Utilice esta plantilla para la presentación de un **PROYECTO O ANTEPROYECTO**. Las siguientes son recomendaciones basadas en las ediciones más actualizadas de las normas APA (6th Ed. 2010) e Icontec (NTC1486-2008), sin embargo consulte con su asesor, director o coordinador de trabajo de grado para aplicación de aspectos particulares. Primero, de clic en **Archivo** y luego en **Guardar Como** para almacenar una copia personal. Borre este cuadro de texto y edite la información necesaria, eliminando o remplazando los datos adecuados.

**Estructura de Documento** (NTC1486-2008)

- **Márgenes:** Superior: 3.0 cm. - **Inferior:** 3.0 cm. - **Izquierdo:** 4.0 cm. - **Derecho:** 2.0 cm.
- **Tipo de Fuente:** Arial - **Tamaño:** 12 - **Interlineado:** 1.5

**Citas y Referencias:** APA 6th Edition (2010)
DESARROLLO DE APLICACIONES DE COMUNICACIÓN ENTRE DISPOSITIVOS MÓVILES BASADOS EN SISTEMAS OPERATIVOS ANDROID

JOHAN SEBASTIAN ZAPATA RESTREPO

UNIVERSIDAD DE SAN BUENAVENTURA SECCIONAL MEDELLÍN
FACULTAD DE INGENIERÍAS
INGENIERÍA DE SISTEMAS
MEDELLÍN
2013
DESARROLLO DE APLICACIONES DE COMUNICACION ENTRE DISPOSITIVOS MÓVILES BASADOS EN SISTEMAS OPERATIVOS ANDROID

JOHAN SEBASTIAN ZAPATA RESTREPO

Proyecto de grado presentado para optar al título de ingeniero de sistemas

Asesor
Carlos Arturo Castro Castro

UNIVERSIDAD DE SAN BUENAVENTURA SECCIONAL MEDELLÍN
FACULTAD DE INGENIERÍAS
INGENIERIA DE SISTEMAS
MEDELLIN
2013
<table>
<thead>
<tr>
<th>Capítulo</th>
<th>Título</th>
<th>Página</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>INTRODUCCION</td>
<td>6</td>
</tr>
<tr>
<td>2</td>
<td>PLANTEAMIENTO DEL PROBLEMA</td>
<td>8</td>
</tr>
<tr>
<td>3</td>
<td>JUSTIFICACION</td>
<td>9</td>
</tr>
<tr>
<td>4</td>
<td>OBJETIVO GENERAL</td>
<td>11</td>
</tr>
<tr>
<td>5</td>
<td>OBJETIVOS ESPECÍFICOS</td>
<td>11</td>
</tr>
<tr>
<td>6</td>
<td>DISEÑO METODÓLOGICO PRELIMINAR</td>
<td>12</td>
</tr>
<tr>
<td>7</td>
<td>MARCO REFERENCIAL</td>
<td>13</td>
</tr>
<tr>
<td>7.1</td>
<td>ESTADO DEL ARTE</td>
<td>13</td>
</tr>
<tr>
<td>7.2</td>
<td>TIPOS DE COMUNICACION</td>
<td>13</td>
</tr>
<tr>
<td>7.2.1</td>
<td>WIFI</td>
<td>13</td>
</tr>
<tr>
<td>7.2.2</td>
<td>BLUETHOOT</td>
<td>14</td>
</tr>
<tr>
<td>7.3</td>
<td>ANDROID</td>
<td>14</td>
</tr>
<tr>
<td>7.4</td>
<td>IDE ECLIPSE</td>
<td>15</td>
</tr>
<tr>
<td>8</td>
<td>MATRIZ DE COMPARACIÓN</td>
<td>16</td>
</tr>
<tr>
<td>9</td>
<td>ARQUITETURA DE HARDWARE DE UN SISTEMA GENERICO ANDROID</td>
<td>17</td>
</tr>
<tr>
<td>9.1</td>
<td>Arquitectura de Hardware Android</td>
<td>17</td>
</tr>
<tr>
<td>9.2</td>
<td>Descripción de cada componente del hardware</td>
<td>17</td>
</tr>
<tr>
<td>9.2.1</td>
<td>Kernel De Linux</td>
<td>17</td>
</tr>
<tr>
<td>9.2.2</td>
<td>Librerias</td>
<td>18</td>
</tr>
<tr>
<td>9.2.3</td>
<td>Entorno De Ejecución</td>
<td>18</td>
</tr>
<tr>
<td>9.2.4</td>
<td>Framework De Aplicaciones</td>
<td>19</td>
</tr>
<tr>
<td>9.2.5</td>
<td>Aplicaciones</td>
<td>20</td>
</tr>
<tr>
<td>10</td>
<td>ARQUITETURA DE SOFTWARE DE ANDROID</td>
<td>21</td>
</tr>
<tr>
<td>10.1</td>
<td>CICLO DE VIDA DE LOS ACTIVITIES</td>
<td>21</td>
</tr>
<tr>
<td>10.2</td>
<td>CICLO DE VIDA DE LOS SERVICIES</td>
<td>23</td>
</tr>
<tr>
<td>10.3</td>
<td>Diagrama De Clases De Android</td>
<td>24</td>
</tr>
<tr>
<td>10.4</td>
<td>Diagrama de secuencia de un sistema android</td>
<td>24</td>
</tr>
<tr>
<td>10.5</td>
<td>Eclipse y sus componentes</td>
<td>25</td>
</tr>
</tbody>
</table>
10.5.1 Estructura De Un Proyecto En Android................................................................. 26
11 METODOLOGIA DESARROLLO DE SOFTWARE ...................................................... 30
  11.1 MODELO.................................................................................................................. 31
  11.2 VISTA ..................................................................................................................... 31
  11.3 CONTROLADOR..................................................................................................... 31
12 PROYECTO AGV.......................................................................................................... 32
  12.1 SIMULAR LOS DATOS RECIBIDOS POR BLUETOOTH DE UN DISPOSITIVO AGV........... 37
    12.1.1 ENCUESTA DE COMUNICACIÓN....................................................................... 37
    12.1.2 DIAGRAMA DE CASOS DE USOS................................................................. 38
  12.2 DESARROLLO E IMPLEMENTACIÓN ....................................................................... 43
    12.2.1 CREAR CONEXIÓN CON AGV......................................................................... 43
13 DEFINICIÓN DE ACTIVIDADES EN CAPACITACIÓN..................................................... 51
14 COMPONENTES E INSTALACIÓN PARA PROGRAMAR PARA DISPOSITIVOS MOVILES CON
   SISTEMA OPERATIVO ANDROID...................................................................................... 52
  14.1 INSTALACIÓN ADT(ANDROID DEVELOPMENT TOOLS) EN ECLIPSE.................... 55
    14.1.1 CONFIGURACIÓN DEL SDK EN ECLIPSE....................................................... 55
  14.2 CAPACITACIÓN SEMILLERO.................................................................................... 56
15 CONCLUSIONES........................................................................................................... 57
16 REFERENCIAS BIBLIOGRÁFICAS................................................................................. 58
1 INTRODUCCION

En la sociedad actual, en continuo movimiento, los avances tecnológicos aparecen para dar respuesta a las necesidades de estar en continua conexión con la información y las comunicaciones. Es por esta razón por la que aparecen las tecnologías móviles que van a configurar un nuevo paradigma social, cultural y educativo. (Americas, 2012) Los cuales nos llevaran a la transformación de nuestro mundo, permitiéndonos estar conectados en todo momento y actualizados con todo tipo de información así como lo requiere la actualidad.

En el mundo actual nuestro desarrollo profesional se relaciona cada vez más con el uso de las tecnologías, de tal forma que se han posicionado como una herramienta de uso constante en el desempeño laboral. El avance en la ciencia ha reestructurado las formas en las que nos desarrollamos, ahora, las tareas diarias puedan complementarse fuera de una oficina. Una gran parte de los empleos se apoyan en la red como instrumento de trabajo, y dentro de un mundo que cambia a cada segundo, nos encontramos frente al reto de actualizarnos y adaptarnos a los novedosos implementos en el ámbito laboral. (Herrera, 2013)

Las necesidades presentadas actualmente nos dan la posibilidad de que por medio de un dispositivo móvil se puedan realizar transferencias de datos a otro dispositivo móvil, ya que por medio de diferentes tipos de comunicación existentes en la actualidad dentro de los dispositivos o también conocidos como Smartphone o tabletas, los cuales nos permitirán una búsqueda precisa de un servicio ofreciendo un soporte técnico que desarrolla una funcionalidad de búsqueda avanzada.

En cuanto a tecnologías existentes el poderle brindar a las pequeñas empresas productos al alcance de sus medidas a muy bajos costos es implantar nuevas oportunidades de negocio bajo tecnologías, las cuales nos ayudan a mejorar la productividad de nuestro país; en cuanto a estos avances tecnológicos como lo son los dispositivos móviles con sistema operativo androide el cual posee a nivel mundial un apogeo por su nivel de servicios y aplicaciones existentes

El desarrollo de la herramienta, generará beneficios significativos a los empresarios ya que tendrán la facilidad de ofertar sus productos y servicios mediante herramientas tecnológicas digitales que coadyuvan a la comercialización y la productividad del servicio.

En la actualidad existen varias tecnologías las cuales van aplicadas a todos los sectores industriales proporcionando información adecuada a las necesidades actuales del
mercado, pero no se cuenta con recursos necesarios para implementar estas nuevas tecnologías en nuestro país o para competir con países desarrollados en implementación de estas tecnologías.

En cuanto a desarrollo de aplicativos para dispositivos móviles para productos, existen varios aplicativos creados en la actualidad los cuales utilizan diferentes sistemas de comunicación con entre sí a través de varias conexiones conocidas como Bluetooth y WIFI.

En los últimos años se ha producido un perfeccionamiento y simplificación de la tecnología de comunicación por campo próximo NFC, el resultado ha sido la creciente aparición de gadgets o dispositivos móviles (teléfonos móviles inteligentes, Tablets-PC, PDAs, PCs, iPods/iPads, smartcards, cámaras digitales, kioskos, etc.) que añaden esa funcionalidad NFC y que pueden leer etiquetas RFID convencionales pasivas o RFID con BAP (con batería asistida, capaces de leer a mayor distancia) (Bertolín), los costos de implementación de estas herramientas para dispositivos móviles son costos bajos ya que solo se necesita poseer un dispositivo móvil con sistema operativo androide para su implementación.

Como consecuencia estos aplicativos permiten una interacción entre tecnologías ofreciendo productos al alcance de un dispositivo el cual puede ser adquirido por usuarios convencionales y permitiéndoles estar conectados constantemente dependiendo de cada necesidad tecnológica ofrecida por el mercado a costos increíbles

en el semillero de investigación en ingeniería del software de la universidad de san buenaventura de Medellín -SisUsbMed- se conformó un grupo de interés en el 2013 de programación de dispositivos móviles, motivados en capacitar a los estudiantes por las necesidades del mercado y en el hecho que muy pocos estudiantes sabían del tema y los que tenían un poco de conocimiento lo desarrollaban en su proyecto de grado, además, no existía esta temática de enseñanza en dispositivos móviles en las asignaturas de la carrera. Adicionalmente en el año 2012 se realizó un aplicativo para un vehículo guiado autónomo similar a un robot (AGV) el aplicativo lo realizo un estudiante que pertenecía al semillero para que este conocimiento no se perdiera surgió la idea de capacitar a integrantes del semillero y darle continuidad al proyecto.

Este proyecto está conformado de la siguiente manera: planteamiento del problema, justificación, objetivo general, objetivos específicos, diseño metodológico preliminar, marco referencial, conclusiones y bibliografía.
PLANTEAMIENTO DEL PROBLEMA

La necesidades actuales del mundo en cuanto a desarrolladores de software se encuentra en una brecha muy grande ya que no son tanto los ingenieros de sistemas que les guste programar, esto se debe a que desde la infancia no fueron inculcados conocimientos básicos de lógica de programación y dese ahí viene la dificultad de que los estudiantes universitarios se decidan a desarrollar software. Según Ana Bulnes, citando a Mark Zuckerberg (Bulnes, 2013) utiliza un argumento similar al de code.org, indicando que uno de los límites de crecimiento que tiene el sistema es que simplemente todavía no hay suficientes ingenieros con los conocimientos adecuados de programación para rellenar todos esos puestos de trabajo. Esta brecha de programación que hay a nivel mundial lleva a que no se estén encontrando intereses en la programación para dispositivos móviles con android ya que según estudios por la consultora Flurry hecha entre enero y mayo del año 2012 nos da una muestra considerable, que aunque no sea 100% segura nos da algunos datos que podemos tomar como interesantes. De esto se sacan datos interesantes, como que por cada aplicación que se han desarrollado para la plataforma de Google se han creado dos para terminales de la apple, una proporción que a gran escala es importante. El gran problema al que los desarrolladores encuestados y los analistas achacan es por el tema de la gran variedad de dispositivos Android, precisamente una de sus virtudes. iOS juega con un sistema en el que si funciona en un dispositivo seguramente funcione en todos los demás dispositivos, pero en Android la variedad es tan enorme que es difícil que algo que requiera algunos recursos funcione en todo terminal. Muchos desarrolladores se ven obligados moralmente y no técnicamente a optimizarlo todo al máximo en Android a todo terminal posible y ello hace que los gastos aumenten y los beneficios se vean lastrados (Naruedyoh, 2012).

 unos de los problemas más frecuentemente que se encuentran en la programación es que los desarrolladores de software son cada vez menos y sobre todo en un país donde se requiere alta tecnología como lo es Colombia, el cual representa el segundo sector más grande de tecnologías de información de América latina (Bertolín). este comportamiento se debe al crecimiento del sector de software que está compuesto por casi 3.000 empresas que emplean más de 32.000 trabajadores y generan 270 millones de dólares al año.
3 JUSTIFICACION

Code.org, una organización sin ánimo de lucro dedicada precisamente a difundir la necesidad (y proporcionar herramientas) de una mayor “alfabetización” en materia de programación informática, mantiene dos discursos paralelos: por un lado, defiende dirigiéndose a los niños que programar es divertido y que, si bien puede parecer algo intimidante al principio, enseguida se le coge el truco y se puede hacer cualquier cosa. El otro mensaje es el económico: en plena época de crisis, un millón de “los mejores puestos de trabajo” de Estados Unidos podrían quedarse sin cubrir porque solo uno de cada diez colegios enseñan programación. Cada vez hay más herramientas destinadas a enseñar a niños los principios de la programación, empresas que organizan cursos en colegios y lugares insospechados que de pronto se han convertido en el lugar al que todo el mundo mira para ver si es posible introducir estos programas en el currículo escolar y cómo funcionan.(Bulnes, 2013)

A nivel regional apps.co promueve cursos para aquellas personas que estén interesadas en aprender a programar para dispositivos móviles ya que por medio del bootcamps, el cual nos ofrece clases virtuales de programación para diferentes sistemas operativos tales como android, ios, html5 y windows, los cuales son gratuitos nos facilitan el aprendizaje para la creación de aplicaciones las cuales pueden ser ofertadas en el mercado para si lograr ingresos económicos los cuales nos ayuden a un mejoramiento y continuo aprendizaje para evitar una crisis a nivel de desarrolladores.

En Medellín se ofrecen convocatorias para presentar propuestas que sean innovadoras basadas en aplicaciones para dispositivos móviles en sistemas operativos android que faciliten alas empresas y personas comunes facilidad de acceso a información de cualquier tipo como lo ofrecen la gobemación de Antioquia y ruta n, esto para apropiar la tecnología disminuyendo la brecha que existe en la oferta de programadores puesto que se entiende que es un sistema operativo que es de orden mundial. Para evitar que la crisis aumente se ha decidido que en la san buenaventura se incluya varias maneras de capacitar a sus estudiantes en la creación de aplicaciones para dispositivos móviles, decidiendo realizar semilleros de investigación en android e incluyendo en su pensum una materia de programación para dispositivos móviles la cual será incluida dentro de su currículo para el próximo año. Apoyando así a la gestión tecnológica que el mundo provee como crisis dentro de los próximos años a nivel de desarrollo de software.

Es por esto que el éxito de android se debe a varios factores, pero uno de ellos y quizás el más importante es que es un sistema operativo gratuito y no solo eso, sino que su código fuente es libre. Al ser un sistema operativo gratuito y de código abierto, esto hace que los distintos fabricantes para móviles hayan adoptado esta base, como su sistema operativo preferido, entre otras cosas, ya no solo por el coste el cual es ninguno, sino también, porque pueden realizar personalizaciones, dando a cada fabricante su toque personal a la
interfaz de usuario, lo que les permite buscar puntos diferenciadores con la competencia. (eXentic, 2013)

Adicionalmente el país es el tercer mercado en crecimiento después de Brasil y México. Es por esto que se deben implementar herramientas que nos ayuden a atraer más desarrolladores para no permitir que se genere un crisis a nivel de software en Colombia. En Colombia se han desarrollado pocas aplicaciones para sistemas basados en sistemas operativos android es por esto que se toma la iniciativa de apoyo a emprendedores del negocio de las aplicaciones del Ministerio de Tecnologías de la Información y las Comunicaciones, Apps.co. Se crearon aplicaciones que ya se encuentran al final del proceso de ‘Consolidación’ y solo están a la espera de la validación del mercado, para que puedan ser ofrecidas en las distintas tiendas de aplicaciones. como parte de la solución del problema identificado debido a la escases de programadores, se han planteado caminos para ingresar al negocio del desarrollo de aplicaciones que están conformadas por los ‘Bootcamps’, donde los interesados pueden aprender la información básica para el desarrollo de aplicaciones en distintos lenguajes de programación, como Android, iOS, HTLM5 o Windows. Los cursos se realizan en modalidad virtual y son gratuitos. En Colombia se está logrando consolidar las aplicaciones para android es por esto que se buscan preparar personas para que este mercado tecnológico sea aprovechado en este país. (Mendez, 2010)

En Medellín ruta n una corporación creada por la alcaldía de Medellín, UNE y EPM promueve el desarrollo de negocios innovadores basados en tecnologías busca por medio de convocatorias personas interesadas en desarrollar aplicaciones para dispositivos móviles hace poco lanzaron en julio una convocatoria junto con otras empresas y universidades de Medellín como comfama, la universidad de Medellín las cuales buscaban conformar un grupo de personas que llevaran propuestas para construir aplicaciones para dispositivos móviles, las mejores propuestas se ganaría el derecho de aprender a programar en diferentes lenguajes de programación al lado de un experto en el tema de dispositivos móviles (rutan@rutanmedellin.org, 28)otra propuesta son las convocatorias realizadas por la gobernacion de antioquia como por ejemplo esta llamada quién se le mide las cual promueve la programacion de dispositivos moviles, para la construcion de aplicaciones que ayuden al mejoramiento de procesos dentro de las empresas con aplicaciones que faciliten a los usuario estar conectados con las empresas en todo momento. (Antioquia, 16)

Revisando la Oferta de Empleo de Colombia y buscando con la palabra clave desarrolladores móviles en fechas recientes se encontraron 2 ofertas de empleo en Medellín y 17 ofertas de empleo a nivel nacional (empleo, 2013), es importante saber que el mercado busca desarrolladores es por eso que se debe buscar los conocimientos necesarios para aprender a desarrollar para dispositivos móviles.
4 OBJETIVO GENERAL

Capacitar sobre la creación de aplicaciones para la comunicación entre dispositivos móviles basados en sistemas operativos android.

5 OBJETIVOS ESPECÍFICOS

Definir diferentes tipos de comunicación entre dispositivos Android
Definir arquitectura de componentes (Hardware y Software) de un sistema de comunicación genérico
Describir arquitectura de software multicapas para programar aplicativos Android
Definir el currículo de capacitación para el semillero de investigación
6  DISEÑO METODÓLOGICO PRELIMINAR

Fase 1: Definir diferentes tipos de comunicación entre dispositivos Android
1. Describir diferentes tipos de comunicación

Fase 2: Definir arquitectura de componentes (Hardware) de un sistema de comunicación genérico
1. Elaborar diagrama de componentes
2. Describir cada componente

Fase 3: Describir arquitectura de software multicapas para programar aplicativos Android
1. Diagrama de arquitectura de software
2. Diagrama de clases
3. Diagrama de secuencias básica
4. Describir metodología en eclipse
5. Describir metodología para el desarrollo de aplicativos para dispositivos móviles

Fase 4: Definir el currículo de capacitación para el semillero de investigación
1. Definir el plan de trabajo con los temas de capacitación
2. Crear contenidos como apoyo a la capacitación
3. Evaluar resultado de la capacitación

El diagrama de actividades estará en el anexo 3(diagrama de actividades).
En el comienzo de esta nueva era tecnológica y con la necesidades actuales del mercado al cual nos exponemos nacen nuevas tecnologías las cuales nos proporcionan mas agilidad y mayor eficiencia en la adquisición de productos o recursos los cuales ya podrán ser efectuados a través de un dispositivo móvil o Smartphone por medio de un sistema operativo llamado android el cual nos proporciona por medio de un sistema de comunicación una interconexión de datos entre dispositivos y nos proporciona mas agilidad a la hora de consulta de datos para un usuario convencional.

7.1 ESTADO DEL ARTE

La empresa viper desarrollo una aplicación para blackberry, ios y android que permite el control de todo un auto con un Smartphone, entre sus características se encuentra la activación de alarma del automóvil, así como el encendido del motor, control de ventanas, aire acondicionado entre muchas otras funciones.

Actualmente la empresa se encuentra desarrollando mejoras de su catalogo [http://www.viper.com/SmartStart/Features.aspx] de aplicaciones pero el proyecto ha tenido muy buena aceptación entre el público. (Trujillo, 2012)

7.2 TIPOS DE COMUNICACION

Existen diversos tipos de comunicación que se pueden utilizar en la creación de aplicaciones para dispositivos con sistema operativo android como son varios los tipos de comunicación solo nos enfocaremos en dos tipos de comunicación los cuales son los que se describirán a continuación.

7.2.1 WIFI

En el mundo de las telecomunicaciones existen una gran variedad de conexiones entre dispositivos y ordenadores los cuales nos acercan mas a la información en cualquier instante o momento, una de estas conexiones es el WIFI que es una de las tecnologías de comunicación inalámbrica mediante ondas más utilizada hoy en día. WIFI, también llamada WLAN (wireless lan, red inalámbrica) o estándar IEEE 802.11.

En la actualidad podemos encontrarnos con dos tipos de comunicación WIFI:

802.11b, que emite a 11 Mb/seg, y
802.11g, más rápida, a 54 MB/seg.

De hecho, su velocidad y alcance (unos 100-150 metros en hardware asequible) lo convierten en una fórmula perfecta, este tipo de comunicación puede ser utilizado como puente entre el Smartphone y el dispositivo receptor. (wifi)

7.2.2 BLUETOOTH
Bluetooth es una tecnología inalámbrica de ondas de radio de corto alcance (2.4 gigahertzios de frecuencia) cuyo objetivo es el simplificar las comunicaciones entre dispositivos informáticos, como ordenadores móviles, teléfonos móviles, otros dispositivos de mano y entre estos dispositivos e Internet. También pretende simplificar la sincronización de datos entre los dispositivos y otros ordenadores.

Permite comunicaciones, incluso a través de obstáculos, a distancias de hasta unos 10 metros. (juan, 2011)

7.3 ANDROID

Android es un sistema operativo para smartphones basado en Linux Kernel que incluye un SDK que proporciona los instrumentos y APIs necesarios para comenzar a desarrollar las aplicaciones sobre la plataforma de Android utilizando el lenguaje de programación Java. Android incluye un conjunto de bibliotecas de núcleo que proporciona la mayor parte de la funcionalidad disponible en las bibliotecas principales del lenguaje de programación Java. Este se basa en la versión 2.6 de Linux para los servicios básicos del sistema como la seguridad, la gestión de memoria, la dirección de proceso, la pila de red, y el modelo del controlador. El núcleo también actúa como una capa de abstracción entre el hardware y el resto del software. (Trujillo, 2012)

Android es una plataforma para dispositivos móviles desarrollada por Google. La cual proporciona un conjunto completo del desarrollo de software: el sistema operativo, herramientas y APIs necesarias para empezar a desarrollar aplicaciones [4]. La programación basada en Java que hace Android es ampliamente utilizada en el desarrollo de aplicaciones móviles. (Trujillo, 2012)

I. COMPONENTE DE LA APLICACIÓN

Los componente de la aplicación son la base fundamental de una aplicación android. Cada uno del componente es un lugar diferente en el que el sistema puede ingresar a la aplicación. No todos los componente son punto de entrada para el usuario y varios dependen de la funcionalidad de un padre para que funcionen los hijos, pero cada uno

II. ACTIVACION DE LOS COMPONENTES

Para poder activar los componentes es necesario tener claro que solo tres de cuatro componentes se activan por medio de un mensaje asíncrono que llama una intención. Una intención o un objeto Intent es la encargada de definir la acción a realizar, en algunos casos se puede iniciar una actividad para recibir un resultado, en este caso la actividad devuelve un resultado a una Intent. [http://developer.android.com/guide/components/fundamentals.html] (Trujillo, 2012)

III. ARCHIVO MANIFEST

Antes de que el sistema android inicie cualquier componente de la aplicación es necesario saber que existe el componente en el archivo AndroidManifest.xml. toda aplicación que se realice tiene por obligación que declarar que componentes del celular utilizara. [http://developer.android.com/guide/components/fundamentals.html] (Trujillo, 2012)

7.4 IDE ECLIPSE

La Plataforma Eclipse, es un entorno de desarrollo integrado (IDE) para todo, y nada en particular. Aunque esta Plataforma cuenta con una gran cantidad de funciones integradas, es posible extender su sistema. El plug-in es la menor unidad funcional de la Plataforma Eclipse, el cual puede ser desarrollado y entregado por separado. Todas las funcionalidades de Eclipse se entregan a través de la interface del plug-in, a excepción de un pequeño núcleo conocido como la plataforma de ejecución. Una característica simple, es que puede ser agregada a través de un único plug-in y desarrollarse sin ningún tipo de codificación. Un IDE completo para un lenguaje de programación se puede dividir en varios plug-in’s, cada uno codificado en Java. Así, las herramientas conectadas en el Plataforma, suministran las características específicas que realizan un desarrollo adecuado en los nuevos tipos de aplicaciones. (Trujillo, 2012)
Se definirá la matriz de comparación de los dos tipos de comunicación en el cual se utilizaran variables para definir cuál es el de mejor alcance y cual genera menor costo y nos ayudara a definir qué tipo de comunicación puede ser más eficiente y cual puede generar mejor impacto.

<table>
<thead>
<tr>
<th>VARIABLES</th>
<th>WIFI</th>
<th>BLUETHOTH</th>
</tr>
</thead>
<tbody>
<tr>
<td>Uso en dispositivos</td>
<td>Se utiliza como medio para conectarse a internet y para crear redes locales</td>
<td>Se utiliza para conectar dispositivos a alta velocidad</td>
</tr>
<tr>
<td>Alcance de conexión</td>
<td>Se utiliza para conexiones de mayor distancia</td>
<td>Se utiliza para conexiones de mejor alcance como de 10 metros</td>
</tr>
<tr>
<td>Costos manejo</td>
<td>El costo de wifi es mayor ya que se necesitan nodos de comunicación dependiendo de la distancia</td>
<td>El costo es menor ya que solo se necesita tener los dispositivos y estar a 10 metros de distancia como máximo</td>
</tr>
<tr>
<td>Tipo de comunicación</td>
<td>Red local o wireles</td>
<td>Peer to peer(comunicación directa)</td>
</tr>
</tbody>
</table>

Los dos tipos de comunicación sirven como puente de comunicación entre los diferentes tipos de dispositivos pero se puede analizar que es más efectivo utilizar la comunicación Bluetooth puesto que esta sirve para la comunicación entre dispositivos móviles y ordenadores de información a una alta velocidad mientras que la comunicación Wifi nos serviría para una conexión por medio de una red local y sus costos son mayores ya que se necesitan nodos de conexión a cierta distancia para la comunicación con el origen de la red.
9 ARQUITETURA DE HARDWARE DE UN SISTEMA GENERICO ANDROID

9.1 Arquitectura de Hardware Android

Este es el diagrama de componentes de un sistema genérico en android tanto de hardware como software

![Diagrama de Arquitectura Hardware Android](image)

Figura 1: Arquitectura Hardware Android

9.2 Descripción de cada componente del hardware

Para empezar con el desarrollo de aplicaciones en Android es importante conocer cómo está estructurado este sistema operativo. A esto le llamamos arquitectura y en el caso de Android está formada por varias capas que facilitan al desarrollador la creación de aplicaciones. Además, esta distribución permite acceder a las capas más bajas mediante el uso de librerías para que así el desarrollador no tenga que programar a bajo nivel las funcionalidades necesarias para que una aplicación haga uso de los componentes de hardware de los teléfonos.

Cada una de las capas utiliza elementos de la capa inferior para realizar sus funciones, es por ello que a este tipo de arquitectura se le conoce también como *pila*. (condesa, 2011)

9.2.1 Kernel De Linux

El núcleo del sistema operativo Android está basado en el kernel de Linux versión 2.6, similar al que puede incluir cualquier distribución de Linux, como Ubuntu, solo que adaptado a las características del hardware en el que se ejecutará Android, es decir, para dispositivos móviles.
El núcleo actúa como una capa de abstracción entre el hardware y el resto de las capas de la arquitectura. El desarrollador no accede directamente a esta capa, sino que debe utilizar las librerías disponibles en capas superiores. De esta forma también nos evitamos el hecho de quebrarnos la cabeza para conocer las características precisas de cada teléfono. Si necesitamos hacer uso de la cámara, el sistema operativo se encarga de utilizar la que incluya el teléfono, sea cual sea. Para cada elemento de hardware del teléfono existe un controlador (o driver) dentro del kernel que permite utiliarlo desde el software.

El kernel también se encarga de gestionar los diferentes recursos del teléfono (energía, memoria, etc.) y del sistema operativo en sí: procesos, elementos de comunicación (networking), etc. (condesa, 2011)

9.2.2 Librerías

La siguiente capa que se sitúa justo sobre el kernel la componen las bibliotecas nativas de Android, también llamadas librerías. Están escritas en C o C++ y compiladas para la arquitectura hardware específica del teléfono. Estas normalmente están hechas por el fabricante, quien también se encarga de instalarlas en el dispositivo antes de ponerlo a la venta. El objetivo de las librerías es proporcionar funcionalidad a las aplicaciones para tareas que se repiten con frecuencia, evitando tener que codificarlas cada vez y garantizando que se llevan a cabo de la forma “más eficiente”.

Entre las librerías incluidas habitualmente encontramos OpenGL (motor gráfico), Bibliotecas multimedia (formatos de audio, imagen y video), Webkit (navegador), SSL (cifrado de comunicaciones), FreeType (fuentes de texto), SQLite (base de datos), entre otras. (condesa, 2011)

9.2.3 Entorno De Ejecución

Como podemos apreciar en el diagrama, el entorno de ejecución de Android no se considera una capa en sí mismo, dado que también está formado por librerías. Aquí encontramos las librerías con la funcionalidades habituales de Java así como otras específicas de Android.

El componente principal del entorno de ejecución de Android es la máquina virtual Dalvik. Las aplicaciones se codifican en Java y son compiladas en un formato específico para que esta máquina virtual las ejecute. La ventaja de esto es que las aplicaciones se
compilan una única vez y de esta forma estarán listas para distribuirse con la total garantía de que podrán ejecutarse en cualquier dispositivo Android que disponga de la versión mínima del sistema operativo que requiera la aplicación.

Cabe aclarar que Dalvik es una variación de la máquina virtual de Java, por lo que no es compatible con el bytecode Java. Java se usa únicamente como lenguaje de programación, y los executables que se generan con el SDK de Android tienen la extensión .dex que es específico para Dalvik, y por ello no podemos correr aplicaciones Java en Android ni viceversa. (condesa, 2011)

9.2.4 Framework De Aplicaciones

La siguiente capa está formada por todas las clases y servicios que utilizan directamente las aplicaciones para realizar sus funciones. La mayoría de los componentes de esta capa son librerías Java que acceden a los recursos de las capas anteriores a través de la máquina virtual Dalvik. Siguiendo el diagrama encontramos:

1. Activity Manager. Se encarga de administrar la pila de actividades de nuestra aplicación así como su ciclo de vida.

2. Windows Manager. Se encarga de organizar lo que se mostrará en pantalla. Básicamente crea las superficies en la pantalla que posteriormente pasarán a ser ocupadas por las actividades.

3. Content Provider. Esta librería es muy interesante porque crea una capa que encapsula los datos que se compartirán entre aplicaciones para tener control sobre cómo se accede a la información.

4. Views. En Android, las vistas los elementos que nos ayudarán a construir las interfaces de usuario: botones, cuadros de texto, listas y hasta elementos más avanzados como un navegador web o un visor de Google Maps.

5. Notification Manager. Engloba los servicios para notificar al usuario cuando algo requiera su atención mostrando alertas en la barra de estado. Un dato importante es que esta biblioteca también permite jugar con sonidos, activar el vibrador o utilizar los LEDs del teléfono en caso de tenerlos.

6. Package Manager. Esta biblioteca permite obtener información sobre los paquetes instalados en el dispositivo Android, además de gestionar la instalación de nuevos paquetes. Con paquete nos referimos a la forma en que se distribuyen las aplicaciones Android, estos contienen el archivo .apk, que a su vez incluyen los archivos .dex con todos los recursos y archivos adicionales que necesite la aplicación, para facilitar su descarga e instalación.
7. Telephony Manager. Con esta librería podremos realizar llamadas o enviar y recibir SMS/MMS, aunque no permite reemplazar o eliminar la actividad que se muestra cuando una llamada está en curso.

8. Resource Manager. Con esta librería podremos gestionar todos los elementos que forman parte de la aplicación y que están fuera del código, es decir, cadenas de texto traducidas a diferentes idiomas, imágenes, sonidos o layouts. En un post relacionado a la estructura de un proyecto Android veremos esto más a fondo.

9. Location Manager. Permite determinar la posición geográfica del dispositivo Android mediante GPS o redes disponibles y trabajar con mapas.

10. Sensor Manager. Nos permite manipular los elementos de hardware del teléfono como el acelerómetro, giroscopio, sensor de luminosidad, sensor de campo magnético, brújula, sensor de presión, sensor de proximidad, sensor de temperatura, etc.

11. Cámara: Con esta librería podemos hacer uso de la(s) cámara(s) del dispositivo para tomar fotografías o para grabar vídeo.


9.2.5 Aplicaciones

En la última capa se incluyen todas las aplicaciones del dispositivo, tanto las que tienen interfaz de usuario como las que no, las nativas (programadas en C o C++) y las administradas (programadas en Java), las que vienen preinstaladas en el dispositivo y aquellas que el usuario ha instalado.

En esta capa encontramos también la aplicación principal del sistema: Inicio (Home) o lanzador (launcher), porque es la que permite ejecutar otras aplicaciones mediante una lista y mostrando diferentes escritorios donde se pueden colocar accesos directos a aplicaciones o incluso widgets, que son también aplicaciones de esta capa. (condesa, 2011)
10 ARQUITECTURA DE SOFTWARE DE ANDROID

Android es un sistema operativo basado en el núcleo Linux diseñado para dispositivos móviles (teléfonos inteligentes y tabletas). Fue desarrollado inicialmente por Android Inc., una empresa dedicada al desarrollo de software para teléfonos móviles. En julio de 2005 Google adquirió Android Inc y el 5 de noviembre de 2007 presentó el sistema operativo Android OS junto con el anuncio de la creación de la Open Handset Alliance, un consorcio de 78 compañías de hardware, software y telecomunicaciones dedicadas al desarrollo de estándares abiertos para dispositivos móviles. En octubre del 2008 se puso a la venta el primer dispositivo móvil con este sistema operativo, el G1 fabricado por HTC. Desde entonces ha aparecido en el mercado un gran número de dispositivos móviles con este sistema operativo y se han publicado numerosas actualizaciones. Estas actualizaciones arreglan bugs y agregan nuevas funciones. La última versión de Android es la 4.0, que introduce mejoras en la conectividad y en la interfaz gráfica. (google, 2011)

En Android una aplicación consiste normalmente en un conjunto de componentes. Existen cuatro tipos de componentes:

Activities: Una actividad representa una única pantalla con una interfaz de usuario. Por ejemplo, un editor de texto podría tener una actividad para editar un documento, otra actividad para mostrar la lista de documentos, y otra para generar una vista previa del documento.

Servicios: Un servicio es un componente que se ejecuta en segundo plano para realizar alguna tarea. Los servicios no disponen de interfaz de usuario.

Content Providers: Componente que guarda y recupera datos y los hace accesibles a todas las aplicaciones. Es la única forma de compartir datos entre aplicaciones.

Broadcast receivers: Este tipo de componente responde a evento del sistema tales como, apagado de pantalla, batería baja, etc. Aunque los broadcast receivers no muestran una interfaz de usuario pueden crear una notificación en la barra de estado para avisar cuando ocurre un evento.

10.1 CICLO DE VIDA DE LOS ACTIVITIES

En Android, las actividades y los servicios tienen un ciclo de vida. En cada ciclo de vida de una actividad se invocan los siguientes manejadores
Figura 2: Ciclo de vida de una actividad (google, 2011)

onCreate(): Este método es ejecutado la primera vez se crea la actividad.

onRestart(): Este método es ejecutado después de que la actividad haya sido detenida.

Onstart(): Este método se ejecuta antes de que la actividad sea visible al usuario.

onResume(): Este método se ejecuta justo antes de que la actividad empiece a interaccionar con el usuario.

onPause(): Este método es ejecutado cuando se está empezando a reanudar otra actividad.

onStop(): Este método se ejecuta cuando la actividad ya no va a ser visible al usuario. Esto puede suceder cuando la actividad va a ser destruida o cuando se esté reanudando otra actividad.
onDestroy(): Este método es el último que se ejecuta antes de que la actividad finalice

10.2 CICLO DE VIDA DE LOS SERVICIES

En el caso de los servicios los métodos ejecutados son:

OnCreate(): Cuando el servicio se está creando,
onStartCommand(): El servicio está comenzando debido a la llamada del método.
startService(): por parte de otro componente de la aplicación.
onBind(): Un cliente solicita el servicio.
onUnbind(): Un cliente decide dejar de utilizar el servicio.
onReBind(): Un cliente que dejó de utilizar el servicio vuelve a solicitar usar el servicio.
onDestroy(): El servicio no va a ser usado más y es destruido.

Figura 3: Ciclo de vida de un servicio (google, 2011)
10.3 Diagrama De Clases De Android

Se definirá el sistema de clases que componen un sistema android, solo se definirán las clases genéricas de un sistema android ya que para la construcción de un sistema se deberán definir estas clases más las que se necesiten en cada sistema.

![Diagrama de Clases de Android](image)

Figura 4: Diagrama de clases de un sistema android (eXentic, 2013)

10.4 Diagrama de secuencia de un sistema android

Se definirá el diagrama de secuencia de un sistema android genérico. solo se definirán los componentes de android para la construcción de un sistema completo se deberán definir los demás componentes de la aplicación mas estos que se definirán a continuación.
La Plataforma Eclipse, es un entorno de desarrollo integrado (IDE) para todo, y nada en particular. Aunque esta Plataforma cuenta con una gran cantidad de funciones integradas, es posible extender su sistema. El plug-in es la menor unidad funcional de la Plataforma Eclipse, el cual puede ser desarrollado y entregado por separado. Todas las funcionalidades de Eclipse se entregan a través de la interface del plug-in, a excepción de un pequeño núcleo conocido como la plataforma de ejecución. Una característica simple, es que puede ser agregada a través de un único plug-in y desarrollarse sin ningún tipo de codificación. Un IDE completo para un lenguaje de programación se puede dividir en varios plug-in’s, cada uno codificado en Java. Así, las herramientas conectadas en el Plataforma, suministran las características específicas que realizan un desarrollo adecuado en los nuevos tipos de aplicaciones. (JulienGuyon, 17)

El ADT (Herramientas de desarrollo de Android) es un plugin para el IDE de Eclipse el cual está diseñado para proporcionar un contexto más potente e integrando para llevar a cabo la creación de aplicaciones en Android. (Trujillo, 2012)

El ADT amplía las capacidades de Eclipse para que pueda configurar rápidamente nuevos proyectos de Android, creando así una aplicación de usuario, la cual agrega unos paquetes basados en el marco API de Android, y clasifica sus aplicaciones utilizando las herramientas del SDK de Android. (Trujillo, 2012)

Eclipse se entrega con una herramienta para Java. Las herramientas de desarrollo Java (JDT) posee programas que tienen la capacidad de desarrollar la Plataforma, así como cualquier versión completa de Java IDE (gestión de proyectos, editor de la fuente de código, las búsquedas complejas, recopilación incremental, el apoyo de depuración, etc.)
Eclipse también es compatible con una C / C ++ IDE equivalente: la CDT. Todas estas herramientas se han desarrollado utilizando la infraestructura Eclipse API, el cual es un proceso abierto y documentado. (Trujillo, 2012)

Los lenguajes orientados a objetos son muy populares, sin embargo, aún no tienen características de gran alcance como la que proporciona la combinación de patrones expresivos gracias a los lenguajes de programación funcionales. En la práctica, estas características interactúan vagamente con los mecanismos de abstracción de datos que son fundamentales para los lenguajes orientados a objetos. (Trujillo, 2012)

La Plataforma Eclipse tiene la ventaja de ser muy simple, permitiendo así un entorno más favorable para promover un lenguaje y desarrollo más fácil y rápido. La plataforma genera respuestas a los problemas que otras instalaciones complejas poseen, proporcionando un entorno integrado que contiene dos tipos de editores, compiladores y constructores. Además, es capaz de suministrar un editor completo y especializado, por ejemplo para TOM, ofreciendo una sintaxis de color, palabras clave, y reportes de errores en un entorno uniforme. Por último, Eclipse también ofrece soporte para la clasificación y depuración buscando reutilizar dichos elementos para integración de TOM. (Trujillo, 2012)

La plataforma Eclipse tiene un proceso definido de respuestas automáticas, el cual se activa cada vez que un recurso es modificado. El flujo de trabajo de recolección muestra cómo este proceso puede ser utilizado para compilar automáticamente un proyecto: cada vez que se modifica un recurso, sea cual sea, tiene que volver a compilarse. Además, cuando una definición se ha modificado, dependiendo del Los archivos, también tienen que ser re-compilados. Y Para completar el proceso de construcción, todos los archivos generados de Java tienen que ser compilados.

Cuando se utiliza un entorno de desarrollo integrado, una de las características más importante es la posibilidad de visualizar los errores de programación. Este es, sin lugar a dudas, un punto muy fuerte que proporciona el IDE de Eclipse. Además tiene la ventaja de tener un proceso sencillo a la hora de desarrollar los asistentes y los procesos de construcción automáticos. (Trujillo, 2012)

Al proporcionar un proceso de generación automática y un buen manejo de la gestión en el mecanismo de errores, el presente plug-in de Eclipse logra simplificar considerablemente el desarrollo de programas basados en TOM. (Trujillo, 2012)

10.5.1 Estructura De Un Proyecto En Android

Esta es la estructura de un proyecto de android en eclipse como se observa en la imagen
Se explicará continuación cada carpeta y componente de un proyecto en android y se realizará un hola mundo como muestra con su respectivo código.

1. Assets: En este directorio se pone cualquier tipo de fichero externo que sea necesario por la aplicación que se esté ejecutando. Por ejemplo: HTML para el navegador sin necesidad de conexión, tipografías, jars de terceros. Es un repositorio de archivos.

2. Bin: Los binarios compilados de la aplicación.

3. Res: Los recursos, formada por 3 directorios: – Drawable: ficheros de imágenes. – Layout: Ficheros de diseño de las interfaz de usuario. Ficheros xml que describen las interfaces. – Values: definición de variables. Ficheros xml en cada uno de ellos constantes, se agrupa por temática, es decir: definición de colores, definición de texto...
4. Gen: Se almacenan los archivos auto-generados por el compilador de Android. Contiene el archivo “R.java”, que es un índice a todos los recursos definidos en el proyecto.

5. Src: Aquí va todo el código Java de la aplicación.

6. project .classpath: ficheros necesarios para abrir el proyecto con Eclipse.

7. AndroidManifest.xml: es un archivo requerido para cada aplicación. Describe los valores globales de su paquete, incluida la aplicación, componentes (actividades, servicios, etc), las clases para cada uno de los componentes, qué tipo de datos puede manejar cada uno, y donde puede ser lanzado.

```xml
<manifest version="1.0" encoding="utf-8">
  <manifest xmlns:android="http://schemas.android.com/apk/res/android"
    package="com.example.helloAndroid"
    android:versionCode="1"
    android:versionName="1.0">
    <application android:icon="@drawable/icon" android:label="@string/app_name">
      <activity android:name=".helloAndroid"
        android:label="@string/app_name">
        <intent-filter>
          <action android:name="android.intent.action.MAIN"/>
          <category android:name="android.intent.category.LAUNCHER"/>
        </intent-filter>
      </activity>
    </application>
  </manifest>
</manifest>
```

**Figura 7: Archivo Android Manifest**

- **Application**: android:icon="drawable resource" ? Se define el icono de la aplicación. Se mostrará cuando la instalación esté instalada en el dispositivo. android:name="string" ? A través de este atributo se declara el nombre de la aplicación, aparecerá debajo del icono al instalar. android:theme="resource or theme" ? Podemos crear un tema general y usar en toda la aplicación. activity, service, provider… Cada Activity, Service o Content Provider que creamos en nuestra aplicación debe de estar aquí. En caso contrario, la aplicación no los va a reconocer.

- **Permissions**: Dentro de la etiqueta <manifest> podemos añadir el nodo <users-permission>. Se utiliza para definir las partes a las que la aplicación necesita acceder y los permisos de los usuarios. -Recibir SMS -Capturar imágenes desde la cámara -hacer llamadas -Acceder a internet… Cuando se instalan aplicaciones desde el Android Market, algunas tienen un listado de permisos, donde el usuario debe permitirlos o aceptarlos si se quiere instalar la aplicación en el dispositivo.
8. **Version:** La etiqueta `<uses-sdk>`, se define con qué versión del SDK la aplicación fue creada, con lo que se puede definir una compatibilidad hacia versiones anteriores.

**Ejemplo de código** y visualización de cómo se vería una aplicación en Eclipse de un `hola mundo`

```java
package com.example.helloAndroid;

import android.app.Activity;
import android.os.Bundle;

public class helloAndroid extends Activity {
    /** Called when the activity is first created. */
    @Override
    public void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.main);
    }
}
```

![Imagen Inicial](condesa, Androideity, 2011)

```java
package com.example.helloAndroid;

import android.app.Activity;
import android.os.Bundle;
import android.widget.TextView;

public class helloAndroid extends Activity {
    /** Called when the activity is first created. */
    @Override
    public void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        TextView tv = new TextView(this);
        tv.setText("Hola Mundo");
        setContentView(tv);
    }
}
```

![Imagen Final](condesa, Androideity, 2011)

**Figura 8:** Ejemplo De Código De Un Hola Mundo (condesa, Androideity, 2011)

**Figura 9:** Maquina Virtual Con El Resultado (condesa, Androideity, 2011)

El resultado es el siguiente.
Para el desarrollo de aplicaciones en Android se define una estructura de 3 capas como lo es el modelo de vista controlador cuya principal bondad consiste en separar los datos de una aplicación, la interfaz de usuario y la lógica de negocios en tres componentes distintos que se relacionarán para al final tener como resultado nuestra aplicación.

Figura 10: Modelo vista controlador (condesa, Androideity, 2012)
11.1 MODELO

Nos referimos con modelo a las representaciones que construiremos basadas en la información con la que operará nuestra aplicación. En Java, el modelo viene siendo análogo a los beans que tienen la particularidad de ser reutilizables y nos ayudan a cumplir con el proverbio de oro “Don’t Repeat Yourself” (DRY) haciendo a nuestras aplicaciones escalables. En esta parte del modelo también juega la decisión de qué modelo para almacenar información utilizaré. ¿Base de datos? ¿Web services? El modelo que elijas depende obviamente de las necesidades de información de tu aplicación. (condesa, Androideity, 2012)

11.2 VISTA

La vista no es más que la interfaz con la que va a interactuar el usuario. En Android, las interfaces las construimos en XML. Suelo utilizar mucho la analogía de que esta parte es realmente parecida a lo que hacemos en el desarrollo web con los CSS. Construimos nuestro esqueleto en XML que equivale al HTML de un sitio. Posteriormente, con ayuda de estilos, que también los escribimos en XML, podemos empezar a darle formato de colores, posiciones, formato, etc. a nuestro esqueleto. Esto equivale a los CSS. Si vienes de un ambiente web, aprovecha ese conocimiento para aplicarlo a crear aplicaciones Android. (condesa, Androideity, 2012)

11.3 CONTROLADOR

Finalmente nos topamos con el controlador que son todas esas clases que nos ayudarán a darle vida a esas interfaces bonitas que ya construimos y nos permitirán desplegar y consumir información de/para el usuario. Estos controladores se programan en lenguaje Java y son el core de la aplicación. (condesa, Androideity, 2012)

Para empezar a aplicar esta arquitectura debes saber también su flujo: (condesa, Androideity, 2012)

1. Todo parte cuando el usuario interactúa con tu aplicación, el usuario actual es la vista.
2. El controlador recibe la notificación de la acción solicitada.
3. El modelo es llamado para ser modificado.
4. Una vez que tenemos la publicación que quiere hacer el usuario, el controlador nuevamente toma partida para llamar a la vista correcta que desplegará la vista actualizada.
5. ¡Listo! El usuario ya tiene la nueva interfaz para seguir interactuando con la aplicación y volver a iniciar el ciclo cuando solicite otra acción.
El proyecto AGV (vehículo autónomo guiado) fue desarrollado a partir de una necesidad que se requiere en la universidad de San Buenaventura de que por medio de un dispositivo móvil se pueda controlar un robot que nos busque libros dentro de la biblioteca de la universidad. Esta es la estructura del proyecto.

La plataforma Android es compatible con la pila de red Bluetooth, que permite a un dispositivo intercambiar datos de forma inalámbrica con otros dispositivos Bluetooth. El marco de aplicación permite el acceso a la funcionalidad Bluetooth a través de la API de Android Bluetooth. Estas API permiten que las aplicaciones se conecten de forma inalámbrica a otros dispositivos Bluetooth, permitiendo realizar conexiones punto a punto y multipunto, realizando funciones inalámbricas. (Trujillo, 2012)

Utilizando el API del Bluetooth, una aplicación para Android puede realizar las siguientes actividades:
• Analizar en busca de otros dispositivos Bluetooth
• Consultar el adaptador Bluetooth local para dispositivos Bluetooth vinculados
• Establecer canales RFCOMM
• Conectarse a otros dispositivos a través de descubrimiento de servicios
• Transferencia de datos hacia y desde otros dispositivos
• Administrar conexiones múltiples (Trujillo, 2012)

Figura 12: Este es un ejemplo de 2 tipos de dispositivos conectándose por medio del bluetooth. (Trujillo, 2012)

En todo proceso de conexión es necesario saber quién va ser el receptor de la información que se va enviar. La búsqueda de dispositivos es un procedimiento de exploración que investiga en el área dispositivos Bluetooth habilitados solicitando información sobre cada uno de ellos. Sin embargo, un dispositivo Bluetooth responderá a una solicitud de descubrimiento sólo si está habilitado para ser visible. Si un dispositivo se puede detectar, responderá a la petición de descubrimiento mediante el intercambio de cierta información, como por ejemplo el nombre del dispositivo, la clase, y su dirección MAC única. Con esta información, el dispositivo realiza el descubrimiento y puede entonces elegir si desea iniciar una conexión con el dispositivo descubierto. (Trujillo, 2012)
Antes de realizar una búsqueda de dispositivos disponibles en el área, consultamos los dispositivos emparejados para visualizar si el dispositivo al que se desea conectar el usuario está en la lista de ya conocidos. (Trujillo, 2012)

Para iniciar el descubriendo de nuevos dispositivos solo es necesario llamar el `startdiscovery()`. el proceso de descubrimiento es asincrónico y el método devolverá inmediatamente con un valor booleano que indica si el descubrimiento se ha iniciado con éxito. (Trujillo, 2012)

```java
private void doDiscovery() {
    if (DBG) Log.d(TAG, "doDiscovery()";

    // Indicate scanning in the title
    setProgressBarIndeterminateVisibility(true);
    setTitle(R.string.scanning);

    // Turn on sub-title for new devices
    findViewById(R.id.title_new_devices).setVisibility(View.VISIBLE);

    // If we're already discovering, stop it
    if (mBtAdapter.isDiscovering()) {
        mBtAdapter.cancelDiscovery();
    }

    // Request discover from BluetoothAdapter
    mBtAdapter.startDiscovery();
}
```

La aplicación se debe registrar con el fin de recibir información de cada dispositivo detectado. (Trujillo, 2012)
La comunicación entre el Smartphone y el AGV se realizara por medio de una conexión bluetooth. Dentro del aplicativo android se llamara consumidor conexión CDB (Consumidor de servicios bluetooth) que se encargara de verificar si el dispositivo AGV cumple con las características necesarias para que el CDB cree un canal de comunicación entre ambos dispositivos. (Trujillo, 2012)

Si el canal de conexión es aprobado por el CDB ambos dispositivos están sincronizados y listo para la trasmisión de datos. La conexión entre ambos dispositivos es una conexión peer to peer que permite que ambos dispositivos sean cliente y servidor dependiendo de las necesidades de la interacción entre ambos. (Trujillo, 2012)

La aplicación control AGV cuenta con un web service KSOAP que permite la conexión del Smartphone y la nube haciendo peticiones a servicios publicados en ella. (Trujillo, 2012)

El wsdl publicado por el servidor permite realizar las búsquedas de los datos solicitados por el Smartphone al servidor evitando saturar el teléfono con datos que un futuro será usado para auditar el sistema. (Trujillo, 2012)
Figura 14: Diagrama de comunicación de cada uno de los dispositivos del sistema (Trujillo, 2012)

Figura 15: Diagrama de comunicación por web service entre el Smartphone y el servidor (Trujillo, 2012)
12.1 SIMULAR LOS DATOS RECIBIDOS POR BLUETOOTH DE UN DISPOSITIVO AGV

Para poder simular los datos recibidos es necesario investigar realmente que variables se van a utilizar para la comunicación, por eso es necesario realizar una encuesta que permitirá identificar que se va hacer y cómo se debe hacer. (Trujillo, 2012)

12.1.1 ENCUESTA DE COMUNICACIÓN

- ¿Qué es un AGV?

Los vehículos guiados autónomos son un tipo especial de robots que se ocupan tradicionalmente en el medio industrial de tareas como la recolección, transporte y ubicación de elementos (piezas, mercancías, cargas, etc.), sin la necesidad de la mediación directa de un operario. (Trujillo, 2012)

- ¿Qué datos serán enviados desde el AGV al dispositivo móvil?

Para poder tener una verdadera auditoría y control del AGV es necesario que se envíen datos de localización (latitud, longitud), estado del dispositivo, cantidad de objetos recolectados. (Trujillo, 2012)

- ¿Cómo quiere visualizar los datos en el dispositivo móvil?

Para tener una verdadera idea de donde se encuentra el dispositivo AGV mientras realiza tareas de recolección es necesario representar su posición en un mapa representativo del área en función. (Trujillo, 2012)

- ¿Qué proyecto futuros se tiene con esta misma plataforma?

Conexión a un servidor para el proceso de triangulación, también pretendemos utilizar la triangulación directamente desde el AGV al incorporarle un módulo GPS que permita obtener datos geográficos. (Trujillo, 2012)
12.1.2 DIAGRAMA DE CASOS DE USOS

Figura 16: Diagrama caso de uso alto nivel (Trujillo, 2012)

Figura 17: Diagrama de caso de uso extendido e incluidos (Trujillo, 2012)
### Nombre
Validar ingreso

### Fecha
junio 14 2012

### Versión
1.0

### Actores
Usuario.

### Propósito
Control de ingreso aplicativo.

### Resumen
Este caso de uso comienza cuando el usuario inicia la aplicación instalada en el Smartphone.

### Flujo Principal

<table>
<thead>
<tr>
<th>Eventos Actor</th>
<th>Eventos Sistema</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. Ingresar al aplicativo.</td>
<td>1.1. El sistema presenta el pantallazo inicial donde se debe identificar el actor.</td>
</tr>
<tr>
<td>2. Ingresar Nombre de Usuario y Contraseña.</td>
<td>2.1. El sistema verifica si los datos ingresados corresponden a un usuario del sistema.</td>
</tr>
<tr>
<td>3. Visualiza el mensaje de error o de aceptación que generó el aplicativo.</td>
<td>3.1. Devuelve mensajes de verificación y validación del usuario mostrando el pantallazo respectivo al aplicativo.</td>
</tr>
</tbody>
</table>

### Flujo Alternativo
Si el Usuario y Contraseña son incorrectos, el sistema genera mensaje de error para que el usuario corrija la información ingresada.

### Nota

### Responsable
Juan David Trujillo robayo.

---

### Nombre
Activar Bluetooth

### Fecha
junio 14 2012
<table>
<thead>
<tr>
<th>Versión</th>
<th>1.0</th>
</tr>
</thead>
<tbody>
<tr>
<td>Actores</td>
<td>Usuario.</td>
</tr>
<tr>
<td>Propósito</td>
<td>Activar el dispositivo bluetooth.</td>
</tr>
<tr>
<td>Precondición</td>
<td>Se debe ejecutar el caso de uso Validar usuario.</td>
</tr>
<tr>
<td>Resumen</td>
<td>Este caso de uso comienza cuando el usuario inicia sesión con su respectivo nombre de usuario y contraseña que lo acredite como tal.</td>
</tr>
</tbody>
</table>

**Flujo Principal**

<table>
<thead>
<tr>
<th>Eventos Actor</th>
<th>Eventos Sistema</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. visualiza la alerta generada por el aplicativo.</td>
<td>1.1. El aplicativo dispara una alerta para pedirle al usuario la aprobación de inicio del bluetooth</td>
</tr>
</tbody>
</table>
| 2. el usuario acepta o declina la solicitud que el aplicativo proporciona. | 2.1. si el usuario acepta la activación del bluetooth continua el caso de uso.  
2.2 si el usuario declina la petición de la aplicación el aplicativo finaliza terminando el caso de uso. |

**Nota**

Responsable: Juan David Trujillo Robayo.

**Nombre**

Hacer visible

**Fecha**

junio 14 2012

**Versión**

1.0

**Actores**

Usuario.

**Propósito**

Permitir ver el Smartphone visible para el AGV.

**Precondición**

Se debe ejecutar el caso de uso Activar bluetooth.

**Resumen**

Este caso de uso comienza cuando el usuario presiona el
<table>
<thead>
<tr>
<th>Flujo Principal</th>
<th>Eventos Actor</th>
<th>Eventos Sistema</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. el actor da clic en el botón de hacer visible.</td>
<td>1.1. El aplicativo dispara una confirmación para pedirle al usuario la aprobación de ejecución.</td>
<td></td>
</tr>
<tr>
<td>2. el usuario acepta o declina la solicitud que el aplicativo proporciona.</td>
<td>2.1. Si el usuario acepta el dispositivo entra en estado visible para el AGV.</td>
<td>2.2 si el usuario declina la petición de la aplicación regresa a la pantalla principal.</td>
</tr>
</tbody>
</table>

**Nota**

**Responsable** Juan David Trujillo Robayo.

---

<table>
<thead>
<tr>
<th>Nombre</th>
<th>Conectar con dispositivo</th>
</tr>
</thead>
<tbody>
<tr>
<td>Fecha</td>
<td>junio 14 2012</td>
</tr>
<tr>
<td>Versión</td>
<td>1.0</td>
</tr>
<tr>
<td>Actores</td>
<td>Usuario.</td>
</tr>
<tr>
<td>Propósito</td>
<td>Conectar el Smartphone con un dispositivo AGV.</td>
</tr>
<tr>
<td>Precondición</td>
<td>Se debe ejecutar el caso de uso Activar bluetooth.</td>
</tr>
<tr>
<td>Resumen</td>
<td>Este caso de uso comienza cuando el usuario presiona el botón de conectar un dispositivo.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Flujo Principal</th>
<th>Eventos Actor</th>
<th>Eventos Sistema</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. el actor da click en el botón conectar un dispositivo.</td>
<td>1.1. El aplicativo despliega en la pantalla los dispositivos ya vinculados con el Smartphone y un botón de búsqueda de nuevos dispositivos.</td>
<td></td>
</tr>
<tr>
<td>2. el actor elige un dispositivo vinculado.</td>
<td>2.1. el aplicativo se conecta automáticamente si el dispositivo se encuentra disponible.</td>
<td></td>
</tr>
</tbody>
</table>
3. el actor presiona el botón de búsqueda de nuevos dispositivos

| 3.1. se ejecuta el caso de uso buscar nuevos dispositivos. |

**Nota**

**Responsable** Juan David Trujillo robayo.

---

**Nombre** buscar nuevos dispositivos

**Fecha** junio 14 2012

**Versión** 1.0

**Actores** Usuario.

**Propósito** Buscar nuevos dispositivos en el área.

**Precondición** Se debe ejecutar el caso de uso Activar bluetooth y Conectar con dispositivo.

**Flujo Principal**

<table>
<thead>
<tr>
<th>Eventos Actor</th>
<th>Eventos Sistema</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. el actor da clic en el botón buscar nuevos dispositivos.</td>
<td>1.1. El aplicativo inicia la búsqueda de nuevos dispositivos e imprime la lista de los encontrados.</td>
</tr>
<tr>
<td>2. el actor elige el dispositivo al cual desea conectarse.</td>
<td>2.1. El aplicativo imprime una alerta de solicitud de sincronización, solicitando el pin de conexión.</td>
</tr>
<tr>
<td>3. el actor digita el pin de conexión correspondiente al dispositivo.</td>
<td>3.1 el aplicativo confirma los datos y establece una conexión con el dispositivo.</td>
</tr>
</tbody>
</table>

**Flujo Alternativo** Si el pin no es correcto el aplicativo retorna a la pantalla de inicio y no establece ninguna conexión.

**Nota**

**Responsable** Juan David Trujillo robayo.

---

**Nombre** Consultar

**Fecha** junio 14 2012

**Versión** 1.0

**Actores** Usuario.

**Propósito** Conocer los datos que el AGV envía.

**Precondición** Se debe ejecutar el caso de uso Activar bluetooth y Conectar con dispositivo.

**Resumen** Este caso de uso comienza cuando se conecta con un dispositivo.

**Flujo Principal**

<table>
<thead>
<tr>
<th>Eventos Actor</th>
<th>Eventos Sistema</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. el actor ingresa a la pantalla principal de la aplicación.</td>
<td>1.1. El aplicativo recibe datos provenientes del AGV e imprime la información.</td>
</tr>
<tr>
<td>2. El actor visualiza la</td>
<td></td>
</tr>
</tbody>
</table>

---
Expuestos los objetivos y diagramas de caso de uso de la aplicación, a continuación se desmenuzan los aspectos relacionados con su implementación.

La explicación general de cómo está diseñada la aplicación esperan ayudar a otros desarrolladores que quieren ampliar su capacidades de programación en lenguajes como android, o incluso generar nuevas ideas para la construcción de aplicaciones que se basen en fundamentos iguales o parecidos a la desarrollada en este proyecto. (Trujillo, 2012)

12.2 DESARROLLO E IMPLEMENTACION

12.2.1 CREAR CONEXIÓN CON AGV

- PERMISOS DEL BLUETOOTH

Para poder crear una conexión con un AGV es necesario que la aplicación declare los permisos correspondientes para poder solicitar cualquier acción como la de solicitar una conexión, aceptar una conexión etc. (Trujillo, 2012)

Este es un ejemplo de cómo se debe declara el permiso en el archivo AndroidManifest.xml de android. (Trujillo, 2012)

```xml
<uses-permission android:name="android.permission.BLUETOOTH_ADMIN"/>
<uses-permission android:name="android.permission.BLUETOOTH"/>
```

- CONFIGURACION BLUETOOTH

Antes de que la aplicación inicie y pueda establecer una comunicación con el bluetooth, se debe verificar que realmente el Smartphone donde se va ejecutar tenga bluetooth. (Trujillo, 2012)
Si la aplicación detecta que el dispositivo no es compatible o no contiene un bluetooth la aplicación debe cerrar, pero si sucede el caso contrario y se encuentra desactivado es necesario solicitar al usuario activar el bluetooth para que la aplicación continúe sin ningún problema. (Trujillo, 2012)

```java
public void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    if(D) Log.e(TAG, "+++ ON CREATE +++");

    //carga el diseño de la aplicacion
    setContentView(R.layout.main);

    //Obtener el adaptador bluetooth local
    mBluetoothAdapter = BluetoothAdapter.getDefaultAdapter();

    //Verifica si el adaptador si existe o es compatible
    if (mBluetoothAdapter == null) {
        Toast.makeText(this, "Bluetooth is not available", Toast.LENGTH_LONG).show();
        finish();
        return;
    }
}
```

Figura 18: Ejemplo de verificación de existencia de bluetooth en el Smartphone (Trujillo, 2012)

```java
public void onStart() {
    super.onStart();
    if(D) Log.e(TAG, "++ ON START ++");

    //SI EL BLUETOOTH NO ESTA HABILITADO
    if (!mBluetoothAdapter.isEnabled()) {
        Intent enableIntent = new Intent(BluetoothAdapter.ACTION_REQUEST_ENABLE);
        //SE GENERA LA VENTANA DE APROVACION AL USUARIO
        startActivityForResult(enableIntent, REQUEST_ENABLE_BT);
        //De lo contrario, la configuración de la sesión de chat
    } else {
        if (mChatService == null) setupChat();
    }
}
```

Figura 19: Ejemplo de activación del bluetooth por medio de una ventana de confirmación (Trujillo, 2012)
Figura 20: Vista real de la aplicación, Mensaje de solicitud de activación. (Trujillo, 2012)

- **BÚSQUEDA DE DISPOSITIVOS EN EL AREA**

Con el bluetooth activado se puede realizar búsquedas de dispositivo activos en el área además se puede consultar los dispositivos que se conectaron al menos una vez con el Smartphone, permitiendo realizar una conexión más rápida ya que sus datos se encuentran almacenados en el Smartphone. (Trujillo, 2012)

La búsqueda de dispositivos es un proceso de exploración que permite la obtención de nuevos apareamientos con diferentes emisores. Solo es posible detectar un nuevo dispositivo si este se encuentra en estado visible en el medio. (Trujillo, 2012)
private void doDiscovery() {
    if (D) Log.d(TAG, "doDiscovery()”);

    // Indicate scanning in the title
    setProgressBarIndeterminateVisibility(true);
    setTitle(R.string.scanning);

    // Turn on sub-title for new devices
    findViewById(R.id.title_new_devices).setVisibility(View.VISIBLE);

    // If we're already discovering, stop it
    if (mBTAdapter.isDiscovering()) {
        mBTAdapter.cancelDiscovery();
    }

    // Request discover from BluetoothAdapter
    mBTAdapter.startDiscovery();
}

Figura 21: Ejemplo de búsqueda de nuevos dispositivos (Trujillo, 2012)

/**
 * Obtener un conjunto de dispositivos vinculados en ese momento
 */
Set<BluetoothDevice> pairedDevices = mBTAdapter.getBondedDevices();

// Si hay dispositivos vinculados, agregue cada uno de los ArrayAdapter
if (pairedDevices.size() > 0) {
    findViewById(R.id.title_paired_devices).setVisibility(View.VISIBLE);
    for (BluetoothDevice device : pairedDevices) {
        mPairedDevicesArrayAdapter.add(device.getName() + "\n" + device.getAddress());
    }
} else {
    String noDevices = getResources().getText(R.string.none_paired).toString();
    mPairedDevicesArrayAdapter.add(noDevices);
}

Figura 22: Ejemplo de búsqueda de dispositivos pareados o vinculados con el Smartphone (Trujillo, 2012)
Figura 23: Vista real de la aplicación, pantalla de búsqueda de dispositivos pareados y nuevos. (Trujillo, 2012)

- **CONECTAR CON DISPOSITIVO**

Cuando se realiza una conexión común entre 2 dispositivos necesitamos tener claro que ambos dispositivos necesitan datos de entrada y salida, para poder llegar a ese punto se necesita crear un canal RFCOMM donde cada dispositivo puede obtener flujos datos. (Trujillo, 2012)

La forma más adecuada para poder obtener este tipo de conexión es preparar la aplicación para que sea un servidor, dejando un socket abierto que está atento a las conexiones entrantes. (Trujillo, 2012)
Figura 24: Ejemplo de conexión bluetooth con AGV (Trujillo, 2012)

- RASTREANDO DISPOSITIVO EN MAPA

Para poder rastrear los dispositivos AGV es necesario tener un mapa del área donde el dispositivo va interactuar. En este caso el área de estudio simulada es determinada por la siguiente imagen. (Trujillo, 2012)
El mapa permite interpretar los datos que envía el dispositivo AGV al Smartphone dibujando su localización dentro del área definida de estudio. Para poder dibujar el mapa en el dispositivo se utilizó la librería disponible por android llamada Canvas. Un ejemplo claro de cómo funciona es el siguiente. (Trujillo, 2012)

```
DisplayMetrics metrics = new DisplayMetrics();

HEIGHT = metrics.heightPixels;
WIDTH = metrics.widthPixels;
WIDTH = getWidth() -2;
HEIGHT = getHeight() -10;

Bitmap imagen = BitmapFactory.decodeResource(getResources(), R.drawable.planooblio);
imagen = Bitmap.createScaledBitmap(imagen, WIDTH, HEIGHT, true);
canvas.drawBitmap(imagen, 0, 0, null);
```

Se utiliza el DisplayMetrics para poder interpretar cuanto es el tamaño real de la pantalla del dispositivo que está ejecutando la aplicación, es necesario ya que sin esta propiedad la aplicación no sería acta para la ejecución en Smartphone con pantallas diferentes al tamaño estándar definido a la hora de la construcción de la aplicación. (Trujillo, 2012)

Para que el Canvas pueda pintar la imagen definida es necesario convertir la imagen en un mapa de bits utilizando la propiedad Bitmap, esta propiedad es la encargada de dar la resolución y escalado que la imagen necesita para que el Canvas ubique la imagen según la coordenada definida. (Trujillo, 2012)

Una vez el Smartphone y el AGV este sincronizados para transmitir datos es necesario definir en la aplicación un receptor de información, que se encargara de escuchar todo aquello que el AGV envié. Por eso es necesario declarar una clase encargada de la recepción de información de la siguiente manera. (Trujillo, 2012)

```
// El controlador que obtiene información de vuelta de la BluetoothChatService
private final Handler mHandler = new Handler()
{
    @Override
    public void handleMessage(Message msg) {
        switch (msg.what) {
        case MESSAGE_STATE_CHANGE:
            if (msg.arg1 == MESSAGE_STATE_CHANGE) {
             case BluetoothChatService.STATE_CONNECTED:
            // setStatus(getString(R.string.title_connected_to, connectedDeviceName));
            // conversationArrayAdapter.clear();
            break;
            case BluetoothChatService.STATE_CONNECTING:
            // setStatus(R.string.title_connecting);
            break;
            case BluetoothChatService.STATE_NONE:
            // setStatus(R.string.title_not_connected);
            break;
        }
        break;
        case MESSAGE_WRITE:
        byte[] writeBuf = (byte[]) msg.obj;
        // construir una cadena de la memoria intermedia
        String writeMessage = new String(writeBuf);
        //ConversationArrayAdapter.add("Ri: " + writeMessage);
        break;
        case MESSAGE_READ:
        byte[] readBuf = (byte[]) msg.obj;
        // construir un string desde los bytes en el buffer
        String readMessage = new String(readBuf, 0, msg.arg1);
```

Una vez ya establecido el receptor de los datos, se procede al envío y separación de la información enviada del AGV, para poder dibujar el punto de localización enviando por el dispositivo es necesario utilizar **SurfaceView**. (Trujillo, 2012)

El **SurfaceView** proporciona una superficie adicional a la ya utilizada para la impresión del mapa, ayuda a crear una jerarquía de capas permitiendo dibujar planos en el lugar correcto de la pantalla. (Trujillo, 2012)

Un ejemplo de programación de esa propiedad sería de la siguiente manera. (Trujillo, 2012)

```java
public class WaveformView extends SurfaceView implements SurfaceHolder.Callback{

    // plot area size
    int WIDTH;
    int HEIGHT;

    private static int x;
    private static int y;

    private WaveformPlotThread plot_thread;

    private Paint grid_paint = new Paint();
    private Paint cross_paint = new Paint();
    private Paint outline_paint = new Paint();

    public WaveformView(Context context, AttributeSet attrs)
    {
        super(context, attrs);
        getHolder().addCallback(this);
        // initial values
        for(int x=0; x<WIDTH; x++){
            // ch1_data[x] = CH1_pos;
            // ch2_data[x] = ch2_pos;
        }
        plot_thread = new WaveformPlotThread(getHolder(), this);
        grid_paint.setColor(Color.rgb(100, 100, 100));
        cross_paint.setColor(Color.rgb(70, 100, 70));
        outline_paint.setColor(Color.OLIVE); 
    }
```
12.3 Participación Proyecto AGV

La participación en el proyecto agv fue adicionar estas líneas de código en la parte de la comunicación con el dispositivo ya que cuando se ejecutaba no se limpiaban los campos de datos

```java
mWaveform.set_data(Integer.parseInt(divimessage[2]), Integer.parseInt(divimessage[3]));
divimessage[0]="";
divimessage[1]="";
divimessage[2]="";
divimessage[3]="";
```

El proyecto agv (Vehículo Autónomo Guiado) fue construido por un estudiante de la san buenaventura el cual se perfecciono y se logro llegar a resultados que se mostraran en un anexo 1 el cual tendrá el nombre de video del prototipo funcional avg.

13 DEFINICIÓN DE ACTIVIDADES EN CAPACITACIÓN

El currículo de actividades que se desarrollaran en la capacitación de los estudiantes del semillero será el siguiente donde los estudiantes se comprometerán al final del semillero a entregar productos que demuestren el conocimiento brindado.

Los temas son:

1. Pasos instalación Android SDK
   1.1 Descargar eclipse June (15 min)
   1.2 Descargar JDK (15 min)
   1.3 Descargar Android SDK (15 min)
   1.4 Descargar Kies dependiendo del smartphone (samsung kies, motorola kies, Nokia kies...) (15 min)

2. Instalación Eclipse, JDK, SDK (40 minutos)

3. Crear máquina virtual con el SDK (15 minutos)

4. Descargar API's del SDK (1 hora)

5. Configuración Eclipse y SDK (1/2 hora)
6. Verificación de la instalación (15 minutos)
todos los componentes funcionando y "subida" de la máquina virtual

7. Crear Hola Mundo:
7.1 Botón de comando, etiqueta (label), cajas de texto ejecución con comando y con vista
(1/2 hora)
7.2 ejecución del proyecto hola mundo en un smartphone. Copiando el .apk con el cable
de datos o también simulando el entorno mediante KIES (1 hora)

8. Proyecto de comunicación entre dispositivos móviles por medio de Bluetooth
8.1 Creación clases y vistas (1 hora)
8.2 Controlador androide manifiest (1 hora)
8.3 Prueba y Ejecución bidireccional (dispositivo móvil vs dispositivo móvil: ejemplo
 tableta y smartphone) (1 hora)

9. Proyecto de comunicación entre dispositivos móviles por medio de WIFI
9.1 Creación clases y vistas (1 hora)
9.2 Controlador androide manifiest (1 hora)
9.3 Prueba y Ejecución bidireccional (dispositivo móvil vs dispositivo móvil: ejemplo
tableta y smartphone) (1 hora)

10. Demostración de comunicación entre dispositivos móviles en un sistema AGV
(Vehículo Guiado Autónomo)
En este proyecto se integra: clases, vistas, imágenes, coordenadas (x,y),
posicionamiento, comunicación con bluetooth.
10.1 Creación clases y vistas (1 hora)
10.2 Controlador androide manifiest (1 hora)
10.3 Prueba y Ejecución bidireccional (dispositivo móvil vs AGV ejemplo smartphone vs
smartphone ) (1 hora)

14 COMPONENTES E INSTALACIÓN PARA PROGRAMAR PARA DISPOSITIVOS
MOVILES CON SISTEMA OPERATIVO ANDROID

Para la programación de dispositivos móviles con sistemas operativos basados en
androide se requiere de estos aplicativos para su correcto funcionamiento

La herramienta eclipse la cual es una plataforma de programación.

El JDK (Java Development Kit) el cual nos permite la integración de conexión entre
dispositivos.

EL androide SDK el cual es la api que nos permite la comunicación entre las plataformas
de programación con androide.
La instalación de estos componentes y su configuración se realiza de la siguiente manera

1) se descargar el eclipse de este link
http://www.eclipse.org/downloads/

2) se descarga el jdk de este link

3) se descarga el SDK de este link

4) Se configura una maquina virtual en el SDK como se muestra en las siguientes imágenes

Se abre el AVD manager el cual nos ayuda a crear la maquina virtual y se le da en new

![Figura 24: Creación de Maquina Virtual](image)

Y se crea la maquina virtual con sus respetivos campos
Se descargarán los kies que son necesarios para simular en un Smartphone las aplicaciones realizadas en Eclipse como se ve en la figura. Se abre el SDK manager y se descargan los tools para el funcionamiento de Android. Dentro de las plataformas de programación se recomienda descargar la versión 2.2 siempre por ser compatible con cualquier Smartphone.
Se configura el SDK con el eclipse para simular un dispositivo android dentro de eclipse con los siguientes pasos. Esto es para el path

En windows xp
- Botón derecho en MiPC -> Propiedades -> Pestaña Opciones Avanzadas -> Botón Variables de entorno
- En la lista de Variables del sistema, hacemos doble clic en la variable Path, y en Valor de la variable, agregamos al final de todos los valores que contiene, el siguiente texto:

  ; C:\Archivos de programa\Android\android-sdk\tools

En windows 7
- Botón derecho en MiPC -> Propiedades -> Pestaña Opciones Avanzadas -> Botón Variables de entorno
- En la lista de Variables del sistema, hacemos doble clic en la variable Path, y en Valor de la variable, agregamos al final de todos los valores que contiene, el siguiente texto:

  C:\Users\consejo\AppData\Local\Android\android-sdk\tools

14.1 INSTALACION ADT (ANDROID DEVELOPMENT TOOLS) EN ECLIPSE
- En nombre ponemos Android, por ejemplo. En Location colocamos la dirección: http://dl-ssl.google.com/android/eclipse/
- Pulsamos Ok.

14.1.1 CONFIGURACION DEL SDK EN ECLIPSE
En windows xp
- Ejecutamos Eclipse -> Windows -> Preferences -> Android.
- Pulsamos Browse y seleccionamos el directorio raíz de la instalación del SDK de Android (normalmente C:\Archivos de programa\Android\android-sdk).

En windows 7
- Ejecutamos Eclipse -> Windows -> Preferences -> Android.
- Pulsamos Browse y seleccionamos el directorio raíz de la instalación del SDK de Android (normalmente C:\Users\consejo\AppData\Local\Android\android-sdk)

NOTA: los kíes funcionan dependiendo del modelo de Smartphone que tengamos. Ejemplo en un Samsung se descargar el Samsung kíes de este link http://www.samsung.com/co/support/usefulsoftware/KIES/JSP
Estos son los pasos que se deben de seguir para la instalación del androide SDK y su correcta configuración en el anexo 2(instalacion del sdk y creacion de proyectos) está el video tutorial de la instalación completa

14.2 CAPACITACION SEMILLERO

En la capacitación del semillero se propusieron metas que se deben lograr por los participantes del semillero, como son estudiantes de distintos semestres y unos cuentan con más capacidad para desarrollar que otros, se propuso que los estudiantes de últimos semestres crearan una aplicación para el final del semillero y que los de primeros semestres crearan video tutoriales de todo lo que se construya durante el semillero estos avances se mostraran al final del semillero lo cual no quedaran justificados en este proyecto.

En el anexo 4 se presentan algunas evidencias del proceso de capacitación
Se requiere de más desarrolladores en programas para dispositivos móviles lo que justifica aumentar la motivación en la creación de semilleros, grupos de estudio y ampliar estas temáticas en las asignaturas y en los programas de ingeniería de sistemas, ingeniería informática, ingeniería del software y afines.

La programación para dispositivos móviles especialmente para dispositivos con sistema operativo android es de gran actualidad y de gran aplicación en la industria de la tecnología de la información.

Se ha podido verificar la coexistencia de varios protocolos y tecnologías de comunicación en el desarrollo de aplicativos para dispositivos móviles, así como la integración con plataforma de hardware desarrolladas desde la ingeniería electrónica. Esto último confirma una alta interdisciplinariedad para lograr estos productos.

Según lo analizado, consultado y lo ya realizado con algunos sistemas como lo son el avg y RFID se han creado sistemas de comunicación entre dispositivos para la diferentes problemáticas que se han presentado en el mercado actual.

Se realiza las capacitaciones a los estudiantes del semillero logrando dar a conocer la herramienta SDK para programar dispositivos móviles y demostrando los diferentes tipos de comunicación existentes consiguiendo así los diferentes tipos de comunicación que hay en la actualidad y logrando dar a conocer lo que se puede obtener en cuanto a conexión entre dispositivos y todo lo que se puede construir para el mercado actual el cual lo exige así.
16 REFERENCIAS BIBLIOGRÁFICAS


Bertolín, J. A. (s.f.). Conectrónica. Recuperado el 15 de Noviembre de 2012, de Identificación, valoración y análisis de la tecnología NFC: http://www.conectronica.com/Redes-Wireless/Identificacion%23Bn-valoracion%23Bn-y-an%23A1lis-de-la-tecnolog%23ADa-NFC.html


Trujillo, J. D. (18 de 10 de 2012). *proyecto de grado agv*. Recuperado el 22 de 09 de 2013, de proyecto de grado agv.