cómo serán implementados los requisitos, así como las restricciones que se le suponen. Sus principales objetivos se basan en transformar los requisitos en un diseño del sistema, evolucionar hacia una arquitectura sólida y adaptar el diseño para que se ajuste al entorno de implementación del cual depende, así como del lenguaje de programación (Fernández, 2010).

2.6.1. Diagrama de clases del diseño

Un diagrama de clases del diseño (DCD) es una representación concreta de los métodos y atributos de cada una de las clases del sistema que se deben implementar. Para mostrar de forma simple la colaboración y las tareas de cada una de ellas en relación al sistema que conforman. Estos diagramas representan la parte estática del sistema (SCOTT, 2014). El siguiente DCD contiene las principales clases con sus respectivos métodos y atributos para el CU Gestionar usuario.

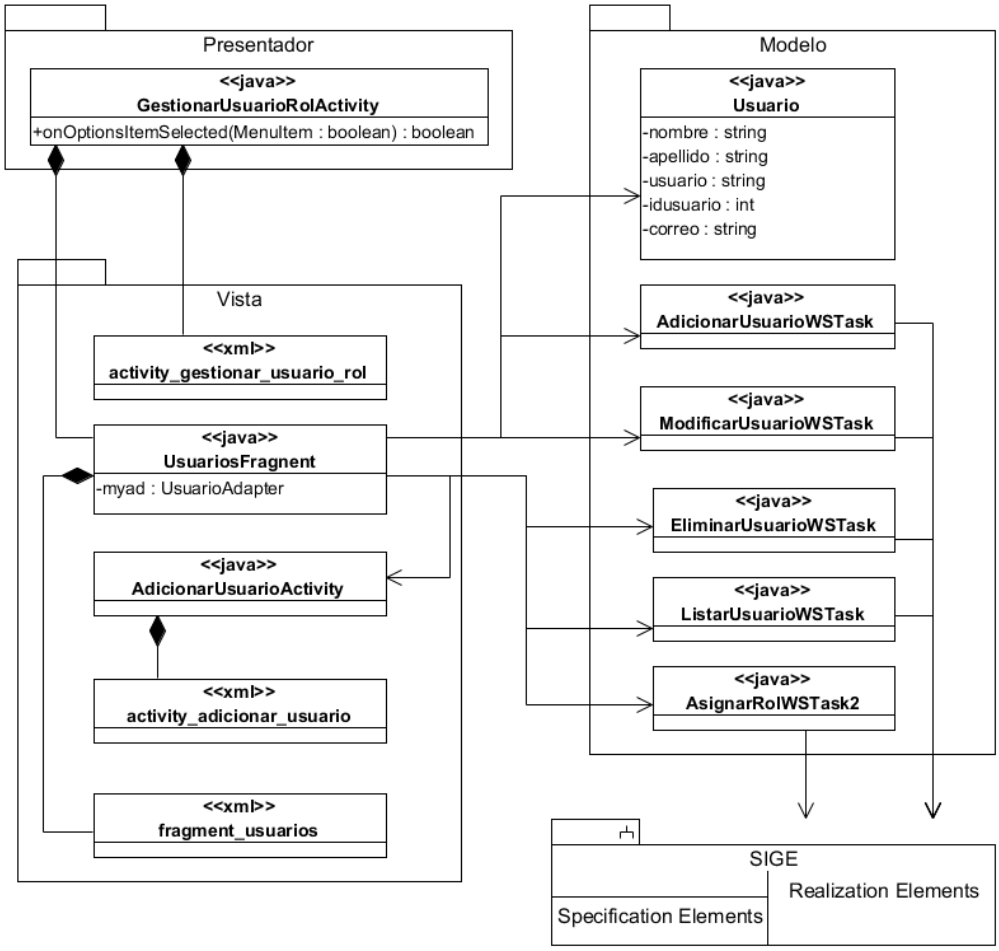


Fig. 3: DCD del CU Gestionar usuario.

La capa Presentador contiene la clase GestionarUsuarioRolActivity.java que es la encargada de ejecutar las acciones correspondientes al CU Gestionar usuario. Dicha clase está compuesta por las clases activity_gestionar_usuario_rol.xml y UsuariosFragment.java. La clase activity_gestionar_usuario_rol.xml contiene todos los componentes visuales de la clase presentadora, mientras que esta última realiza llamadas a la clase UsuariosFragment.java cuando el usuario decide realizar alguna acción. La clase UsuariosFragment.java está compuesta por fragment_usuarios.xml, la cual contiene el componente visual de tipo Lista, posibilitando mostrar el listado de usuarios.

En caso de que el administrador desee adicionar un nuevo usuario, este selecciona el botón adicionar el cual hace una llamada al método AdicionarUsuario de la clase AdicionarUsuarioActivity.java; siendo esta la encargada de capturar los datos introducidos por el administrador en la clase visual activity_adcionar_usuario.xml. Una vez obtenida la información necesaria, la clase UsuariosFragment.java

hace llamados a las clases encargadas de consumir los servicios web (AdicionarUsuarioWSTask.java, ModificarUsuarioWSTask.java, EliminarUsuarioWSTask.java, AsignarRolWSTask.java y ListarUsuarioWSTask.java), quienes se encargarán de realizar las acciones en la BD de SIGE.

Los servicios web a consumir se encuentran en la capa de servicios que brinda la herramienta SIGE. La clase entidad Usuario.java contiene la información necesaria para la creación de los objetos de tipo usuario, los cuales son mostrados posteriormente en la lista de usuarios.

2.6.2 Diagrama de secuencia

Los diagramas de secuencia permiten explicar gráficamente las operaciones que un actor solicita al sistema mediante el uso de escenarios. Estos permiten obtener una descripción gráfica de las interacciones entre el actor y el sistema, así como las operaciones que las mismas originan. Además incluyen los objetos y clases que se usan para implementar el escenario y mensajes intercambiados entre los objetos (VALENCIA, 2015). A continuación se representa el escenario “Adicionar usuario” perteneciente al CU Gestionar usuario.

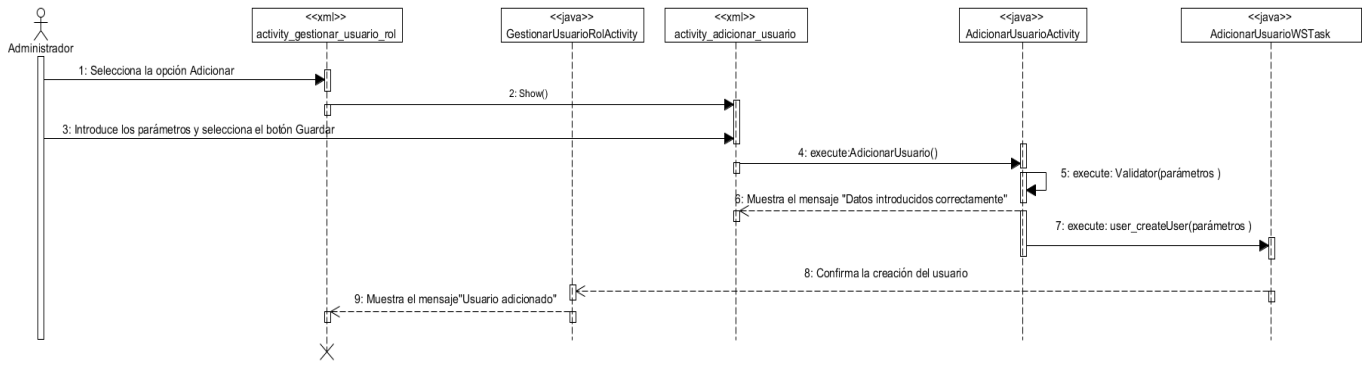


Fig. 4: Diagrama de secuencia del escenario Adicionar usuario.

Este escenario es inicializado una vez que el administrador selecciona la opción adicionar. La herramienta muestra la interfaz para adicionar que está compuesta por las clases activity_adicionar_usuario.xml donde el usuario introduce los datos para la creación de un nuevo usuario y selecciona el botón Guardar. Una vez seleccionado dicho botón se ejecuta el método AdicionarUsuario de la clase AdicionarUsuarioActivity.java, en el cual se valida que los datos introducidos fueron bien definidos.

Posteriormente se muestra un mensaje confirmando la validez de los datos introducidos en la clase activity_adicionar_usuario.xml y se ejecuta el método user_createUser de la clase AdicionarUsuarioWSTask.java. Dicho método se encarga de realizar la conexion con el servicio web que

permite adicionar el nuevo usuario en la BD de SIGE. Después se muestra un mensaje en la clase activity_gestionar_usuario_rol.xml, confirmándose la creación del nuevo usuario finalizando de esta forma el escenario Adicionar usuario.

2.7 Patrones empleados en la solución

Los patrones de diseño son soluciones para problemas típicos y recurrentes que se pueden encontrar a la hora de desarrollar una aplicación. Aunque la aplicación que se desarrolle sea única, tendrá partes comunes con otras aplicaciones: acceso a datos, creación de objetos, operaciones entre sistemas. En lugar de reinventar la rueda, se puede solucionar el problema utilizando algún patrón, ya que son soluciones probadas y documentadas por multitud de programadores (Genbeta, 2014).

2.7.1 Patrones de diseño GRASP

Los patrones GRASP (del inglés General Responsibility Assignment Software Patterns, en español Patrones Generales de Software para Asignar Responsabilidades) describen los principios fundamentales del diseño de objetos y la asignación de responsabilidades (CRAIG , 2016).

En el desarrollo de la herramienta se aplicaron principalmente los siguientes patrones GRASP:

- **Controlador:** es utilizado por todas las clases implicadas en la capa de presentación de la lógica del negocio. Las clases Presentadoras poseen la responsabilidad de controlar el flujo de eventos mediante las actividades correspondientes. Un ejemplo es la clase GestionarUsuarioRolActivity, encargada de controlar la lógica del negocio en lo que respecta a su CU Gestionar usuario.
- **Bajo acoplamiento:** este patrón es utilizado en la creación de clases independientes para la lógica de los diferentes tipos de clases; actividades y entidades. Este patrón se puede observar en la clase Usuario, debido a que a esta solamente accede la clase GestionarUsuarioRolActivity.
- **Alta cohesión:** caracteriza las clases con responsabilidades similares. Cada elemento del diseño debe realizar una labor única dentro del sistema como es el caso de la clase de presentación GestionarUsuarioActivity cuya responsabilidad es la de gestionar los usuarios y no desempeña responsabilidades externas a esta.

2.7.2 Patrones de diseño GoF

Los patrones GoF (del inglés Gang of Four, en español Banda de los Cuatro) se utilizan en situaciones frecuentes debido a que se basan en la experiencia acumulada al resolver problemas reiterativos. Ayudan a construir software basado en la reutilización, a construir clases reutilizables (Pérez , 2004).

En el desarrollo de la herramienta se aplicó el siguiente patrón GoF:

- **Builder (Constructor virtual):** Abstrae el proceso de creación de un objeto complejo, centralizando dicho proceso en un único punto. Este patrón se observa en la clase UsuariosFragment ya que esta clase se encarga de crear los objetos de tipo Usuario haciendo uso de los atributos obtenidos de la clase ListarUsuarioWSTask. Esta última obtiene dichos atributos de la ejecución del servicio web que devuelve el listado de usuarios existentes en la BD de SIGE. Dichos objetos de tipo usuario que se construyeron son posteriormente mostrados como elementos de la lista de usuarios en la herramienta.

2.8 Modelo de datos

Un modelo de datos es la estructura o representación física de las tablas de la base de datos obtenido a partir del diagrama de clases persistentes. Aporta la base conceptual para diseñar aplicaciones que hacen un uso intensivo de datos. De igual forma, permiten describir las estructuras de datos de la BD, las restricciones de integridad y las operaciones de manipulación de los datos (Sánchez, 2004). A continuación se muestra el modelo de datos de la aplicación.

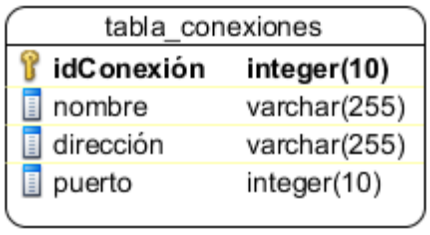


Fig. 5: Modelo de datos de SIGE-Droid.

El modelo de datos de SIGE-Droid está conformado por una única tabla la cual permite almacenar las conexiones que configure el usuario. El resto de las operaciones que realiza la aplicación son ejecutadas sobre las tablas implicadas en la propia BD de SIGE.

2.9 Modelo de despliegue

El diagrama de despliegue modela la topología del hardware sobre el que se ejecuta un sistema. De igual manera muestra la configuración de los nodos que participan en la ejecución. Además, representa el despliegue físico de un componente (CASTRO, 2016). A continuación se presenta el modelo de despliegue de SIGE-Droid.

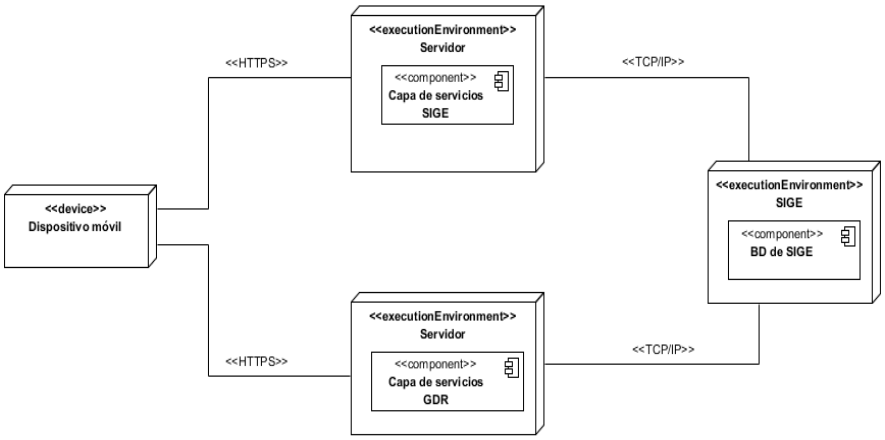


Fig. 6: Diagrama de Despliegue de SIGE-Droid.

Los nodos representados en el diagrama de despliegue son:

Dispositivo móvil: Es el dispositivo móvil que tendrá la herramienta SIGE-Droid instalada.
Capa de servicios de SIGE: Servidor sobre el cual SIGE-Droid realizará las acciones correspondientes a la gestión de seguridad. La conexión entre el dispositivo móvil y la capa de servicios de SIGE se llevará a cabo a través del protocolo HTTPS.
Capa de servicios de GDR: Servidor sobre el cual SIGE-Droid realiza las acciones correspondientes a la gestión de seguridad. La conexión entre el dispositivo móvil y la capa de servicios de GDR se llevará a cabo a través del protocolo HTTPS.
BD de SIGE: BD en la que se almacenarán los usuarios, roles, así como los reportes existentes en SIGE. La conexión entre la BD de SIGE y ambas capa de servicios se realizará a través del protocolo TCP/IP.

2.10 Conclusiones del capítulo

La realización del modelo de dominio permitió lograr un mejor entendimiento del entorno real de la herramienta. Posibilitó conocer cómo los usuarios de SIGE realizan actualmente los procesos de gestión de

usuarios, gestión de roles, así como la visualización de reportes estadísticos. Se identificaron 18 RF agrupados en 5 CU, los cuales fueron relacionados mediante un diagrama de CU del sistema; se les realizó una descripción detallada logrando un mayor acercamiento a lo que la aplicación debe hacer. Para definir la estructura general de la solución propuesta se aplicó el patrón arquitectónico MVP. Con el propósito de representar la organización de la herramienta se diseñaron los DCD mostrando la relación entre clases, atributos e interfaces haciendo uso de los siguientes patronos de diseño: Controlador, Alta cohesión, Bajo acoplamiento y Constructor. Los diagramas de secuencia elaborados permitieron describir gráficamente los escenarios posibles de cada CU. La realización del diagrama de despliegue propició una visión de cómo estará distribuida la herramienta físicamente.

Capítulo 3: Implementación y pruebas de SIGE-Droid

En este capítulo se elaboran los artefactos correspondientes a la implementación tomando como entrada los resultados obtenidos en la etapa de diseño. Se representa el diagrama de componentes que detalla la forma en que está estructurada la aplicación, reflejando la transformación de los elementos del modelo del diseño en términos de componentes, así como las dependencias entre ellos. Además, se diseñan y aplican las pruebas para comprobar el correcto funcionamiento de la aplicación.

3.1 Modelo de implementación

El modelo de implementación describe cómo los elementos de diseño se implementan en componentes. Además, describe los componentes a construir y su organización en nodos físicos en los que funcionará el sistema (Hassan, 2011).

3.1.1 Diagrama de componentes

Los diagramas de componentes describen la descomposición física de los elementos de un sistema. Estos permiten visualizar la estructura de alto nivel del sistema y el comportamiento del servicio que estos componentes proporcionan y usan a través de interfaces (Matías, 2014). A continuación se muestra el diagrama de componentes correspondiente al CU Gestionar Usuario.

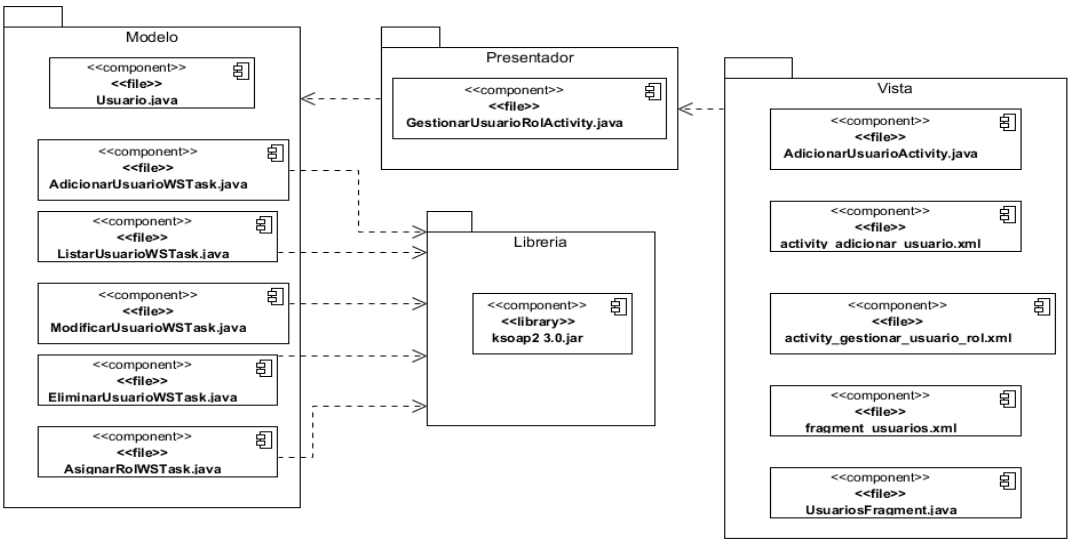


Fig. 7: Diagrama de Componentes del CU Gestionar usuario.

El diagrama de componentes mostrado en la fig. 8 cuenta con tres paquetes de implementación básicos mencionados a continuación:

- **Vista:** agrupa las clases AdicionarUsuarioActivity.java, activity_adicionar_usuario.xml, activity_gestionar_usuario_rol.xml, fragment_usuarios.xml, UsuariosFragment.java, encargadas de manejar la interfaz de usuario.
- **Presentador:** contiene la clase presentadora que es la encargada de manejar la interfaz principal del CU Gestionar usuario.
- **Modelo:** agrupa la clase entidad Usuario.java y las clases AdicionarUsuarioWSTask.java, ListarUsuarioWSTask.java, ModificarUsuarioWSTask.java, EliminarUsuarioWSTask.java, AsignarRolWSTask.java, encargadas de interactuar con los servicios web.

3.2 Código fuente

El código fuente es un conjunto de líneas de texto que son las instrucciones que debe seguir la computadora para ejecutar un programa informático. Por tanto, en el código fuente de un programa está escrito por completo su funcionamiento. Estas líneas de texto están escritas en un lenguaje de programación específico y que puede ser leído por un programador. Debe traducirse a lenguaje máquina para que pueda ser ejecutado por la computadora, este proceso se denomina compilación (Bonet, 2010).

3.2.1 Estándar de codificación

Un estándar de codificación completo comprende todos los aspectos de la generación de código. Un código fuente completo debe reflejar un estilo armonioso, como si un único programador hubiera escrito todo el código de una sola vez. Al comenzar un software, se establece un estándar de codificación para asegurarse de que todos los programadores del proyecto trabajen de forma coordinada. Cuando el proyecto de software incorpore código fuente previo, o bien cuando realice el mantenimiento de un sistema de software creado anteriormente, el estándar de codificación debería establecer cómo operar con la base de código existente (Microsoft, 2010).

En la propuesta de solución se tendrán en cuenta las siguientes convenciones definidas por el equipo de desarrollo:

- Se empleó el idioma español para la asignación de nombres.
- Los comentarios multilíneas se escriben comenzando con los caracteres “/*” y terminando con “*/”,

los comentarios de una sola línea comienzan con los caracteres "//".

- Todo el código desarrollado tiene una indentación de 8 espacios. La razón principal de tener una buena indentación es que se puede observar la estructura general del código a simple vista.
- Se declara cada variable en una línea distinta, de esta forma cada variable se puede comentar por separado.
- Las clases interfaces se escriben en minúsculas y separados con guiones de suelo.
- Se inicializa cada variable en su declaración, a menos que su valor inicial dependa de algún cálculo.
- No se declaran variables con el mismo nombre dentro de un método.
- Todas las sentencias del tipo *if*, *for* o *while* tienen llaves aunque sólo contengan una sentencia, de esta forma se evita la introducción accidental de errores si se añaden posteriores sentencias.
- Para todos los nombres de las clases Java, la primera letra debe de ser Mayúscula, si son varias palabras se debe de intercalar entre mayúsculas y minúsculas, este mecanismo de nombre es llamado *LowelCamelCase*.
- La asignación de nombres se basa en las siguientes normas genéricas:

  1. Se usan descriptores que dejan claro el cometido de la variable, método o clase.

  2. Se usa terminología aplicable al dominio.

  3. No se usarán siglas en los nombres a menos que sean muy largos y sea necesario abreviar.

A continuación se muestra un fragmento de código fuente de la herramienta evidenciándose el uso del estilo de codificación definido. Este fragmento es extraído de la clase GestionarUsuarioRolActivity.

```
public boolean onOptionsItemSelected(MenuItem item) {

    int id = item.getItemId();
    Fragment fragment= getCurrentFragment();

    switch (id) {
        case R.id.action_reload:
            if (fragment instanceof UsuariosFragment)
                ((UsuariosFragment) fragment).ActualizarLista();
            else
                ((RolesFragment) fragment).ActualizarLista();

            return true;
        case R.id.action_add:
            if (fragment instanceof UsuariosFragment)
                ((UsuariosFragment) fragment).CrearUsuario(false, -1);
            else
                ((RolesFragment) fragment).CrearRol(false, -1);
            return true;

        case R.id.action_setRol:
            if (fragment instanceof UsuariosFragment)
                ((UsuariosFragment) fragment).AsignarRol();
            return true;

        case R.id.action_exit:
            startActivity(new Intent(this,LoginActivity.class));
            finish();
            return true;
    }

    return super.onOptionsItemSelected(item);
}

public Fragment getCurrentFragment() {
    int i = mTabLayout.getSelectedTabPosition();
    ViewPagerAdapter adapter = (ViewPagerAdapter) mViewPager.getAdapter();
    Fragment fragment = adapter.getItem(i);

    return fragment;
}
```

Fig. 8: Fragmento de código fuente de la clase GestionarUsuarioRolActivity.

3.3 Pruebas de software

Las pruebas de software son las investigaciones empíricas y técnicas cuyo fin es proporcionar información objetiva e independiente sobre la calidad del producto. Esta actividad forma parte del proceso de control de calidad global. Las pruebas son básicamente un conjunto de actividades dentro del desarrollo de software y dependiendo del tipo de pruebas, estas actividades podrán ser implementadas en cualquier momento del proceso de desarrollo (MONTES, 2014).

Para asegurar el correcto funcionamiento de las aplicaciones el equipo de pruebas debe realizar diversas pruebas a diferentes niveles. Los niveles corresponden a las siguientes clasificaciones que son: pruebas de unidad, integración, sistema y aceptación. Las pruebas pueden ejecutarse en cualquier punto del proceso de desarrollo de software, es entonces que los niveles de prueba permiten entender con claridad los diferentes puntos o etapas en donde pueden ejecutarse ciertos tipos de prueba. Para verificar que el desarrollo de la herramienta cumple con los requisitos previamente identificados se realizan las pruebas a nivel de integración y de aceptación.

3.3.1 Pruebas de integración

Las pruebas de este nivel son ejecutadas por el desarrollador y consisten en la comprobación de que los elementos del software que interactúan entre sí funcionan de manera correcta (Zapata, 2013). Con la aplicación de dichas pruebas se comprueba la correcta integración de los componentes de la aplicación.

Dentro de las pruebas del nivel de integración se decide aplicar como tipo de prueba las pruebas funcionales las cuales se enfocan en comprobar que los RF fueron implementados correctamente. Para la ejecución de dichas pruebas se utiliza el método de Caja Negra con su técnica partición de equivalencia; pues el mismo se centra en los RF del software permitiendo obtener conjuntos de condiciones de entradas válidas e inválidas que ejerciten completamente todos los RF de un programa. Se utilizaron los casos de prueba basados en CU (por cada CU se realiza un caso de prueba) como herramienta de apoyo a las pruebas funcionales. Cada caso de prueba contiene la descripción de las secciones en las cuales se divide, las variables que intervienen en el CU y los escenarios de prueba con los valores válidos e inválidos a probar. A continuación se muestra un ejemplo del caso de prueba para el CU Gestionar usuario.

Tabla 4 Secciones de prueba para el CU Gestionar usuario

Nombre de la sección	Descripción de la funcionalidad
SC1 Adicionar usuario	El administrador decide insertar un nuevo usuario, el sistema envía un mensaje indicando que la acción se realizó exitosamente, terminando así el CU.
SC2 Editar usuario	El administrador decide editar un usuario existente, el sistema envía un mensaje indicando que la petición se realizó exitosamente, terminando así el CU.
SC3 Eliminar usuario	El administrador decide eliminar un usuario existente, el sistema envía un mensaje indicando que la acción se realizó exitosamente, terminando así el CU.
SC4 Listar usuario	El administrador se autentica en el sistema con permisos de administración, el sistema envía un mensaje indicando que la acción se realizó exitosamente, terminando así el CU.

Tabla 5 Descripción de las variables del caso de prueba del CU Gestionar usuario

No	Nombre de campo	Clasificación	Valor Nulo	Descripción

1	Nombre	Campo de texto	No	Cadena de caracteres que se corresponden al nombre del usuario. Alfanumérico de 1 a 25 caracteres. Admite caracteres de a-z, A-Z.
2	Apellido	Campo de texto	No	Cadena de caracteres que se corresponden al apellido del usuario. Alfanumérico de 1 a 30 caracteres. Admite caracteres de a-z, A-Z.
3	Usuario	Campo de texto	No	Cadena de caracteres que se corresponden al identificativo del usuario. Alfanumérico de 1 a 25 caracteres. Admite caracteres de a-z y 0-9.
4	Password	Campo de texto	No	Cadena de caracteres que se corresponden la contraseña de acceso del usuario al sistema. Alfanumérico de 1 a 25 caracteres. Admite todo tipo de carácter.
5	Correo	Campo de texto	No	Cadena de caracteres que se corresponden al correo electrónico del usuario. Alfanumérico de 1 a 25 caracteres. Admite caracteres de a-z, 0-9, @, .
6	Dirección IP	Campo de texto	No	Cadena de caracteres que se corresponden a la dirección ip desde la cual el usuario accederá al sistema. Alfanumérico de 1 a 25 caracteres. Admite caracteres 0-9, a-z, A-Z - _ ( ). : /.

Tabla 6 Escenarios del caso de prueba Adicionar usuario del CU Gestionar usuario

Escenario	**Descripción**	**Variables**						**Respuesta del sistema**
		1	**2**	**3**	**4**	**5**	**6**	
EC 1.1 Adicionar usuario.	Permite adicionar un usuario en SIGE	V	V	V	V	V	V	Se adiciona el usuario, se muestra el mensaje "La operación se realizó satisfactoriamente" y se actualiza el listado de usuarios.
EC 2.2 Adicionar usuario con datos incorrectos.	En este escenario se realiza la adición de un nuevo usuario en la BD de SIGE con datos incorrectos.	I	V	V	V	V	V	La herramienta valida que los campos estén correctamente y muestra un mensaje informando que existen datos no válidos.
		V	I	V	V	V	V	
		V	V	I	V	V	V	

		V	V	V	I	V	V	
		V	V	V	V	I	V	
		V	V	V	V	V	I	
EC 1.3 Cancelar operación.	En este escenario se cancela la operación de adicionar un nuevo usuario							Se cancela la operación de adicionar un nuevo usuario.

Ejecución de las pruebas de integración

Se realizaron tres iteraciones de pruebas funcionales a la herramienta desarrollada aplicando los cinco casos de prueba diseñados. Estas pruebas arrojaron un total de 13 NC (No Conformidades). Se detectaron nueve NC en la primera iteración y cuatro NC en la segunda clasificadas como significativas. En la tercera iteración se comprobó que las NC de las iteraciones anteriores estuvieran resueltas y no se detectaron nuevas NC, comprobándose de este modo la validez de la herramienta. A continuación se muestra un gráfico de las NC encontradas por iteración.

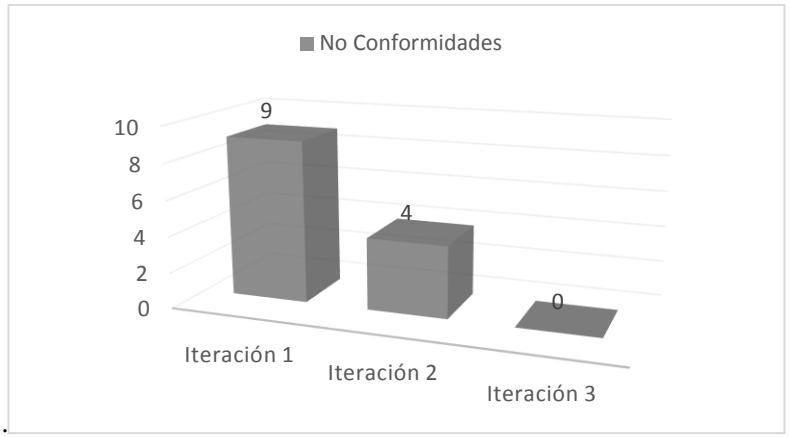


Fig. 9: Gráfica de iteraciones de pruebas funcionales.

3.3.2 Pruebas de aceptación

Las pruebas de aceptación son aquellas ejecutadas por el propio usuario final para comprobar que el sistema realiza lo que el cliente especificó de manera correcta. Dichas pruebas aseguran la validez y

conformidad de los clientes con el producto recibido en base a lo que se acordó inicialmente. De manera general las pruebas de aceptación pueden afrontarse mediante dos tipos de procedimiento para realizarlas: pruebas alfa y pruebas beta. (Dominguez Mayo, 2014)

Dentro del nivel de aceptación se decide aplicar las pruebas alfa ya que en ellas se le entrega a un usuario final todo el producto terminado, junto a su documentación correspondiente. Pues el cliente es el encargado de ejecutarlas en presencia del desarrollador y en entornos previamente preparados para la ejecución de las mismas. Una vez que el cliente detecta errores los va informando al desarrollador para su corrección.

Ejecución de las pruebas de aceptación

Se realizaron dos iteraciones de pruebas Alfa, detectándose un total de tres NC. En la primera iteración se detectaron tres NC y en la segunda iteración no se detectaron NC. El cliente se mostró conforme con la herramienta desarrollada y emitió una carta de aceptación (ver Anexo 1). A continuación se muestra el gráfico de las NC detectadas por el cliente en cada iteración.

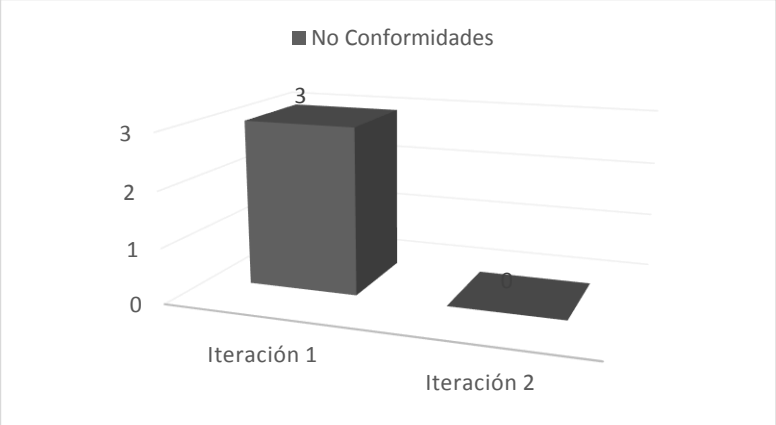


Fig. 10: Gráfica de iteraciones de pruebas Alfa.

3.4 Interfaces de la aplicación SIGE-Droid

La herramienta desarrollada permite a los administradores del sistema realizar los procesos de gestión de usuarios y gestión de roles desde dispositivos móviles con SO Android desde cualquier espacio de la institución mediante el uso de la conexión inalámbrica. De esta manera ya los administradores de SIGE pueden gestionar la seguridad en dicha herramienta sin estar limitados a sus puestos de trabajo.

De igual forma la herramienta permite la visualización de reportes estadísticos por parte de los directivos de la empresa, posibilitando el apoyo al proceso de toma de decisiones. Permite además la obtención de información actualizada desde cualquier espacio de la institución, en el momento y espacio deseado. De este modo se evita la impresión de la información a tratar en las reuniones, además de aumentar el área de acceso a la información estadistica almacenada en SIGE.

Interfaz para gestionar usuarios y roles

Dicha interfaz permite la gestión de usuarios (Adicionar usuario, modificar usuario, eliminar usuario, listar usuarios y asignar rol) y roles (Adicionar rol, modificar rol, eliminar rol, listar roles) en SIGE.

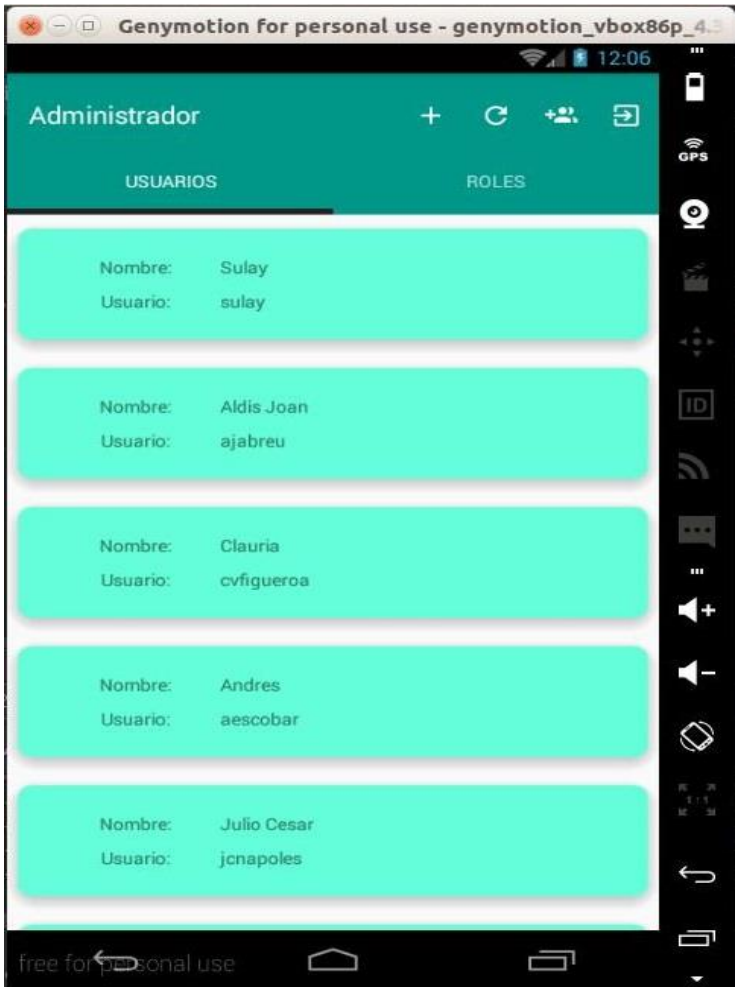


Fig. 11: Interfaz principal para la gestión de seguridad en SIGE-Droid.

Interfaz para visualizar reporte

Para visualizar un reporte es necesario seleccionar el reporte de la lista de reportes existentes en el sistema, posteriormente se introducirán las datos necesarios para la generación del reporte, los cuales son introducidos por el usuario. Una vez realizadas dichas acciones se puede observar en detalles el contenido del reporte.

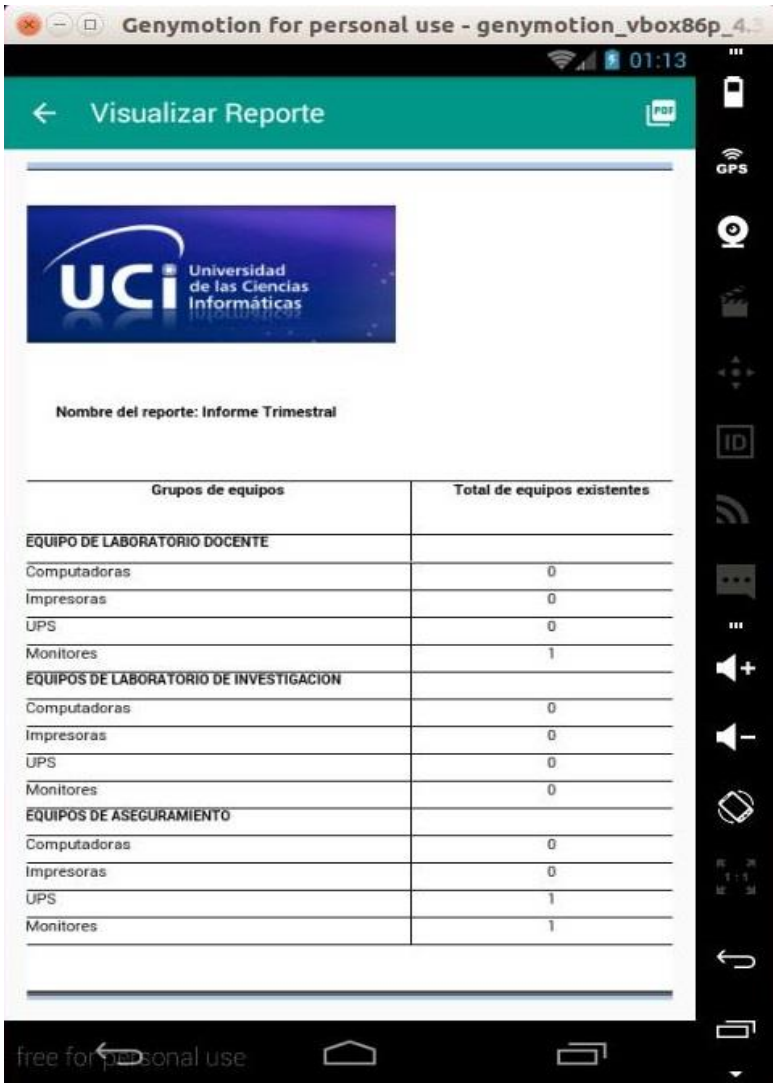


Fig. 12: Interfaz para visualizar reporte en SIGE-Droid

3.5 Conclusiones del capítulo

Los diagramas de componentes permitieron observar la organización de todas las clases de la herramienta a nivel de componentes, representándose las dependencias existentes entre ellos. Se realizó la implementación de SIGE-Droid siguiendo el estándar de codificación definido para dar cumplimiento a los

18 RF identificados. Para comprobar el correcto funcionamiento de la herramienta se aplicaron las pruebas funcionales utilizando el método Caja Negra con la técnica partición de equivalencia. También se realizaron pruebas de aceptación por el propio usuario final para comprobar que la solución propuesta realiza lo que el cliente especificó. Dichas pruebas permitieron detectar un total de 15 NC las cuales fueron corregidas satisfactoriamente. Las pruebas de aceptación permitieron demostrar que SIGE-Droid permite el apoyo al trabajo con SIGE, obteniéndose el Acta de Aceptación del cliente.

Conclusiones generales

Una vez culminada la investigación se puede afirmar que se le dio cumplimiento a los objetivos planteados, arribando a las siguientes conclusiones:

- Con el estudio de SIGE se definieron las principales funcionalidades que debe poseer SIGE-Droid para desarrollar una solución apropiada.
- El estudio de las aplicaciones Android que se conectan a BD externas permitió determinar el uso de servicios web para posibilitar la conexión entre SIGE y SIGE-Droid.
- Con el proceso de obtención de requisitos se determinaron un total de 18 RF y 5 FNF, los cuales debe cumplir SIGE-Droid.
- Mediante la implementación de las funcionalidades definidas se dio cumplimiento a los 18 RF identificados en las etapas de análisis y diseño de SIGE-Droid.
- La etapa de pruebas permitió corregir un total de 15 NC detectadas aplicando las pruebas funcionales y de aceptación.
- Como resultado se obtuvo la herramienta SIGE-Droid que permite la visualización de reportes y la gestión de seguridad en SIGE desde dispositivos móviles con SO Android.

Recomendaciones

- Agregar funcionalidades a SIGE-Droid que permitan la comunicación con la capa de servicios de la nueva versión de SIGE.

Referencias Bibliográficas

1. **academiaandroid. 2014.** academiaandroid. *android-studio.* [En línea] 2014. [Citado el: 19 de Noviembre de 2015.] http://academiaandroid.com/android-studio-v1-caracteristicas-comparativa-eclipse/.

2. **ALEGSA. 2009.** lenguajes-de-programacion. [En línea] 2009. [Citado el: 17 de Noviembre de 2015.] http://www.lenguajes-de-programacion.com/lenguajes-de-programacion.shtml.

3. **AMPUERO, Margarita Andre. 2015.** Análisis comparativo de modelos y estándares para evaluar la calidad del producto de software. Revista Cubana de Ingeniería. 2015, Vol. vol. 6.

4. **ARLOW, Jim. 2016.** UML 2 a unifikovaný proces vývoje aplikacíon. s.l. : Albatros Media as, 2016, 2016.

5. **Barry & Associates, Inc. 2015.** www.service-architecture.com. *www.service-architecture.com.* [En línea] 2015. [Citado el: 5 de mayo de 2016.] http://www.service-architecture.com/articles/web-services/web_services_definition.html.

6. **Boehm, B. W. 2012.** eumed. [En línea] 2012. [Citado el: 17 de noviembre de 2015.] www.eumed.net/tesis-doctorales/2014/jlcv/software.htm.

7. **Bonet, Eugeni . 2010.** www.carlospes.com. [En línea] 2010. [Citado el: 25 de 5 de 2016.] http://www.carlospes.com/minidiccionario/codigo_fuente.php.

8. **CASTRO, German Gomez. 2016.** *MobiLearn: Context-Aware Mobile Learning System.* s.l. : IEEE Latin America Transactions, vol. 14, no 2, p. 958-964., 2016.

9. **CRAIG , LARMAN. 2016.** Introducción al análisis y diseño orientado a objetos. [aut. libro] LARMAN CRAIG. *Aplicación de Patrones de Diseño para Garantizar Alta Flexibilidad en el Software.* s.l. : Tecnología & Desarrollo (Trujillo), vol. 12, no 1, p. 77-82., 2016.

10. **Dominguez Mayo, Francisco José . 2014.** *Pruebas de aceptación orientadas al usuario:contexto ágil para un proyecto de gestión.* Sevilla : s.n., 2014.

11. **Eclipse Foundation. 2013.** enterpriseunifiedprocess. [En línea] 2013. [Citado el: 17 de Noviembre de 2015.] http://www.enterpriseunifiedprocess.com/.

12. **Editorial McGraw-Hill. 2012.** www.cavsi.com. [En línea] 2012. [Citado el: 2015 de 11 de 29.] www.cavsi.com/preguntasrespuestas/que-es-un-sistema-gestor-de-bases-de-datos-o-sgbd.

13. **ESCALANTE, Lain Cárdenas. 2016.** *El patrón de arquitectura n-capas con orientación al dominio como solución en el diseño de aplicaciones empresariales.* s.l. : Tecnología & Desarrollo (Trujillo), vol. 11, no 1, p. 59-66., 2016.

14. **ESTRADA, Lisandra Guibert. 2015.** Domain knowledge representation for programming teaching. . s.l. : IEEE Latin America Transactions, 2015, vol. 13, no 5, p. 1528-1533., 2015.

15. **Facebook Developers. 2012.** developers.facebook. [En línea] 2012. [Citado el: 11 de Mayo de 2016.] https://developers.facebook.com/docs/graph-api.

16. **Fernández, Francisco J. . 2010.** https://books.google.com.cu. [En línea] 2010. [Citado el: 22 de febrero de 2016.] https://books.google.com.cu/books?id=gQWd49zSut4C&pg=PA299&lpg=PA299&dq=pagina+oficial+de+modelo+de+dise%C3%B1o+en+ingenieria+de+software&source=bl&ots=s654pozBya&sig=dwgnb-rPe6oyGQ0JV4W86tWm9tA&hl=es&sa=X&ved=0ahUKEwicq_6l39PMAhWCOyYKHRWgAXAQ6AEILzAD#v=o.

17. **Genbeta. 2014.** Genbeta:dev. [En línea] 14 de Julio de 2014. [Citado el: 18 de Abril de 2016.] http://www.genbetadev.com/metodologias-de-programacion/patrones-de-diseno-que-son-y-por-que-debes-usarlos.

18. **Google Developers. 2013.** developers.google. [En línea] 2013. [Citado el: 11 de mayo de 2016.] https://developers.google.com/gmail/.

19. **—. 2013.** developers.google. [En línea] 2013. [Citado el: 11 de Mayo de 2016.] https://developers.google.com/games/services/web/api/#TurnBasedMatches.

20. **Hassan, Yusef . 2011.** https://prezi.com. [En línea] 2011. [Citado el: 20 de 5 de 2016.] https://prezi.com/ijpanrlnvhyj/modelo-de-implementacion/.

21. **Holgate, Colin. 2012.** LiveCode Mobile Development Beginner´s Guide. 2012.

22. **2010.** https://books.google.com.cu. [En línea] 2010. [Citado el: 21 de febrero de 2016.] https://books.google.com.cu/books?id=u1JjAgAAQBAJ&pg=PA23&lpg=PA23&dq=sitio+oficial+de+patrones+de+caso+de+uso&source=bl&ots=R7r9iojv3V&sig=8Q0jU247Obh2STsks82bkaUajeo&hl=es&sa=X&ved=0ahUKEwjoi9jF19PMAhWJKx4KHeWLBs0Q6AEIJzAC#v=onepage&q=sitio%20oficial%20.

23. **InformaticaSONs. 2011.** http://www.sosinformatica.net. [En línea] 2011. [Citado el: 18 de noviembre de 2015.] http://www.sosinformatica.net/evi/VisualBasic/guia_rapida/vb_guia_bd01.htm.

24. **Jacobson, Ivar . 2011.** umsl. [En línea] 2011. [Citado el: 17 de Noviembre de 2015.] http://www.umsl.edu/~sauterv/analysis/F08papers/View.html.

25. **JARAMILLO. 2013.** Reglas Sintáctico-semánticas para Relacionar los Objetivos Organizacionales y los Problemas en el Contexto de la Educción Temprana de Requisitos de Software. s.l. : Revista Latinoamericana de Ingenieria de Software, 2013.

26. **KIM, Sunghwan. 2015.** PUG-SOAP and PUG-REST: web services for programmatic access to chemical information in PubChem. . *PUG-SOAP and PUG-REST: web services for programmatic access to chemical information in PubChem. .* s.l. : Nucleic acids research, 2015, p. gkv396., 2015.

27. **Londoño, Jorge Hernan Abad. 2005.** Ingeniería de Software. [En línea] 6 de Abril de 2005. [Citado el: 2 de Abril de 2016.] http://ing-sw.blogspot.com/2005/04/tipos-de-pruebas-de-software.html.

28. **Matías, Jesús . 2014.** www.academia.edu. [En línea] 2014. [Citado el: 20 de 5 de 2016.] https://www.academia.edu/9680467/Metodolog%C3%ADa_UML.

29. **Mellor, Steven . 2011.** www.altova.com/. [En línea] 2011. [Citado el: 15 de 5 de 2016.] http://www.altova.com/es/umodel/uml-package-diagrams.html.

30. **Microsoft. 2010.** msdn.microsoft. [En línea] 2010. [Citado el: 25 de 5 de 2016.] https://msdn.microsoft.com/es-es/library/aa291593%28v=vs.71%29.aspx.

31. **MONTES, Martha. 2014.** *Testing process for small software development organizations. .* s.l. : Facultad de Ingeniería, 2015, vol. 24, no 39, p. 55-70., 2014.

32. **Pérez , Mariñán. 2004.** "Patrones de Diseño (Design Patterns). [aut. libro] Pérez Mariñán. *Design Patterns.* 2004.

33. **Pressman. 2009.** Cap_13_Estrtegia_Prueba. 2009.

34. **Rodríguez, Saúl Cuesta. 2014.** SG Buzz. [En línea] 2014. [Citado el: 12 de Febrero de 2016.] http://sg.com.mx/.

35. **—. 2014.** SG Buzz. [En línea] 2014. [Citado el: 12 de Febrero de 2016.] http://sg.com.mx/.

36. **ROMERO, Juan F. 2015.** Un lenguaje de modelado para el desarrollo de software auto-adaptativo. *Un lenguaje de modelado para el desarrollo de software auto-adaptativo.* s.l. : Anuario de Jóvenes Investigadores, 2015, no 8, p. 57-59., 2015.

37. **Rómmel, Filein . 2012.** sqlite.org. [En línea] 2012. [Citado el: 2015 de 11 de 29.] http://sqlite.org/about.html.

38. **Rumbaugh, Jim . 2009.** visual-paradigm. [En línea] 2009. [Citado el: 17 de Noviembre de 2015.] http://www.visual-paradigm.com/aboutus/.

39. **Sánchez, Jorge . 2004.** https://books.google.com.cui. [En línea] 2004. [Citado el: 18 de Marzo de 2016.] https://books.google.com.cu/books?id=udFECQAAQBAJ&pg=PR7&lpg=PR7&dq=concepto+oficial+de+modelo+de+datos+relacional&source=bl&ots=lgQd1rhmbR&sig=iWymVd2oGlnQrCx9YRL7GmL_3tQ&hl=es&sa=X&ved=0ahUKEwj67dX4pdbMAhVIWCYKHWPiBcoQ6AEILzAE#v=onepage&q=concepto%20ofi.

40. **SCHAFFNER, Donald W. 2015.** *The Influence of Soap Characteristics and Food Service Facility Type on the Degree of Bacterial Contamination of Open, Refillable Bulk Soaps. .* s.l. : En 2015 Annual Meeting (July 25-28, 2015). Iafp, 2015., 2015.

41. **SCOTT, KENDALL . 2014.** http://datateca.unad.edu.co. [En línea] 2014. [Citado el: 21 de febrero de 2016.] http://datateca.unad.edu.co/contenidos/200609/exeuml/leccin_39_diagrama_de_clases_de_diseo.html.

42. **scribd. 2015.** Scribd Inc. [En línea] 2015. [Citado el: 16 de Enero de 2015.] http://es.scribd.com/doc/37187866/Requerimientos-funcionales-y-no-funcionales#scribd.

43. **Software, Departamento Ingeniería y Gestión de. 2014.** *Modelando el Sistema Con Escenario CU.* 2014.

44. **VALENCIA, Paula Ortiz. 2015.** *Valoración de una estrategia didáctica para la enseñanza y el aprendizaje del modelado de software utilizando el Proyecto Zero. .* s.l. : TRILOGÍA. Ciencia, Tecnología y Sociedad, 2016, vol. 8,, 2015.

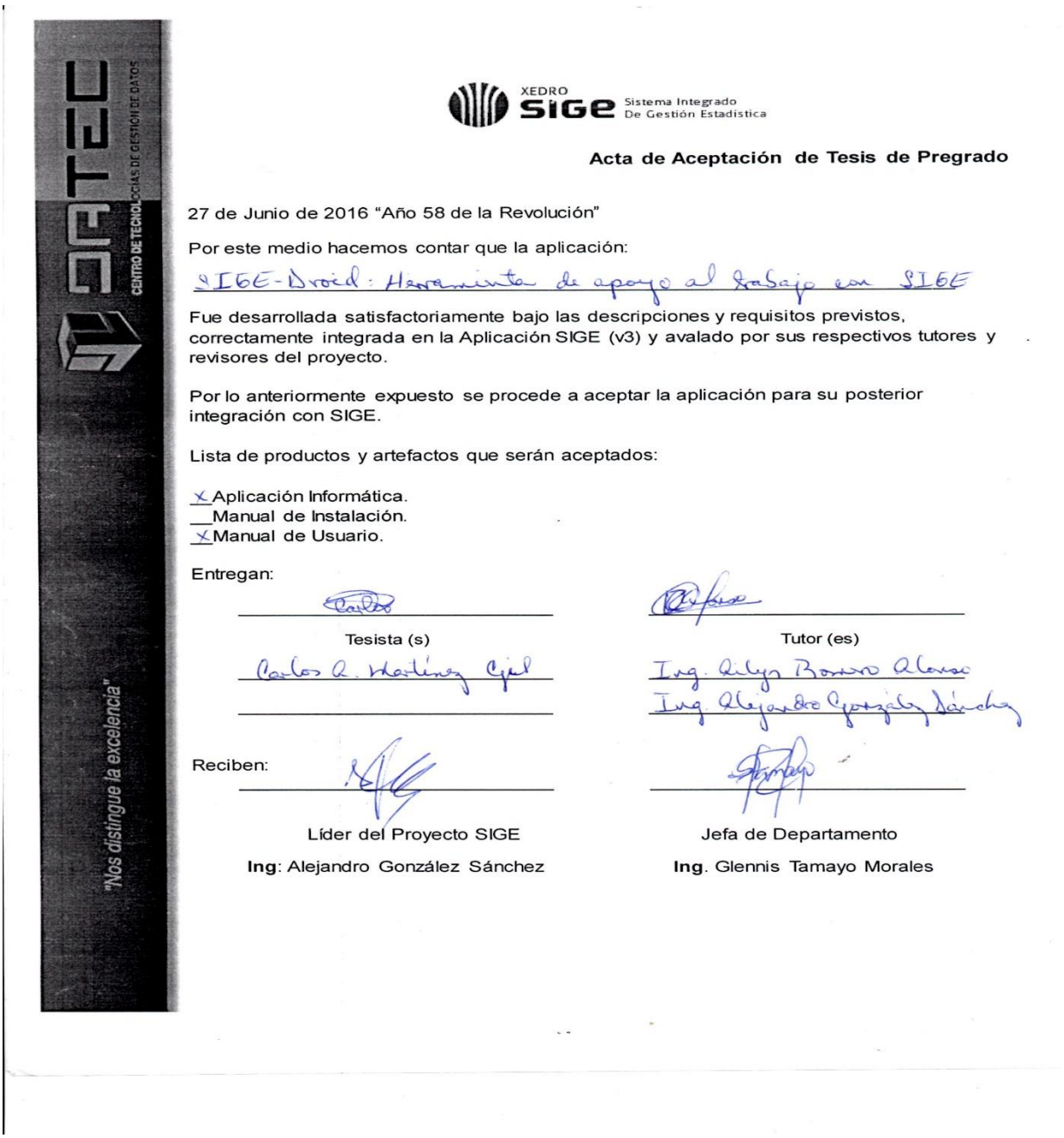
45. **2010.** www.synergix.wordpress.com. [En línea] 2010. [Citado el: 19 de Febreo de 2016.] https://synergix.wordpress.com/2008/06/07/casos-de-uso-avanzados-relacion-de-inclusion/.

46. **Zapata, Javier . 2013.** pruebasdelsoftware.wordpress. [En línea] 21 de Enero de 2013. [Citado el: 30 de mayo de 2016.] https://pruebasdelsoftware.wordpress.com/.

47. **ZAPATA, Sergio G. 2014.** Indicios Experimentales Respecto del Uso Técnicas Tradicionales de Elicitación de Requisitos de Software en Ambientes de Desarrollo Distribuidos. . s.l. : Revista Colombiana de Computación-RCC, vol. 14, no 2., 2014.

Anexos

Anexo 1: Carta de aceptación de SIGE-Droid.



CENTRO DE TECNOLOGÍAS DE GESTIÓN DE DATOS

"Nos distingue la excelencia"

XEDRO SIGE Sistema Integrado De Gestión Estadística

Acta de Aceptación de Tesis de Pregrado

27 de Junio de 2016 "Año 58 de la Revolución"

Por este medio hacemos contar que la aplicación:

SIGE-Droid: Herramienta de apoyo al trabajo con SIGE

Fue desarrollada satisfactoriamente bajo las descripciones y requisitos previstos, correctamente integrada en la Aplicación SIGE (v3) y avalado por sus respectivos tutores y revisores del proyecto.

Por lo anteriormente expuesto se procede a aceptar la aplicación para su posterior integración con SIGE.

Lista de productos y artefactos que serán aceptados:

X Aplicación Informática.
__Manual de Instalación.
X Manual de Usuario.

Entregan:

Tesista (s)

Carlos A. Martínez Ojel

Tutor (es)

Ing. Rilys Bonora Alonso

Ing. Alejandro González Sánchez

Reciben:

Líder del Proyecto SIGE

Ing: Alejandro González Sánchez

Jefa de Departamento

Ing. Glennis Tamayo Morales

Anexo 2: Descripción del CU Gestionar rol

Objetivo	Gestionar los roles
Actores	Administrador(Inicia)
Resumen	El CU se inicia cuando el Administrador selecciona una de las siguientes operaciones: Crear un nuevo rol, Modificar rol, Eliminar rol, Listar roles existentes. El CU finaliza cuando el Administrador realiza alguna de las operaciones previamente mencionadas.
Complejidad	Media
Prioridad	Crítico
Precondiciones	El usuario debe estar previamente autenticado en el sistema.
Postcondiciones	En dependencia de la acción del Administrador: Se adiciona un nuevo rol. Se modifica un rol existente. Se elimina un rol existente. Se muestra un listado de los roles existentes.

Flujo de eventos		
	Actor	**Sistema**
1.	Selecciona la opción **"Gestionar rol".**	
2.		Muestra el listado de los roles existentes y se activan las opciones correspondientes a la gestión de roles. El Administrador puede:

		a. Adicionar un nuevo rol: ir a la sección “Adicionar rol” b. Modificar datos de un rol: ir a la sección “Modificar rol” c. Eliminar rol: ir a la sección “Eliminar rol” d. Listar rol: Los roles se listan automáticamente al iniciarse el CU
Sección 1: “Adicionar Rol**”**		
Flujo básico < Adicionar Rol **>**		
	Actor	**Sistema**
1.	Selecciona el botón con el signo **“+”**.	
2.		Muestra la interfaz para crear un nuevo rol, solicitando los siguientes datos: Rol Descripción
3.	Introduce los datos correspondientes y selecciona la opción **“Aceptar”.**	
4.		Valida los datos introducidos.
5.		Registra el nuevo rol, recarga el listado de roles y muestra el mensaje “Rol adicionado correctamente”, finalizando así el CU.

Flujos alternos		
Nº 4.a <Validación>		
	Actor	**Sistema**
1.		Muestra los campos con errores de validación subrayados en rojo.
2.	Se posiciona sobre el campo que tiene errores en los datos.	
3.	Corrige los datos y presiona opción **"Aceptar".**	
4.		Muestra el mensaje de error de acuerdo al campo donde se posicione el Administrador.
Flujos alternos		
Nº 5.a <Cancelar**>**		
	Decide no modificar el rol y presiona la opción **"Cancelar".**	
		Cancela la operación y cierra la interfaz correspondiente finalizando así el CU.
Sección 2: "Modificar Rol**"**		
Flujo básico < Modificar Rol **>**		
	Actor	**Sistema**
	Selecciona con un clic sostenido un rol del listado de roles que se muestra y selecciona la opción **"Editar"**.	

		Muestra una interfaz con los datos del rol seleccionado previamente.
	Modifica los datos del rol que desea actualizar de la interfaz mostrada previamente por el sistema y selecciona la opción **"Aceptar"**.	
		Valida los campos editados.
		Actualiza el rol, recarga el listado de roles para mostrar la actualización realizada y muestra el mensaje "Rol actualizado correctamente", finalizando así el CU.
Flujos alternos		
Nº 4.a <Validación**>**		
1.		Muestra los campos con errores de validación subrayados en rojo.
2.	Se posiciona sobre el campo que tiene errores en los datos.	
3.		Muestra el mensaje de error de acuerdo al campo donde se posicione el Administrador.
4.	Corrige los datos y presiona opción **"Aceptar"**.	
Flujos alternos		
Nº 5.a < Cancelar**>**		
	El Administrador decide no modificar un rol y presiona la opción **"Cancelar"**.	

		La aplicación Cancela la operación y cierra la interfaz correspondiente finalizando así el CU.
Sección 3: "Eliminar Rol**"**		
Flujo básico < Eliminar Rol **>**		
	Actor	**Sistema**
	El Administrador selecciona con un clic sostenido un rol del listado de roles que se muestra y selecciona la opción Eliminar.	
		El sistema brinda una alerta: "¿Está seguro que desea eliminar el rol seleccionado?".
	El Administrador selecciona la opción **"Aceptar"**.	
		La aplicación elimina el rol, recarga el listado de roles y muestra el mensaje "Rol eliminado correctamente", finalizando así el CU.
Flujos alternos		
Nº 3.a <Selección Cancelar**>**		
1.	El Administrador selecciona la opción **"Cancelar"**.	
2.		La aplicación cierra la alerta finalizando así el CU.

Anexo 3: DCD del CU Gestionar rol