Universidad de la República
Montevideo, Uruguay

Proyecto de Grado
SIURE - Sistema Único de Receta Electrónica

Estudio, proyecto, desarrollo, pruebas y documentación de un sistema original para facilitar la receta de fármacos y la adherencia de los pacientes a los tratamientos, presentado como requerimiento parcial de la Unidad Curricular “Proyecto de Grado” de la Carrera de Ingeniería en Computación.

Mathías Fernández

Fernando Pedro

Tutores:
Franco Simini, Lucía Grundel y Antonio López

Núcleo de Ingeniería Biomédica de las Facultades de Medicina e Ingeniería

Trabajo realizado en colaboración con la
Escuela Universitaria de Tecnología Médica (EUTM) con la participación de
Natalia De León, Stephanie Peluffo y Valeria Silvera bajo la guía de la
Prof. Saadia Zawadski, directora de la Licenciatura en registros Médicos, en la modalidad de
PIRIM - Proyecto Interdisciplinario de Registros Informáticos Médicos

Montevideo, URUGUAY marzo 2015 – diciembre 2016

Índice

1 Introducción

La receta electrónica [1] es un programa informático mediante el cual el médico graba la medicación para su paciente, sustituyendo así la clásica receta en papel. Con la receta electrónica, el paciente podrá acudir a una farmacia con su documento identificatorio, donde se le brindará su medicación. Para esto, la farmacia deberá estar interconectada con el sistema informático de receta electrónica en el cual se realizan las prescripciones.

Mediante el uso de receta electrónica se mejora [2] la accesibilidad a los tratamientos y a su adherencia, ya que al haberse prescrito el tratamiento y sea vigente, no se deberá volver al centro de salud a renovar las recetas, sino que con estas se podrán recoger los medicamentos en cualquier farmacia. Además, la receta electrónica contribuye al uso racional de los medicamentos.

Entre las implementaciones exitosas de la receta electrónica, se encuentra la comunidad autónoma de Andalucía, donde un sistema de estas características se ha implantado con éxito desde el año 2005 [3].

La prescripción electrónica ha sido regulada [4] en esta comunidad autónoma de España por el decreto reglamentario 181/2007.

Derivado del término inglés “Prescription” que significa “Receta”, en el momento actual los términos “receta electrónica” y “prescripción electrónica” son prácticamente sinónimos. El uso de sistemas de receta electrónica está siendo bien recibido por los profesionales sanitarios. Diversos estudios comparativos de errores detectados [5] con prescripción electrónica versus receta manual concluyen que la primera reduce del orden de 30% los errores y contribuye a la seguridad de la farmacoterapia aplicada a los pacientes.
Anteriormente, el sistema más común en los hospitales y centros de salud españoles [5] era la prescripción médica manuscrita, que se transcribe en el servicio de farmacia y oficinas de farmacia a un programa electrónico. Con la electrónica es el médico quien escribe directamente en el programa informático, y ello mejora el sistema si la prescripción electrónica se integra en la historia clínica del paciente con las ventajas esto brinda.

El objetivo del presente proyecto es diseñar e implementar un sistema que permita aumentar el seguimiento de los pacientes y de su adherencia de los medicamentos mediante la receta electrónica, y permite al profesional médico mantener un mejor conocimiento del consumo de los fármacos por parte del paciente. Esto permitirá al sistema de salud tomar acciones tempranas en caso de no adherencia. El sistema deseado fue nombrado SIstema Único de Receta Electrónica (SIURE).

Se inicia el trabajo estudiando los antecedentes publicados de sistemas de receta electrónica donde se destaca el desarrollo de investigación que se realizan en varios países como en España. Este proyecto se enmarca en el contexto de colaboración entre la Escuela Universitaria de Tecnología Médica, el Núcleo de Ingeniería Biomédica y la Facultad de Ingeniería de la UdelaR.

El documento se encuentra separado en 12 capítulos. La introducción contextualiza el tema referido a la receta electrónica. El estado del arte muestra el conocimiento acumulado sobre dicho tema. El tercer capítulo se refiere a la especificación relevada a partir de las reuniones con la contraparte de Registros Médicos. En el cuarto y quinto capítulo se describen los requisitos bajo forma de Casos de Uso. El sexto capítulo presenta la arquitectura general, la distribución física y el funcionamiento interno de los distintos componentes. En el séptimo capítulo se expone la implementación de algunas funcionalidades mediante diagramas de estado. En el octavo capítulo se encuentra la implementación de SIURE en cuanto a herramientas y lenguajes utilizados para el desarrollo del sistema. En el noveno capítulo se describen las pruebas propuestas por el cliente para la aceptación del sistema. El décimo capítulo trata sobre la gestión del proyecto, esto incluye las horas destinadas al trabajo, el manejo de recursos y el plan de desarrollo. En el onceavo capítulo se incluye un manual de uso del sistema, mostrando imágenes con las funcionalidades del mismo. Finalmente, en el doceavo capítulo se exhiben las conclusiones del proyecto y el análisis del resultado obtenido.

2 Estado del arte

2.1 Definición de receta electrónica

Por receta electrónica entendemos [6] una modalidad de servicio digital de apoyo a la asistencia sanitaria que permite al facultativo emitir y transmitir prescripciones por medios electrónicos, basados en las tecnologías de la información y comunicaciones, y que posteriormente pueden ser objeto de dispensación.

Junto con la receta electrónica, es necesario abordar las condiciones de prescripción y dispensación: por ejemplo, la categorización de medicamentos en dos tipos, aquellos que pueden dispensar sin receta o son medicamentos de venta libre, y aquellos que requieren de receta para su dispensación.

En Uruguay los medicamentos que requieren receta se dividen en tres [7], distinguidos por sus colores: la receta naranja (a nivel privado) o amarilla (a nivel público) para la prescripción de estupefacientes; la receta verde (a nivel privado) o celeste (a nivel público) para la prescripción de psicofármacos; y la receta blanca para la prescripción de medicamentos no controlados incluidos en las categorías anteriores y de medicamentos de venta libre.

Para las búsquedas bibliográficas realizadas fueron utilizadas las palabras clave: “Electronic Prescription”, “Receta electrónica en Uruguay”, etc. Las bases de datos bibliográficas analizadas fueron PUBMED, MEDLINE, TIMBO.

2.2 Europa

Durante los años 2008 a 2013, se impulsó el Proyecto epSOS [8] (European Patients Smart Open Services) el cual es considerado el mayor proyecto de la región en materia de salud electrónica.

El objetivo del proyecto era mejorar la atención sanitaria de las personas cuando se encontraban fuera de su país permitiendo que los profesionales de la salud del país donde estuviera la persona pudieran acceder a sus datos médicos para brindarle un tratamiento más adecuado.

2.3 España

En España fue aprobada en 2003 una Ley que atribuye aspectos relacionados con la receta médica electrónica. Atribuye al Ministerio de Sanidad, Servicios Sociales e Igualdad (MSSSI) la competencia de [6] garantizar la interoperabilidad y la

circulación de los datos de receta electrónica a través de la intranet sanitaria, además del desarrollo del sistema de información sanitaria en el SNS incorporando, entre otros, datos de farmacia y productos sanitarios.
El real decreto 16/2012 incluye medidas urgentes para garantizar la sostenibilidad del SNS y mejorar la calidad y seguridad de las prestaciones introduce modificaciones al Real Decreto 1718/2010 de receta médica y órdenes de dispensación. MARCO REGULATORIO Subdirección General de Información Sanitaria e Innovación – Área de Receta Electrónica 3.

Como objetivos se plantearon:

- Conseguir que el ciudadano pueda obtener su medicación en cualquier oficina de farmacia del país, independientemente del lugar donde le hayan realizado la prescripción, utilizando la receta electrónica.
- Evitar tareas administrativas al prescriptor en desplazamientos de pacientes fuera de la comunidad autónoma.
- Avanzar en la implantación de sistemas de información, que permitan que las comunidades autónomas conozcan las transacciones que se realizan entre ellas.
- Disponer de un sistema integrado de receta electrónica en el SNS.

En España el problema principal era la interoperabilidad entre las comunidades autónomas a lo largo del territorio nacional, ocasionando que un ciudadano no pueda obtener su medicación en una oficina de farmacia situada en una comunidad autónoma diferente a aquella donde le han realizado la prescripción.
El proyecto de interoperabilidad de la receta electrónica en el Sistema Nacional de Salud (RESNS), coordinado por el MSSSI, que consiste en:

- realizar una serie de tareas internas en los sistemas informáticos de las CCAA que les permita transmitir la información contenida en la receta médica de una forma entendible, es decir, interoperable,
- efectuar un análisis de criterios comunes y adaptar los sistemas de prescripción y dispensación para incorporarlos, y
- desarrollar los procesos necesarios que permitan la incorporación de pacientes.

2.4 Estados Unidos

Estados Unidos es otro ejemplo de éxito en el desarrollo de la prescripción electrónica y la receta médica electrónica donde el uso de la receta se ha extendido en poco tiempo.

El sistema **E-Prescribing** [9] empezó a ver sus primeras prescripciones electrónicas en el año 2006, donde llegando al año 2014 todos los médicos ya podían emitir

recetas electrónicas, superando el 70% de las prescripciones en formato electrónico en 28 estados.

2.5 Chile

En Chile [7], la Ley modificó el código sanitario en materia de regulación de farmacias y medicamentos introduciendo en el artículo 101 la posibilidad de emitir recetas electrónicas.

La norma citada deja a decisión del paciente el método de creación de la receta, manteniendo la forma habitual (en formato gráfico) o el formato electrónico. Si se decide utilizar el formato electrónico, esta debe ser generada con una firma electrónica avanzada.

2.6 Uruguay

En Uruguay se está viviendo un cambio de paradigma [7], que es el proceso de desmaterialización de los registros sanitarios al que estamos asistiendo desde finales del siglo pasado.

Como la sociedad está cambiando en todo lo relacionado a lo electrónico como, por ejemplo, las relaciones comerciales, los medios de pago electrónico, la transferencia electrónica de fondos, el documento de identidad electrónico, es esperable que se esté hablando de la historia clínica electrónica y una parte de esto es la prescripción médica electrónica.

Actualmente se encuentra en el Poder Ejecutivo un proyecto de ley que establece: “La receta médica electrónica se considera plenamente admisible, válida y eficaz de conformidad con lo dispuesto por la Ley Nº 18.600, de 21 de setiembre de 2009, cumpliendo con los siguientes contenidos mínimos: forma farmacéutica, posología, vía de administración y concentración del medicamento implicado, identificación del prescriptor, identificación del usuario y vigencia en función de la fecha de expedición de la receta. El Poder Ejecutivo reglamentará los procesos electrónicos para la prescripción, la expedición y el control de las recetas electrónicas de estupefacientes y psicofármacos”.

Es necesario tener en cuenta que desde 2009 en nuestro país la receta médica electrónica es un documento electrónico que posee plena validez legal de acuerdo con la Ley Nº 18.600 de documento y firma electrónicos.

Por otro lado, ya podemos ver que el Casmu [10] incorporó el recetario electrónico. Este sistema cuenta un médico farmacólogo que está viendo prácticamente casi de inmediato durante el día la receta por si hay interacción de los medicamentos. Esto

permite informarle al médico tratante si está ignorando una interacción medicamentosa que pueda poner en riesgo al paciente.

3 Objetivo

El proyecto SIURE se enmarca en una línea de investigación, relevamiento de datos y desarrollo sobre la Receta Médica Electrónica. El objetivo es desarrollar un software que permita cumplir las necesidades de la prescripción de medicamentos a partir de los requerimientos relevados junto al grupo de estudiantes de Registros Médicos.

Además, el software debe brindar funcionalidades que ayuden tanto a los profesionales médicos como a los pacientes de cada institución médica. Las principales funcionalidades que debe proveer se centralizan en el seguimiento a la adherencia del consumo de un medicamento por parte del paciente y en la creación de las recetas por parte de los médicos.

Es de gran ayuda para los profesionales médicos poder mejorar el seguimiento al paciente de alguna forma. Interesa también mantener un histórico a lo largo de un tratamiento para cada paciente que pueda llevar un registro del consumo de medicamentos del paciente en cada intervalo.

El software desarrollado debe además contemplar la interoperabilidad con la HCEN de Salud.uy para anexar los datos de su adherencia al consumo de medicamentos a su historia clínica electrónica. Para lograr esto se deben generar datos que cumplan con el mismo estándar de documentación que cumplen los datos de la HCEN de Salud.uy. Estos datos se deberán almacenar cumpliendo el estándar de documentación CDA [11] el cual está basado en el uso de archivos XML.

Para el paciente se deberá brindar un mecanismo de ayuda para el consumo de sus medicamentos. Este mecanismo puede significar una forma de notificación a la hora del consumo. De esta forma se pueden reducir las veces que los pacientes olvidan consumir sus medicamentos en hora. La información correspondiente a cada consumo se debe almacenar en el software de modo de generar los documentos CDA para integrarlos con la HCEN.

Dado que los datos de las recetas y de los consumos de medicamentos son datos privados de los pacientes y de los médicos se deben brindar mecanismos de seguridad. Se debe de autorizar sólo a quien realmente tiene el permiso de acceder a los datos, privando de acceso a cualquiera que no sea usuario de SIURE. Finalmente se deben ocultar los datos de modo que no sean comprendido e interpretado a simple vista, salvo en los extremos de la comunicación.

4 Especificación

4.1 Alcance

SIURE cuenta con dos partes importantes: (1) la aplicación web donde el personal del instituto médico podrá registrar usuarios, prescribir medicamentos, generar notificaciones a los usuarios discriminando por filtros, y generar informes y (2) tenemos la aplicación móvil, que le permite al paciente loguearse, registrar eventos, además genera alarmas según lo prescripto por el médico. Estas funcionalidades de ambas aplicaciones se restringen según roles, Médico, Administrador, Atención al Usuario, Dirección, Informática de la Institución y Paciente.
Las operaciones asociadas a cada rol:

- Usuario: Editar Datos del Usuario
- Administrador y Dirección: Editar Datos del Usuario, Registrar Usuario, Ver Medicamentos, Ver Composiciones del Medicamento, Ver Líneas de la Receta, Ver Tipos de Composición, Ver Adherencia y Documento PDF de Adherencia, Enviar Notificaciones y Dar de Baja un Usuario
- Atención al Usuario: Editar Datos del Usuario, Registrar Usuario y Dar de Baja un Usuario
- Informática de la Institución: Editar Datos del Usuario, Registrar Usuario, Ver Medicamentos, Ver Composiciones del Medicamento, Ver Líneas de la Receta, Ver Tipos de Composición y Enviar Notificaciones
- Médico: Ver Medicamentos, Ver Composiciones del Medicamento, Ver Líneas de la Receta, Ver Tipos de Composición, Ver Adherencia y Documento PDF de Adherencia, Agregar Línea a la Receta, Crear Receta.

Además, SIURE cuenta con integraciones con los sistemas OpenEMPI [12] para la verificación de la existencia del paciente y SNOMED [13] para obtener la información de nomenclatura de medicamentos. Por otro lado, el sistema SIURE contará con generación automática de documentos CDA de prescripción y de control de consumo o adherencia a los tratamientos farmacológicos. El envío de los documentos CDA al repositorio XDS de Salud.uy no está incluido en el alcance de este proyecto.

4.2 Especificación de requerimientos

Las especificaciones de SIURE fueron definidas en colaboración con los estudiantes de Registros Médicos y están presentadas aquí bajo forma de funcionalidades

requeridas. En una segunda etapa estas funcionalidades fueron llevadas al formato de casos de uso descritos en los capítulos 4 y 5.

4.2.1 Prescribir medicamentos

Entrada: el médico podrá realizar prescripciones de medicamentos a los pacientes a través de la aplicación web, indicando medicamento, el tipo de composición (gr, ml, píldora, pastilla), frecuencia de toma, cantidad en el tipo de composición (2 píldoras), duración del tratamiento y fecha deseada para inicio de ingesta por medicamento (el médico indica una fecha posible en la que el paciente debería empezar con la ingesta del medicamento).
Los medicamentos serán obtenidos y validados por los Servicios Terminológicos, donde también se obtendrán las composiciones asociadas a los medicamentos.
Procesamiento: SIURE guarda la prescripción quedando asociada al usuario para poder ser consultada, se programará una notificación para recordarle al paciente que debería comenzar la ingesta del medicamento (está fecha está relacionada con la fecha deseada que ingresa el médico para cada medicamento)
Salida: SIURE informa al médico que la receta fue guardada correctamente y se le notificará al dispositivo móvil del paciente que cuenta con medicamentos disponibles para el comienzo de la ingesta.

4.2.2 Registrar usuario

Entrada: SIURE permite registrar a los pacientes en SIURE ingresando su cédula de identidad. Si el paciente existe en OpenEMPI (sistema para gestión de usuarios) se auto-completan los datos del usuario, sino, deben ser completados manualmente.
Procesamiento: SIURE registra el paciente, enviando un correo electrónico al e-mail indicado en el registro.
Salida: SIURE informará que se registró correctamente el paciente.

4.2.3 Notificación a los pacientes

Entrada: SIURE permite definir varios filtros (CI, fecha de nacimiento, edades, sexo, nombre y apellido) para enviar mensajes mediante notificaciones a usuarios que cumplan con estos filtros. Posibles mensajes:

- Promoción de eventos de la Institución (formato texto y/o enlace).
- Cambios de procedimientos y/o normativas (formato texto y/o enlace).
- Otros que podrán ser implementados por parte de la Institución (formato texto y/o enlace).

Procesamiento: busca todos los pacientes que cumplan con los filtros y envía una notificación a los dispositivos móviles asociados a cada paciente.
Salida: SIURE muestra un mensaje que se enviaron correctamente las notificaciones.

4.2.4 Dar de baja a un usuario

Entrada: SIURE permite dar de baja a un usuario ingresando su cédula de identidad.
Procesamiento: SIURE dejará inhabilitado el usuario para acceder a las aplicaciones, pero manteniendo los datos del usuario para una futura reintegración.
Salida: SIURE muestra un mensaje que se realizó la baja correctamente.

4.2.5 Reseteo de Contraseña

Entrada: SIURE permite cambiar la contraseña en caso de olvido ingresando cédula de identidad y mail, enviado un mail que cuenta con un link para cambiar la contraseña.
Procesamiento: cuando SIURE recibe un pedido de cambio de contraseña, genera un link que es enviado por mail que identifica al usuario que pide cambiar la contraseña.
Salida: SIURE muestra un mensaje que se cambió correctamente la contraseña y re-dirige al inicio de sesión.

4.2.6 Iniciar sesión

Entrada: todos los usuarios podrán iniciar sesión en la aplicación web usando su cédula de identidad y contraseña, lo mismo en la aplicación móvil (es necesario para está que el usuario tenga el rol paciente).
Procesamiento: según el rol del usuario, se generan los menúes habilitados para el mismo.
Salida: muestra la información pertinente al rol asociado al usuario.

4.2.7 Ver medicamentos nuevos

Entrada: el usuario con acceso a la aplicación móvil podrá ver los nuevos medicamentos prescritos por el médico, donde podrá asignar el comienzo de la automedicación.
Procesamiento: al momento de asignarle una fecha y hora, se auto programarán alarmas que coinciden con los intervalos pre establecidos por el médico (frecuencia de ingestión y duración del tratamiento).
Salida: se muestra en el calendario de la aplicación móvil, todas las alarmas configuradas para este tratamiento.

4.2.8 Agenda

Entrada: el usuario podrá crear recordatorios dentro de la aplicación móvil, ingresando un texto y fecha para ser recordado.

Procesamiento: se guarda la configuración del recordatorio y se genera una alarma que sonará en la fecha indicada.
Salida: se muestra un mensaje que se generó el recordatorio correctamente.

4.2.9 Calendario

Entrada: la aplicación móvil de SIURE cuenta con un calendario, donde muestra los días con marcas de colores correspondiente al evento que tengamos asignado para ese día (ingesta o recordatorio).
Procesamiento: al seleccionar el día, se procesa la información para poder ser mostrada.
Salida: se visualiza datos relevantes al evento o a la automedicación como por ejemplo que medicamento y cuántas unidades debo consumir.

4.2.10 Ya tomé

Entrada: el paciente contará con tres posibilidades para comunicar que tomó medicamento, “Ya tomé” donde indica que se produjo una ingesta de un medicamento, “Posponer” mostrará unos tiempos para que sea comunicado nuevamente el paciente para ingerir el medicamento y “Cancelar” dónde se cancelara la ingesta.
Procesamiento: al momento de indicar “Ya tomé” o “Cancelar” se guardará esta información para poder ser usará en reportes de consumo para uso del médico.
Salida: N/A

4.2.11 Ver perfil de usuario

Entrada: N/A
Procedimiento: N/A
Salida: se muestra la información detallada del usuario (nombres, apellidos, mail, teléfono, sexo). SIURE permite cambiar la contraseña dentro del perfil de usuario.

4.2.12 Ver medicamentos

Entrada: un usuario con permisos tiene la posibilidad de entrar en el “menú de medicamentos”.
Procedimiento: N/A
Salida: muestra los medicamentos que fueron pedidos a sistemas externos.

4.2.13 Ver recetas

Entrada: el usuario logueado debe tener permisos de médico, entonces apara
Procedimiento: N/A
Salida: muestra las recetas que fueron prescritas por el médico.

4.2.14 Ver información de recetas

Entrada: selecciona una receta en la que le interesa saber en detalle el contenido de la misma.
Procedimiento: N/A
Salida: muestra los medicamentos prescritos en la receta, cual es la frecuencia de cada uno, la cantidad de la dosis y detalle general.

4.2.15 Ver adherencia del Usuario

Entrada: para ver la adherencia de un usuario, es necesario buscarlo por cédula de identidad e indicar entre que fechas se requiere ver la adherencia
Procedimiento: N/A
Salida: muestra en una tabla el porcentaje de adherencia que tiene la persona entre el tiempo mencionado en la entrada.

4.3 Especificación de requerimientos no funcionales

Aparte de todos los requerimientos funcionales definidos anteriormente, también forman parte importante del sistema los requerimientos no funcionales.

- La transferencia de datos debe ir bajo protocolos seguros, contar con límites de funciones definidas por roles y autenticaciones en el intercambio de datos.
- La instalación de SIURE debe ser independiente del sistema operativo sobre el cual va a ser ejecutado.
- El sistema debe proporcionar mensajes de error que sean informativos y orientados a usuario final.
- La aplicación móvil debe poder ejecutarse en dispositivos Android de forma nativa.

5 Casos de uso: Web

Los casos de usos fueron diseñados usando los flujos descritos por los estudiantes de Registros Médicos. La metodología usada fue, anotar paso por paso qué funciones o acciones esperarían los usuarios al usar la aplicación. Por problemas de tiempo se decidió no incluir los diagramas UML para el desarrollo de los casos de usos. A continuación, se pasan a detallar los casos de uso del sistema donde quedan asociados a requerimientos funcionales detallados en el capítulo 3.

5.1 Iniciar Sesión

Descripción
Un usuario registrado desea ingresar al sistema, ver especificación SIURE párrafo 4.2.6.

Pre-condiciones
El usuario no ha iniciado sesión en el sistema.

Flujo de eventos principal
El usuario ingresa a la web.

1. El sistema despliega la pantalla de inicio con el formulario para ingresar su C.I y su password.
2. El usuario ingresa los datos y presiona “Login”.
3. El sistema valida los datos ingresados y redirige al usuario a la página de su perfil mostrando sus datos.
4. Fin.

Flujo de eventos alternativos
[*Datos Inválidos*]

1. El usuario ingresa a la web.
2. El sistema despliega la pantalla de inicio con el formulario para ingresar su C.I y su password.
3. El usuario ingresa los datos y presiona “Login”.
4. El sistema verifica que el usuario no existe o su password es incorrecta y muestra en pantalla un mensaje de error.
5. Fin.

Post-condiciones
Se crea una sesión para el usuario en el sistema y se redirige al usuario a su perfil mostrando sus datos.

5.2 Ver Perfil del Usuario

Descripción
Un usuario ve o modifica sus datos, ver especificación SIURE párrafo 4.2.11.

Pre-condiciones
El usuario ha iniciado sesión previamente en el sistema.

Flujo de eventos principal
1. El usuario presiona el ítem del menú “Perfil”.
2. El sistema redirige al usuario a la página de perfil.
3. Fin.

Flujo de eventos alternativo
[*Modificar Datos del Usuario*]
1. El usuario presiona el ítem del menú “Perfil”.
2. El usuario presiona el botón “Editar”.
3. El usuario edita los datos y presiona el botón “Guardar”.
4. El sistema guarda los datos del usuario y muestra un mensaje de información.
5. Fin.

Post-condiciones
[*Modificar Datos del Usuario*]
Los datos del usuario son editados y guardados en la base de datos.

5.3 Ver Medicamentos

Descripción
El usuario con los permisos necesarios ve los medicamentos del sistema, ver especificación SIURE párrafo 4.2.12.

Pre-condiciones
El usuario posee al menos un rol que lo autorice a ingresar a esta página.

Flujo de eventos principal
1. El usuario presiona el ítem del menú “Medicamentos” y luego presiona el subitem “Ver Medicamentos”.

2. El sistema redirige al usuario a la página correspondiente donde se listan los medicamentos y los Tipos de Composición del sistema.
3. Fin.

Flujo de eventos alternativos
No hay.

Post-condiciones
No hay.

5.4 Ver Recetas

Descripción
Un usuario con los permisos necesarios ve las recetas creadas en el sistema, ver especificación SIURE párrafo 4.2.13.

Pre-condiciones
El usuario posee los permisos necesarios para ver las recetas del sistema.

Flujo de eventos principal
1. El usuario presiona el ítem del submenú “Medicamentos” y luego presiona en el submenú “Recetas”.
2. El sistema lo redirige a la página correspondiente donde puede ver las recetas existentes.
3. Fin.

Flujo de eventos alternativos
No hay.

Post-condiciones
No hay.

5.5 Ver Información de Receta

Descripción
Un usuario con los permisos necesarios para realizar el caso de uso anterior ve la información de una receta en particular, ver especificación SIURE párrafo 4.2.14.

Pre-condiciones
El usuario posee los permisos necesarios para ver las recetas del sistema.

Flujo de eventos principal {<<extends>> 4.1.5}
1. El usuario presiona “Ver Detalle” para la receta que desee inspeccionar.

2. El sistema despliega un popup con la información de dicha receta.
3. Fin.

Flujo de eventos alternativos
No hay.
Post-condiciones
No hay.

5.6 Prescribir medicamentos

Descripción
Un usuario con el rol “Médico” ingresa una nueva receta para un paciente determinado en el sistema, ver especificación SIURE párrafo 4.2.1.

Precondición
El usuario posee el rol “Médico”.

Flujo de eventos principal {<<extends>> 4.1.5}
1. El usuario presiona en “Agregar Receta”.
2. El sistema despliega un popup donde el usuario ingresa la C.I del paciente al cual desea recetar medicamentos.
3. El usuario ingresa la C.I y presiona “Verificar Paciente”.
4. El sistema muestra un formulario en el mismo popup donde el usuario debe ingresar el nombre del medicamento a buscar, en caso de que no exista un usuario con el rol “Usuario” cuya C.I coincida con la ingresada, se despliega un mensaje de error.
5. El usuario ingresa el nombre del medicamento y presiona “Buscar”.
6. El sistema lista los medicamentos encontrados y en caso de no encontrar ninguno despliega un mensaje que informa lo sucedido.
7. El usuario selecciona uno presionando en “Seleccionar”.
8. El sistema verifica si el medicamento posee una composición y despliega en pantalla un combo para seleccionar la composición deseada, caso contrario despliega un formulario para crearla.
9. El usuario ingresa los datos y presiona “Crear Composición”.
10. El sistema despliega otro formulario para ingresar los datos correspondientes al consumo del medicamento seleccionado asociado a la composición creada.
11. El usuario ingresa los datos y presiona “Confirmar”.
12. El sistema guarda los datos y despliega en el mismo popup una tabla con los medicamentos agregados hasta el momento.
13. El usuario puede borrar las líneas creadas con el botón “Borrar Línea” o agregar más líneas a la receta presionando el botón “Agregar Línea” en cuyo

caso se vuelve a iniciar el caso de uso. Para finalizar la creación de la receta el usuario presiona “Guardar Receta”.

14. Fin.

Flujo de eventos alternativos
No hay.

Post-condiciones
Se crea una receta en el sistema para el paciente con los medicamentos seleccionados y se envía una notificación al móvil del paciente informando la adjudicación de medicamentos.

5.7 Enviar Notificación

Descripción
Un usuario con los permisos necesarios envía una notificación a uno o a muchos usuarios, ver especificación SIURE párrafo 4.2.3.

Pre-condiciones
El usuario tiene permisos suficientes para enviar notificaciones.

Flujo de eventos principal

1. El usuario presiona el ítem del menú “Notificaciones”.
2. El sistema lo redirige a la página correspondiente donde se despliega un formulario para el ingreso de datos por parte del usuario para filtrar los usuarios a los que desea enviar la notificación.
3. El usuario ingresa los datos y presiona “Notificar”.
4. El sistema muestra en pantalla un popup donde el usuario ingresa el mensaje a enviar.
5. El usuario ingresa el mensaje y presiona “Enviar Notificación”.
6. Fin.

Flujo de eventos alternativos
No hay.

Post-condiciones
Se envía la notificación a los smartphones de los usuarios filtrados.

5.8 Ver Adherencia del Usuario

Descripción
Un usuario ve la adherencia a un medicamento de un usuario en particular en un periodo dado, ver especificación SIURE párrafo 4.2.15.

Pre-condiciones
No hay.

Flujo de eventos principal
1. El usuario presiona el ítem del menú “Reportes” y luego presiona “Consumo”.
2. El sistema lo redirige a la página correspondiente donde se despliega un formulario para el ingreso de la C.I del paciente, y para ingresar la fecha de inicio del periodo y la de fin.
3. El usuario ingresa los datos correspondientes (los cuales son obligatorios) y presiona “Buscar”.
4. El sistema lista las adherencias del paciente seleccionado en una tabla con los datos asociados a los mismos.
5. Fin.

Flujo de eventos alternativos
[*El usuario no ingresa al menos un dato de los requeridos*]
1. El usuario presiona el ítem del menú “Reportes” y luego presiona “Consumo”.
2. El sistema lo redirige a la página correspondiente donde se despliega un formulario para el ingreso de la C.I del paciente, y para ingresar la fecha de inicio del periodo y la de fin.
3. El usuario no ingresa al menos uno de los datos requeridos y presiona “Buscar”.
4. El sistema muestra en pantalla un mensaje de error por cada dato requerido que no fue ingresado.
5. Fin.

Post-condiciones
No hay.

5.9 Cambiar contraseña

Descripción
Un usuario cambia su contraseña actual por una nueva, ver especificación SIURE párrafo 4.2.11.

Pre-condiciones

El usuario ha iniciado sesión en el sistema.

Flujo de eventos principal

1. El usuario presiona “Cambiar Contraseña”.
2. El sistema despliega un popup donde debe ingresar su nueva contraseña y su confirmación.
3. El usuario ingresa ambas y presiona “Guardar”.
4. El sistema guarda los datos y cierra el popup.
5. Fin.

Flujo de eventos alternativos

[*Las contraseñas no coinciden*]

1. El usuario presiona “Cambiar Password”.
2. El sistema despliega un popup donde debe ingresar su nueva contraseña y su confirmación.
3. El usuario ingresa ambas las cuales son diferentes y presiona “Guardar”.
4. El sistema muestra un mensaje de error en el popup.
5. El usuario puede corregir su error y reintentar o presionar “Cancelar” para salir del caso de uso.
6. Fin.

Post-condiciones

Cambia la password del usuario a la que el usuario haya ingresado.

5.10 Dar de Baja un Usuario

Descripción

Un usuario con los permisos necesarios da de baja un usuario del sistema, ver especificación SIURE párrafo 4.2.4.

Pre-condiciones

El usuario ha iniciado sesión en el sistema.

Flujo de eventos principal

1. El usuario presiona el ítem del menú “Usuarios” y luego presiona el subitem “Dar de Baja Usuario”.
2. El sistema muestra un popup en pantalla donde el usuario debe ingresar la C.I del usuario y verificarla.
3. El usuario ingresa la C.I del usuario a dar de baja y presiona “Verificar Usuario”.
4. El sistema verifica que exista un usuario cuya C.I coincida con la C.I ingresada y muestra dos botones para confirmar o cancelar la dada de baja del usuario.

5. El usuario presiona “Dar de Baja”.
6. El sistema da de baja al usuario seleccionado y cierra el popup.
7. Fin.

Flujo de eventos alternativos
[*No existe un usuario cuya C.I coincida con la ingresada*]

1. El usuario presiona el ítem del menú “Usuarios” y luego presiona el subitem “Dar de Baja Usuario”.
2. El sistema muestra un popup en pantalla donde el usuario debe ingresar la C.I del usuario y verificarla.
3. El usuario ingresa la C.I del usuario a dar de baja y presiona “Verificar Usuario”.
4. El sistema verifica que no existe un usuario cuya C.I coincida con la ingresada y muestra un mensaje de error en el popup.
5. El usuario puede volver a ingresar una C.I y verificarla o presionar “Cancelar” para salir del caso de uso.
6. Fin.

Post-condiciones
Se da de baja el usuario seleccionado del sistema.

5.11 Resetear contraseña

Descripción
Un usuario que olvidó su contraseña la resetea, ver especificación SIURE párrafo 4.2.5.

Pre-condiciones
No hay.

Flujo de eventos principal

1. El usuario ingresa a la web y presiona “Olvide mi Contraseña”.
2. El sistema despliega un popup en pantalla con un formulario para ingresar la C.I y el email asociado al usuario.
3. El usuario ingresa los datos y presiona “Confirmar”.
4. El sistema valida que exista un usuario cuya C.I y e-mail coincidan con los ingresados y envía un correo a el e-mail del usuario.
5. El usuario verifica que le llegó el correo y hace click en el enlace “cambiar”.
6. El sistema lo redirige a la página para resetear su contraseña la cual consta de un formulario donde el usuario debe ingresar la nueva password y su confirmación.
7. El usuario ingresa los datos y presiona “Guardar”.

8. El sistema verifica que las contraseñas coincidan, guarda los cambios y redirige al usuario a la página de login.
9. Fin.

Flujo de eventos alternativos
[*Las contraseñas no coinciden*]

1. El usuario ingresa a la web y presiona “Olvide mi Contraseña”.
2. El sistema despliega un popup en pantalla con un formulario para ingresar la C.I y el email asociado al usuario.
3. El usuario ingresa los datos y presiona “Confirmar”.
4. El sistema valida que exista un usuario cuya C.I y e-mail coincidan con los ingresados y envía un correo a el e-mail del usuario.
5. El usuario verifica que le llegó el correo y hace click en el enlace “cambiar”.
6. El sistema lo redirige a la página para resetear su contraseña la cual consta de un formulario donde el usuario debe ingresar la nueva contraseña y su confirmación.
7. El usuario ingresa las passwords las cuales no coinciden y presiona “Guardar”.
8. El sistema valida que las passwords no coinciden y despliega un mensaje de error.
9. Fin.

[*No existe un usuario cuya C.I y/o e-mail coincidan con los ingresados*]

1. El usuario ingresa a la web y presiona “Olvide mi Contraseña”.
2. El sistema despliega un popup en pantalla con un formulario para ingresar la C.I y el email asociado al usuario.
3. El usuario ingresa los datos y presiona “Confirmar”.
4. El sistema verifica que no existe usuario cuya C.I y e-mail coincidan con los ingresados y no envía el correo.
5. Fin.

Post-condiciones
Se ha enviado un correo al email ingresado por el usuario y se cambia la contraseña por la ingresada por el mismo.

5.12 Registrar Usuario

Descripción
Un usuario con los permisos necesarios registra un nuevo usuario en el sistema, ver especificación SIURE párrafo 4.2.2.

Pre-condiciones

El usuario ha iniciado sesión en el sistema y el usuario posee permisos necesarios para registrar a otro usuario en el sistema.

Flujo de eventos principal

1. El usuario presiona el ítem del menú “Usuarios” y luego en “Registrar Usuario”.
2. El sistema lo redirige a la página correspondiente donde deberá ingresar la C.I del usuario a registrar para verificar la existencia de un usuario de OpenEMPI con dicha C.I.
3. El usuario ingresa la C.I del nuevo usuario a registrar y presiona “Verificar en OpenEMPI”.
4. El sistema se comunica con OpenEMPI para obtener los datos personales de un usuario de OpenEMPI cuya C.I coincida con la ingresada y despliega en pantalla otro formulario con campos para editar los datos obtenidos de OpenEMPI.
5. El usuario ingresa los roles del usuario presionando “Agregar Rol” procediendo de la siguiente manera:
    - Presiona el botón “Agregar Rol”.
    - El sistema abre un popup en pantalla con un combo que contiene los roles posibles a elegir.
    - El usuario selecciona uno y presiona “Guardar”.
    - El sistema agrega el rol a la lista de roles del usuario a registrar.
6. El usuario puede ver los roles del sistema en un campo del formulario o presionando en “Ver Roles” pudiendo borrar los roles presionando en “Borrar Rol” en el popup que el sistema despliega
7. El usuario modifica los datos y presiona “Registrar”
8. El sistema registra al nuevo usuario en el sistema, le envía un correo al email ingresado y redirige al usuario a la página de inicio
9. Fin

Flujo de eventos alternativos

[*Usuario no existente en OpenEMPI*]

1. El usuario presiona el ítem del menú “Usuarios” y luego en “Registrar Usuario”.
2. El sistema lo redirige a la página correspondiente donde deberá ingresar la C.I del usuario a registrar para verificar la existencia de un usuario de OpenEMPI con dicha C.I.
3. El usuario ingresa la C.I del nuevo usuario a registrar y presiona “Verificar en OpenEMPI”.
4. El sistema se comunica con OpenEMPI para obtener los datos personales de un usuario de OpenEMPI cuya C.I coincida con la ingresada y no obtiene resultados, por lo que despliega un formulario con campos para ingresar los datos del nuevo usuario a mano.

5. El usuario ingresa los roles del usuario presionando “Agregar Rol” procediendo de la siguiente manera:
    - Presiona el botón “Agregar Rol”.
    - El sistema abre un popup en pantalla con un combo que contiene los roles posibles a elegir.
    - El usuario selecciona uno y presiona “Guardar”.
    - El sistema agrega el rol a la lista de roles del usuario a registrar.
6. El usuario puede ver los roles del sistema en un campo del formulario o presionando en “Ver Roles” pudiendo borrar los roles presionando en “Borrar Rol” en el popup que el sistema despliega.
7. El usuario ingresa los datos y presiona “Registrar”.
8. El sistema registra al nuevo usuario en el sistema, le envía un correo al email ingresado y redirige al usuario a la página de inicio.
9. Fin.

Post-condiciones

Se envía un correo al email del usuario registrado y se crea un nuevo usuario en el sistema.

6 Casos de uso: Aplicación Móvil

6.1 Iniciar Sesión

Descripción
Un usuario registrado desea ingresar al sistema, ver especificación SIURE párrafo 4.2.6.

Pre-condiciones
El usuario no ha iniciado sesión en el sistema.

Flujo de eventos principal

1. El usuario ingresa a la aplicación.
2. El sistema despliega la pantalla de inicio con el formulario para ingresar su C.I y su password.
3. El usuario ingresa los datos y presiona “Login”.
4. El sistema valida los datos ingresados y redirige al usuario a la página de su perfil mostrando sus datos.
5. Fin.

Flujo de eventos alternativos
[*Datos Inválidos*]

1. El usuario ingresa a la aplicación.
2. El sistema despliega la pantalla de inicio con el formulario para ingresar su C.I y su contraseña.
3. El usuario ingresa los datos y presiona “Iniciar Sesión”.
4. El sistema verifica que el usuario no existe o su contraseña es incorrecta y muestra en pantalla un mensaje de error.
5. Fin.

Post-condiciones
Se crea una sesión para el usuario en el sistema, se crea la base de datos SQLite para el usuario y se redirige al usuario a la pantalla de Inicio, luego se piden los datos del usuario a la Capa de Servicios.

6.2 Crear recordatorio en la Agenda

Descripción

Un usuario crea un recordatorio en el sistema, ver especificación SIURE párrafo 4.2.8.

Pre-condiciones
El usuario ha iniciado sesión en el sistema.

Flujo de eventos principales
1. El usuario presiona en la fecha deseada para el evento.
2. El sistema despliega un popup para ingresar los datos del evento (nombre y fecha de inicio).
3. El usuario ingresa los datos y presiona “Nuevo Evento”.
4. El sistema guarda el nuevo evento, crea una nueva alarma para ese evento, cierra el popup y actualiza el calendario en la pantalla de Inicio.
5. Fin.

Flujo de eventos alternativos
No hay.

Post-condiciones
Se crea un evento en el sistema y se crea una alarma para dicho evento.

6.3 Ver Medicamentos Nuevos Asignados

Descripción
Un usuario ve los medicamentos nuevos que se le han asignado, ver especificación SIURE párrafo 4.2.1.

Pre-condiciones
El usuario ha iniciado sesión en el sistema.

Flujo de eventos principal
1. El usuario accede al menú lateral izquierdo y presiona en el ítem “Medicamentos Nuevos”.
2. El sistema lo dirige a la pantalla correspondiente listando los medicamentos asignados al usuario.
3. El usuario puede confirmar el inicio del consumo de medicamentos presionando “Confirmar Inicio”.
4. Fin.

Flujo de eventos alternativos
No hay.

Post-condiciones

No hay.

6.4 Confirmar Inicio del Consumo del Medicamento

Descripción
Un usuario confirma el inicio de consumo de un medicamento, ver especificación SIURE párrafo 4.2.1.

Pre-condiciones
El usuario ha iniciado sesión en el sistema.

Flujo de eventos principal {<<extends>> 4.2.3}
1. El usuario presiona en “Confirmar Inicio”.
2. El sistema despliega un popup para confirmar la fecha de inicio.
3. El usuario ingresa la fecha de inicio y presiona “Confirmar”.
4. El sistema genera y guarda las alarmas asociadas al consumo de dicho medicamento y redirige al usuario a la pantalla de Inicio.
5. Fin.

Flujo de eventos alternativos
No hay.

Post-condiciones
Se crean y se guardan las alarmas asociadas al consumo del medicamento en el sistema.

6.5 Ver Agenda del Día

Descripción
Un usuario ve los recordatorios y consumos de medicamentos del día, ver especificación SIURE párrafo 4.2.9.

Pre-condiciones
El usuario ha iniciado sesión en el sistema.

Flujo de eventos principal
1. El usuario presiona en el día que desea consultar.
2. El sistema muestra un popup en pantalla con los eventos y consumos de medicamentos del día.
3. Fin.

Flujo de eventos alternativos
No hay.

Post-condiciones
No hay.

6.6 Ver Medicamentos del Paciente

Descripción
Un usuario desea ver los medicamentos que está consumiendo actualmente, ver especificación SIURE párrafo 4.2.12.

Pre-condiciones
El usuario ha iniciado sesión en el sistema.

Flujo de eventos principal
1. El usuario abre el menú lateral izquierdo y presiona el ítem “Medicamentos”.
2. El sistema lo redirige a la pantalla correspondiente listando los medicamentos que está consumiendo actualmente.
3. El usuario puede ver los detalles del medicamento y de su consumo presionando “Ver Detalles”.
4. Fin.

6.7 Confirmar adherencia al tratamiento

Descripción
Un usuario confirma el consumo de un medicamento, ver especificación SIURE párrafo 4.2.10.

Pre-condiciones
Se ejecutó la alarma para el consumo del medicamento y el usuario ha iniciado sesión en el sistema.

Flujo de eventos principal
1. El sistema muestra la pantalla de confirmación del consumo del medicamento.
2. El usuario presiona “Confirmar”.
3. El sistema confirma la ejecución de la alarma, y luego guarda y envía los datos a la Capa de Servicios y cierra la pantalla.
4. Fin.

Flujo de eventos alternativos

[*Postergar el consumo del medicamento*]

1. El sistema muestra la pantalla de confirmación del consumo del medicamento.
2. El usuario presiona “Postergar”.
3. El sistema despliega un popup en pantalla con un combo para seleccionar los minutos que se requiera postergar el consumo pudiendo elegir entre 5, 10 o 15 minutos.
4. El usuario selecciona los minutos y presiona “Confirmar”.
5. El sistema confirma la ejecución de la alarma, crea una alarma correspondiente a la postergación del consumo y cierra la pantalla.
6. Fin.

[*Cancelar el consumo del medicamento*]

1. El sistema muestra la pantalla de confirmación del consumo del medicamento.
2. El usuario presiona “Cancelar”.
3. El sistema confirma la ejecución de la alarma y cierra la pantalla.
4. Fin.

Post-condiciones

Se envía la información del consumo a la Capa de Servicios.

7 Arquitectura

7.1 Componentes de SIURE

La arquitectura de SIURE está formada por varios componentes. El componente EAR [14] de la aplicación el cual es un típico contenedor de Java EE [15] que puede ejecutarse sobre un servidor de aplicaciones (por ejemplo: JBoss [16], Tomcat, GlassFish, WebSphere, etc) es el que encierra los siguientes componentes:

- Capa de Acceso a Datos: Esta capa es la encargada de implementar las entidades del sistema las cuales representan los conceptos especificados en los requerimientos del sistema, las mismas son las que se persisten en la base de datos MySQL [17] en tablas y contienen relaciones entre sí. Un ejemplo de ello es la relación entre los roles de un usuario, un usuario tiene uno o más roles en el sistema y cada rol está compuesto por una o más operaciones las cuales restringen el acceso que tienen los usuarios al sistema.
- Capa de Negocio: En esta capa se deposita la lógica de negocio del sistema, la cual es brindada mediante EJBs (Enterprise Java Beans) los cuales forman parte de la implementación de Java EE. Son clases Java diseñadas para encargarse de la lógica de un sistema y poseen ciertas características que los hacen interesantes:
    - No existe la posibilidad de crear una nueva instancia de estas clases, ya que el encargado de manejar el acceso a las instancias de los EJBs es el contenedor EAR [14], la forma de obtener una instancia de estas clases es mediante la inyección de dependencias, la cual se puede lograr de diversas maneras, pero especialmente se puede realizar mediante anotaciones de Java especiales para realizar esto.
    - Estas clases siguen el patrón de diseño Singleton [18] el cual se define como la existencia de una sola instancia de la clase que siga el patrón, por lo que como se mencionó en el punto anterior, el contenedor maneja una única instancia de cada EJB, las cuales se inicializan por demanda (la primer vez que se referencia al EJB) salvo que se explicite que se inicialice al inicio del servidor de aplicaciones (existe la anotación @Startup para ello).
    - Uno no accede realmente a la instancia que maneja el contenedor EAR, sino que se crea un proxy [19] que “representa” la instancia del EJB, por lo que el manejo de los EJBs está restringido al uso de las herramientas que brinda Java EE para la implementación y utilización de los mismos, ya sea mediante anotaciones o inyección de dependencias.

    - Una característica muy importante que brindan y particularmente es importante para un sistema que potencialmente puede tener una cantidad de usuarios no menor como es el caso de SIURE, es la transaccionalidad implícita a nivel de método. Lo anterior significa que todo lo que se realice dentro del método o se ejecuta correctamente en un todo, o no se ejecuta nada. Fácilmente puede verse que esto es muy útil en sistemas de información donde pueden ocurrir múltiples sucesos que disparen excepciones a nivel de los EJB al acceder a la base de datos y se pueda realizar el rollback correspondiente sin siquiera tener que explicitar por ejemplo utilizando el EntityManager de javax.persistence. Es posible no hacer uso de esta característica mediante una anotación.
- Capa de Servicios: Este componente se encarga de exponer servicios tanto para la Aplicación Android como para sistemas externos que deseen integrarse con SIURE en un futuro. Los servicios se exponen a través de web services REST [20]. Esta capa hace uso intensivo de la Capa de Negocios para realizar los impactos requeridos por la invocación de los servicios en la base de datos del sistema.
- Capa Web: Es en este componente que se implementa el Frontend del sistema, el cual es accesible utilizando cualquier navegador web o dispositivo móvil, ya que el diseño es responsive. El uso de JSF [21] brinda un fácil manejo de datos ingresados en formularios, tal y como son los que se utilizaran en SIURE, ya que se establece una relación entre los valores a mostrar en pantalla y acciones a tomar al dar click en un botón, con métodos de Java incrustados en clases llamadas Managed Beans, que forman parte de la Capa Web también y se encargan de brindar una interfaz entre la Capa de Negocio y la Capa Web.
- Aplicación Android: Se implementa una aplicación Android para SIURE que se encarga del manejo de alarmas y del disparo de las mismas, así como también de la creación de eventos personalizados por el usuario para utilizar como recordatorios. Todo se persiste en una base de datos SQLite [22] asociada a la aplicación donde se guardan los datos correspondientes al usuario. A su vez la aplicación Android se divide en 4 subcomponentes:
    - Main UI Thread: Este es el hilo principal de la aplicación, el que controla la interfaz gráfica y las Activities de Android, una a la vez dependiendo de la que esté activa.
    - Background Thread: Hilo que corre por debajo del hilo principal, utilizado para iniciar sesión en el sistema y para la llamada a web services.
    - Capa de Acceso a Datos: Consiste en una clase llamada “SQLiteHelper” que brinda herramientas para el manejo de bases de datos SQLite, se utilizan clases propias de SIURE que hacen uso de SQLiteHelper para implementar el acceso a datos.

    - Cliente Http Rest: Cliente Rest que se encarga de hacer los pedidos a la Capa de Servicios a través de web services REST, accediendo al servicio requerido por su URL y pasando los parámetros que corresponda.
- Google Cloud Messaging [23]: Es un middleware que se utiliza en SIURE para el envió de notificaciones push en Android. Se realiza una petición HTTP GET y mediante el envío de un JSON con el contenido del mensaje se envía la notificación. La notificación puede enviarse a todos los dispositivos, como a uno en particular utilizando un token que genera Google play services el cual se pide cuando se inicia sesión en SIURE y luego se guarda en la base de datos de SIURE mediante la invocación de un servicio en la Capa de Servicios mediante la aplicación Android.
- Web Service SOAP Snomed : Se hace uso de un web service SOAP [24] de Snomed [13] que se ejecuta sobre la plataforma de Salud.uy [25] para consultar datos sobre medicamentos a la hora de recetar a un paciente por parte del médico.
- Open EMPI [12]: Este componente es utilizado por el sistema para obtener los datos existentes de los usuarios ya registrados en dicho componente, y guardarlos en la base de datos de SIURE a la hora de registrar al nuevo usuario.

Como se puede apreciar en la figura 7.1, vemos el componente EAR de la aplicación el cual es un típico contenedor de Java EE [15] que puede ejecutarse sobre un servidor de aplicaciones (por ejemplo: JBoss [16], Tomcat, GlassFish, WebSphere, etc) es el que encierra los siguientes componentes:

Como se puede apreciar en el diagrama y en la explicación de cada uno de los componentes, se trata de una arquitectura típica de un sistema de información que interactúa con sistemas externos y que contiene dos grandes componentes diferenciables: Componente EAR y Aplicación Android.

Para que fuera posible la implementación de diversos requerimientos fue necesaria la integración con varios sistemas externos, la cual se realizó en todos los casos mediante la llamada a web services que dichos sistemas exponen (uno RESTful por parte de OpenEMPI y uno SOAP por parte de Snomed).

La decisión de exponer los servicios brindados por SIURE utilizando el estándar RESTful en lugar de SOAP se basa principalmente en la diferencia de performance y de carga en la red entre uno y otro.
En un principio se utilizó SOAP para exponer los servicios, pero la performance comparada con la utilización de RESTful fue mucho menor, además de que no es necesaria ninguna librería externa para implementar un cliente REST en Android, pero si es necesario para SOAP (en nuestro caso utilizamos ksoap2 [26], lo cual

brinda menos dependencia de herramientas desarrolladas por terceros, que claramente es una ventaja.

7.2 Interconexión de componentes de SIURE

En la figure 7.1 se muestra el diagrama de arquitectura, que refleja los componentes de SIURE. Es sencillo entender a SIURE mediante el diagrama y su explicación debido a la separación de responsabilidades de cada uno de los componentes.

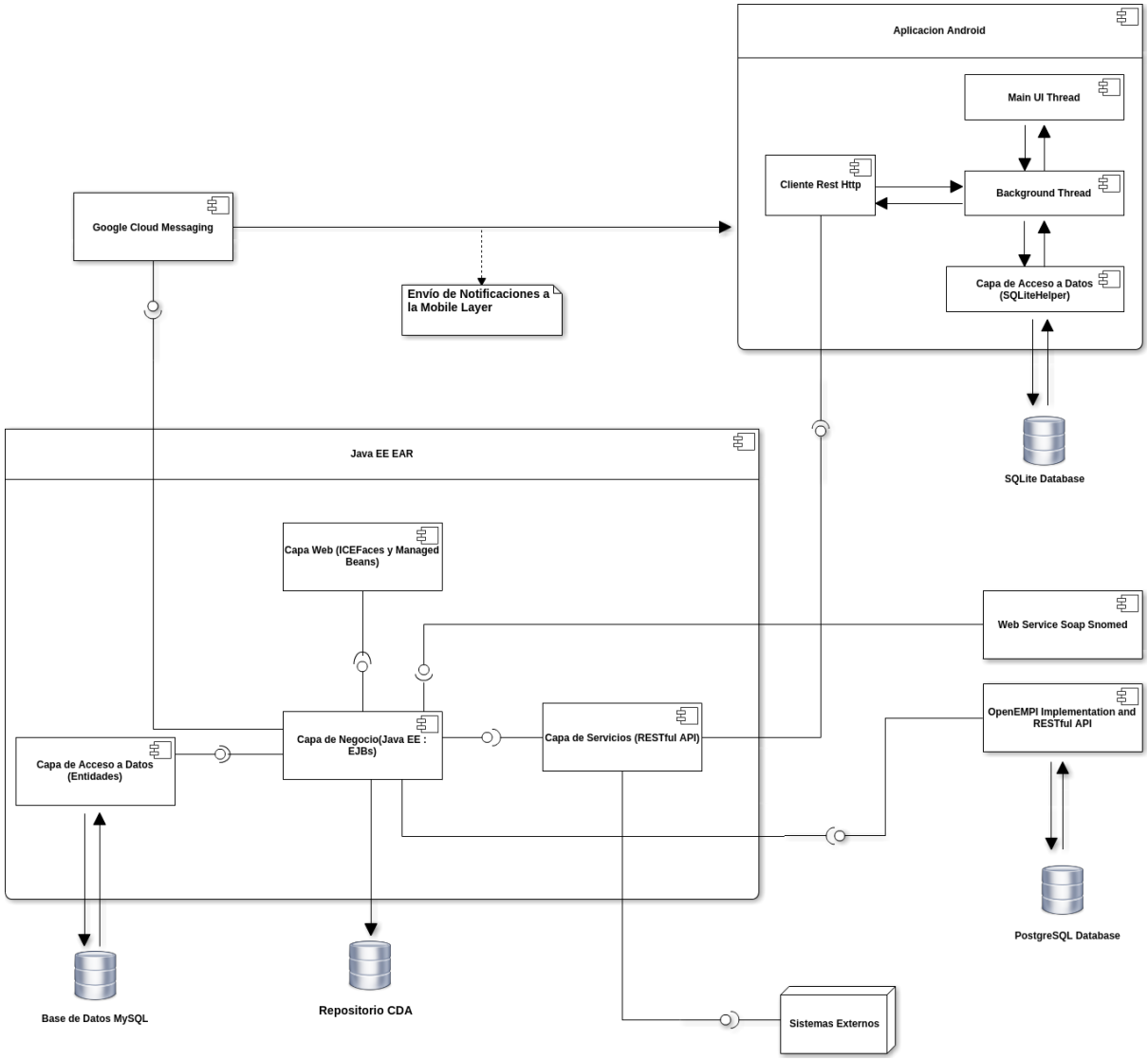

Figura 7.1 - Diagrama de arquitectura

8 Lógica de operación de SIURE

8.1 Creación de la Receta

La funcionalidad principal del sistema es la creación de la receta electrónica mediante la capa web por parte de un usuario con el rol “Médico”. Para que esta funcionalidad sea cumplida es necesario realizar una serie de pasos para ingresar la información de la receta.

El único rol que puede acceder a dicha funcionalidad es el rol “Médico”, ya que es el único que legalmente puede recetar medicamentos a los pacientes. Esto se respeta en nuestro sistema al poder verificar los roles de cada usuario que inicia sesión en SIURE, y luego verificar que dicho usuario posea el rol o no y en función de ello se visualiza el ítem “Recetas” del submenú “Medicamentos”.

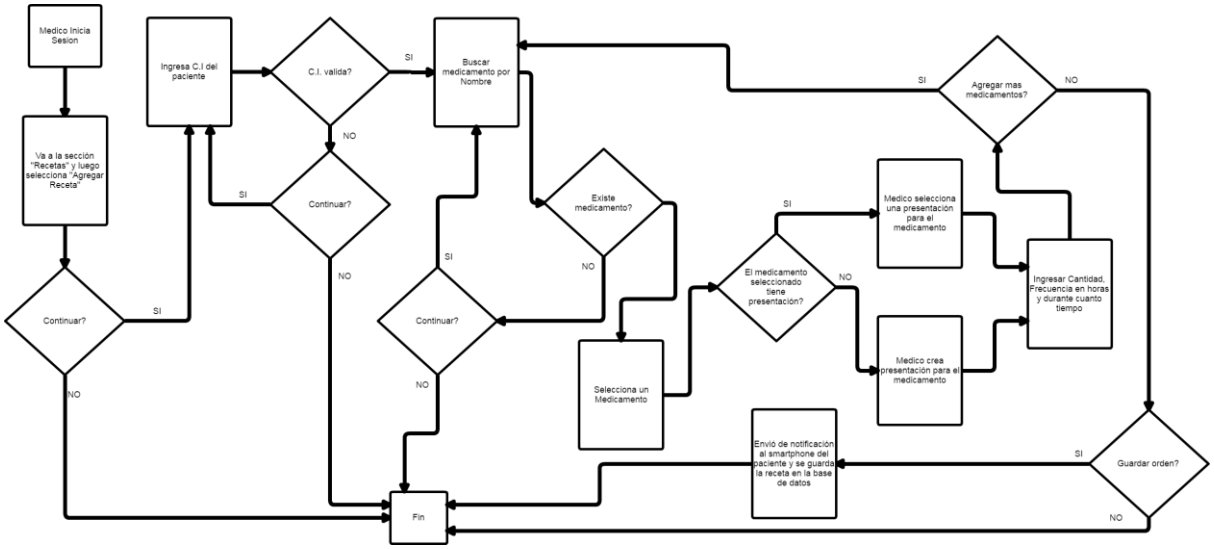

Figura 8.1 - Flujo de creación de la Receta Electrónica

La figura 8.1 presenta la serie de pasos para que el médico pueda recetar correctamente.

Una de las partes fundamentales de la creación es el envío de la notificación al smartphone del paciente, ya que, pese a que el paciente siempre puede verificar los “Medicamentos Nuevos” que se le han asignado, mediante el envío de la notificación puede automáticamente, al abrir el enlace en la notificación, confirmar el inicio del consumo en el momento mismo de la creación de la receta.

Lo anterior puede no tener mucho sentido si el paciente se encuentra en la consulta con el médico en el momento que se crea la receta, si el paciente tiene su smartphone Android conectado a Internet. Pero si ese no es el caso, el usuario

puede ver la notificación ni bien conecte su dispositivo a Internet y si el mismo ya posee el medicamento puede confirmar la fecha de inicio efectivo e inmediatamente recibe la notificación.

Luego de la creación de la receta hace falta que el paciente confirme el inicio del consumo de el/los medicamento/s recetados por parte del médico para que efectivamente comience en seguimiento al mismo por parte del médico y se generen los reportes del lado del servidor, precisamente en la capa de negocios del sistema.

8.2 Confirmación del inicio del consumo de medicamentos

Como se mencionó anteriormente, hace falta que el usuario confirme la fecha de inicio efectivo del consumo de el/los medicamento/s que el médico le recetó previamente utilizando la capa web de SIURE.

Al confirmar el inicio efectivo, se setean alarmas en el smartphone del paciente las cuales garantizan que se ejecutarán en tiempo y forma gracias a la recuperación de alarmas mencionada en el punto 8.5 a menos que el dispositivo se encuentre apagado en dicho momento y no se puedan ejecutar.

Además del recordatorio mediante un sonido, la alarma también contiene un texto descriptivo del medicamento que el paciente debe ingerir en ese horario junto a su cantidad, por lo que se cumplen las necesidades de recordar al paciente el horario en que debe consumir, el medicamento que debe ingerir y su cantidad, brindando una gran ayuda al paciente.

A su vez, cuando el paciente desea confirmar el consumo del medicamento, tiene la posibilidad de hacerlo mediante el botón “Ya Tomé” en la pantalla de la alarma en su smartphone. Dicho botón realiza una invocación a el web service “confirmar consumo” en la capa de servicios de SIURE que se encarga de impactar en la base de datos el consumo del medicamento seteando un atributo booleano llamado “Confirmación” en la entidad “Evento” que corresponde a la alarma disparada en el dispositivo.

Mediante dicho atributo es que se sabe si el paciente consumió su medicamento para la fecha y hora dadas a menos que haya presionado el botón “Cancelar” en cuyo caso el atributo se setea en “false” lo cual significa que el paciente no consumió el medicamento en ese horario.

Esto es de suma importancia para el seguimiento, ya que semanalmente se realiza la creación de documentos CDA que se guardan en una ruta establecida en el servidor en el que se ejecuta SIURE, los cuales son los que formalizan el seguimiento al consumo de medicamentos del paciente mediante porcentajes de adherencia al tratamiento. En la figura 8.2 se presenta la confirmación del inicio del consumo de medicamentos.

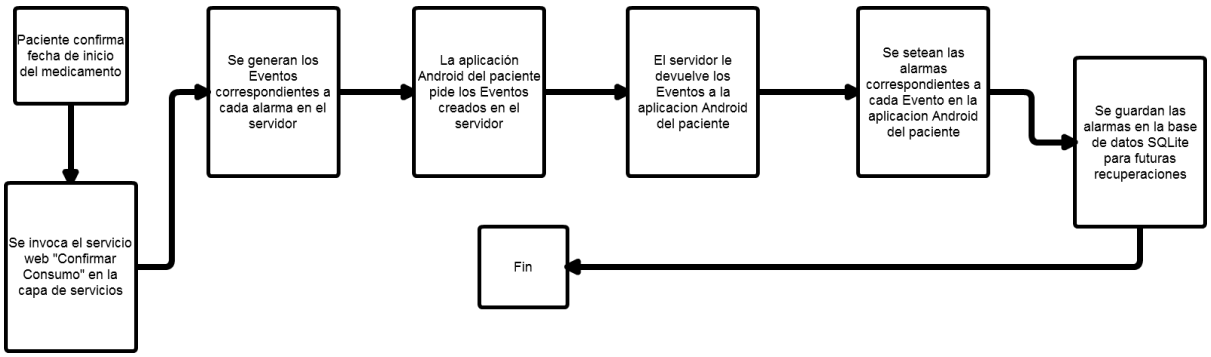

Figura 8.2 - Flujo de inicio de consumo por parte del paciente

8.3 Consumo y Adherencia al tratamiento

Dos de los aspectos más importantes para SIURE son el consumo y la adherencia al tratamiento del paciente. Es necesario definir formalmente ambos términos para poder manejarlos adecuadamente. Llamamos consumo al acto de ingerir un cierto medicamento M en un entorno alrededor del instante t. Cuando se crea la receta se generan todos los consumos correspondientes a lo largo de la validez de la receta. Estos datos son la cantidad de días, la frecuencia en horas dentro de cada día y la cantidad de medicamento a ingerir en cada consumo.

Cada consumo puede ser cancelado o confirmado por el paciente a la hora en que se dispara la alarma correspondiente al consumo. Habiendo mencionado lo anterior es que se define a la adherencia del paciente al tratamiento de la siguiente manera:

Dados c’ y C, con c’ = cantidad de consumos confirmados por el paciente, y C = cantidad de consumos totales de la receta, se define la adherencia del paciente al tratamiento como A = c’ / C.

La anterior definición nos lleva a concluir que al ser c’ < = C se puede expresar A como un porcentaje del total de consumos de la receta. Este porcentaje es el denominado **Indicador de adherencia** del paciente al tratamiento para cada medidor de cada receta.

Con respecto a los consumos, el instante de tiempo t es un tiempo teórico el cual puede no coincidir con el tiempo de ingesta del medicamento. Para cada

medicamento, se define una cierta tolerancia p (entre tiempo T1 y T1') la cual sumada a t (llamado T1 en la figura 7.3) forman el tiempo máximo en el cual el consumo se acepta como válido. Como se puede apreciar en la figura 7.3, existe un periodo p dentro del cual los consumos confirmados por el usuario son considerados válidos, pasando ese periodo los consumos son descartados.
En SIURE versión 0.1, se considera T1' = T2 marcando como trabajo a futuro la implementación de configuración de las tolerancias p de cada uno de los medicamentos por parte del médico.

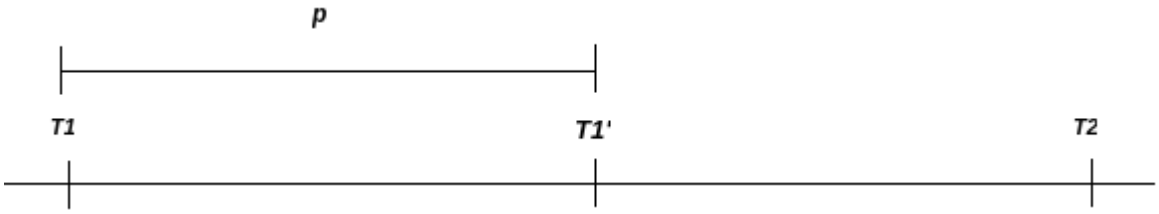

Figura 7.3 - Tolerancia temporal para evaluar adherencia El período p constituye la tolerancia temporal admitida para un fármaco recetado para ser tomado en el instante T1. Si el paciente ingiere el fármaco entre T1' y T2 SIURE asigna la toma al período siguiente, dejando incumplida la toma anterior.

8.4 Generación de documentos CDA para consumo de medicamentos

El documento electrónico CDA (Clinical Document Architecture) [11] mantiene el detalle de la prescripción y los datos de los consumos del paciente. CDA es un estándar ANSI reconocido en los 2000 que propone una estructura de documentos en formato XML, lo cual permite la generación de cualquier documento clínico a objetos interpretables por aplicaciones y la transferencia a través de cualquier medio electrónico.

El sistema SIURE utiliza template CDA configurados para las necesidades de ambos pedidos, y la herramienta javax.xml para manipular el DOM del documento CDA.
Para recordar, el requerimiento de "Generación de CDA para consumo de medicamentos" especificado en el capítulo 3 habla de la generación del documento CDA semanalmente, por lo tanto, se decidió que la generación del CDA sería cada lunes en la madrugada. Por lo tanto, se necesitaba una herramienta que permitiera programar automáticamente la generación de estos documentos, para esto se utilizó la herramienta Quartz.

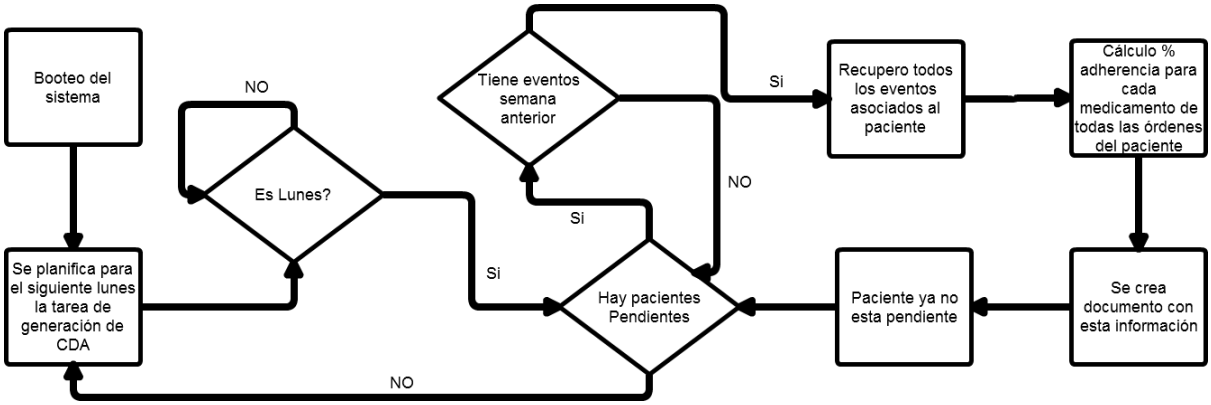

Figura 8.4 - Generación de documento CDA de adherencia al tratamiento para cada paciente los lunes de madrugada.

8.5 Manejo de eventos y alarmas

Dado que el consumo de medicamentos es importante para el paciente, es esencial que SIURE ayude en el consumo dado una alerta en cada dosis con una alarma en el celular del paciente.

Existe un contratiempo para que esto ocurra y es que el smartphone que el paciente porte se reinicie o se apague. Este contratiempo hace que las alarmas seteadas previamente se borren del sistema y no se ejecuten nunca, ya que Android setea las alarmas en la memoria y no se persisten a disco duro.

Para que las alarmas estén disponibles en todo momento primero se tuvo que buscar la forma de persistirlas en disco duro para poder tener los datos siempre disponibles incluso luego de reiniciar o apagar el dispositivo.

Segundo, se tuvo que buscar la forma de recuperar los datos y volverlos a setear en la memoria para que Android pueda ejecutar las alarmas en el momento indicado nuevamente luego del reinicio o apagado/encedido del smartphone.

Estos dos problemas se resolvieron mediante dos estrategias:

- Persistir los datos de las alarmas y eventos en una base SQLite creada para la aplicación.
- Crear un algoritmo de recuperación de los datos de las alarmas y eventos que obtenga los datos de dicha base de datos y los procese para luego setearlos en la memoria del dispositivo y puedan ejecutarse normalmente como si nada hubiera pasado.

Además de las dos estrategias nombradas, es necesario que la segunda estrategia pueda ejecutarse tan pronto como el smartphone se haya iniciado completamente, por lo que para ello se hizo uso de una herramienta de Android llamada

“BroadcastReceiver” [27] en conjunto con la declaración de dicha herramienta en el Android Manifest que es un archivo xml dentro del proyecto de la aplicación Android.

Esta herramienta es una clase Java a la cual se debe extender para poder utilizarla para lo que se debía hacer. Se declara un intent-filter dentro de las tags <receiver></receiver> de la declaración del BroadcastReceiver en el manifest de la siguiente forma:

```
<intent-filter>
  <action android:name="android.intent.action.BOOT_COMPLETED" />
</intent-filter>
```

Esas líneas en lenguaje XML indican que dicho BroadcastReceiver se disparará cuando el dispositivo haya completado correctamente su booteo. Pero hace falta además agregar un permiso en el manifest para que eso suceda, el cual se declara así:

```
<uses-permission
android:name="android.permission.RECEIVE_BOOT_COMPLETED" />
```

Resumiendo, la declaración de la clase que se implementó extendiendo de BroadcastReceiver para implementar el algoritmo queda de la forma:

```
<receiver
android:name="recetaelectronica.pirim.com.notificaciones.BooteoSmartphone"
  android:enabled="true"
  android:exported="true" >
  <intent-filter>
     <action android:name="android.intent.action.BOOT_COMPLETED" />
  </intent-filter>
</receiver>
```

Donde la clase BooteoSmartphone es la clase nombrada, que junto al permiso RECEIVE_BOOT_COMPLETED hacen que el algoritmo sea posible de implementar en Android.

En la figura 8.5 se detalla el funcionamiento del algoritmo que se implementó

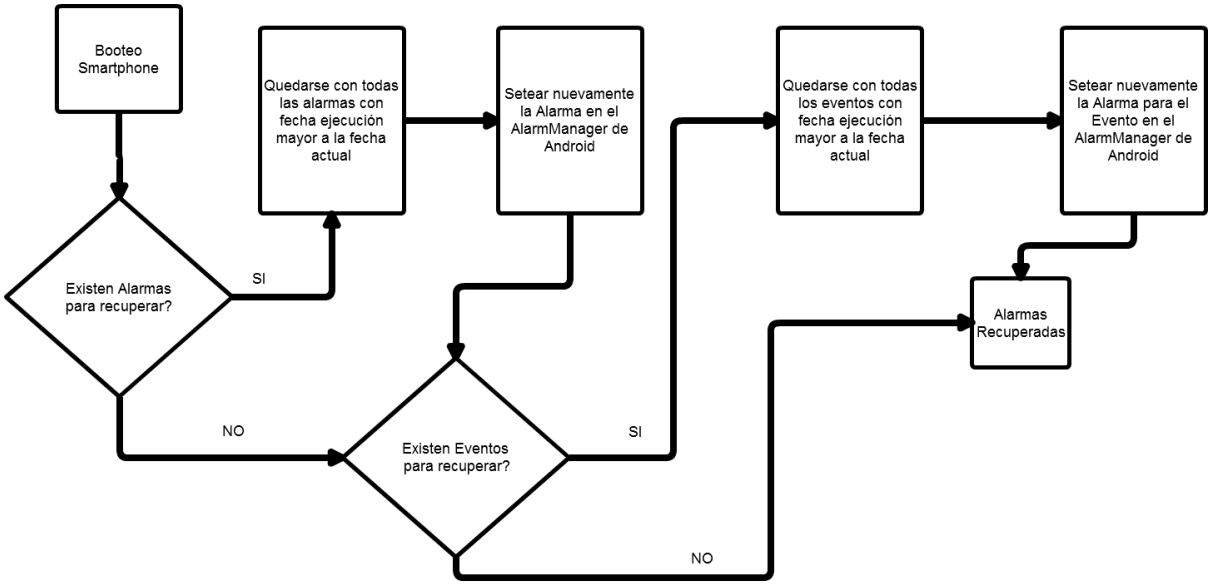

Figura 8.5 - Flujo de recuperación de alarmas

Como puede apreciarse, el algoritmo se basa fuertemente en los datos existentes en la base de datos SQLite, filtrando las alarmas y eventos que posean una fecha y hora de inicio posterior a la fecha y hora actual.

9 Implementación

9.1 Decisiones tomadas

En un principio, dados los requerimientos y casos de uso del sistema, se analizaron diversas plataformas, lenguajes de programación y tecnologías que pudieran ser útiles y que puedan brindar todas las herramientas necesarias para el desarrollo de ésta aplicación.

La decisión sobre cual o cuales optar se basó principalmente de los siguientes factores:

- Curva de aprendizaje.
- Solidez de la herramienta/lenguaje de programación/framework.
- Documentación.
- Experiencia de ambos integrantes del equipo trabajando o utilizando dicha herramienta.
- Posibilidad de implementar los requerimientos con dicha herramienta en tiempo y forma.

Dados los cambios repentinos en lo que se promueve en la industria del software con el término “tecnología de punta” las diferentes herramientas elegibles para este proyecto abarcan una gran gama de frameworks, arquitecturas y lenguajes de programación de las cuales ya se podría realizar un artículo completo.

Debido a lo anterior es que se decidió elegir a las herramientas más funcionales, robustas, estables, bien documentadas y para las cuales se hayan desarrollado mayor cantidad de librerías, extensiones, plugins, etc. Aparte de esto existen una serie de factores que podrían afectar a la hora de implementar los requerimientos y enfrentarse a un error.

Es por todo lo anterior que se listaron una serie de herramientas, las cuales son prácticamente arquitecturas en sí. Estas herramientas tienen comunidades de gran tamaño, o están en pleno crecimiento, lo cual es un buen factor a la hora de determinar las tecnologías a utilizar para el desarrollo de un sistema. Principalmente para mitigar los conocidos como “riesgos tecnológicos” y tratar de prevenirlos. Un ejemplo de éste tipo de riesgos puede ser:

“Enfrentarse a un error o problema particular al desarrollar una aplicación utilizando el lenguaje de programación “A” junto a su framework “B” y no encontrar una solución o no encontrarla rápidamente ya sea investigando en su documentación o realizando búsquedas en diversos sitios web como por ejemplo StackOverflow.”

El principio por el que se rigen la mayoría de las aplicaciones y sistemas informáticos hoy en día que establece que las aplicaciones y sistemas deben ser diseñados de una forma modular y se debe buscar lograr una alta cohesión y bajo acoplamiento de los componentes. Para seguir dicho principio, se divide a SIURE en dos componentes principales. Estos componentes son el Backend y Frontend, de forma de diferenciar bien los roles que cumple cada uno de ellos.

El Backend se encarga de ejecutar la lógica de negocio de la aplicación, controlar el acceso a la base de datos y exponer servicios a el Frontend.

El Frontend se encarga de brindar la presentación de la aplicación al usuario y de interactuar con el mismo, de modo de poder realizar las acciones o casos de uso que el usuario desea ejecutar de forma exitosa.

Las herramientas listadas a continuación fueron parte de la decisión tomada a la hora de elegir las tecnologías para el desarrollo de la aplicación:

- .NET Framework utilizando C# como lenguaje de programación, en conjunto con ASP.NET MVC 4
- Java EE 6 en el Backend de la aplicación utilizando EJBs (Enterprise Java Beans) y JSF combinado con uno de los frameworks de JSF como por ejemplo PrimeFaces [28], ICEFaces [29], RichFaces, etc en el Frontend de la aplicación.
- Spring junto a Java EE 6 en el Backend (utilizando los Beans de Spring) y Spring MVC en el Frontend utilizando las herramientas que provee Spring, por ejemplo, el manejo de transacciones, sistema de logueo de información, etc.
- HTML5, CSS3 y AngularJS en el Frontend y NodeJS en el Backend

Además de lo relacionado a la aplicación web planteada en los requerimientos, también existe la aplicación móvil que debía ser desarrollada para Android, por lo que se plantearon las oportunidades:

- HTML5, CSS3 y Javascript y utilizar PhoneGap para la generación de una aplicación Híbrida.
- Desarrollo de la aplicación móvil desarrollando nativamente en Android utilizando Android SDK y el lenguaje de programación Java.

Una de las decisiones clave fue el mantener un único lenguaje de programación en el desarrollo de la aplicación móvil y de la aplicación web. Por lo que se concluye que la opción de utilizar los productos de Microsoft relacionados al .NET Framework se descarta, ya que ninguna de las opciones planteadas para el desarrollo de la aplicación móvil utiliza .NET Framework ni C# como herramientas.

Otra opción que se descartó es la utilización de Spring para la aplicación web dado que ningún miembro del equipo tenía experiencia previa con él. Esto adiciona un mayor tiempo de investigación, experimentación y aprendizaje del framework, por lo que no era viable su utilización ya que no provee nada extra que Java EE 6 con JSF.

Por una razón similar es que no se seleccionó la opción del desarrollo utilizando HTML5, CSS3 y AngularJS en Frontend y NodeJS en el Backend. Sin embargo, a ésta opción hay que sumarle el hecho de que ésta arquitectura es relativamente nueva. Además, realizando una pequeña investigación se pudo ver que no posee tantas herramientas útiles como posee Java, que es una plataforma que ya tiene muchos años en la industria del software y una comunidad mucho mayor de desarrolladores en la cual ampararse ante cualquier percance.

Finalmente, debido a la experiencia laboral de cada miembro del equipo y a la experiencia universitaria con la plataforma Java EE 6, es que se decidió la segunda opción mencionada para el Frontend y Backend. Para el desarrollo de la aplicación móvil se utilizó Android SDK y Java como lenguaje de programación.

A su vez, dada la experiencia en proyectos universitarios y laborales con manejadores de bases de datos como MySQL y PostgreSQL, se decidió desechar la opción del uso de manejadores pagos, ya sea Oracle o Microsoft SQL Server.

Dado que recientemente a nivel laboral y universitario se utilizó MySQL en los proyectos realizados se determinó que sería el manejador a utilizar para éste proyecto. Esto fue debido al gran número de herramientas que provee tanto su opción de consola de comandos como su Workbench, el cual es muy potente e intuitivo.

Debajo se listan las tecnologías utilizadas:

Frontend:

- JSF junto al framework ICEFaces en su versión 3.3.0, combinado con CSS y HTML5

Backend:

- Java EE 6 utilizando EJB (Enterprise Java Beans) e Hibernate como ORM [30] framework (Object-Relational Mapping)
- API Rest utilizando JAX-RS
- Para el desarrollo del Frontend y Backend se utilizó el IDE Netbeans en su versión 8.0.2 [31].

Mobile:

- Android SDK en su versión 16 utilizando Java como lenguaje de programación
- Para el desarrollo de la aplicación Android se utilizó el IDE Android Studio en su versión 1.2.2 [32].

9.2 Estructura del código del Componente EAR

Para la implementación del proyecto además de las decisiones técnicas detalladas en la sección anterior, hubo decisiones en cuanto a la estructura del código que conforman a SIURE.

Al decidir utilizar Maven, se debió estructurar a SIURE en varios proyectos Java, los cuales se detallan a continuación:

receta-entidades: Aquí se implementan las entidades que conforman al sistema, que son utilizadas a lo largo de todo el resto de los proyectos. Ejemplos de estas entidades son “Usuario”, “Medicamento”, “TipoComposicion”, “ComposicionMedicamento”, etc. Cabe destacar que este es el proyecto que se encarga de implementar la Capa de Acceso a Datos detallada anteriormente

receta-ejb: Este proyecto es el encargado de implementar la Capa de Negocio del sistema conteniendo toda la lógica de SIURE. Dicha lógica es utilizada por la Capa Web para el manejo de las diferentes operaciones de un nivel de abstracción menor a la misma como por ejemplo el registro de un usuario en el sistema guardando sus datos en la base de datos de SIURE, el envío de notificaciones a los usuarios, guardar los datos correspondientes a la receta creada por el médico desde la web, etc. También es utilizada por la Capa de Servicios para exponer las funcionalidades del sistema.

recetaEAR-ear: Proyecto que se encarga de crear la estructura del archivo con extensión ear que será deployado en el servidor de aplicaciones JBoss necesario para el funcionamiento del sistema. No contiene código, pero a pesar de ello es sumamente importante en la estructura de proyectos que conforman el sistema al igual que los demás. Dado que sin este proyecto no se obtendría el archivo necesario para desplegar la aplicación en el servidor de aplicaciones.

receta-ws: Encargado de implementar la Capa de Servicios de SIURE. Es fundamental para la integración con la Aplicación Android la cual es uno de los componentes más importantes de todo el sistema dado que es la que tiene interacción directa con el paciente. Para implementar los servicios, este proyecto hace uso extensivo de los EJBs (Enterprise Java Beans) implementados en el proyecto receta-ejb manteniendo una comunicación directa entre la Capa de Servicios y la Capa de Negocio. Cabe destacar también que este proyecto

implementa la API Rest que culmina la implementación de la Capa de Servicios exponiendo los mismos a potenciales sistemas externos y a la Aplicación Android para que puedan utilizarlos.

receta-web: Uno de los proyectos más importantes de todo el sistema ya que implementa la Capa Web, fundamental en la interacción con la gran mayoría de los roles de SIURE. Principalmente posibilita el registro de nuevos usuarios, permite la creación de recetas por parte de los usuarios con rol "Médico" y otras funcionalidades. Está implementada en Java para los Managed Beans y en JSF para las páginas web que componen las vistas del sistema (más precisamente utilizando ICEFaces como framework web junto con JSF). También contiene archivos de propiedades de Java (Property files) para los mensajes llamados "Labels" mostrados en las páginas y también para la internacionalización de la Capa Web permitiendo múltiples idiomas en la aplicación, lo cual es importante para el uso global de la misma.

recetaelectronica: Este es el proyecto padre en la estructura de proyectos Maven llamado "Parent". Es muy importante ya que permite construir todos los proyectos anteriores. Para ello es necesario crear y configurar un archivo xml fundamental en la estructura de archivos que forman un proyecto Maven. Este archivo se llama "pom.xml" y es el archivo que se utiliza para construir un proyecto Maven. Los proyectos anteriores se declararon como módulos de este proyecto en ese mismo archivo, lo cual permite construirlos todos al construir este proyecto, y a su vez dichos proyectos declaran a este proyecto como su proyecto padre para que dicha construcción sea posible.

Claramente se puede ver que los proyectos detallados siguen la estructura clásica de un proyecto Maven formado por sub proyectos donde cada uno se encarga de implementar una parte específica del sistema. Esto es importante para reducir la responsabilidad de cada uno de dichos proyectos a una parte reducida del sistema, haciendo que sea más fácil de probar y de mantener.

Dichas características son vitales a largo plazo ya que una vez que un sistema se pone en producción es muy normal que sea expuesto a cambios potenciales que involucren la modificación del código del sistema en cierta parte de la arquitectura.

Al separar el sistema en subsistemas con roles claramente diferenciados es mucho más simple la tarea de realizar un cambio, incluso cambios que impliquen modificar código en diversos subsistemas. Esto es posible ya que al tener claro el rol de cada uno de ellos se puede buscar fácilmente el lugar en cada clase Java o página web a modificar.

Para el caso de la creación de nuevas funcionalidades, también es sencillo ya que se puede tomar como “template” otra funcionalidad y adaptarla según las necesidades.

Gran parte del tiempo invertido al comienzo del desarrollo se dedicó a la prevención de minimizar el uso del tiempo disponible para la búsqueda del código que se debe modificar, a la reestructuración de una o varias partes del sistema y a cambios que impacten en la arquitectura de SIURE. Ese tiempo se justificó realmente ya que cada vez que cierta funcionalidad debía ser implementada, el tiempo en horas era relativamente corto y lo mismo si había cambios a realizar al sistema o si se debía integrar un proyecto con otro y así sucesivamente.

9.3 Estructura de la Aplicación Android

La Aplicación Android está formada por un proyecto Android nativo, desarrollado en el lenguaje de programación Java y haciendo uso extensivo del SDK de Android, específicamente en su versión 16 así como de librerías y herramientas externas.

Para el manejo de dependencias y para hacer uso de librerías externas se utilizó Gradle y dichas librerías fueron declaradas en el archivo build.gradle correspondiente al proyecto.

El código de la aplicación está estructurado en “Paquetes Java” bien diferenciados por los roles que cumplen las clases contenidas en cada uno de ellos. Por ejemplo, uno de los paquetes contiene las “Activities” de Android, otro para el cliente REST, otro más para los Datatypes del sistema, otro para las clases encargadas de la interacción con la base de datos SQLite y así sucesivamente.

Dentro de la carpeta “res” se encuentran los archivos de configuración xml correspondientes a las vistas de la aplicación, estilos visuales de la misma, iconos e imágenes utilizadas, archivo de strings, etc.

Para el manejo de archivos de propiedades se utilizó la carpeta “assets” la cual contiene un archivo de propiedades llamado “servers.properties” con el servidor que expone la API Rest de SIURE. Aunque dicho archivo fue utilizado en un ámbito de pruebas, es sencillo cambiar el servidor al que se debe conectar la aplicación ya que solo es necesario cambiar una línea en dicho archivo.

Hablando de la estructura de la implementación de la aplicación, se decidió utilizar 3 activities de Android para la interacción con el usuario las cuales son:

Login.java: Es la activity de inicio en la aplicación, cada vez que la aplicación no se encuentra cargada en la memoria del dispositivo esta es la activity que se invoca, la

cual comprueba si el usuario ya ha iniciado sesión o no en la aplicación consultando en la base de datos SQLite. En caso positivo esta redirige la aplicacion a la activiy Inicio.java directamente y en caso contrario muestra el formulario de login para el usuario.

Inicio.java: Esta activity es la que implementa la gran mayoria de las funcionalidades de la aplicación. Está compuesta por un FrameLayout de Android que tiene el contenido a mostrar en pantalla iniciando por defecto en la vista del calendario. Pudiendo cambiarla para ver los medicamentos del paciente, y también para ver los medicamentos nuevos que le fueron asignados al mismo, cambiando el Fragment que compone a dicho FrameLayout. Esta clase también implementa la creación de Eventos personalizados por el paciente, ya sea para generar un recordatorio de una cita con el médico o para precisamente recordar la visita a la farmacia para la adquisición del nuevo medicamento que se le asignó. También la confirmación del inicio del consumo de un medicamento asignado, la vista de los Eventos y consumos del día al seleccionar la fecha en el calendario, etc.

AlarmReceiverActivity.java: En esta activity se implementa la manipulación de datos a mostrar en la pantalla que se muestra en el dispositivo al iniciar la alarma seteada para el consumo del medicamento o para el recordatorio de un evento creado por el usuario. También se puede confirmar el consumo del medicamento desde esta activity, posponer el consumo o recordatorio 5, 10 o 15 minutos o cancelar el consumo.

9.4 Seguridad

SIURE es un sistema informático de salud que almacena información correspondiente a los medicamentos que consumen los pacientes. Estos medicamentos son recetados por el médico con el que el paciente se atiende. Dicha información es confidencial y no debería poder ser accedida por todo el público. Para poder alcanzar dicho objetivo se añadió seguridad a SIURE de la siguiente forma:

- Autenticación entre la API REST brindada por la Capa de Servicios de SIURE y la aplicación Android. De esta forma solo la aplicación Android podrá acceder a la API REST.
- Cifrado de la información que viaja en los pedidos y respuestas HTTP de modo que su contenido sea ininteligible.

La autenticación se implementó mediante el uso del estándar JWT (JSON Web Tokens) [33] Al invocar al servicio de login, se crea el JWT (JSON Web Token) en base a una clave secreta almacenada en un archivo de Java properties en el Componente EAR de SIURE. Dicho Token se devuelve a la aplicación Android la

cual incrusta dicho token en la cabecera “Authorization” del pedido HTTP que va hacia la API REST. Haciendo uso de la clave ya nombrada se comprueba que el token es válido y se autoriza el pedido para acceder al recurso brindado por la API REST.

La gran desventaja de esto es que el token viaja en una de las cabeceras del pedido HTTP por lo cual, es posible que un atacante acceda a él tan solo obteniendo el contenido de dicha cabecera. Para evitar que esto ocurra es necesario ocultar el contenido de la cabecera de alguna forma.

El cifrado del pedido HTTP entre la API REST y la aplicación Android se realizó utilizando SSL (Secure Sockets Layer) [34]. Esto permite mantener las comunicaciones más seguras, para la transferencia de datos sensibles. Esto es posible mediante un certificado avalado por una autoridad certificadora, (CA *Certificate authority*). Para que esto funcione es necesario generar un keystore (almacén de certificados con claves privadas), donde incluye el certificado de clave privada y los certificados avalados por la CA.
Por lado de servidor, en nuestro caso como contamos con un JBoss la configuración simplemente se hace agregando un conector en “standalone.xml”

```
<connector name="https" protocol="HTTP/1.1" scheme="https" socket-binding="https" secure="true">
                    <ssl name="siure-ssl" key-alias="siure" password="xxxxxxx" certificate-key-file="${jboss.server.config.dir}/siure.keystore" certificate-file="${jboss.server.config.dir}/siure.cert" protocol="TLSv1"/>
            </connector>
```

Además se deben agregar los socket bindings correspondientes a los conectores como se especifica aquí debajo, solo hace falta cambiar los puertos por los que se necesite:

```
<socket-binding name="http" port="8081"/>
<socket-binding name="https" port="8080"/>
```

9.5 Base de Datos

Como se mencionó en la sección 9.1 se utilizó Hibernate como ORM y MySQL como Manejador de Base de Datos. Hibernate brindó una mayor abstracción al interactuar con la base de datos utilizando objetos de Java llamados “Entidades”. Mediante el uso de Anotaciones de Java fue posible crear las Entidades que se utilizaron en el sistema.

Además de las entidades se utilizó el “Entity Manager” el cual es una interfaz de Java para interactuar con el “contexto de persistencia”. Un contexto de persistencia es un conjunto de entidades las cuales están identificadas por su “ID” donde para cada ID existe una única entidad.

El Entity Manager provee operaciones para crear, borrar y actualizar entidades, para buscar entidades en la base datos por su ID y para realizar consultas.
La experiencia al utilizar Hibernate y el Entity Manager fue satisfactoria, brindando una mayor abstracción a la hora de interactuar con la base de datos.

En la figura 9.1 podemos ver el modelo de datos de SIURE.

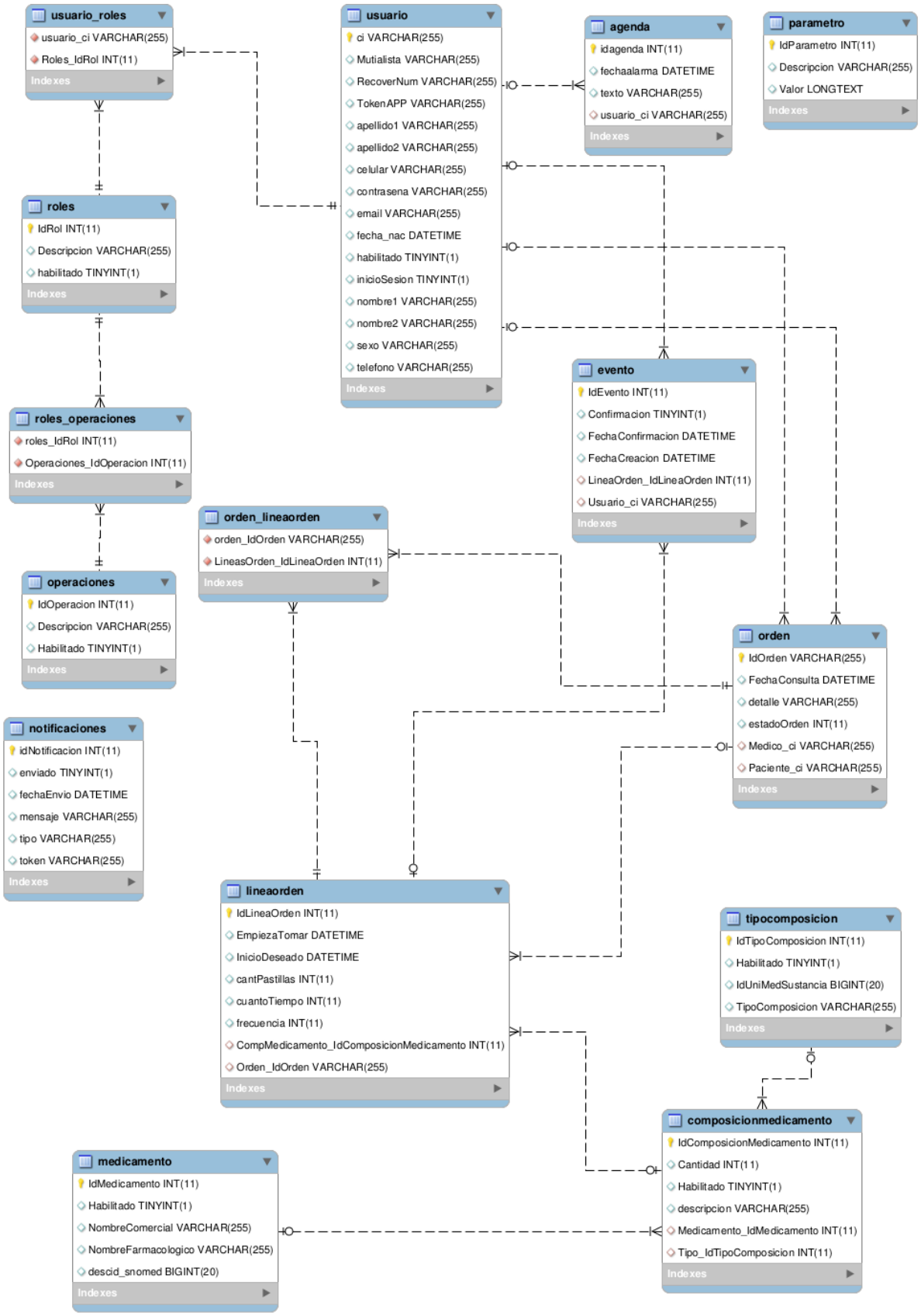

Figura 9.1 – Modelo de datos de SIURE

Cabe destacar que la tabla “orden” representa a la receta médica, donde cada línea representa el consumo de un medicamento con sus datos específicos.
En la tabla “agenda” se guardan los datos correspondientes a los eventos creados por pacientes en sus dispositivos Android. Los eventos pueden representar un cierto recordatorio para el usuario, ya sea una visita al médico o la compra de los medicamentos incluidos en su receta, etc.

Los datos correspondientes al consumo de medicamentos se almacenan en la tabla “evento”. En esa tabla se guarda la información correspondiente al consumo tanto para los consumos confirmados como para los no confirmados.

La tabla “notificaciones” almacena los datos necesarios para enviar notificaciones a los pacientes cuando se cumple la fecha esperada de comienzo del consumo de un medicamento.

Los roles como puede verse se almacenan en la tabla “roles” los cuales se definen por un conjunto de operaciones definidas en la tabla “operaciones”. De esa forma es sencillo restringir o brindar acceso a las operaciones del sistema basados en el rol del usuario.

9.6 Vistas de pantallas

Acercándonos un poco más SIURE, podemos observar en la figura 9.2 cómo luce la aplicación web y la aplicación móvil.

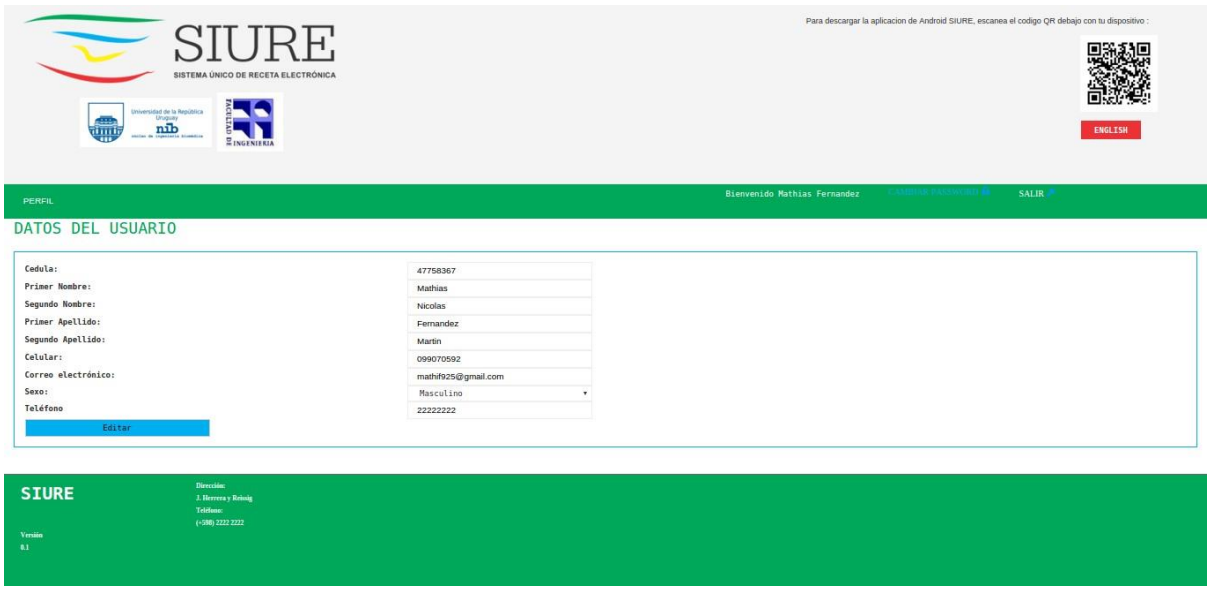

Figura 9.2 – Perfil de usuario en la aplicación web

En la figura 9.2 podemos observar el perfil del usuario conectado a la aplicación web, donde se puede apreciar toda la información de este usuario, permitiendo editar la misma.

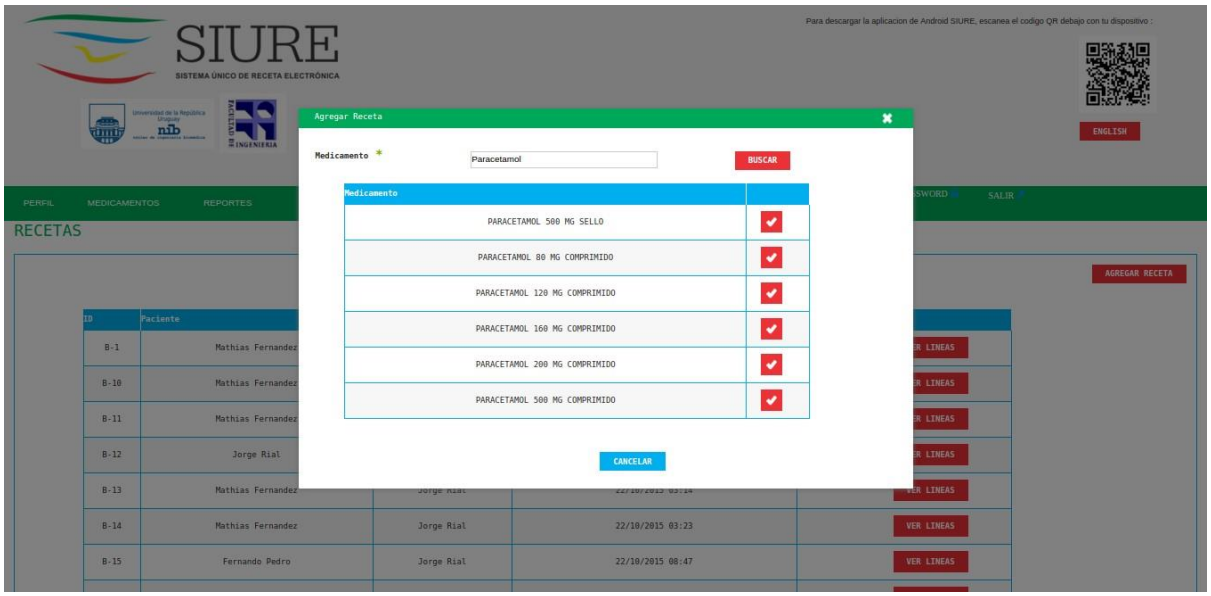

Figura 9.3 – Selección de medicamentos después de consulta al servicio terminológico.

Ahora podemos observar en la figura 9.3 los medicamentos retornados al consultar “Paracetamol” en los Servicios terminológicos de SNOMED

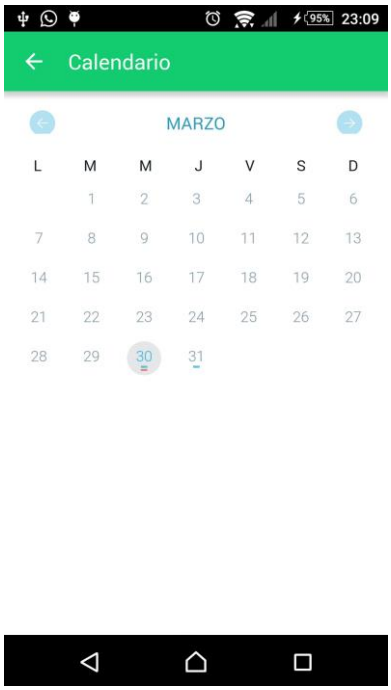

Figura 9.4 – Calendario en la aplicación Android

Por parte de la aplicación móvil en la figura 9.4, podemos ver el calendario como parte central de la aplicación. En el día 30, podemos ver que tenemos una marca roja que quiere decir que tenemos un o más eventos para ese día y azul, para alarmas asociadas a un consumo.

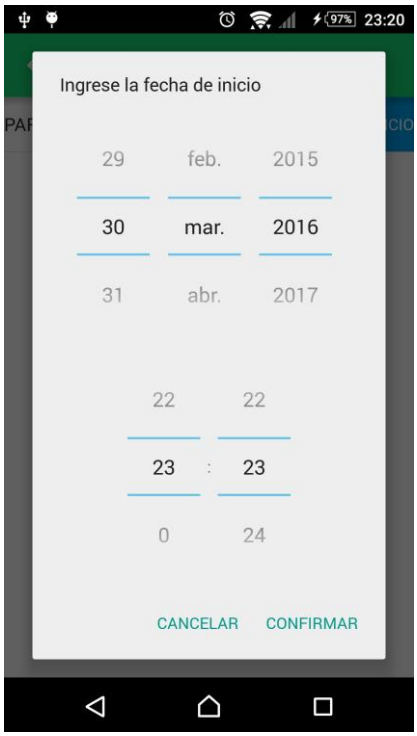

Figura 9.5 – Ingreso de fecha de inicio en la aplicación Android

En la figura 9.5 tenemos la selección de fecha de inicio de consumo como se explicaba en el requerimiento.

10 Pruebas

10.1 Descripción de las pruebas

Las pruebas de aceptación acordadas con la contraparte del proyecto fueron decididas para dar base al cumplimiento de los requisitos solicitados. Las pruebas fueron planificadas y ejecutadas por los integrantes de Ingeniería. A continuación, se pasarán a explicar las pruebas:

Login en web:
Se podrá realizar login con el usuario admin para poder iniciar en el sistema.

Registro paciente:
Al momento de registrar, se ingresa la cédula lo cual verificaría si el paciente existe en el sistema de OpenEMPI. En caso afirmativo se devolverán todos los datos asociados a este paciente, en caso contrario se permitirá ingresar los datos a mano. Para concluir se asignará el rol usuario.

Registrar medico:
Se ingresa la cédula correspondiente con el Médico. Se obtienen los datos de OpenEMPI y por último se le asigna rol Médico.

Login en el móvil:
Se deberá poder ingresar a la aplicación móvil con cualquier cuenta con rol usuario.

Prescribir:
Para hacer más completa esta prueba, primero el paciente ingresara a la aplicación móvil. Luego que el usuario con rol médico ingrese a la aplicación web, entrara a la sección de prescripción e ingresa los datos solicitados, empezando por la cédula del paciente, luego agregando un medicamento a la orden de prescripción asignando unidad de consumo, cantidad de la unidad, frecuencia y duración. Se podrán agregar tantos medicamentos como se desee.
Para terminar, el médico decide concretar la prescripción y en ese momento se le avisará al paciente mediante una notificación que cuenta con medicamentos disponibles, donde este podrá elegir la fecha en que empieza la ingesta de los mismos.

Ya tomé:
Después de configurar el comienzo del consumo de un medicamento, esperar a que suene la primera alarma donde aparecerá la opción de “Cancelar”, “Ya tomé” o “Posponer”. Primero elegiremos la opción de posponer y esperaremos esos minutos seleccionados. Al paso de estos minutos debería volver a sonar la alarma y en esta

ocasión elegiremos la opción de “Ya tomé”. Después ingresamos como administrador a la aplicación web y nos aseguremos que en la pestaña reportes -> consumos aparezca este consumo.

Notificaciones:
Para poder cumplir con esta prueba, primero asegurarse de tener varios dispositivos con la aplicación móvil y con un usuario logueado, luego seleccionar, varios tipos de filtros y enviar notificaciones.

Dado que el envío de documentos CDA al repositorio XDS de Salud.uy no está dentro del alcance del proyecto, no se realizaron pruebas de interoperabilidad.

10.2 Resultado de pruebas

La siguiente tabla muestra la prueba realizada, con los datos utilizados, el resultado esperado y el resultado obtenido:

Prueba	**Datos**	**Resultado Esperado**	**Resultado Obtenido**
Login en web	usuario: 12345678 password: admin	Login correcto	Login correcto
Login en web	usuario: 12345678 password: pass	Login incorrecto (password invalida)	Login incorrecto (password invalida)
Registro Paciente	C.I: 47758367	Registro Correcto	Registro Correcto
Registro Paciente	C.I: 47758367	Registro Incorrecto (Usuario existente)	Registro Incorrecto (Usuario existente)
Registrar Medico	C.I: 78547854	Registro Correcto	Registro Correcto
Login en el móvil	usuario: 12345678 password: admin	Login correcto	Login correcto
Login en el móvil	usuario: 12345678 password: pass	Login incorrecto (password invalida)	Login incorrecto (password invalida)
Prescribir	C.I paciente: 47758367 Medicamento: Bromhexina 50 MG - 10 MG por dosis - durante 5	Medicamento asignado correctamente, alarmas seteadas a lo largo del tiempo estipulado,	Medicamento asignado correctamente, alarmas seteadas a lo largo del tiempo estipulado,

	días cada 24 horas comenzando el 4/04/2016 a las 21:14	documento CDA generado correctamente	documento CDA generado correctamente
Ya Tomé (Confirmar)	Consumo Bromhexina 10 MG - 18/04/2016 a las 23:35	Se confirma el consumo y se puede ver en la página de consumos	Se confirma el consumo y se puede ver en la página de consumos
Ya Tomé (Cancelar)	Consumo Bromhexina 10 MG - 19/04/2016 a las 1:35	Se cancela el consumo y no se visualiza en la página de consumos	Se cancela el consumo y no se visualiza en la página de consumos
Ya Tomé (Posponer)	Consumo Bromhexina 10 MG - 19/04/2016 a las 0:35	Se posterga 10 minutos el consumo y cuando la alarma se dispare nuevamente se presiona “Ya Tome” y se visualizan los 10 minutos adicionados a la fecha de confirmación en la página de consumos	Se posterga 10 minutos el consumo y cuando la alarma se dispare nuevamente se presiona “Ya Tome” y se visualizan los 10 minutos adicionados a la fecha de confirmación en la página de consumos

10.3 Prueba de Carga

Se realizó una prueba de carga para el caso de uso “Ya Tomé” variando la cantidad de usuarios que llamaban al servicio que implementa dicho caso de uso. Para esto, se implementó una aplicación Java que simula los usuarios como Hilos de Java que ejecutan concurrentemente. Dichos hilos leen un archivo .csv con los id de consumos a confirmar generados a partir de un nuevo servicio creado específicamente para generar eventos para esta prueba.

En una computadora se ejecutó esta aplicación la cual llamaba a un servicio de la Capa de Servicios del Componente EAR de SIURE, el cual está ejecutándose en un servidor de Webfaction [35] contratado por los estudiantes de Ingeniería integrantes

del equipo. Las características del hardware del servidor se pasan a detallar a continuacion:

- CPU: Intel Xeon E3-1270 V2 3.50GHz 64 bits
- Nucleos: 4
- Subprocesos: 8

Cabe destacar que el plan contratado brinda tan solo acceso a un núcleo y 1Gb de memoria RAM.

Por otro lado la computadora donde se ejecutó la aplicación que llama al servicio mencionado del componente EAR tiene las siguientes características :

- CPU: Intel Core I7-2670QM 2.20GHz
- Núcleos: 4
- Subprocesos: 8
- Memoria: 8Gb

En la figura 10.1 se adjunta la gráfica de comparación entre cantidad de usuarios y el tiempo que llevo la culminación de las llamadas de cada usuario en finalizar.

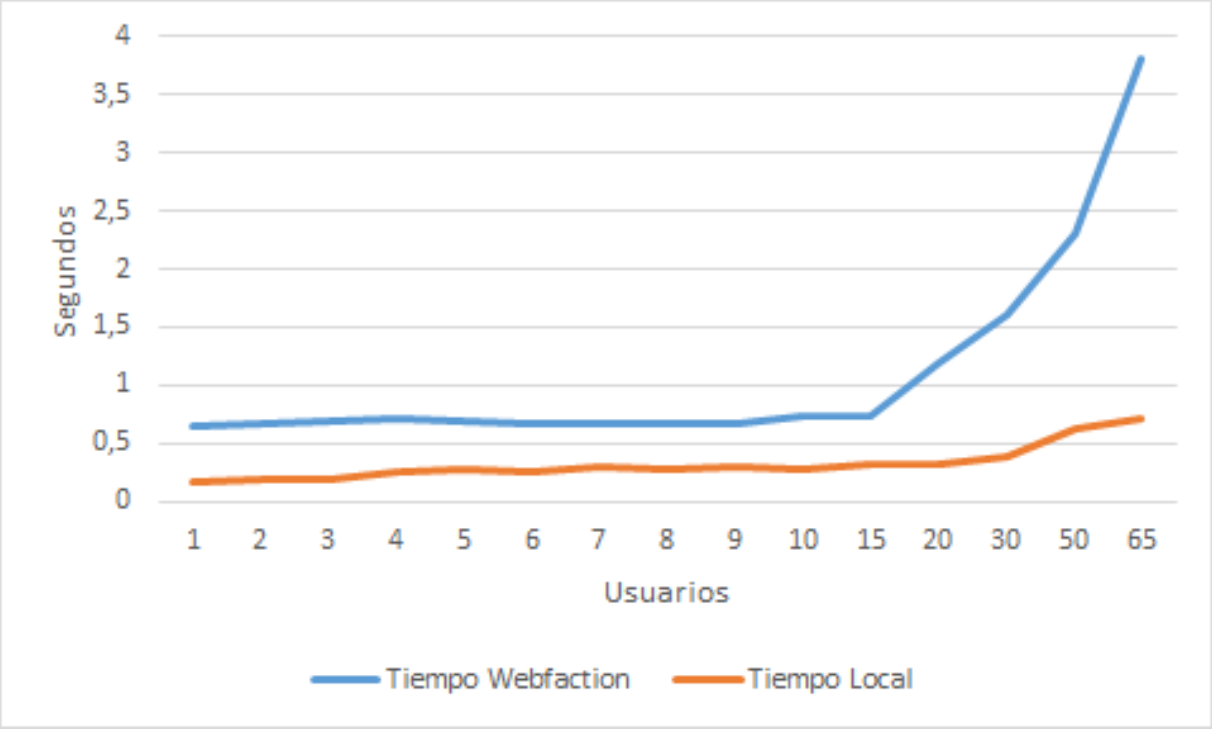

Figura 10.1 – Tiempos de respuesta de SIURE para acceso de un conjunto de usuarios simultáneos entre 1 y 65. Más de 15 usuarios SIURE se enlentece con las computadoras usadas.

Además, se calculó el promedio del tiempo de ejecución para un usuario como se aprecia en la figura 10.2.

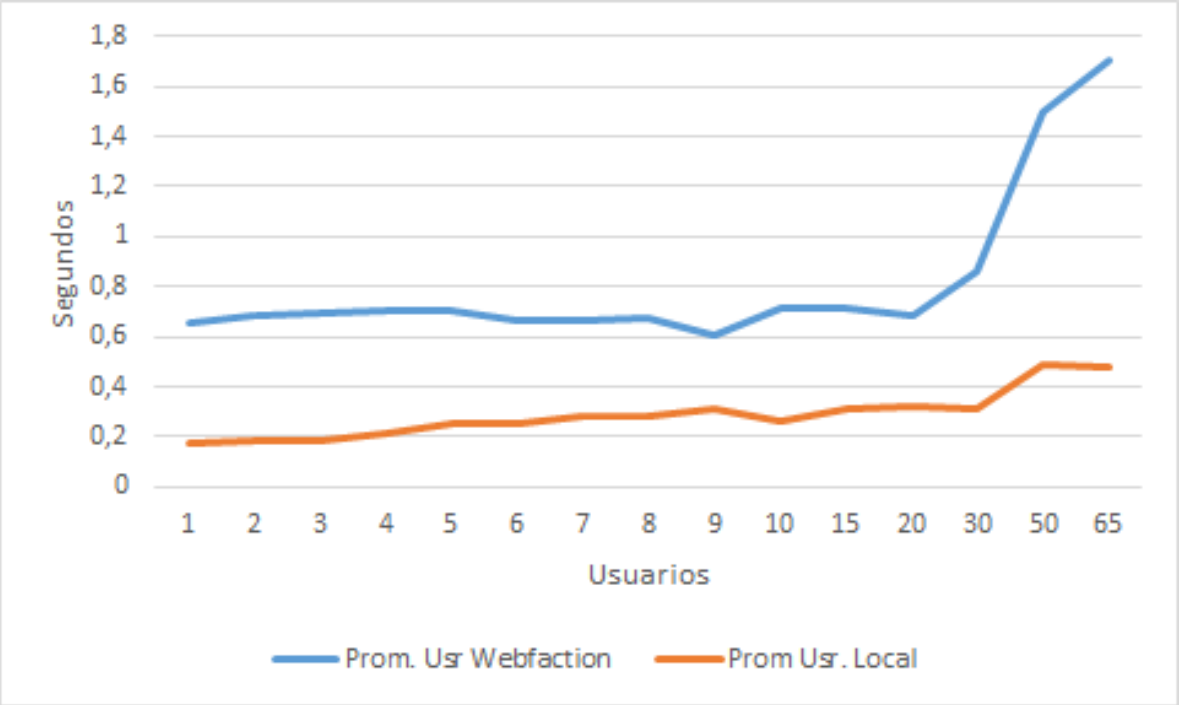

Figura 10.2 – Promedio del tiempo de respuesta para cada usuario en local y remoto. Con más de 20 usuarios aumentando el tiempo de respuesta remoto.

El análisis del acceso remoto a SIURE comprende la latencia total de varios usuarios y la latencia individual de cada uno, expresada en promedio. De la figura 10.1 se deduce que el tiempo de espera para atender hasta 15 usuarios simultáneos es del orden de 0.7 segundos. Debido al multithreading, es interesante notar, como se ve en la figura 10.2, que el promedio de respuesta a cada usuario es también del orden de 0.7 segundos (mientras no superen los 20 usuarios simultáneos).

11 Gestión del proyecto

11.1 Introducción

Esta sección contiene una visión general del proyecto y el producto a desarrollar, y la estrategia de evolución del Plan.

11.2 Alcance del Proyecto

El alcance se define a partir de la implementación de los siguientes casos de uso (a mayor número, mayor prioridad):

CASOS DE USODE SIURE Y PRIORIDADES DE DESARROLLO

Número de CU	Nombre	Prioridad
1	Prescribir un medicamento	4
2	Crear nuevo usuario	2
3	Notificación a pacientes	3
4	Dar de baja a un usuario	1
5	Reseteo de contraseña	1
6	Iniciar sesión	2
7	Ver medicamentos nuevos	2
8	Recibir Notificaciones	3
9	Agenda	2
10	Calendario	3
11	Ya tomé	4

Tabla 11.1 – se muestran todos los requerimientos funcionales y su prioridad ante

Como se puede ver en la tabla 11.1 los casos de uso con mayor prioridad son “Prescribir un Medicamento” y “Ya tomé”. Claramente son los casos de uso más importantes de SIURE ya que definen las principales funcionalidades del sistema: Recetar y Confirmar el consumo de un medicamento.

El desarrollo de SIURE comenzó por estos casos de uso, después se desarrolló el caso de uso “Crear nuevo usuario” y luego siguieron los de prioridad 3 y así sucesivamente.

11.3 Proceso de Gestión

11.3.1 Objetivos y Prioridades de Gestión

El objetivo de la gestión del proyecto es planificar adecuadamente las iteraciones y administrar óptimamente los recursos, para realizar satisfactoriamente el proyecto y sus actividades.

11.3.2 Condiciones asumidas, dependencias y restricciones

- Reuniones mensuales con el tutor del proyecto.
- Reuniones todos los viernes a las 19hs entre los integrantes del grupo de ingeniería.
- Dedicación al proyecto por parte de los integrantes de alrededor 450 horas a lo largo del proyecto.
- Reuniones quincenales con los clientes del proyecto (estudiantes de Registros Médicos).
- Conocimientos de programación y de ingeniería de software por parte de todos los integrantes.

11.3.3 Riesgos existentes

Relacionamiento con los clientes, estudiantes de Registro Médicos.
Como una de las principales características de este proyecto era el relacionamiento con los estudiantes de Registro Médicos, por lo tanto, el buen relacionamiento era muy importante para poder implementar el producto solicitado.

Integración con sistemas externos.
Al proponerse la utilización de sistemas externos (SNOMED y OpenEMPI), fue necesario una investigación previa para la utilización de los mismos. Esto llevó a la búsqueda de documentación al respecto de los métodos de comunicación que proveían y el uso de los mismos.

Aplicación móvil.
La poca experiencia en el entorno de desarrollo móvil supuso un esfuerzo adicional a la hora de implementar los requerimientos. Lo cual expuso un riesgo a la hora de evaluar la posibilidad de completar el desarrollo de la aplicación en tiempo y forma.

11.3.4 Gestión de horas

En la Figura 11.2 podemos ver el uso de horas a lo largo del proyecto.

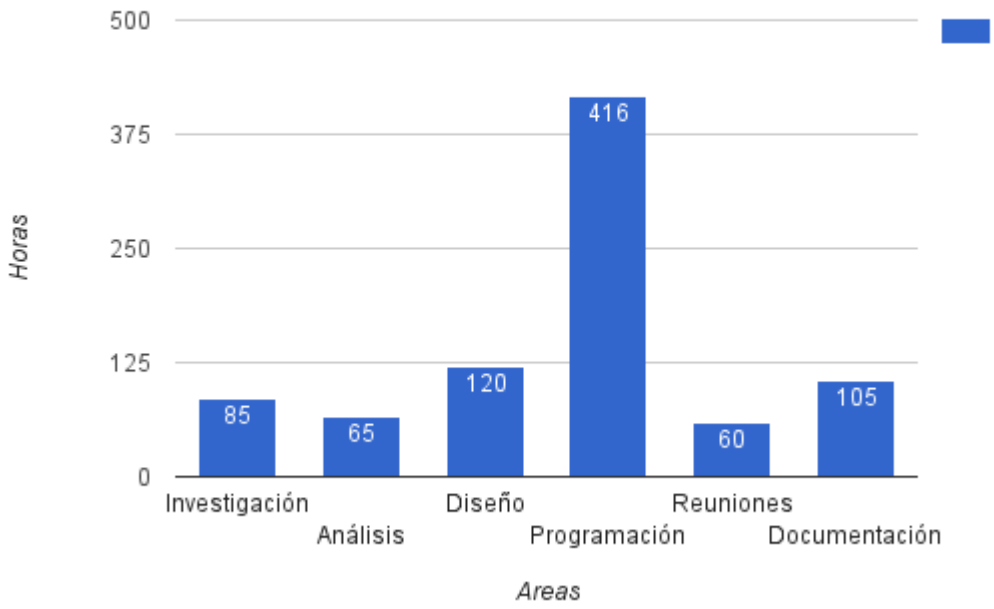

Figura 11.2 - Distribución horas-persona en tareas de SIURE. En los 18 meses se empleó un total de 851 horas hasta el 1 de agosto 2016.

Como podemos apreciar en la Figura 11.2 la gran mayoría de las horas dedicadas a SIURE fueron destinadas al desarrollo del sistema. Pero cabe destacar que las horas invertidas en el análisis y el diseño fueron fundamentales para una buena utilización de las horas destinadas a la programación del sistema. Por otro lado, podemos ver que se utilizaron horas de investigación para entender el entorno en el cual se implantaría el sistema y las dificultades que esto conlleva.

Las horas de reuniones fueron vitales para tomar decisiones importantes en conjunto al cliente, se tomaron alrededor de unas 36 horas. Cabe destacar, que las 24 horas de reuniones con los tutores fueron importantes para que el proyecto siguiera un rumbo correcto hacia la completitud de los objetivos establecidos al inicio del mismo.

11.4 Puesta en Producción

Para poner en producción a SIURE, sería necesaria la dedicación de horas para organizar mejor el código y automatizar la instalación del sistema. Es deseable contar con un archivo ejecutable tanto sea para Windows como para Linux que se encargue de la instalación y configuración del sistema.

De esa forma la instalación es más fácil y se tiene una mayor seguridad de que el sistema se encuentra listo para utilizar en producción una vez que ha quedado instalado. Esto reduce la introducción de errores por parte del encargado de la instalación manual.

El objetivo es reducir los problemas que le puedan surgir a la institución al utilizar a SIURE, por lo que el ejecutable instalador de SIURE debe ser implementado cuidadosamente. Es necesario instalar previamente la máquina virtual de Java en su versión 1.7, MySQL 5.5 y contar con el archivo .zip con la estructura de directorios del servidor JBoss 7.1.1. Finalmente es necesario contar con el archivo ear que contiene el componente EAR de SIURE para colocarlo en la ruta ruta_a_jboss/standalone/deployments y luego ejecutar el archivo ruta_a_jboss/bin/standalone.bat para Windows o el archivo ruta_a_jboss/bin/standalone.sh para Linux.

Habiendo realizado lo anteriormente mencionado el sistema ya está puesto en producción y puede utilizarse normalmente. Hay que recordar que además es necesario configurar en la base de datos mediante un registro en la tabla Parámetro la ruta al archivo apk con la aplicación Android de SIURE que va a utilizar el paciente.

11.5 Ensayo sobre comercialización

Siendo SIURE un producto estable y con funcionalidades importantes como la prescripción electrónica y “Ya tomé”, podemos embarcarnos en el mundo de la comercialización. SIURE está pensado para ser implantado en instituciones y usado como única licencia, que permite a todos los médicos de la institución tener su usuario como así los demás funcionarios.
Con respecto a la licencia se brindarán paquetes de instalaciones, donde podrán elegir paquetes mensuales como anuales, donde los últimos tendrán un mayor descuento.
Por otro, lado las licencias cuentan con manual de usuarios y capacitaciones, también se brindará horas de soporte para realizar consultas sobre SIURE.

12 Conclusión

12.1 Evaluación de la gestión del proyecto

A lo largo del proyecto se enfrentaron diversos retos sobretodo tecnológicos, los cuales significaron potenciales riesgos para el desarrollo del sistema propuesto. Al establecer una arquitectura bien definida se logra reflejar los propósitos de cada componente y la integración entre ellos de forma clara. Como resultado, se obtiene un producto mantenible, lo cual es un gran beneficio para sistemas de gran y mediano porte como lo es **SIURE**.

La inversión de tiempo al comienzo del proyecto fue necesaria para establecer una estructura bien definida. La utilización de Maven [36] dio buenos resultados al implementar cambios y funcionalidades del sistema. Esto se debe a que dichos proyectos reflejan bien su rol en la arquitectura del sistema, pudiendo localizar los componentes a cambiar fácilmente.

Para el mantenimiento de un sistema como **SIURE** es deseable tener documentación donde poder consultar toda duda que surja. Además de una buena documentación desde el inicio, descubrimos que la planificación y la dedicación constante tienen un rol importante en el transcurso del proyecto. Un ejemplo de esto son las reuniones con los tutores y con el grupo de RRMM durante las cuales el intercambio aportó elementos claves para el diseño. En cada una de las reuniones se propusieron mejoras al sistema, con gran agilidad en la implementación.

Las decisiones tomadas detalladas en el capítulo 8 fueron de suma importancia para la finalización del proyecto. Es realmente importante a la hora de implementar una serie de requerimientos, el tener conocimiento y experiencia en el lenguaje de programación. Sobre todo, cuando se deben cumplir plazos de entrega y al implementar ciertos requerimientos en un tiempo adecuado y de forma correcta.

Pese a que el aprendizaje de un nuevo lenguaje de programación, framework o herramienta no sea un impedimento, ahorra mucho tiempo el conocerlos con detalle. Por lo que se concluye que es realmente importante tener conocimientos y experiencia en lenguajes de programación y frameworks para la implementación de una aplicación o sistema informático.

12.2 Aprendizaje Interdisciplinario

Desde el inicio, se estableció el objetivo de lograr el intercambio interdisciplinario entre ambas partes para construir un sistema informático. Tanto sean los integrantes del grupo de estudiantes de la Licenciatura de Registros Médicos como

los estudiantes de Ingeniería en Computación, hicieron posible la construcción de un producto sumamente útil y funcional que podría ponerse en producción.

Pese a que los requerimientos para el sistema no estaban del todo claros en un principio, con el transcurso del tiempo se establecieron de forma clara. Esto fue gracias a la interacción en cada reunión planteada en la cual se incluyó una demostración del avance en la implementación del sistema hasta el momento.

Por lo tanto, el beneficio resultó mutuo de ambas partes, ya que cada una pudo aprender un poco de la disciplina de la otra. El producto obtenido refleja el trabajo de ambos grupos, ya sea tanto elaborando los requerimientos del sistema como desarrollándolos.

Pese a lo anteriormente nombrado, no se logró aprender e intercambiar más conocimiento del que realmente fue estrictamente necesario para el desarrollo de **SIURE**. El objetivo del aprendizaje interdisciplinario más amplio y abarcativo no se cumplió como se deseaba. Esto se debe a que resultó difícil entender las necesidades de la otra parte, ya sea por falta de conocimiento del área como por falta de comunicación entre las partes.

De lo anterior se concluye que es posible realizar un proyecto interdisciplinario entre estudiantes y/o profesionales de diferentes áreas. Cabe destacar que existen grandes diferencias entre ambos equipos en cuanto al conocimiento poseído por los individuos que los integran. Es justamente a partir de esas diferencias donde se obtienen beneficios que no se obtienen en proyectos puramente de Ingeniería. Lo más importante es que se aprende un poco de la profesión de los integrantes del otro equipo al tratar de entender sus requerimientos y el por qué surgen los mismos.

12.3 Conocimientos Adquiridos

Se han explorado diversas técnicas de programación a lo largo del proyecto. Principalmente en el desarrollo de la aplicación Android y en la Capa de Negocio donde se pudo utilizar nuevas herramientas y nuevas prácticas de programación.

Se utilizaron herramientas útiles en Android tales como las tareas asíncronas (AsyncTask). Fue posible implementar un cliente RESTful que haga peticiones HTTP a la Capa de Servicios y almacenar datos relevantes al paciente y los medicamentos que consume en una base de datos SQLite. También se hizo uso de un “Boot Service” de Android para recuperar las alarmas al iniciar el smartphone y colocarlas en la memoria.

A su vez, se incluye una librería sumamente útil y eficiente en la Capa de Negocio como lo es Quartz. Esta librería fue vital a la hora de la generación de documentos

CDA asociados al consumo de medicamentos y al seguimiento que se lleva del paciente en **SIURE**.

Es posible implementar los requerimientos sin estas herramientas, pero llevaría mucho más tiempo de lo que se ha invertido. Además, dada la confiabilidad que la mayoría de dichas herramientas proporcionan, se las pudo incluir a cada proyecto para que cumplan su cometido. Esto permitió dedicar el tiempo en el desarrollo de requerimientos en sí y no volver a implementar funcionalidades existentes.

Sin embargo, no solo se lograron aprender detalles técnicos y mejorar habilidades de programación. También fue posible adquirir la experiencia de desarrollar un sistema completo desde la especificación de requerimientos hasta la obtención del producto y sus pruebas documentadas. Las tareas que se desempeñaron trascienden las tareas puramente de desarrollo, involucrando planificación, documentación, análisis, diseño, entre otras. Fueron estas tareas en su conjunto las que brindaron un panorama global de lo que actualmente se trata la ingeniería de software en la industria.

12.4 Resultados Obtenidos

Desde el comienzo se utilizaron prácticas de desarrollo de software para construir un producto mantenible. Esto brinda mayor facilidad a la hora de implementar cambios evolutivos al sistema de manera sencilla y cómoda. Ya sea por parte de los integrantes del proyecto como por parte de otros desarrolladores. Logrando reducir la documentación necesaria para entender el funcionamiento de la aplicación, siendo únicamente necesario consultar los diagramas que explican la implementación de ciertos requerimientos (Ver capítulo 6) y luego analizar el código.

El tener conocimiento en el área de utilización de la aplicación es fundamental para definir correctamente los requerimientos y casos de uso del sistema. También lo es para ayudar a los desarrolladores a entender e implementar el software de acuerdo a las necesidades del cliente, dándole utilidad a los potenciales usuarios del sistema. En este caso los usuarios serían los pacientes registrados en SIURE y los médicos de la mutualista en donde se libere e instale el producto.

Pese a las ventajas que el producto final ofrece, se tienen claros los defectos existentes y detalles que aún faltan mejorar en SIURE. También sería necesario modificar al sistema para hacerlo más robusto, confiable y sin fallas lo cual no es algo sencillo de alcanzar. Sin embargo, se cree que el producto ya tiene estas características y lo único necesario es finalizar algunos detalles a nivel de la estética tanto del Frontend (Capa Web) como de la aplicación Android.

12.5 Trabajo Futuro

Como se dijo en el apartado anterior, se necesita mejorar al producto antes de lanzarlo en producción. La mayor parte de estos detalles son debido a falta de mejoras en la parte gráfica, tanto en la aplicación web como en la Android. Sería necesario invertir tiempo en la mejora de las pantallas cambiando los estilos existentes que se definieron para las dos aplicaciones que confirman al sistema.

Otros aspectos a mejorar son los relacionados a los casos de uso que involucran al médico. Es necesario obtener recomendaciones y opiniones por parte de médicos que usen **SIURE**, para poder adaptar el uso del sistema a sus necesidades. Algunos de los requerimientos pueden no estar implementados exactamente como lo desearía un profesional del área de la medicina. Sin embargo, para una primera fase, el sistema cumple con su cometido de forma correcta, brindando un panorama general del potencial que podría tener en instituciones de salud.

Para aumentar el alcance de la aplicación del paciente y no restringirlo únicamente al sistema operativo Android, sería deseable poder implementarla también para iOS. El beneficio no es menor ya que existe una gran cantidad de personas que utilizan iOS, ya sea desde sus smartphones o desde sus iPads (tabletas). Aunque este es un gran extra para SIURE, el costo no es menor, ya que es necesario poseer conocimiento en Objective C, que es el lenguaje de programación para el desarrollo de aplicaciones iOS.

Para una mejor organización del documento PDF de adherencia, sería necesario separar los consumos en semanas, una semana por página. Cabe destacar que esto sería necesario si el plazo elegido para generar el documento PDF de adherencia supera una semana. El médico puede analizar de forma organizada la adherencia el consumo de medicamentos de sus pacientes.

A su vez, para brindar aún más información, sería útil agregar una gráfica de evolución de adherencia en el tiempo en el mismo documento PDF. Esta gráfica es una herramienta para que el médico pueda analizar la adherencia a lo largo del tiempo. Esto permite realizar un mejor seguimiento al paciente lo cual es vital a lo largo de un tratamiento.

Tal y como se mencionó en el subcapítulo 7.3 es necesario implementar a futuro la configuración de las tolerancias por parte del médico para cada medicamento. A su vez, detectar cuando el consumo superó al tiempo de tolerancia y es descartado.

Finalmente, uno de los puntos a trabajar en el futuro que se consideran más relevantes para SIURE, es la creación de una API para la integración con farmacias. Esto se logra brindando una serie de servicios que proporcionen información a las

farmacias acerca de los medicamentos que forman parte de la receta asociada al paciente. De esa manera las farmacias pueden vender a los pacientes los medicamentos asociados a la receta que SIURE les provee mediante su API.

Cabe destacar que, pese a que SIURE brinda facilidades a la hora del seguimiento a la adherencia del consumo de medicamentos por parte del paciente, este seguimiento no es 100% certero. Esto se debe a que SIURE asume que, a la hora de confirmar el inicio del consumo del medicamento, el paciente tendrá su stock personal a lo largo del tratamiento. Además, si el paciente, cuando se dispara la alarma, selecciona “Ya Tomé” cuando realmente no consumió su medicamento, para SIURE esto es un consumo valido, pero para el paciente no.

Lo mismo si en determinado momento el paciente se queda sin medicamentos y por un corto lapso no dispone de los mismos y las alarmas se disparan. Si el paciente presiona “Ya Tomé” también SIURE los tomará como consumos válidos, aunque el paciente ingiera luego sus medicamentos. Esos falsos consumos llevan a que la adherencia registrada por SIURE no sea del todo cierta, o que contenga un cierto margen de error.

Sería necesario, en el futuro, diseñar un modo de minimizar este margen de error lo más posible. de forma de proveer a los profesionales la información lo más cercana posible a la situación real del paciente.

Además de eso, sería conveniente para la completa puesta en producción a nivel nacional de SIURE el integrar todas o la mayoría de las farmacias en un sistema centralizado. Este sistema podría brindarle al paciente de la farmacia más cercana de su hogar que cuente con stock y además poder brindarle el mejor precio disponible u ofertas que las farmacias brinden, etc. De ese modo se reduce el tiempo de demora desde que el médico receta los medicamentos hasta que el paciente realmente cuenta con ellos para comenzar el tratamiento.

Finalmente, para poder realizar una prueba de si es vial la puesta en producción a nivel nacional sería necesario acotar el dominio de producción de SIURE a una sola institución (por ejemplo, el Hospital de Clínicas), una patología (por ejemplo, Cardiología), 1 medicamento, 2 o 3 médicos y un número pequeño de pacientes (por ejemplo 15). De este modo se puede enfocar el uso de SIURE en un entorno reducido pudiendo analizar los datos generados correspondientes a la adherencia de cada paciente de forma correcta sin la necesidad de analizar un gran conjunto de datos correspondientes a un mayor número de pacientes manejando un número mayor de patologías en múltiples instituciones.

12.6 Conclusiones Finales

Nuestra conclusión final es que las horas invertidas en la estructuración del código y configuración del ambiente fueron vitales para el desarrollo del proyecto. A eso se suma la investigación realizada en los diversos requerimientos, la experiencia adquirida al utilizar nuevas herramientas y buenas prácticas de programación. Por lo tanto, el proyecto aportó enormemente a la formación profesional de los integrantes de Ingeniería, tanto para seguir contribuyendo a este producto como para el desarrollo de otros productos en el futuro.

Por todo lo anteriormente mencionado es que se destaca el valor del aprendizaje adquirido por parte de los integrantes del proyecto. Tanto a nivel técnico como a nivel de comunicación, organización, dedicación y gestión los cuales serán de suma importancia en el desempeño laboral y profesional de los integrantes.

13 Referencias

[1] Boticariagarcia, “Como funciona la receta electrónica,” 2014. [Online]. Available: http://boticariagarcia.com/2014/02/18/como-funciona-la-receta-electronica/. [Accessed: 28-Jun-2015].

[2] salud.uncomo.com, “Cómo funciona la receta electrónica.” [Online]. Available: http://salud.uncomo.com/articulo/como-funciona-la-receta-electronica-6885.html. [Accessed: 12-Aug-2015].

[3] S. A. de Salud, “Uso de receta electrónica.” [Online]. Available: http://www.juntadeandalucia.es/servicioandaluzdesalud/principal/documentosacc.asp?pagina=gr_farmacia_2_atphist#. [Accessed: 04-Oct-2015].

[4] M. C. GONZÁLEZ, “Regulación de la receta médica electrónica,” 2007. [Online]. Available: http://www.juntadeandalucia.es/boja/2007/123/d2.pdf. [Accessed: 05-Oct-2015].

[5] E. De-La-Poza-Plaza, I. Barrachina-Martínez, J.-L. Trillo-Mata, and R. Usó-Talamantes, “SISTEMA DE PRESCRIPCIÓN Y DISPENSACIÓN ELECTRÓNICA EN LA AGENCIA VALENCIANA DE SALUD,” 2011. [Online]. Available: http://www.elprofesionaldelainformacion.com/contenidos/2011/mayo/13.pdf. [Accessed: 05-Oct-2015].

[6] S. S. e Ig. MSSSI - Ministerio de Sanidad, “Interoperabilidad de receta electrónica en el Sistema Nacional de Salud.” [Online]. Available: http://www.msssi.gob.es/profesionales/recetaElectronicaSNS/Doc_Bas_Proyect_Interop_RESNS_v2.1.pdf. [Accessed: 25-Sep-2015].

[7] D. Flavia Baladán, D. Jimena Hernández, and D. Esc María José Viega, “RECETA MEDICA ELECTRONICA: MARCO LEGAL.” [Online]. Available: https://hcen.salud.uy/documents/22124/23517/Receta+m%C3%A9dica+electr%C3%B3nica+en+Uruguay/75f79a93-6984-473e-8813-43c447a56b73. [Accessed: 17-Aug-2015].

[8] “epSOS: About epSOS.” [Online]. Available: http://www.epsos.eu/home/about-epsos.html. [Accessed: 11-May-2016].

[9] M. H. Gabriel and M. Swain, “E-Prescribing Trends in the United States,” 2014. [Online]. Available: https://www.healthit.gov/sites/default/files/oncdatabriefe-prescribingincreases2014.pdf. [Accessed: 22-Aug-2015].

[10] “El Casmu implementa las recetas electrónicas.” [Online]. Available: http://www.espectador.com/sociedad/232579/el-casmu-implementa-las-recetas-electronicas. [Accessed: 17-Aug-2015].

[11] S. T. V.-C. H. Spain, “Guía para el desarrollo de CDA,” 2007. [Online]. Available: http://www.hl7spain.org/documents/comTec/cda/GuiaElementosMinimosCDA.pdf. [Accessed: 11-Jul-2015].

[12] OpenHIE, “OpenEMPI.” [Online]. Available: https://ohie.org/project/openempi/. [Accessed: 07-Aug-2015].

[13] AGESIC, “Extensión Uruguaya de SNOMED-CT.” [Online]. Available: http://www.agesic.gub.uy/innovaportal/v/4829/1/agesic/primer-release-de-la-extension-uruguaya-de--snomed-ct%C2%AE.html. [Accessed: 05-Dec-2015].

[14] Oracle, “EAR Container Structure.” [Online]. Available: http://docs.oracle.com/javaee/6/tutorial/doc/bnaby.html. [Accessed: 06-May-

2015].
[15] Oracle, “Java EE.” [Online]. Available: http://www.oracle.com/technetwork/java/javaee/overview/index.html. [Accessed: 06-May-2015].
[16] redhat, “JBoss.” [Online]. Available: http://www.jboss.org/technology/. [Accessed: 06-May-2015].
[17] Oracle, “MySQL.” [Online]. Available: https://www.mysql.com/. [Accessed: 06-May-2015].
[18] Microsoft, “Patrón Singleton.” [Online]. Available: https://msdn.microsoft.com/es-es/library/bb972272.aspx. [Accessed: 24-Jul-2015].
[19] U. A. de Madrid, “Patrones de Diseño,” 2007. [Online]. Available: http://arantxa.ii.uam.es/~eguerra/docencia/0708/11 Proxy.pdf. [Accessed: 25-Jul-2015].
[20] Oracle, “RESTful Web Services.” [Online]. Available: http://docs.oracle.com/javaee/6/tutorial/doc/gijqy.html. [Accessed: 08-Jul-2015].
[21] Oracle, “JavaServer Faces Technology.” [Online]. Available: http://www.oracle.com/technetwork/java/javaee/javaserverfaces-139869.html. [Accessed: 15-May-2015].
[22] SQLite, “SQLite.” [Online]. Available: https://www.sqlite.org/. [Accessed: 28-Aug-2015].
[23] Google, “Google Cloud Messaging.” [Online]. Available: https://developers.google.com/cloud-messaging/gcm. [Accessed: 15-Aug-2015].
[24] “SOAP.” [Online]. Available: http://www.service-architecture.com/articles/web-services/soap.html. [Accessed: 08-Jul-2015].
[25] AGESIC, “Salud.uy.” [Online]. Available: http://www.agesic.gub.uy/innovaportal/v/4422/19/agesic/que-es.html?idPadre=4425. [Accessed: 18-Sep-2015].
[26] kobjects, “kSOAP.” [Online]. Available: http://kobjects.org/ksoap2/index.html. [Accessed: 08-Jul-2015].
[27] Google, “BroadcastReceiver.” [Online]. Available: https://developer.android.com/reference/android/content/BroadcastReceiver.html. [Accessed: 05-Aug-2015].
[28] PrmieTek, “PrimeFaces.” [Online]. Available: http://www.primefaces.org/whyprimefaces. [Accessed: 15-May-2015].
[29] ICESOFT, “ICEfaces.” [Online]. Available: http://www.icesoft.org/java/projects/ICEfaces/overview.jsf. [Accessed: 15-May-2015].
[30] redhat, “Hibernate ORM.” [Online]. Available: http://hibernate.org/orm/. [Accessed: 20-May-2015].
[31] Oracle, “NetBeans IDE.” [Online]. Available: https://netbeans.org/. [Accessed: 08-May-2015].
[32] JetBrains, “Android Studio.” [Online]. Available: https://developer.android.com/studio/index.html. [Accessed: 20-Jun-2015].
[33] “JSON Web Token Introduction - jwt.io.” [Online]. Available: https://jwt.io/introduction/. [Accessed: 18-Jun-2016].
[34] P. K. Freier, Alan, Philip Karlton, “The Secure Sockets Layer (SSL) Protocol Version 3.0.” [Online]. Available:

https://httpd.apache.org/docs/2.4/ssl/ssl_intro.html. [Accessed: 18-Jun-2016].
[35] “Hosting for developers - WebFaction.” [Online]. Available: https://www.webfaction.com/. [Accessed: 20-Sep-2015].
[36] “Maven – Welcome to Apache Maven.” [Online]. Available: https://maven.apache.org/. [Accessed: 10-Jun-2015].

14 Anexo

14.1 Manual del Desarrollador

SIURE fue desarrollado usando Java EE 6 en el Backend, MySQL como Manejador de Base de Datos, Hibernate com ORM Framework y ICEFaces en el Frontend.
La aplicación Android de SIURE fue desarrollada utilizando el Android SDK (Software Development Kit).

Para la administración de los proyectos Java se utilizó Maven 3. La estructura del componente EAR y de la aplicación Android están detalladas en las secciones 9.2 y 9.3 respectivamente.

El acceso a las diferentes funcionalidades del sistema se brinda a través de la tabla “operaciones” en la base de datos. Las operaciones de cada usuario de SIURE son accesibles a través de sus roles. Cada rol está definido en la tabla “roles” y un rol está formado por N operaciones.

Es posible agregar una operación al sistema insertando un registro en la tabla “operaciones” y para hacerla utilizable es necesario insertar un registro en la tabla “roles_operaciones” definiendo el ID del rol y el ID de la operación.

14.1.1 Componente EAR

Los EJB (Enterprise Java Beans) utilizados en SIURE se encuentran en el proyecto recetaejb. Cada uno de ellos tiene nombres mnemotécnicos para facilitar la localización de la implementación de las funcionalidades del Backend. El acceso a la base de datos, el envío de notificaciones y la generación de documentos CDA se implementaron en este proyecto.

La aplicación web de SIURE está implementada en el proyecto recetaweb. Para la interacción entre la web y los EJBs en el Backend se utilizan Managed Beans de Java que mapean los componentes HTML de las páginas a atributos de Java. Mediante inyección de dependencias se inyectan los EJBs en los Managed Beans para poder acceder a sus métodos. Para esto se utilizó la API de CDI (Contexts and Dependency Injection).

Las entidades del sistema se implementaron en el proyecto recetaentidades. Se utilizaron Annotations de Java del API de Hibernate para la declaración de entidades y de sus columnas.

La Capa de Servicios se implementó como una API Rest en el proyecto recetaws. Se utilizó JAX-RS para implementar los Web Services. Los recursos de la API Rest están definidos en la clase PresentacionMobile.java.

Para el manejo de tareas programadas se utilizó la librería Quartz la cual brinda una gran facilidad para definir las tareas y ejecutarlas. Además, es una librería muy utilizada en la comunidad y existen muchos ejemplos documentados. Las tareas programadas se usan para el envío de notificación del inicio esperado del consumo de medicamentos a los pacientes y para la generación semanal de documentos CDA. En TareasProgramadasEJB.java se ejecuta un Job de Quartz que genera los documentos CDA semanales del consumo de medicamentos.

Para el envío de notificaciones a la aplicación Android de los pacientes se utilizó Google Cloud Messaging. Google Cloud Messaging es un servicio gratis de Google que permite el envío de mensajes a aplicaciones móviles. En NotificacionesEJB.java se implementó el envió de notificaciones utilizando un token generado por cada aplicación Android de SIURE a la hora de iniciar sesión mediante la API Rest. Ese token se guarda en la tabla “usuarios” el cual se utiliza cada vez que se requiera notificar a ese paciente.

14.1.2 Aplicación Android

La aplicación Android está dividida en Fragmentos de Android. Un Fragmento en Android es una parte de la interfaz gráfica de una Activity. Una Activity es un componente de la aplicación que contiene una pantalla con la que los usuarios pueden interactuar para realizar una acción la aplicación consta de dos Activities de Android. Una de ellas se encarga del Login y del envió del token de Google Cloud Messaging y se llama Login.java y la otra se encarga de interactuar con el usuario una vez logueado en el sistema. Esta última es la Activity principal y se llama Inicio.java.

Inicio.java está formada por 3 fragments, un fragment muestra la pantalla de inicio con el calendario llamado InicioFragment.java, otro muestra los medicamentos siendo consumidos actualmente llamado MedicamentosFragment.java y el ultimo muestra los medicamentos recetados al paciente que no ha comenzado a consumir y se llama MedicamentosNuevosFragment.java.

Para el acceso a la base de datos SQLite se utiliza la clase DatosSQLite.java, la cual provee métodos para la manipulación de todos los datos manejados en la aplicación Android.

Para la recuperación de las alarmas en memoria se utiliza la clase AlarmReceiverActivity.java que extiende de BroadcastReceiver. Para que esta clase se ejecute al momento del inicio del dispositivo fue necesario declararla en el AndroidManifest.xml.

14.2 Siglas

SIURE	Sistema Único de Receta Electrónica
epSOS	European Patients Smart Open Services
Casmu	Centro de Asistencia del Sindicato Médico del Uruguay
CDA	Clinical Document Architecture
XDS	Cross-Enterprise Document Sharing
OpenEMPI	Open Enterprise Master Patient Index
UML	Unified Modeling Language
E.R.	Especificación de requerimiento
EAR	Enterprise Application aRchive
Java EE	Java Enterprise Edition
EJB	Enterprise Java Beans
REST	Representational State Transfer
JSF	JavaServer Faces
UI	User interface
JSON	JavaScript Object Notation
SOAP	Simple Object Access Protocol
XML	eXtensible Markup Language
ANSI	American National Standards Institute
MVC	Model-View-Controller
JS	JavaScript
HTML5	HyperText Markup Language versión 5
CSS3	Cascading Style Sheets versión 3
SDK	Software Development Kit
APK	Android aPlication pacKcage
IDE	Integrated Development Environment
JWT	JSON Web Token
HTTP	Hypertext Transfer Protocol
SSL	Secure Sockets Layer
CA	Certificate authority
ORM	Object-Relational mapping
iOS	iPhone Operating System
PDF	Portable Document File
API	Application Programming Interface
HCEN	Historia Clínica Electrónica Nacional

MSSSI Ministerio de Sanidad, Servicios Sociales e Igualdad