CIS1630AP03

TAppi: Triage Application

ANDRÉS PINEDO GUTIÉRREZ

PAULA ALEJANDRA ROCHA SABOGAL

LUISA ÁLVAREZ VALENCIA

PONTIFICIA UNIVERSIDAD JAVERIANA

FACULTAD DE INGENIERIA

CARRERA DE INGENIERIA DE SISTEMAS

BOGOTÁ, D.C.

2016

CIS1630AP03

TAppi: Triage Application

Autores:

Andrés Pinedo Gutiérrez

Paula Alejandra Rocha Sabogal

Luisa Álvarez Valencia

MEMORIA DEL TRABAJO DE GRADO REALIZADO PARA CUMPLIR UNO DE LOS REQUISITOS PARA OPTAR AL TITULO DE INGENIERO DE SISTEMAS

Director

Leonardo Flórez Valencia

Jurados del Trabajo de Grado

JulioErnesto Carreño

Luisa Fernanda Barrera León

Página web del Trabajo de Grado

http://pegasus.javeriana.edu.co/~CIS1630AP03

PONTIFICIA UNIVERSIDAD JAVERIANA

FACULTAD DE INGENIERIA

CARRERA DE INGENIERIA DE SISTEMAS

BOGOTÁ, D.C.

Noviembre 28 2016

PONTIFICIA UNIVERSIDAD JAVERIANA

FACULTAD DE INGENIERIA

CARRERA DE INGENIERIA DE SISTEMAS

Rector Magnífico

Jorge Humberto Peláez Piedrahita, S.J.

Decano Facultad de Ingeniería

Ingeniero Jorge Luis Sánchez Téllez

Director de la Carrera de Ingeniería de Sistemas

Ingeniero Germán Alberto Chavarro Flórez

Director Departamento de Ingeniería de Sistemas

Ingeniero Rafael Andrés González Rivera

Artículo 23 de la Resolución No. 1 de Junio de 1946

"La Universidad no se hace responsable de los conceptos emitidos por sus alumnos en sus proyectos de grado. Sólo velará porque no se publique nada contrario al dogma y la moral católica y porque no contengan ataques o polémicas puramente personales. Antes bien, que se vean en ellos el anhelo de buscar la verdad y la Justicia"

AGRADECIMIENTOS

ADios, por darnos la vida llena de capacidades y virtudes para hacer el bien a nuestros semejantes.

A nuestras familias, en especial a nuestros padres, quienes nos han motivado y apoyado en el desarrollo de nuestros proyectos.

A la Pontificia Universidad Javeriana, por abrir sus puertas y permitirnos crecer espiritual, personal y profesionalmente.

A los maestros de la universidad, quienes con su conocimiento y experiencia motivaron la transformación de nuestros sueños.

A nuestro director de trabajo de grado, Leonardo Flórez, por su apoyo y orientación en el desarrollo de esta investigación.

A nuestros jurados, por sus valiosos aportes en el trabajo de grado.

A nuestros compañeros de pregrado, testigos de nuestras alegrías, progresos, desvelos, tristezas y experiencias.

¡Muchas gracias!

CONTENIDO

1. Tablas

2. Ilustraciones

ABSTRACT

The Triage process is a technique used to prioritize patients entering the emergency room. These type of classification is based on reviewing the vital signs, the pain scale and other indexes to carry out such prioritization. Measurements allow the rapid identification of patients at life-threatening situations and ensure prioritization by providing patients with a waiting time; however, this situation is not reflected in the national reality. Considering the above, the following thesis proposes to perform an intelligent data capture to streamline the process through a mobile application.

RESUMEN

El proceso de Triage es la técnica empleada para priorizar los pacientes que ingresan a la sala de urgencias. Estas clasificaciones están basadas en revisar los signos vitales, la escala de dolor y otros índices para realizar dicha priorización. Las medicionespermiten identificar rápidamente los pacientes en situación de riesgo vital y asegurar la priorización otorgando a los pacientes un tiempo de espera máximo;sin embargo, esta situación no se ve reflejada en la realidad nacional. Teniendo en cuenta lo anterior, el siguiente trabajo de grado propone realizar la captura inteligente de datos para agilizar el proceso mediante una aplicación móvil.

INTRODUCCIÓN

El presente trabajo grado procura mostrarel desarrollo de la aplicación móvil TAppi: Triage Application que permite realizar la captura inteligente de datos para agilizar el proceso del Triage. Este documento se divide en siete secciones: descripción general, marco teórico, análisis, diseño, desarrollo de la solución, resultados y conclusiones.

La primera sección contiene la descripción general de la problemática, la formulación y la justificación, contiene también la descripción de los objetivos y la metodología utilizada para el desarrollo del trabajo de grado.

La siguiente sección contiene el marco teórico con una breve explicación de la explicación del proceso de Triage y algunas aplicaciones que se han desarrollado a nivel mundial. Dentro de la sección de Análisis, se encontrarán los requerimientos, los casos de uso y la trazabilidad de los requerimientos versus los casos de uso.

La sección de diseño contiene las vistas arquitecturales, la estructura de sistema y el modelo de persistencia y el modelo de dominio. La siguiente sección describe el desarrollo de la solución, representando el sistema basado en reglas y la interfaz de usuario.

Dentro de la sección de resultados se encuentran las pruebas realizadas por diferentes roles/usuarios del sistema. Por último, la sección de conclusiones muestra el análisis del impacto esperado y las conclusiones y el trabajo a futuro que se propone.

I - DESCRIPCIÓN GENERAL

1. Oportunidad, Problemática, Antecedentes

Formulación del problema que se resolvió

El sistema de salud según la OMS "engloba todas las organizaciones, instituciones y recursos cuyo principal objetivo es llevar a cabo actividades encaminadas a mejorar la salud" [1]. Uno de los servicios ofrecidos por este sistema de salud, es la unidad de urgencias hospitalarias definida como "una organización de profesionales sanitarios que ofrece asistencia multidisciplinar, ubicada en un área específica del hospital, que cumple unos requisitos funcionales, estructurales y organizativos, de forma que garantiza las condiciones de seguridad, calidad y eficiencia adecuadas para atender a las urgencias y emergencias" [2].

Los profesionales sanitarios ubicados en la unidad de urgencias llevan a cabo un proceso para determinar el ingreso al servicio de un paciente. Este proceso lleva el nombre Triage cuyo origen es la palabra francesa trier que significa escoger, separar o clasificar [3]. Existen diferentes tipos de Triage y cada uno presenta diferentes niveles dependiendo del procedimiento usado, país o procedencia [4][5]. Sin embargo, todas las clasificaciones están basadas en revisar los signos vitales, una escala de dolor y otros índices para realizar dicha priorización [6]. Las preguntas y mediciones, orientadas a identificar la gravedad del estado del paciente, permiten identificar rápidamente los pacientes en situación de riesgo vital y asegurar la priorización del nivel de clasificación acorde con la urgencia de la condición clínica del paciente otorgándoles un tiempo de espera máximo[7], situación que no se ve reflejada en la mayoría de los centros de salud del país.

Actualmente, los pacientes se enfrentan a una serie de obstáculos en el momento de entrar al servicio de urgencias;algunos de ellos pueden permanecer en la sala de espera varias horas antes de poder ser atendidos[9][10]. Las razones por las cuales se generan demoras en urgencias se deben a una sobrecarga de pacientes, que ocasionan cuellos de botella en la clasificación del Triage. Adicionalmente, hay personas que acuden a la unidad de urgencias para suplir otros servicios, sin conocer que la sintomatología que presentan no es considerada

una urgencia: “no todo lo que el paciente percibe como una urgencia es una urgencia desde el punto de vista médico”[11].

Otra causa es la falta de un sistema que permita realizar una captura de datos eficiente [12]de manera que el usuario no tenga que brindar datos básicos u otro tipo de información, de manera reiterativa, al funcionario encargado de estandarizar las dolencias [13][14]. La labor de la captura de datos de forma manual se puede ver desencaminada debido al cansancio del personal sanitario[12]; sin embargo, la tarea de recopilación de datos básicos podría ser realizada por cualquier persona ayudada por la tecnología sin la necesidad del conocimiento médico. Para resolver este tipo de problemática se planteó la pregunta de ¿cómo desarrollar una aplicación móvil que permita realizar la captura inteligente de datos para agilizar el proceso del Triage acorde a los diferentes roles (usuarios) que tiene el sistema?

Justificación del problema

La propuesta de este trabajo de grado presenta un componente interdisciplinario entre la medicina y la ingeniería de sistemas donde la ingeniería permite automatizar procesos utilizando tecnología y la medicina usa las herramientas que brinda la ingeniería para capturar datos de los pacientes. Ayudados por la “revolución digital que ha hecho posible que la información digitalizada sea fácil de capturar, procesar, almacenar, distribuir, y transmitir”[15], se aprovechó las ventajas mencionadas anteriormente para agilizar el Triage uniendo las dos disciplinas [16].

Hoy en día, varias aplicaciones efectúan el proceso del Triage hospitalario. Cabe resaltar que la mayoría de ellas se encuentran enfocadas a usuarios con conocimientos avanzados en temas de salud [17], además de ser nativos digitales (“aquellos individuos que han crecido inmersos en la tecnología digital”) [18]. Se han desarrollado aplicaciones como MET (Mobile Emergency Triage), ITriage [17], Quick Triage App [19], las cuales sirven de apoyo al sistema del Triage para individuos que conocen terminología médica, con el fin de clasificar los diferentes tipos de dolor y generar resultados de priorización. Estos desarrollos capturan las circunstancias de los usuarios/pacientes y posteriormente la aplicación analiza los datos suministrados [16]. El soporte que otorgan puede ser utilizado en cualquier momento y lugar.

De acuerdo con el párrafo anterior, se puede deducir que para el uso adecuado de dichas aplicaciones es necesario conocimiento extensivo en el área de medicina y, por lo tanto, cualquier usuario no es capaz de utilizarlas. Es aquí donde se presenta el mayor diferenciador frente a desarrollos similares. La aplicación propuesta, dirigida a la población colombiana, muestra la información dependiendo del nivel de conocimiento del usuario acerca de la salud, este nivel de conocimiento se evalúa con el registro de usuario al ingresar por primera vez a la aplicación.

Por lo anteriormente dicho, "diseñar un sistema de Triage estructurado que incluya la recepción, la acogida, la clasificación y el seguimiento a los pacientes" mejorar los tiempos de espera en urgencias [10]. El proyecto fue enfocado hacia la estandarización de una aplicación que permita la captura inteligente de datos con el fin de agilizar el proceso del Triage proponiendo una clasificación previa [20]. Igualmente, otorga a los individuos una orientación frente a lo que los aflige antes de acudir a un centro de salud.

Impacto Esperado

La implementación de esta aplicación trae a corto plazo grandes ventajas para los pacientes, las instituciones prestadoras de servicio y el personal sanitario. Puesto que constituye un nuevo formato que permite el fácil diligenciamiento de los datos básicos del Triage,brindando a los usuarios la oportunidad de ingresar sus datos de manera inteligente, antes de asistir al centro de salud para agilizará el proceso del Triage.

Se pretende a largo plazo, crear una estandarización de captura de datos, con un lenguaje común que permitirá asegurar los tiempos de espera en los diferentes centros de salud para realizar el proceso del Triage. Emparejar las diferentes instituciones médicas, hará que los pacientes puedan recurrir a cualquier centro de salud sin incomodidades [21]. De igual forma, la aplicación contendrá suficiente información para todas las dolencias que pueda presentar el ser humano (niños, adultos, adultos mayores).

2. Descripción del Proyecto

Objetivo general

Desarrollar una aplicación móvil que permita realizar la captura inteligente de datos para agilizar el proceso del Triage acorde a los diferentes roles (usuarios) del sistema.

Objetivos específicos

- Especificar los requerimientos de la aplicación
- Diseñar la arquitectura teniendo en cuenta los diferentes roles de usuarios, basada en el análisis del objetivo anterior.
- Desarrollar el prototipo funcional de la aplicación descrita en los objetivos anteriores.
- Validar el software con pruebas de aceptación por parte del director de trabajo de grado.
- Documentar los manuales de la aplicación.

3. Metodología

Esta sección presenta el proceso de desarrollo del trabajo de grado. A continuación, se explicarán las metodologías utilizadas y cómo se relacionan entre sí. Se tomaron características de diferentes técnicas de desarrollo de software, donde dichas singularidades son las que más se acomodaron a la realización de TAppi: Triage Application. Se utilizaran las siguientes metodologías: Extreme Programing [22]y RUP (Rational Unified Process)[23].

Extreme Programing es un método utilizado cuando se desarrolla software que presenta requerimientos cambiantes o vagos. Cuenta con tres principios importantes: diseños simples, pruebas automatizadas y refinamiento constante del diseño [24].Adicionalmente se tuvieron en cuenta las siguientes prácticas: planeación (determina el alcance del siguiente ciclo), Metáfora (los desarrollos están guiados en como funcionara el sistema en su totalidad), Testing (continuamente se escriben pruebas unitarias y se programan), programación en parejas (dos personas con un solo computador), autoría colectiva (cualquiera puede cambiar el código en cualquier momento) y estándares de código (los programadores codifican con el mismo estilo).

RUP es un modelo de ciclo de vida iterativo incremental con el fin de lograr la producción de software de alta calidad. Las mejores prácticas de RUP son el desarrollo de software iterativo, manejo de requerimientos, usar una arquitectura basada en componentes, modelar visualmente el software y verificar su calidad [25]. El ciclo de vida del proyecto se dividió en fases que se pueden llevar a cabo en una o más iteraciones; cada iteración es un producto ejecutable o de un subconjunto del producto final. Las fases con su conjunto de tareas traducidas al proyecto a realizar son: fase de concepción, fase de elaboración, fase de construcción (de RUP) y la fase de validación.

Fase Metodológica 1 - Concepción

Esta fase tuvo como principal objetivo la especificación de los requerimientos de la aplicación. Constó del levantamiento de la información requerida, priorización posterior de esos requerimientos con el fin de determinar el nivel de impacto para la aplicación [26]. Adicionalmente, en esta fase se determinó el alcance del proyecto, este dictó el límite del desarrollo de la aplicación, su arquitectura y la elaboración general.

Esta primera fase metodológica (RUP) se basa en “entender los requerimientos y determinar el alcance del desarrollo” [27]. Una buena idea en esta fase es especificar una visión del producto final. De la metodología ágil XP se rescató la autoría colectiva en el desarrollo de los documentos que se llevaron a cabo en esta fase y la planeación del alcance de la siguiente fase.A continuación, se listan las actividades que se realizaron para la concepción metodológica:

- Especificar requerimientos:
  - Levantar requerimientos
  - Describir requerimientos
  - Priorizar requerimientos
- Determinar alcance:
  - Seleccionar los aspectos claves que debe incluir la aplicación.
  - Validar alcance.
- Visión general de la arquitectura

El resultado de esta fase es el documento SRS (software requirements specification [28]). Este documento describe el sistema de software a ser desarrollado, muestra los requerimientos funcionales y no funcionales e incluye un conjunto de casos de uso que el software debe proveer.

Fase Metodológica 2 – Elaboración

Esta fase tuvo como principal objetivo diseñar la arquitectura teniendo en cuenta los diferentes roles de usuarios. Cabe resaltar que de acuerdo al Software Engineering Institute (SEI), la Arquitectura de Software se refiere a "las estructuras de un sistema, compuestas de elementos con propiedades visibles de forma externa y las relaciones que existen entre ellos" [29]. Adicionalmente en esta fase se llevó a cabo la validación de esta arquitectura, pasando a través del diseño contra los requerimientos actuales y cualquier posible requerimiento a futuro [30]. También se efectuó la especificación de los casos de uso, siendo estos un reconocimiento de los requisitos que debe alcanzar el proyecto [31].

En la fase de elaboración se planearon las actividades necesarias y los recursos requeridos para especificar las características y el diseño de la arquitectura. Esta fase se enfocó en la realización de los casos de uso a partir de los requerimientos funcionales [27]. Adicionalmente para visualizar la arquitectura de software se utilizó UML[32]. Cabe resaltar que la presente fase se realizó de manera iterativa. De la metodología ágil XP, se rescató la autoría colectiva en el desarrollo de los documentos que se lleven a cabo en esta fase. Se realizó el refinamiento de los resultados entregados en la fase anterior y también la planeación del alcance de la siguiente fase.A continuación, se listan las actividades que se llevaran a cabo para la concepción metodológica:

- Refinamiento de requerimientos
- Diseñar la arquitectura teniendo en cuenta los roles de usuarios
    - Realizar diagrama de componentes
    - Realizar diagrama de despliegue
    - Realizar diagrama de interacción
    - Realizar diagrama de secuencia
    - Realizar diagrama de entidades

- Especificación de casos de usos y su arquitectura
    - Realizar diagrama de casos de uso
    - Especificación de escenarios
- Validar arquitectura

Los resultados esperados de esta fase fueron:

- SRS - refinamiento
- SDD (software design document [33]) contiene la descripción del sistema en términos de las vistas arquitecturales para diferenciar los aspectos de esta.

Fase Metodológica 3 – Construcción

Esta fase llevó a cabo dos objetivos principales del proyecto: desarrollar prototipo funcional de la aplicación y documentar los manuales de la aplicación.La fase de construcción se centró en la implementación del software basado en el diseño realizado de las fases anteriores. La construcción del producto de software tuvo una visión evolutiva en la cual se entrega un producto a un usuario final [27].

Cabe resaltar que la presente fase metodológica se realizó de manera iterativa. De la metodología ágil XP, se rescató la autoría colectiva en el desarrollo de los documentos y el código del software que se lleve a cabo en esta fase y también la planeación del alcance de la siguiente fase, la programación por parejas con estándares de código. Se realizó el refinamiento de los resultados entregados en la fase anterior.A continuación, se listan las actividades que se llevaran a cabo para la concepción metodológica:

- Refinar requerimientos
- Refinar diseño
- Desarrollar prototipo funcional de la aplicación
    - Refinar requerimientos
    - Implementar requerimientos funcionales
    - Verificar cumplimiento de requerimientos no funcionales
- Documentar los manuales de la aplicación

Los resultados esperados de esta fase fueron:

- SRS- refinamiento
- SDD- refinamiento
- Prototipo funcional
- Manuales de la aplicación
    - Memoria del trabajo de grado

Fase Metodológica 4 – Verificación y Validación

Esta fase no se encuentra especificada en la metodología RUP. Sin embargo, se considera importante su realización ya que determinó la eficacia por medio del equipo técnico y usabilidad del prototipo llevado a cabo a través de usuarios. La verificación y validación actúan sobre los productos intermedios que se desarrollan para detectar y corregir sus defectos y desviaciones respecto al objetivo fijado. También se utilizan para disminuir riesgos, mejorar la calidad y fiabilidad del software [34].

Cabe resaltar que la presente fase metodológica se realizó de manera iterativa. De la metodología ágil XP, se rescató la autoría colectiva en el desarrollo de los documentos y el código del software que se lleve a cabo en esta fase y también la planeación del alcance de la siguiente fase, la programación por parejas con estándares de código. Se realizó el refinamiento de los resultados entregados en la fase anterior.A continuación, se listan las actividades que se llevaran a cabo para la concepción metodológica:

- Refinar diseño
- Refinar prototipo
- Validar el software con pruebas de aceptación por parte del director de trabajo de grado
- Pruebas:
    - Pruebas unitarias
    - Pruebas de sistema
- Ajustar errores
- Evaluar progreso.

Los resultados esperados de esta fase fueron:

- SDD - refinamiento
- Prototipo funcional
- Pruebas de validación y verificación del ministerio de salud [8].

II – MARCO TEÓRICO

1. Marco Conceptual

De acuerdo con el Ministerio de Salud de Colombia, el Triage consiste en una valoración clínica breve que determina el tiempo y la secuencia en que será atendido el paciente, con unos recursos limitados. El objetivo es definir las pautas que permitan priorizar la atención de los usuarios a su llegada a urgencias a través de una evaluación de enfermería rápida y eficiente. Aunque esta es una guía para pacientes adultos puede ser utilizada para pacientes pediátricos y diseñada para subespecialidades debido a que los conceptos que maneja el sistema son amplios y generales [21].

La estandarización de los sistemas del Triage trae ventajas para médicos, pacientes, personal de salud e instituciones de salud puesto que formaliza un lenguaje común para la escala de prioridades y el proceso de toma de decisiones. Algunos de los objetivos del sistema de Triage son identificar los casos en los que la vida del paciente se encuentra en riesgo, organizar un grupo de pacientes que puede continuar con el proceso regular de atención y asignar un área de tratamiento acorde a las necesidades [25].

El proceso del Triage se realiza de acuerdo con el diagrama de flujo de toma de decisiones. Inicia valorando al paciente aplicando criterios que definirán la prioridad más urgente. En caso de ser afirmativo, se debe iniciar la reanimación y el traslado al área de tratamiento. Si se determina que no es prioridad I, pero sí prioridad II, se debe ingresar al área de tratamiento recibiendo atención inmediata de enfermería. Si el paciente es determinado como prioridad III, IV o V; el paciente se registra en admisiones, aguardará en la sala de espera el tiempo determinado dependiendo de su nivel de prioridad y recibirá atención médica y de enfermería. Estas prioridades serán explicadas en mayor detalle en la tabla de Escala de priorizaciones.

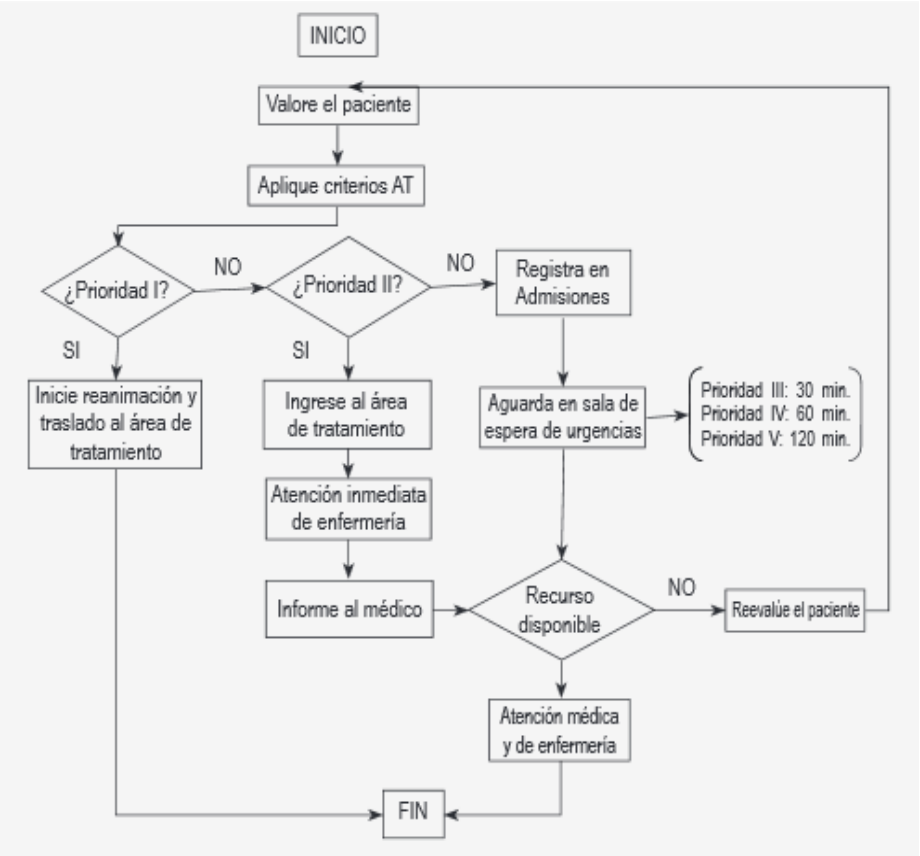


Ilustración 1: Diagrama de decisiones en Triage[21]

El siguiente formato corresponde al formato oficial del Ministerio de Salud de Colombia con los respectivos espacios que se deben completar con la información pertinente. Este formato permite completar: la fecha en la que el paciente ingresa a la sala de urgencias; el nombre del mismo; la edad; HC (historia clínica) en donde se colocará el número de identificación del paciente; el Motivo de la Consulta, campo completado de manera abierta; GSC: __/15, alerta (consiente) - obnubilación (somnoliento, alerta por estímulos menores) – estupor (previo al coma, paciente dormido que sólo responde a estímulos vigorosos) – coma (falta de respuesta a cualquier tipo de estímulo interno o externo), escala de Glasgow de coma que dice qué tan consiente está un paciente [21][35]; los signos vitales: PA (presión arterial), FC (frecuencia cardiaca), FR (frecuencia respiratoria), temperatura; marcar con una casilla si el pulso es regular o irregular; oximetría de pulso, cantidad de hemoglobina oxigenada [36];

glucometría, es la cantidad de glucosa en sangre [37]; antecedentes relevantes; y por último la firma de la enfermera que atendió al paciente al ingresar a urgencias.

PRIORIDAD I-REANIMACIÓN ☐ PRIORIDAD II-EMERGENCIA ☐
PRIORIDAD III-URGENCIA ☐ PRIORIDAD IV-URGENCIA MENOR ☐
PRIORIDAD V-NO URGENTE ☐

FECHA: _____ NOMBRE_____
HC: _____ EDAD _____

MOTIVO DE CONSULTA: _____
_____
_____

GSC: ____/15 ALERTA ☐ OBNUBILACIÓN ☐ ESTUPOR ☐ COMA ☐

SIGNOS VITALES:

PA: _____/_____ FC: _____ FR: _____ Temperatura: _____
PULSO: REGULAR IRREGULAR
OXIMETRÍA DE PULSO: _____% GLUCOMETRÍA:_____mg/dL
ANTECENDENTES RELEVANTES: _____
_____

FIRMA DE LA ENFERMERA: _____

Ilustración 2: Formato Triage [21]

Luego de completar este formato, el personal sanitario se encargará de marcar la casilla correspondiente de acuerdo a la prioridad que considere. La tabla presentada muestra los diferentes tipos de prioridades con su respectivo tiempo de respuesta y una breve descripción del estado del paciente clasificado en esa categoría.

Escala	Tiempo de respuesta	Descripción
Prioridad I (Reanimación)	Atención: inmediata	El paciente presenta una condición que amenaza su vida y requiere intervención inmediata. (Dificultad respiratoria severa).
Prioridad II (Emergencia)	Atención: 15 minutos	Paciente cuyo problema representa un riesgo potencial de amenaza a la vida o pérdida de una extremidad u órgano si no recibe una intervención médica rápida.
Prioridad III (Urgencia)	Atención: 30 minutos	Paciente con estabilidad respiratoria y cuyas condiciones pueden progresar a problemas serios que requieren intervención de emergencia.
Prioridad IV (Urgencia menor)	Atención: 60 minutos	Condiciones relacionadas con la edad del paciente, angustia o deterioro potencial o complicaciones, que se beneficiará de la intervención o de tranquilizarlo.
Prioridad V (No urgente)	Atención: 120 minutos	Condiciones que pueden ser agudas, pero no comprometen el estado general del paciente y no representan un riesgo evidente

Tabla 1: Escala de prioridades de Triage [23]

2. Marco Contextual

Trabajos similares han sido planteados con anterioridad en diferentes países del mundo.Soluciones que brindan ayudas a los usuarios y médicos, han sido diseñadas para automatizar el proceso del Triage. Se presentan en seguida aplicaciones importantes en el área:

ITriage

Es una aplicación que está disponible para Android y IOS, así como para exploradores de Internet en idioma inglés. Permite buscar los síntomas utilizando un avatar o una lista ordenada alfabéticamente y seleccionar las causas de las dolencias desde las más comunes

hasta las menos frecuentes. Posteriormente, la aplicación brindará las posibles enfermedades y sus tratamientos. Esta aplicación está orientada principalmente al personal de salud que tiene conocimientos avanzados en medicina.Adicionalmente, es capaz de recomendar hospitales, farmacias o médicos según lo requiera el usuario y además brinda información para poder pedir una cita, dando el tiempo de espera para ser atendido e información de las salas de emergencia. Por último, esta aplicación le permite al usuario almacenar sus datos médicos en el dispositivo[17][39].

Triage territorial

Es una aplicación que está disponible para Android y IOS que integra dos tipos de Triage: START Triage, para adultos y JUMP START para niños. Se encuentra disponible en inglés e italiano. La aplicación permite ingresar los síntomas y las posibles causas para determinar una posible aproximación a las clasificaciones del Triage[40].

CTAS Triage Official

Canadian Triage and Scale es una aplicación para Android y IOS que guía al usuario a través de las quejas de los pacientes en la asistencia a los centros clínicos para asignar un nivel de prioridad. Se encuentra en inglés o francés dependiendo de la configuración del idioma del dispositivo. Esta aplicación está actualmente disponible para Canadá y permite ingresar los síntomas y las causas de las dolencias [42].

EMS ACLS Guide

EMS ACLS Guide está disponible para Android y IOS en inglés. Es una aplicación utilizada por los paramédicos para realizar el Triage y manejar el trauma. Provee un acceso rápido y fácil de información vital asistida, dosis de medicamentos, pasos sencillos para leer el electrocardiograma, recopilación de síntomas y por último instrucciones para reanimación[44].

Existen algunas otras aplicaciones que brindan una priorización del Triage, pero se encuentran enfocadas a accidentes de grandes cantidades de personas. Unos ejemplos de ellas son:

Quick Triage App

Esta aplicación se encuentra disponible para Android y IOS. Integra dos tipos de Triage: START Triage, para adultos y JUMP START para niños. El principal objetivo es proporcionar la asistencia rápida para la clasificación y etiquetado de las muertes en un incidente de víctimas en masa. Cabe resaltar que esta aplicación sólo funciona para casos en los que los pacientes sean víctimas de accidentes graves [19].

Fast Triage App Lite

Fast Triage App Lite está disponible para Android en inglés. La aplicación filia, clasifica y establece prioridades asistenciales en incidentes con múltiples lesionados y/o catástrofes. El componente principal de esta aplicación es que estima porcentualmente la probabilidad de supervivencia de las personas accidentadas junto con una orientación de posibles intervenciones terapéuticas [43].

La tabla a continuación, presenta en forma de resumen las diferentes características de las aplicaciones analizadas a lo largo de esta sección junto conla aplicación TAppi. El signo +*(positivo)*significa que la aplicación cumple con esa característica; el – *(negativo)*, significa que la aplicación no cumple con esa característica.

	ITriage	Triage Territorial	CTAS - Triage – OFFICIAL	EMS ACLS Guide	TAppi
Posibles causas: el usuario puede ingresar las posibles causas que provocaron su dolencia	+	+	+	-	+
Direccionamiento al médico: se presenta al usuario una lista de los posibles hospitales o clínicas a las que puede dirigirse	+	-	-	-	+
Direccionamiento a la farmacia: se presenta al usuariouna lista de las posibles farmacias a las que puede dirigirse	+	-	-	-	+

Líneas de emergencia: se presenta al usuario una lista de las posibles líneas de emergencia a las que puede comunicarse	+	-	-	-	+
Tiempo promedio de espera en el hospital: Se presenta al usuario un promedio de tiempos en los que será atendido de acuerdo a los hospitales presentados en la lista	+	-	-	-	+
Citas con una selección de doctores: se presenta al usuario la posibilidad de agendar una cita con alguno de los doctores disponibles	+	-	-	-	-
Almacenamiento del historial médico: el usuario podrá guardar información básica acerca de su historial médico que no interfiera con la confidencialidad médico-paciente	+	-	-	-	+
Posibilidad de ser realizado por diferentes roles: usuarios con o sin conocimiento de medicina. Para cada tipo de usuario se muestra una adaptación al despliegue	-	-	-	-	+

Tabla 2: Tabla resumen comparación aplicaciones

La aplicación TAppise basa en el desarrollo de un sistema de Triage estructurado que incluye la recepción, la acogida y la clasificación tentativade los pacientes [10]. El proyecto está enfocado en la estandarización de una aplicación que permita la captura inteligente de datos y de información básica del usuario (nombre, edad, sexo, cédula de ciudadanía), con el fin de agilizar el proceso del Triage proponiendo una clasificación tentativa.

Otra de sus funcionalidades, debido a que funcionará en un dispositivo Smartphone, es la adaptación al despliegue respecto a los diferentes usuarios que vayan a usar la aplicación y no respecto al dispositivo.Finalmente, cabe resaltar que esta aplicación se basa en algoritmos de toma de decisiones desarrollados a partir del sistema basado en reglas[46], los cuales transforman datos en una estructura organizada para su posterior uso.

III – ANÁLISIS

"Es más fácil cambiar las especificaciones para que encajen con el software que hacerlo al revés" **Alan Perlis**

1. Requerimientos

El objetivo de realizar el levantamiento de requerimientos tiene como fin la aceptación de las funcionalidades que se han especificado. Es importante resaltar que este proceso recibió retroalimentaciones por parte del cliente, por lo que los nuevos requerimientos fueron añadidos como una descripción de las nuevas funcionalidades [47].

Una vez realizado el levantamiento y la revisión de los requerimientos, se hizo necesario llevar a cabo el proceso de especificación para describir de manera explícita las características y funcionalidades del sistema. Esta especificación tiene como objetivo ser la herramienta guía para el grupo de trabajo encargado de desarrollar la aplicación [47].

Para el desarrollo de Tappi: Triage Application y teniendo en mente el objetivo principal, se han definido los diferentes tipos de actores; la tabla a continuación muestra una descripción de cada uno de ellos y otros actores importantes para los casos de uso.

Actor-Rol	Descripción
Usuario	Usuario sin conocimientos médicos.
Usuario Medico	Usuario que presenta conocimientos médicos y maneja lenguaje técnico.
Celular	Es el dispositivo en el cual la aplicación interactúa con el cliente.
Base de datos	Es donde se almacena la persistencia de los datos que utiliza la aplicación.

Tabla 3: Actores del sistema

La siguiente tabla muestra cada uno de los requerimientos, los primeros requerimientos hacen referencia al CRUD (create, read, update, delete) de la aplicación, seguidos a estos están los requerimientos referentes al Triage y por último los requerimientos no funcionales.

No.	Requerimiento
1	El sistema debe dar la opción de iniciar sesión con cuentas de Facebook (la validación está implícita usando este sistema de login)
2	El sistema debe verificar que el usuario ha aceptado los términos y condiciones de Tappi
3	El sistema debe mostrar el enlace a los términos y condiciones durante el proceso de registro
4	El sistema debe notificarle al usuario si ha realizado su registro correctamente.
5	El sistema debe dar la opción al usuario de editar los datos de su cuenta
6	El sistema debe dar la opción al usuario de ingresar sus síntomas
7	El sistema debe verificar que los datos que se van a cambiar sean correctos
8	El sistema debe tener un botón para aceptar los cambios realizados
9	El sistema debe tener una sección para editar cuentas
10	El sistema debe permitir cerrar la sesión
11	El sistema debe dar la opción de borrar una cuenta creada
12	El sistema debe mostrar un botón de confirmación antes de ser borrada la cuenta
13	El sistema debe impedir acceder a una cuenta si esta ha sido borrada
14	El sistema debe eliminar la cuenta de la base de datos cuando esto sea solicitado
15	El sistema permitirá que el usuario administrado bloquee a un usuario.
16	El sistema debe mostrar un mapa geográfico de la ubicación del centro de salud

	más cercano
17	El sistema debe mostrar un mapa geográfico de la ubicación de la farmacia más cercana
18	El sistema debe mostrar una lista con las líneas de emergencia
19	El sistema debe mostrar una lista de posibles enfermedades congénitas.
20	El sistema debe almacenar el motivo de la consulta.
21	El sistema debe tener una sección para consulta de servicios
22	El sistema debe almacenar los tratamientos quirúrgicos que el paciente pudo ser sometido.
23	El sistema presenta una lista de antecedentes de trastornos mentales posibles (depresión mayor, trastorno bipolar, trastornos psicóticos, ansiedad, trastorno alimentario, trastorno personalidad, impulsividad y agresión)
24	El sistema debe mostrar una lista de opciones para la anamnesis remota en un vocabulario que comprenda los diferentes usuarios del sistema.
25	Anamnesis remota: El sistema debe almacenar los antecedentes mórbidos (HTA, dislipidemia, asma, diabetes historia de cáncer.
26	Anamnesis remota: El sistema debe almacenar los antecedentes gineco-obstétricos (Menarquia, menopausia)
27	Anamnesis remota: El sistema debe almacenar los hábitos del usuario (Tabaco, Alcohol, Otras).
28	Anamnesis remota: El sistema debe calcular el IMC (Requiere de peso y altura).
29	Anamnesis remota: El sistema debe almacenar los medicamentos que la persona consume.

30	Anamnesis remota: El sistema debe almacenar las alergias que el paciente presenta.
31	Anamnesis remota: El sistema almacena los Antecedentes familiares.
32	Anamnesis remota: El sistema almacena las inmunizaciones de usuario.
33	El sistema debe evitar ataques de Inyección SQL.
34	El sistema debe poder ser accedido desde cualquier smartphone Android
35	El sistema debe ser fácil de entender para el usuario.
36	El sistema debe tener los colores definidos de tappi
37	El sistema debe responder a las acciones solicitadas en menos de un minuto
38	La información que se envía a través de internet será encriptada por código HASH.
39	Un usuario no autenticado no puede hacer ningún tipo de operaciones en el sistema aparte de crear cuenta o autenticarse.
40	Un usuario podrá realizar cualquier operación del sistema siguiendo los pasos descritos en el manual.
41	Cuando se instala una nueva versión del sistema todos los ajustes y configuraciones que hayan sido definidos previamente persistirán.
42	El sistema debe soportar 4 peticiones simultáneas.
43	El sistema debe ser programado en Android Studio.
44	El versionamiento de la aplicación deberá ser administrado en github
45	La aplicación deberá utilizar un paradigma orientado a objetos.
46	El sistema debe almacenar datos correspondientes a 2 GB de espacio.
47	Los usuarios deben autenticarse ante el servicio de directorio centralizado de

	usuarios de TAppi.
48	El sistema debe recuperar su estado frente a un fallo.
49	El sistema debe emitir una alerta cuando se encuentre un fallo.
50	El sistema debe utilizar un vocabulario simple para el usuario paciente.
51	El sistema debe utilizar un vocabulario técnico para el paciente médico.
52	La persistencia del sistema basado en reglas se encontrará en un archivo XML

Tabla 4: Requerimientos

2. Casos de uso

Tan pronto como se realizó la ingeniería de requerimientos, fue imprescindible continuar con la descripción de los casos de uso. Estos se utilizan en el modelado de las aplicaciones y representan las acciones que realiza cada tipo de usuario[48], [49].

El diagrama de casos de uso mostrado a continuación, representa las funcionalidades que tiene la aplicación. El diagrama muestra las relaciones de cada uno de los actores con las posibles interacciones de las funcionalidades de la aplicación.

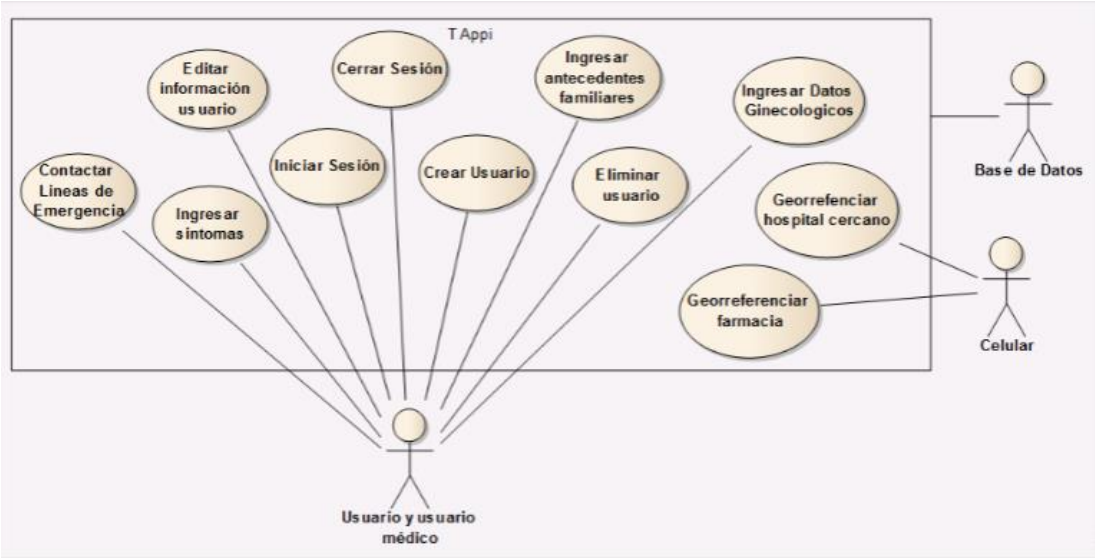


Ilustración 3: Diagrama de casos de uso

La priorización de los casos de uso se llevó a cabo por los tres miembros del grupo junto con el director de trabajo de grado, verificando la prioridad, el contenido de la propuesta del trabajo de grado y la dificultad. Cada uno de los miembros discutió la relevancia de dicho caso de uso y se asignó un puntaje de alto, medio o bajo, justificando su respuesta con una razón grupal.

Id CU	Caso de uso	Prioridad	Razón
1	Iniciar sesión	Alto	Sin inicio de sesión no es posible acceder a las funcionalidades
2	Cerrar sesión	Baja	Al no estar asociado a otras funcionalidades, no es muy importante
3	Crear usuario	Alto	Un usuario registrado puede hacer uso de las funcionalidades de la aplicación
4	Eliminar usuario	Bajo	Al no estar asociado a otras funcionalidades, no

			es muy importante
5	Ingresar antecedentes familiares	Alto	Este tipo de información ayudará con el proceso de priorización del Triage
6	Editar información de usuario	Medio	En caso de que el usuario requiera modificar alguno de los campos de información
7	Ingresar síntomas	Alto	Este caso de uso es la funcionalidad primordial del trabajo de grado. Sin él no es posible realizar el proceso del Triage.
8	Ingresar datos ginecológicos	Alto	Este tipo de información ayudará con el proceso de priorización del Triage
9	Georreferenciar hospitales	Bajo	Indica los hospitales más cercanos (funcionalidad de TApppi)
10	Georreferenciar farmacias	Bajo	Indica las farmacias más cercanas (funcionalidad de TApppi)
11	Contactar líneas de emergencia	Bajo	Muestra una lista de las líneas de emergencia disponibles en Colombia.

Tabla 5: Priorización casos de uso

IV – DISEÑO

"Hay dos maneras de diseñar software: una es hacerlo tan simple que sea obvia su falta de deficiencias, y la otra es hacerlo tan complejo que no haya deficiencias obvias" – C. A. R. Hoare

En la siguiente sección del documento se consignaron los modelos, diagramas y artefactos de diseño que son relevantes para la construcción del software. La realización de estos diagramas representó para el grupo de trabajo una abstracción sobre los componentes que representan los aspectos físicos y lógicos del sistema. Cabe resaltar que los diagramas aquí mostrados demuestran el comportamiento ideal, la manipulación y el almacenamiento de la información y la interacción con el usuario.

Como primera medida se muestran las herramientas escogidas para el desarrollo de la aplicación.La tabla mostrada a continuación, presenta de forma detallada las interfaces de software. Para cada una de las interfaces se encuentra una breve descripción, su razón de uso y la versión que se utilizó. El servidor de la aplicación se encuentra alojado en la nube, pagando un servicio a un tercero con el fin de tener mayor escalabilidad y acceso a los servicios.

	GitHub	Android	Glassfish	SQL Developer	Derby	Glassfish driver	Ubuntu
Descripción	Plataforma de sistema de control de versiones distribuido que ofrece el almacenamiento de código y documentos mediante un repositorio basado en Git. [50]	Sistema Operativo móvil basado en Linux [51].	Servidor http utilizado para sostener la lógica de negocio de JavaEE.	Es una herramienta visual para el diseño de las bases de datos con el fin de facilitar su administración. [52]	Es un sistema de manejo de base de datos relacionales.	Permite conexiones con la base de datos.	Sistema operativo basado en Debian [53].

Propósito de uso	Gracias a que GitHub no posee limitantes en la cantidad de colaboradores, permite que los desarrolladores puedan subir, bajar, eliminar o actualizar elementos al repositorio de la página web. [50]	Es uno de los sistemas operativos móviles más usados en el mercado.	Permite utilizar el protocolo http.	Diseñar la base de datos relacional que va a soportar la información (tablas) de la aplicación.	Almacena la información de los usuarios y es accesible por múltiples usuarios.	Conectarse con la base de datos por pool y jdbc.	Manejar los procesos del servidor para retornar datos a los clientes.
Versión	GitHub, Inc. https://github.com/about	Samsung A300M	4.1	4.1.3.20	Relacionada con Glassfish.	4.1	14.04.1 Versión que se conecta con el servidor.

Tabla 6: Herramientas de desarrollo

Respecto a la base de datos la elección se encontraba entre Derby y Orión, base de datos proveída por la Pontificia Universidad Javeriana. Se eligió la primera puesto que el manejo de la información, el inicio y restauración la pudieron llevar a cabo los desarrolladores sin recurrir a terceros. Adicionalmente, esta se encuentra alojada en el mismo servidor por lo que el acceso a los servicios y datos se consideró más rápido que en una base de datos externa. El diseño y desarrollo de la base de datos se llevó a cabo en SQL Developer, puesto que era un software con el cual los desarrolladores ya se encontraban familiarizados y por lo tanto su curva de aprendizaje, uso y realización de los diagramas se llevó a cabo de manera rápida y fácil.

Se eligió una plataforma móvil y no web para desarrollar la aplicación ya que en el 2015 el mercado de los teléfonos inteligentes aumento en un 13% según IDC (International Data Corporation) [54]. Además, el lenguaje de programación para el dispositivo móvil fue

Android ya que el mercado desde el 2013 hasta el 2015 ha estado dominado por este sistema operativo hasta un 84.8% del mercado según la IDC cifras hasta agosto de 2015. Por último, para el protocolo de comunicación se estaba entre SOAP y REST, REST sin seguridad funciona más rápido usando java como lenguaje de programación con cadenas más largas por tiempo en milisegundos [55], debido a ello se eligió este como opción.

Para consultar información acerca de la comparación de herramientas, ver Anexo Comparación de herramientas

Vista lógica

El diagrama de componentes que representa la vista lógica del sistema, muestra los diferentes artefactos que representan elementos de la vida real, así como interfaces y las relaciones entre ellos. Para el trabajo de grado TAppi: Triage Application se utiliza el patrón arquitectural MVC, modelo-vista-controlador [56], que facilitará el desarrollo según el comportamiento de los artefactos. Adicionalmente utiliza una arquitectura basada en JavaEE, lo que "es tanto un lenguaje de programación como una plataforma de desarrollo, Java es un lenguaje de alto nivel orientado a objetos. Java Platform, Enterprise Edition (Java EE) provee un API y un entorno de tiempo de ejecución para desarrollar aplicaciones de múltiples capas, escalables, confiables y seguras" [57].

"El patrón Modelo-Vista-Controlador (MVC) surge con el objetivo de reducir el esfuerzo de programación, necesario en la implementación de sistemas múltiples y sincronizados de los mismos datos, a partir de estandarizar el diseño de las aplicaciones. El patrón MVC es un paradigma que divide las partes que conforman una aplicación en el Modelo, las Vistas y los Controladores, permitiendo la implementación por separado de cada elemento, garantizando así la actualización y mantenimiento del software de forma sencilla y en un reducido espacio de tiempo" [58]. La información relevante del servidor es la siguiente:

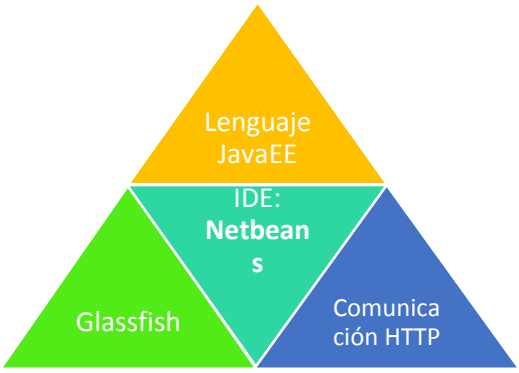


Ilustración 4: Arquitectura general

La vista de componentes contiene las diferentes capas de software que debe presentar el sistema. A continuación, se enuncian estas con su contenido.

Android_TAppi:

- Vista_CelTAppi: contiene las pantallas (vistas) más relevantes que debe poseer la aplicación en Android.
- Controlador_CelTAppi: los controladores del celular que ejecutan los comando a llevar a cabo.
- Integración_TAppi: es la capa que contiene el proxy para consumir los servicios del servidor.

Servidor:

- Servicios_CelTAppi: es la capa de servicios del servidor que brinda la información al celular.
- Negocio_TAppi: es la capa que contiene toda la lógica de negocio de la aplicación con sus respectivas fachadas.
- Entidades: son las entidades que utiliza el software con el fin de asignar, transportar y utilizar información de la base de datos.

Elementos externos:

- GPS: componente del celular para brindar la localización del mismo.
- Base de datos: contiene los datos del software y maneja la persistencia

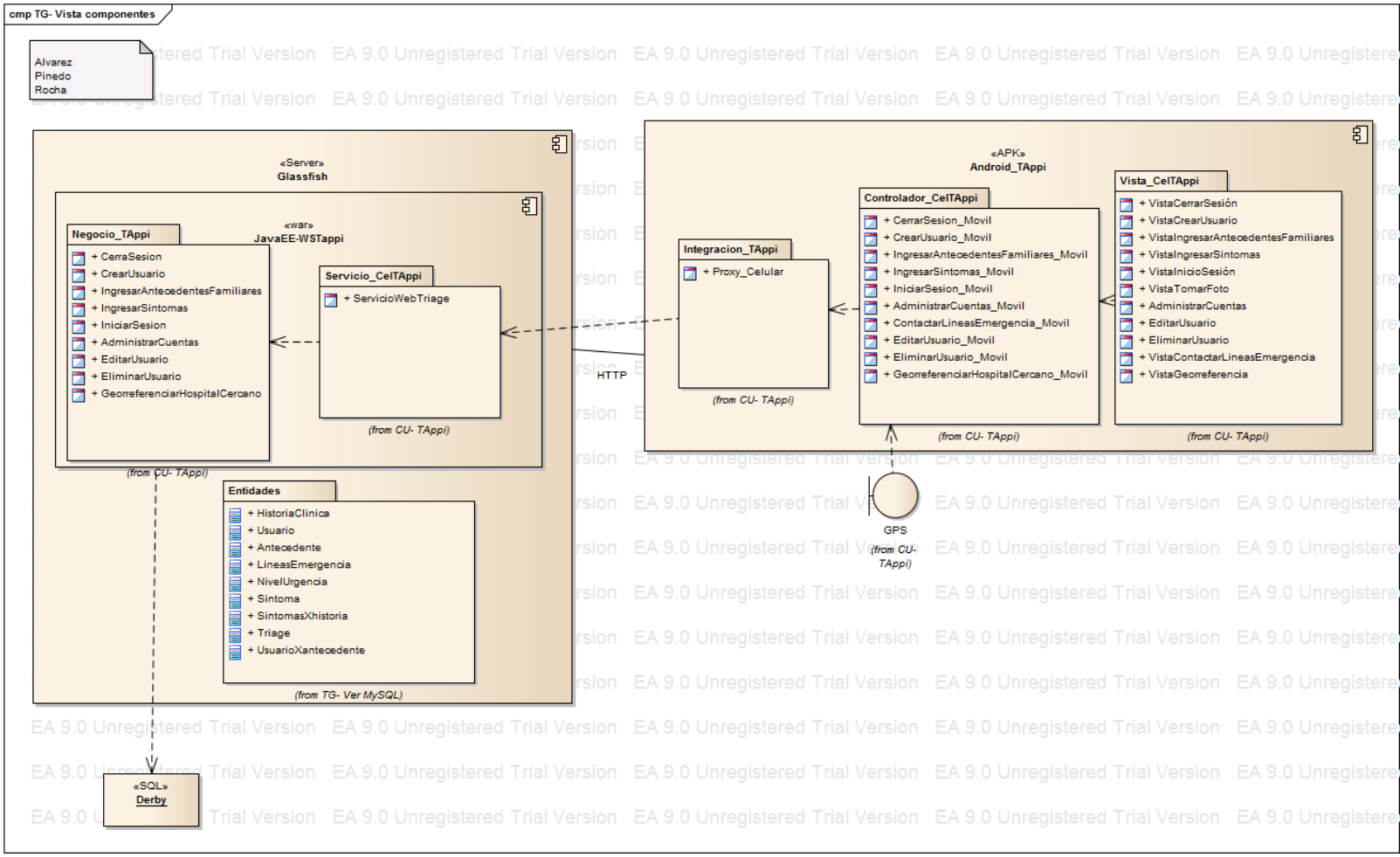


Ilustración 5: Vista de componentes

Vista Física, despliegue del Sistema

El siguiente diagrama de despliegue muestra los componentes físicos (hardware) más importante del sistema y dónde los componentes de software serán instalados. Para mayor información referirse al documento de diseño SDD.

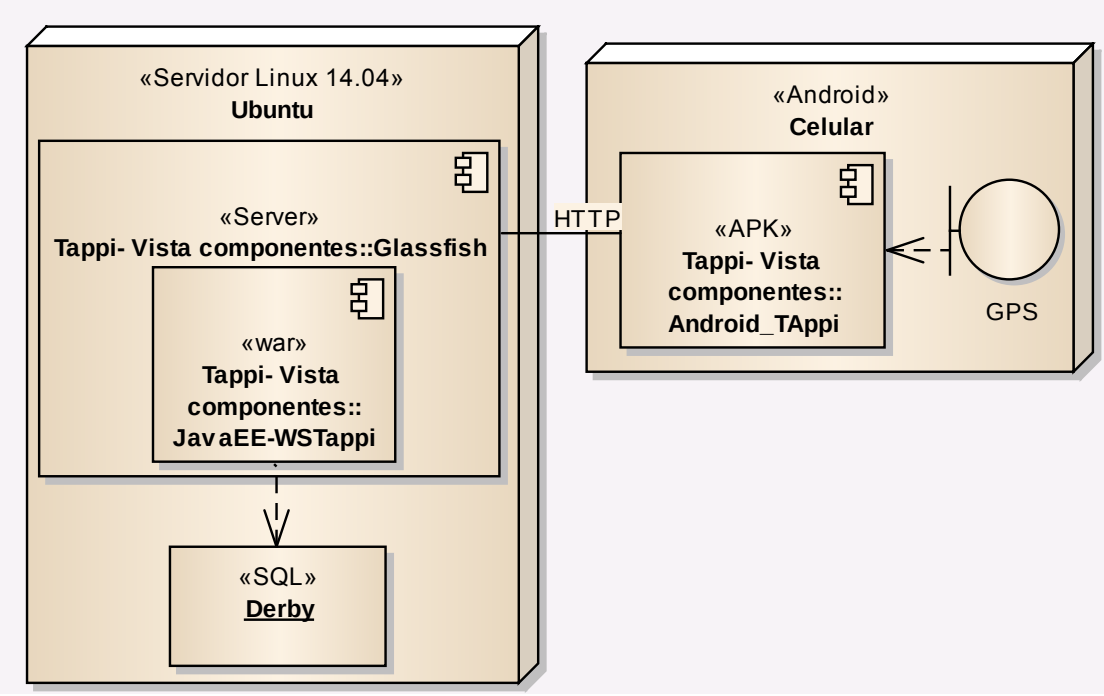


Ilustración 6: Vista despliegue

Máquina virtual

Se tuvo una máquina virtual la cual constó de lo siguiente:

- Servidor alojado en http://donweb.com
- Número de Cores (CPU): 2 v
- Espacio en RAM: 2GB
- Espacio en Disco: 20 GB (SSD)
- Sistema operativo: Linux (Ubuntu 14.04.4 64 bits)
- Puertos TCP usados:
  - 1527, base de datos
  - 8080, puerto para consultas de la aplicación HTTP

- 4848, consola de administración Glassfish

- Ip publica: 200.58.126.15
- Ip privada: 192.168.200.22
- Precio: $70.000 (COP) mensuales.

Celular

Dispositivo Android con las siguientes características:

- Sistema operativo: Android 4.4.4
- Procesador mínimo: 1.2 GHz Quad Core
- Memoria del sistema: 4.08 GB
- Espacio libre en memoria interna del teléfono: 11.92 GB
- Memoria interna:16.00 GB
- Función del GPS: si
- APK: archivo ejecutable, que contiene la aplicación de TAppi con el fin de que el usuariopueda utilizarla.

Modelo de clases del sistema

El modelo de clases del sistema esquematiza las clases que utilizó el software en la lógica de negocio con su determinada cardinalidad o relación entre clases, de ese modo se conoce como estas interactúan en la aplicación. Cabe resaltar que algunas clases que se encuentran en la imagen representan las llaves primarias de acceso para acceder a ciertos servicios de las entidades con el fin de mantener un buen seguimiento de la información (estas tienen las siglas PK). Las clases que contiene dicho diagrama son las siguientes:

- Usuario: almacena la información relevante del usuario de la aplicación, tal como el nombre, correo, datos ginecológicos, etc.
- Historia clínica: almacena los datos concernientes a la consulta de una dolencia del usuario, esta tiene la fecha de dicha consulta y su priorización tentativa del Triage.
- SíntomasXhistoria: esta clase provee la información de intersección entre la lista de síntomas que puede elegir el paciente y una historia clínica determinada.

- Síntoma: esta clase contiene todos los síntomas que puede presentar un paciente, los tiene en términos médicos y términos coloquiales.
- Nivel de urgencia: dicha clase contiene la asociación entre un grupo de síntomas y el nivel de Triage tentativo que estos manejan.
- Triage: almacena la información de prioridad tentativa en un Triage determinado como valor numérico.

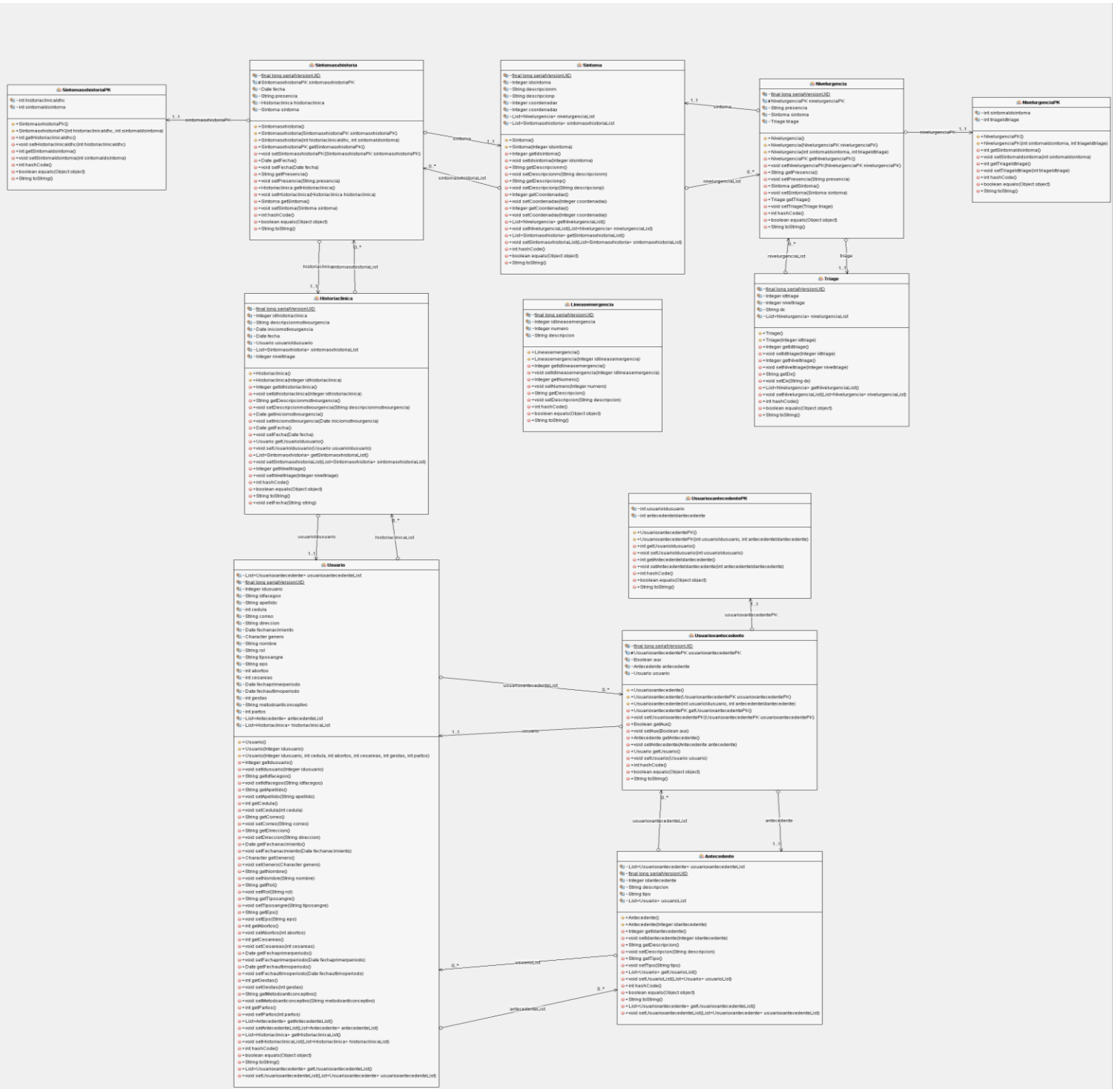


Ilustración 7: Modelo de clases

Vista de Procesos del Sistema

A continuación, se describe en detalle el proceso de un usuario que navega a través de la aplicación.

En primera instancia aparecerá la pantalla principal la cual brinda la opción de ingreso por medio de cuentas de Facebook. En caso de que el usuario ingrese de forma correcta sus datos, la aplicación permite escoger entre 5 opciones: Ingresar síntomas, Números de emergencia, Cuenta, Historial, Salir. Cada uno de estas opciones serán explicadas con mayor detalle más adelante. El ciclo finaliza cuando el usuario decide dejar de navegar en la aplicación.

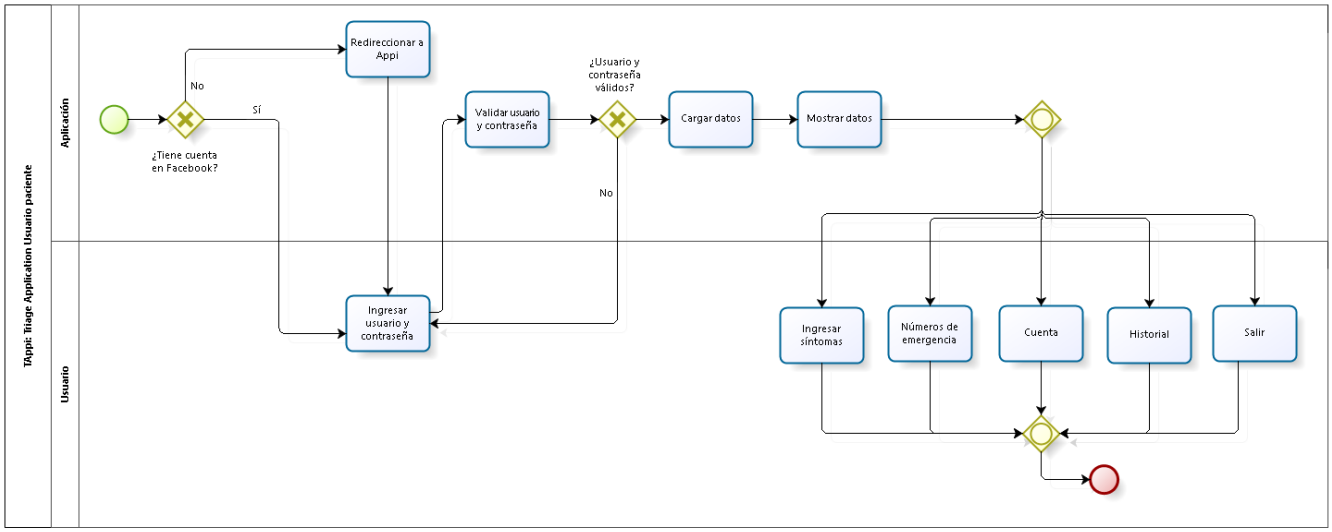


Ilustración 8: Procesos del sistema usuario paciente

Crear o editar información de usuario

Cuando un nuevo usuario quiere crear un perfil completamente nuevo, debe completar sus datos sin dejar casillas vacías. Posteriormente, la información ingresada será validada y almacenada en la base de datos.

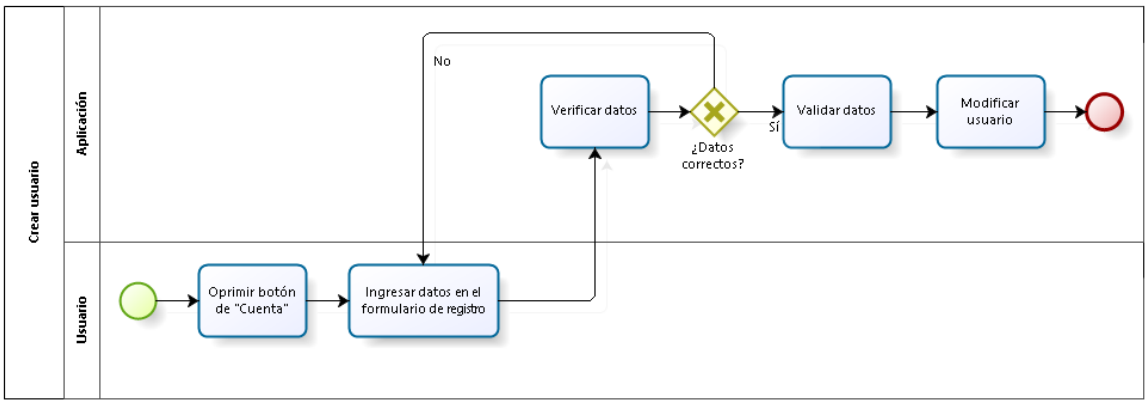


Ilustración 9: Procesos del sistema, crear usuario

Ingresar síntomas

EL proceso de ingresar síntomas para un usuario paciente, inicia al oprimir el botón de *Ingresar síntomas*, es importante resaltar que de acuerdo al nivel de conocimiento del usuario se le mostrará de diferente manera la información. El usuario debe activar el GPS de su *smartphone* en caso de que esté desactivado. Cuando el usuario finaliza el ingreso de síntomas, la aplicación le mostrará una priorización de Triage tentativa y una lista de hospitales más cercanos de acuerdo a su posición geográfica.

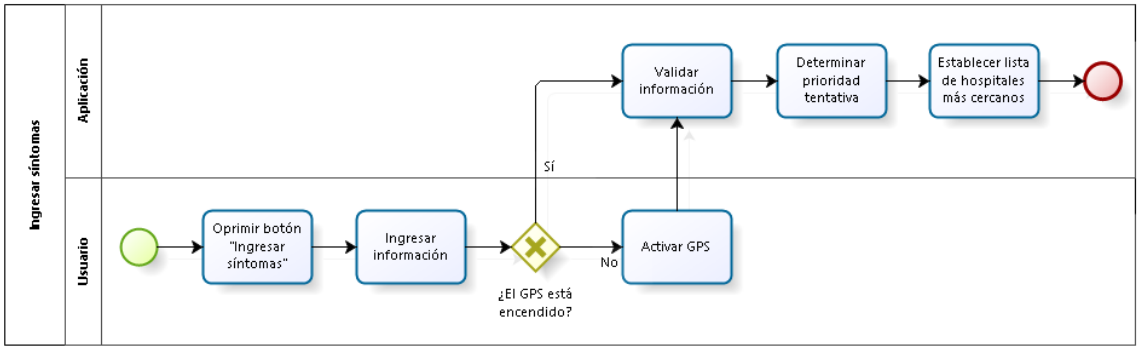


Ilustración 10: Procesos del sistema, ingresar síntomas

Estructura del Sistema

Esta sección consta de la información relevante para que el desarrollador comprenda los detalles de la estructura de los componentes de software del sistema. Provee una guía de las

cuales son las vistas, los controladores y la lógica de negocio que se debe utilizar en cada uno de los respectivos casos de uso.

Casos de uso – diseño general

Un caso de uso que requiere de lo siguiente para ser implementado:

- Paquete Vista_CelTAppi
  - Actividad CasoUsoPantalla, es una vista en Android de lo que se debe mostrar por pantalla cuando el usuario desea crear su sesión.
- Paquete Controlador_ CelTAppi
  - View CUcontrolador_Movil, es un controlador el cual se encarga de llevar a cabo las acciones que la pantalla desea realizar.
- Paquete Integración TAppi
  - Proxy_Celular, consta de un proxy el cual consume recursos del servidor con el fin de tener acceso a la lógica de negocio.
- Paquete Servicios_CelTAppi
  - Web Service ServicioWebTriage, es el servicio web que provee información del servidor al celular.
- Paquete Negocio_TAppi
  - Fachada CUlogicanegocioFacade, es quien contiene la fachada en la cual se debe implementar toda la lógica de negocio del servidor con tal de que el software lleve a cabo las acciones que le corresponden.
- Paquete Entidades, corresponde a las entidades que se van a utilizar con el fin de que el servidor pueda utilizar datos de persistencia. Estas se encuentran almacenadas en Derby, para mayor información de la base de datos consultar el modelo de dominio en el SRS.
- GPS, es el dispositivo que el celular tiene con el fin de proporcionar la ubicación del usuario a la aplicación. Este sólo se evidencia en ciertos casos de uso; por lo tanto, únicamente se debe tener en cuenta su interacción con el controlador en esos.

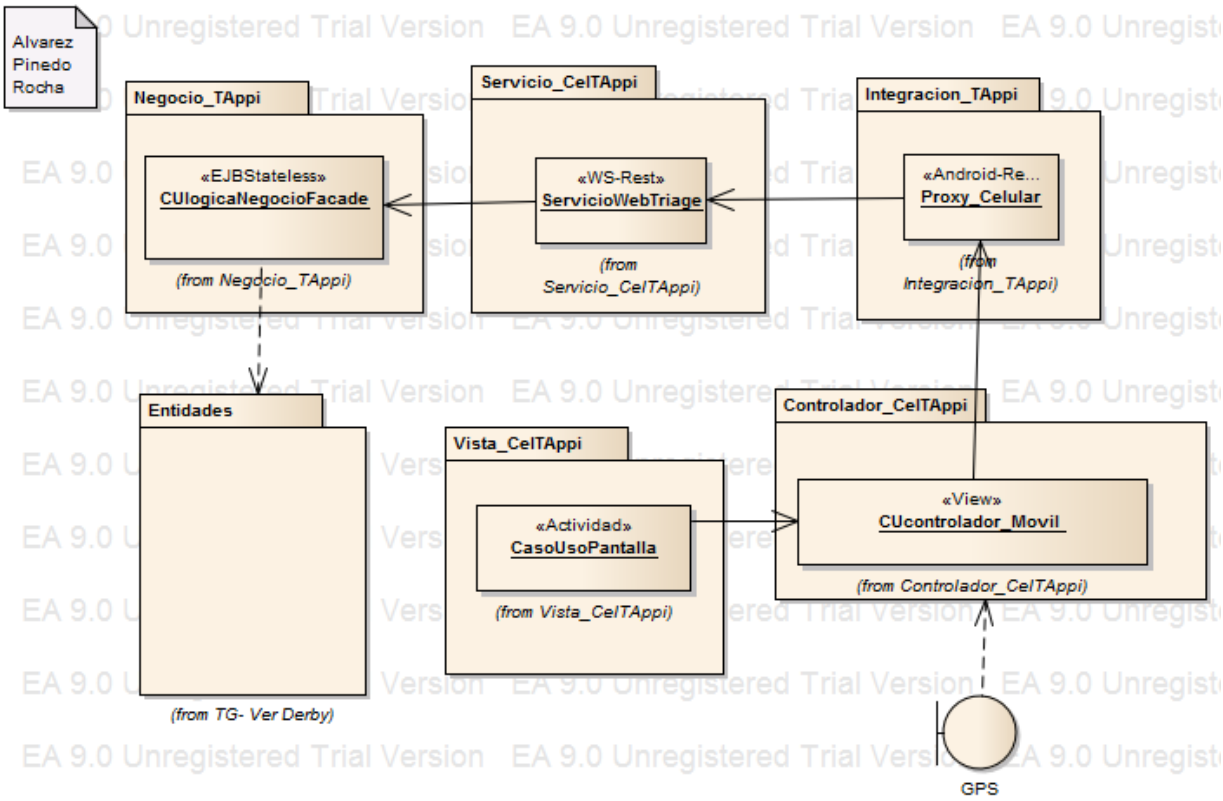


Ilustración 11: TAppi general paquetes

Comportamiento del Sistema - Diseño general

El propósito de esta sección es proveer una guía general de la secuencia que se debe llevar a cabo en los diferentes componentes del software con el fin de que se ejecute de manera efectiva, concreta y correcta.

La ilustración a continuación muestra cómo se relacionan los elementos que se encuentran en el celular con la lógica de negocio. Primero hay una actividad, la cual posteriormente se conecta con una vista (controlador), para conectarse por medio de un proxy y un web service a la lógica de negocio. Cabe resaltar que este es el flujo general de la información de la aplicación, teniendo en cuenta que en cada caso de uso cambian las entradas y las salidas.

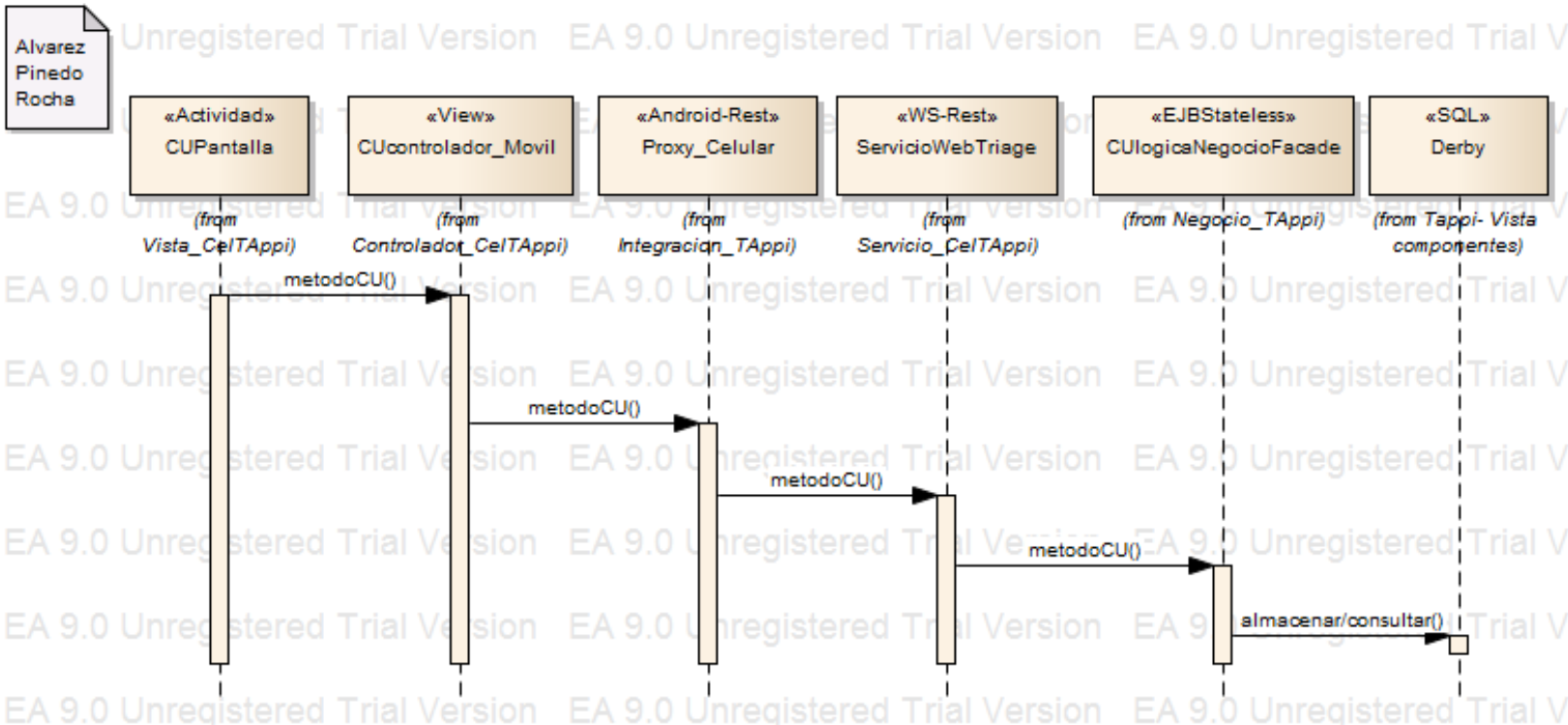


Ilustración 12: Caso de uso general TAppi secuencia

Si la acción a desarrollar fuera calcular valor tentativo Triage, el cual hace parte del caso de uso Ingresar Síntomas, se debió tener en cuenta lo siguiente:

- Entradas: los síntomas que el usuario ingresa en la pantalla del celular. Dichos síntomas son extraídos de la base de datos, luego se exponen en el celular, el usuario posteriormente elige los que desea, esta información es enviada a través del proxy y del web service a la lógica de negocio, por último, es almacenada en la base de datos utilizando la clase síntomasxhistoria.
- Salidas: nivel tentativo Triage. Dicho nivel es calculado por la lógica de negocio, utilizando las entradas del usuario y los pasos dispuestos por TAppi.

Persistencia, Modelo de dominio

En esta sección se ilustra cómo se van a persistir los datos de TAppi en la base de datos. Cabe resaltar que si se desea mayor detalle referirse al documento SRS en el cual se explica cada una de las tablas con sus respectivos atributos en la sección de Modelo de Dominio.

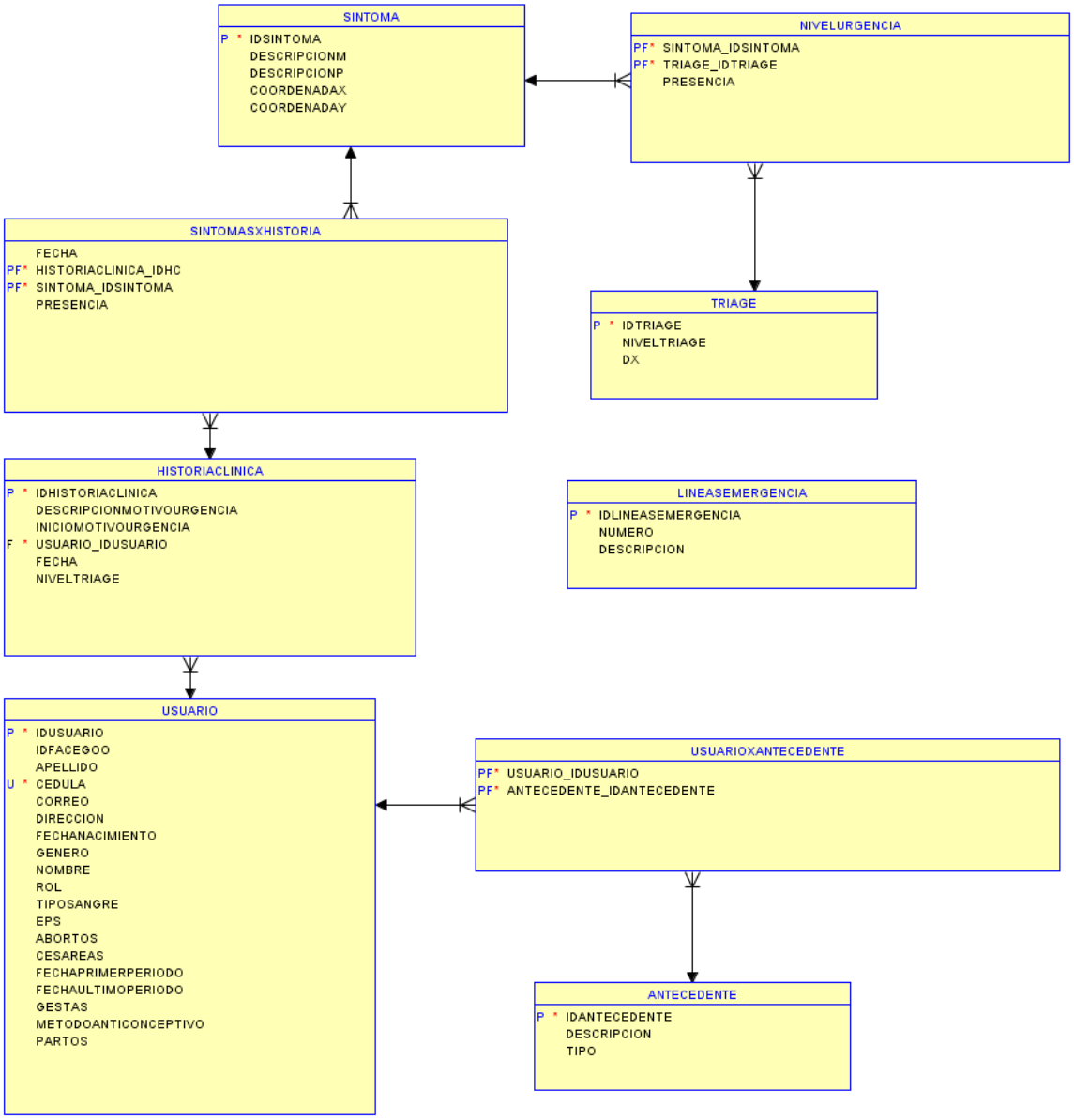


Ilustración 13: Modelo de dominio

V – DESARROLLO DE LA SOLUCIÓN

*"Todas las piezas deben unirse sin ser forzadas. Debe recordar que los componentes que está re-ensamblando fueron desmontados por usted, por lo que si no puede unirlos debe existir una razón. Pero, sobre todo, no use un martillo"***Manual de mantenimiento de IBM, 1925**

1. Construcción del sistema basado en reglas

De acuerdo con lo planteado en la Propuesta de Trabajo de grado, el proceso mediante el cual se planeaba desarrollar el proceso de priorización del Triage era un árbol de toma de decisiones. Esta decisión se vio entorpecida cuando los síntomas descritos en el documento del Ministerio de salud no permitían tomar decisiones inclusivas. Como nuevo modelo de solución a esta problemática, se utilizó un sistema basado en reglas.

Las reglas deterministas constituyen una sencilla metodología que contiene las variables y el conjunto de reglas que definen el problema y el motor de inferencia que obtiene las conclusiones aplicando la lógica clásica a estas reglas. Cada una de estas reglas contiene una proposición lógica que relaciona dos o más objetos e incluye una premisa y una conclusión. Una regla se escribe como "si premisa, entonces conclusión" y estas están conectada mediante operadores lógicos *y,o, no.* Tómese como ejemplo el uso de una tarjeta para extraer fondos de una cuenta bancaria [38]:

Objeto	Conjunto de posibles valores
Tarjeta	{verificada, no verificada}
Fecha	{expirada, no expirada}
NIP	{correcto, incorrecto}
Intentos	{excedidos, no excedidos}
Balance	{suficiente, insuficiente}
Límite	{excedido, no excedido}
Pago	{autorizado, no autorizado}

Ilustración 14: Objetos y posibles valores para el ejemplo del cajero automático

El conjunto de reglas posibles para este caso es:

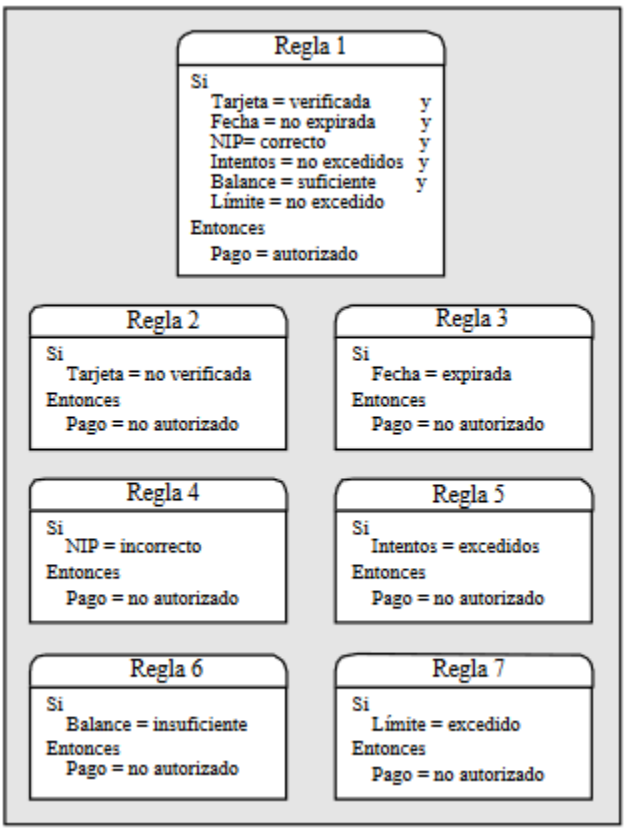


Ilustración 15: Ejemplo de reglas para sacar dinero de un cajero automático

Para obtener las conclusiones se utilizan diferentes tipos de estas reglas: *modus ponens* y *modus tollens.* En *modus ponens* concluye que, si A es cierto, entonces B es cierto y se sabe que además A es cierto. En *modus tollens* se utiliza *y* si es falsa; por lo tanto, si A es cierto, entonces B es cierto, pero se sabe que B es falso. Existen también estrategias de inferencia que ayudan a deducir las conclusiones simples y compuestas.

El encadenamiento de reglas es utilizado principalmente cuando las premisas de ciertas reglas coinciden con las conclusiones de otras. La siguiente figura muestra un ejemplo de seis reglas que relacionan trece objetos (de la A a la M):

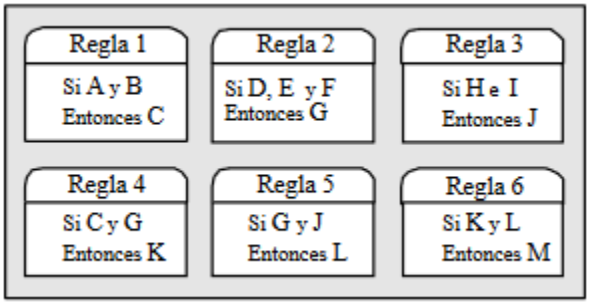


Ilustración 16: Ejemplo de un conjunto de seis reglas relacionando trece objetos

Nótese que las conclusiones de las Reglas 1 y 2 (objetos C y G) son las premisas de la Regla 4.

El encadenamiento de reglas orientado a un objetivo requiere que el usuario seleccione una variable o un nodo objetivo y el algoritmo navega a través de las reglas buscando una conclusión. Si no se obtiene ninguna conclusión, el algoritmo fuerza preguntar al usuario en busca de nueva información sobre los objetos relevantes.

"Las estrategias de encadenamiento se utilizan en problemas en los que los hechos (por ejemplo, síntomas) se dan por conocidos y se buscan algunas conclusiones (por ejemplo, enfermedades). Por el contrario, las estrategias de encadenamiento orientadas a un objetivo se utilizan en problemas en los que se dan algunos objetivos (enfermedades) y se buscan los hechos (síntomas) para que estas sean posibles." [38]

Tómese como ejemplo el siguiente fragmento del proceso de priorización del Triage según el Ministerio de salud de Colombia:

SIGNOS Y SÍNTOMAS	PRIORIDAD I-Reanimación
ABDOMINALES Y GASTROINTESTINALES	Trauma abdominal cerrado o penetrante con dolor severo, sangrado y signos de *shock* severo. Enterorragia masiva con

Ilustración 17: Signos y síntomas abdominales y gastrointestinales

Para construir el sistema basado en reglas, los síntomas deben escribirse de la siguiente manera:

- Trauma abdominal cerrado
- Trauma abdominal penetrante

- Dolor severo
- Sangrado
- Signos de shock

Ahora, en el caso de utilizar un árbol de toma de decisiones, los síntomas estarían organizados de la siguiente manera:

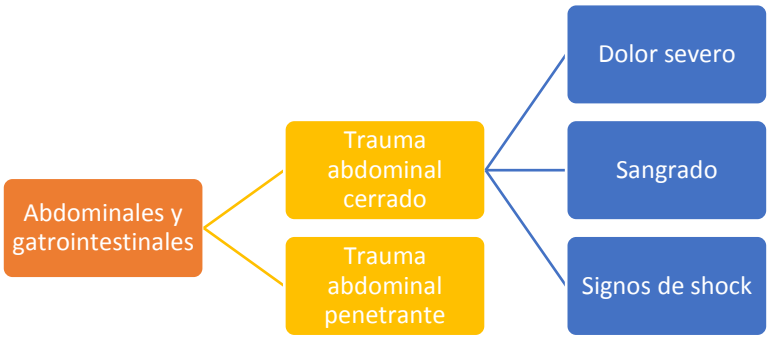


Ilustración 18: Árbol toma de decisión síntomas gastrointestinales y abdominales

Si el usuario quisiera marcar dolor severo y sangrado, no podrá marcar sangrado ya que la distribución de síntomas no permite realizar un cambio dentro de los niveles. Otro de los problemas encontrados es que los síntomas dolor severo, sangrado y signos de shock tendrían que repetirse para el síntoma trauma abdominal penetrante, causando un uso de memoria innecesario.

Con el uso de un sistema basado en reglas compuesto por reglas que contienen premisas y hechos también llamados conclusiones, fue posible generar un conjunto de expresiones lógicas entre las palabras claves, en este caso los síntomas. Adicionalmente, se requirió del encadenamiento de estas premisas para cada uno de los niveles de priorización del Triage. Siguiendo con el ejemplo de los síntomas abdominales y gastrointestinales, los síntomas quedaron agrupados de la siguiente manera:

Triage1: (trauma abdominal cerrado ∩ dolor severo ∩ sangrado ∩ signos de shock) U (trauma abdominal cerrado ∩ dolor severo ∩ sangrado ∩ signos de shock)

Modelo de conceptos

A continuación, se muestra la lógica de negocio del sistema basado en reglas para los síntomas descritos en el documento del Ministerio de Salud.

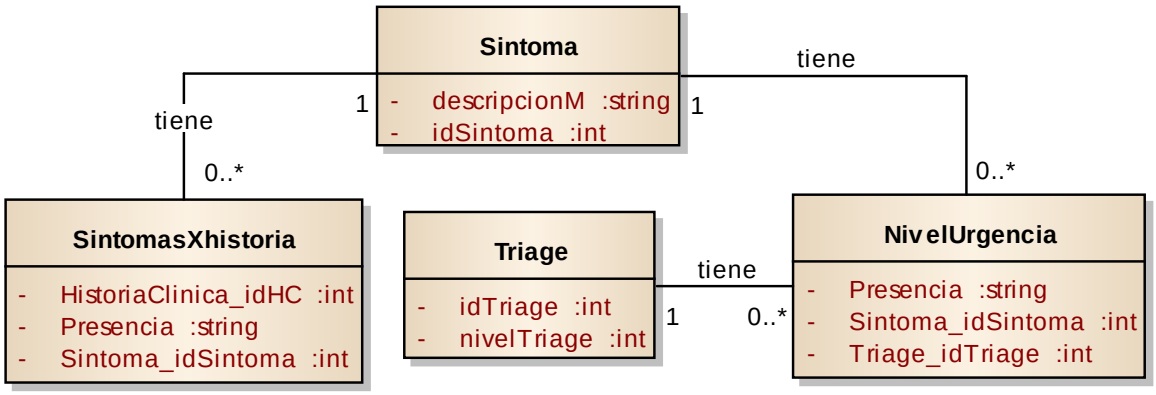


Ilustración 19: Lógica de negocio del sistema basado en reglas

Donde:

- Síntoma: contiene la lista de síntomas posibles que tiene la aplicación
- Triage: contiene la información que indica el id de los niveles del Triage
- NivelUrgencia: contiene la lista de síntomas, a qué diagnóstico pertenecen para asignar la priorización al conjunto
- SíntomaxHistoria: contiene los síntomas que el paciente ingresa como su dolencia

Se muestra enseguida un ejemplo de las relaciones entre las entidades con su respectivo contenido. La entidad síntoma contiene: un identificador, síntoma en términos médicos

Síntoma	
Id	**Descripción**
1	Trauma abdominal cerrado
2	Trauma abdominal penetrante
3	Dolor severo

4	Sangrado
5	Signos de shock

Tabla 7: Ejemplo entidad Síntomas

La entidad Triage contiene: un identificador y el nivel de Triage asociado. Este identificador pertenece al conjunto de síntomas asociados a ese nivel de priorización. Recordemos que:

Triage1: (dolor abdominal cerrado ∩ dolor severo ∩ sangrado ∩ signos de shock) 100

(dolor abdominal cerrado ∩ dolor severo ∩ sangrado ∩ signos de shock) 101

Es decir, Triage 1:

(1 Λ 3 Λ 4Λ 5)100(2 Λ 3 Λ 4Λ 5)101

Triage	
Id	**Nivel Triage**
100	1
101	1

Tabla 8: Ejemplo entidad Triage

La entidad NivelUrgencia contiene: el identificador de los síntomas, el identificador del conjunto de los síntomas y si el síntoma tiene presencia.

Nivel Urgencia		
Id Síntoma	**Id Triage**	**Presencia**
1	100	Sí
3	100	Sí

4	100	Sí
5	100	Sí
2	101	Sí
3	101	Sí
4	101	Sí
5	101	Sí

Tabla 9: Ejemplo entidad Nivel urgencia

La entidad SíntomaxHistoria contiene: un identificador, el identificador de los síntomas que ha ingresado el paciente y si están o no presentes. Esta entidad permite almacenar las historias clínicas de los pacientes y los síntomas asociados a ellas.

Síntoma x Historia		
Id Historia	**Id Síntomas**	**Presencia**
1	1	Sí
1	3	Sí
1	4	Sí
1	5	Sí

Tabla 10: Ejemplo entidad Síntoma x Historia

Gracias a esta nueva propuesta el diagnóstico tentativo del Triage se puede llevar a cabo de manera más sencilla y rápida, debido a que las decisiones de los síntomas pueden ser inclusivas y se pueden repetir los diagnósticos.

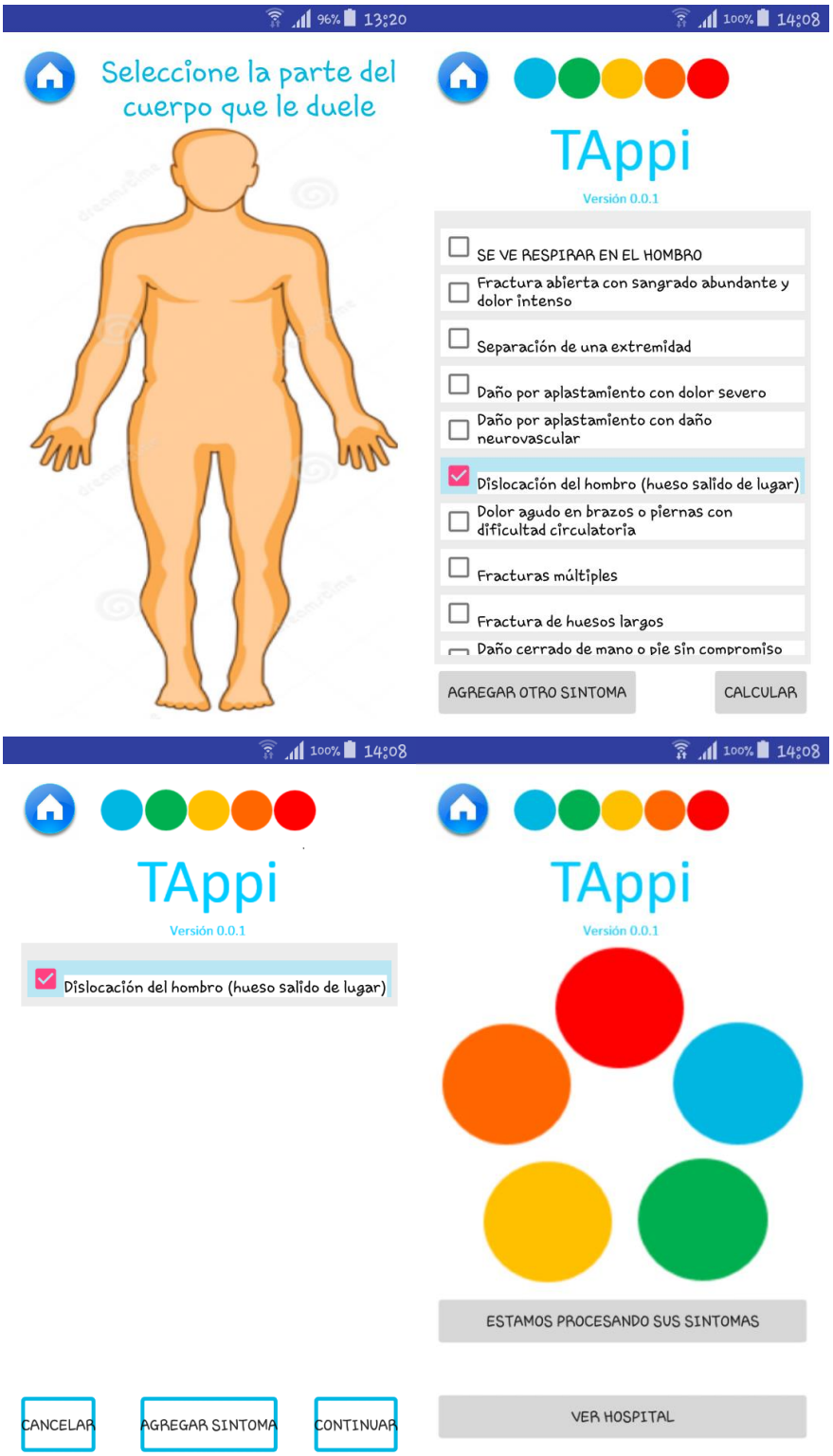
Para consultar el sistema basado en reglas de los síntomas del Triage ver Anexos TG - LogicaDelTriage, Consultas para la lógica de Triage.

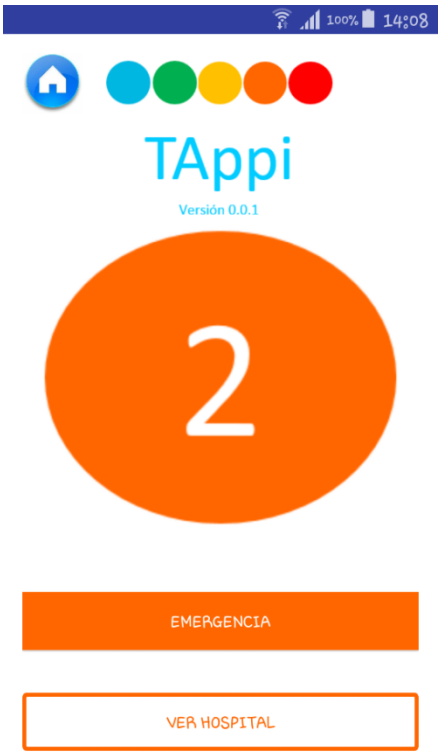
2. Interfaz de Usuario

A continuación, se muestra el diagrama de flujo de las interfaces de los usuarios. Las primeras etapas son el registro y el inicio de sesión.

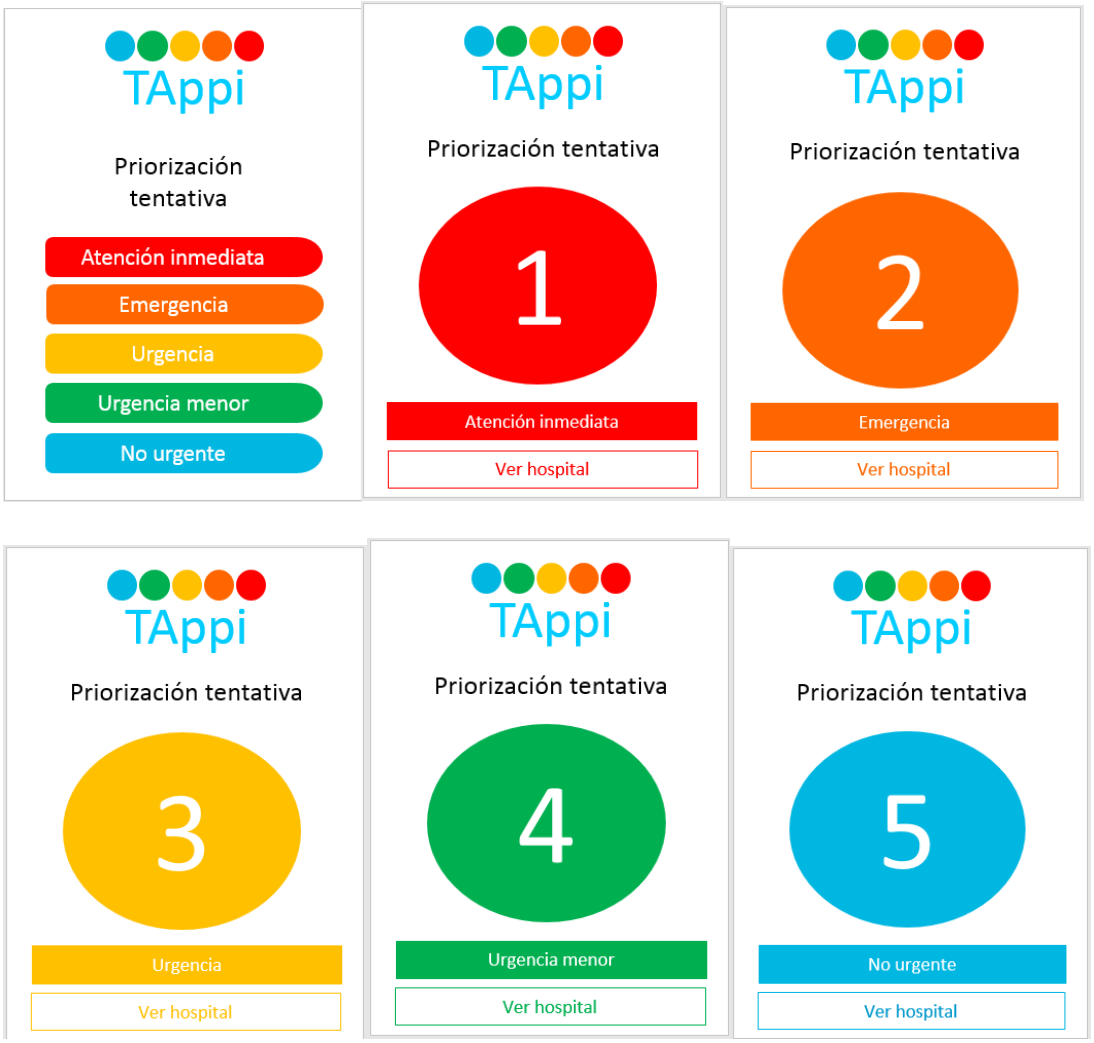
Ingresar síntomas

La siguiente imagen muestra la interfaz de usuario paciente para el ingreso de los síntomas.





Una vez que el usuario ingresa a la pantalla de *Ingresar Síntomas*se le muestra el dibujo del cuerpo humano para que el paciente señale el lugar del cuerpo en donde presenta los sintomas. De acuerdo con el lugar señalado, se mostrará una lista con los posibles síntomas que pueda presentar. Si desea agregar más partes del cuerpo en las que presenta molestias, el usuario oprime el botón *Agregar síntoma* que lo retorna a la pantalla del cuerpo humano. Una vez ingresados todos los síntomas, se le mostrará al usuario una priorización tentaiva del Triage. Las pantallas siguientes muestran la forma en la que se presentará esta priorización.



Números de emergencia

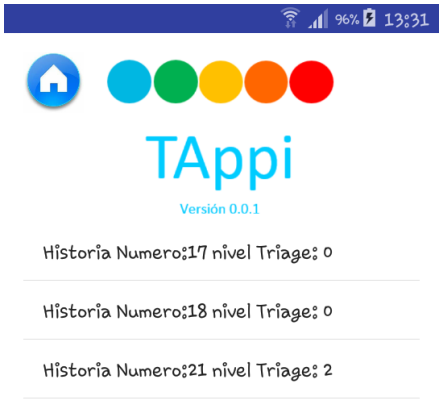
La interfaz de usuario correspondiente a *Números de emergencia*, presenta una lista de las líneas de emergencia nacionales a las que puede llamar.

Cuenta

La interfaz de usuario paciente correspondiente a *Cuenta*, permite al usuario editar los datos asociados a su cuenta, para los usuarios que marquen género femenino se mostrará adicionalmente una pantalla para ingresar sus datos ginecológicos. En la parte inferior los usuarios también pueden especificar si poseen conocimientos en medicina.

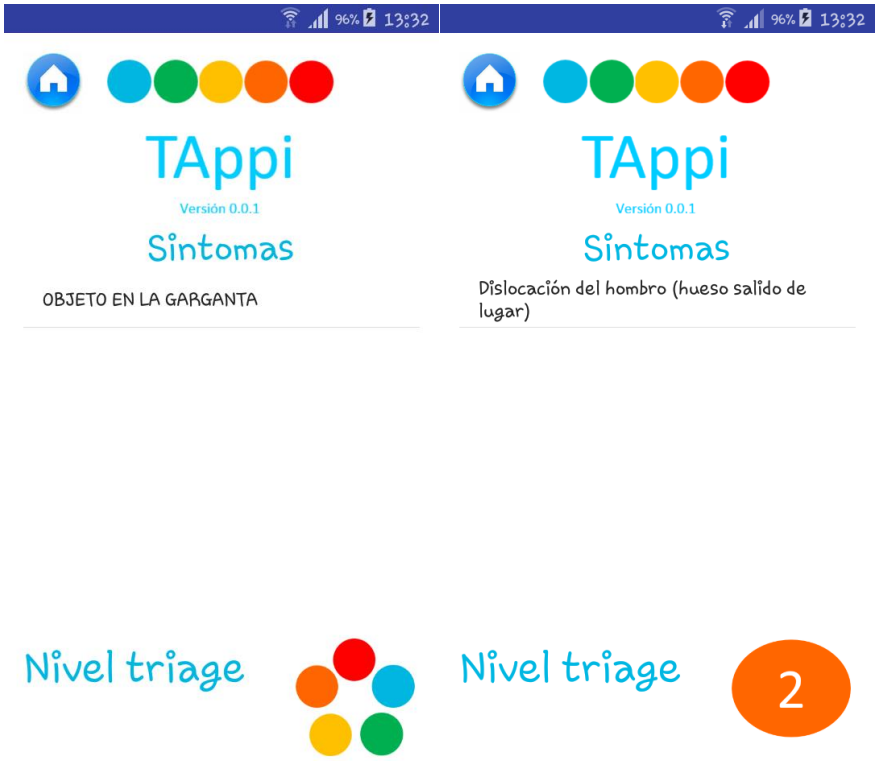
Historial

La interfaz de usuario correspondiente a *Historial*, presenta todas las historias clínicas del paciente. Cada una de estas historias muestra los síntomas ingresados y la priorización tentativa que sugirió la aplicación.



A continuación, se muestran dos tipos de historias; la primera (Historia 17 o Historia 18) muestra una priorización 0 que corresponde a visitar el centro de salud puesto que los síntomas marcados no

permiten determinar una priorización tentativa, y la segunda (Historia 21) presenta una priorización de nivel 2.



VI – RESULTADOS

Siguiendo con la metodología planteada, la última fase de desarrollo de este trabajo de grado es la fase de validación y verificación. Aunque esta no se encuentra especificada en la metodología RUP bajo el mismo nombre, se considera importante su realización ya que evalúa la usabilidad del prototipo llevado a cabo. Las siguientes actividades nos permitieron realizar las validaciones con ayuda del director de trabajo de grado, usuarios sin conocimientos avanzados en medicina y usuarios con conocimientos de medicina:

- Refinar diseño y realizar ajustes necesarios en el documento SDD
- Refinar prototipo funcional
- Validar el software con pruebas de aceptación por parte del director de trabajo de grado
- Pruebas de software (ver anexo Servidor):
    - Pruebas unitarias
    - Pruebas de sistema

Cabe resaltar que el ajuste de errores y la evaluación del progreso fueron actividades necesarias para que estos resultados fueran positivos. Las pruebas de validación por parte del ministerio de salud se llevaron a cabo con la revisión de cada uno de los síntomas presentados en la Guía de Manejo de urgencias tomo III.

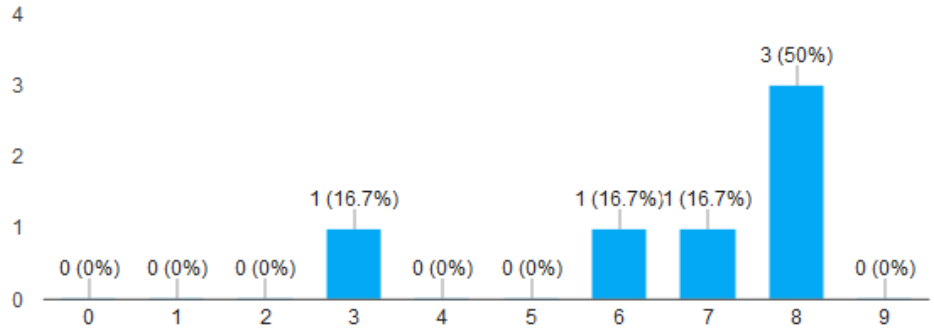
Desarrollo encuestas a usuarios

Para el desarrollo de las encuestas a los usuarios que utilizaron la aplicación, se llevó a cabo una serie de preguntas iguales tanto para los usuarios médicos (con conocimiento de medicina)y los usuarios sin conocimiento de términos médicos. Los usuarios tipo médico presentan un perfil de estudiantes y profesionales en el área a quienes encargamos la tarea de evaluar la funcionalidad de la aplicación en términos de los síntomas presentados y el nivel de Triage tentativo dado por la aplicación. Para el otro tipo de usuarios se encomendó evaluar la presentación de la aplicación y los atributos de calidad descritos en el documento SRS.

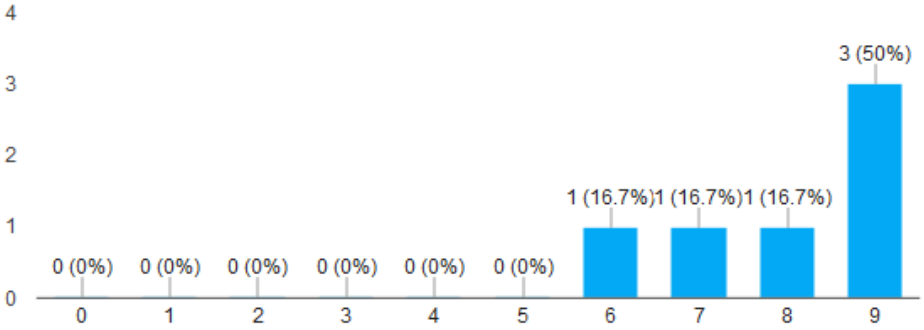
Las gráficas presentadas muestran el porcentaje de respuestas dada por los usuarios.

Reacción general del software

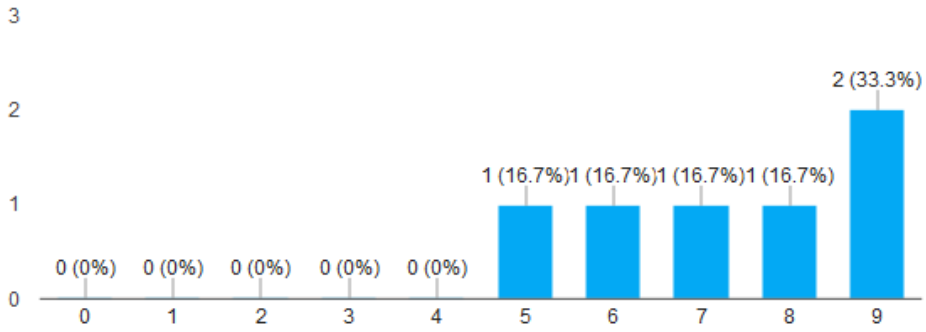
- Concepto general:0 (terrible), 9 (maravilloso)



- Usabilidad: 0 (frustrante), 9 (satisfactorio)

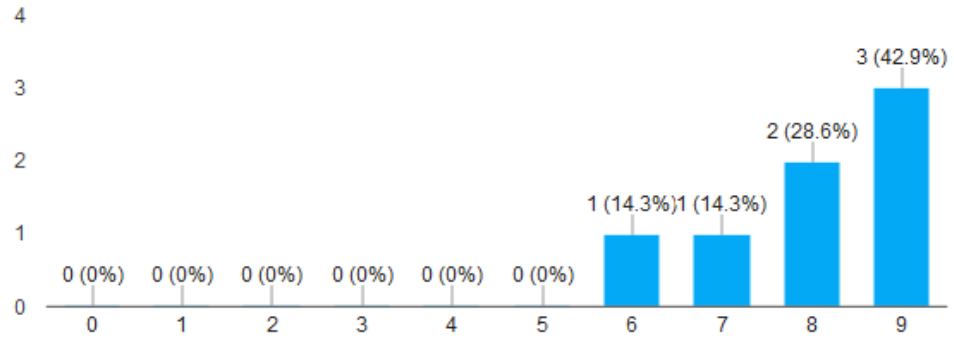


- Usabilidad: 0 (difícil),9 (fácil)

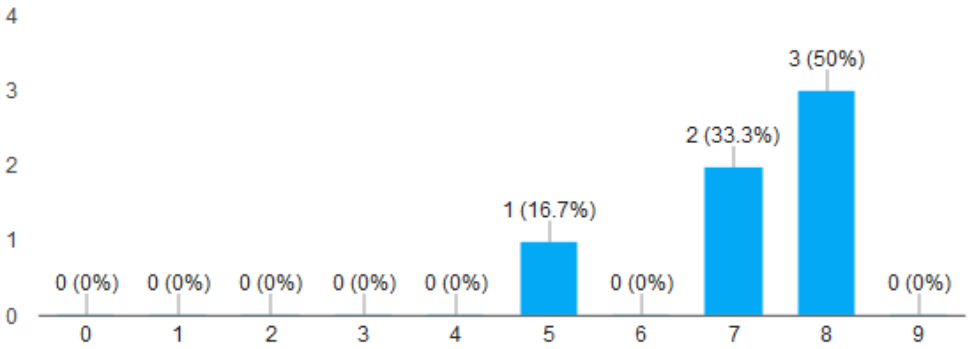


Pantalla

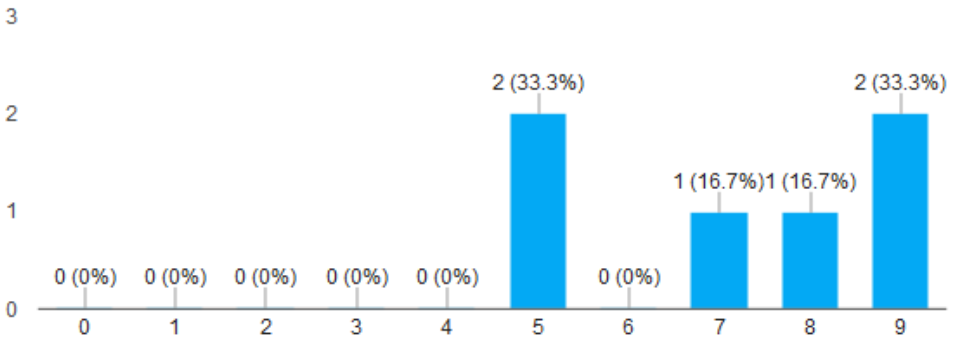
- Lectura de los caracteres en la pantalla:0 (difícil), 9 (fácil)



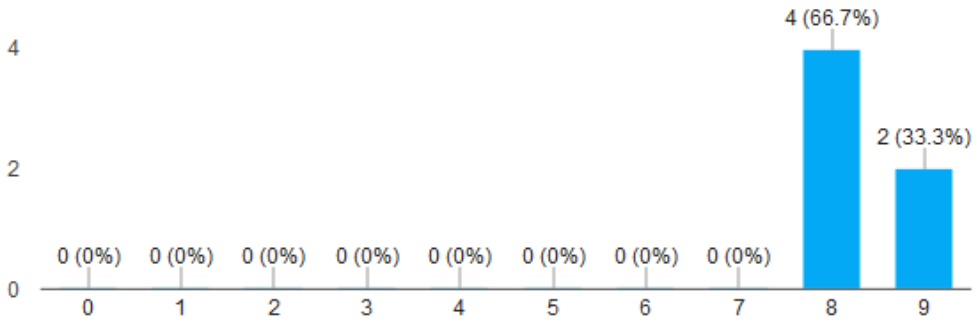
- Subrayado de las acciones: 0 (nulo), 9 (mucho)



- Organización de la información: 0 (confuso), 9 (muy claro)

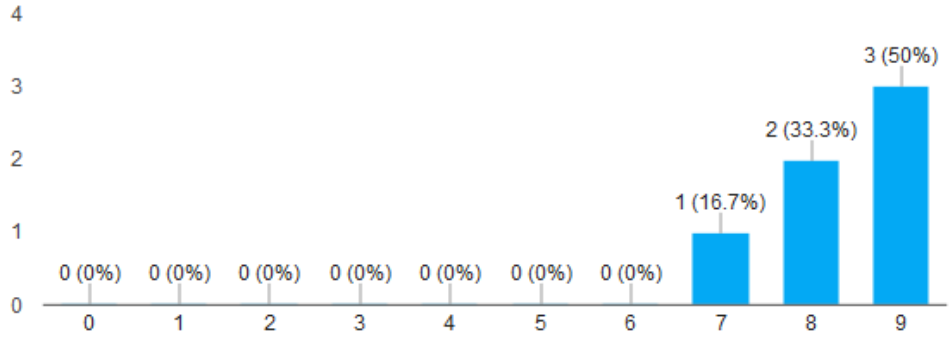


- Secuencia de pantallas: 0 (confuso), 9 (muy claro)

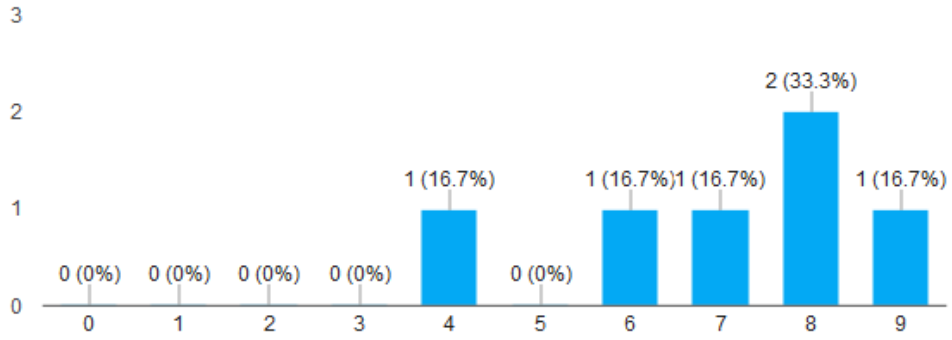


Terminología de la información

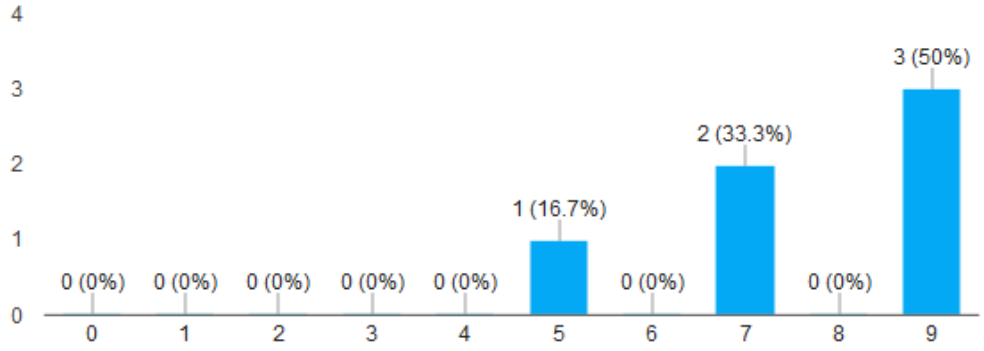
- Términos utilizados a través del sistema: 0 (inconsistentes), 9 (consistentes)



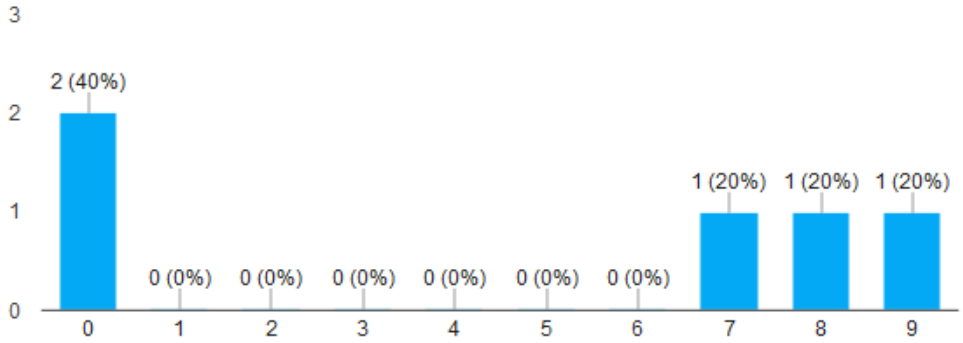
- Términos relacionados con la tarea:0 (nunca), 9 (siempre)



- Entrada de caracteres:0 (confusa), 9 (muy clara)

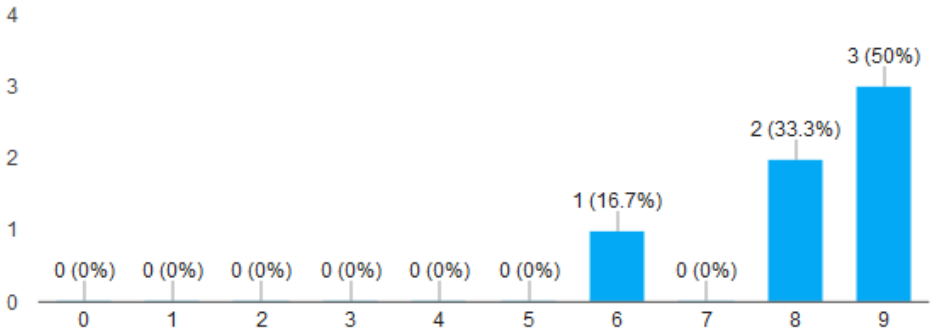


- Mensaje de error:0 (inútiles), 9 (útiles). Los usuarios que marcaron cero en mensaje de error, no recibieron este tipo de mensajes durante el uso de la aplicación.

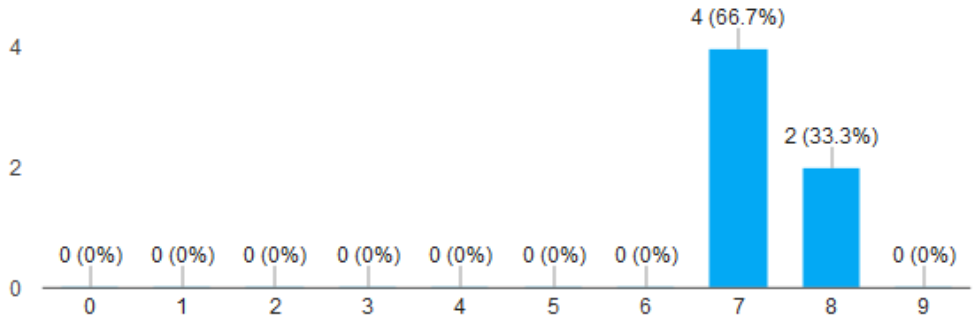


Aprendizaje

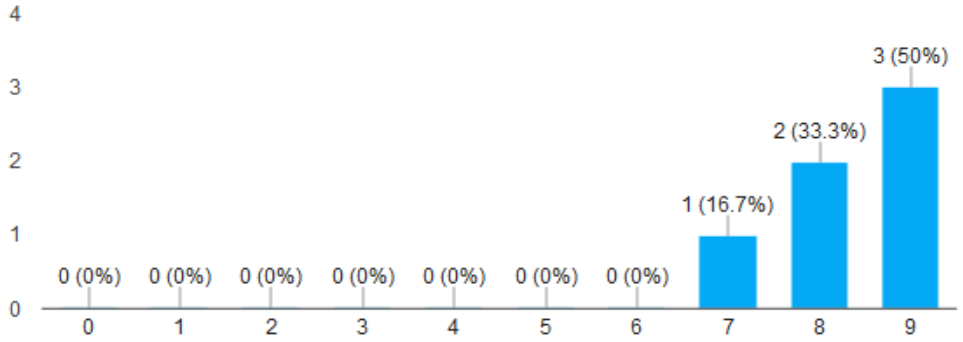
- Aprender a usar el sistema:0 (difícil), 9 (fácil)



- Recordar nombres de los comandos: 0 (difícil), 9 (fácil)

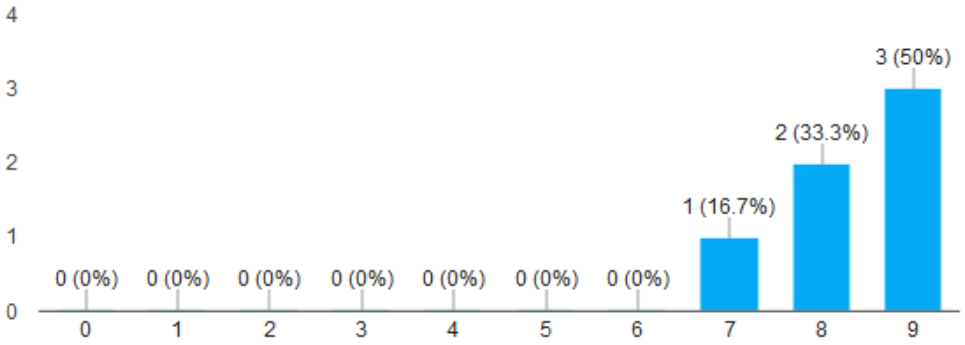


- Realizar tareas simples con el sistema: 0 (difícil), 9 (fácil)

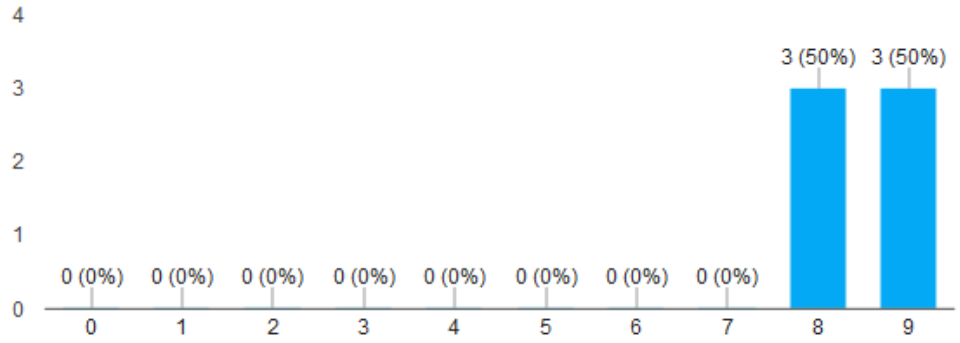


Capacidades del sistema

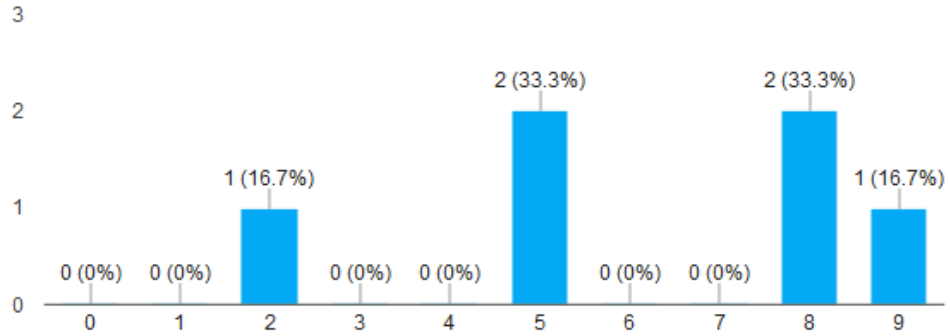
- Velocidad de respuesta: 0 (muy lento), 9 (muy rápido)



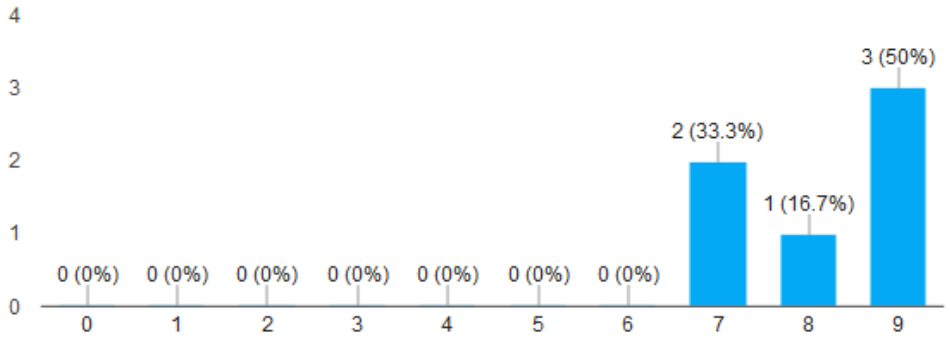
- Confiabilidad del sistema: 0 (poco confiable), 9 (muy confiable)



- Corregir errores en la interacción:0 (difícil), 9 (fácil)



- Diseñada para diferentes usuarios: 0 (nunca), 9 (siempre)



Comentarios

A continuación, se muestran los comentarios de las pruebas realizadas con usuario con conocimiento de medicina:

- No tiene botón atrás
- No resalta la parte del cuerpo elegida
- Faltan muchos síntomas.

- No llama a las líneas de emergencia
- Necesita un botón de pánico

Gracias a estos comentarios podemos saber que lo importante es completar la lista de síntomas y especificar de mejor manera la parte del cuerpo afectada. Esto se debe a que siendo personal de salud es fundamental para ellos establecer de maneras exactas las dolencias del paciente.

A continuación, se muestran los comentarios de las pruebas realizadas con usuario sin conocimiento de medicina:

- Letra pequeña listas grandes
- No es claro el proceso cuando pone a llamar a un médico y no da el nivel de Triage, puede ser más claro el porqué
- Mucha gente no tiene celulares costosos

Por otro lado, podemos ver que los usuarios sin conocimientos de medicina se enfocaron en la presentación de la aplicación dejando en segundo plano el contenido de la misma.
Para ambos tipos de usuario los aspectos a resaltar fueron:

- Facilidad de uso
- Llamativa e interactiva.
- A futuro bien diseñada queda muy bien
- Rápido.
- En un futuro se podría no usar la enfermera
- Todo depende de la sinceridad del paciente.
- Práctico y eficiente
- Interfaz amigable
- Rapidez para la atención, evita largas colas y esperas

Es importante resaltar que los usuarios con conocimiento de medicina solicitaron que se estableciera una sección de *Términos y condiciones* en la que se exponga que la aplicación sólo es una guía en el proceso del Triage. Situación debida a que los pacientes deben ser evaluados dentro de la sala de emergencias para verificar signos vitales tales como la

presión arterial y la frecuencia respiratoria ya que son factores decisivos dentro del proceso de priorización.

VII – CONCLUSIONES

1. Análisis de Impacto del Desarrollo

Con lo obtenido en la elaboración del trabajo de grado se puede concluir que desde el punto de vista disciplinar, se ha dejado la arquitectura de una aplicación móvil que constituye un nuevo formato que permite el fácil diligenciamiento de los síntomas del Triage. Se deja también una base de datos que contiene todos los síntomas contemplados en La Guía de Manejo de urgencias Tomo III del Ministerio de Salud, así mismo todo el sistema basado en reglas con las posibles combinaciones de síntomas para cada uno de los niveles del Triage.

Desde el punto de vista social, la elaboración de TAppi: Triage Application es una herramienta para pacientes, instituciones prestadoras de servicio de salud y personal sanitario, que brinda a los usuarios la oportunidad de ingresar los datos de manera inteligente antes de asistir a un centro de salud para agilizar el proceso del Triage. De la misma manera, y buscando solucionar los cuellos de botella presentes en las salas de urgencia, el desarrollo de la aplicación traerá ventajas para el personal de salud quienes no requerirán de solicitar datos básicos del paciente de manera iterativa y prestar la debida atención a pacientes con un nivel de Triage prioritario.

2. Conclusiones y Trabajo Futuro

Siguiendo con lo descrito anteriormente se puede concluir que:

- El objetivo específico, especificar los requerimientos de la aplicación, se puede ver evidenciado dentro del documento SRS (Software RequieremenSpecification)
- El objetivo específico, diseñar la arquitectura teniendo en cuenta los diferentes roles de usuarios, se puede ver evidenciado dentro del documento SDD (Software Design Document)
- El objetivo específico, desarrollar el prototipo funcional de la aplicación descrita en los objetivos anteriores, se cumplió con la implementación de la aplicación TAppi desarrollada en Android Studio.
- Los objetivos específicos, validar el software con pruebas de aceptación por parte del director de trabajo de grado y documentar los manuales de la aplicación, se pueden ver en los anexos de documentación del código y del servidor.

Otras conclusiones que se pueden obtener del trabajo de grado son:

- Durante el proceso de diseño de la lógica de negocio y presentación fue necesaria la re elaboración y reestructuración de los componentes asociados debido a la falta de experiencia con la integración de las capas del modelo arquitectural.
- La exposición de los servicios del servidor mediante un Web Service (en XML y JSON) permitió que se desarrollara una aplicación móvil que consumiera los servicios implementados en un ambiente diferente. Esto también permite que diferentes interfaces puedan conectarse al servidor gracias a que su desarrollo fue para ser consumido por diferentes entornos.
- Para desarrollar un glosario de términos para usuarios sin conocimiento de medicina fue necesaria la ayuda de estudiantes de medicina quienes desinteresadamente nos facilitaron la explicación de los términos más complicados de definir.
- Con el desarrollo de esta aplicación, Colombia ahora pertenece al grupo de países que cuentan con una aplicación móvil que agiliza el proceso del Triage.

Como trabajo a futuro se propone:

- Extender la posibilidad de conectarse con diferentes usuarios desde la cuenta asociada de Facebook.
- Crear un usuario administrador que se encargue del manejo de las cuentas. Este usuario podría eliminar cuentas que causen problemas por peticiones al servidor o que ingrese datos ficticios que afecten las funcionalidades de la aplicación.
- Aumentar la funcionalidad "Agregar Familiares", tener diferentes perfiles de usuario. Como ejemplo se tiene personas no nativas digitales (niños o adulto mayor); esta funcionalidad permitiría que el usuario dueño del *smartphone* cree historias clínicas con los nombres de sus familiares.
- Derivado de la funcionalidad anterior, se propone también que el usuario pueda editar o eliminar la información de sus familiares.
- Aumentar la funcionalidad para un usuario *Médico,* que estando en el centro de salud pueda consultar los síntomas ingresados y el nivel de priorización determinado.

Estas funcionalidades se pueden ver especificadas en detalle en el Anexo de trabajo a futuro.

- Desarrollar la funcionalidad *Ayuda*, que contendrá diferentes manuales incluidos dentro de la aplicación; así mismo, la sección de preguntas frecuentes y un contacto para aquellos usuarios que presenten problemas con la aplicación.
- Desarrollar la funcionalidad *ver hospital y ver farmacia*, funcionalidades no desarrolladas debido a restricciones de tiempo.

VIII- REFERENCIAS Y BIBLIOGRAFÍA

[1] "OMS | ¿Qué es un sistema de salud?" [Online]. Available: http://www.who.int/features/qa/28/es/. [Accessed: 01-Apr-2016].

[2] "UNIDAD DE URGENCIAS HOSPITALARIA - UUH.pdf." [Online]. Available: http://www.msssi.gob.es/organizacion/sns/planCalidadSNS/docs/UUH.pdf. [Accessed: 01-Apr-2016].

[3] R. Silvariño, V. Acevedo, M. Moyano, E. Méndez, E. Paolillo, and J. Álvarez, "Experiencia de triaje estructurado en el departamento de urgencia," *Rev. Médica Urug.*, vol. 27, no. 2, pp. 88–93, 2011.

[4] "Método START de triage | Urgencias, Emergencias, Catástrofes y Rescate." [Online]. Available: http://www.e-mergencia.com/threads/metodo-start-de-triage.11654/. [Accessed: 01-Apr-2016].

[5] "Triage | Products | TSG Associates - Home of SMART MCI." [Online]. Available: http://www.smartmci.com/products/triage/smart_tag.php. [Accessed: 01-Apr-2016].

[6] J. Resendiz, M. Montiel, and R. Limona, "Triage en el servicio de urgencias," *Med. Interna México*, vol. 22, no. 4, 2006.

[7] N. de Argila Fernández-Durán, "Evaluación del impacto tras la implantación del Triage de adulto gestionado por enfermería en Urgencias."

[8] Ministerio de Salud Colombia, "Información para Instituciones Prestadoras de Salud (IPS)," *minSalud*, 03-Feb-2016. [Online]. Available: https://www.minsalud.gov.co/salud/PServicios/Paginas/informacion-de-interes.aspx. [Accessed: 02-Mar-2016].

[9] C. E. E. Tiempo, "Polémica por paciente que murió mientras esperaba ser atendido - Bogotá," *El Tiempo*. [Online]. Available: http://www.eltiempo.com/bogota/muere-hombre-al-no-ser-atendido-en-urgencias/16386779. [Accessed: 18-Mar-2016].

[10] M. A. L. Betancur, M. L. G. Henao, M. C. M. Ramírez, and C. F. Pulido, "Dificultades para la atención en los servicios de urgencias: la espera inhumana," *Invest Educ Enferm*, vol. 28, no. 1, pp. 64–72, 2010.

[11] G. González, M. L. Valencia, N. A. Agudelo, L. Acevedo, and I. C. Vallejo, "Morbilidad sentida de las urgencias médicas y la utilización de los servicios de salud

en Medellín, Colombia, 2005-2006," *Biomédica*, vol. 27, no. 2, pp. 180–189, 2007.

[12] "DESGASTE EMOCIONAL Y ESTRATEGIAS DE AFRONTAMIENTO EN PERSONAL DE ENFERMERÍA DE URGENCIAS." [Online]. Available: http://dspace.sheol.uniovi.es/dspace/bitstream/10651/31157/6/vilari%c3%b1o.pdf. [Accessed: 21-Feb-2016].

[13] K. A. Holtermann and A. G. R. González, *Desarrollo de sistemas de servicios de emergencias médicas: experiencia de los Estados Unidos de América para países en desarrollo, octubre 2003, Washington*. Pan American Health Org, 2003.

[14] "Técnicas de resolución de problemas Triage - Salud - amhasefer.com." [Online]. Available: http://www.amhasefer.com/q8kZWPjR/. [Accessed: 01-Apr-2016].

[15] J. C. Riquelme, R. Ruiz, and K. Gilbert, "Mineria de datos: Conceptos y tendencias," *Rev. Iberoam. Intel. Artif.*, vol. 10, no. 29, pp. 11–18, 2006.

[16] W. Michalowski, R. Slowinski, and S. Wilk, "Mobile emergency triage support system," in *AAAI*, 2004, pp. 1018–1019.

[17] "iTriage - Health, Doctor, Symptoms and Healthcare search on the App Store," *App Store*. [Online]. Available: https://itunes.apple.com/us/app/itriage-health-doctor-symptoms/id304696939?mt=8. [Accessed: 14-Feb-2016].

[18] F. García, J. Portillo, J. Romo, and M. Benito, "Nativos digitales y modelos de aprendizaje.," in *SPDECE*, 2007.

[19] "Quick Triage App - Aplicaciones Android en Google Play." [Online]. Available: https://play.google.com/store/apps/details?id=com.jxt.QuickTriage. [Accessed: 17-Feb-2016].

[20] M. L. Valencia-Sierra, G. González-Echeverri, N. A. Agudelo-Vanegas, L. Acevedo-Arenas, and I. C. Vallejo-Zapata, "Acceso a los Servicios de Urgencias en Medellín, 2006," *Rev. Salud Pública*, vol. 9, no. 4, pp. 529–540, 2007.

[21] Ministerio de Salud Colombia, "Guías para manejo de urgencias -Tomo III.pdf," 2009. [Online]. Available: https://www.minsalud.gov.co/Documentos%20y%20Publicaciones/Gu%C3%ADas%20para%20manejo%20de%20urgencias%20-Tomo%20III.pdf. [Accessed: 21-Feb-2016].

[22] K. Beck, *Extreme programming explained: embrace change*. addison-wesley professional, 2000.

[23] P. Kroll and P. Kruchten, *The rational unified process made easy: a practitioner's guide to the RUP*. Addison-Wesley Professional, 2003.

[24] D. Phillips, *The software project manager's handbook: principles that work at work*, vol. 3. John Wiley & Sons, 2004.

[25] H. Guang-yong, "Study and practice of import Scrum agile software development," in *Communication Software and Networks (ICCSN), 2011 IEEE 3rd International Conference on*, 2011, pp. 217–220.

[26] "Introducción a la ingeniería de requerimientos." [Online]. Available: http://www.fceia.unr.edu.ar/~mcristia/publicaciones/ingreq-a.pdf. [Accessed: 15-Apr-2016].

[27] P. Kruchten, *The rational unified process: an introduction*. Addison-Wesley Professional, 2004.

[28] R. P. TUW and I. Fraunhofer, "Software requirement specification."

[29] "Arquitectura de Software," *Software Guru*. [Online]. Available: http://sg.com.mx/revista/27/arquitectura-software#.VxBYd3qG_IX. [Accessed: 15-Apr-2016].

[30] Fernando Barraza A. MS.c., "Modelado y Diseño de Arquitectura de Software." [Online]. Available: http://cic.javerianacali.edu.co/wiki/lib/exe/fetch.php?media=materias:s2_conceptosdemodelado.pdf. [Accessed: 15-Apr-2016].

[31] "IBM Knowledge Center," 01-Jan-2013. [Online]. Available: http://www.ibm.com/support/knowledgecenter/SSWSR9_11.0.0/com.ibm.pim.dev.doc/pim_tsk_arc_definingusecases.html?lang=es. [Accessed: 15-Apr-2016].

[32] E. T. López, A. O. Ramon, E. M. Sarroca, and C. G. Seone, *Diseño de sistemas software en UML*. Univ. Politèc. de Catalunya, 2004.

[33] "Software Design Document - Example-SoftwareDesignDocument-LegalXMLUtility.pdf." [Online]. Available: https://www.oasis-open.org/committees/download.php/24846/Example-SoftwareDesignDocument-LegalXMLUtility.pdf. [Accessed: 21-May-2016].

[34] Carlos Blanco Bueno, "Ingeniería del Sofware II -Tema 01. Construcción y Pruebas de Sofware." [Online]. Available: http://ocw.unican.es/ensenanzas-tecnicas/ingenieria-del-software-ii/materiales/tema1-pruebasSistemasSoftware.pdf. [Accessed: 15-Apr-2016].

[35] "Coma," 11:47:32 UTC.

[36] "WHO | Pulse oximetry," *WHO*. [Online]. Available: http://www.who.int/patientsafety/safesurgery/pulse_oximetry/en/. [Accessed: 03-May-2016].

[37] "OMS | Qué es la diabetes," *WHO*. [Online]. Available: http://www.who.int/diabetes/action_online/basics/es/index1.html. [Accessed: 03-May-2016].

[38] "Reglas.pdf." [Online]. Available: http://personales.unican.es/gutierjm/cursos/expertos/Reglas.pdf. [Accessed: 09-Oct-2016].

[39] GeekMedico, "iTriage, simplemente WOW," *Geek + Medico*, 26-Nov-2012. .

[40] "Triage Territorial - Aplicaciones Android en Google Play." [Online]. Available: https://play.google.com/store/apps/details?id=peris.triageterritoriale. [Accessed: 17-Feb-2016].

[41] "START Free - Aplicaciones Android en Google Play." [Online]. Available: https://play.google.com/store/apps/details?id=com.bobmcd.START_Free. [Accessed: 17-Feb-2016].

[42] "CTAS - Triage - OFFICIAL - Aplicaciones Android en Google Play." [Online]. Available: https://play.google.com/store/apps/details?id=com.ctas. [Accessed: 17-Feb-2016].

[43] "Fast Triage App Lite - Aplicaciones Android en Google Play." [Online]. Available: https://play.google.com/store/apps/details?id=com.innomax.fasttriageapp_lite. [Accessed: 17-Feb-2016].

[44] "EMS ACLS Guide - Aplicaciones Android en Google Play." [Online]. Available: https://play.google.com/store/apps/details?id=com.informedpublishing.EMSALS. [Accessed: 17-Feb-2016].

[45] J. C. Alvarez Abad, "Diseño del sistema de información georeferencial y red de comunicaciones de salud para el cantón Cuenca," 2000.

[46] C. P. López, *Minería de datos: técnicas y herramientas*. Editorial Paraninfo, 2007.

[47] Hernández Esteban *et al.*, "GitHub-SnoutPoint-Networks: Proyecto de SnoutPoint, red social para mascotas," 2015. [Online]. Available: https://github.com/Mutisantos/SnoutPoint-Networks. [Accessed: 27-Jun-2016].

[48] G. Booch, J. Rumbaugh, I. Jacobson, J. S. Martínez, and J. J. G. Molina, *El lenguaje unificado de modelado*, vol. 1. Addison-Wesley, 1999.

[49] C. Larman, *UML y Patrones*. Pearson, 1999.

[50] "GitHub," *GitHub*. [Online]. Available: https://github.com. [Accessed: 15-Jun-2016].

[51] "Android Developers." [Online]. Available: http://developer.android.com/index.html. [Accessed: 08-Mar-2015].

[52] MySQL, "MySQL Workbench." [Online]. Available: https://www.mysql.com/products/workbench/. [Accessed: 22-Apr-2015].

[53] Ubuntu, "New Ubuntu OpenStack Fundamentals training courses." [Online]. Available: http://www.ubuntu.com/. [Accessed: 22-Apr-2015].

[54] IDC, "IDC: Smartphone OS Market Share," *www.idc.com*. [Online]. Available: http://www.idc.com/prodserv/smartphone-os-market-share.jsp. [Accessed: 12-Sep-2016].

[55] J. Nahon, "School of Computing," 2011.

[56] B. P. Douglass, *Real-Time UML: Developing Efficient Objects for Embedded Systems*. Boston, MA, USA: Addison-Wesley Longman Publishing Co., Inc., 1997.

[57] Oracle, "Differences between Java EE and Java SE - Your First Cup: An Introduction to the Java EE Platform." [Online]. Available: http://docs.oracle.com/javaee/6/firstcup/doc/gkhoy.html. [Accessed: 10-Sep-2016].

[58] Y. D. González and Y. F. Romero, "Patrón Modelo-Vista-Controlador.," *Rev. Telem Tica*, vol. 11, no. 1, pp. 47–57, 2012.

[59] "SSD cloud server, cloud hosting, cloud computing, vps, vps colombia," *DonWeb.com by Dattatec*. [Online]. Available: http://donweb.com/es-co/hosting-cloud-servers-vps. [Accessed: 12-Sep-2016].

[60] I. Voras *et al.*, "Evaluating open-source cloud computing solutions," in *MIPRO, 2011 Proceedings of the 34th International Convention*, 2011, pp. 209–214.

[61] "Windows server vs Linux server." [Online]. Available: https://ubuntuforums.org/showthread.php?t=2048062. [Accessed: 12-Sep-2016].

[62] S. Van Vugt, *Pro Ubuntu Server Administration*, vol. 1. Apress, 2009.

[63] "Whats is better and why? Linux CentoS or Ubuntu?," *DigitalOcean*. [Online]. Available: https://www.digitalocean.com/community/questions/whats-is-better-and-why-linux-centos-or-ubuntu. [Accessed: 12-Sep-2016].

[64] "Usage Statistics and Market Share of Linux for Websites, September 2016." [Online]. Available: https://w3techs.com/technologies/details/os-linux/all/all. [Accessed: 12-Sep-2016].

[65] C. Negus and T. Boronczyk, *CentOS*. Hoboken: Wiley [Imprint] John Wiley & Sons, 2009.

[66] B. TABERNER AGUAS, "Informatización de una PYME aplicando una metodología ágil: un caso real," 2015.

[67] "Apache Tomcat/8.0.9 Vs GlassFish Server 4.1 - entre Desarrolladores." [Online]. Available: http://entredesarrolladores.com/6896/apache-tomcat-8-0-9-vs-glassfish-server-4-1. [Accessed: 11-Sep-2016].

[68] "WildFly Homepage · WildFly." [Online]. Available: http://wildfly.org/. [Accessed: 11-Sep-2016].

[69] "Documentation." [Online]. Available: https://tomee.apache.org/documentation.html. [Accessed: 11-Sep-2016].

[70] "Oracle WebLogic Server Technical Information." [Online]. Available: http://www.oracle.com/technetwork/middleware/weblogic/overview/index.html. [Accessed: 11-Sep-2016].

IX - ANEXOS

1. Lista de anexos

Documentación del software

Los siguientes documentos hacen parte de la documentación del sotware. Cada uno de estos anexos puede ser descargado de la página de Trabajo de Grado de Tappi.

- SRS (Software Requirement Specification)
- SDD (Software Design Document)
- Requerimientos
- Líneas de emergencia
- Triage.eap (documento con el diseño completo de la aplicación desarrollado en la herramienta Enterprise Architect)
- Síntomas (documento que incluye el sistema basado en reglas para todos los síntomas descritos en el apartado del documento del ministerio de salud[21])

Documentación del servidor

Los siguientes documentos hacen parte de la documentación del servidor. Cada uno de estos anexos puede ser descargado de la página de Trabajo de Grado de Tappi.

- Manual de uso: Consumir Servidor (contiene las url del servidor para acceder a los servicios)
- TappiTree (contiene la lógica inicial de la secuencia de reglas como un árbol)
- Lógica del triage: contiene la lógica de la secuencia de reglas
- Diseño de la implementación (contiene las consultas para la lógica del triage)
- Implementación de las pruebas del servidor: contiene la implementación de las pruebas
- Plan de pruebas del servidor: contiene el diseño y el plan de pruebas
- Valores de pruebas Jmeter: contiene los datos resultantes de las pruebas de Jmeter
- Tutoriales para hacer las siguientes acciones:
    - Pruebas Junit
    - Cliente Tappi

- Nueva funcionalidad
- Web service

Documentación del código

Los siguientes documentos hacen parte de la documentación del código fuente. Cada uno de estos anexos puede ser descargado de la página de Trabajo de Grado de Tappi.

- Vista
- Servidor:
  - WSTappi: Web Service de Tappi (código fuente que contiene la lógica de negocio)
  - Prueba (métodos POST y GET por interfaz gráfica)
  - WebTappiManager (creación, edición, edición y borrado de las tuplas que contienen los síntomas)
  - WebViewTappi (interfaz gráfica de pruebas de usuario)
- Base de datos
  - DBderby (contiene el sql de la base de datos)
  - DBTappi (contiene el diseño de la base de datos)
  - Insert Tuplas (contiene los comandos insert para las pruebas de la base de datos)

Otros anexos

- Trabajo a futuro: documento que contiene los diagramas de proceso de las propuestas de trabajo a futuro.

2. Comparación de herramientas

Hosting

- Nube: las ventajas que se ofrecen son una escalabilidad horizontal incrementando la cantidad de VPS's y verticalmente ampliando los recursos de los servidores propios, flexibilidad por la cantidad de servidores y recursos necesarios; disponibilidad por

servicios redundantes; fácil manejo de los recursos; infraestructura dinámica; incluye el costo de una ip publica [59]. [60].

- Computador propio: Se requiere de comprar el servicio de ip pública con el proveedor de internet. Requiere tener el computador prendido todo el tiempo. Las políticas de seguridad y la información de propietario de computador se pueden ver comprometida. Baja escalabilidad. Baja flexibilidad Baja disponibilidad ya que no posee servicios redundantes.

Sistema operativo servidor

- Ubuntu 14.04: es un software libre que funciona por consola, al no tener interfaz gráfica no se genera un mayor peso en el servidor; tiene menores requerimientos de hardware para funcionar que Windows. No se tiene que reiniciar tanto cada vez que se realiza un cambio, por lo tanto, tiene menores costos para su implementación en la nube. Contiene un Firewall integrado con el host para proteger el servidor que se encuentran en internet lo que brinda una mayor robustez en el kernel. La integración con directorio activo es provista por Likewise Open. Basado en Debian. El 34.1% de las páginas de internet utilizan Ubuntu de todas las páginas que utilizan Linux. [59][61][62][63][64].

- CentOS (Community ENTerprise Operating System):Software libreBasado en Red Hat Enterprise Linux (RHEL). Brinda un mayor soporte que Ubuntu.El 34.1% de las páginas de internet utilizan Ubuntu de todas las páginas que utilizan Linux. [63][64][65].

- Windows: Software propietario por lo tanto para un hosting en la nube de este servicio se debe pagar licencia. Brinda interfaz gráfica. Es principalmente utilizado en páginas de internet con mayor tráfico [59][61][64].

Servidor

- GlassFish es un servidor de aplicaciones de software libre desarrollado por Sun Microsystems. Es código abierto. Implementa tecnologías Java EE. Tiene como base al servidor Sun Java System Application Server de Oracle Corporation, un derivado de Apache Tomcat. Utiliza fichero .war brindando conexiones bidireccionales y unidireccionales al igual que conexiones síncronas y asíncronas. Utiliza una interfaz gráfica para despliegue. Maneja EJB, JMS, JTA, RMI. Integración con Netbeans sencilla. Como ventaja brinda un alto acoplamiento con la aplicación. [66][67].
- Apache HTTP Server.Código abierto multiplataforma sin interfaz gráfica. No soporta EJB y la especificación de EJB expuestas como REST no es posible.[66]
- Wildfly: Soporta EJB, JMS. Interfaz gráfica de despliegue.Para el despliegue necesita host virtuales los cuales se deben hacer manualmente[68].
- Apache TomEE: Apache JavaEE 6. Extension de Apache Tomcat. Servlets JPA, JTA. Open Source. Comercialmente soportado [69]
- WebLogic es un Software propietario [70]

De acuerdo con las características descritas en la parte superior se pueden observar los diferentes hostings, sistema operativo (servidor) y servidores.El tipo de hosting que se eligió para la aplicación está enfocado en la nube, ya que como se puede observar con claridad posee mayores atributos de calidad. El sistema operativo que se utilizó fue Ubuntu ya que tenía mayor soporte por los proveedores de servicios en la nube adicionalmente a que los desarrolladores de la lógica de negocio de la aplicación se encontraban más familiarizados con este. Respecto al servidor se tuvo en cuenta que se debía poder trabajar con JavaEE, lo que es un idioma tanto como una estructura elegida para el desarrollo del software. Se tomó como elección Glassfish ya que se probó instalando Wildfly y presentó problemas en el despliegue del war debido a los hosts virtuales, también se probó Apache y el inconveniente es su carencia de interfaz gráfica para el despliegue de los proyectos y conexiones por JDBC pool a bases de datos.