

**Universidad Nacional Mayor de San Marcos**

Facultad de Ingeniería de Sistemas e Informática

E.A.P de Ingeniería de Software

=========================================================

Desarrollo de una Aplicación en Android utilizando Realidad Aumentada basada en Geolocalización para la Promoción y Difusión de eventos realizados en la ciudad universitaria de la Universidad Nacional Mayor de San Marcos

Tesis para optar el grado profesional de Ingeniero de Software

Autor:

Braulio Valentín Sánchez Vinces

Asesora:

Doctora Luzmila Elisa Pró Concepción

Lima - Perú 2015

=========================================================

Hoja en Blanco

FICHA CATALOGRÁFICA

SÁNCHEZ VINCES, BRAULIO VALENTÍN

Desarrollo de una Aplicación en Android utilizando Realidad Aumentada basada en Geolocalización para la Promoción y Difusión de eventos realizados en la ciudad universitaria de la Universidad Nacional Mayor de San Marcos

C.0.3 TECNOLOGÍAS DE LA INFORMACIÓN Y COMUNICACIÓN/ C.0.3. 22. Ingeniería de Software

Lima, Perú 2015

Tesis, Facultad de Ingeniería de Sistemas e Informática, Escuela Académico Profesional de Ingeniería de Software, Pregrado, Universidad Nacional Mayor de San Marcos.

Formato 28 x 20 cm Páginas

DEDICATORIA:

Este trabajo está dedicado a mi madre que siempre creyó en mí.

AGRADECIMIENTOS

Agradecer primero a mis padres por el apoyo, cariño y paciencia con la que me han criado y guiado durante toda mi vida.

Agradecer a todas las personas que han estado conmigo durante el desarrollo de esta tesis, a aquellas que me han acompañado en algún momento de mi vida.

A mi hermano que siempre ha sido un compañero, confidente y alguien con quién siempre vale la pena competir.

A mi asesora de tesis, quién, con su experiencia y pedagogía, me guió durante todo el desarrollo de esta tesis.

Agradecer, por último, a mis compañeros de universidad con los cuales he compartido muchos momentos de penas y alegrías.

**UNIVERSIDAD NACIONAL MAYOR DE SAN MARCOS**

**FACULTAD DE INGENIERIA DE SISTEMAS E INFORMÁTICA**

**ESCUELA ACADEMICO PROFESIONAL DE INGENIERÍA DE SOFTWARE**

**Desarrollo de una Aplicación en Android utilizando Realidad Aumentada basada en Geolocalización para la Promoción y Difusión de eventos realizados en la ciudad universitaria de la Universidad Nacional Mayor de San Marcos**

Autor: BRAULIO VALENTÍN SÁNCHEZ VINCES

Asesor: LUZMILA ELISA PRÓ CONCEPCIÓN

Fecha: Julio 2015

**RESUMEN**

El objetivo de la presente tesis es dar a conocer el desarrollo de una aplicación en Android utilizando realidad aumentada basada en geolocalización para la promoción y difusión de eventos realizados en la ciudad universitaria de la San Marcos. Se realiza una breve descripción de las actividades que realizan las diferentes facultades y los inconvenientes que presenta al momento de difundirlas, es por ello que se presenta el aplicativo como una solución y alternativo canal de difusión.

**Palabras claves:** realidad aumentada, android, geolocalización, GPS, brújula.

**UNIVERSIDAD NACIONAL MAYOR DE SAN MARCOS**

**FACULTAD DE INGENIERIA DE SISTEMAS E INFORMÁTICA**

**ESCUELA ACADEMICO PROFESIONAL DE INGENIERÍA DE SOFTWARE**

**Desarrollo de una Aplicación en Android utilizando Realidad Aumentada basada en Geolocalización para la Promoción y Difusión de eventos realizados en la ciudad universitaria de la Universidad Nacional Mayor de San Marcos**

Autor: BRAULIO VALENTÍN SÁNCHEZ VINCES

Asesor: LUZMILA ELISA PRÓ CONCEPCIÓN

Fecha: Julio 2015

**ABSTRACT**

The aim of this thesis is to present the development of an Android application using augmented reality based on geolocation in the promotion and dissemination of events in the San Marcos university city. A brief description of the activities carried out by the different faculties and the disadvantages when spread them, which is why the application is presented as an alternative solution and broadcast channel is performed.

**Key words:** augmented reality, android, geolocation, GPS, compass.

INDICE DE CONTENIDOS

[INTRODUCCIÓN 10](#_Toc424853586)

[CAPÍTULO I: PLANTEAMIENTO METODOLÓGICO 11](#_Toc424853587)

[1.1 Formulación del problema 11](#_Toc424853588)

[1.2 Objetivo principal 11](#_Toc424853589)

[1.3 Objetivos específicos 11](#_Toc424853590)

[1.4 Justificación 11](#_Toc424853591)

[1.5 Alcance 12](#_Toc424853592)

[CAPÍTULO II: MARCO TEÓRICO 13](#_Toc424853593)

[2.1 Sistema operativo Android 13](#_Toc424853594)

[2.2 Teléfonos inteligentes 14](#_Toc424853595)

[2.3 Realidad Aumentada 15](#_Toc424853596)

[2.4 Context-aware Computing 18](#_Toc424853597)

[2.5 Entorno de Desarrollo Integrado 20](#_Toc424853598)

[2.6 VISION SDK® 20](#_Toc424853599)

[CAPÍTULO III: ESTADO DEL ARTE 23](#_Toc424853600)

[3.1 Desarrollo de una aplicación con geolocalización para Android [13] 23](#_Toc424853601)

[3.2 memoAR - An Augmented Reality Application on Urban Story Telling [14] 23](#_Toc424853602)

[3.3 A framework for context-aware applications using augmented reality: A train station navigation proof-of-concept on Google Android [15] 24](#_Toc424853603)

[3.4 Aplicativos similares 25](#_Toc424853604)

[3.4.1 Wikitude 25](#_Toc424853605)

[3.4.2 Layar 26](#_Toc424853606)

[3.5 Metodologías 27](#_Toc424853607)

[3.5.1 Scrum 27](#_Toc424853608)

[3.5.2 XP 27](#_Toc424853609)

[3.5.3 RAD 28](#_Toc424853610)

[3.6 Comparativas 28](#_Toc424853611)

[CAPÍTULO IV: APORTE TEÓRICO 30](#_Toc424853612)

[4.1 Metodología 30](#_Toc424853613)

[4.2 Cómo trabaja Scrum 30](#_Toc424853614)

[4.3 Por qué Scrum es efectivo en la gestión de proyectos de software 36](#_Toc424853615)

[4.3.1 Un mecanismo sistemático de reducción de riesgo 36](#_Toc424853616)

[4.3.2 Un ciclo de vida de desarrollo de software ágil 37](#_Toc424853617)

[4.3.3 Un proceso de gestión de proyectos más adaptativo 38](#_Toc424853618)

[4.3.4 Un marco de gestión de proyectos y el proceso de desarrollo basado en la motivación y el orgullo de la gente 39](#_Toc424853619)

[CAPÍTULO V: APORTE PRÁCTICO 40](#_Toc424853620)

[5.1 Análisis de requisitos 40](#_Toc424853621)

[5.1.1 Introducción 41](#_Toc424853622)

[5.1.2 Descripción general 42](#_Toc424853623)

[5.1.3 Requisitos específicos 45](#_Toc424853624)

[5.2 Diseño 55](#_Toc424853625)

[5.2.1 Sprints 55](#_Toc424853626)

[5.2.2 Arquitectura de la aplicación 59](#_Toc424853627)

[5.2.3 Diagrama de clases 60](#_Toc424853628)

[5.2.4 Sistema web – http://visoraserver-bsanchez.rhcloud.com/ 62](#_Toc424853629)

[CAPÍTULO VI: CONCLUSIONES Y RECOMENDACIONES 68](#_Toc424853630)

[6.1 Conclusiones 68](#_Toc424853631)

[6.2 Recomendaciones 68](#_Toc424853632)

[6.2.1 Comunidades VisoRA 68](#_Toc424853633)

[6.2.2 Paneles flotantes más informativos 68](#_Toc424853634)

[6.2.3 Product owner, scrum master y scrum team 69](#_Toc424853635)

[6.2.4 Pago por disponibilidad 69](#_Toc424853636)

[REFERENCIAS BIBLIOGRÁFICAS 70](#_Toc424853637)

# INTRODUCCIÓN

En la presente tesis se plantea una alternativa de solución para la publicación efectiva de los eventos culturales organizados por las facultades ubicadas en la ciudad universitaria de San Marcos.

La distribución del contenido empieza con el planteamiento del problema en el CAPÍTULO I: PLANTEAMIENTO METODOLÓGICO y se cubre hasta la elección de la metodología de desarrollo como parte del CAPÍTULO III: ESTADO DEL ARTE. En el que también se muestran tres trabajos similares al presente y de los cuáles se han podido obtener alcances con respecto al desarrollo del aplicativo.

En el CAPÍTULO IV: APORTE TEÓRICO se presenta con detalle la metodología seleccionada para el desarrollo de esta tesis; el CAPÍTULO V: APORTE PRÁCTICO tiene la finalidad de mostrar tanto los requisitos planteados para este aplicativo, bajo la perspectiva del estándar IEEE 830, así como mostrar su diseño y arquitectura.

Por último el CAPÍTULO VI: CONCLUSIONES Y RECOMENDACIONES presenta una compilación de lecciones aprendidas.

# CAPÍTULO I: PLANTEAMIENTO METODOLÓGICO

## 1.1 Formulación del problema

La ciudad universitaria alberga a 17 de las 20 facultades que forman parte de la Universidad de San Marcos. Estas casas de estudio tienen un itinerario académico y cultural. Muchas de estas actividades culturales son poco conocidas por la mayoría de la población universitaria.

Eventos de cine, conversatorios, talleres, conferencias, recitales, concursos, etc; muchos de los cuales no son lo suficientemente publicitados como para obtener el alcance esperado. Si bien es cierto existen las redes sociales tales como Facebook, Twitter, Google+, entre otras, que pueden ser utilizadas como un medio masivo de difusión; pero para poder llegar a su público objetivo necesitan tener algún tipo de contacto con este público.

El uso de las páginas web de las mismas facultades que organizan los eventos no es muy recomendable dado a que no cuentan con un buen diseño, la información que presentan en las mismas no es de fácil acceso. En consecuencia la difusión del evento muchas veces pasa desapercibida.

Como última opción a tomar en cuenta por los organizadores de eventos es la elaboración de volantes, afiches o pancartas que son pegadas en las inmediaciones de la facultad organizadora, en algunos pasajes concurridos dentro de la misma ciudad universitaria. Incluso son colocados en los exteriores.

## 1.2 Objetivo principal

El objetivo principal es el desarrollo de una aplicación en Android que proporcione una solución alternativa al problema de la eficiente difusión de eventos realizados en la ciudad universitaria, aplicando realidad aumentada basada en geolocalización.

## 1.3 Objetivos específicos

1. Diseñar e implementar un sistema web centralizado para que las facultades puedan registrar aquellos eventos que vayan a realizar.
2. Diseñar e implementar una aplicación móvil en Android, la cual permitirá geolocalizar las facultades que tengan eventos registrados previamente y que estén vigentes, ello utilizando realidad aumentada.

## 1.4 Justificación

Gracias a los avances tecnológicos por parte de la industria del hardware, los dispositivos móviles han alcanzado una capacidad de cómputo antes impensable. Esto ha permitido que los terminales móviles no se limiten a realizar funciones de telefonía *clásica*, sino que han derivado en una suerte de ordenador portátil. Para explotar estas nuevas capacidades los fabricantes han decidido hacer públicas la mayoría de interfaces de programación para que desarrolladores ajenos a ellos puedan generar aplicaciones para sus terminales.

De esta manera se crea una especie de sinergia, ya que los desarrolladores tienen un nuevo ámbito en el que trabajar y los terminales se hacen más atractivos de cara al consumidor gracias a estas nuevas aplicaciones.

En la actualidad, el poder tener un dispositivo inteligente sea teléfono o tablet, es relativamente fácil. Si nos enmarcamos en la población de la ciudad universitaria son mayoría los que poseen un smartphone. Sobre este hecho, el poder contar con una aplicación móvil que permite geolocalizar los distintos tipos de eventos dentro de la ciudad universitaria es de mucha ventaja en comparación con los métodos de publicación antes mencionados.

Esta aplicación será completamente gratuita y no es necesario ser alumno sanmarquino para poder utilizarla, ya que esto limitaría la difusión de la información. Toda persona que pueda acceder a esta aplicación móvil, luego de aplicar algunos filtros según interés, podrá visualizar eventos vigentes a realizarse en la ciudad universitaria. Detalles como fecha, expositores, tema, cronograma; todo ello acompañado de una atractiva interfaz de realidad aumentada basada en geolocalización.

## 1.5 Alcance

La aplicación sólo permitirá visualizar puntos de interés geolocalizados dentro de la ciudad universitaria. Estos puntos, en adelante POIs (Point of Interest, por sus iniciales en inglés) [3] identificarán la situación geográfica de las diecisiete facultades que alberga la ciudad universitaria.

Tener disponible una conexión a internet para poder recibir la información actualizada del registro de eventos.

El desarrollo de esta aplicación está dirigida hacia la plataforma de teléfonos inteligentes o *smartphones* y tablets. El sistema operativo objetivo es Google Android OSTM, desde la versión 2.3.3 y superiores, requerimiento mínimo solicitado por las librerías a utilizar para la implementación de la aplicación (VISION SDK®)

# CAPÍTULO II: MARCO TEÓRICO

## 2.1 Sistema operativo Android

Android comenzó como Android Inc., una compañía fundada por Andy Rubin, Chris White, Nick Sears, y Rich Minero en octubre de 2003. Se centraron en la creación de dispositivos móviles que pudieran estar habilitados para contener la información y ubicación de las preferencias de las cuentas de usuarios. Después de navegar con éxito la demanda del mercado y las dificultades financieras, Google adquirió Android Inc., en agosto de 2005. Durante el período siguiente, Google comenzó a construir alianzas con compañías de hardware, software y empresas de telecomunicaciones con la intención de entrar en el mercado móvil.

En noviembre de 2007, la *Open Handset Alliance* (OHA) fue anunciada. Este consorcio de empresas, que incluían 34 miembros fundadores encabezados por Google, comparte un compromiso con la apertura. Además, se pretende acelerar la innovación de la plataforma móvil y ofrecer a los consumidores una experiencia móvil más completa, menos costosa, y mejor. La OHA ha crecido desde entonces a 84 miembros. Los cuales representan a todas las partes del ecosistema móvil, incluyendo los operadores móviles, fabricantes de teléfonos, compañías de semiconductores, compañías de software, y más. [9]

Con la OHA en su lugar, Google anunció su primer producto móvil, Android. Sin embargo, Google todavía no llevaba los dispositivos con Android al mercado. Finalmente, después de un total de cinco años, Android fue puesto a disposición del público en general en octubre de 2008. El lanzamiento del primer teléfono Android a disposición del público, el HTC G1, marcó el comienzo de una era.

Antes de la primera versión comercial de Android, el sistema operativo tenía Alfa y Beta. El Alfa era distribuida sólo para los miembros de Google y OHA. Android Beta fue lanzado el 5 de noviembre de 2007, que es la fecha que se considera popularmente como el cumpleaños Android.

La primera versión comercial, la versión 1.0, fue lanzado el 23 de septiembre de 2008, y la próxima versión, la versión 1.1, estaba disponible el 9 de febrero de 2009. Esas fueron las dos únicas versiones que no tienen un convenio de denominación por su nombre en clave.

A partir de Android 1.5, que fue lanzado el 30 de abril de 2009, los nombres en clave de las principales versiones fueron ordenados alfabéticamente con el nombre de postres. La versión 1.5 tiene el nombre en código *Cupcake*. En la Figura 1 se puede apreciar la distribución histórica de estas versiones desde el 2009 hasta abril de 2014.

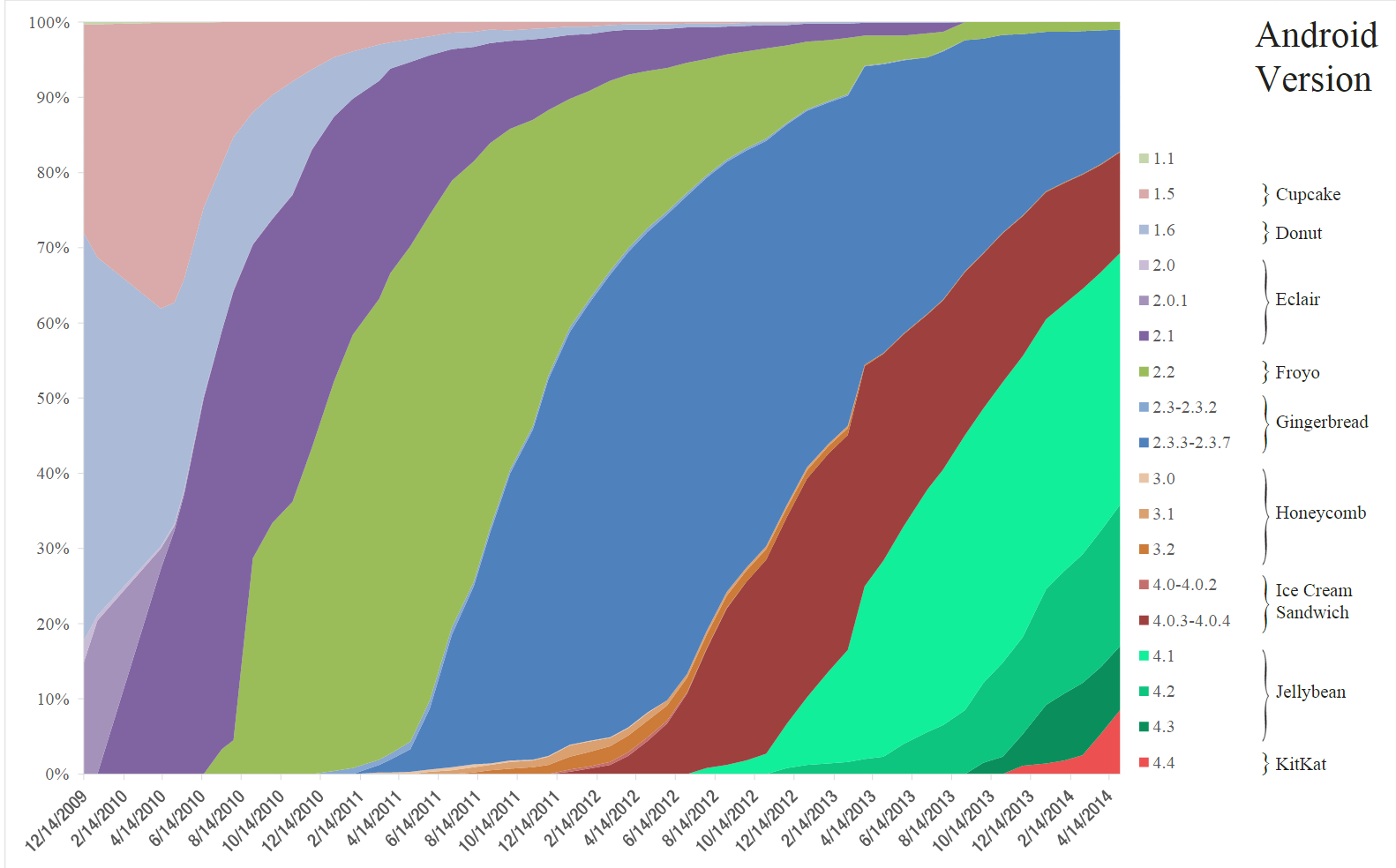


Figura 1: Distribución histórica de las versiones de Android

Fuente: <http://upload.wikimedia.org/wikipedia/commons/e/ee/Android_historical_version_distribution_-_vector.svg>

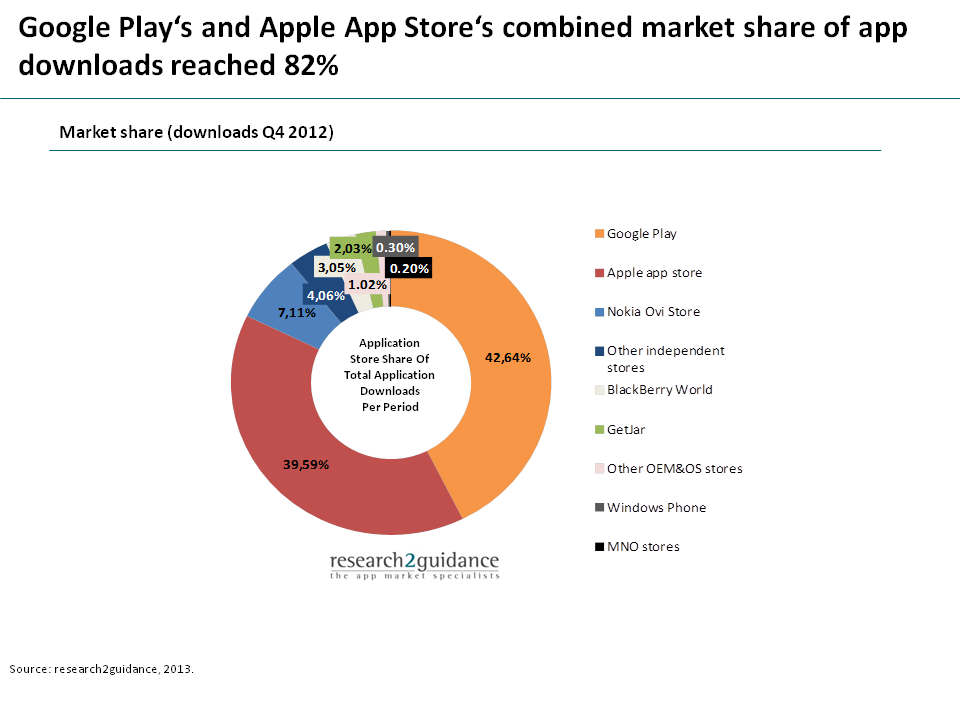
Adicionalmente cabe resaltar una característica importante en Android, es multimarca; es decir, varios fabricantes tienen modelos que utilizan el sistema Android, los más populares son HTC y Samsung. Además Google tiene su propia marca de teléfonos llamada Nexus, los cuales son fabricados también por la compañía taiwanesa HTC. [4]

## 2.2 Teléfonos inteligentes

El término teléfono inteligente o *smartphone*, tiene sentido si analizamos la historia de los terminales móviles. En un principio los teléfonos móviles sólo tenían capacidad para hacer llamadas y enviar mensajes de texto (sms). Por otro lado aparecieron los PDAs, grandes agendas electrónicas, donde el usuarios podía apuntar citas en el calendario, información de contactos e incluso descargar correos electrónicos de su ordenador.

Lo que diferencia al teléfono inteligente de las PDAs (actualmente en desuso [5]) y de los teléfonos corrientes son las capacidades. Todo teléfono inteligente debe tener: un sistema operativo (muchas de las cuales son basadas en versiones ligeras de Linux), software (que permita realizar acciones tales como editar un fichero de texto), acceso a internet (tanto por WiFi como por la red telefónica), teclado QWERTY (ya sea físico o táctil) y mensajería (tanto sms, mms como o correos electrónicos). [6]

La verdadera revolución que provocó la actual popularidad de estos teléfonos fue la aparición del App Store en Apple (y todos sus homólogos como el Android Market o el BlackBerry App World). La posibilidad de descargar contenidos abría a desarrolladores de aplicaciones un mercado potencial enorme. La bajada de los precios de los terminales, sumado a la de los planes de datos provocó que ese mercado potencial se volviese una realidad. Tal como se puede apreciar en la Figura 2. [2]

Figura 2: Mercado compartido de aplicaciones para teléfonos inteligentes

Fuente: <http://www.research2guidance.com/wp-content/uploads/2013/04/SAMM9-Blog-Post.png>

Uno de los avances introducidos a los teléfonos inteligentes han sido el GPS y los sensores diversos con los que ahora cuentan los dispositivos inteligentes. Ambas abren un campo nuevo para los desarrolladores y uno de ellos es la Realidad Aumentada.

## 2.3 Realidad Aumentada

La realidad aumentada, RA en adelante, está relacionada con el concepto de realidad virtual. La cual intenta crear un mundo artificial que una persona puede experimentar y explorar de forma interactiva, principalmente a través de su sentido de la visión, sino además a través del audio, sensación táctil, y otras formas de retroalimentación. La RA también trae consigo una experiencia interactiva, pero tiene como objetivo complementar el mundo real, en lugar de crear un entorno completamente artificial. Los objetos físicos en el entorno del individuo se convierten en el telón de fondo y de base para los objetos virtuales generados por ordenador.

Diferentes investigadores suscriben definiciones más estrechas o más amplias de lo que constituye exactamente AR. Si bien la comunidad científica está de acuerdo en gran medida en la mayor parte de los elementos de los sistemas de RA, ayudado por el intercambio y debates en varias conferencias internacionales en el campo, todavía hay pequeñas diferencias de opinión y de nomenclatura [3].

A pesar de que el término realidad aumentada fue acuñado en la década de los noventa, en 1968 Ivan Sutherland y algunos colegas suyos desarrollaron el primer sistema de realidad aumentada. Se trataban de unas gafas con un display que mostraba información sobreimpresa y se convirtió en la base de todos los estudios de hardware gráfico. Entre los años setenta y ochenta, la realidad aumentada fue uno de los ámbitos de investigación de varias instituciones estadounidenses, tales como el Ames Research Center de la NASA, el MIT y la Universidad de Carolina del Norte en Chapel Hill. Por último un sistema contextual y que además usaba realidad aumentada fue el *Chameleon*, desarrollado por la Universidad de Toronto a través de un monitor de mano [7].

No fue hasta 1997 en que la realidad aumentada pasó a ser móvil, gracias a los avances tecnológicos. Fue *The Touring Machine* de la Universidad de Columbia presentada en el International Symposium on Wearable Computing [8], la que rompió esta barrera por primera vez. El proyecto trataba de hacer una guía turística del campus de la Universidad de Columbia. El usuario debía cargar con una mochila que llevaba un ordenador, un módulo de GPS, una tarjeta de red para conectarse a internet, una fuente de alimentación y un ordenador, un medidor de campo magnético que hacía de brújula. Con esta información podemos hacernos una idea del avance que ha alcanzado el campo de la realidad aumentada gracias al permanente avance tecnológico.

Höllerer [3] establece que para desarrollar una aplicación de realidad aumentada son necesarios los siguientes módulos: computadora, pantalla, sistema de posicionamiento, interfaz con el usuario, conectividad inalámbrica, bases de datos y algún medio de acceso a ellas. Es notable que para el equipo desarrollador de *The Touring Machine* fue necesario un módulo físico casi para cada funcionalidad, mientras que actualmente es suficiente un terminal móvil de prestaciones estándar.

En la actualidad hay tres variantes en el campo de la realidad aumentada: basada en marcadores, basada en reconocimiento de objetos y basada en GPS o posicionamiento. Esta última es la variante a utilizar para el desarrollo de la aplicación móvil. El posicionamiento puede ser realizado por GPS o por otros métodos como la fuerza con la que la señal de WiFi llega al dispositivo. Uno de los inconvenientes de este sistema es la poca precisión que se obtiene, llegando a ser poco útil si se necesita una precisión de varios metros (ya que ésta depende del alcance del router WiFi).

Para el año 2011 el cuadrante mágico de Gartner ubicaba a la Realidad Aumentada como una tecnología emergente con alta expectativa. Actualmente la expectativa ha descendido, como se puede ver en la Figura 3.

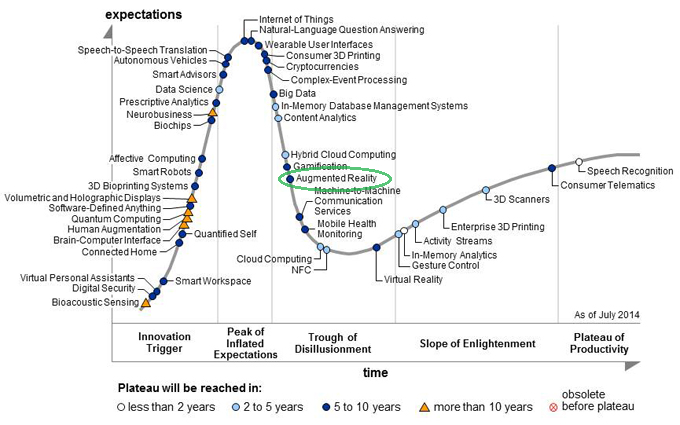


Figura 3: Gartner’s 2014 Hype Cycle for Emerging Technologies, con *Augmented Reality* remarcado.

Fuente: <http://www.gartner.com/newsroom/id/2819918>

Además de estar posicionada en *Trough of Disillusionment*, lo cual no significa que haya perdido vigencia ni mucho menos, cabe resaltar que el grupo Gartner trata el campo de realidad aumentada como una sola entidad. Sin embargo, existe una variación entre las diferentes áreas de aplicación de realidad aumentada; que se mueven a diferentes velocidades a lo largo de la curva del *hype* y algunos están todavía en las primeras etapas, mientras que otros son lo suficientemente maduros para ser explotados.

La realidad aumentada combinada con la geolocalización generará oportunidades de negocio y revolucionará la movilidad en 2014

* La utilización de la realidad aumentada mediante dispositivos como Google Glass desarrollará nuevas oportunidades de negocio y la expansión de la movilidad durante 2014. Se apuesta por la accesibilidad, usabilidad y geolocalización como las tres variables claves para el éxito de cualquier aplicación de movilidad.
* La evolución en el mundo de la movilidad debe basarse en tres parámetros: funcionalidad, tiempo y posición. Los simuladores serán tendencia, sobre todo en el área de la formación, aportando estímulos nuevos que incentivarán notablemente el aprendizaje. [1]

## 2.4 Context-aware Computing

Es un paradigma de programación que encuentra y usa información del entorno. Hay dos visiones principales sobre el concepto de context-computing. La primera es de Weiser en *Ubiquitous Computing* [10]. Para él hay dos factores principales que la creación de este paradigma. El primero es que cuanta menos interacción del usuario necesita una aplicación, mejor es ésta. El segundo es la creciente capacidad de cómputo de los sistemas embebidos y su reducido coste. Según Weiser, estos dos factores están íntimamente relacionados y abrirían la posibilidad de que la computación formase parte del entorno (context-computing).

|  |
| --- |
| Aplicación |
| Almacenamiento/Gestión |
| Pre procesado |
| Recuperación de datos en bruto |
| Sensores |

Tabla 1: Abstracción por capas del context-aware

La otra visión principal la tiene Ishii en *Tangible Bits* [11], que tiene como origen el mencionado anteriormente. En él Ishii plantea que el humano vive en dos mundos distintos, el mundo de la computación (*bits*) y el mundo real (*átomos*). Aunque el mundo real sea con el que el humano está más familiarizado, su interacción con el mundo de la computación se ha hecho sencillas, a través de la evolución. Actualmente la mayor parte de la interacción está basada en teclas y un ratón, mientras que Ishii se ha dedicado a buscar interacciones más amplias.

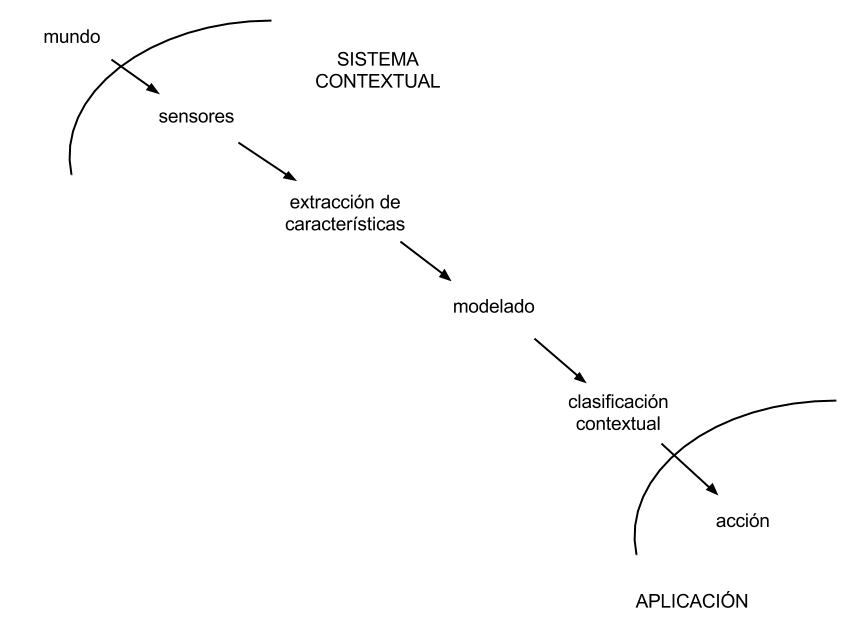


Figura 4: Arquitectura de un sistema de context-aware

En la Figura 4 se muestra lo que tienen en común las dos visiones antes expuestas, que es que toman el mundo como interfaz. Según Paul Dourish, en su artículo *Seeking a Foundation for Context-Aware Computing*, hay muy poco consenso en lo que context-aware computing es exactamente. Lo que queda evidente es que la aplicación móvil a desarrollar para esta tesis utiliza conceptos de este paradigma ya que dependiendo del entorno donde se encuentre el usuario, los resultados que aparecerán en la pantalla variarán.

|  |  |  |
| --- | --- | --- |
|  | **Manual** | **Automática** |
| **Información** | *selección por proximidad e información contextual* | *reconfiguración contextual automática* |
| **Comando** | *comandos contextuales* | *acciones contextualmente ejecutadas* |

Tabla 2: Dimensiones de un programa context-aware

Según Schilit [12], existen en el context-aware computing cuatro dimensiones, presentadas en la Tabla 2. La aplicación que se ha de desarrollar para esta tesis, vista desde las dimensiones de Schilit, se encontraría en Acciones Contextualmente Ejecutadas. Ya que la aplicación ajustará la información a mostrar según reglas o filtros especificados.

## 2.5 Entorno de Desarrollo Integrado

IDE es el acrónimo de *Integrated Development Environment* o entorno de desarrollo integrado.

Un IDE es una aplicación que facilita el desarrollo de aplicaciones. En general, un IDE es una interfaz de usuario (GUI) basada en el *workbench* de trabajo gráfico, diseñada para ayudar en la construcción de aplicaciones de software para un desarrollador con un entorno integrado combinado con todas las herramientas necesarias a la mano.

La mayoría de las características comunes, tales como la depuración, control de versiones y la estructura de datos de navegación, ayudan a un desarrollador a ejecutar rápidamente acciones sin cambiar a otras aplicaciones. Por lo tanto, ayuda a maximizar la productividad al proporcionar interfaces de usuario similares (UI) para los componentes relacionados y reduce el tiempo necesario para aprender un *lenguaje de programación*. Un IDE soporta *lenguajes* individuales o múltiples.

El concepto de IDE evolucionó desde un simple software basado en comandos que no era tan útil como software basado en menúes. Los IDEs modernos se utilizan sobre todo en el contexto de la programación visual, donde las aplicaciones se crean rápidamente moviendo bloques de construcción de programación o nodos de código que generan los diagramas de flujo y diagramas de estructura, que se compilan o interpretan.

La selección de un buen IDE se basa en factores como la compatibilidad con el idioma, el sistema operativo que necesita y los costos asociados con el uso de la IDE etc.

Para la construcción de la aplicación móvil y el sistema web centralizado se utilizará el IDE Eclipse en dos de sus muchas versiones, SDK 4.4 y para desarrolladores Java EE respectivamente.

## 2.6 VISION SDK®

VISION SDK® es una librería de vistas de Realidad Aumentada (RA) para dispositivos Apple® iOS o Google Android OSTM, que permite incorporar el modelo de uso definido por la RA a cualquier aplicación de Geolocalización sin necesidad de invertir tiempo en la programación de dichas vistas. VISION SDK® está basado en el producto VISION® *augmented user experiences* de GEOMOBILE, gracias al cual, cualquier App puede verse enriquecida rápida y fácilmente con las posibilidades que ofrece esta nueva tecnología, que integra información digital a la vista general del entorno de un modo natural.

Esta librería dispone de un conjunto de métodos para crear aplicaciones nativas mediante un *framework* que proporciona a los desarrolladores las facilidades necesarias para permitirles personalizar el comportamiento y la interfaz gráfica de sus Apps que integren RA y Geolocalización. Todo esto de forma rápida y sencilla, optimizando el tiempo invertido, garantizando la calidad y eliminando posibles errores, ya que VISION SDK® está desarrollado en lenguaje nativo, con código estable y muy probado.

Este modelo de uso de Realidad Aumentada con Geolocalización que implementa VISION SDK® se basa en combinar las imágenes que capta la cámara del *smartphone* o *tablet*, la información del acelerómetro, de la brújula digital así como la del GPS, para geoposicionar el dispositivo móvil del usuario y representar información digital (virtual) sobre la imagen real que está captando.

Esta información virtual son Puntos de Interés (POIs) que VISION SDK® permite agruparlos en Categorías y Subcategorías, siendo éstas las *capas de información*. Los POIs están georeferenciados mediante coordenadas y pueden tener asociada diferente información en forma de textos, imágenes, enlaces web o vídeos. Además, en función de la temática o funcionalidad de la aplicación, el desarrollador puede establecer distintos criterios de ordenación, selección y búsqueda para el usuario final.

VISION SDK® permite incorporar de un modo fácil y rápido el innovador modelo de uso de Realidad Aumentada al desarrollo de una aplicación, ejecutando vistas de RA de manera integrada con el resto de funcionalidades de una aplicación. De este modo, cualquier desarrollador puede incorporar la tecnología de la RA sin necesidad de desarrollar códigos de programación ni ser especialista en ello.

VISION SDK® favorece la integración entre la información digital (virtual) que proviene de la App con la imagen real del entorno, potenciando su entendimiento y aumentando la cantidad de información que proviene de ese entorno.

Pero las posibilidades de la RA y de la información asociada a cualquier POI hacen de VISION SDK® un potente aliado para completar los contenidos de todo tipo de Apps, en las que la información digital enriquezca la funcionalidad de una aplicación que se sirva de la ubicación de su usuario. Por ejemplo, información sobre ofertas dentro de un supermercado, localización de especialidades en el interior de un hospital, ubicación de obras en el interior de un museo, localización de expositores en una feria comercial, etc.

La Realidad Aumentada aplicada a una App supone un elemento diferencial y aporta una serie de ventajas para mejorar su utilidad y facilitar su uso:

* Muestra información en el punto de decisión o acción.
* Facilita la comprensión del entorno.
* Mejora la percepción al integrar la imagen real con la virtual.
* Incrementa el tiempo de uso de la aplicación.
* Permite nuevas posibilidades y capacidades de las aplicaciones.
* Impulsa el uso de contenidos, la participación y colaboración del usuario.

# CAPÍTULO III: ESTADO DEL ARTE

En este capítulo se analizarán algunos trabajos que tienen relación con esta tesis y con la aplicación a desarrollar.

## 3.1 Desarrollo de una aplicación con geolocalización para Android [13]

Se trata de una aplicación realizada por Elías Pardo que, dependiendo de la localización en la que se encuentre el terminal, muestra al usuario los puntos de interés cercanos a él mediante una interfaz gráfica. De esta manera el usuario puede obtener información acerca de los lugares que visita o se encuentra sin la necesidad de consultar una guía, mapa o preguntar a la gente de la localidad. La utilidad principal de la aplicación es servir de guía turística al usuario.

Otra de las características más destacables de la aplicación es que la interfaz es amigable, con la intención de que la aceptación sea la más alta posible. La aplicación utiliza base de datos en XML, que tienen la particularidad de ser fácilmente editables y sobre las que el usuario podrá ingresar las suyas propias. Ello permite un conjunto de posibles usos que pueda dársele a la aplicación, ya que al crear sus propias bases de datos definirán un nuevo uso para la misma. La intención del autor es que la aplicación sea lo más social posible, de manera que atraiga a nuevos usuarios, además de motivar la creación de nuevas bases de datos y nuevos usos.

Básicamente la aplicación funciona de la siguiente manera, el usuario ve algo que le interesa, ello puede ser tanto un edificio como un monumento o algún lugar en particular, luego procede a iniciar la aplicación. Toma una foto y se le muestra la información del punto. El nombre del lugar lo obtiene gracias a la base de datos de la aplicación, pero toda la demás información la obtendrá a partir de búsquedas en internet.

Finalmente, se puede apreciar que, dado el enfoque utilizado al aplicar la capa de realidad aumentada, la aplicación es versátil. En cierta forma, la posibilidad de cargar las bases de datos propias permite personalizar el uso del aplicativo, sin embargo limita el área útil de acción, como bien lo menciona el autor en sus conclusiones.

## 3.2 memoAR - An Augmented Reality Application on Urban Story Telling [14]

Esta tesis, desarrollada por Martin Nielsen, presenta una aplicación en realidad aumentada (RA) *location-aware*, que implica historias y entrevistas basadas en audio con etiquetas geográficas a lugares en un paisaje urbano. El proyecto es parte de una colaboración con *New Zealand Historic Places Trust* en *Christchurch*, Nueva Zelanda, donde los terremotos han dejado la ciudad en ruinas, y han creado un interesante potencial de historias urbanas. Una de las características de memoAR es su adaptación a contextos de otras ciudades.

Se realizaron dos experimentos. Un test de usabilidad en el que se investigó el potencial de los métodos de interacción basados en puntos y comparación de aquellos contra un único enfoque táctil. Los métodos basados en puntos son superados por el método de contacto único en términos de tiempo de finalización de tareas y errores, pero aún así se supone que poseen ventajas para tareas más complicadas en un escenario de interacción de audio.

También se llevó a cabo una prueba empírica de campo de la aplicación en uso, se llevó a cabo en dos escenarios urbanos: Un entorno concurrido y ruidoso en Aalborg, así como un entorno más rural y tranquilo en el pueblo de Sindal. A través de observaciones combinadas con cuestionarios, el test ha demostrado que la aplicación es muy fácil de utilizar, tanto a nivel de funcionalidades y en la aceptación de los usuarios.

El estudio evalúa la respuesta de la gente con respecto a la forma en que se presenta la información audible y/o publicada, toma en consideración un criterio importante para poder evaluar la distribución de los puntos de interés en el caso de que hubieran muchos de ellos en un área reducida, para lo cual propone el mecanismo de centrar el punto en la pantalla, lo cuál gatilla la información a recibir.

Por último se aprecia que la tesis ha tratado de poner las *historias de la calle* y *memoAR* en perspectiva mediante la presentación de los planes para la integración a redes sociales. En el contexto de la historias de la ciudad, es natural pensar en maneras de atraer a los usuarios socialmente, proporcionando oportunidades para que ellos generen su propio contenido (activamente), en lugar de simplemente disfrutar de contenido generado previamente (pasivamente). El objetivo general de las futuras experiencias móviles es probable que aumente con respecto al *context-aware* de los usuarios y hacerles interactuar con amigos, así como a extranjeros en relación a otros lugares y eventos de su ciudad. Después de haber desarrollado una experiencia de móvil de tour estático en historias de la ciudad, el siguiente paso es ampliar el alcance social de un tour de audio, en particular, así como las experiencias móviles, en general.

## 3.3 A framework for context-aware applications using augmented reality: A train station navigation proof-of-concept on Google Android [15]

En esta tesis, elaborada por Freek Uijtdewilligen, se presenta un framework llamado el ARCA Framework, que combina el soporte de aplicaciones para realidad aumentada y *context-aware*. El framework consiste en un software que se ejecuta en un dispositivo móvil y una aplicación de servidor, llamada *Context Information Service* (CIS). El CIS facilita la recolección y el refinado de información de contexto, y provisiona dicha información a otro software y sistemas, tales como el dispositivo móvil. Además, permite definir reglas que se pueden utilizar para especificar condiciones de contexto, junto con las notificaciones que se activan cuando se cumplen estas condiciones.

El software en el dispositivo móvil ofrece una visión en realidad aumentada, que muestra la información y el manejo de las notificaciones que han sido proporcionados por el CIS. Fue diseñado con mecanismos para ajustar el framework y así facilita especificar el comportamiento de la aplicación, interfaces de usuario, gráficos y el manejo de notificaciones.

Aunque el framework se ha diseñado e implementado para ser independiente del proceso de localización utilizado, una técnica probabilística, basada en los puntos de acceso WLAN, se ha implementado para proporcionar apoyo en los ambientes interiores.

La funcionalidad del framework ha sido evaluada por la implementación de una aplicación de navegación en una estación de tren, llamada *NS Navigator*, que es soportada por el framework.

El middleware que se utilizó para cumplir el papel de CIS, ofreció una base decente para el componente context-aware, pero, debido a las limitaciones en la implementación del middleware, se requiere de mayor tiempo y esfuerzo para ampliar su funcionalidad. Sin embargo, mediante la creación de componentes abstractos para las fuentes de contexto y los consumidores de contexto, implementando un número de procesadores de contexto sobre la base de estos componentes, y mejorando el código fuente del middleware, el framework se utilizó con éxito para proporcionar refinamiento de contexto, el contexto mismo y el aprovisionamiento de contenido, y por lo tanto, asegurar el soporte para el context-aware.

## 3.4 Aplicativos similares

### 3.4.1 Wikitude

Wikitude (aplicativo para Android e iOS) es un navegador de realidad aumentada que utiliza la cámara del celular y los datos de localización para ayudarte a ubicar monumentos históricos, restaurantes cercanos o incluso personas. Sólo se tiene que introducir un término de búsqueda, por ejemplo "restaurantes tailandeses", y luego por medio de la cámara del smartphone echar un vistazo alrededor, con una superposición digital de visualización de cualquier coincidencia cercana. También permite seleccionar categorías particulares o *worlds* de contenido si estás buscando restaurantes, lugares históricos, próximo eventos y otra información.

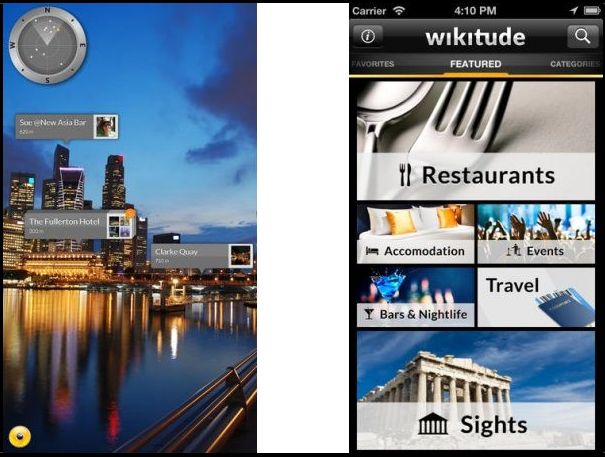


Figura 5: Interfaz gráfica de wikitude

### 3.4.2 Layar

Es un navegador que utiliza realidad aumentada, desarrollado para las plataformas Andriod e iOS. Los datos se presentan en forma de capas. Estas capas son servicios web REST, los cuáles proveen puntos de interés geolocalizados en las proximidades del usuario. Las capas son desarrolladas y mantenidos por terceros mediante una API libre.

Layar como empresa es responsable de la validación de las capas diseñadas utilizando su API, en el proceso de publicación. Los proyectos de terceros que utilizan Layar pueden ser aplicativos que requieren utilizar códigos QR, *geolayers*. El aplicativo también permite la reproducción de contenido multimedia a través de reconocimiento de imágenes, como por ejemplo desplegar un reproductor de video con algún trailer al posicionar el visor sobre el poster de una película en cartelera.

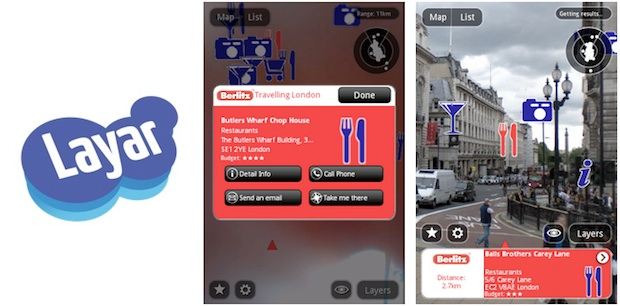


Figura 6: Interfaz gráfica de Layar

## 3.5 Metodologías

En esta sección se definirá la metodología a utilizar para el desarrollo de esta tesis. La elección de metodología es determinante durante el desarrollo puesto que, dependiendo de ella, habrá más cargas de trabajo durante ciertas fases y menos en otras. Hay también condicionantes a la hora de elegirla, por ejemplo la experiencia del desarrollador o la disponibilidad para realizar reuniones con el cliente o el supervisor del proyecto. A continuación se presentan tres opciones, ventajas y desventajas de cada una y por último un cuadro de puntuación para validar la elección de una de ellas.

### 3.5.1 Scrum

Es una metodología ágil que puede ser aplicada a casi cualquier proyecto; Sin embargo, la metodología Scrum es más comúnmente utilizado en el desarrollo de software. El proceso Scrum es adecuado para los proyectos que cambian rápidamente o los requisitos son muy emergentes. El desarrollo de software en Scrum progresa a través de una serie de iteraciones llamadas *sprints*, que duran de uno a cuatro semanas. El modelo Scrum sugiere que cada sprint comience con una breve reunión de planificación y concluya con un examen de autoevaluación. En Scrum se definen artefactos tales como las *historias de usuario* o requisitos, el *product backlog* o pila de requisitos, *sprint backlog* o historias de usuario a trabajar en un sprint, entre los más importantes. También podemos identificar tres roles, el *scrum master* o jefe de proyecto, el *scrum team* o equipo de desarrollo y el *product owner* o el cliente usuario. [16]

### 3.5.2 XP

Extreme Programming o programación extrema es una disciplina de desarrollo de software basado en los valores de simplicidad, comunicación, retroalimentación, coraje y respeto. Esta metodología pone más énfasis en la adaptabilidad que en la previsibilidad. Tomando en cuenta que los cambios de requisitos sobre la marcha en un proyecto de desarrollo de software es natural, inevitable e incluso deseable. Por lo tanto, ser capaz de adaptarse a los dichos cambios en cualquier punto de la vida del proyecto es una aproximación mejor y más realista que intentar definir todos los requisitos al comienzo del proyecto e invertir esfuerzos después en controlar los cambios en los requisitos. [17]

### 3.5.3 RAD

Rapid Application Development o desarrollo rápido de aplicaciones, es un modelo de proceso del desarrollo del software lineal secuencial que enfatiza un ciclo de desarrollo extremadamente corto. RAD es una adaptación a *alta velocidad* en el que se logra el desarrollo rápido utilizando un enfoque de construcción basado en componentes. Si se comprenden bien los requisitos y se limita el ámbito del proyecto, el proceso RAD permite al equipo de desarrollo crear un *sistema completamente funcional* dentro de períodos cortos de tiempo. [18]

## 3.6 Comparativas

En este apartado se realiza una comparativa de las metodologías antes revisadas utilizando una matriz con características basadas en algunos procesos CMMI niveles 2 y 3 de madurez [19]. Se utiliza un rango de calificación de 0 a 5, dónde 0 es característica no presente y 5 característica bien definida.

|  |  |  |  |
| --- | --- | --- | --- |
|  | **SCRUM** | **XP** | **RAD** |
| *Administración de requisitos* | 5 | 5 | 4 |
| *Planificación del proyecto* | 3 | 0 | 3 |
| *Monitoreo y control del proyecto* | 5 | 5 | 3 |
| *Medición y análisis* | 3 | 3 | 5 |
| *Aseguramiento de la calidad del producto* | 5 | 5 | 5 |
| *Gestión de la configuración* | 5 | 5 | 3 |
| *Administración del riesgo* | 5 | 5 | 5 |
| *Integración del producto* | 3 | 3 | 3 |
| *Verificación* | 5 | 5 | 5 |
| *Validación* | 5 | 5 | 5 |
| **PUNTUACIÓN** | **54** | **51** | **41** |

Tabla 3: Comparativa puntuada entre SCRUM, XP y RAD

En base a los resultados obtenidos luego de asignar puntos según algunas características representativas en el proceso de desarrollo de software, se ha llegado a la conclusión de optar por la metodología de desarrollo ágil Scrum.

# CAPÍTULO IV: APORTE TEÓRICO

En este capítulo se analizará la metodología a utilizar para la realización del aplicativo motivo de esta tesis.

## 4.1 Metodología

En esta sección definiremos la metodología escogida en el capítulo anterior, luego de ser evaluada bajo los criterios de algunos procesos CMMI niveles 2 y 3 de madurez.

Scrum, históricamente, proviene de un artículo publicado por Hirotaka Takeuchi y Ikujiro Nonaka en el Harvard Business Review en el año de 1986. En ese paper titutlado *The New New Product Development Game* [20], Takeuchi y Nonaka describieron un enfoque holístico en el cuál los equipos de proyecto están formados por pequeños equipos multifuncionales, trabajando con éxito hacia una meta común.

Mientras trabajaba en la elaboración de una herramienta *Object Oriented Analysis and Design* (OOAD) para *Easel*, *Jeff Sutherland*, entonces vicepresidente de ingeniería en Easel Inc. se dio cuenta que su equipo de software necesitaría una versión mejorada de la metodología RAD (Rapid Application Development). Lo que buscaba era un proceso similar a Scrum donde al final de cada iteración corta, el CEO de Easel viera la demostración del código trabajando, en lugar de papeles con diagramas de Gantt.

Durante más o menos el mismo período, *Ken Schwaber* estuvo observando cómo podría ayudar a su compañía, *Advanced Development Methods, Inc.* (ADM), a mejorar sus procesos de software así como la productividad de sus equipos de desarrollo.

Después de analizar más a fondo cómo otros exitosos proveedores de software construían software, Ken se dio cuenta que todos sus procesos de desarrollo eran similares en que todos ellos utilizaban procesos empíricos, los cuales requieren inspección constante y adaptación.

Luego, en 1995, a solicitud de la *Object Managment Group* (OMG), Jeff y Ken trabajaron juntos para juntar toda su experiencia obtenida a través de los años, ellos crearon una nueva metodología, a la cual llamaron Scrum, y describieron en un artículo de Schwaber, *Scrum and the Perfect Storm* [21].

## 4.2 Cómo trabaja Scrum

Tal como se aprecia en la figura 7, el *Scrum team*, el cuál puede ser un equipo multifuncional, está compuesto de un *ScrumMaster*, un *product owner*, y un *team development* (o simplemente, el *team*), con todas las habilidades necesarias para construir el producto de software.

Incluso en las mejores situaciones, cuando la composición del equipo Scrum es altamente multidisciplinaria, no se sigue necesariamente que los miembros del proyecto Scrum son, organizacional y jerárquicamente, toda la parte de la estructura del equipo de proyecto Scrum permanente. A menos que la compañía sea un defensor decidido de la implementación de Scrum en toda la empresa y se halla reorganizado completamente a sí misma, siguiendo las recomendaciones de estructura organizacional de Scrum. Los equipos de Scrum todavía son tomados de diferentes organizaciones a las que pertenecen, como un analista de calidad, arquitecto empresarial, preproducción, o DBA.

Incluso algunas de esas empresas habiendo adoptado Scrum, todavía cuentan con áreas de gestión de proyectos, control de calidad, pre-producción, sistema de negocio, grupos de análisis y arquitectura empresarial de trabajo, como funciones separadas en TI y los equipos de proyecto tienen que negociar con estos grupos para conseguir sus proyectos Scrum en marcha.

Muy rara vez una empresa despedirá a todos sus jefes de proyecto (Scrum no tiene un gerente de proyecto *per se*) o cambiar su título a Scrum Master o reorganizar todos sus grupos de TI separados alrededor de los equipos de proyectos Scrum multidisciplinarios. Eso sería ideal, pero todavía no es así en realidad.

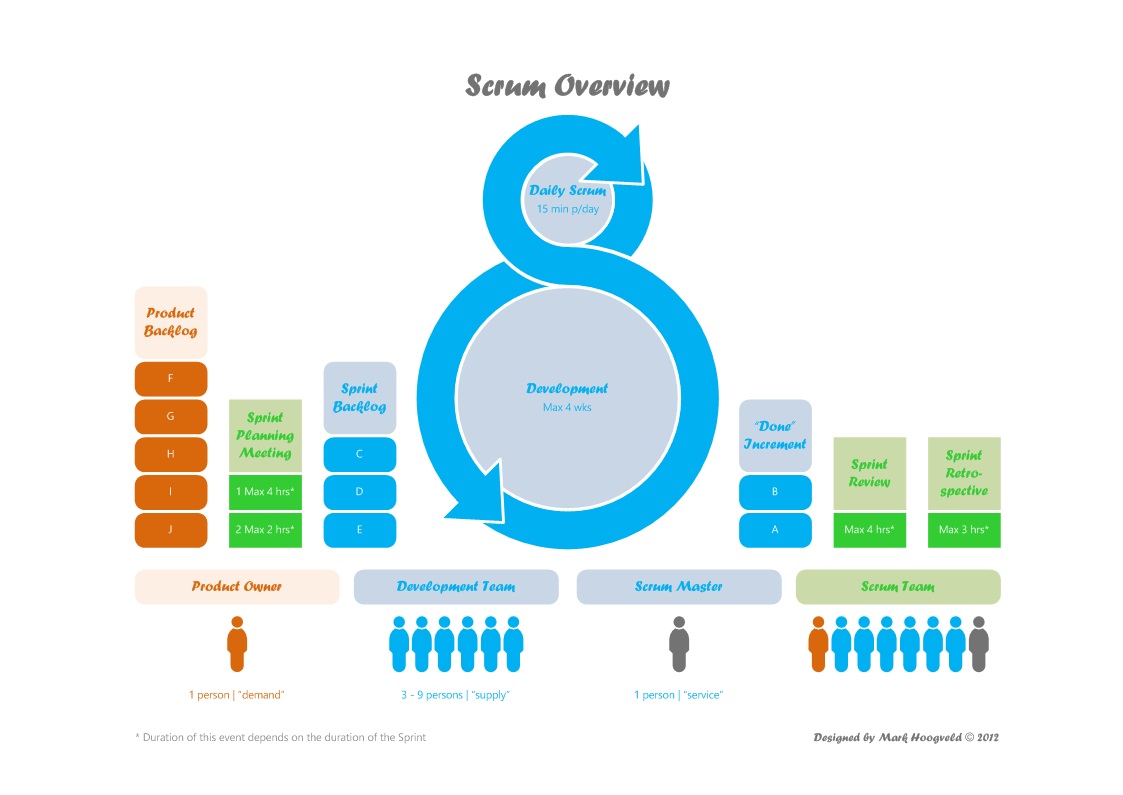


Figura 7: Product backlog, release backlog y el Sprint backlog.

Fuente: <http://calvinx.com/2014/05/22/why-scrum-why-agile-development/>

Se puede apreciar como un proyecto Scrum empieza con el *product owner*, quien es el responsable de tomar todas las inquietudes de los diferentes *stakeholders*, o usuarios que los representan, para elaborar una lista de requerimientos para crear el *product backlog*.

Simplemente colocarlo, el *product backlog* es una lista priorizada de los requerimientos, los cuales pueden incluir todo desde características de negocio hasta tecnologías, desde problemas técnicos hasta errores corregidos.

Algunos profesionales y autores, tales como Henrik Knitberg en su libro, *Scrum and XP from the Trenches* [22] quien prefiere mantener el *product backlog* a nivel de negocio para incluir únicamente requerimientos de negocio.

Los requerimientos de usuarios en el *product backlog* están usualmente agrupados como cortas historias de usuario durante un taller de requisitos de uno o dos días antes de la liberación y la reunión de planificación de *Sprint*.

Si bien la planificación de lanzamiento en Scrum era algo opcional en sus primeros días, se ha demostrado que ayuda a muchos equipos de Scrum a que se vuelven aún más eficaces a lo largo de los años. Por lo tanto, se recomienda que el *product owner* pasa por la planificación de la liberación con el *team*, incluso si se trata de un ejercicio difícil, ya que se requiere aprender sobre el producto antes de la reunión de planificación.

Es más, ya que el *product owner* conoce el producto, puede ayudar más al equipo. El objetivo clave de la planificación de la liberación es que el equipo Scrum pueda identificar todas las versiones del producto de software, además de tener un calendario de entrega probable. Normalmente, la planificación de la liberación debería realizarse las últimas cuatro horas por Sprints de cuatro semanas.

Además de la planificación de la liberación, el *Scrum team* también debe pasar por un poco de planificación del Sprint, ya sea como parte del proceso de planificación de liberación o de forma independiente luego de realizada la planificación de la liberación.

Normalmente, la reunión de planificación del Sprint debe ser alrededor de ocho horas por Sprints de cuatro semanas y debe ser ajustada a cuatro horas por Sprints de dos semanas.

Como una práctica común, la reunión de planificación del Sprint debería ser dividida en dos reuniones de cuatro horas iguales.

Durante la primera parte de la reunión, el *product owner* irá a través de los requerimientos, como historias de usuarios, para decidir, con el feedback del *team*, cuáles deben ser parte de qué Sprint y cuáles son sus objetivos. La primera de las dos partes de la reunión es principalmente la respuesta a la pregunta *QUÉ*.

Durante la segunda parte de la reunión de planificación del Sprint, el cual se enfoca en el *CÓMO*, el equipo de desarrollo intentará identificar tareas desde las historias de usuario previamente elegidas y estimar cuánto tiempo les tomará convertir esas tareas en incrementos de productos potencialmente entregables. A menos que el *team* utilice algún tipo de software de planificación, todas las tareas de desarrollo que son parte del Sprint normalmente serán consignadas a un *Task Board* (tablero de tareas), una especie de pizarra en una pared, para la fácil asignación y seguimiento del equipo.

Tan pronto como se realiza la liberación y la reunión de planificación del Sprint, luego comienza el trabajo real, como se ve en la figura 8, junto con sus 15 minutos diarios Scrum, o *Daily Standup*.

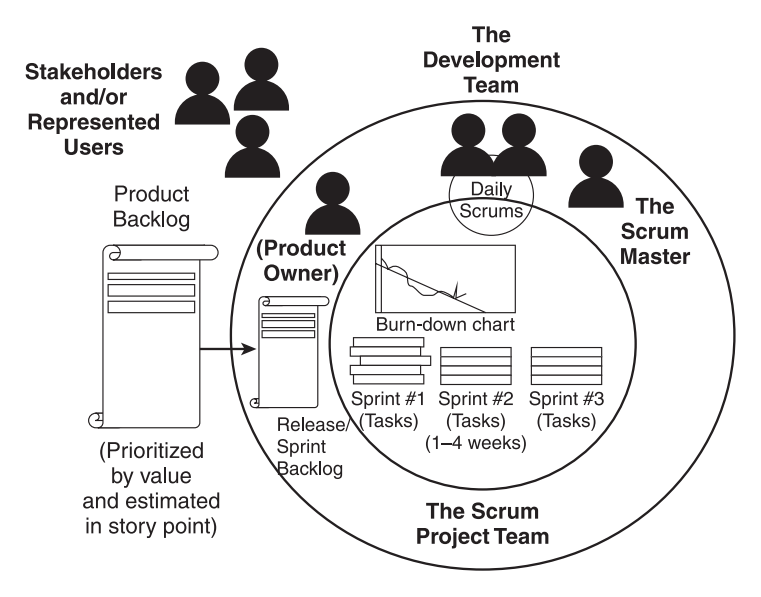


Figura 8: Sprints, burndown chart y daily Scrums.

Originalmente, el *Daily Standup* podría durar hasta 30 minutos, pero como parte de la evolución de Scrum (o su ajuste), su duración ha sido reducida más y más en la práctica a 15 minutos actualmente.

Normalmente, la duración de un Sprint será de una a cuatro semanas. Excepto en circunstancias muy especiales, no hay elementos adicionales a ser añadidos o borrados del *product backlog* del Sprint, mientras que el Sprint está en marcha, a menos que el equipo y el dueño del producto están de acuerdo con ello, pero esto es algo así como una excepción y no la norma.

A diferencia del proceso tradicional, donde el director del proyecto es el responsable de la organización de reuniones semanales para hacer un seguimiento del estado del proyecto, con Scrum, el equipo se reunirá todos los días para inspeccionar, no el estado del proyecto, sino el progreso del equipo hacia la meta del Sprint.

Para mantener el seguimiento del progreso del equipo hacia el objetivo del Sprint, un *burndown chart* será creado por el *team* para mostrar cuánto trabajo le queda al equipo hasta que el Sprint esté terminado. A pesar de que la creación de esta tabla *burndown* es responsabilidad del equipo, puede ser actualizado por el *Scrum Master* siempre que el equipo no tenga el tiempo para hacerlo.

Justo antes del final de cada Sprint, el *team* se reunirá con el *product owner*, como parte del mecanismo de inspección y adaptación de Scrum, para ir a través de lo que se conoce como un *Sprint review* organizado por el Scrum Master, tal como se aprecia en la figura 9. Se trata de una nueva reunión que normalmente dura cuatro horas para un período de Sprints de cuatro semanas o dos horas para Sprints dos semanas.

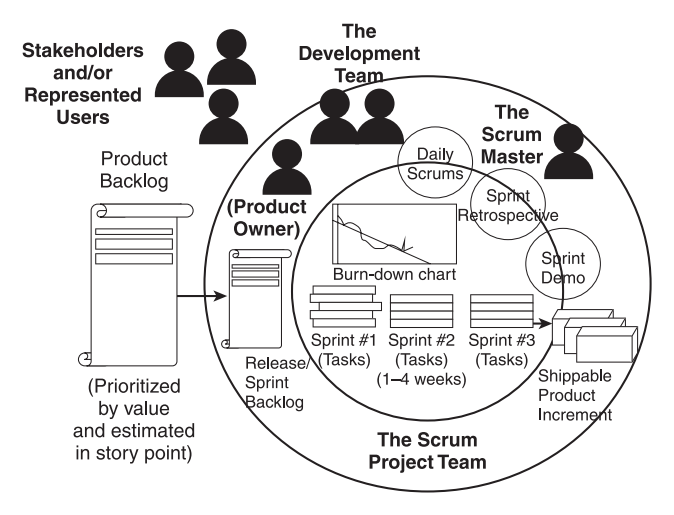


Figura 9: Sprint review y restrosprectiva

El objetivo de esta reunión es múltiple:

* La primera es para que el *Scrum team* y el *product owner* discutan qué fue hecho y qué no fue hecho.
* La segunda es para que el *team* demuestre qué fue construido al *product owner* y obtener su feedback.
* Finalmente, el tercer objetivo es obtener actualizaciones del *product owner* con respecto a nuevos cambios en el producto o dirección del mercado.

Justo después el *Sprint review* y antes del siguiente Sprint, el *Scrum team* se reunirá también para pasar por una retrospectiva del Sprint terminado para identificar lo que funcionó y lo que no funcionó. La intención es ver cómo pueden hacer su colaboración aún más eficaz antes de entrar en el próximo Sprint.

Como una práctica común, la reunión de retrospectiva debería normalmente realizarse las tres últimas horas para un Sprint mensual, pero su duración debería ser ajustada como las otras reuniones, en proporción a la longitud del Sprint, así como dos horas para un Sprint de dos semanas.

La figura 10 provee un gráfico total de las responsabilidades colaborativas de los miembros del *Scrum team*.



Figura 10: Toda la colaboración entre el *team*, el *Scrum Master* y el *product owner*

En conjunto, Scrum no parece tan complicado como un framework de proyectos general. Sin embargo, mientras parece simple en teoría, puede llegar a ser muy complicado de implementar, especialmente si la organización es aún nueva en la práctica de Scrum o si se busca *turbo cargar* el proyecto Scrum con algunas de las mencionadas prácticas.

## 4.3 Por qué Scrum es efectivo en la gestión de proyectos de software

Como ha sido mencionado anteriormente, Scrum puede ser difícil de implementar, sin embargo ha demostrado ser extremadamente eficaz cuando se ha desplegado correctamente.

Como lo presenta Henrik Knitberg en su libro *Scrum and XP from the Trenches*, Scrum es normalmente más efectivo en el desarrollo y administración de proyectos. Hay, sin embargo, cuatro ventajas a mencionar antes:

### 4.3.1 Un mecanismo sistemático de reducción de riesgo

Toda persona o grupo de personas que son responsables de la gestión o ejecución de un proyecto conocen cuán importante es reducir el nivel de riesgo o incertidumbre a cero o al menor nivel posible.

Si bien hay cuatro maneras de hacer frente a los riesgos (evitar, transferir, aceptar y mitigar) [23], los administradores de proyecto terminan mitigando el riesgo al final. Esto es donde Scrum destaca por su frecuente ciclo de inspección y adaptación.

### 4.3.2 Un ciclo de vida de desarrollo de software ágil

En la figura 11 se puede apreciar un ciclo de vida de desarrollo de software tradicional, el cual se caracteriza por su longitud.

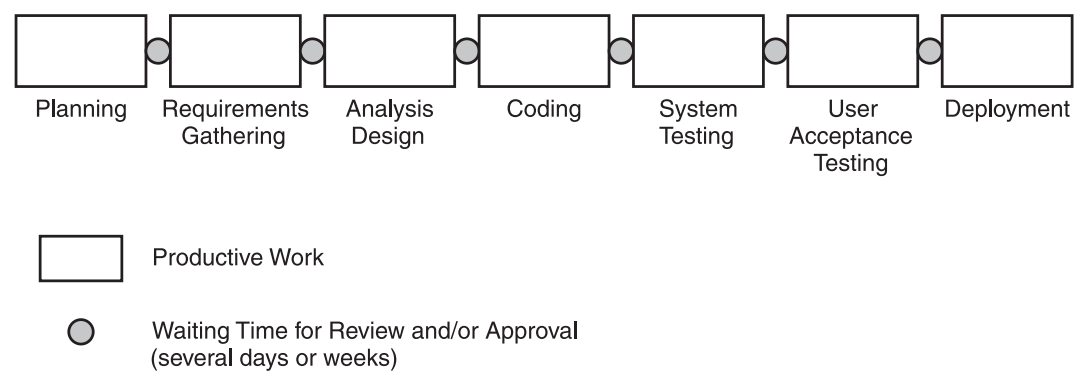


Figura 11: Tradicional flujo de valor

Mientras que otro equipo que utiliza Scrum puede utilizar un ciclo de vida mucho más *delgado* (figura 12).

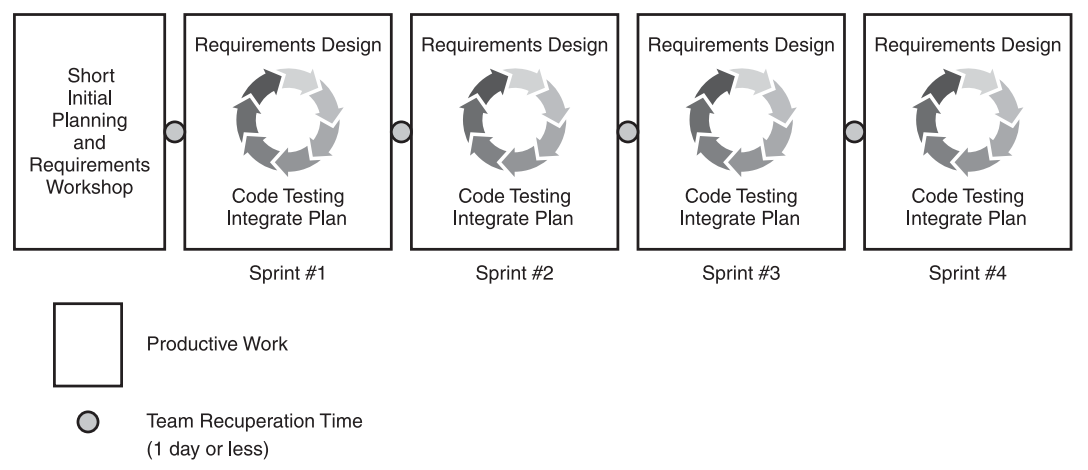


Figura 12: Scrum flujo de valor

### 4.3.3 Un proceso de gestión de proyectos más adaptativo

A diferencia de los procesos secuenciales usados en el entorno *cascada*, el cuál considera la estabilidad del proyecto fundamental (figura 13). Scrum observa más al cambio como la única constante (figura 14).

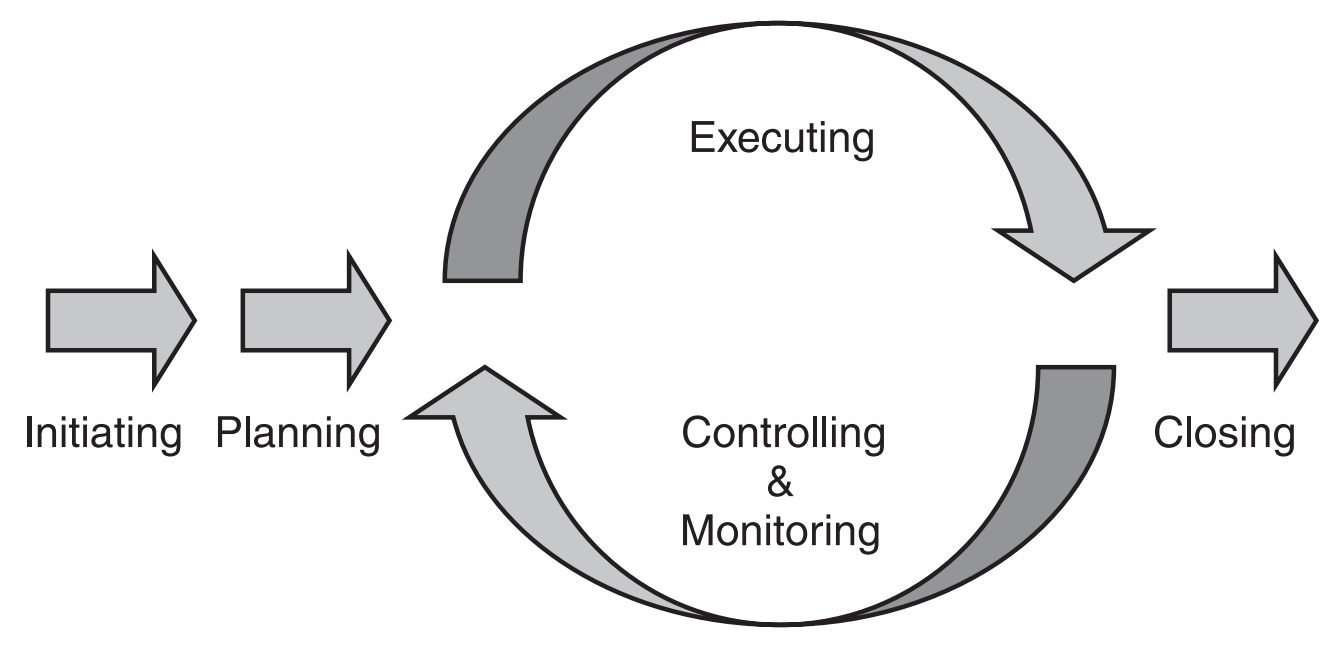


Figura 13: Procesos tradicionales secuenciales

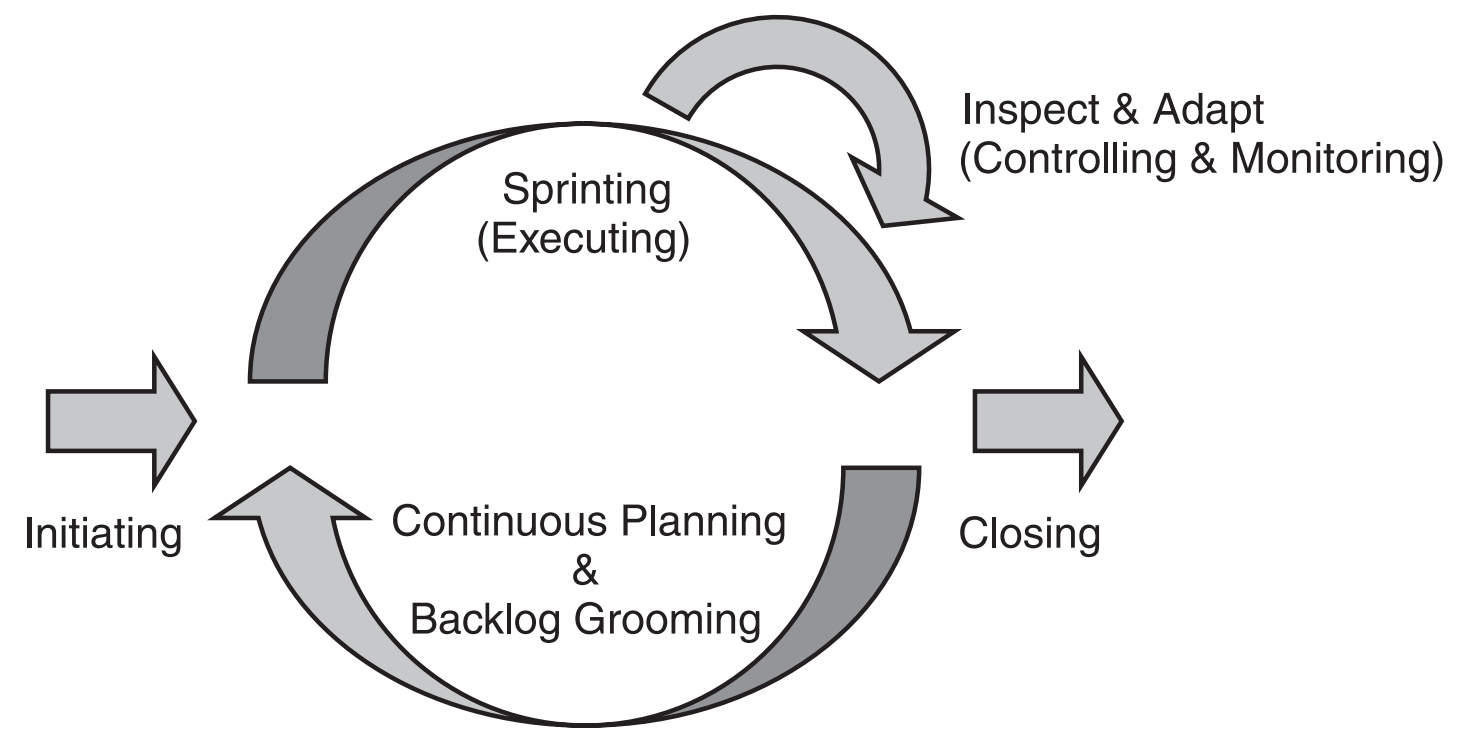


Figura 14: Gestión de proyectos adaptativo con Scrum

### 4.3.4 Un marco de gestión de proyectos y el proceso de desarrollo basado en la motivación y el orgullo de la gente

Más que otras características este puede ser uno de los más poderosos principios de Scrum. El nuevo enfoque no es tener un gerente diciendo a los miembros del equipo lo que deberían estar haciendo, sino dejar al equipo que decidan por ellos mismos cómo van con el cumplimiento de su trabajo.

Scrum propone un nuevo marco de trabajo para la gestión de proyectos de software, el cual está basado en la *auto organización*, *motivación*, *propiedad* y *orgullo* de los logros del equipo del proyecto.

# CAPÍTULO V: APORTE PRÁCTICO

En el presente capítulo se describirán las actividades y documentos realizados mientras se desarrollan el aplicativo en Android y el aplicativo web.

## 5.1 Análisis de requisitos

Para realizar el análisis de requisitos se va a utilizar como guía el estándar IEEE 830 [24]. Este estándar establece que para describir correctamente los requisitos de una aplicación, el documento debe seguir el índice que se presenta a continuación:

1. Introducción

1.1. Propósito

1.2. Alcance

1.3. Definiciones, acrónimos y abreviaciones

1.4. Referencias

1.5. Visión global del documento

2. Descripción general

2.1. Perspectiva del producto

2.2. Funciones del producto

2.3. Características del usuario

2.4. Restricciones

2.5. Suposiciones y dependencias

2.6. Requisitos futuros

3. Requisitos específicos

4. Apéndices

Algunas de las secciones establecidas por el estándar no serán tomadas en cuenta para la elaboración del documento, dado que el mismo estándar deja a criterio de quién lo utilice la aplicación de los puntos de sus índice, motivo por el cual se suprimirán los puntos: 1.3 Definiciones, acrónimos y abreviaciones; 1.4 Referencias; y 4 Apéndices. El índice del documento quedaría de la siguiente manera:

1. Introducción

1.1. Propósito

1.2. Alcance

1.3. Visión global del documento

2. Descripción general

2.1. Perspectiva del producto

2.2. Funciones del producto

2.3. Características del usuario

2.4. Restricciones

2.5. Suposiciones y dependencias

2.6. Requisitos futuros

3. Requisitos específicos

Otras recomendaciones que hace el estándar es que el análisis sea rastreable, esto quiere decir, que se pueda hacer seguimiento en el tiempo de las modificaciones realizadas sobre el documento y por consiguiente los requisitos que se han ido definiendo en historias de usuario.

Los primeros requisitos fueron crear una aplicación que pudiese mostrar una etiqueta geolocalizando el lugar de los eventos.

Una vez cumplidos estos requisitos, para conseguir un esbozo de lo que sería la aplicación final, se añadió a la aplicación la capacidad de obtener la lista actualizada de eventos registrados desde una aplicación web, que haría las veces de servidor de eventos, a través de un servicio web. Esta capacidad de actualizar y mostrar los eventos es disparada luego de identificar el punto de interés e interactuar con él por medio de un toque.

Con respecto a la aplicación web que hace de servidor, cuenta con una interfaz de usuario para poder registrar los eventos por facultad. Se implementó un servicio web que consumirá el aplicativo en Android.

### 5.1.1 Introducción

#### Propósito

Atendiendo a la metodología que ha sido elegida para el desarrollo de la aplicación, la definición de requisitos se traducirá en historias de usuario que permitirán elaborar el product backlog priorizado. La aparición de nuevos requisitos y la modificación de los ya identificados fueron manejados en las reuniones Sprint semanales y sus respectivos reviews.

#### Alcance

La aplicación desarrollada para esta tesis se llama VisoRA. VisoRA es una aplicación de realidad aumentada que utiliza geolocalización para la promoción de eventos realizados en la ciudad universitaria del Universidad Nacional Mayor de San Marcos, soportada en smartphones y tables con plataforma Android 2.3.3 o superior. Los usuarios de la aplicación será cualquier persona que tenga interés por encontrar información actualizada y con un mínimo esfuerzo de las diversas actividades culturales, políticas, deportivas y sociales realizadas en la ciudad universitaria.

El aplicativo está limitado al uso de conexión de internet para recibir la lista actualizada de eventos desde el servidor de eventos (<http://visoraserver-bsanchez.rhcloud.com/>).

#### Visión global

En las siguientes secciones del Análisis de Requisitos se procederá a hacer una descripción global del producto, así como las funcionalidades que debe implementar, el prototipo de usuario, además de las condiciones que afectan al desarrollo de este producto.

### 5.1.2 Descripción general

#### Perspectiva del producto

El aplicativo VisoRA, es un producto autocontenido con respecto a los puntos de interés que geolocalizan las facultades de la ciudad universitaria, tiene dependencia con respecto al servidor de eventos, ya que consume el servicio web para poder actualizar la lista de eventos cada vez que el usuario interactúa con el marcador en realidad aumentada.

Como se mencionó en el capítulo 1, en el apartado 1.5 Alcance; la aplicación ha sido desarrollada para smartphones y tables con sistema operativo Android 2.3.3. Puesto que las nuevas versiones de Android que aparezcan en el mercado son retrocompatibles, se espera que la aplicación no tenga ningún problema para instalarlo en versiones posteriores.

La interfaz de usuario debe ser lo más sencilla posible y toda interacción debe ser por la pantalla táctil (sin teclado alfanumérico).

Las ventanas deberán seguir una progresión lineal, es decir, si el usuario realiza una acción y quiere volver atrás, sea posible hacerlo.

La conexión que usará la aplicación para internet se realizará mediante redes de datos inalámbricas (WiFi) y redes de datos de operadores telefónicos. La localización se obtendrá mediante el sistema GPS y la funcionalidad extendida de triangulación proporcionada por la librería VISION SDK.

#### Funciones del producto

En la siguiente tabla se muestra la pila de backlog priorizada de las historias. La tabla contiene los siguientes campos:

* **ID**: una identificación única, es número auto incremental.
* **Nombre**: breve nombre descriptivo de la historia. Normalmente de 2 a 10 palabras.
* **Imp.**: importancia definida por el product owner.
* **Est.**: el tiempo estimado se calcula utilizando la siguiente fórmula tomada del libro de Henrik Kanitberg [22]

Pregunta al equipo “si tú puedes tomar un número óptimo de personas para esta historia (no tan pocos ni tantos, normalmente dos), y los encierras en un cuarto con mucha comida y trabajo completamente sin distracciones, ¿luego de cuántos días vendrán con una implementación terminada, demostrable, testeable y liberable?” Si la respuesta es tres personas en aproximadamente cuatro días, entonces el tiempo estimado inicial sería de 12 puntos.

Páginas 6 - 7

* **Cómo demostrarlo**: una descripción de alto nivel de cómo la historia se demostrará. Esto es esencialmente una simple especificación de prueba.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| PRODUCT BACKLOG | | | | |
| **ID** | **Nombre** | **Imp.** | **Est.** | **Cómo demostrarlo** |
| 1 | Mostrar puntos de interés | Alta | 2 | Abrir la aplicación y lanzar la vista de realidad aumentada. Observar los marcadores de geolocalización. |
| 2 | Desplegar la lista de eventos | Alta | 4 | Abrir la aplicación y lanzar la vista de realidad aumentada. Tocar un marcador. Observar la lista de eventos actualizados. |
| 3 | Distinguir los eventos por categorías | Media | 1 | Abrir la aplicación y lanzar la vista de realidad aumentada. Observar que los marcadores tengan indicadores por categoría. |
| 4 | Registrar eventos | Media | 4 | Loguearse en el sistema web. Registrar evento. Verificar la lista de eventos registrados. |
| 5 | Comunicar eventos registrados hacia la aplicación | Media | 3 | Loguearse en el sistema web. Registrar evento.  Abrir aplicación y lanzar la vista de realidad aumentada. Verificar evento nuevo registrado. |
| 6 | Administrar los eventos registrados | Media | 5 | Loguearse en el sistema web. Editar los eventos registrados, cambiar nombre, eliminar evento, terminar evento. Verificar las modificaciones en la lista de eventos. |

Tabla 4: Product Backlog

#### Características del usuario

Con respecto al perfil del usuario habrá que distinguir un límite mínimo y máximo entre los 16 y los 65 años. Está orientado a usuarios con cualquier nivel de educación, pero sobre todo a aquellos universitarios interesados en diversos eventos de la ciudad universitaria.

Se presupondrá cierto conocimiento e interés por las nuevas tecnologías.

#### Restricciones

El mayor condicionante en este desarrollo es la plataforma objetivo, un dispositivo móvil. Las limitaciones principales causadas por este motivo son la duración de la batería, conectividad a internet limitada, escasa capacidad de cómputo, poca memoria RAM y el tamaño de la pantalla. La libertad de movimiento que ofrecen estos terminales también se convierte en una restricción, aunque para este caso, es el que le da sentido a la aplicación.

#### Suposiciones y dependencias

En un futuro, con el avance de las capacidades de los terminales móviles, las especificaciones se cambiarán acordemente. Además es de suponer que las nuevas actualizaciones del sistema operativo Android, cada vez más frecuentes, permitirán al programador desarrollar aplicaciones más avanzadas.

Además, dado que existen numerosos dispositivos Android, es poco viable comprobar el correcto funcionamiento de la aplicación en todos ellos. Lo más factible sería realizar pruebas sobre aquellos dispositivos más conocidos en el mercado. Para este caso se utilizó un dispositivo Sony Xperia modelo Z1 Compact D5503.

El aplicativo interactúa con varios módulos disponibles en las librerías Android y VISION SDK. Estos se llamarán directamente desde el código fuente de manera que no hay que implementarlo otra vez. Además utiliza servicios tales como el GPS y la conexión a internet (ya sea por red de datos telefónica como por WiFi).

#### Requisitos futuros

Se han previsto mejoras futuras que no han sido consideradas para la versión actual del aplicativo. A continuación se muestran los más importantes:

* Utilizar la funcionalidad de poder cargar puntos de interés de un servicio web según la posición actual del dispositivo al momento de ingresar a la aplicación. Esta función está disponible en la librería VISION SDK, sin embargo el alcance de esta tesis sólo contempla la ciudad universitaria de San Marcos.
* Poder crear categorías propias y agrupas diversos puntos de interés arbitrariamente a criterio del usuario.
* Crear una comunidad de usuarios de VisoRA para poder compartir categorías creadas o modificadas.

### 5.1.3 Requisitos específicos

#### Requisitos para interfaces externas

*Interfaces de usuario*

Las vistas que se muestran al usuario deben ser lo más intuitivas posibles, indicando al usuario en todo momento que puede hacer para proceder a utilizar la aplicación. En ningún caso se mostrarán más de tres botones, exceptuando la lista de facultades.

La sobreimpresión de los paneles flotantes sobre los puntos de interés geolocalizados debe realizarse de manera que estén alineados con el punto de interés real.

El panel flotante que aparece en la vista principal muestra el nombre de la facultad, la distancia con respecto al usuario, el logo de la aplicación y un marcador de colores según la categoría de los eventos registrados en la facultad (figura 16).



Figura 16: Panel flotante

Las vistas que están disponibles para el usuario son las que ofrece la librería VISION SDK y fueron personalizadas para el desarrollo de este aplicativo son las siguientes. La vista de lista (figura 17), la vista principal de paneles (figura 18), la vista de flechas (figura 19), la vista de radar (figura 20) y la vista de eventos registrados (figura 21)



Figura 17: Vista de lista



Figura 18: Vista de panales



Figura 19: Vista de flechas

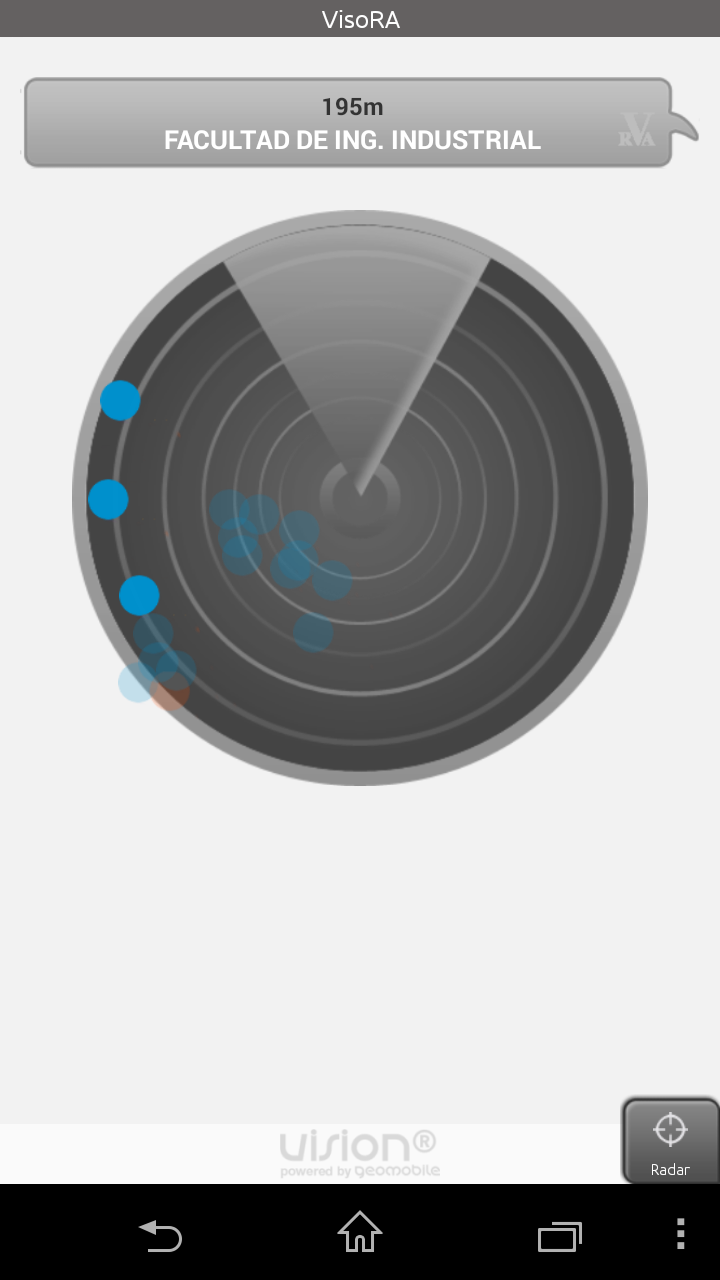


Figura 20: Vista de radar

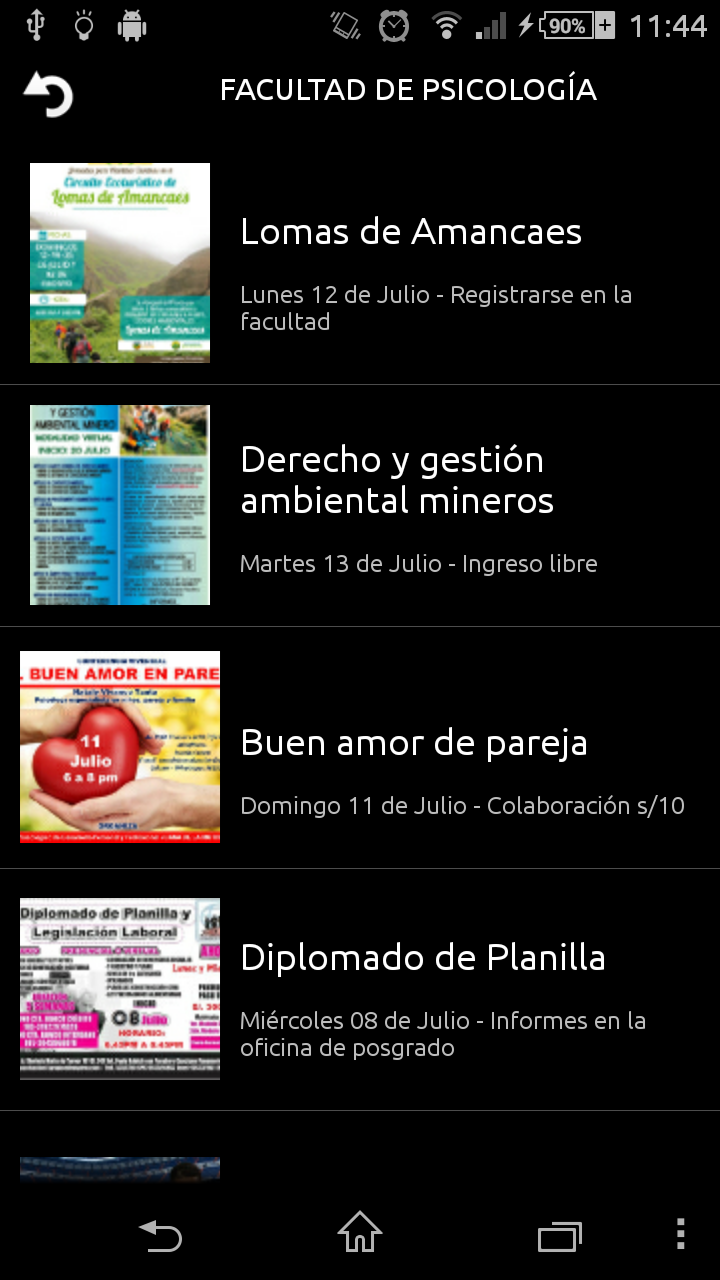


Figura 21: Vista de eventos registrados

Todas estas vistas pertenecen a la librería VISION SDK y están disponibles para su personalización, al agregar o eliminar funcionalidades.

Se deberá dar al usuario la posibilidad de que él mismo seleccione la lista de eventos registrados para una facultad y que no aparezcan en la misma vista principal (figura 18). De esta manera, si el usuario sólo está interesado en los eventos registrados, por poner un ejemplo, en la facultad de ingeniería de sistemas e informática, el mismo usuario verá varios paneles sobre la vista principal, pero no la totalidad de los eventos en cada panel. Este evento se gatilla luego de realizar un toque sobre el panel que indica Facultad de Ing. de Sistemas e Informática.

El usuario podrá interactuar entre las diferentes vistas personalizadas para la aplicación, las cuáles difieren dependiendo de la posición en la que se encuentre el móvil, como se aprecia en las figuras 22 y 23.

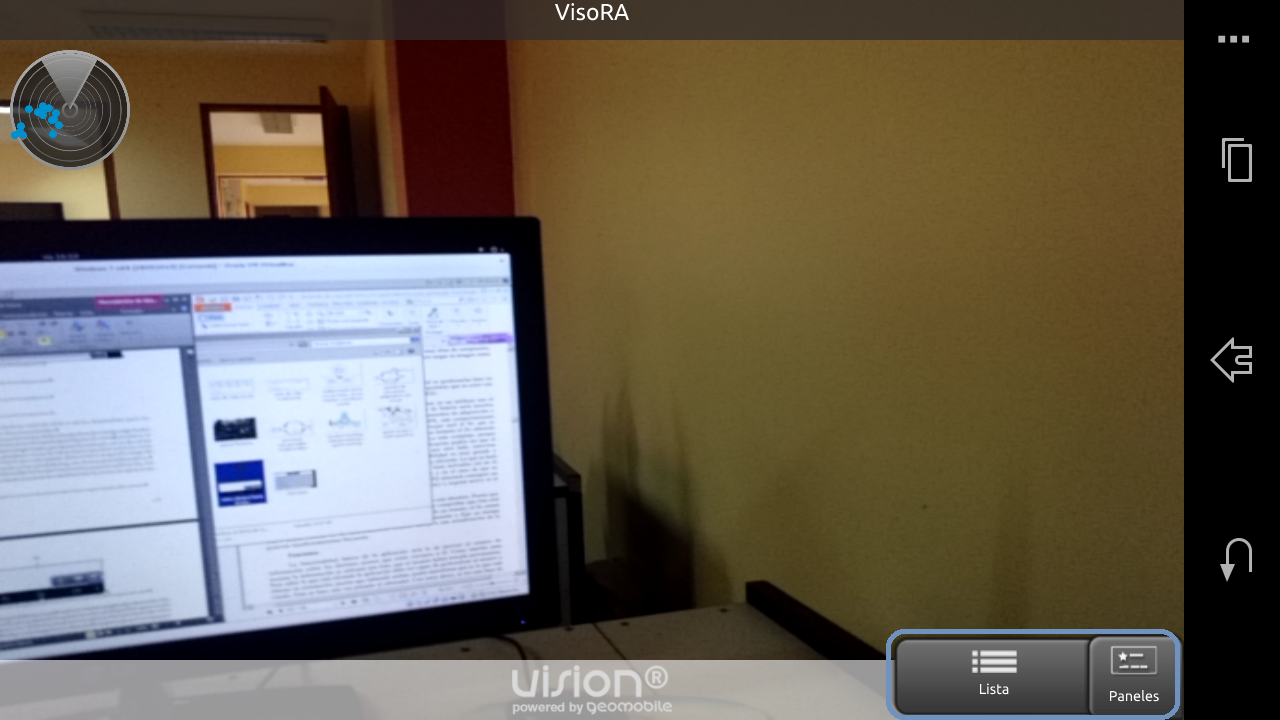


Figura 22: Vistas disponibles en la aplicación con la cámara posterior apuntando al horizonte



Figura 23: Vistas disponibles en la aplicación con la cámara posterior dirigida al suelo

Todas las vistas personalizadas disponibles en la aplicación no tienen ningún tipo de restricción con respecto a la posición del móvil, esto quiere decir, que si se coloca el dispositivo apuntando con al horizonte y luego se desplaza la visión hacia el piso la aplicación cambiará de vista automáticamente de paneles a flechas. Con respecto a la vista de flechas esta se activa siempre que la cámara trasera del móvil apunte al piso directamente, esto es, la posición horizontal con respecto al suelo debe ser de esa manera, si se colocara el dispositivo en la misma posición horizontal pero con la cámara trasera apuntando al cielo, la vista activa sería la de paneles y las opciones disponibles serían las mismas que aparecen en la figura 22.

La interacción posible o exigida al usuario no debe exceder la cantidad de dos acciones por vista. Es decir que para pasar de una vista a otra el usuario sólo deberá realizar dos acciones como máximo, ya sea pulsando un botón desplegable y luego escogiendo alguna de las opciones disponibles, clickar en algún punto de la pantalla. Esto se aplicará en todas las vistas.

*Interfaces de Hardware*

El aplicativo hará uso de la cámara trasera del teléfono móvil, la cuál será utiliza como visor para la vista principal de paneles.

La aplicación hará uso de las señales GPS para geolocalizar al usuario y a los puntos de interés para identificar a las facultades la ciudad universitaria. Este requisito es indispensable para que el aplicativo muestre la información correcta con respecto a la geolocalización.

El terminal deberá tener conexión a internet, ya sea mediante transmisión de datos como por WiFi. La aplicación seguirá funcionando a pesar de no tener conexión, pero no podrá acceder a ninguna información con respecto a la lista de eventos registrados.

El aplicativo hará uso de la brújula, por lo que el dispositivo móvil deberá tener disponible el servicio de los sensores.

*Interfaces de Software*

La aplicación no debe usar ninguna librería que no sea la de la librería estándar de Android y la de VISION SDK.

Los puntos de interés se representan mediante la clase VisoraGeoPoi, la clase VisoraARManager para gestionar el origen de los datos de la aplicación. Estas clases serán explicadas a detalle en el apartado de Diseño.

La aplicación podrá consumir como mucho 13 megabytes de memoria RAM, es un requisito impuesto por el sistema operativo Android. Puesto que los elementos que más recursos consumen son la cámara posterior y los receptores GPS.

*Interfaces de Comunicación*

En cuanto a las interfaces de comunicación, lo esencial es gestionarlas bien para así lograr un mejor uso del recurso batería del móvil. Además aparece un riesgo importante de seguridad al utilizar estas interfaces, sobre todo con las redes WiFi.

Una de las funcionalidades que más batería consumen en un teléfono son el adaptador GPS y WiFi. Lo óptimo para reducir el consumo de batería sería tenerlos deshabilitados durante toda la ejecución, exceptuando los momentos de adquisición y transmisión (en el caso de WiFi) de datos.

En el caso del GPS, el uso de la librería VISION SDK permite una gestión de esta interfaz de manera autónoma, es decir, la misma librería administra cuándo se accede y cuándo no. Luego de personalizar la librería se ha restringido mucho más el uso del GPS, tan sólo para las vistas principales. El caso del adaptador WiFi es un poco más complejo, debido a que sólo se utilizaría la interfaz para cuando el usuario ingresara al panel indicador para ver los eventos registrados.

#### Funciones

La funcionalidad básica de la aplicación será la de proveer al usuario información sobre los distintas facultades de la ciudad universitaria de la San Marcos (puntos de interés). Como interfaz para mostrar esta información se utilizarán las diferentes vistas presentadas previamente, vista de panel (vista principal), vista de flechas, vista de radar y vista de lista.

Para saber lo que está mirando la aplicación debe ser capaz de geolocalizar al usuario y obtener su orientación, puesto que sabiendo ambas, podrá determinar qué es lo que está observando por el visor. Esto se hace pulsando el botón TOQUE de la pantalla de inicio (figura 22).



Figura 23: Pantalla de inicio

La aplicación deberá seguir la siguiente secuencia de ejecución presentada en la figura 24.

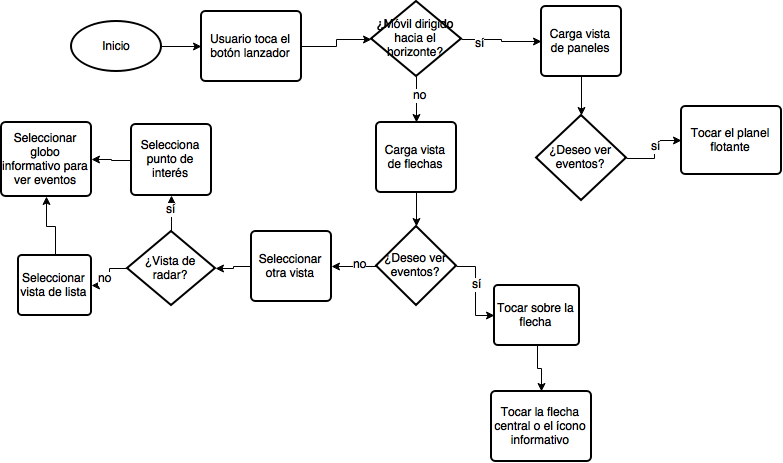


Figura 24: Diagrama de flujo

En el diagrama de flujo se presentan las actividades que ha de realizar el usuario para interactuar con el aplicativo móvil. La vista principal es lanzada luego de presionar el botón *toque* en la pantalla de inicio, los paneles se habilitan cuando el móvil está dirigido hacia el horizonte. Caso contrario la vista cambia a la de flechas, en ambos caso para poder ver los eventos de las diferentes facultades la interacción es la misma, tocar sobre el punto de interés geolocalizados.

Por motivos de simplicidad el usuario no va a tener la necesidad de configurar nada, al menos en la versión actual de VisoRA. El flujo de las actividades es lo más intuitivo posible. También se habilitó un botón de retroceso o *back button* para poder retroceder a una vista previa y navegar con mayor facilidad. Además para evitar que la vista que muestra los eventos registrados consulte continuamente el servicio web del servidor *visoraserver* , se ha forzado que no cambie de orientación y se mantenga en vertical, esto se hace para reducir el consumo de recursos al utilizar la interfaz de conexión a internet.

#### Requisitos de rendimiento

La aplicación sólo se instanciará una vez por móvil, limitando el número de usuarios posibles por móvil a uno. La cantidad de información que manejará debe ser mínima, dadas las variadas capacidades de los teléfonos. Debe ser capaz de ejecutarse en un terminal con las siguientes características mínimas:

* Android Google API 10
* 128 MB de RAM.
* Procesador de 200 MHz o mayor.

Puesto que estos son los requisitos mínimos en cuanto a rendimiento del móvil para que pueda ejecutarse el sistema operativo Android correctamente, puesto que se van a utilizar funcionalidades adicionales como el GPS o internet, se presupone que los requisitos serán mayores.

#### Atributos del software

A continuación se expondrán alguno de los atributos que debe tener el software y que pueden ser considerados como requisitos.

*Fiabilidad*

El aplicativo deberá poder ejecutar todas sus funcionalidades siempre que se cumplan las siguientes condiciones:

* El móvil tiene conexión a internet, bien será por WiFi o red de datos.
* El móvil dispone de servicio GPS y este está activo.
* El móvil tiene cámara con lente posterior.
* El móvil dispone de batería suficiente.

Además de las condiciones anteriores, es necesario que al móvil cumpla con unos requisitos adiciones para el correcto funcionamiento del aplicativo. En el caso de estar conectados a una red de datos o una conexión WiFi, verificar que se tenga acceso a internet. Respecto al servicio GPS, además de tener activado y habilitado el adaptador GPS, también es imperativo asegurar que la señal se recibe correctamente en el móvil, para esto hay que evitar espacios cubiertos, en general cualquier obstáculo que pueda reducir la intensidad o anular las señales de los satélites.

*Seguridad*

El aplicativo deberá solicitar al usuario permiso para conectarse a internet, utilizar el GPS y controlar hardware (la cámara). No se requerirá ningún otro permiso más allá de los antes mencionados. Ellos serán solicitados antes de la instalación, bien sea por medio del Play Store de Google o vía instalación directa por archivo de extensión apk.

Al utilizar conexión inalámbrica vía WiFi la encriptación de los datos transmitidos será responsabilidad del punto de acceso.

El aplicativo no permite el uso indebido o malicioso del mismo.

*Mantenimiento*

El mantenimiento del aplicativo debe ser mínimo, a parte de las implementaciones nuevas que agreguen a futuras versiones. El mantenimiento será requerido para el sistema web, <http://visoraserver-bsanchez.rhcloud.com/>, en el cuál se administran los eventos registrados, así mismo la base de datos. Este servicio está desplegado en un PaaS de OpenShift.

*Portabilidad*

La finalidad el esta tesis no era desarrollar un aplicativo que pueda ser portable, pero al elegir como plataforma Android, es factible que sea mucho más fácil su aceptación y masificación.

Para versiones futuras estaría disponible para IOS.

## 5.2 Diseño

En este apartado empieza con la descripción de cada uno de los cuatro Sprints definidos en base al product backlog elaborado en la tabla 4. Luego se describe la arquitectura general de la aplicación, tanto de cliente *VisoRA* como del sistema web *visoraserver*. Por último se hace el análisis del modelo de base de datos para el sistema web.

### 5.2.1 Sprints

Partiendo del product backlog elaborado en el apartado Funciones del producto del punto 5.1.2 Descripción general, se han elaborado cinco Sprint backlogs, en los cuáles se especifican las actividades o tareas a realizar para las historias de usuario definidas.

Cabe aclarar, dado que este desarrollo no cuenta con una persona que tenga el rol de product owner, tal como lo norma la metodología Scrum, el autor toma este rol además del de scrum master.

La duración del sprint es de dos semanas cada uno y están definidos en las siguientes tablas (tabla 5, tabla 6, tabla 7, tabla 8 y tabla 9):

|  |  |  |
| --- | --- | --- |
| **Product backlog** | **Sprint backlog #1** | |
| Ítem #1  MOSTRAR PUNTOS DE INTERÉS | Ítem #1  Configurar el proyecto en Android para utilizar la librería VISION SDK. | Puntuación |
| 2 |
| Ítem #2  Diseñar las clases para administrar la personalización de la librería. | 8 |
| Ítem #3  Precargar los 17 puntos de interés geolocalizados (títulos, subtítulos, coordenadas y demás datos que solicite la librería) | 3 |
| Ítem #4  Diseñar los estilos e interacción para las vistas de paneles, listas, flechas y radar. | 5 |
| Ítem #5  Prueba de integración de las tareas previas. | 3 |

Tabla 5: Sprint backlog #1

|  |  |  |
| --- | --- | --- |
| **Product backlog** | **Sprint backlog #2** | |
| Ítem #2  DESPLEGAR LA LISTA DE EVENTOS | Ítem #1  Diseñar las clases y actividad que se encargarán de mostrar la lista de eventos por punto de interés. | Puntuación |
| 8 |
| Ítem #2  Prueba de integración de las tareas previas. | 3 |

Tabla 6: Sprint backlog #2

|  |  |  |
| --- | --- | --- |
| **Product backlog** | **Sprint backlog #3** | |
| Ítem #3  DISTINGUIR LOS EVENTOS POR CATEGORÍAS | Ítem #1  Definir las categorías y cómo diferenciarlas en las vistas. | Puntuación |
| 5 |
| Ítem #2  Diseñar las clases para gestionar las categorías. | 5 |
| Ítem #3  Prueba de integración de las tareas previas. | 3 |

Tabla 7: Sprint backlog #3

|  |  |  |
| --- | --- | --- |
| **Product backlog** | **Sprint backlog #4** | |
| Ítem #4  REGISTRAR EVENTOS | Ítem #1  Diseñar el modelo de base de datos para el sistema web. | Puntuación |
| 5 |
| Ítem #2  Securizar el acceso a los recursos del sistema web. | 5 |
| Ítem #3  Definir la arquitectura y flujo de información para registrar eventos por facultad. | 8 |
| Ítem #4  Prueba de integración de las tareas previas. | 3 |

Tabla 8: Sprint backlog #4

|  |  |  |
| --- | --- | --- |
| **Product backlog** | **Sprint backlog #5** | |
| Ítem #5  COMUNICAR EVENTOS REGISTRADOS HACIA LA APLICACIÓN | Ítem #1  Diseñar el servicio web para comunicar los eventos registrados. | Puntuación |
| 5 |
| Ítem #6  ADMINISTRAR LOS EVENTOS REGISTRADOS | Ítem #2  Diseñar la interfaz de usuario que permita administrar los eventos registrados. | 3 |
| Ítem #3  Diseñar el backend que permita administrar los eventos registrados. | 3 |
| Ítem #4  Pruebas de integración de las tareas previas. | 3 |

Tabla 9: Sprint backlog #5

### 5.2.2 Arquitectura de la aplicación

Para tener una mejor comprensión en que el aplicativo móvil fue desarrollado, en la siguiente figura (figura 25) se muestra la arquitectura general de la aplicación.

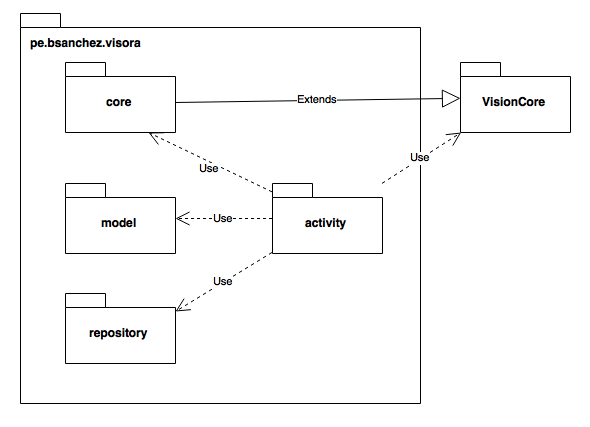


Figura 25: Arquitectura general de la aplicación VisoRA

Esta arquitectura está compuesta de los siguientes paquetes:

#### pe.bsanchez.visora

Este es el paquete principal y dentro del cual se organizan los demás. Este nombre está registrado dentro del aplicativo en el archivo AndroidManifest.xml

#### core

En este paquete se encuentran las clases que permiten programar el funcionamiento principal del aplicativo, éstas extienden de la librería VISION SDK.

#### activity

En este paquete se ubican las clases que permiten gestionar las diferentes vistas del aplicativo.

#### model

Dentro de este paquete se ubican las clases que sirven para mapear las entidades que se utilizan para gestionar la información de los eventos y puntos de interés.

#### repository

Este paquete responde al estándar de arquitectura que permite administrar las diferentes fuentes para obtener los datos que el aplicativo va a mostrar, como son los eventos y las categorías que las agrupan.

### 5.2.3 Diagrama de clases

En la siguiente figura se muestra el diagrama de clases simplificado (figura 26). Éste da una idea general de la distribución de clases en la aplicación, es decir sólo se muestran los atributos y métodos más importantes.

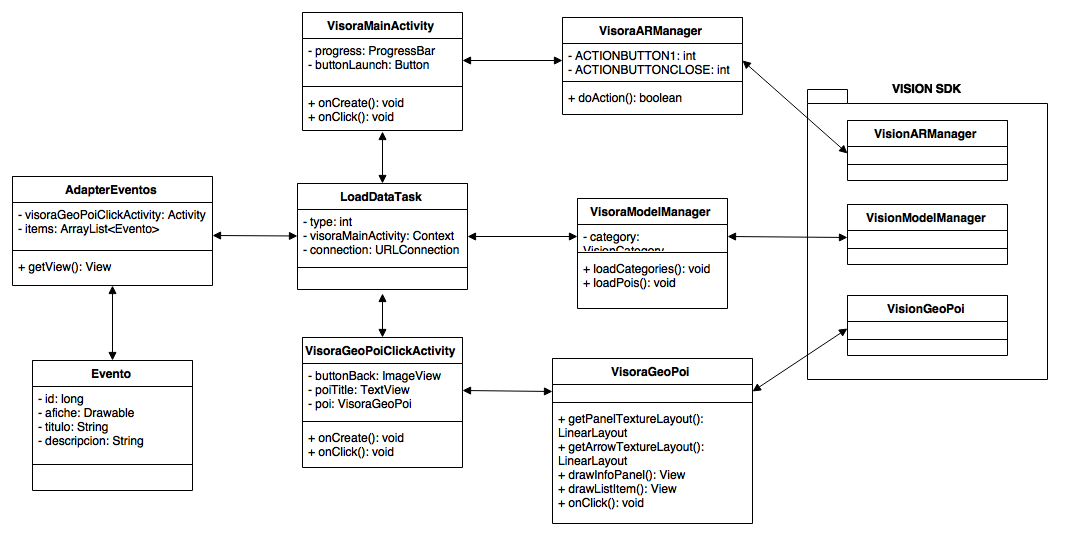


Figura 26: Diagrama de clases de la aplicación

Las clases que heredan de *android.app.Activity* son las que administran el flujo principal de la aplicación, por medio de ellas el aplicativo interactúa con la librería VISION SDK. Además de corresponderse con las interfaces gráficas (o layouts) de inicio y la vista de eventos luego de tocar cualquiera de los puntos de interés.

#### VisoraMainActivity

Esta es la actividad principal del aplicativo, se comunica con el core por medio de las clases que heredan directamente de la librería VISION SDK.

#### VisoraGeoPoiClickActivity

Esta clase gestiona la vista o layout de eventos (figura 21). Luego de realizar un toque sobre cualquiera de las figuras que representen un punto de interés, puede ser paneles flotantes, flechas o ítems en la lista, esta actividad permite la carga de los eventos registrados y para ello utiliza el utilitario *LoadDataTask*.

La personalización en VISION SDK se basa en la creación de nuevas clases que heredan de las existentes, modificando solamente aquellas partes para las que se requiera un nuevo comportamiento y/o apariencia. De esta manera es mucho más sencilla la reutilización de código y se simplifica al máximo la creación de nuevas especializaciones, ya que solamente se modificarán las partes necesarias y no toda la estructura. Por tanto, se puede decir que existen dos puntos principales de intervención para personalizar VISION SDK; cambiar la implementación de los módulos principales asociados al core y extender/modificar la implementación de los objetos de datos.

Se puede apreciar la relación directa de herencia que existe entre la librería VISION SDK y las clases *VisoraARManager*, *VisoraModelManager* y *VisoraGeoPoi*. Éstas permiten la personalización y configuración del motor de realidad aumentada gestionado por la librería.

#### VisoraARManager

Extiende de *VisionARManager* adaptándolo a las nuevas necesidades el comportamiento de las vistas de VISION SDK en la aplicación. Al extender el *VisionARManager* se consigue personalizar el aspecto y acceso a las vistas de realidad aumentada desde la nueva aplicación y hacer que se ajuste a las necesidades concretas del nuevo proyecto.

#### VisoraModelManager

Hereda de *VisionModelManager* adaptándolo a las necesidades de origen de datos de la nueva aplicación. Esta clase personalizada puede establecer una nueva implementación para los métodos que realizan la carga de datos para la realidad aumentada.

#### VisoraGeoPoi

Esta clase amplía la implementación de los puntos de interés para poder personalizar *VisionGeoPoi* y poder añadir nuevos campos de datos y editar elementos visuales asociados a cada elemento en VISION SDK.

Además de las clases *activity* y las que pertenecen al *core* del aplicativo, existen algunas auxiliares que se encargan principalmente de la adquisición de los datos, representación de los mismo y llamadas al sistema operativo para obtener alguna funcionalidad (como lo son la cámara o los datos GPS).

#### LoadDataTask

Esta clase auxiliar fue implementada para poder gestionar la carga de datos y poblar los puntos de interés, también es utilizada para utilizar el servicio web por medio del cual se poblarán los datos para los eventos de las diferentes facultades. Esta clase hereda de *android.os.AsyncTask* para poder ejecutarse de manera asíncrona y evitar el desborde de la DalvikVM.

#### AdapterEventos

Este utilitario permite gestionar la adecuación de los datos en la vista de Eventos. Hereda de *android.widget.BaseAdapter*.

Cabe mencionar que, aunque no se menciona explícitamente la interacción con la librería de Android, se sobrentiende que directa o indirectamente se da.

### 5.2.4 Sistema web – http://visoraserver-bsanchez.rhcloud.com/

Este sistema ha sido diseñado con la finalidad de dar el soporte web en el cual se gestionarán los eventos registrados. Este sistema web tendrá un acceso para un usuario con los permisos para registrar los eventos por facultad. El desarrollo de este sistema está contemplado en los sprint #4 y #5.

En la figura 27 se puede observar la arquitectura general del sistema.

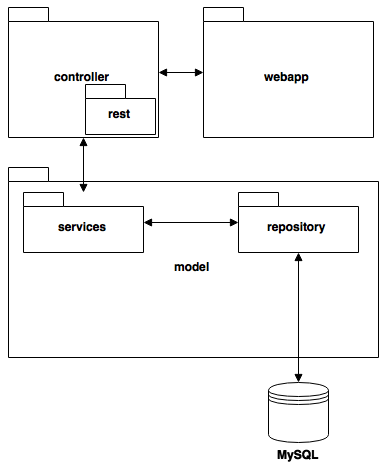


Figura 27: Arquitectura general del sistema *visoraserver*

Como se puede observar, la distribución de los paquetes se corresponde al patrón de arquitectura Modelo – Vista – Controlador (MVC). El flujo de la información está distribuido en tres capas principales.

#### webapp

En este paquete se distribuyen todos los archivos estáticos y diferentes vistas que utilizará el sistema. Esta capa es la presentación del modelo. Las vistas principales son la del login y las de gestión de eventos.

#### controller

Es la capa que reacciona con la petición del cliente, ejecuta la acción adecuada y crea el modelo solicitado. Los controladores que gestionan los servicios web se encuentran dentro del paquete *rest*, se ha utilizado la arquitectura REST para su implementación.

#### model

En esta capa se encuentra el núcleo de la funcionalidad (dominio) del sistema. Es en dónde se encapsula el estado de la aplicación. Tal como lo norma el patrón, es independiente del controlador y la vista. Dentro de la misma se observan la paquetería de servicios (*services*) que se comunican directamente con el paquete de acceso a los datos (*repository*).

En las siguientes figuras se muestran prototipos de las vistas principales del sistema web.

#### Login

Esta vista permite la funcionalidad básica de una aplicación web que tenga securizado el acceso a los recursos que permite gestionar. Cada facultad tendrá asignado un usuario para poder registrar y administrar los eventos.



Figura 28: Login

#### Registro de evento

En esta interfaz el sistema web permite que el usuario autorizado registre los eventos que la facultad quiera promover. Los únicos datos a ingresar son el título del evento, la descripción, fecha de inicio y fecha de término, cargar alguna imagen que haya sido diseñado como afiche y por último seleccionar alguna de las cuatro categorías disponibles que agrupan a los eventos. Estas categorías están distinguidas en por colores tal como se puede ver en la tabla- 10.

|  |  |
| --- | --- |
| **Categoría** | **Color** |
| Cultural |  |
| Deportivo |  |
| Político |  |
| Social |  |

Tabla 10: Categorías y sus correspondientes colores



Figura 29: Registro de evento

#### Lista de eventos registrados+

Esta vista permite al usuario autorizado gestionar los eventos que ya hayan sido registrados. Para la versión actual del sistema web, la administración de los eventos se limita a cambiar de afiches, modificar las fechas de inicio y/o fin y archivar o desarchivar un evento, esto implica que si un evento ha sido registrado, éste puede aún no ser visto desde el aplicativo principal VisoRA si está archivado, si el evento es desarchivado automáticamente aparece como visible dentro de los eventos que se puedan apreciar en VisoRA al tocar el punto de interés que identifique a dicha facultad.

Las categorías no pueden ser modificadas en esta versión de visoraserver.



Figura 30: Lista de eventos registrados

Para la persistencia de los datos se utilizó una base de datos MySQL en su versión5.5. En la figura 31 se ve el modelo de datos diseñado, esta estructura cumple con los requisitos planteados en los dos últimos sprint (tabla 8 y tabla 9).

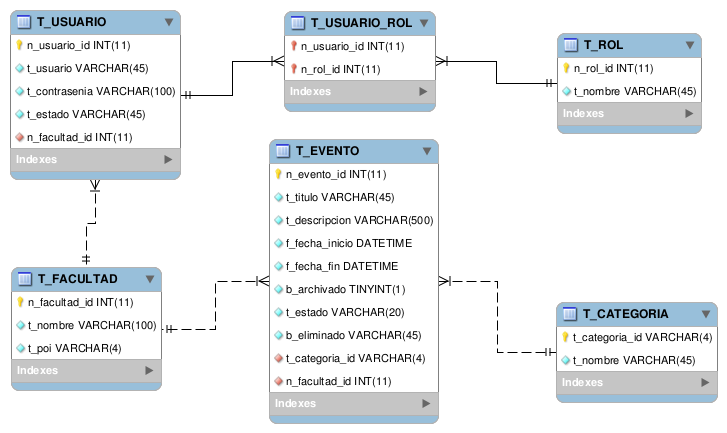


Figura 31: Modelo de datos

Esta estructura soporta la gestión de usuarios por roles. Las tablas T\_EVENTO, T\_FACULTAD y T\_CATEGORIA permiten categorizar los eventos y asignarlos por facultad. Cada facultad cuenta con un usuario encargado del registro de los eventos.

La disponibilidad del sistema debe ser 24/7 dado que es necesario que esté disponible en cualquier momento para que el usuario pueda registrar y administrar los eventos de su facultad. Esto se consiguió utilizando un servicio de la plataforma de productos de OpenShift [25], la Public PaaS que permite la construcción y rápido lanzamiento de sistemas web en la nube de manera pública, todo ello soportado por RedHat.

Si bien es cierto que al ser público no hay pago alguno, siempre que se opte por la opción gratuita, sin embargo ocurre que la disponibilidad de las aplicaciones está limitada al uso de las mismas, esto es, que si la plataforma no registra ningún uso del sistema, la plataforma automáticamente repliega la aplicación. Ello lo hace por ser un servicio gratuito, optimización de recursos en la nube de RedHat y el hecho de no existir ningún tipo de suscripción que defina un ANS.

Por último cabe mencionar, el sistema web <http://visoraserver-bsanchez.rhcloud.com/> está diseñado administrar una fuente común de eventos registrados por facultad.

# CAPÍTULO VI: CONCLUSIONES Y RECOMENDACIONES

En este último capítulo se analizarán los resultados obtenidos en esta tesis, así como las estimaciones y recomendaciones del posible futuro de la aplicación.

## 6.1 Conclusiones

El objetivo principal propuesto era el de desarrollar una aplicación en Android que proporcione una solución alternativa al problema de la eficiente difusión de eventos realizados en la ciudad universitaria, aplicando realidad aumentada basada en geolocalización. Este objetivo se ha cumplido satisfactoriamente, la aplicación resultante cumple con los requisitos fundamentales ya que utiliza la tecnología de realidad para geolocalizar los diferentes puntos de interés definidos en las facultades de la ciudad universitaria. El aplicativo no cuenta, como ha sido demostrado en capítulos previos, únicamente con la vista de realidad aumentada, sino que también lo hace a través de otros elementos visuales personalizados de la librería utilizada VISION SDK.

Uno de los objetivos secundarios era diseñar e implementar un sistema web centralizado en el que las facultades puedan registrar aquellos eventos que vayan a realizar. Este objetivo ha sido cumplido parcialmente, debido al inconveniente presentado con la disponibilidad de la plataforma pública de OpenShift mencionado en el apartado anterior.

El sistema web no cuenta aún con las interfaces de usuario para registro y gestión de eventos. Sin embargo se diseñó e implementó la base de datos y el servicio web para la comunicación de los eventos registrados hacia el aplicativo.

## 6.2 Recomendaciones

Este último apartado está segmentado en títulos que definen, en primer lugar, algunas consideraciones para implementaciones futuras y, seguidamente, recomendaciones en base a las lecciones aprendidas a lo largo del desarrollo de esta tesis.

### 6.2.1 Comunidades VisoRA

La escalabilidad y versatilidad de poder personalizar la librería VISION SDK puede permitir diseñar e implementar la capacidad de poder crear y compartir puntos de interés agrupados en categorías de eventos. Por ejemplo; un grupo de amigos podría, en función a gustos en común, crear una categoría agrupando puntos de interés que designen karaokes de música criolla. El sistema web permitiría crear estos nuevos puntos de interés agrupados y el aplicativo en Android les permitiría visualizar, en sus diferentes vistas, estos puntos de interés compartidos. Creando comunidades que tengan puntos de interés en común.

### 6.2.2 Paneles flotantes más informativos

Estos elementos visuales son el principal llamativo del uso de realidad aumentada en la aplicación, en la versión actual sólo presentan información que indica la categoría de los eventos registrados para el punto de interés, su nombre y la distancia al mismo con respecto al usuario. Además podría adicionarse un archivo audiovisual para enriquecer la información propia del punto de interés y que sea accedido luego de realizar algún gesto en particular sobre el panel flotante. Esto para diferenciarlo del toque necesario para poder visualizar la lista de eventos, uno de los motivos principales del desarrollo del aplicativo.

### 6.2.3 Product owner, scrum master y development team

Este punto de vista es muy particular y deriva de mi experiencia al realizar este desarrollo bajo la metodología ágil Scrum. Si bien algunas dinámicas de la metodología han sido obviadas dado la limitante del número de integrantes, una persona, la gestión de roles que propone Scrum no me resultó del todo cómoda al momento de ser el cliente, el gestor y el desarrollador, dado que hacía de juez y parte. Sin embargo, podría haber hecho un rol mixto en el que encaje un desarrollo bajo Scrum unipersonal.

### 6.2.4 Pago por disponibilidad

Esta es una lección aprendida a mitad de camino de la implementación del sistema web visoraserver. Como bien se explicó en el apartado 5.2.4 Sistema web – <http://visoraserver-bsanchez.rhcloud.com/>, la disponibilidad que se requería para el sistema y el servicio web es de 24/7 para comodidad del usuario. Sin embargo, luego de un día de no haber ingresado a visoraserver, la misma plataforma la dejaba no disponible, lo cual podría generar muchos inconvenientes al momento de querer registrar o gestionar un evento. No había presupuesto designado para el pago de una suscripción y poder adquirir un servicio de mayor disponibilidad.

En conclusión, siempre evaluar contar con una cantidad de dinero a presupuestar para cubrir cualquier contratiempo que exija algún tipo de pago, y así no retrasar o limitar el alcance de un proyecto similar.

# REFERENCIAS BIBLIOGRÁFICAS

[1] Social Media, Empresas & TIC - Geolocalización + Realidad Aumentada = Trabajo y Negocio

[en línea, 15 de noviembre de 2014]

<http://www.socialetic.com/geolocalizacion-realidad-aumentada-trabajo-y-negocio.html>

[2] Research2Guidance, The mobile research specialist - Apple and Android account for 82% of all app downloads but 2013 might be the turning point of the dominance for both platforms

[en línea, 12 de noviembre de 2014]

<http://research2guidance.com/apple-and-android-account-for-82-of-all-app-downloads-but-2013-might-be-the-turning-point-of-the-dominance-for-both-platforms/>

[3] Höllerer, Tobias H. & Feiner, Steven K. (2004), **Telegeoinformatics: Location-Based Computing and Services**, Editorial Taylor & Francis Books Ltd. Capítulo Nueve - Mobile Augmented Reality

[4] Drake, Joshua J. & Fora, Pau Oliva (2014), **AndroidTM Hacker’s Handbook**, Editorial John Wiley & Sons Inc., Págs. 1 – 4

[5] Muy Computer - Un trocito de historia: evolución y “muerte” de los PDA

[en línea, 12 de noviembre de 2014]

<http://www.muycomputer.com/2014/07/06/un-trocito-de-historia-evolucion-y-muerte-de-las-pda>

[6] About Technology - What makes a Smartphone Smart? by Lianne Cassavoy

[en línea, 15 de noviembre de 2014]

<http://cellphones.about.com/od/smartphonebasics/a/what_is_smart.htm>

[7] Department of Computer Science - George Fitzmaurice, Chameleon Spatially Aware Display

[en línea, 20 de noviembre de 2014]

<http://www.dgp.toronto.edu/~gf/Research/Chameleon/ChameleonResearch.htm>

[8] Computer Graphics and Users Interface Labs, Columbia University - The Touring Machine [en línea, 22 de noviembre de 2014] <http://monet.cs.columbia.edu/projects/mars/touring.html>

[9] Open Handset Alliance

[en línea, 22 de noviembre de 2014]

<http://www.openhandsetalliance.com/>

[10] Ubiquitous Computing by Mark Weiser (1993)

[en línea, 07 de diciembre de 2014]

<http://www.ubiq.com/hypertext/weiser/UbiCACM.html>

[11] Ishii, H. & Ullmer, B. (1997), **Tangible Bits: Towards Seamless Interfaces between People, Bits and Atoms**, Editorial ACM.

[12] William Noah Shilit (1995), **A system architecture for context-aware mobile computing**, Editorial Columbia University.

[13] Pardo de Donlebún Matilla, Elías (2012), **Desarrollo de una Aplicación con Geolocalización para Android,** e-archivo.uc3m.es.

[14] Nielsen Havsager, Martin (2013), **memoAR - An Augmented Reality Application on Urban Story Telling**, Aalborg University.

[15] Uijtdewilligen, Freek (2010), **A framework for context-aware applications using augmented reality: A train station navigation proof-of-concept on Google Android**

[16] Mountain Goat Software - Scrum

[en línea, 10 de diciembre de 2014]

<http://www.mountaingoatsoftware.com/agile/scrum>

[17] XPrograming.com An Agile Software Development Resource - What is Extreme Programming?

[en línea, 10 de diciembre de 2014]

<http://xprogramming.com/xpmag/whatisxp>

[18] Site historique de la méthode RAD - RAD La Méthode

[en línea, 10 de diciembre de 2014]

<http://www.rad.fr/>

[19] Beth Chrissis, Mary & Konrad, Mike (2009), **CMMI® Guía para la integración de procesos y la mejora de productos**, Editorial Pearson Educación, Págs. 53 – 54

[20] Harvard Business Review

[en línea, 31 de mayo de 2015]

<https://hbr.org/1986/01/the-new-new-product-development-game>

[21] My Articles On Scrum etc.

[en línea, 31 de mayo de 2015]

<http://www.controlchaos.com/storage/scrum-articles/Scrum%20and%20The%20Perfect%20Storm.pdf>

[22] Henrik Knitberg (2015), **Scrum and XP from the Trenches 2nd Edition**, C4Media – Publisher of InfoQ.com

[23] ©2013 Project Management Institute, **PMBOK Guide Fifth Edition**, Págs. 344 - 345

[24] Especificación de Requisitos según el estándar de IEEE 830

[en línea, 24 de junio de 2015]

<https://www.fdi.ucm.es/profesor/gmendez/docs/is0809/ieee830.pdf>

[25] OpenShift Products

[en línea, 14 de julio de 2015]

<https://www.openshift.com/products>