2014

UNIVERSIDAD DE LAS PALMAS DE GRAN CANARIA
Escuela de Ingeniería Informática

DISEÑO E IMPLEMENTACIÓN DE UN CONVERTIDOR NUMÉRICO COMO APLICACIÓN ANDROID

GRADO EN INGENIERÍA INFORMÁTICA

AUTOR: ARMANDO PÉREZ GONZÁLEZ
TUTOR: FRANCISCO JAVIER CARRERAS RIUDAVETS

UNIVERSIDAD DE LAS PALMAS DE GRAN CANARIA

TRABAJO DE FIN DE GRADO

ESCUELA DE INGENIERÍA INFORMÁTICA

UNIVERSIDAD DE LAS PALMAS DE GRAN CANARIA

AUTOR: ARMANDO PÉREZ GONZÁLEZ

TUTOR: FRANCISCO JAVIER CARRERAS RIUDAVETS

TÍTULO: DISEÑO E IMPLEMENTACIÓN DE UN CONVERTIDOR NUMÉRICO COMO APLICACIÓN ANDROID.

FECHA: 15/12/2014

LUGAR: ADMINISTRACIÓN DE LA ESCUELA DE INGENIERÍA INFORMÁTICA DE LA UNIVERSIDAD DE LAS PALMAS DE GRAN CANARIA

ÍNDICE DE CONTENIDO

1 – INTRODUCCIÓN

Android es un sistema operativo basado en el kernel de Linux diseñado principalmente para dispositivos móviles con pantalla táctil, como teléfonos inteligentes o tabletas. El SO fue inicialmente desarrollado por Android Inc., empresa a la que Google respaldó económicamente y más tarde compró en 2005. Android fue presentado en 2007 junto a la fundación del Open Handset Alliance: un consorcio de compañías de hardware, software y telecomunicaciones para avanzar en los estándares abiertos de los dispositivos móviles. El primer móvil con el sistema operativo Android fue el HTC Dream y se vendió en octubre de 2008.

Android como SO, basa gran parte de su correcto funcionamiento en las llamadas apps. Dichas apps no son otra cosa que un software desarrollado para usarse en plataformas Android. Las apps de Android están disponibles en el Android Market o Google Play Store, en el Amazon Appstore y en diversos sitios alrededor del mundo.

El Desarrollo de apps para Android se hace habitualmente con el lenguaje de programación Java y el conjunto de herramientas de desarrollo (SDK, Software Development Kit), pero hay otras opciones disponibles. Entre dichas opciones se encuentran:

- Native Development Kit

- Android Open Accessory Development Kit

- App Inventor para Android

- HyperNext Android Creator

- Appcelerator Titanium

El proyecto que nos concierne en este TFG ha surgido para solucionar una problemática que se estaba encontrando el Instituto Universitario de Análisis y Aplicaciones Textuales (en adelante IATEXT). La problemática consiste en que varios productos de este instituto solo contaban con acceso web, dificultando las vías de accesos a potenciales usuarios. Este proyecto pretende facilitar y mejorar el acceso a toda la información referente al

Servicio Web Números TIP, de manera tal que los usuarios no solo puedan acceder desde la web, sino desde una aplicación para terminales Android creada para esto específicamente.

Para dar solución a esta problemática nos hemos dado a la tarea de crear una aplicación para terminales Android, cuya principal funcionalidad será conectarse al Servicio Web Números TIP y realizar una petición. La respuesta a dicha petición será mostrada en el terminal móvil del usuario siguiendo unos patrones predefinidos por los tutores y logrando de esta manera que el usuario final pueda tener acceso a dicho servicio de una forma mucho más fácil y agradable a la vista.

1.1 – OBJETIVOS Y MOTIVACIÓN

Las Empresas o Grupos de Trabajo tienden, de manera natural, a maximizar la eficiencia de sus recursos y sacarles el máximo partido. En conjunción con la creciente tendencia del uso de Smartphones, desarrollar aplicaciones móviles para servicios web se convierte en una estrategia que puede aportar numerosas ventajas.

El principal objetivo de este proyecto es proporcionarle al usuario de terminales móviles con sistema operativo Android una novedosa forma de acceso al Servicio Web Números TIP. Esto es necesario porque el navegador web tradicional, en un terminal móvil, no suele ser de fácil uso. Dicha forma será una aplicación Android que implemente las mismas funcionalidades que ofrece el servicio Números TIP mediante su acceso web.

Otro objetivo buscado es el fortalecimiento de la marca. Esto es una de las principales ventajas o beneficios de desarrollar aplicaciones móviles propias. Además, tener presencia en los dispositivos móviles posiciona mucho mejor la marca, dado que los clientes y usuarios pueden consultar la aplicación en cualquier parte, especialmente en el tiempo libre o en desplazamientos largos. La posibilidad de sincronización de la aplicación con las redes sociales mejora la difusión y viralización de contenidos, lo cual

se convierte en otra gran ventaja. Los propios usuarios se encargarán mediante las redes de darte a conocer a sus amigos y ampliar así el abanico de usuarios potenciales.

Como objetivo secundario se busca estandarizar el diseño para la posterior creación de aplicaciones Android que implementen el acceso de los demás productos del grupo IATEXT.

Como motivación personal decidí realizar este proyecto para lograr un mayor dominio de la programación para terminales Android, ya que es un mundo que te abre muchas posibilidades de futuro y está en constante evolución. Además de lo antes mencionado, dicho proyecto me permitiría solidificar mis conocimientos del lenguaje de programación Java y me introduciría en el mundo de los servicios web, los cuales tienen gran uso actualmente y cuentan con un alto grado de popularidad entre los usuarios finales.

1.2 – APORTACIONES

En la actualidad, el producto Números TIP desarrollado por el Grupo IATEXT de la Universidad de Las Palmas de Gran Canaria solo consta de acceso mediante la web. Esto supone un contratiempo en la utilización de dicho producto, ya que gran parte del acceso a la información en la sociedad actual se realiza a través de los Smartphones, apoyándose en aplicaciones desarrolladas para el sistema operativo del terminal móvil del usuario.

El uso de esta aplicación aporta la comodidad de automatizar el proceso de acceso al Servicio Web Números TIP, afectando positivamente la economía de los usuarios finales ya que consumen menos tarifa de datos que accediendo mediante la web desde el móvil. Además de lo antes expuesto ahorra tiempo porque tiene una interfaz de fácil uso y donde todos los datos aparecen distribuidos de forma entendible y de fácil acceso.

Otra gran ventaja, para el cliente sobre todo, es la de una comunicación fluida y sin limitaciones. Es decir, con la aplicación móvil el cliente tiene acceso a contactar con la empresa en cualquier momento que lo necesite, sin tener que esperar a llamar por teléfono

durante los horarios de atención y sin que le cueste un céntimo. En definitiva aumentar la eficiencia y reducir costes.

El proyecto está estructurado de forma que se pueden añadir o integrar nuevas funcionalidades si se desease. Además, facilita el acceso, desde terminales móviles, a uno de los productos del IATEXT, incrementando y expandiendo de esta manera su uso.

Si la aplicación cumple con el nivel de calidad exigido se publicará en el Play Store de manera gratuita para su posterior descarga.

1.3 – ESTRUCTURA DEL DOCUMENTO

Con el objetivo de facilitar la lectura y comprensión del documento se presenta en este epígrafe su estructura general, aspectos y características más relevantes.

Tras haber introducido el Trabajo y haber destacado los principales objetivos del mismo, se desarrolla a continuación el apartado de aportaciones. Luego se desarrollan cada una de las fases. En primer lugar se presentan los estudios realizados acerca del estado del arte, es decir, las herramientas que realizan funciones similares a la que desarrollaremos.

Posteriormente, se detallan las herramientas y recursos utilizados, tanto hardware como software, para desarrollar la aplicación y su posterior prueba y despliegue. En el siguiente capítulo se detallan las metodologías que nos guiaron en el proceso de desarrollo a lo largo del ciclo de vida del mismo, así como la planificación, la temporización y el presupuesto del proyecto.

En el capítulo 5 y 6 se detallan el análisis, diseño y desarrollo de la aplicación, de forma exhaustiva, para finalizar el documento con los resultados y conclusiones obtenidas, junto con el posible trabajo futuro que podría realizarse. Además se indica la bibliografía consultada.

En cada una de las fases presentadas en este documento se podrán encontrar diferentes apartados, organizados de manera que facilite la lectura y compresión de la misma.

1.4 – COMPETENCIAS ESPECÍFICAS CUBIERTAS

Con el desarrollo de este Trabajo de Fin de Grado de la carrera Grado en Ingeniería Informática, se deben cubrir las competencias asignadas a éste, las cuales son: CII01, CII02, CII04, CII018 y TFG01. A continuación, se listan cada una de ellas junto con una explicación de cómo se han cubierto.

CII01: Capacidad para diseñar, desarrollar, seleccionar y evaluar aplicaciones y sistemas informáticos, asegurando su fiabilidad, seguridad y calidad, conforme a principios éticos y a la legislación y normativa vigente.

Este trabajo consistía en diseñar y desarrollar una aplicación Android para solucionar la problemática planteada. También nos enfrentamos a diferentes procesos de toma de decisión, sobre todo a la hora de seleccionar el entorno de programación, las herramientas de desarrollo, etc. También tuvimos muy en cuenta, durante todo el proceso de desarrollo que se cumplieran todas las legislaciones y normativas vigentes.

CII02: capacidad para planificar, concebir, desplegar y dirigir proyectos, servicios y sistemas informáticos en todos los ámbitos, liderando su puesta en marcha y su mejora continua y valorando su impacto económico y social.

Para lograr desarrollar esta aplicación se realizó una correcta planificación y dirección de todo el proceso de desarrollo del proyecto. El mismo consistió en el desarrollo de una aplicación Android que se desplego en varios terminales móviles de usuarios. Además, fuimos capaces de identificar limitaciones y posibles mejoras en el futuro cercano, para, de esta manera lograr aumentar la presencia del grupo IATEXT en el mundo de las aplicaciones móviles.

CII04: capacidad para elaborar el pliego de condiciones técnicas de una instalación informática que cumpla los estándares y normativas vigentes.

En el desarrollo de este documento se incluyeron en un apartado los requisitos de hardware y software necesarios para el correcto desarrollo de esta aplicación y para su despliegue. Además, se especificaron todas las licencias de las herramientas utilizadas y el marco legal en que se ampara el presente proyecto.

CII18: conocimiento de la normativa y la regulación de la informática en los ámbitos nacional, europeo e internacional.

Para el desarrollo de este o cualquier proyecto, los desarrolladores deben estar al tanto de lo legal y conocer la normativa y la regulación de la informática en el ámbito nacional, de la Unión Europea, e internacional.

TFG01: ejercicio original a realizar individualmente y presentar y defender ante un tribunal universitario, consistente en un proyecto en el ámbito de las tecnologías específicas de la Ingeniería en Informática de naturaleza profesional en el que se sinteticen e integren las competencias adquiridas en las enseñanzas.

Luego de haber terminado el desarrollo del Trabajo de Fin de Grado, se debe hacer una presentación y defensa del mismo, ante el tutor y un tribunal constituidos por miembros del claustro de profesores, para cumplir con lo establecido con el TFG01. De esta manera se demuestra la consistencia en el ámbito de las tecnologías específicas de la Ingeniería Informática, entre las cuales se destacan la captura de requerimientos, análisis y diseño del software, así como la implementación, pruebas, manejo de base de datos, administración de servidores y diseños de interfaces de usuarios. Todos estos conocimientos han sido adquiridos durante la formación que se obtuvo a lo largo de la carrera Grado en Ingeniería Informática.

2 – ESTADO DEL ARTE

En este capítulo de la memoria se realizará un análisis del estado actual del mercado donde queremos introducirnos mediante este trabajo. También analizamos a grosso modo el estado de los portales web más relevantes que prestan servicios parecidos o similares al nuestro, es decir, traducir los números de su forma numérica a su forma cardinal, ordinal, etc.

Comenzaremos hablando del estado actual de las aplicaciones para terminales móviles Android. Luego de realizar un exhaustivo estudio y varias pruebas, podemos afirmar que en estos momentos no existe en el Play Store (Android Market) ninguna aplicación que siquiera se acerque al nivel de detalles que proporciona la aplicación creada con este trabajo. Solo hemos encontrado pequeñas aplicaciones que traducen el número a su forma cardinal y todas en inglés. Además de lo antes mencionado hemos encontrado pequeñas aplicaciones didácticas que en su mayoría van dirigidas a enseñar a escribir los números a niños pequeños. Con esta aplicación estamos introduciendo un producto completamente nuevo en el mercado, que creemos será de gran ayuda para el usuario final. Dicho usuario final puede ser desde una persona que hable otra lengua hasta una maestra que quiera enseñar a sus estudiantes las diferentes formas de pronunciar un número.

Por su parte, existen algunos portales webs que prestan servicios parecidos al nuestro y podríamos comenzar comentando que ha sido imposible encontrar un portal que preste un servicio tan completo como el nuestro. Hemos realizado un estudio a fondo de los portales y la gran mayoría de ellos ofrecen solamente la función de traducir un número a su forma cardinal. Por ejemplo los portales http://nosetup.org/ y http://numeros-a-letras.todala.info/ solo son capaces de convertir el número introducido a su forma cardinal. Estos sitios tienen su utilidad aunque bastante limitada, sobre todo, desde el punto de vista didáctico. Solamente hemos encontrado un portal que va un poco más allá, y además de transformar el número a su forma cardinal, también lo hace a su forma ordinal y partitiva. El portal http://lenguaje.com/cgi-bin/motnum.exe?T1=22&S1=1&B1=Convertir convierte, como habíamos comentado

antes, el número a su forma ordinal y partitiva además de la cardinal. Dicho portal también es capaz de realizar la operación inversa, es decir, transformar un número en su forma cardinal, ordinal o partitiva a su forma numérica.

2.1 – APLICACIONES ANDROID

En este apartado analizaremos las aplicaciones encontradas en el Play Store que realicen una función similar a la nuestra. Se tomará en cuenta diferentes factores para el análisis como por ejemplo el nivel de detalle que ofrezcan en sus respuestas, si son multilenguaje o no, si ofrecen ejemplos de uso, etc.

2.1.1 – SPELL A NUMBER

Dibujo 1 - Interfaz de la app Spell a Number

Aspectos positivos:

- Permite obtener la forma cardinal del número introducido.

- No permite introducir otra cosa que no sea un número, evitando de esta manera los posibles errores de escritura.

Aspectos negativos:

- No devuelve otras formas del número introducido, como por ejemplo la ordinal.

- No devuelve ejemplos de uso, ni notas.

- No permite introducir un número romano.

- Produce errores de traducción cuando se le introduce números grandes

2.1.2 – CHEQUES: NÚMEROS

Dibujo 2 - Interfaz de la app Números a Palabras

Aspectos positivos:

- Permite obtener la forma cardinal del número introducido en diferentes idiomas.

- No permite introducir otra cosa que no sea un número, evitando de esta manera los posibles errores de escritura.

Aspectos negativos:

- No devuelve otras formas del número introducido, como por ejemplo la ordinal.

- No devuelve ejemplos de uso, ni notas.

- No permite introducir un número romano.

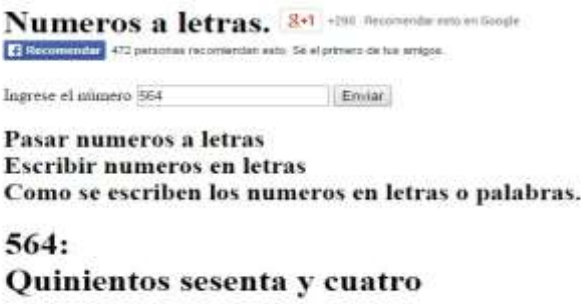
- El margen de números para traducir es relativamente pequeño, solamente 12 cifras.

Después de analizar detenidamente cada una de las aplicaciones anteriores llegamos a la conclusión que casi todas, desde el punto de vista de la UI son bastante pobres, además de que solo obtienes la forma cardinal del número con todas ellas, si bien es cierto que una de ellas proporciona dicha forma en tres idiomas diferentes. Ninguna de ellas explica el funcionamiento de la aplicación. También llegamos a la conclusión de que algunas tienen errores de programación y otras poseen un rango de traducción muy pequeño. Ya para finalizar, una de las aplicaciones posee el multilenguaje, de manera que sería de gran utilidad para una persona que no hable nuestra lengua.

2.2 – SITIOS WEB

En este apartado se analizarán en sentido general las aplicaciones encontradas que permiten obtener algún tipo de conversión de números en cifras a su forma cardinal, ordinal u otras. También tendremos en cuenta aspectos relevantes como por ejemplo si dichas aplicaciones están desarrolladas como aplicaciones web o aplicaciones finales. Se explicará cuáles de ellas o si alguna de ellas aportan ejemplos o notas de cómo deben ser usadas. En fin, se realizará un estudio exponiendo características positivas y limitaciones de cada una de ellas y se distinguirá cuales son multilenguaje o mono lenguaje.

2.2.1 – NÚMEROS A LETRAS



Dibujo 3 - Interfaz del portal web Números a Letras

Esta aplicación web permite introducir un número y obtenerlo en su forma cardinal. Como se puede observar es una aplicación muy básica y solamente devuelve el cardinal del número, nada de ejemplos de uso o notas alternativas.

Aspectos positivos:

- Permite obtener la forma cardinal del número introducido.

Aspectos negativos:

- No devuelve otras formas del número introducido, como por ejemplo la ordinal.
- No devuelve ejemplos de uso, ni notas.
- No indica si existe un error en el número introducido.
- No permite obtener la forma numérica de un número romano introducido.
- Existen errores a la hora de traducir números de gran tamaño.

2.2.2 – NÚMERO A TEXTO

Dibujo 4 - Interfaz del portal web Número a Texto

Esta aplicación es muy similar a la anterior. Devuelve el número en su forma cardinal. También es capaz de devolver un número romano en su forma numérica. Como la anterior es bastante pobre desde el punto de vista de la interfaz y tampoco es capaz de reconocer un error en el número introducido, aunque si transforma el número que encuentre en la expresión dada.

Aspectos positivos:

- Permite obtener la forma cardinal del número introducido.

- Permite obtener la forma numérica de un número romano introducido.

Aspectos negativos:

- No devuelve otras formas del número introducido, como por ejemplo la ordinal.

- No devuelve ejemplos de uso, ni notas alternativas.

- Traduce cualquier número que encuentre en la cadena de caracteres introducida y no indica si existe un error.

2.2.3 – MOTOR NUMÉRICO

Dibujo 5 - Interfaz del portal web Motor Numérico

Esta aplicación, como podemos observar, está un poco más desarrollada desde el punto de vista de la interfaz de usuario. Además de lo antes mencionado es capaz de devolver varias formas de un número como por ejemplo la forma cardinal u ordinal. También tiene tratamiento de errores y además, en la misma página, una pequeña explicación de cómo funciona y que hace

Aspectos positivos:

- Permite obtener varias formas del número introducido (cardinal, ordinal, partitiva, multiplicativa).

- Permite saber si existe un error en el número introducido y corregirlo.

Aspectos negativos:

- No devuelve otras formas del número introducido, como por ejemplo la colectiva, poliedra.

- No devuelve ejemplos de uso, ni notas alternativas.

- No permite obtener la forma numérica de un número romano introducido.

- El rango de números a traducir es muy pequeño, solo hasta 4000 millones.

Después de analizar detenidamente cada una de las aplicaciones anteriores llegamos a la conclusión que casi todas, desde el punto de vista de la UI son bastante pobres, además de que solo una de ellas explica el funcionamiento de la aplicación. Casi todas solamente devuelven la forma cardinal del número, excepto la última, que devuelve tres formas más. Además, solo la última realiza un tratamiento de errores, las demás no son capaces de detectar uno, e incluso algunas de ellas traducen números introducidos en cadenas erróneas. También llegamos a la conclusión de que algunas tienen errores de programación y otras poseen un rango de traducción muy pequeño. Ya para finalizar, ninguna de las aplicaciones posee el multilenguaje, de manera tal que una persona que no hable nuestra lengua lo tendrá difícil para poder interactuar con ellas.

3 – HERRAMIENTAS

En este capítulo se describen los recursos utilizados para el desarrollo de la aplicación y los recursos mínimos para el correcto funcionamiento de la misma.

3.1 – RECURSOS DE SOFTWARE

ECLIPSE

Eclipse es un entorno de desarrollo integrado (IDE por sus siglas en ingles). Contiene inicialmente un entorno de desarrollo básico para desarrollar en Java, el cual se puede enriquecer mediante un sistema de plug-in extensible. Dicho sistema permite personalizar el entorno de desarrollo. Está escrito en su mayoría en Java y es utilizado mayormente para el desarrollo de aplicaciones de cliente enriquecido. Es muy útil ya que, mediante el sistema de plug-in, permite desarrollar aplicaciones en otros lenguajes de programación que no sean Java (C, C++, PHP, Ruby, Haskell, etc.).

Eclipse fue desarrollado originalmente por IBM como el sucesor de su familia de herramientas para VisualAge. Eclipse es ahora desarrollado por la Fundación Eclipse, una organización independiente sin ánimo de lucro que fomenta una comunidad de código abierto y un conjunto de productos complementarios, capacidades y servicios.

ECLIPSE ADT (ANDROID DEVELOPMENT TOOLS)

Android Development Tools (ADT) es un plugin para el IDE de Eclipse que está diseñado para proporcionar un entorno integrado donde desarrollar y construir aplicaciones Android. El ADT extiende las capacidades de Eclipse y de esta forma permite a los desarrolladores crear nuevos proyectos para la plataforma Android. Además de crear dichos proyectos también crea la interfaz de usuario de las aplicaciones, agrega

extensiones para el framework de Android usado, depura aplicaciones usando el SDK Tools y exporta ficheros .apk para lograr distribuir las aplicaciones. Es un programa gratuito disponible para su descarga.

ANDROID SDK

El SDK (Software Development Kit) de Android, incluye un conjunto de herramientas de desarrollo. Comprende un depurador de código, biblioteca, un simulador de teléfono basado en QEMU, documentación, ejemplos de código y tutoriales. Las plataformas de desarrollo soportadas incluyen Linux (cualquier distribución moderna), Mac OS X 10.4.9 o posterior, y Windows XP o posterior. La plataforma integral de desarrollo (IDE, Integrated Development Environment) soportada oficialmente es Eclipse junto con el complemento ADT (Android Development Tools plugin), aunque también puede utilizarse un editor de texto para escribir ficheros Java y XML y utilizar comandos en un terminal (se necesitan los paquetes JDK, Java Development Kit y Apache Ant) para crear y depurar aplicaciones. Además, pueden controlarse dispositivos Android que estén conectados (es decir, reiniciarlos, instalar aplicaciones en remoto, etc.).

Las Actualizaciones del SDK están coordinadas con el desarrollo general de Android. El SDK soporta también versiones antiguas de Android, por si los programadores necesitan instalar aplicaciones en dispositivos ya obsoletos o más antiguos. Las herramientas de desarrollo son componentes descargables, de modo que una vez instalada la última versión, pueden instalarse versiones anteriores y hacer pruebas de compatibilidad.

JAVA DEVELOPMENT KIT (JDK)

Java Development Kit (JDK) es una implementación de la Plataforma de Java, ya sea para la edición estándar, la edición para empresas o la edición micro. Fue publicado por la Corporación Oracle en forma de producto binario dirigido a los desarrolladores de Java en Solaris, Linux, Mac OS X o Windows. El JDK incluye una Máquina Virtual de Java (JVM) entre otros recursos para ser el recipiente final de aplicaciones desarrolladas en Java. Desde su introducción en el mercado ha sido, de lejos, el SDK más utilizado. En noviembre de 2006 sus creadores anunciaron que sería publicado bajo la licencia GNU, convirtiéndose así en software libre.

APP INVENTOR FOR ANDROID

Google anunció en julio de 2010 la disponibilidad de App Inventor para Android, que es un entorno de desarrollo visual Web, para programadores noveles, basado en la biblioteca Open Blocks Java, del MIT. Este entorno proporciona acceso a funciones GPS, acelerómetro y datos de orientación, funciones de teléfono, mensajes de texto, conversión habla a texto, datos de contacto, almacenamiento permanente, y servicios Web, incluyendo inicialmente Amazon y Twitter. Hal Abelson, director de proyecto en el MIT, dijo: "Sólo hemos podido hacerlo porque la arquitectura Android es tan abierta". Después de un año de desarrollo, la herramienta de edición de bloques se ha utilizado para enseñanza a principiantes en ciencias de computación en Harvard, MIT, Wellesley, y en la Universidad de San Francisco, donde el profesor David Wolber, desarrolló un curso de introducción a la ciencia de los ordenadores y un libro de enseñanza para estudiantes que no estudian computación, basado en App Inventor para Android.

LIBRERÍA K-SOAP

El proyecto K-SOAP para Android proporciona una ligera y eficiente librería para los clientes de la plataforma Android que necesiten conectarse a un servicio web. Se ha probado mayormente en Android pero puede trabajar perfectamente en otras plataformas que usen librerías de Java. Actualmente, K-SOAP se sigue mejorando y enriqueciendo con más funcionalidades.

3.2 – LENGUAJES DE PROGRAMACIÓN

JAVA

Java es un lenguaje de programación de propósito general, concurrente, orientado a objetos y basado en clases que fue diseñado específicamente para tener tan pocas dependencias de implementación como fuera posible. Su intención es permitir que los desarrolladores de aplicaciones escriban el programa una vez y lo ejecuten en cualquier dispositivo (conocido en inglés como WORA, o "write once, run anywhere"),

lo que quiere decir que el código que es ejecutado en una plataforma no tiene que ser recompilado para correr en otra. Java es, a partir de 2012, uno de los lenguajes de programación más populares en uso, particularmente para aplicaciones de cliente-servidor de web, con unos 10 millones de usuarios reportados.

El lenguaje de programación Java fue originalmente desarrollado por James Gosling de Sun Microsystems (la cual fue adquirida por la compañía Oracle) y publicado en 1995 como un componente fundamental de la plataforma Java de Sun Microsystems. Su sintaxis deriva en gran medida de C y C++, pero tiene menos utilidades de bajo nivel que cualquiera de ellos. Las aplicaciones de Java son generalmente compiladas a bytecode (clase Java) que puede ejecutarse en cualquier máquina virtual Java (JVM) sin importar la arquitectura de la computadora subyacente.

Este lenguaje se utilizó para desarrollar todo el código fuente de la aplicación. Toda la conexión al servicio web, el tratamiento de cadenas y la parte dinámica de la interfaz se lograron con este lenguaje. Es el lenguaje más popular para desarrollar en Eclipse.

UML

El Lenguaje Unificado de Modelado, UML por sus siglas en inglés, es un lenguaje gráfico para visualizar, especificar, construir y documentar un sistema. Está estandarizado por ISO (ISO/IEC 19501:2005). Su versión actual es la 2.5. Es muy útil en los sistemas software, como en este caso, para describir la interacción entre las clases que componen el mismo.

Este lenguaje se utilizó para realizar todo el diseño de clases y todos los diagramas, como por ejemplo el de secuencia.

XML

El lenguaje de marcas extensible busca dar solución al problema de expresar información estructurada de la manera más abstracta y reutilizable posible. Que la información sea estructurada quiere decir que se compone de partes bien identificadas, y que esas partes se componen a su vez de otras partes. Una etiqueta XML consiste en una marca hecha en el documento, que señala una porción de éste como un elemento. Un pedazo de

información con un sentido claro y definido. Las etiquetas tienen la forma <nombre>, donde nombre es el nombre del elemento que se está señalando.

Esta tecnología fue usada para describir los datos que se intercambian entre el servidor y el cliente, con ello se facilita la realización de declaraciones de contenido y la obtención de resultados en varias plataformas. Además los archivos de recursos están basado en XML se utilizan para almacenar cadenas de la interfaz de usuario que deben ser traducidas a otros idiomas.

3.3 – RECURSOS DE HARDWARE

El proyecto fue desarrollado usando un ordenar personal marca HP con las siguientes características:

- Procesador Intel Core i3-3240; 3,40 GHz.

- Memoria RAM 8,0 GB DDR3.

- Pantalla de 22 pulgadas.

- Sistema Operativo: Windows 7 Home Profesional 64 bits.

Además se utilizó un móvil marca Huawei P6-U06 con las siguientes características:

- Procesador ARM Cortex-A9 Quad-Core 1500MHz 32 bits.

- Memoria RAM 2,0 GB.

- Pantalla de 4,7 pulgadas.

- Sistema Operativo: Android 4.4 KitKat.

4 – METODOLOGÍAS Y PATRONES

El mundo de desarrollo de software está en constante evolución. Hace poco más de una década se comenzó a introducir el concepto de Metodología Ágil de Desarrollo. Dichas metodologías formarán parte esencial de nuestro proceso de desarrollo por lo tanto vale la pena comentar un poco sobre su origen.

Para tener una visión global y comprender el origen y la razón de ser de las metodologías ágiles hay que remontarse al año 1968 o año de la "Crisis del Software". Dicha crisis no fue otra cosa que el nombre que se le puso a los continuos retrasos, sobrecostes y deficiencias de calidad o utilidad que se producían en el desarrollo de software. Un problema que se producía asiduamente y que había que solucionar. Esto dio lugar a que todas las personas interesadas comenzarán a buscar una solución a este problema.

No fue hasta el año 2001 en una reunión celebrada en Salt Lake City donde se debatía sobre desarrollo de software cuando se acuña el término "Métodos Ágiles" y surgen los cuatro principios que conforman el "Manifiesto Ágil", base de este "agilísimo" y de métodos de gestión, tales como: DSDM, Scrum, XP, FDD, …

Estos nuevos métodos de desarrollo surgen como respuesta a un nuevo entorno cambiante, muy competitivo, donde los lanzamientos de productos, los cambios en requisitos y prioridades de los clientes y las mejoras son cada vez más continuos y se producen en menores intervalos de tiempo y el valor en alza, que otorga la ventaja competitiva para estar en los primeros puestos de un sector, es la innovación.

Debido a que nuestro proyecto, en principio, podría estar sujeto a cambios en requisitos y demás, tomé la decisión de aplicar Metodologías Agiles para el desarrollo del software, ya que era la forma de podernos adaptar más eficazmente al constante cambio fruto de las continuas ideas surgidas de las reuniones.

En este capítulo explicaremos a fondo la metodología ágil utilizada para desarrollar la aplicación y por qué fue escogida. Haremos un recorrido por los principales roles dentro del desarrollo, los documentos obtenidos y el ciclo de vida en general.

4.1 – METODOLOGÍAS

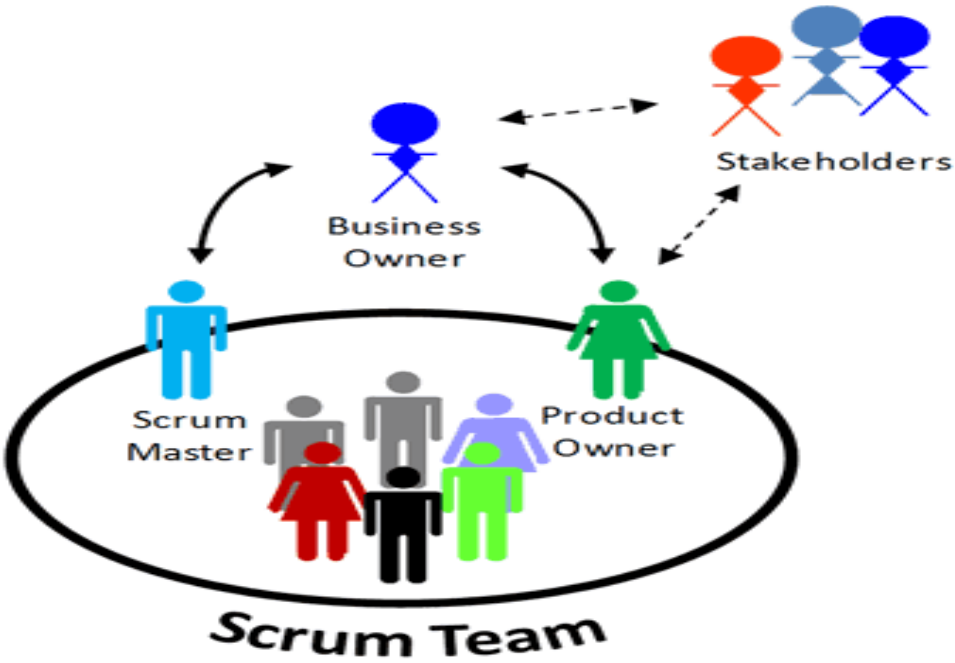
La metodología escogida para el desarrollo del proyecto fue la metodología Scrum. La decisión fue tomada basándonos en las múltiples ventajas que ofrece dicha metodología como por ejemplo:

- Entrega mensual (o quincenal) de resultados (los requisitos más prioritarios en ese momento, ya completados) lo cual proporciona las siguientes ventajas:
    - Gestión regular de las expectativas del cliente y basada en resultados tangibles.
    - Resultados anticipados (*time to market*).
    - Flexibilidad y adaptación respecto a las necesidades del cliente, cambios en el mercado, etc.
    - Gestión sistemática del Retorno de Inversión (ROI).
    - Mitigación sistemática de los riesgos del proyecto.
- Productividad y calidad.
- Alineamiento entre el cliente y el equipo de desarrollo.

4.1.1 – SCRUM

Scrum es una metodología ágil de desarrollo de software, iterativa e incremental que se utiliza para la gestión de desarrollo de productos. Define "una estrategia flexible del desarrollo de productos, donde un equipo de desarrollo trabaja como una unidad para alcanzar un objetivo común". Desafía constantemente los supuestos del "enfoque tradicional y secuencial" para el desarrollo de productos. Además de lo antes expuesto

permite a los equipos de desarrollo auto-organizarse y fomenta la colaboración interpersonal, la comunicación diaria cara a cara, exhortando a los equipos a desarrollar el software en la misma ubicación física. Además, también fomenta la cooperación entre las diferentes disciplinas del proyecto de desarrollo.



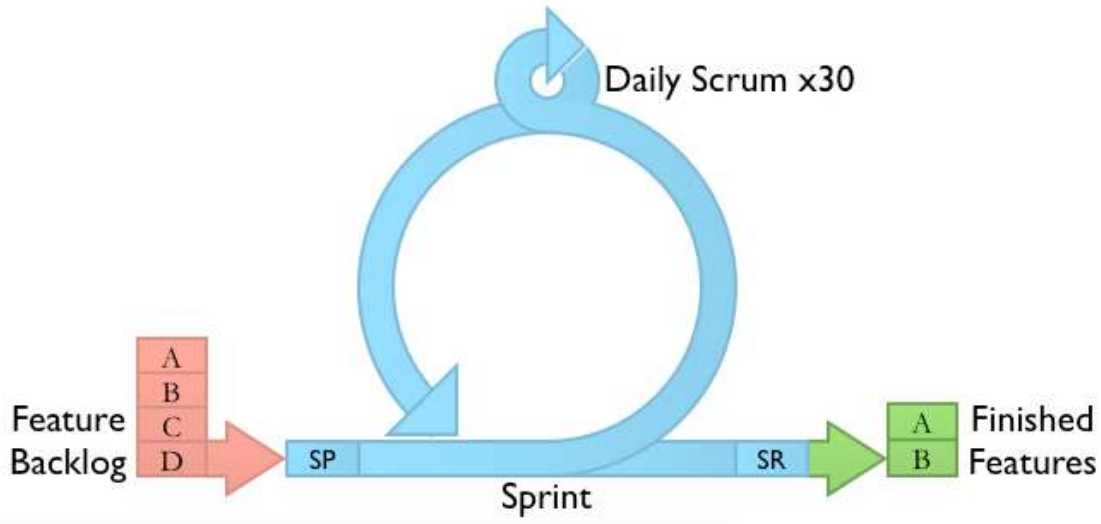
Dibujo 6 - Principales roles en la metodología Scrum

Un principio clave de Scrum es el reconocimiento de que durante un proyecto los clientes pueden cambiar de opinión acerca de lo que quieren y necesitan (a menudo llamados "requisitos churn"), y que los desafíos impredecibles no pueden ser fácilmente abordado de una manera predictiva, planificada o tradicional. Como tal, Scrum adopta un enfoque empírico, aceptando de antemano que el problema no puede ser completamente entendido o definido, centrándose en su lugar en la maximización de la capacidad del equipo para responder y entregar rápidamente los requerimientos emergentes.

Scrum tiene 3 roles bien definidos. El 'Scrum Master' o 'Scrum Manager', que mantiene los procesos y trabaja de forma similar al director de proyecto. El 'Product Owner', que representa a los interesados. Y por último, el Team, que es el equipo de desarrolladores.

La metodología SCRUM tiene dos tipos de eventos muy bien definidos. El primero de ellos es el llamado SPRINT y el otro son las REUNIONES.

Un sprint (o iteración) es la unidad básica de desarrollo de Scrum. El SPRINT es un trabajo de desarrollo "timeboxed" o de tiempo limitado; es decir, se limita a una duración específica. La duración se fija de antemano para cada sprint y es normalmente entre una semana y un mes, aunque dos semanas es típico. A continuación una imagen para su mejor comprensión.



Dibujo 7 - Ciclo de vida de un Sprint en la metodología Scrum

Las reuniones de la metodología sirven, sobre todo, para planificar los siguientes pasos, hacer retrospectiva y aprender de lo que salió bien y mal en antiguos Sprint. Las más conocidas son la de planificación del sprint, la que se realiza diariamente durante el sprint y las de cierre del sprint.

Ya por último mencionar que la metodología Scrum produce una serie de extensiones. Una de ellas es el Product Backlog, que es un documento de alto nivel para todo el proyecto. Contiene descripciones genéricas de todos los requisitos y funcionalidades deseables y sus prioridades. Es lo que va a ser construido. Es abierto y solo puede ser modificado por el Product Owner. Otro ejemplo interesante de extensiones es el Sprint

Backlog, que no es otra cosa que un documento detallado donde se describe el cómo el equipo va a implementar los requisitos durante el siguiente sprint.

Vale mencionar que se siguieron la gran mayoría de directrices planteadas por el método ágil. Se realizaron las reuniones de planificación y revisión de cada ciclo. Se realizaron los SPRINTS, donde se desarrollaba las funcionalidades acordadas en las reuniones. Se obtuvieron también documentos implícitos de la metodología SCRUM, como el PRODUCT BACKLOG y el SPRINT BACKLOG. También, a la hora de concertar que requisitos se desarrollarían en cada SPRINT, se tomó como referencia la prioridad y el tiempo de desarrollo de cada uno de ellos, características especificadas en el PRODUCT BACKLOG. Algunas directrices se pasaron por alto por cuestiones de logística y tiempo, como por ejemplo, las reuniones diarias. También la gráfica BURN DOWN TO CHART no se realizó en ningún momento.

4.2 – PATRONES

En este apartado comentaremos y haremos referencias a todos los patrones de diseño y programación, que de una forma u otra nos fueron útiles a la hora de desarrollar todo el código fuente de la aplicación.

SRP

Single Responsibility Principle, según este principio un módulo debe tener una única función. De esta forma cada clase tiene una funcionalidad limitada y concreta, es preferible tener mayor cantidad de módulos con menor cantidad de código que menos módulos menos manejables. También aumenta la posibilidad de que estos módulos sean reutilizados en otro proyecto.

OCP

Open-Closed Principle, según este principio un módulo debe estar cerrado a cambios pero abierto a extensiones. Una excepción a esto es que puede estar abierto a corrección de

errores. De esta forma módulos escritos en el pasado son compatibles con nuevos proyectos mediante la inversión de dependencias.

MVC

Modelo – Vista – Controlador es un patrón de arquitectura de software de uso extendido que separa los datos y la lógica de negocio (modelo) de la interfaz de usuario (vista) del módulo encargado de gestionar los eventos a controlar (controlador) en una aplicación.

Modelo: es la representación lógica interna con la que el programa opera. Formada por datos que representan la situación que se está tratando.

Controlador: es el código encargado de recoger los datos del usuario a través de la vista, procesar los mismos y devolver el resultado de nuevo a la vista para que se muestren al usuario.

Vista: es el componente encargado de mostrar al usuario los datos procesados por el controlador del programa.

De esta forma el usuario interacciona con el programa a través de la vista (interfaz de usuario), las acciones realizadas por este son recogidas por el controlador, quien procesa los datos recogidos haciendo uso del modelo. Tras procesar éstos, se envían datos actualizados a la interfaz de usuario.

CLASES SINGLETON

Patrón de diseño de clases que permite que en cada momento exista una única instancia de esta clase.

COMMAND

El patrón Command permite diseñar clases que encapsulan las acciones que se deben ejecutar como respuesta a un evento ocurrido en la interfaz de usuario.

4.3 – PLANIFICACIÓN Y TEMPORALIZACIÓN

En este apartado realizamos un desglose de las horas empleadas en el desarrollo de la aplicación. También se desglosa en que se gastaron cada hora de cada Sprint.

FASES DEL DESARROLLO	HORAS
INICIO	**20 H**
- REUNIONES CON LOS CLIENTES	4 H
- CAPTURA DE REQUISITOS	12 H
- CREACION DEL PRODUCT BACKLOG	4 H
SPRINT 1	**56 H**
- REUNION DE PLANIFICACION DEL SPRINT	8 H
- DESARROLLO DE REQUISITOS Y FUNCIONALIDADES	40 H
- REUNION DE REVISION DEL SPRINT	4 H
- REUNION RETROSPECTIVA DEL SPRINT	4 H
SPRINT 2	**60 H**
- REUNION DE PLANIFICACION DEL SPRINT	8 H
- DESARROLLO DE REQUISITOS Y FUNCIONALIDADES	44 H
- REUNION DE REVISION DEL SPRINT	4 H
- REUNION RETROSPECTIVA DEL SPRINT	4 H
SPRINT 3	**64 H**
- REUNION DE PLANIFICACION DEL SPRINT	8 H
- DESARROLLO DE REQUISITOS Y FUNCIONALIDADES	48 H
- REUNION DE REVISION DEL SPRINT	4 H
- REUNION RETROSPECTIVA DEL SPRINT	4 H
SPRINT 4	**56 H**
- REUNION DE PLANIFICACION DEL SPRINT	8 H
- DESARROLLO DE REQUISITOS Y FUNCIONALIDADES	40 H
- REUNION DE REVISION DEL SPRINT	4 H
- REUNION RETROSPECTIVA DEL SPRINT	4 H
SPRINT 5	**56 H**
- REUNION DE PLANIFICACION DEL SPRINT	8 H
- DESARROLLO DE REQUISITOS Y FUNCIONALIDADES	40 H
- REUNION DE REVISION DEL SPRINT	4 H
- REUNION RETROSPECTIVA DEL SPRINT	4 H
TOTAL	**312 H**

5 – DESARROLLO

En este capítulo abarcaremos todos los aspectos de interés relacionados con el desarrollo de la aplicación. Primero abarcaremos toda la fase de análisis y diseño y luego entraremos a explicar algunas de las funcionalidades fundamentales que son los pilares de nuestro trabajo. Todo el código fuente de la aplicación se adjuntó a la mejoría en un CD.

5.1 – SERVICIO WEB

Un Servicio Web o Web Service es una tecnología que utiliza un conjunto de protocolos y estándares que sirven para intercambiar datos entre aplicaciones. Es un método de comunicación de datos entre dos dispositivos electrónicos con conexión a la red. El Web Service está ubicado en un servidor con una dirección en la red y siempre debería estar operativo. Además está diseñado para soportar la interacción "machine to machine" en la red. Esto permite que distintas aplicaciones de software desarrolladas en lenguajes de programación diferentes, y ejecutadas sobre cualquier plataforma, pueden utilizar los servicios web para intercambiar datos en redes de ordenadores como Internet. La interoperabilidad se consigue mediante la adopción de estándares abiertos. Las organizaciones OASIS y W3C son los comités responsables de la arquitectura y reglamentación de los servicios Web. Una forma de resumir lo que es un Web Service es comentar que es un servidor que atiende las peticiones de los clientes web y les envía los datos y recursos solicitados.

Los Web Service se apoyan en diversos protocolos y estándares para el intercambio de datos. A continuación se comentan algunos de ellos:

- Web Services Protocol Stack: Así se le denomina al conjunto de servicios y protocolos de los servicios Web.

- XML (Extensible Markup Language): Es el formato estándar para los datos que se vayan a intercambiar.

- SOAP (Simple Object Access Protocol) o XML-RPC (XML Remote Procedure Call): Protocolos sobre los que se establece el intercambio.

- HTTP (Hypertext Transfer Protocol), FTP (File Transfer Protocol), o SMTP (Simple Mail Transfer Protocol): los datos en XML también pueden enviarse de una aplicación a otra mediante estos protocolos

- WSDL (Web Services Description Language): Es el lenguaje de la interfaz pública para los servicios Web. Es una descripción basada en XML de los requisitos funcionales necesarios para establecer una comunicación con los servicios Web.

- UDDI (Universal Description, Discovery and Integration): Protocolo para publicar la información de los servicios Web. Permite comprobar qué servicios web están disponibles.

- WS-Security (Web Service Security): Protocolo de seguridad aceptado como estándar por OASIS (Organization for the Advancement of Structured Information Standards). Garantiza la autenticación de los actores y la confidencialidad de los mensajes enviados.

Números TIP es un servicio web en línea que realiza la conversión de un número escrito con cifras a su texto correspondiente ofreciendo todas las posibilidades según diversos contextos interpretativos: cardinalidad, ordinalidad, fracción y multiplicación. Así mismo, se muestra el texto correspondiente a otros ámbitos en los que participan los números: romanos, conjuntos o colectivos, polígonos, poliedros y edades.

Números TIP realiza la conversión de una cifra a su texto cardinal, ordinal, fraccionario o partitivo, multiplicativo, romano…, y ofrece información morfológica, ortográfica y gramatical de cada uno de los términos. Además, se incluyen ejemplos que ayudan a la comprensión y buen uso. Los contextos implementados son:

- Texto de número cardinal
- Texto de número ordinal
- Texto de número fraccionario
- Texto de una fracción
- Texto de número multiplicativo
- Numero romano
- Texto de un número de colectivo
- Texto de un número de silabas
- Nombre de polígonos
- Nombre de poliedros
- Edades
- Nacido

5.2 – ANÁLISIS DE REQUISITOS

La principal razón de ser de este trabajo era crear una aplicación móvil que fuera capaz de proporcionarle al usuario final una forma más amena y fácil de poder acceder a todos los datos proporcionados por el Servicio Web Números TIP. Dicha aplicación debía cumplir unos parámetros de diseño y requerimientos para poder llegar a más usuarios finales.

El primer requisito funcional de la aplicación debería ser el de darle la oportunidad al usuario de introducir un número para recibir toda la información referente a este en sus formas cardinales, ordinales, etc. También debería existir una funcionalidad que velará por la correcta introducción de números, es decir, que no permitiera introducir letras ni otros signos. También se debía crear una funcionalidad que controlará la conexión a internet del dispositivo móvil, ya que sin ella sería imposible acceder al servicio web.

Por lo antes expuesto uno de los principales requerimientos de la aplicación era crear una funcionalidad que fuera capaz de conectarse a un servicio web. Para ello decidí que la forma más práctica y sencilla era hacerlo a través de la librería Ksoap para Android con la cual podremos hacer peticiones de forma muy sencilla.

Otro requisito fundamental era lograr que la aplicación fuera capaz de desplegarse en pantallas de diferentes tamaños y densidades. Para ello se utilizaron los archivos de recursos, que son lo que nos permiten realizar un diseño para cada tipo de pantalla diferente.

El gran reto que planteaba esta aplicación era lograr que toda la información devuelta por el servicio web fuera desplegada, en las reducidas pantallas de los dispositivos móviles, de manera tal que el usuario final pudiera acceder a ella sin tener que deslizar la pantalla excesivamente o dar muchos clics.

Era importante también darle la oportunidad al usuario de introducir nuevos parámetros a la búsqueda sin tener que cerrar la aplicación o volver a la pantalla de inicio. Esto se

logró introduciendo un buscador en la barra de menú. Más adelante se explicará a fondo el tema.

Uno de los principales requerimientos era una interfaz que proporcionara varios idiomas para la interacción de usuarios finales, logrando de esta forma mayor internacionalización y mayor esparcimiento de nuestro idioma a nivel europeo y mundial.

5.3 – INTERFAZ

Sería interesante, antes de proseguir, dar una pequeña explicación de qué y cómo están compuestas las aplicaciones Android. Para ello tendríamos que introducir el concepto de Actividad (Activity).

Una actividad es un componente de aplicación que proporciona una interfaz con la que los usuarios pueden interactuar con el fin de hacer algo, como marcar el teléfono, tomar una foto, enviar un correo electrónico, o ver un mapa. Cada actividad posee una ventana en la que se dibuja su interfaz de usuario. La ventana típicamente llena la pantalla, pero puede ser menor que la pantalla y flotar en la parte superior de las demás ventanas. Esta interfaz se dibuja en un fichero .XML. Cada actividad también posee un fichero .java que es donde se desarrolla el código que controla dicha actividad.

Una aplicación por lo general contiene múltiples actividades que están estrechamente relacionadas entre sí. Por lo general, existe una actividad en la aplicación especificada como actividad "principal", que se presenta al usuario al iniciar la aplicación por primera vez. Cada actividad puede entonces comenzar otra actividad con el fin de realizar diferentes acciones. Cada vez que se inicia una nueva actividad, la actividad anterior se detiene, pero el sistema conserva la actividad en una pila. Cuando se inicia una nueva actividad, se inserta en la pila y toma la atención del usuario. La pila sigue el mecanismo básico "last in, first out", por lo que, cuando el usuario termina con la actividad actual y presiona el botón Atrás, se destruye dicha actividad y la actividad anterior se reanuda.

Una vez explicado el tema de las Android Activities proseguiremos a mostrar las interfaces de las tres actividades que componen nuestra aplicación. Explicaremos cada una de ellas y al lado mostraremos una imagen para que los usuarios puedan ver los componentes de cada una de ellas.

La primera interfaz que el usuario final ve cuando entra en la aplicación es la que mostramos en la imagen de la derecha. Su nombre en el código fuente es fragment_inicio.xml. Como podemos observar es una interfaz plenamente orientativa. En ella se muestra el logo de la aplicación, que contiene el nombre y una referencia al grupo IATEXT. En la parte inferior de la interfaz podemos ver un texto que exhorta a las personas a utilizar nuestra aplicación y con ello ganar todos los conocimientos que proporciona el Servicio Web Números TIP. Es importante conocer que dicha interfaz tiene, en el .java de la clase, un código para que, a los 1,5 segundos, inicie la actividad "Number Finder". Este código termina esta actividad e inicia la próxima y se encuentra en el método onCreate de la clase Inicio.java. En dicho código hacemos uso de la clase Handler, que se utiliza para procesar acciones asociadas con un hilo de programa. Ya por último comentar, que tanto para esta como para las otras UI hemos apostado por un fondo degradado entre amarillo y azul. A continuación se muestra una imagen con el fragmento de código del que se habló anteriormente.

Dibujo 8. Interfaz de Inicio

```
/***********ESTE CODIGO HACE SALTAR A LOS 1.5 SEGUNDOS DE INICIO A NUMBERFINDER*
final Handler handler = new Handler();
handler.postDelayed(new Runnable() {
    public void run() {
        //
        Intent principal_page = new Intent(Inicio.this, NumberFinder.class);
        Inicio.this.startActivity(principal_page);
        Inicio.this.finish();
    }
}, 1500);
```

Dibujo 9. Fragmento de código Handler

Dibujo 10 - Interfaz Number Finder

Esta es la segunda interfaz de la aplicación, a la cual se llega desde la anterior. Esta interfaz está dedicada a darle la posibilidad al usuario de introducir un número del cual se quiera saber todos los contextos. Como se puede observar mantiene la misma tonalidad de fondo que la anterior y tiene un cartel orientativo para el usuario final. Luego, unos pixeles más abajo, poseen un cuadro de texto editable donde el usuario introducirá el número deseado. También aparecen dos botones. El derecho, como su texto lo indica, es para salir de la aplicación completamente. El botón izquierdo es para realizar la petición del número deseado al servicio web. Se desarrollará más adelante pero, como adelanto, podemos comentar que para realizar la petición nos apoyamos en una clase de Android llamada Intent que, entre otras funciones, es la encargada de iniciar otra Actividad. También vale comentar que, como en el caso anterior, la actividad "NumberFinder" se cierra cuando se inicia la otra. En el

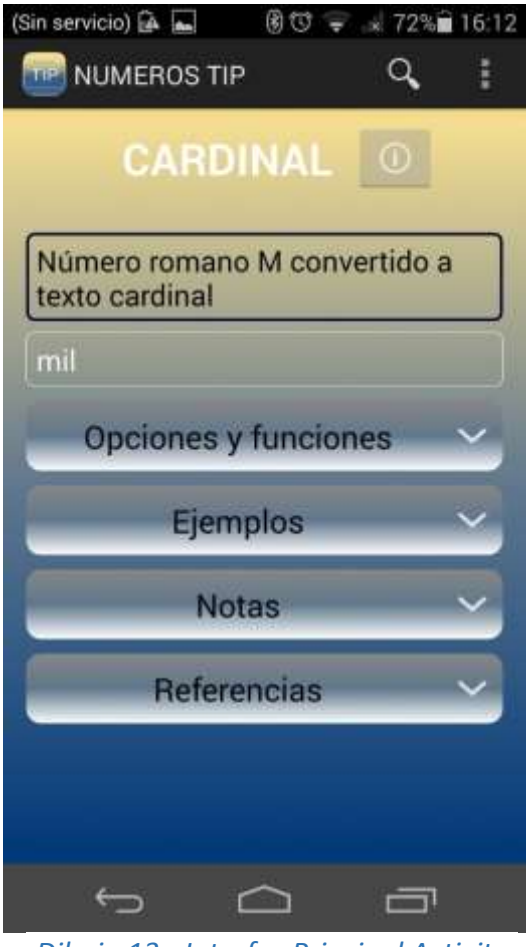
fichero .java correspondiente a esta interfaz hemos creado un pequeño script que es el encargado de controlar que el usuario tenga conexión a internet (3G o WiFi) en el momento que realice la petición al servicio web y también controlamos que haya introducido algún texto en el momento de realizar la petición. No obstante a lo antes señalado en el mismo servidor se realiza un tratamiento de errores del cual comentaremos más adelante. A la derecha del texto podemos ver una imagen de cómo quedaría la interfaz con el teclado virtual de Android. Se pude observar que el usuario es capaz de introducir el texto que desee y luego el servicio web realiza un extenso tratamiento de errores. Vale comentar que se ha introducido el código pertinente para que el botón de Enviar del teclado virtual sea capaz de realizar la petición al servicio web ahorrándonos de esta manera tener que esconder dicho teclado y realizar la petición con el botón buscar de la interfaz.

Dibujo 11 - Teclado Virtual de Interfaz Number Finder

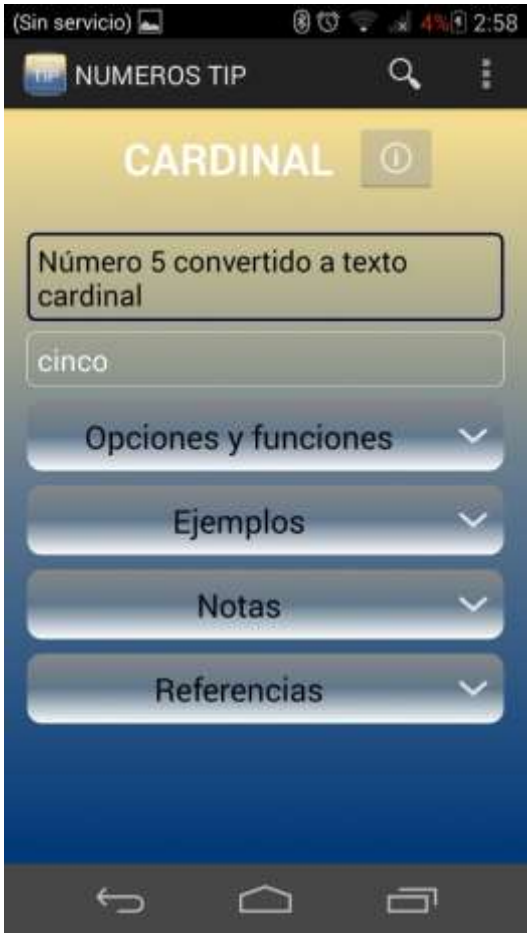
Esta es la interfaz final de la aplicación (Dibujo 12), ya que es donde se muestran todos los datos solicitados. Los colores y tonalidades son iguales a las anteriores pero en esta aparece la barra de menú, que en este caso utilizamos para introducir la búsqueda de un segundo número. Además, tiene otras opciones que serán analizadas más adelante. También se puede observar que contexto está siendo mostrado, en este caso el cardinal. Al lado del nombre del contexto aparece un botón de información que en caso de ser pulsado mostrará un cuadro de dialogo con la descripción, por la RAE, de lo que es un número cardinal. Luego aparece, haciendo uso de una combinación de botones y cajas de texto, toda la información referente a la forma cardinal del número solicitado. Para lograr mostrar esta gran cantidad de información, sin que fuera tedioso para el usuario el hecho de tener que desplazarse hasta el final de la interfaz introdujimos "Buttons" que son los

encargados de hacer visibles los "Relative Layouts", que contienen todas las "Text View" agregadas de forma dinámica durante el proceso de dibujo de la interfaz. También hacemos uso del componente "Scroll View" que nos permite desplazarnos en la información solicitada de manera táctil. A modo de resumen, podemos comentar que la interfaz principal es dibujada en tiempo de ejecución y contiene los siguientes componentes:

- Relative Layouts
- Buttons
- Text View
- Scroll View



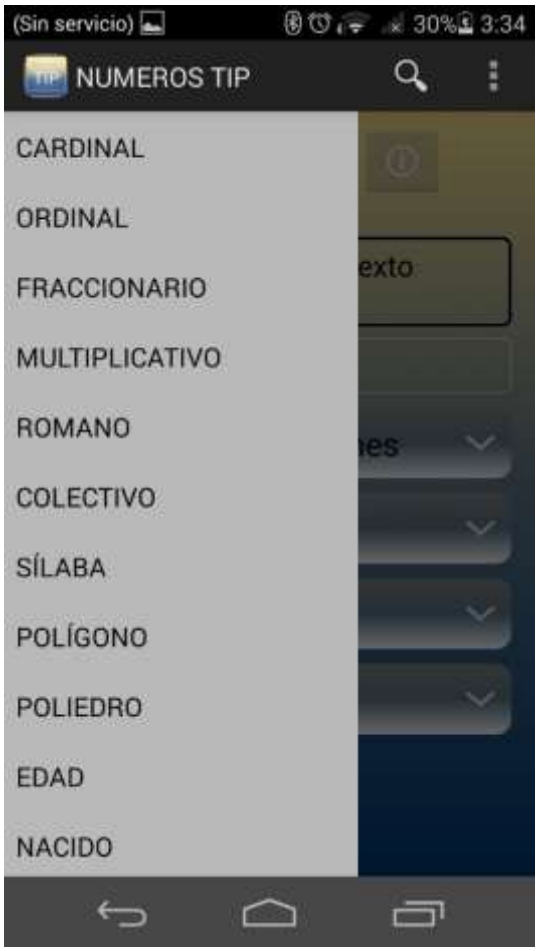
Dibujo 13 - Interfaz Principal Activity Romanos



Dibujo 12 - Interfaz Principal Activity

A continuación mostraremos una imagen de cómo quedaría la interfaz en el caso de ser introducido un número romano (Dibujo 13). No existe mucha diferencia en la forma de crearla, solo que en este caso el número está escrito con letras y no en su forma numérica. Vale mencionar que la aplicación también acepta números negativos, números fraccionarios y números decimales. En el caso de que el servicio web devuelva que el número entrado es erróneo también se muestra la interfaz de error con algunos consejos sobre cómo introducir un número correctamente y que tipo de números acepta el servicio web.

Esta otra imagen muestra la solución que le dimos al hecho de tener tanta información y tan poco espacio para mostrarla. Para no poner toda la información junta y hacer deslizar la pantalla en demasía al usuario final, decidimos dibujar un panel izquierdo con enlaces a cada uno de los contextos del número que devolvía el servicio web. Luego, mediante enlaces, logramos que en cuanto se pinche en alguna otra forma, se dibuje la información referente a ella en la pantalla principal. Para dibujar este panel nos hemos apoyado en el componente "List View" en el cual se dibujan todos los nombres y enlaces a los diferentes contextos. Más adelante se explica detalladamente que código utilizamos para lograr este comportamiento.



Dibujo 14 - Interfaz Principal Left Panel

Ya por último mencionar que creamos un fichero XML para cada actividad, para cada uno de los cuatro tamaños de pantallas estandarizados en Android.

5.4 – IMPLEMENTACIÓN DE FUNCIONALIDADES

En este apartado de la memoria explicaremos a grosso modo las principales funcionalidades que requería nuestra aplicación. Encontraremos imágenes orientativas y del código fuente de las mismas. Explicaremos el enfoque tomado para el desarrollo de cada una de ellas. Para mayor información referente al código, consultar el código fuente que aparece en un fichero en el CD.

5.4.1 – INTRODUCIR UN NÚMERO

Esta funcionalidad esta implementada para darle la oportunidad al usuario de introducir un número para iniciar todo el proceso. Lo primero que he creado es un cuadro de texto editable para la introducción del número. Vale recalcar que este cuadro de texto está configurado para que se pueda introducir cualquier ristra de caracteres, de manera tal que el servidor es el encargado de realizar todo el tratamiento de errores. A pesar de todo, y como comentamos anteriormente, realizamos algunas comprobaciones antes de iniciar la conexión con el servicio web. Para tener una imagen más grafica de la interfaz para esta funcionalidad remitirse al Dibujo 9.

Luego de introducir el número proseguimos a pulsar el botón Buscar de la misma interfaz. Antes de proseguir mostraremos una imagen del código mediante el cual hemos logrado que el botón Enviar del teclado virtual tenga un comportamiento similar al del botón Buscar. Para ello nos apoyamos en la clase onEditorActionListener.

```
EditText editText = (EditText) findViewById(R.id.editText1);
editText.setOnEditorActionListener(new OnEditorActionListener() {
    @Override
    public boolean onEditorAction(TextView v, int actionId, KeyEvent event) {
        boolean handled = false;
        if (actionId == EditorInfo.IME_ACTION_SEND) {
           conectar(v);
           handled = true;
        }
        return handled;
    }
});
```

Dibujo 15 - Código del botón de Enviar virtual.

Al pulsar este botón se ejecuta un método llamado conectar que es el encargado de, haciendo uso de otros métodos, comprobar algunos parámetros como por ejemplo que haya algún número escrito o que el terminal móvil tenga conexión a internet. Para realizar todas estas comprobaciones dicho método se apoya en el método scrip_controlador que

comprueba las conexiones a internet y el tamaño de la ristra escrita. Para más información acerca del código fuente de este método revisar el CD entregado junto con la memoria.

Luego de todas estas comprobaciones, el método conectar realiza la llamada a la clase principal, proporcionándole la cadena introducida.

Para esta llamada se utiliza un elemento o clase llamado Intent. Los Intents son mensajes asincrónicos que permiten a las actividades solicitar funcionalidades de otros componentes o actividades de Android. Los Intents te permiten interactuar tanto con componentes de la misma aplicación como con componentes externos. Por ejemplo, una actividad puede iniciar una actividad externa para tomar una foto o puede iniciar una interna para procesar una información. Vale mencionar que con los Intents puedes facilitar datos a las actividades iniciadas, sean tanto internas de la aplicación como externas a ella.

A continuación mostraremos algunas imágenes en las que se puede observar cómo, a la nueva actividad que se va a iniciar, se le facilita la ristra de caracteres escrita por el usuario y que se le pasará como parámetro al servicio web. Para ver todo el código fuente de estos métodos remitirse al fichero NumberFinder.java en el código fuente adjuntado a esta memoria.

Dibujo 16 - Código del método conectar (). Ejemplo de utilización de los Intents

5.4.2 – CONECTAR CON EL SERVICIO WEB

Esta funcionalidad se implementa con el objetivo de realizar la conexión al Servicio Web Números TIP. Dicha parte del código es la encargada de realizar la petición y recepción de datos y luego, realizar un complicado y complejo proceso de tratamiento de cadenas para devolver ya las respuestas listas para ser dibujadas en la interfaz de usuario.

Antes de proseguir valdría recalcar que en una de las reuniones con el tutor realicé la proposición de crear un método en el Web Service encargado de realizar un tratamiento de errores. Este método fue desarrollado e introducido en el código fuente del Web Service. La idea viene dada porque el usuario promedio por lo general introduce muchos números que nos son correctos o que tienen un pequeño error en su formato, como por ejemplo el hecho de introducir los números de millares sin el espacio intermedio que exige la RAE. Por la razón mencionada anteriormente este método internamente contiene disímiles funcionalidades, de las cuales se explicarán algunas a continuación.

Una de las más importantes es el tratamiento de los puntos y comas para los miles, millones y números decimales. De esta manera se asume que si existe solo un punto o una coma el usuario está intentando indicar que es un número decimal. Si existe más de una repetición de cualquier signo de puntuación entonces se trata como un número entero asumiendo que el usuario quería separa los miles y millones. Si existen los dos signos en la misma cadena introducida entonces se asume que el usuario estaba intentando ingresar un número decimal pero quería separar los miles y millones y así sucesivamente. Para más información adjuntaré en el CD el código fuente de dichos métodos.

Otra funcionalidad es la encargada de eliminar todos los ceros a la izquierda y todos los ceros a la derecha después de una coma. Además se eliminan todos los espacios en blanco mal colocados y se introducen donde vayan correctamente. Además de lo antes mencionado se realiza tratamiento de posibles errores para la mayoría de signos y errores de la notación científica, etc.

También creé otro método encargado de realizar tratamiento de errores en las cadenas de números romanos introducidas por el usuario.

Este método funciona de manera muy sencilla. Primero se comprueba que toda la cadena introducida son caracteres utilizados por los romanos. Luego se comienza a leer la cadena de derecha a izquierda. Si se encuentra un número menor se resta y si es mayor o igual se suma y el resultado se almacena en una variable de tipo entero. Vale recalcar que solo se resta una vez, esto quiere decir que si, por ejemplo, nos introducen la cadena VVVV el resultado almacenado en la variable sería 15. Luego el resultado de la variable de tipo entero se convierte a número romano utilizando algoritmos encontrados en la red. Se comparan ambos resultados y si son iguales significa que el número romano está bien escrito. En caso de que sea diferente se devuelve un error y algunos ejemplos de cómo se deberían escribir los números romanos.

Estos métodos nos han permitido ampliar nuestra flexibilidad a la hora de aceptar una determinada cadena. Nos han permitido mostrar más respuestas, y sobre todo, aceptar más cadenas introducidas por los usuarios. Si no dispusiésemos de estos métodos, la cantidad de errores de escritura en los números introducidos sería elevada y la aplicación y el servicio en general sería de difícil uso y baja reputación.

Volviendo a la funcionalidad comentada anteriormente para su desarrollo decidí crear una clase aparte (WebService.java) en la cual están todas las variables y métodos necesarios para el correcto funcionamiento del proceso. Vale destacar que para esta conexión nos apoyamos en la librería K-SOAP que fue descrita en la sección de software del proyecto.

Para la conexión se necesitan cuatro variables proporcionadas por los creadores del servicio web y la cadena de caracteres introducida por el usuario. Vale recalcar que el servicio web tiene la posibilidad de devolver datos tanto en inglés como en español y para ello solo se tiene que solicitar el idioma usado en el teléfono. En base a dicho idioma toma la decisión de presentar datos en español o inglés. El proceso de conexión consta de cinco fases de las cuales se muestra una imagen a continuación.

```
/*DEFINIENDO LA PETICION O REQUEST*/
SoapObject request = new SoapObject(namespace, Metodo);
request.addProperty("numeroText", number);
request.addProperty("lang", "es");

/*DEFINIENDO EL SOBRE O ENVELOPE*/
SoapSerializationEnvelope envelope = new SoapSerializationEnvelope(SoapEnvelope.VER11);
envelope.dotNet = true;
envelope.setOutputSoapObject(request);

/*DEFINIENDO EL CANAL DE TRANSPORTE O TRANSPORT CHANNEL*/
HttpTransportSE transport = new HttpTransportSE(url);

/*REALIZANDO LA LLAMADA POR DATOS AL WEBSERVICE*/
transport.call(accionSoap, envelope);

/*RECIBIENDO LOS DATOS DEL WEBSERVICE*/
web_service_response = (SoapObject)envelope.getResponse();
one_or_another = true;
```

Dibujo 17 - Fases de conexión al Servicio Web Números TIP

Luego de que todos los datos estén en el objeto web_service_response de tipo SoapObject se comienza el proceso de tratamientos de cadena en el cual intervienen los métodos webServiceResultTreatment () y resultLineCleaner (), dejando todas las cadenas de datos en un arreglo de cadenas.

El método webServiceResultTreatment () es el encargado de devolver un arreglo de strings con todas las cadenas de datos devueltas por el servidor, pero para llegar al resultado lo primero que hace es limpiar la cadenas de código HTML. Para ello se utiliza el método resultLineCleaner () que, mediante una serie de sustituciones, limpia todos los elementos de código HTML.

Es imprescindible y muy importante destacar que este método no quita todo el código HTML. La razón por la cual no se hace es que, como se explicará en la siguiente funcionalidad, existen algunos signos que tomamos como códigos o marcas a la hora de dibujar un botón o una caja de texto, permitiéndonos de esta manera saber que hacer y lograr implementar un "protocolo" en el cual lo dibujado en la interfaz se adapta totalmente a los datos devueltos por el Web Service. Esto quiere decir que si, por alguna razón, en el futuro se decide cambiar los datos, esto no afectará la forma de dibujarlos en

pantalla. Ejemplos de signos mencionados son el & o el #, con los cuales sabemos que debemos dibujar un botón o que nos indica que la cadena es una cabecera respectivamente.

Volviendo al tema del que hablábamos anteriormente proseguimos explicando que el método comienza a recorrer toda la longitud de la cadena y cuando se encuentra un signo de punto y coma reconoce que esa cadena de datos se ha terminado y la introduce en el arreglo de strings. Luego sigue recorriendo la cadena desde el punto en que la dejó y así sucesivamente hasta que llegue al final de la misma. Durante este proceso reconoce las cabeceras de contexto (Cardinal; Ordinal; etc.) y las introduce en un arreglo de strings llamado "headers". Al final de la ejecución de dicho método tenemos dos arreglos de strings, uno que contiene todas las cadenas de datos devueltas por el servidor y otro con las cabeceras de contextos. Para realizar un estudio más a fondo sobre el tratamiento de cadenas se puede revisar el código fuente adjunto en el CD.

5.4.3 – DIBUJAR DATOS EN PANTALLA

Esta funcionalidad, como su nombre lo indica, es la encargada de dibujar los datos en pantalla. Se subdivide en dos partes. La primera de ellas es la encargada de dibujar el panel izquierdo, en el cual existirán enlaces a los datos de cada contexto devuelto por el servidor. Por ejemplo, con solo pulsar el enlace "ORDINAL" se desplegará en la pantalla principal todos los datos referentes al contexto ordinal. Vale remarcar que todos estos enlaces a contextos se extraen del arreglo "headers" creado durante la conexión al servicio web y explicado en el apartado anterior. Para dibujar el panel izquierdo se utiliza el método leftSidePanelDrawing (), el cual, haciendo uso de la clase ArrayAdapter y utilizando el "List View" izquierdo, explicado en el apartado de interfaz, como lienzo dibuja todos los elementos del arreglo "headers" y les asigna a cada uno de ellos un enlace al contexto que representan. Este enlace no es otra cosa que un "Listener" y se asigna en un método llamado setOnItemClickListener (mMessageClickedHandler). Es en la definición de dicho parámetro donde se limpia el lienzo y se vuelve a llamar al método

que dibuja la pantalla principal. Para más información o el código fuente remitirse al CD adjunto a la memoria.

Es importante mencionar que antes de proceder a dibujar en el panel principal cualquier información nueva, siempre se hace uso de un método encargado de limpiar el "lienzo" llamado principalPanelCleaner ().

La segunda parte de la funcionalidad es la encargada de dibujar la pantalla donde el usuario podrá ver todos los datos de cada uno de los contextos devuelto por el servidor. La lógica de esta funcionalidad se implementa en el método principalPanelDrawing (). Para obtener qué dibujar se utiliza un método que devuelve solo los datos relativos a una forma (cardinal, ordinal, etc.).

Este método se llama getSingle_property_attributes () y su lógica consiste en apoyarse en el arreglo de cabeceras para saber desde y hasta que posición del arreglo de datos tiene que comenzar a devolver. Para ello utiliza la posición del elemento pulsado en el panel izquierdo, busca el texto equivalente en el arreglo "headers" y luego busca la posición de dicho texto en el arreglo de datos generales y comienza a devolver todas las cadenas de datos hasta que se encuentra con el texto equivalente a la posición incrementada en uno del arreglo headers.

Luego que se tienen todos los datos de un contexto determinado de utiliza el método resultDistributor (ArrayList <string> arreglo), al cual se le pasa como parámetro el arreglo devuelto por el método explicado anteriormente. Este método tiene la función de distribuir las cadenas de datos en diferentes arreglos más pequeños cuyos elementos serán dibujados en los "Relative Layouts" explicados en el apartado de Interfaz. Para saber que cadena de datos va en que arreglo utiliza una señalización que viene con el resultado del Servicio Web.

Luego que se tienen todos los datos se empieza a dibujar. Para ello se utiliza el método subPropertyDrawer (). Vale mencionar que todos los "Text View" se dibujan y se le asignan características de manera dinámica. A continuación mostraremos algunas imágenes del código y de cómo quedaría la interfaz que se le muestra al usuario.

```
my_web_service.resultDistributor(my_web_service.getSingle_property_attributes(property_requested));
final Button my_buttonn = (Button) findViewById(R.id.button4);

headerDrawer();
subPropertyDrawer(1, R.id.myrelative5, 0, my_web_service.getResult_of_general());
subPropertyDrawer(0, R.id.myrelative1, R.id.button1, my_web_service.getResult_of_options());
subPropertyDrawer(0, R.id.myrelative2, R.id.button2, my_web_service.getResult_of_examples());
subPropertyDrawer(0, R.id.myrelative3, R.id.button3, my_web_service.getResult_of_notes());

if (my_web_service.getResult_of_references().size() != 0)
    my_buttonn.setVisibility(View.VISIBLE);
else
    my_buttonn.setVisibility(View.GONE);

subPropertyDrawer(0, R.id.myrelative4, R.id.button4, my_web_service.getResult_of_references());
```

Dibujo 18 - Código de la funcionalidad para dibujar el panel principal.

```
for(int i = start_of_iterator; i < property.size(); i++)
{
    if (i == 0)

    {
        if (property.get(i).contains("&Todas las "))
            property.set(i, property.get(i).replaceAll("Todas las o", "O"));
            //property.set(i, property.get(i).substring(10));

        final Button my_button = (Button) findViewById(button);
        my_button.setText(property.get(i).substring(1));
    }

    else
        if (property.get(i).contains("#")) {
            lineDrawer(rlayout, i, property, true);
        } else {
            lineDrawer(rlayout, i, property, false);

        }

}
```

Dibujo 19 - Código de un método auxiliar e interfaz resultante para el usuario.

Con el código expuesto en las imágenes superiores se crean “Text Views”, a los cuales se les asignan todas las propiedades dinámicamente. Estos “Text Views” se dibujan o agregan dentro de los “Relative Layouts”. La visibilidad de estos últimos elementos está sujeta a los “Buttons”, los cuales tienen asignado un método que cambia su estado de visibilidad con las pulsaciones o clics.

De esta manera hemos logrado resolver uno de los grandes retos que planteaba el proceso de desarrollo. Hemos logrado que el usuario pueda acceder a cualquier dato sin tener que deslizar la pantalla en demasía o tener que hacer demasiados clics.

Sería imprescindible volver a recalcar que el diseño de la interfaz es flexible y además se crea o dibuja en función de los datos devueltos por el servicio web. De esta manera logramos uno de los mayores retos de cara al futuro y es el de lograr darle independencia a la aplicación del servicio web.

5.4.4 – VISUALIZACIÓN EN DIFERENTES PANTALLAS E IDIOMAS

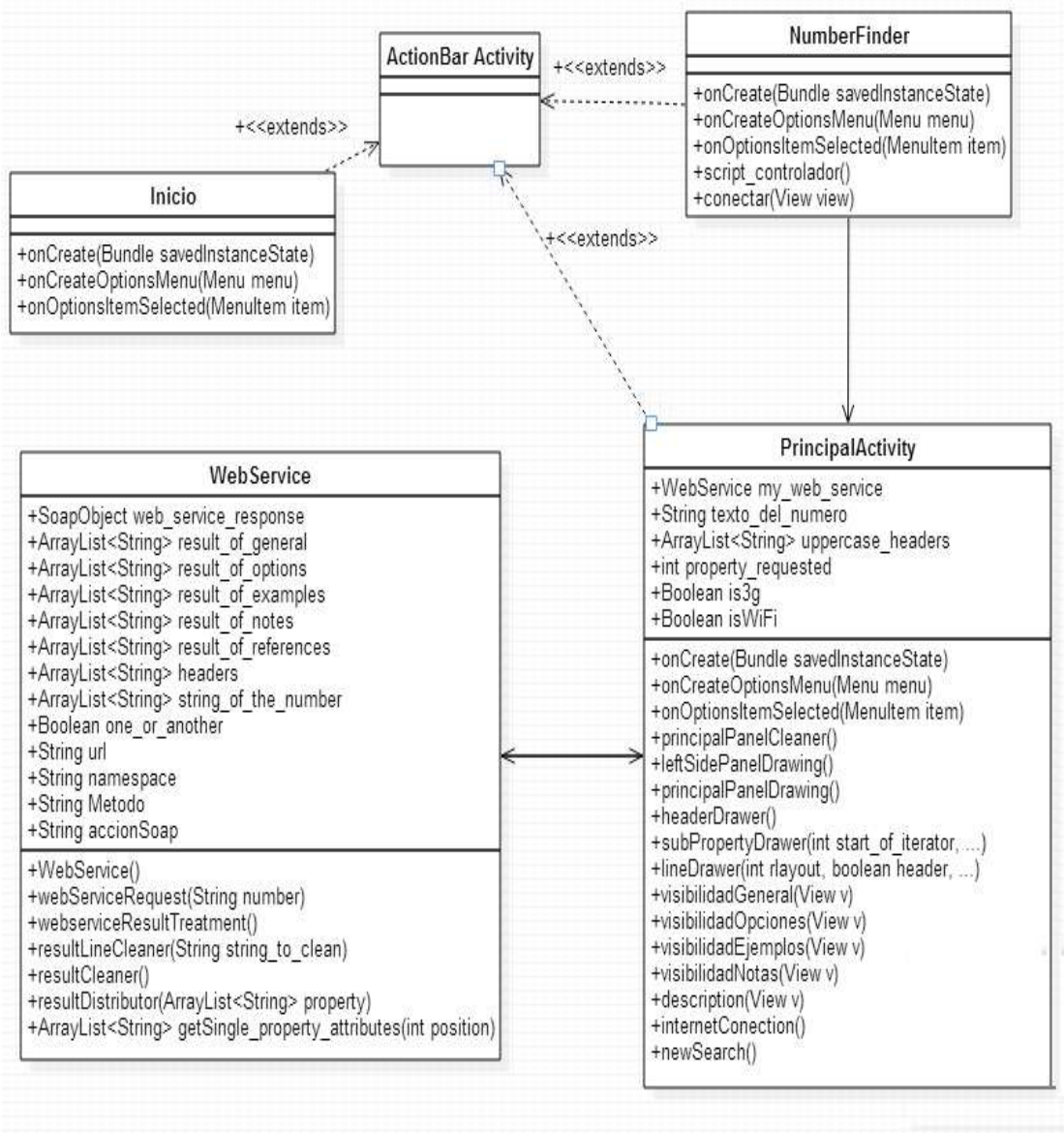
Para lograr que cualquier aplicación sea capaz de desplegarse de manera idónea en pantallas de cualquier tamaño Android proporciona los calificadores de configuración. Estos proporcionan recursos para cada tamaño en específico. Dentro de la carpeta res del IDE Eclipse se crearon las carpetas layout-small, layout-normal, layout-large y layout-xlarge. Dentro de cada una de ellas se creó un fichero XML para cada uno de los diferentes tamaños posibles. De esta manera bien fácil logramos adaptarnos a todos los tipos de pantallas posibles.

Es muy similar en cuestiones de idiomas. Dentro de la carpeta res del IDE Eclipse se creó la carpeta value-"código ISO 639" para cada idioma que se quisiera manejará la aplicación. Dentro de esta carpeta están los ficheros strings.xml, que es donde se introducen todas las palabras que utiliza la aplicación de forma informativa. Cada uno de estos ficheros tiene las palabras en los respectivos idiomas y el mismo SO Android, según la configuración de idioma que tenga el usuario final, se encarga de escoger un fichero string.xml u otro. A continuación proporcionamos algunas imágenes donde se puede observar parte de un fichero XML y además como quedarían las carpetas de "values" dentro del IDE Eclipse.

Dibujo 20 - Fichero XML de idioma Ingles e imágenes de las carpetas del proyecto.

5.5 – DIAGRAMA DE CLASES

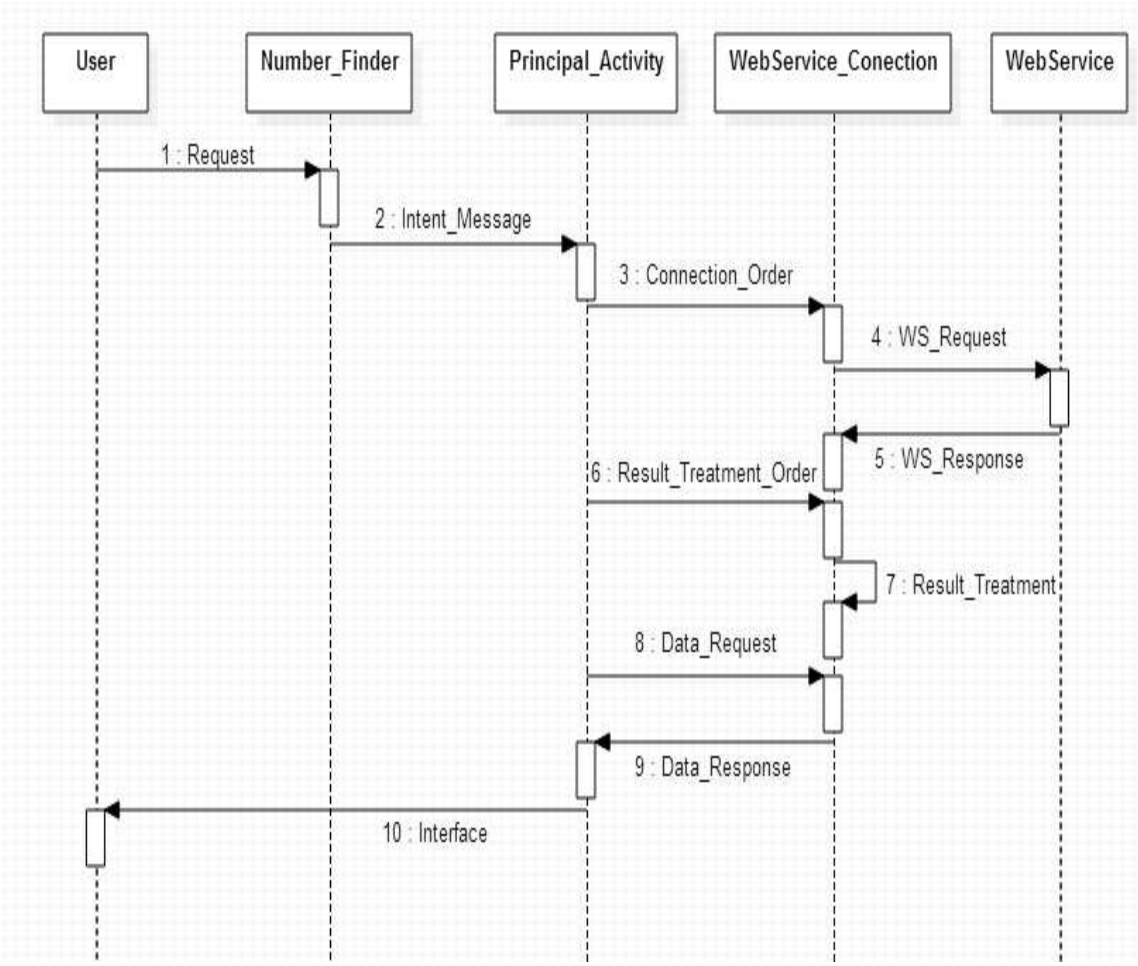
En este apartado de la memoria presentaremos el diagrama de clases de nuestro trabajo. Se intentó desde un primer momento regirnos por los principios de patrón modelo-vista-controlador. Este patrón nos permitirá en el futuro agregar funcionalidades o reutilizar el código para cualquier otro proceso de desarrollo. Con este patrón separamos completamente la lógica de negocio con la forma que es presentada al cliente.



Dibujo 21 - Diagrama de Clases UML

5.6 – DIAGRAMA DE SECUENCIA

A continuación mostramos el diagrama de secuencia de nuestro trabajo. En dicho diagrama se puede observar todo el flujo de la información, desde que sale del usuario en forma de número, hasta que se le presenta en forma de contextos.



Dibujo 22 - Diagrama de Secuencias UML

5.7 – PRUEBAS DE USO

La aplicación ha sido probada en diferentes terminales móviles, de diferentes marcas y con diferentes características de hardware. Todas las pruebas se han realizado sobre terminales que hacen uso del SO Android, aunque si es cierto que se han usado diferentes versiones del mismo.

Hemos hecho especial énfasis en intentar utilizar la aplicación en situaciones anormales, como por ejemplo: sin acceso a internet. Para todas estas situaciones la aplicación ha proporcionado una solución acertada, en algunos casos indicándole al cliente la imposibilidad de comunicarse con el servicio web y en otros casos indicándole que los parámetros entrados no eran correctos.

Para mejorar el diseño y obtener la mejor interfaz de usuario posible, dicho diseño se ha puesto a prueba con más de veinte usuarios reales, a los cuales se les ha solicitado que comenten sobre lo que ellos cambiarían y que harían diferente. También nos hemos encargado que la aplicación sea probada en terminales con diferentes tamaños de pantalla.

Esta forma de validación es la más útil y la que más información proporciona, ya que es el usuario final y consumidor a quien va dirigida la aplicación. Ellos son lo que tienen que ser capaces de usarla y encontrarle utilidad. También, ellos mismos deben validar el diseño.

El flujo de información recibido desde los usuarios finales ha sido bastante satisfactorio. Es decir, los potenciales clientes se han quedado bastante contentos con la aplicación realizada. Algunos de ellos han realizado aportaciones e ideas bastante interesantes que serán tomadas en cuenta a la hora de sacar la segunda versión de la aplicación.

6 – CONCLUSIONES Y RESULTADO

Para poder dar por concluido el Trabajo de Fin de Grado es imperativo verificar que los principales objetivos marcados al inicio del mismo fueron cumplidos por la aplicación desarrollada.

En el caso que nos concierne podemos afirmar con total seguridad que todos los objetivos propuestos al inicio del proyecto han sido cumplidos. También es importante recalcar que todas las funcionalidades deseadas por el cliente y plasmadas en el Product Backlog se han desarrollado, creando un valor importante para el grupo IATEXT.

Por todo lo antes expuesto podemos presentar al mercado la aplicación Números TIP para terminales móviles con sistema operativo Android. Dicha aplicación es capaz de convertir un número a diferentes formas escritas del lenguaje, como son la cardinal, ordinal y muchas más. Dicho producto pose una UI mucho más desarrollada y agradable a la vista del usuario final y organiza la información de manera que el usuario final no tenga que interactuar mucho con ella para llegar a algún contenido. También me gustaría agregar que esta herramienta permite al usuario de otras lenguas interactuar con ella, logrando así su internacionalización.

Gracias a la realización de este proyecto se han podido estudiar nuevas tecnologías, haciendo especial énfasis en la conexión y obtención de datos de servicios webs. Valdría la pena recalcar que no solo ha sido necesario conocer los lenguajes de programación utilizados en el desarrollo de la aplicación, sino también algunas metodologías y patrones de diseño que nos han hecho más fácil el desarrollo. Debido a esta circunstancia el trabajo realizado me ha enriquecido enormemente permitiéndome obtener conocimientos prácticos del proceso de desarrollo de un software.

Sin lugar a dudas, la realización de un trabajo de fin de grado de estas características requiere una gran dedicación y esfuerzo, pero se ven recompensados por la satisfacción de haber conseguido desarrollar una herramienta completa desde cero que pueda ser

utilizada por aquellos estudiantes que están aprendiendo a leer y escribir o aquellas personas que necesitan del conocimiento que provee el Servicio Web Números TIP.

Comentar también que se ha desarrollado un proyecto con muchas posibilidades de poder seguir creciendo en el futuro, ya que mediante el feedback del usuario final se podrían mejorar algunas funcionalidades. Dicho proyecto también podría sentar las bases para el posible desarrollo de aplicaciones Android para los demás productos del grupo IATEXT.

Podría concluir diciendo que he logrado cumplir con el objetivo propuesto al principio del trabajo y que se ha logrado un alto grado de satisfacción en el cliente.

7 – TRABAJOS FUTUROS

En este proyecto se ha desarrollado una aplicación para el acceso, desde terminales móviles con sistema operativo Android, al Servicio Web Números TIP.

Durante todo el proceso de desarrollo se ha tratado de modular todas las partes del sistema, para de esta forma lograr adaptarse a la continua evolución de la programación en Android. Por lo antes expuesto, si en algún momento se quieren introducir nuevas características de los nuevos SDK de Android, no es necesario reprogramar la aplicación completamente.

Además de lo mencionado anteriormente, se sientan las bases para crear una aplicación con las mismas características y funcionalidades pero para terminales móviles con sistema operativo IOS. También se siguieron las pautas de diseño de la aplicación Silabeador TIP, por lo tanto, creando una corriente de diseño de interfaz para todos los productos de grupo IATEXT.

Teniendo en cuenta la duración del TFG (300 horas), se han implementado la gran mayoría de funcionalidades solicitadas. Esto no quita que en el futuro se le puedan agregar diferentes funcionalidades al sistema.

- Poder introducir números mediante la voz, para personas con discapacidad visual.

- Poder escuchar respuestas a solicitudes mediante el audio del móvil.

8 – BIBLIOGRAFÍA

Para la realización de este proyecto se han consultado fuentes tanto virtuales como en soporte físico, se han consultado fuentes lo más actuales posibles.

Las fuentes de soporte físico han sido los siguientes libros:

- Curso Android: Desarrollo de aplicaciones móviles, escrito por Stephanie Falla Aroche, Iván E. Mendoza y Adrián Catalán, publicado en junio de 2011 la versión 1.0.

- Introducción a ArcGIS Runtime for Android escrito por Antonio Remírez

- Introducción a Android escrito por Manuel Báez, Álvaro Borrego, Jorge Cordero, Luis Cruz, Miguel González, Francisco Hernández, David Palomero, José Rodríguez de Llera, Daniel Sanz, Mariam Saucedo, Pilar Torralbo, Álvaro Zapata

- Curso de Programacion Android escrito por Salvador Gómez Oliver, publicado en noviembre de 2011 la versión 2.0.

Las fuentes de información en la red han sido las siguientes:

- Android developers guide. https://developer.android.com/training/index.html

- Stackoverflow Site. Android Questions and Answers. http://stackoverflow.com/

- Tutorials Points http://www.tutorialspoint.com/android/

- Learn Android Programming From Scratch – Beta https://www.udemy.com/learn-android-programming-from-scratch-beta/

- Android Programming Tutorials. Developing Mobile Apps in Java http://courses.coreservlets.com/Course-Materials/pdf/android/Android-Networking-2.pdf

- Android Programming Tutorials. Developing Mobile Apps in Java http://courses.coreservlets.com/Course-Materials/pdf/android/Android-Intents-3.pdf

- Android Programming Tutorials. Developing Mobile Apps in Java http://courses.coreservlets.com/Course-Materials/pdf/android/Android-Localization.pdf