Infraestructuras de datos espaciales en dispositivos móviles inteligentes

Maestría en Geomática con mención en Ordenamiento Territorial

Autor: Diego Francisco Pacheco Prado

Director: PhD Daniela Ballari

Cuenca, Ecuador

2013

DEDICATORIA

Este trabajo va dedicado a la persona más importante de mi vida, mi hija Sofía, que con su alegría y ternura han sido la fuente primordial de fuerza y perseverancia.

A mis padres Juan y Yolanda, mis hermanos Juan y Priscila, mi cuñada Irma que siempre me apoyaron y estuvieron conmigo en los momentos más difíciles. Gracias a todos ellos puedo culminar esta etapa.

AGRADECIMIENTOS

Agradezco a mi directora y amiga Daniela Ballari, todo el apoyo y asesoría brindada fueron de vital importancia. Sus consejos permitieron desarrollar este trabajo además que en muchas ocasiones sus palabras de ánimo me dieron fuerzas para seguir adelante.

Al Msc. Omar Delgado, maestro y amigo, que siempre tuvo paciencia y compresión pero ante todo nos impulso al desarrollo como investigador y profesional del SIG.

A María José Vintimilla por muchos sacrificios y esfuerzos realizados para permitirme cumplir esta meta de vida.

A mis profesores, compañeros de maestría, amigos, gente del IERSE, que transmitieron su fuerza y ánimo, importantes en los momentos de flaqueza.

Al Ing. Chester Sellers, amigo de años quien me apoyo y ayudó durante esta etapa educativa. Con sus conocimientos y amistad me permitieron crecer como ser humano y profesional

RESUMEN

Las tecnologías móviles y el desarrollo de la web 2.0 han permitido que los usuarios puedan crear y acceder a gran cantidad de conocimientos (entre ellos información geográfica). El objetivo de esta tesis es crear un aplicativo qué, a través de dispositivos móviles, permita acceder a los datos de una infraestructura de datos espaciales (IDE) de una forma rápida y oportuna. El aplicativo contempla 3 aspectos principales: 1) un visor móvil de cartografía y búsqueda de metadatos; 2) sincronización móvil-PC a través de marcadores; y 3) códigos de respuesta rápida (QR) para generar índices de acceso a metadatos y mapas.

Palabras clave.

Información geográfica, infraestructura de datos espaciales, información geográfica, Android, App Inventor, Escáner QR, marcadores

ABSTRACT

The development of mobile phones and web 2.0 has allowed the users to create and have access to a great amount of knowledge (geographic information among others). The goal of this thesis is to create an application through a mobile device that allows quick and opportune access to data from Spatial Reference Identifiers (SRID). The application contemplates three main aspects: 1) a cartographic visualization and metadata searcher for mobile applications; 2) synchronizing mobile-PC through markers; and 3) Quick Response codes (QR) in order to generate guided access to the metadata and the maps.

UNIVERSIDAD DEL AZUAY
DPTO. IDIOMAS

Translated by,
Diana Lee Rodas

keywords

Geographic information, spatial data infrastructure, Android, App Inventor, QR scanner, markers.

Índice de contenidos

Índice de figuras, tablas y anexos

Diego Francisco Pacheco Prado

Trabajo de graduación

Daniela Ballari

Marzo, 2013

Infraestructuras de datos espaciales en dispositivos móviles inteligentes

INTRODUCCIÓN

El auge de la información geográfica en la web ha consolidado lo que actualmente conocemos como infraestructuras de datos espaciales (IDEs). Las IDEs facilitan el acceso a la información geográfica proveniente de diferentes fuentes, a través del establecimiento de normativa y del desarrollo de geoservicios web estandarizados. Los principales geoservicios de una IDE son los catálogos de metadatos, la visualización de cartografía en la web y el acceso a los datos mismos para su posterior análisis espacial.

El Open Geospatial Consortium (OGC) ha trabajo en la generación de estándares y especificaciones para el intercambio de información geográfica a través de internet. Varias especificaciones de este consorcio, como Web Map Service (WMS), Web Feature Service (WFS) y Web Coverage Service (WCS), han culminado en implementaciones de software como Mapserver y Geoserver que en la actualidad constituyen dos de las plataformas más utilizadas para la construcción y publicación de geoservicios.

En nuestro país varias de estas especificaciones y normas han sido adoptadas como políticas nacionales de geoinformación de registro oficial N° 269 del 1 de septiembre del 2010 (Conage, 2010). Estas políticas son promovidas para que todas las entidades utilicen los mismos criterios con la finalidad de articular los nodos de información geográfica en un único Sistema Nacional de Información (SNI) y a la Infraestructura Ecuatoriana de Datos Geoespaciales. Como consecuencia, los gobiernos locales y regionales dentro del país han notado la necesidad de publicar su información geográfica y han invertido recursos y esfuerzos en el desarrollo e implementación de IDEs institucionales. Un ejemplo es el nodo de la Universidad del Azuay (Pacheco, 2012) que se encuentra implementado en su totalidad bajo software libre utilizando como sistema operativo Linux Centos, el cual tiene una robustez fuerte en servidores; Geoserver y Mapserver para la publicación de servicios OGC y una base de datos espacial

Postgres/Postgis que ha demostrado tener un soporte óptimo para el gran volumen de información almacenado y una amplia funcionalidad para el manejo de información espacial.

La tecnología celular y dispositivos móviles han generado una masiva invasión de estos equipos en la sociedad actual. Se ha potenciado su uso no sólo como teléfono sino también como dispositivo de acceso a internet, navegador gps, y sensores que interactúa con su entorno (medición de ruido, velocidad, etc.). Uno de los sistemas operativos más utilizados en estos dispositivos móviles inteligentes es Android el cual ha demostrado tiene varias ventajas en comparación a otros existentes en el mercado. Entre las ventajas principales podemos mencionar que este sistema se basa en software libre, es un sistema abierto para funcionar en dispositivos de diferentes fabricantes y presenta versatilidad en el manejo de memoria interna. En el Ecuador, la disponibilidad de estos dispositivos móviles está en aumento. Según datos de la encuesta nacional de empleo, subempleo y desempleo (ENEMDU) realizada por el Instituto Nacional de estadísticas y Censos (INEC) en el mes de diciembre del 2011, el 8.4% de los teléfonos activados son inteligentes con tendencia a crecer en los próximos años (INEC, 2011). En cuanto a su arquitectura, el sistema Android se puede subdividir en cinco capas: el kernel y las herramientas de bajo nivel, librerías nativas, runtime de Android, el framework de desarrollo y las aplicaciones (Brähler, S. 2010). Aplicativos como el “App Inventor” permite la programación de aplicaciones en Android usando un lenguaje de programación gráfico de bloques (Wolber, 2011).

En el ámbito de las IDEs, estas nuevas tecnologías promueven importantes cambios. Los dispositivos móviles presentan ventajas en relación al acceso ubicuo, la funcionalidad de pantalla táctil (touch-screen) para la navegación de mapas, así como la disponibilidad de sensores (cámara, gps, sonido, velocidad y movimiento) que permiten interactuar con el entorno. También son ideales para promover la capacidad de movilidad de los usuarios y para reportar o captar eventos geográficos en tiempo real (Goodchild, 2007). Actualmente cada vez son más comunes los aportes que realizan los usuarios sobre cualquier problemática que involucre un componente geográfico (tráfico vehicular, ruido, etc.) a través de sus móviles. Esta forma de aportación por parte de los usuarios se denomina información geográfica voluntaria (Castelein, Grus, Crompvoest, & Bregt, 2010). Los accesos y reportes se realizan desde dispositivos móviles inteligentes a través de un portal web adaptado a las funcionalidades de los mismos. Es decir, usuarios no expertos utilizan los conceptos fundamentales de un sistema de información geográfico sin darse cuenta de ello.

A pesar de estas ventajas, el acceso a los principales geoservicios IDE desde un entorno móvil no es un problema de solución directa, dadas las limitaciones de tamaño de pantalla, tamaño de memoria, consumo de energía, ancho de banda y capacidad de procesamiento. Como

consecuencia, las IDEs implementadas para ser accesibles a través de una PC, dejan de ser el medio óptimo cuando el acceso a la información geográfica se realiza desde dispositivos móviles.

Para la visualización de información geográfica en dispositivos móviles se han desarrollado varios aplicativos. Estos emulan algunas de las principales funciones de un SIG de escritorio. Ejemplos son Arcgis (ESRI, n.d.-a), GvSig mini ("gvSIGmini," n.d.-a), OruxMaps ("Oruxmaps," n.d.-a), Locus map ("Locus Map," n.d.-a) y Qgis ("Qgis-android," n.d.). Además, algunos clientes web están siendo adaptados para dispositivos móviles a través de interfaces más ligeras y funcionalidades touch-screen, como es el caso de GeoExt mobile (GXM) (GeoExt, n.d.) y OpenLayers mobile ("OpenLayers mobile," n.d.). Sin embargo, para la búsqueda y consulta de metadatos de la información geográfica a través de dispositivos móviles, no se han detectado desarrollos de aplicativos. Adicionalmente, no se tiene conocimiento sobre el uso de códigos QR (Quick Response code, "código de respuesta rápida") para codificar los criterios de búsquedas de metadatos o mapas y realizar consultas sin necesidad de escribir texto sino únicamente usando un medio digital para leer el código. Como ejemplo de utilización de QR puede mencionarse si usamos esta etiqueta en algún monumento o edificio público, el criterio de búsqueda codificado nos llevará a todos los metadatos y mapas donde se haya usado información de ese objeto geográfico lo que reducirá los errores de sintaxis, modismos y tipeo.

Con dichos antecedentes se plantea como objetivo de esta tesis la creación de una infraestructura de datos espaciales para dispositivos móviles con sistema Android con la finalidad que un usuario pueda usar su dispositivo móvil para explotar y extender las potencialidades de acceso a la información contenida en una IDE. Esta infraestructura de datos espaciales móvil se ha implementado utilizando como caso de uso la IDE de la Universidad del Azuay.

1. MARCO TEÓRICO

Este apartado describe conceptualmente una IDE y los dispositivos móviles inteligentes. Estos conceptos son necesarios para comprender la integración que sufrirán estas dos tecnologías y que convergerán en una IDE móvil.

1.1. Infraestructuras de datos espaciales

*Observación: Sección basada en el Curso e-learning de metadatos. Programa Geosur.

Una infraestructura de datos espaciales (IDE) es un conjunto de normas, políticas y estándares cuya finalidad es la publicación y representación en internet de datos, metadatos y servicios de

forma estándar, garantizando la interoperabilidad de estos y permitiendo el acceso a dicha información de una forma rápida y oportuna (G & Gil, 2013).

El éxito de las IDE es la combinación de los sistemas informáticos con las políticas y normas que permitirán publicar información de una forma estándar que garantice la interoperabilidad de la misma en otros sistemas.

Una IDE se fundamenta en 7 componentes principales: datos, metadatos, servicios, normas, acuerdos y políticas, tecnología y actores. Cada uno de ellos será descrito a continuación, con su respectiva vinculación a las políticas nacionales de geoinformación (Conage, 2010).

a) Datos

Se denomina datos a la información geográfica que será accesible desde internet. Se pueden clasificar en dos grandes categorías:

- Datos base: información básica de propósito general que servirá como punto de partida en la construcción de información temática. Ejemplo de este tipo de información es: límites políticos, hidrografía, vialidad, etc.
- Datos temáticos: describen un aspecto específico del territorio. Dentro de esta clasificación se pueden mencionar información correspondiente a campos de: clima, vegetación, suelos, etc.

Estos datos, antes de ser publicados, deberán estar sujetos a procesos de validación y revisión para garantizar la calidad de los mismos. Esta información puede ser provista por cualquier entidad, ya sea pública o privada. La política nacional de geoinformación del Ecuador dicta que "la calidad de la información geoespacial debe cumplir con normas y estándares nacionales, y documentarse a través de los metadatos geográficos" para exigir que los productores brinden productos de calidad. Además, los mismos tendrán que actualizarse bajo cierta periodicidad con la finalidad que los datos se ajusten lo más posible a la realidad del territorio (Conage, 2010).

Una IDE, desde el punto de vista tecnológico y de similar manera que un sistema de información geográfico, maneja dos modelos para representar los datos geográficos. El modelo vectorial donde los datos se registran por medio de las fronteras de la entidades geográficas y el modelo ráster donde no se registra las fronteras sino su contenido, usando para ello una malla regular de celdas (pixel) donde se registra el valor del atributo asignado (Delgado & Ochoa, 2011).

b) Metadatos

Un metadato se define como “el dato de un dato”. Es decir, consiste en toda la información necesaria para describir un dato generado. En los sistemas de información geográficos los metadatos constituyen el punto de partida para determinar el sistema de referencia, escala y fuentes de la información geográfica. Estos datos permitirán brindar un componente de calidad a la información.

Dentro de las IDE se han planteado normas internacionales (ISO) para estructurar la información a presentar dentro del metadato. La norma base para la publicación de metadatos de información geográfica es la norma ISO 19115:2003 (Geographic Information -- Metadata) que tiene el propósito de documentar la información geográfica como tal y para la publicación de metadatos de servicios es la norma ISO 19119:2005 (Geographic Information -- Services) para describir la información de los geoservicios. Esto se encuentra respaldado por la política nacional de geoinformación que dicta “Todas las instituciones productoras y/o custodias de información geoespacial deben generar los metadatos de acuerdo a la normativa vigente, precautelando la propiedad intelectual del titular” (Conage, 2010).

Las IDE gestionan y manejan sus metadatos a través de catálogos, para facilitar el acceso a la información. Productos populares en esta categoría son: ArcCatalog (ESRI, n.d.-a) y Geonetwork (“Geonetwork,” n.d.), siendo el tercero el que mayor acogida ha tenido en el país al ser mayoritariamente adoptado por instituciones públicas y privadas para la gestión de sus metadatos.

En el Ecuador, existe el perfil ecuatoriano de metadatos (PEM) (Conage, 2010) cuya finalidad es nacionalizar y adaptar una norma internacional (ISO 19115) de acuerdo a las necesidades de documentación de metadatos del país (Conage, 2010). Esta norma ha sido adoptada en el catálogo de metadatos de la IDE de la Universidad del Azuay como base fundamental en la construcción y edición de metadatos.

c) Servicios

Los servicios se pueden definir como un “c*onjunto de operaciones aplicadas sobre datos geográficos que se ofrecen a través de la Web, para ser utilizadas por usuarios o aplicaciones informáticas. Existen servicios para la visualización, descarga, localización, etc. de información geográfica*” (G & Gil, 2013).

Entre los servicios más usados encontramos: Web Map Service (WMS) para visualización de cartografía, Web Feature Service (WFS) para acceso a la información vectorial, Web Coverage

Service (WCS) para acceso a coberturas e información ráster y Web Map Context (WMC) para almacenamiento de vistas de mapas y proyectos en formato XML.

La política nacional de geoinformación dicta los tipos de servicios a ser implementados al mencionar que "Toda institución u organización propietaria y/o custodia de información geoespacial debe contar con una infraestructura de datos geoespaciales (IDE) que garantice el acceso a los servicios de la información que le compete, enlazada a la Infraestructura Ecuatoriana de Datos Geoespaciales (IEDG), facilitando el acceso, búsqueda, visualización y descarga de la información" (Conage, 2010). La IDE de la Universidad del Azuay ha implementado sus servicios bajo las recomendaciones (servicios, hardware y software) realizadas por el IEDG permitiendo la publicación de servicios bajo su visor de mapas.

d) Normas y estándares

Para logar la interoperabilidad y manejo de estos sistemas, es indispensable contar con normas y estándares. Estas son generalmente dictadas por el Open Geospatial Consortium cuando se trata de estándares para servicios y por el *International Organization for Standardization* (ISO) cuando se trata de normas de datos y metadatos. A su vez, ISO adopta varias de las normativas de servicios dictadas por el OGC.

La aplicación de estas normas y estándares a datos y metadatos, permitirá que su acceso y entendimiento sea interpretado adecuadamente por sistemas que implementan los mismos estándares.

Dentro de las políticas nacionales de geoinformación del Ecuador, se menciona que las instituciones productoras de información geoespacial deben garantizar la interoperabilidad de los servicios, además de exigir que las políticas, normas y estándares institucionales deban estar alineados a los nacionales (Conage, 2010).

e) Acuerdos políticos

Estos acuerdos se realizan entre productores de datos y la entidad o institución que los publica o utiliza. Estos acuerdos se dan para alinear los esfuerzos de las instituciones que participan en los proyectos de IDE a un contexto nacional. En el Ecuador esos acuerdos políticos deben alinearse a las normas o políticas nacionales con la finalidad que la información generada por cada institución se integre a la Infraestructura Ecuatoriana de Datos Geoespaciales.

f) Tecnología

El componente tecnológico se encarga de todos los recursos de hardware y software necesarios para que la IDE se comunique a nivel global a través del canal de internet. Por lo general, en el contexto de IDE ha tenido muy buena aceptación el software libre.

Las IDE se comportan de forma semejante a una página web por lo que el modelo de tres capas cliente/servidor se ajusta perfectamente a esta arquitectura. En este modelo la primera capa se refiere al cliente que solicita la información, la segunda capa es la capa de negocios o servidor de aplicaciones la cual accede a la tercera capa denominada de datos (base de datos) para atender la solicitud del cliente. Esto permite que el procesamiento de datos recaiga sobre el servidor y que el cliente solo se concentre en la presentación de los datos.

Para consumir la información o servicios provistos por los servidores de mapas se usa dos tipos de clientes. El primer tipo de cliente son los clientes ligeros que son aplicaciones realizadas en la web para visualizar la información y trabajar con protocolos básicos de las IDE como WMS, WFS, CSW. El segundo tipo de cliente son los clientes pesados que se asocian directamente a los sistemas de información geográfica que actualmente incorporan funcionalidades para la conexión con servidores de mapas y protocolos estándar del OGC.

En cuanto a servidores, a nivel nacional, varias instituciones han adoptado Geoserver como servidor de servicios OGC y Geonetwork como catálogo de metadatos. Algunas instancias públicas todavía mantienen a Mapserver como servidor de mapas y servicios pero por la interoperabilidad de estos no existe dificultades en conectarlos al visor o cliente donde se visualice la información.

g) Actores

Son todas las personas físicas o jurídicas que intervienen en el proceso de publicación de la información, desde los productores de información, quienes la publican hasta los que consumen o utilizan la misma.

Dentro del grupo de actores existe la ciudadanía en general que no tiene conocimientos profundos sobre sistemas de información geográfica o IDE. Para ellos se crea un portal web para acceder a la información de la IDE, a estos portales se los conoce también como Geoportales.

1.2. Programación web aplicada a infraestructura de datos espaciales

Las IDE y geoportales se acceden y usan a través de internet. Debido a ello se utilizan los lenguajes de programación web para acceder y visualizar la información. A nivel de programación web e infraestructura de datos espaciales existe la posibilidad de programar aplicativos en diferentes lenguajes de programación como Java, Ruby, Phyton, entre otros. Para

el caso de la IDE de la Universidad del Azuay se utilizaron los lenguajes clásicos de esta programación como Php y Javascript.

Un cliente ligero es una interfaz web con la funcionalidad básica para acceder y navegar a través de la información geográfica. Se pretende que estas aplicaciones sean de tamaños no excesivos para que se puedan usar en internet sin contratiempos. Existen clientes ligeros pre-programados para montar visores y personalizar la funcionalidad deseada. Dependiendo del cliente escogido se usará lenguajes como Php (*Hypertext pre-processor*) usado en la capa del servidor y permitirá acceder a la información de la base de datos. Una extensión de este lenguaje es el Php/Mapscript el cual añade un conjunto de librerías para trabajar con servidores de mapas y poder construir interfaces de mapas en la web. Otro lenguaje de programación es JavaScript el cual se usa en la capa del cliente para procesar comandos en el navegador web. De este lenguaje se han creado librerías para trabajar con mapas, la más popular de ellas es OpenLayers de código abierto.

Existen dos alternativas principales en software libre para el uso de mapas en móviles. La primera opción es OpenLayers que es una librería JQuery (biblioteca Javascript que facilita la forma de interactuar con documentos HTML) basada en Javascript (lenguaje de programación interpretado). OpenLayers también implementa funciones para realizar consultas asíncronas al servidor a través de funciones y archivos programados con Php. Esta librería permite crear páginas web dinámicas e implementar funciones para interactuar con los mapas. En su versión 2.11 añadió una funcionalidad de soporte de mapas para móviles. La segunda opción es GXM que es la abreviatura de *"Mobile GeoExt"*. Esta, a su vez, se basa en OpenLayers para añadir la funcionalidad de soporte de mapas propia de móviles inteligentes. El resultado es un visor de mapas adaptado para dispositivos móviles. GXM está basado en *Sencha Touch* (framework de desarrollo de web para móviles alto rendimiento en HTML5), gracias a ello este visor es compatible con los navegadores web nativos de Ipad, Iphone, Android y en ambientes de escritorio con navegadores como: Apple Safari y Google Chrome (Terrestris GmbH & Co. KG, n.d.).

1.3. Dispositivos móviles

En los últimos años hemos observado la evolución de dispositivos celulares y como han cambiado de un simple teléfono a un dispositivo con múltiples funcionalidades emulando las características básicas de un computador. Adicionalmente hemos presenciado el nacimiento de dispositivos como Tablet y MID (Mobile Internet Device) cuya principal funcionalidad es interactuar con internet para aprovechar las aplicaciones diseñadas en estos dispositivos. En el mercado actual, a estos móviles con funcionalidad extra se les denomina móviles inteligentes pues su capacidad de procesamiento y conexión mejora con cada versión. Los principales

fabricantes de móviles han invertido esfuerzos en la creación de sistemas operativos que exploten al máximo los recursos de hardware, a continuación (Figura 1) se presenta una comparativa de estos sistemas analizando las características fundamentales de cada uno (G & Gil, 2013).

	Apple iOS 6	Android 4.2	Windows Phone 7	BlackBerry OS 7	Symbian 9.5
Compañía	Apple	Open Handset Alliance	Windows	RIM	Symbian Foundation
Núcleo del SO	Mac OS X	Linux	Windows CE	Mobile OS	Mobile OS
Familia CPU soportada	ARM	ARM, MIPS, Power, x86	ARM	ARM	ARM
Lenguaje de programación	Objective-C, C++	Java, C++	C#, muchos	Java	C++
Licencia de *software*	propietaria	*software* libre y abierto	Propietaria	propietaria	*software* libre
Año de lanzamiento	2007	2008	2010	2003	1997
Motor del navegador web	WebKit	WebKit	Pocket Internet Explorer	WebKit	WebKit
Soporte Flash	No	Sí	No	Si	Sí
HTML5	Sí	Sí	Sí	Sí	No
Tienda de aplicaciones	App Store	Google Play	Windows Marketplace	BlackBerry App World	Ovi Store
Número de aplicaciones	400.000	300.000	50.000	30.000	50.000
Coste publicar	$99 / año	$25 una vez	$99 / año	sin coste	$1 una vez
Plataforma de desarrollo	Mac	Windows, Mac, Linux	Windows	Windows, Mac	Windows, Mac, Linux
Actualizaciones automáticas del S.O.	Sí	depende del fabricante	depende del fabricante	Sí	Sí
Soporte memoria externa	No	Sí	No	Sí	Sí
Fabricante único	Sí	No	No	Sí	No
Variedad de dispositivos	modelo único	muy alta	baja	baja	muy alta
Tipo de pantalla	capacitativa	capacitiva /resistiva	capacitativa	/resistiva capacitativa	capacitiva /resistiva
Aplicaciones nativas	Sí	Sí	No	No	Sí

Figura 1. Comparativa de las principales plataformas móviles
(Fuente: Plataforma Miriada X - Curso programación Android)

Se puede observar que uno de los sistemas que mejores características presenta es Android que está basado en software libre, posee un alto número de aplicaciones disponibles y por su configuración y soporte de lenguaje de programación Java se puede adaptar aplicativos de diferente índole. Del lado del desarrollador de aplicaciones se observa que el costo de publicación de una aplicación es bajo en comparación a otros sistemas (25$ cada vez para Android, 99$/año para Apple y Windows). Además se puede desarrollar desde múltiples plataformas (Windows, Mac, Linux). Es decir la versatilidad y robustez de este sistema lo han

convertido en uno de los preferidos tanto por parte de los usuarios como por desarrolladores. Esto se ve reflejado en la siguiente comparativa (Figura 2) donde cada año se refleja el aumento en la venta de dispositivos con soporte de sistema Android. El estudio fue realizado por la empresa Gratner Croup donde se muestra la evolución del mercado de los sistemas operativos para móviles según el número de terminales vendidos a nivel mundial (G & Gil, 2013).

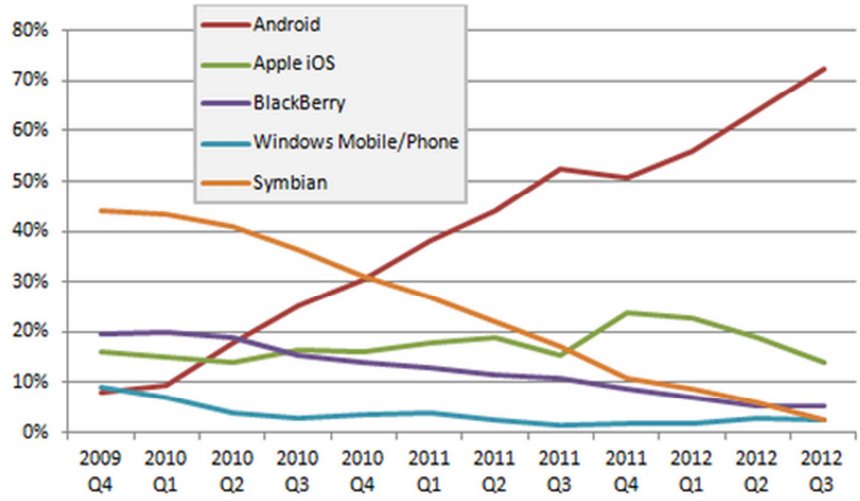

Figura 2. Porcentaje de teléfonos inteligentes vendidos según su sistema operativo hasta el tercer cuarto del 2012 en el mundo
(Fuente: Plataforma Miriada X - Curso programación Android)

Los próximos apartados se centran en describir y profundizar el sistema Android, el cual fue elegido como objetivo de desarrollo en base a las bondades descritas con anterioridad.

1.3.1. Arquitectura de Android

Android basa su funcionamiento en una arquitectura a cinco capas. Cada capa, a su vez, basa su funcionamiento en software libre (G & Gil, 2013).

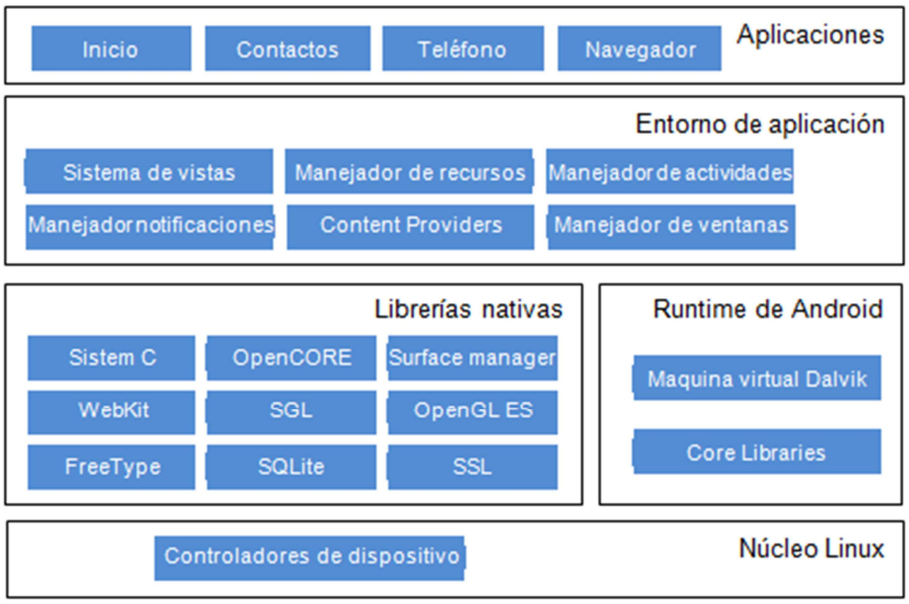

Figura 3. Arquitectura de Android
(Fuente: Plataforma Miriada X - Curso programación aplicaciones Android**)**

La capa de núcleo está directamente relacionada al sistema operativo Linux y abarca funcionalidades como: seguridad, manejo de memoria, multiproceso, y controladores para dispositivos entre otras.

La capa de runtime se basa en una máquina virtual (concepto usado para Java), cabe recalcar que Android no usa la misma máquina virtual de Java sino que Google creo una máquina virtual denominada Dalvik para estos dispositivos por las limitaciones en memoria y procesado en comparación a un computador.

El conjunto de librerías nativas descrito en la figura 3 son ejemplos de librerías usadas en varios componentes de Android y varias de ellas son proyectos de código abierto.

La capa entorno de aplicación proporciona un ambiente de desarrollo libre para aplicaciones e interconexión con los sensores que se incluyen en estos móviles inteligentes (gps, micrófono, etc.). En esta capa interviene el “App Inventor” (interfaz de desarrollo) el cual realiza programación visual a través de bloques, el mismo se analizará con más detalle en un apartado posterior.

Para finalizar, la capa de aplicaciones permite acceder a la información y funcionalidad básica del móvil. Por lo general estas aplicaciones son escritas en Java pero también soporta aquellas escritas en C/C++.

1.3.2. Aplicaciones SIG en móviles Android

Los aplicativos SIG de escritorio están realizando importantes esfuerzos para la generación y adaptación de sus aplicativos para dispositivos móviles. Estas aplicaciones principalmente se encargan de interactuar con el GPS del móvil y visualizar la información de medios locales o remotos en conjunto con la captura de datos del sensor GPS del móvil. Entre estos podemos mencionar a ArcGis (ESRI, n.d.-b), GvSig mini ("gvSIGmini," n.d.-b), OruxMaps ("Oruxmaps," n.d.-b), Locus free ("Locus Map," n.d.-b) y Quantum Gis ("Quantum Gis for Android," n.d.).

Para que estos aplicativos se conecten a los geoservicios de cualquier geoportal deben poseer la opción de conectarse a través de protocolos estándares como el: WMS, WFS, entre otros.

1.3.3. Infraestructuras de datos espaciales en dispositivos móviles Android

El concepto de IDE a nivel de móvil ha sido poco o nada estudiado, he ahí que solamente se han encontrado aplicaciones que visualizan la información geográfica o acceden a geoservicios (principalmente WMS) dejando a un lado uno de los componentes fundamentales de una IDE convencional como son los metadatos.

Al ser cada vez mayor el acceso a recursos de internet desde dispositivos móviles, los desarrolladores de lenguajes de programación web se han visto en la necesidad de proveer herramientas que faciliten la creación de sitios web móviles. Una de ellas son las librerías JQuery basada en Javascript. OpenLayers también implementa funciones para trabajar con JQuery y realizar consultas asíncronas al servidor. Esta librería permite crear páginas web dinámicas e implementan funciones para interactuar con información geográfica y geoservicios.

2. MÉTODOS

Este apartado describe las herramientas y componentes usados para el desarrollo de la IDE UDA móvil para dispositivos Android. Para ello se ha clasificado en cinco aspectos clave. El primero, un análisis de los requisitos del sistema a gran escala. El segundo, la arquitectura de la aplicación vista desde un modelo de tres capas. El tercero, los componentes del lado del servidor que permiten la consulta de información. El cuarto, una descripción de la base de datos que recopila datos de diferentes fuentes de la IDE de la Universidad del Azuay y, para finalizar, el quinto, componentes del lado del cliente.

Debe notarse que en el resto del documento se llamará IDE UDA móvil al aplicativo creado en esta tesis e IDE de la Universidad del Azuay a la IDE tradicional o convencional.

2.1. Requisitos

Dentro del diseño y programación de software se realiza en primera instancia un análisis de los requisitos que debe cumplir la aplicación. Para la propuesta de la IDE de la Universidad del Azuay para dispositivos móviles se plantean los siguientes requisitos en un contexto general:

Contenido: La IDE UDA móvil deberá presentar de la forma más adecuada posible los contenidos de la IDE de la Universidad del Azuay.

Información espacial: El interfaz deberá permitir interactuar con los servicios OGC provisto por la IDE de la Universidad del Azuay además de presentar las herramientas básicas para navegar en la información.

Sensibilidad al tacto en pantalla: En general la aplicación deberá soportar la funcionalidad *touch screen*, es decir, captura los eventos al tacto con la pantalla para la navegación de datos y geoinformación.

Reducción de volumen de información a trasferir: Al ser limitado el ancho de banda soportado por un dispositivo móvil la transferencia de información debe ser lo menor posible.

Procesamiento en servidor: El móvil deberá procesar la información lo mínimo posible, delegando esta tarea al servidor.

2.2. Arquitectura de la aplicación

La IDE UDA móvil basa su funcionamiento en el modelo general de 3 capas del esquema cliente/servidor (Figura 4). Este modelo consiste en un cliente que, pudiendo ser un navegador web móvil o una aplicación Android, accede al servidor de aplicaciones a través de internet. El servidor procesa las rutinas y consulta la información de la base de datos, la formatea y retorna al cliente para su visualización. Tanto el cliente como el servidor presentan un diseño modular basado en componentes. El reto principal consiste en adaptar los componentes fundamentales de una IDE convencional al entorno móvil siguiendo los requisitos establecidos.

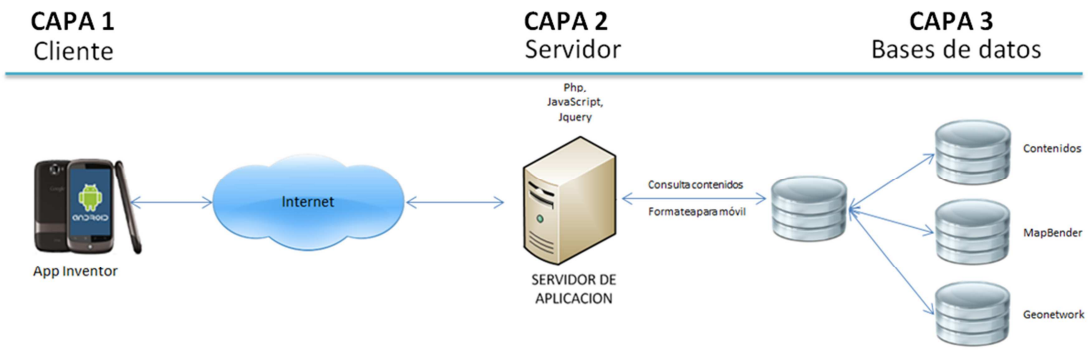

Figura 4. Esquema cliente/servidor aplicación IDE UDA móvil

La aplicación para Android fue desarrollada usando la aplicación App Inventor. La misma es una iniciativa del MIT de EEUU y cuya finalidad es facilitar la programación de aplicaciones para este sistema para usuarios que no poseen conocimientos profundos de los lenguajes de programación de esta plataforma. En esta plataforma cada una de las funciones de programación como estructuras de decisión, estructuras de repetición, objetos propios de este sistema se representan como piezas de un rompecabezas. Cada una de ellas presenta una o varias entradas y salidas dependiendo de su funcionamiento.

En el anexo 1 de este documento se documenta un fragmento de este rompecabezas. Se puede identificar estructuras de decisión como if – else, asignación de variables y la estructura de repetición when de una forma gráfica, cada una de ellas con sus entradas y salidas respectivas.

2.3. Componentes del servidor

La capa del servidor estará compuesta por el software y programas usados para la implementación de una IDE tradicional, es decir servicios de mapas WMS y catálogo de metadatos. Los mismos servidores web y sus lenguajes de programación servirán para obtener y formatear la información básica. Los componentes de la IDE de la Universidad del Azuay principalmente son: Apache y Apache Tomcat como servidor web y servidor de aplicaciones respectivamente, Geoserver como servidor de mapas, CMS Made simple como gestor de contenidos, Geonetwork como catálogo de metadatos, Postgres/Postgis como base de datos donde se almacena la información (geográfica y alfanumérica) y para finalizar Mapbender como cliente ligero para la visualización de cartografía en la web (Pacheco, Ballari, & Delgado, 2012).

.

2.4. Componentes de la base de datos

La base de datos relacional usada en este proyecto es Postgres con su extensión espacial Postgis (extensión que agrega funcionalidad para análisis espacial a la base de datos). Este apartado se centra en describir los componentes de las bases de datos especializados en ambientes de producción.

Como primera tarea se realizó un análisis de los aplicativos y de las bases de datos disponibles en la IDE de la Universidad del Azuay con la finalidad de identificar la información que se podría re-utilizar en la IDE UDA móvil. Se analizaron las bases de datos del gestor de contenidos, del visor de mapas Mapbender y del catálogo de metadatos Geonetwork. Como resultado, se detectaron las tablas a reutilizar en la IDE UDA móvil y se crearon otras tablas propias para la IDE UDA móvil.

Como se puede notar, al re-utilizar tablas de bases de datos existentes, estas se encontrarían dispersa en varios catálogos de datos. Para simplificar el acceso a esta información remota se utilizó el concepto de Vistas (Views) cuyo objetivo es usar la información de una consulta SQL solo para operaciones de lectura sobre los datos. Se puede observar que la información dispersa en varios catálogos ahora se accede desde un único esquema. Adicionalmente, se utilizan las vistas para obtener únicamente la información extremadamente necesaria (resumen) de cada uno de estos componentes.

El resultado es la base de datos de control de la IDE UDA móvil denominada “mide” y su función es concentrar y acceder a los recursos que se encuentran desplegados a través de diferentes bases de datos que cada componente de la IDE de la Universidad del Azuay. A continuación, se detalla la base de datos “mide” compuesta por las tablas re-utilizadas de las bases de datos antes mencionadas y por las nuevas tablas creadas.

De la base de datos del gestor de contenidos se accedió a la tabla *cms_module_news* de donde se obtuvo las noticias cargadas en la IDE de la Universidad del Azuay. De esta se tomaron los campos: news_title (título de la noticia), news_data (cuerpo de la noticia), news_date (fecha de publicación), summary (resumen de la noticia), news_id (identificador de noticia). Esta información se filtra en función al campo *status* donde debe estar con el valor *published* que indica que la noticia se encuentra de acceso público en la web. A continuación se observa ver el código SQL de la vista generada a partir de esta información.

```
SELECT p.titulo, p.datos, p.fechor, p.resumen, p.id
  FROM dblink('host=localhost dbname=***** port=****  password=***** user=****'::text, 'SELECT
news_title,news_data,news_date,summary, news_id FROM cms_module_news where status="published"'::text)
p(titulo character varying(255), datos text, fechor timestamp without time zone, resumen text, id integer);
```

La base de datos del visor Mapbender se utilizó para determinar los servicios que se encuentran publicados y las capas de información disponibles para su visualización desde el visor. De esta manera se obtiene una concordancia entre la información publicada en la IDE de la Universidad del Azuay y la IDE UDA móvil. Mapbender, además, permite tener el control sobre las capas que se visualizan en su entorno y este control es usado por el administrador de la IDE de la Universidad del Azuay para limitar el acceso a recursos que todavía no están públicos por depuración de estilos o atributos.

De las 77 tablas de datos que estructuran el cliente ligero Mapbender se seleccionó las 8 tablas descritas en la figura 5 para re-utilizar contenidos que serán visibles desde la IDE UDA móvil. Entre ellas se tiene: 1) la tabla mb_user se encarga de almacenar la información de los usuarios que pueden acceder a este visor; 2) la tabla gui_mb_user representa el nivel de acceso de los usuarios sobre las aplicaciones; 3) la tabla mb_user_wmc almacena en estándar WMC los datos del mapa que se haya generado en el visor; 4) la tabla wmc_keyword contiene las palabras clave asociadas a estos mapas; 5) la tabla wms almacena la información de los servicios WMS que se encuentran disponibles para el visor de la IDE de la Universidad del Azuay; 6) la tabla layer permite conocer las características de cada capa como su nombre y el estado para determinar si están visibles para el usuario final; 7) la tabla gui contiene las aplicaciones creadas dentro de este cliente; y para finalizar 8) la tabla gui_layer permite determinar que capas están disponibles de acuerdo a la aplicación.

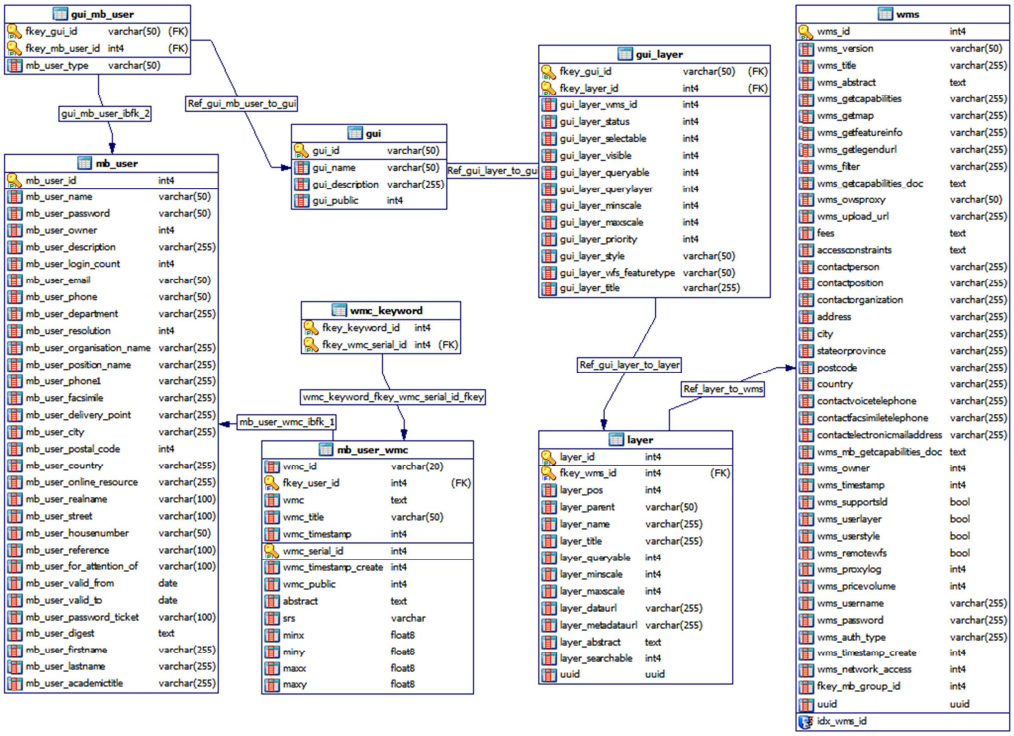

Figura 5. Esquema de tablas re-utilizadas del visor Mapbender

El mismo análisis se realizó en la base de datos del catálogo Geonetwork y se identificaron dos tablas a acceder (Figura 6): 1) la tabla metadata que contiene todos los metadatos publicados en el catálogo y 2) la tabla operationallowed que permite determinar cuales son metadatos que tienen el permiso de públicos para todos los grupos de usuarios de Geonetwork.

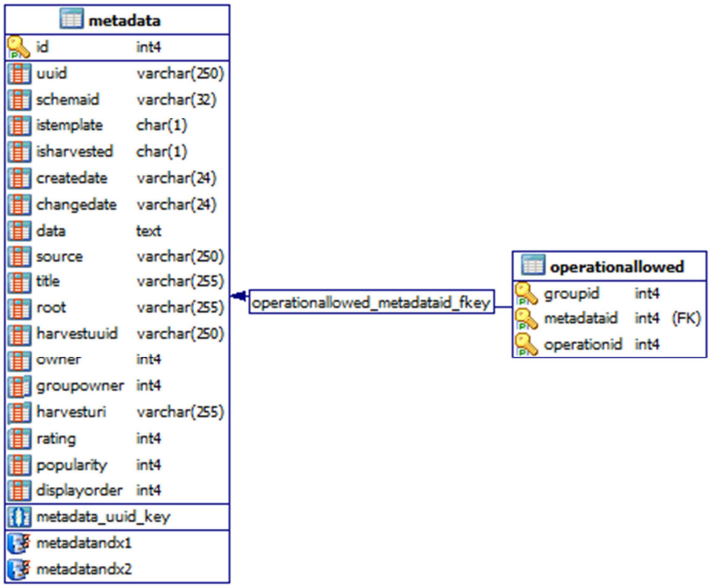

Figura 6. Esquema de tablas re-utilizadas de Geonetwork

La figura 7 presenta el esquema de las nuevas tablas creadas en esta tesis. Se parte de la tabla mini_user con la información de los usuarios que se han registrado en la IDE de la Universidad del Azuay. En la tabla mini_marcadores se almacena la información correspondiente a los marcadores generados por cada usuario. Para finalizar la tabla wmc_usuarios almacena una copia del mapa en el estándar WMC para que pueda ser accedido desde el visor de OpenLayers disponible para los usuarios.

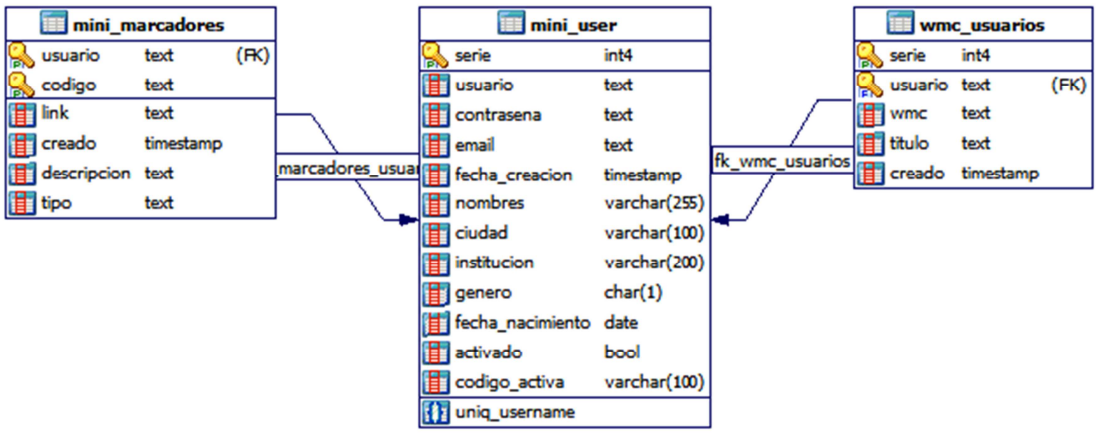

Figura 7. Esquema de las nuevas tablas creadas

2.5. Componentes del cliente

El reto principal para el desarrollo de los componentes del lado del cliente es adaptar los componentes fundamentales de una IDE convencional al entorno móvil, siguiendo para ello los requisitos mencionados anteriormente. Se han desarrollado tres componentes fundamentales para dar acceso a la IDE de la Universidad del Azuay desde un dispositivo móvil: 1) directorio de servicios WMS, 2) visor de mapas y 3) catálogo de metadatos. Adicionalmente, se desarrollaron otros cuatro componentes de soporte que facilitan el acceso móvil a una IDE y potencian las utilidades propias de un dispositivo móvil: 4) registro y gestión de usuarios, 5) marcadores, 5) escáner de códigos QR (disponible solo en Android) y 6) noticias.

El acceso a estos componentes (con excepción del escáner de códigos QR) puede ser accedido desde el interfaz web que consiste en un sitio adaptado a las potencialidades de un móvil y desde un aplicativo instalable en el móvil.

2.5.1. Directorio de servicios WMS

Este componente lista las direcciones web de los distintos servicios IDE para que los usuarios puedan copiar y utilizar los servicios desde otros clientes que soporten conexión a WMS como: OruxMaps, GvSig mini y Locus Free. Para utilizar esta información, dependiendo el aplicativo que use, se deberá especificar un archivo de conexión con los datos del servicio WMS o la dirección web (url) del servicio.

Para obtener esta información se accedió a la tabla gui_wms de Mapbender que permite determinar qué capas están en la aplicación llamada “visor_uda”, es decir que corresponden al visor Mapbender de la IDE de la Universidad del Azuay.

2.5.2. Visor de mapas

Este componente permite la visualización en un dispositivo móvil de todas las capas de información publicadas en la IDE de la Universidad del Azuay. El visor de mapas utiliza un conjunto de librerías conformadas por GeoExt Móvil (GXM) y JQuery. GeoExt Móvil (GXM), programando en lenguaje JavaScript, permite la construcción de la interfaz web de mapas para móviles con soporte Touch-Screen, es decir con sensibilidad al tacto en pantalla. GeoExt utiliza OpenLayers para el tratamiento de mapas y servicios wms, y además soporta la interconexión de mapas generados con la especificación WMC (Web Map Context). WMC es una estructura XML donde se detallan los componentes de capas y servicios de un mapa para que este se pueda reconstruir desde el visor OpenLayers y para continuar con el espacio de trabajo en diferentes visores.

JQuery se utiliza para acceder a la información de servicios y capas contenidas en el visor Mapbender y permite consultar la información necesaria para incluir nuevas capas en el visor móvil. En otras palabras, el visor en su versión móvil permite cargar las capas que estén publicadas en Mapbender, esto permite tener una concordancia entre lo publicado en el visor Mapbender y lo que se visualiza en el visor móvil. La lectura de los recursos de los servicios es asíncrona por lo cual para actualizar las capas y servicios disponibles en el visor móvil se debe acceder al panel de control de Mapbender y refrescar la cargar de los servicios WMS, esto producirá que se actualicen los contenidos de capas en el visor móvil.

Las personalizaciones y configuraciones de este cliente se realizaron sobre el archivo mobile-base.js. Entre las personalizaciones que se realizaron tenemos los servicios wms que se visualizan por defecto, proyección y sistema de referencia del cliente, zoom inicial y estilos. El código fuente de este archivo se basa principalmente en comandos de Javascript y objetos de la librería de OpenLayers. El detalle de este archivo se incluye en el anexo 6.2.

A continuación se detallan algunas de las funciones utilizadas por el visor de mapas.

La función gup de JavaScript permite capturar uno de los parámetros recibidos en el url usando para ello expresiones regulares para evaluar aquellos parámetros separados por el signo “&”.

La función agregar_capa sirve para cargar las capas seleccionadas del listado en el mapa. Se puede apreciar que el objeto map es el mapa y se usa el comando map.addLayer(capa) para agregar otras al visor.

```
function agregar_capa(capa)
{
var tmp;
var tmp1;
tmp = server_seleccionado.split(",");
tmp1 = tmp[1].split("?");
var c1 = new OpenLayers.Layer.WMS(capa.value,
                                                    tmp1[0] + "?",
                                                    {'layers': capa.value, transparent: true},
                                                    {isBaseLayer: false,units: "dd", projection: sm}  );
map.addLayer(c1);
}
```

Con esta función además se configura para que cada capa nueva que se agregue tenga la proyección UTM WGS 84 zona 17 sur con código EPSG: 32717.

Adicionalmente se modificó el archivo index.html para personalizar la ubicación y botones que están disponibles en el visor, llamar a funciones Ajax que traen información de la base de datos. En este archivo se usa la codificación y comandos de Ext. El detalle de estos cambios se documenta en el anexo 6.3.

La función capas_server llama a un panel que se construye a partir de la información de la base de datos donde en primera instancia se lista el nombre de los servicios y una vez seleccionado uno de ellos carga el listado de capas que pueden ser accedidos del servicio según las configuraciones establecidas en el visor de Mapbender.

```
function capas_server()
{
        if (!app.popup2) {
app.popup2 = new Ext.Panel({
floating: true,
modal: true,
centered: true,
hideOnMaskTap: true,
width: 240,
                                height: 240,
items: [{
html: '<img src="./img/cerrar.png" style="float:right;" onClick="app.popup2.hide();"><br><div style="font-size:12px;">Servidores:</div><br> <div class="x-panel-header" id="wms_servers" style="font-size:11px;"></div><br style="font-size:12px;"><div style="font-size:12px;">Capas</div><div class="x-panel-header" style="font-size:11px;" id="capas_server"></div>'
}],
scroll: 'vertical'
});
}
callAjax();
app.popup2.show('pop');

}
```

La función mapas_server crea la ventana o panel donde se visualizará el listado de mapas.

```
function mapas_server()
{
        if (!app.popup1) {
app.popup1 = new Ext.Panel({
floating: true,
modal: true,
centered: true,
hideOnMaskTap: true,
width: 240,
                                height: 240,
items: [{
html: '<img src="./img/cerrar.png" style="float:right;" onClick="app.popup1.hide();"><br>Mapas:<br> <div class="x-panel-header" id="mapas_usuarios" style="font-size:11px;"></div>'
}],
scroll: 'vertical'
});
}
                        cargar_mapas();
app.popup1.show('pop');

}
```

La función listar_capas carga el listado de capas dependiendo el servicio que ha sido seleccionado.

```
function listar_capas(obj)
{
server_seleccionado=obj.value;
$.ajax({
```

```
    type: 'GET',
    url: './ajax.php?id=113&criterio=' + obj.value,
    success: function(result) {
            var s= document.getElementById("capas_server");
            s.innerHTML="";
        $('#capas_server').append(result);
    }
  });

            //var res = document.getElementById("capas_server");
            //res.innerHTML=obj.value;
            }
```

La función callAjax carga en una lista de selección los nombres de los servicios que están disponibles en el visor.

```
function callAjax() {
  $.ajax({
    type: 'GET',
    url: './ajax.php?id=112',
    success: function(result) {
                    var s= document.getElementById("wms_servers");
                    s.innerHTML="";
        $('#wms_servers').append(result);
    }
  });
  }
```

La función cargar_mapas permite cargar el listado de los mapas que han sido almacenados por el usuario con el estándar WMC. En este listado se visualiza un botón por cada mapa que al ser presionado carga la información del mismo en el visor.

```
function cargar_mapas() {
  var usuario;
  usuario=gup('usuario');
            var tmpurl='./ajax.php?id=115&criterio=' + usuario;
   $.ajax({
    type: 'GET',
    url: tmpurl,
    success: function(result) {
                    var s= document.getElementById("mapas_usuarios");
                    s.innerHTML="";
        $('#mapas_usuarios').append(result);
    }
  });
  }
```

Si se ingresa al sistema web y se accede al visor de mapas existe la posibilidad de cargar los mapas almacenados (estándar WMC), ya sea desde el mismo móvil o los generados desde la web. De esta forma se puede usar un móvil y la PC para construir mapas que pueden ser compartidos y visualizados.

2.5.3.Metadatos

El objetivo de este componente es permitir la búsqueda y localización de información geográfica desde los dispositivos móviles, utilizando para ello elementos de los metadatos. Estos se encuentran almacenados en la base de datos de Geonetwork perteneciente al catálogo de metadatos del mismo nombre. Este catálogo permite la interconexión con bases de datos de diferentes fabricantes como por ejemplo: Mysql, Postgres, Oracle, etc. Para el caso de la IDE de la Universidad del Azuay se uso como base de datos Postgres. Los metadatos en la base de datos se encuentran almacenados en un campo de tipo texto manteniendo el formato XML del documento original. Esta estructura sigue las recomendaciones del perfil ecuatoriano de metadatos (PEM) (Conage, 2010).

El cliente móvil accede a la información del XML de los metadatos a través de funciones creadas durante el desarrollo permitiendo: 1) conectarse a la base de datos remota (DB Link) (2) usar funciones nativas del lenguaje PLPSQL de Postgres; y 3) construir vistas personalizadas que permitan la búsqueda y recuperación de los elementos de un archivo de metadato.

Se utiliza JQuery para acceder a la información básica del metadato que se encuentra almacenada en la base de datos y se formatea su presentación en el dispositivo móvil. Para acceder a esta información JQuery llama a aplicaciones de Php que toman la información de la base de datos. El hipervínculo del metadato original puede ser almacenado como un marcador de tal forma que cuando se acceda desde la web se visualizará el metadato completo en Geonetwork.

El siguiente SQL muestra un ejemplo de como se accede a cada uno de los nodos del árbol XML usando la función xpath. Esta función fue de gran utilidad para poder acceder a elementos puntuales del metadato.

Consulta SQL para acceder a elementos de XML

```
SELECT metadatos.uuid,
xpath('/MD_Metadata/identificationInfo/MD_DataIdentification/citation/CI_Citation/title/Charac
terString/text()'::text, replace(replace(metadatos.data, 'gmd:'::text, ''::text), 'gco:'::text,
''::text)::xml)::text AS titulo,
xpath('/MD_Metadata/identificationInfo/MD_DataIdentification/abstract/CharacterString/text()':
:text, replace(replace(metadatos.data, 'gmd:'::text, ''::text), 'gco:'::text, ''::text)::xml)::text AS
abstract, xpath('/MD_Metadata/dateStamp/DateTime/text()'::text,
replace(replace(metadatos.data, 'gmd:'::text, ''::text), 'gco:'::text, ''::text)::xml)::text AS
```

```
fecha_creacion,
xpath('/MD_Metadata/identificationInfo/MD_DataIdentification/descriptiveKeywords/MD_Key
words/keyword/CharacterString/text()'::text, replace(replace(metadatos.data, 'gmd:'::text,
''::text), 'gco:'::text, ''::text)::xml)::text AS palabras_clave,
xpath('/MD_Metadata/contact/CI_ResponsibleParty/individualName/CharacterString/text()'::te
xt, replace(replace(metadatos.data, 'gmd:'::text, ''::text), 'gco:'::text, ''::text)::xml)::text AS
responsables
  FROM metadatos;
```

La función xpath de Postgres solo pudo ser utilizada en la versión 9.1 de esta base de datos. Como la información del catálogo de Geonetwork se encuentra en la versión 8.4 de Postgres se usó la funcionalidad de conexión de datos “dblink” para acceder a la misma.

Vista metadatos desde donde realiza la toma de datos de la base de datos de Geonetwork

```
SELECT t1.uuid, t1.data, t1.istemplate, t1.isharvested
  FROM dblink('dbname=geonetwork host****** user**** password****** port=*****'::text,
'select uuid,data,istemplate,isharvested from metadata where id in (select metadataid from
operationallowed where groupid=1 and operationid=0)'::text) t1(uuid text, data text,
istemplate text, isharvested text)
 WHERE t1.istemplate ~~ 'n'::text AND t1.isharvested = 'n'::text
```

Output pane

Data Output | Explain | Messages | History

	uuid text	data text	istemplate text	isharvested text
1	44b36dfc-9224-4070-b376-7fa4f5765eba	<gmd:MD Metadata xmlns:gmd="http://www.isotc211.org/2005/gmd" xmlns:xsi="http://www.w3.org/2001/X	n	n
2	92a076d1-9f39-4671-90d9-e0d9d870e75c	<gmd:MD Metadata xmlns:gmd="http://www.isotc211.org/2005/gmd" xmlns:xsi="http://www.w3.org/2001/X	n	n
3	b12ea636-79ba-45ac-a5e3-a4fb259486ea	<gmd:MD Metadata xmlns:gmd="http://www.isotc211.org/2005/gmd" xmlns:xsi="http://www.w3.org/2001/X	n	n
4	b2736d58-d1ad-435f-a123-fe1b31135e4b	<gmd:MD Metadata xmlns:gmd="http://www.isotc211.org/2005/gmd" xmlns:xsi="http://www.w3.org/2001/X	n	n
5	4db41989-3085-4b13-99a9-fd2964b94cfb	<gmd:MD Metadata xmlns:gmd="http://www.isotc211.org/2005/gmd" xmlns:xsi="http://www.w3.org/2001/X	n	n
6	17de33a8-627e-4be1-ac01-d6f319499f2a	<gmd:MD Metadata xmlns:gmd="http://www.isotc211.org/2005/gmd" xmlns:xsi="http://www.w3.org/2001/X	n	n
7	17da09e9-ce56-4232-88df-0913fab1c218	<gmd:MD Metadata xmlns:gmd="http://www.isotc211.org/2005/gmd" xmlns:xsi="http://www.w3.org/2001/X	n	n
8	96a17bed-4f63-422b-b00e-90338675cb83	<gmd:MD Metadata xmlns:gmd="http://www.isotc211.org/2005/gmd" xmlns:xsi="http://www.w3.org/2001/X	n	n
9	04bdcc08-2418-45dc-a182-fa4866e054ff	<gmd:MD Metadata xmlns:gmd="http://www.isotc211.org/2005/gmd" xmlns:xsi="http://www.w3.org/2001/X	n	n
10	4732f16f-a291-494a-8b02-e893986414e0	<gmd:MD Metadata xmlns:gmd="http://www.isotc211.org/2005/gmd" xmlns:xsi="http://www.w3.org/2001/X	n	n
11	c1831601-2a9c-4616-9e29-8823539eecc7	<gmd:MD Metadata xmlns:gmd="http://www.isotc211.org/2005/gmd" xmlns:xsi="http://www.w3.org/2001/X	n	n
12	7d468c46-e27c-4d2f-80a8-1040382957c9	<gmd:MD Metadata xmlns:gmd="http://www.isotc211.org/2005/gmd" xmlns:xsi="http://www.w3.org/2001/X	n	n

Figura 8. Resultado de la consulta de Postgres para visualizar metadatos

Para cumplir con el requisito de reducir el volumen de información a trasferir, se limitó el número de elementos de metadatos a recuperar. Para ello se seleccionaron los elementos de metadatos más relevantes como: título, resumen, palabras clave, responsable del metadato y fecha de creación.

2.5.4.Registro y acceso de usuarios

Para realizar la gestión de contenidos personalizados, la IDE de la Universidad del Azuay implementa la opción para crear una cuenta de usuario, la misma que puede ser accedida

desde el móvil o la web. Esta trabaja de forma común a la de otras comunidades donde el usuario ingresa sus datos y luego se da de alta a los servicios.

Al acceder al sistema se presentará un panel de control con las operaciones que podrá realizar cada usuario como por ejemplo: acceso al visor personal, consulta de marcadores, posibilidad de publicar los marcadores en una red social y visualizar su marcador como un código de respuesta rápida QR.

2.5.5. Marcadores

La implementación de marcadores tiene el objetivo sincronizar contenidos de la IDE de la Universidad del Azuay con los de la IDE UDA móvil. En móvil se presenta la información resumida mientras que para revisar detalles más extensos se accede a la versión web. Los marcadores permiten almacenar resultados de búsquedas de metadatos y mapas almacenados con el estilo WMC.

Este contenido se almacena en la base de datos “mide” donde se centraliza las funciones de la IDE UDA móvil. Dentro de la tabla mini_marcadores se almacenan los datos del marcador generado por el usuario, clasificado según su tipo (metadato o mapa) y donde también se almacena el link a la versión web. En caso de acceder a un marcador tipo metadato, se re-direccionará al catálogo de Geonetwork donde podrá visualizar el metadato completo. En caso de acceder a un marcador de mapa, se cargará el visor OpenLayers con el mapa seleccionado.

En particular, para la visualización y construcción de mapas se puede realizar las primeras aproximaciones desde un ambiente móvil y posteriormente culminar la construcción del mapa en la web.

En caso de los metadatos, el cliente móvil accede a la información principal del mismo. Para acceder al metadato completo el marcador, a través de un hipervínculo, carga el recurso para revisarlo en Geonetwork desde una PC.

Los marcadores son accesibles únicamente por el usuario que los crea. Por ello, para utilizar esta funcionalidad en primer lugar se debe crear una cuenta de usuario, la misma que permitirá acceder a diferentes paneles desde donde se administran los contenidos propios de cada usuario. Si se desea compartir este marcador a través de las redes sociales, este perderá su propiedad de dato privado y se convertirá en un dato público.

2.5.6.Lector de códigos QR

Los códigos QR están siendo ampliamente utilizados para acceder a vínculos de descarga desde aplicativos móviles. Su principal ventaja es que evitan problemas de lingüística y errores de tipeo, además de agilizar y facilitar la búsqueda y descarga de recursos.

En la versión Android de la IDE móvil los códigos QR se utilizan principalmente para dos fines: el primero para acceder a mapas y metadatos, en este caso los recursos se abrirán de forma automática en el navegador web del móvil; y segundo para codificar los criterios de búsquedas de contenido de metadatos a través de palabras clave. En este caso, el código QR agregará automáticamente a la ventana de búsqueda el criterio ingresado por el creador del marcador de metadatos, de tal forma que permita al usuario editar su contenido antes de realizar dicha consulta.

La funcionalidad de código QR utiliza como base las librerías del aplicativo “ZXing Barcode Scanner”. Para poder utilizar la funcionalidad es necesario previamente descargar este aplicativo de la tienda de Google. Este lector solo se encuentra disponible en el aplicativo Android.

Desde el visor web se usa el hipervínculo de los marcadores para transformarlos en códigos QR usando para ello una librería libre de Php llamada “tantaqrcode.php”. Esta nos permite crear objetos de tipo imagen con el formato QR estándar a través de parámetros como: ancho, url al que apuntará el código, formato de imagen. El detalle de esta implementación lo encontramos en el anexo 6.4.

2.5.7.Noticias

Cualquier contenido que se encuentre almacenado en la base de datos puede ser capturado, resumido y formateado para un móvil. Prueba de ello es la sección de noticias que presenta el título y resumen de las noticias principales de la IDE de la Universidad del Azuay. Se añade también la opción de revisar el contenido completo en la web a través de un hipervínculo.

3. RESULTADOS Y DISCUSIÓN

El aplicativo resultante es accesible a través de dos interfaces. La primera, desde el navegador del dispositivo móvil a través de http://gis.uazuay.edu.ec. Esta página será re-direccionada automáticamente a la dirección: http://gis.uazuay.edu.ec/miniide/, la cual contiene la versión móvil de la IDE (Figura 9.a). La segunda interfaz es un aplicativo para Android que debe ser instalado en los dispositivos móviles (Figura 9.b). Este aplicativo se descarga desde http://gis.uazuay.edu.ec/miniide/IDE_APP.apk. La diferencia primordial es que en la primera interfaz solo se interactúa con la IDE de la Universidad del Azuay mientras que con la segunda además de eso se puede interactuar con la funcionalidad y sensores propios de un móvil inteligente, como lo es su cámara fotográfica.

Figura 9. Aplicación IDE UDA móvil. a) Vista desde el navegador. b) Vista desde celular Android

La figura 11 representa cómo los o los datos obtenidos de los diferentes catálogos se enlazaron, resumieron y presentaron en la construcción de los componentes de esta IDE-UDA móvil.

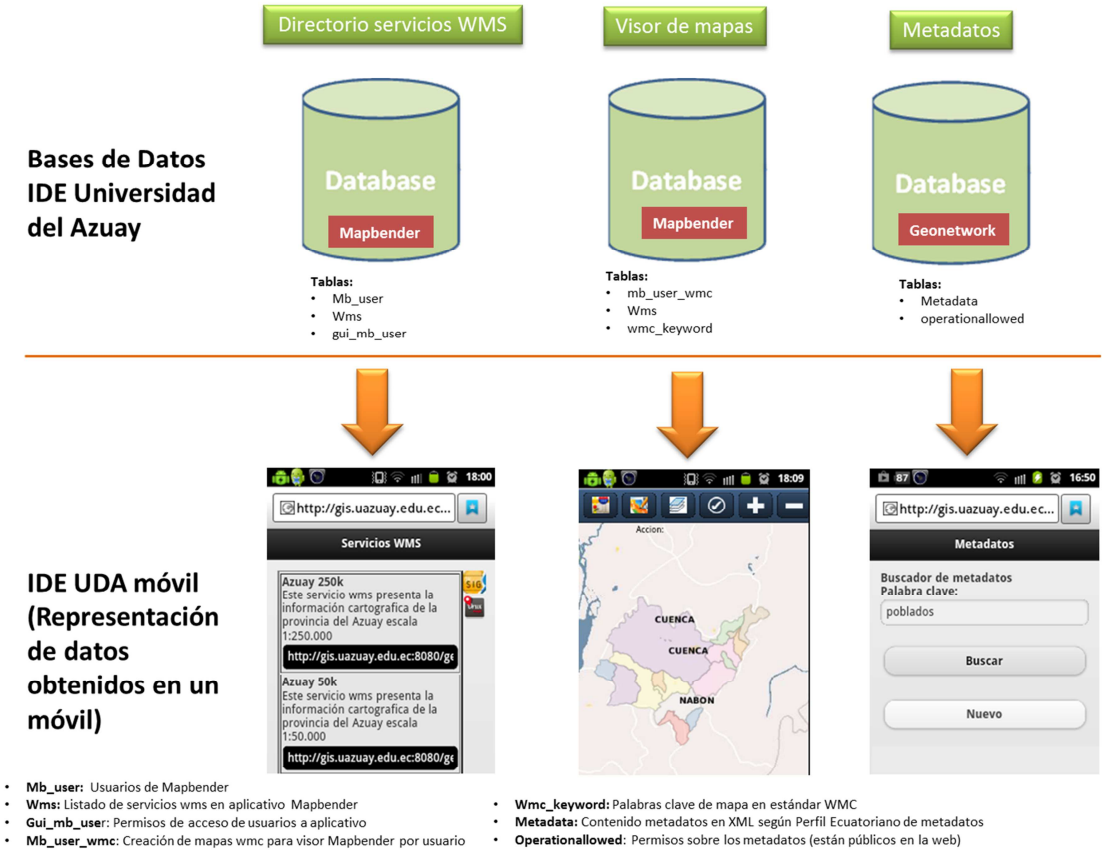

Figura 10. Esquema de datos - componente IDE UDA móvil

3.1.1. Directorio de servicios WMS

La figura 11 muestra la interfaz con el listado de las direcciones web de los geoservicios de la IDE de la Universidad del Azuay. El directorio de servicios WMS obtiene la información de la base de datos Mapbender para mantener concordancia con los servicios y capas visibles desde este visor. En próximas versiones del aplicativo se prevé poder obtener la información directamente de la petición GetCapabilities de cada servicio.

Figura 11. Directorio de servicios WMS contenidos en la IDE de la Universidad del Azuay

Se puede apreciar que en la parte superior derecha de la pantalla se puede descargar archivos de ejemplo sobre como configurar estos servicios WMS para que se conecten a GvSig mini y OruxMaps.

Las figuras 12.a, 12.b y 12.c muestran ejemplos de cómo los servicios WMS de la IDE de la Universidad del Azuay pueden ser consumidos desde SIG adaptados para móviles como GvSIG, OruxMaps, y Locus Free.

(a) *(b)* *(c)*

Figura 12. Consumo de servicios WMS desde SIG móviles. a) Vista de capa de clasificación de suelos del Azuay en GvSig mini; b) Vista de capa de clasificación de suelos del Azuay en OruxMaps; y c Vista de capa de clasificación de suelos del Azuay en Locus Free.

Para OruxMaps y GvSig mini fue necesario que los detalles de conexión al servicio WMS sean definidos en un archivo que se carga al programa (GvSig mini: layers.txt y OruxMaps: wms_services.xml). Estos archivos de configuración pueden ser descargados desde la sección de directorio de servicios WMS.

Para cargar los servicios WMS en GvSig mini se procedió a definir los parámetros de conexión en el archivo layers.txt ubicado dentro de la carpeta layers de la instalación de este programa. El código de este archivo se detalla en la figura 13, donde los parámetros modificados se encuentran resaltados en negrita. Dos parámetros fundamentales a incluir son la versión del servicio WMS (1.1.0) y el código de proyección EPSG: 4326 (sistema de referencia para visualizar los datos). Esto es para que GvSig mini interprete adecuadamente la información. Además, la capa se debe re-proyectar desde su sistema de referencia original a la proyección WGS84 de coordenadas geográficas (EPSG: 4326). Esto puede involucrar la modificación de los parámetros de publicación de la capa en el servidor de mapas.

101|**IDE UDA (Azuay -250k);**5[>],**http://gis.uazuay.edu.ec:8080/geoserver/azuay_250k/wms**,image/png,20,0,256**,-79.76406848538535,-3.631303001102388,-79.76406848538535,-3.63130300110238 8,-78.41964807480329,-2.49527028208188,EPSG:4326**,0.09375000000000004:0.046875000000000002:0.02343750000000001:0.011718750000000005:0.005859375000000003:0.0029296875000000013:0.0014648437500000007:7.324218750000003E-4:3.6621093750000016E-4:1.8310546875000008E-4:9.155273437500004E-5:4.577636718750002E-5:2.288818359375001E-5:1.1444091796875005E-5:5.7220458984375025E-6:2.8610229492187513E-6:1.4305114746093756E-6:7.152557373046878E-7:3.576278686523439E-7:1.7881393432617195E-7:8.940696716308598E-8,**cantones_azuay_hcpa_250k|Azuay_suelos|centros_poblados_azuay_250k|cerros_lomas _azuay_250k,1.1.1**,text/html

Figura 13. Ejemplo archivo de conexión GvSig Mini con servicio WMS de la IDE de la Universidad del Azuay

Para cargar servicios WMS en el visor OruxMaps se debe editar el archivo denominado wms_services.xml el cuan se encuentra dentro de la carpeta mapfiles de la instalación de este cliente. El detalle del mismo puede ser apreciado a continuación (Figura 14):

```
▼<wms_services>
  ▼<wms>
     <name>IDE UDA(Azuay-250k)</name>
     <uid>12</uid>
     <!-- unique identifier in your database cache; >1000  -->
     <desc>Prov Azuay escala 1:250.000</desc>
    ▼<url>
       http://gis.uazuay.edu.ec:8080/geoserver/azuay_250k/wms?
     </url>
     <minzoomlevel>0</minzoomlevel>
     <!--  0 to 20  -->
     <maxzoomlevel>20</maxzoomlevel>
     <!--  0 to 20  -->
     <version>1.1.1</version>
     <!--  do not change  -->
     <layers>azuay_suelos</layers>
     <coordinatesystem>EPSG:4326</coordinatesystem>
     <!--  do not change  -->
     <format>image/jpeg</format>
     <cache>1</cache>
     <!--  not in use  -->
   </wms>
 </wms_services>
```

Figura 14. Vista archivo XML de configuración de servicios WMS en OruxMaps

En el caso de Locus free no fue necesario definir ningún tipo de archivo especial (Figura 15), sino que permitió acceder al servicio a través de la herramienta dentro del mismo software donde luego de colocar la dirección del servicio se lista las capas disponibles y de ellas se selecciona las que deseamos cargar al mapa.

Figura 15. Carga de servicios WMS desde aplicación Locus Free

3.1.2. Visor de mapas

La figura N° 16 muestra la interfaz del visor de ma pas móvil. La interfaz permite ampliar o reducir el mapa por medio de funcionalidad touch screen (depende versión de Android) o por medio de botones, almacenar y recuperar mapas generados por el usuario en el estándar WMC, así como también encender y apagar las capas de los geoservicios de la IDE de la Universidad del Azuay.

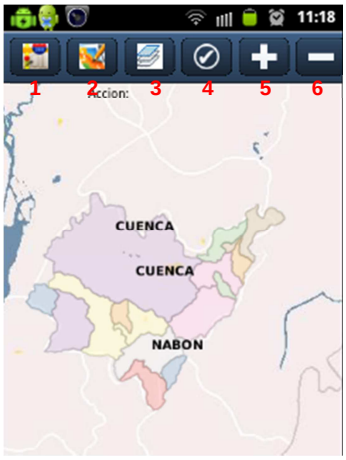

Figura 16. Visor de mapas móvil

Al abrir el visor se desplegará un mensaje de advertencia indicando que para almacenar los mapas que se construyen en el visor se debe acceder con su usuario antes de usar las opciones de recuperación y almacenamiento de datos del visor.

Por defecto se visualizarán las capas de los cantones de la provincia del Azuay del servicio WMS Azuay 250k y el perfil continental provisto por Metacarta. En la parte superior se dispone

del menú de opciones. El resto de la pantalla es ocupada por el mapa. El mapa se puede navegar a través del tacto con la pantalla o con las herramientas de zoom.

La primera opción del menú de opciones permite cargar los mapas que han sido almacenados con anterioridad con el estilo WMC (Figura 17). El segundo botón permite almacenar el mapa que se encuentre cargado actualmente en pantalla. Esto con la finalidad de que la construcción del mismo se pueda ir realizando en etapas y continuar con su creación en el ambiente web en caso de ser necesario. El tercer botón permite acceder a la ventana de los recursos WMS del visor. En primera instancia una lista muestra los servicios disponibles, luego de seleccionar uno de ellos se procede a cargar la capa en el visor (Figura 18). El cuarto botón permite activar/desactivar las capas que se encuentren activas al momento en el visor. Tarea muy similar a la que se puede observar en un SIG (Figura 19). El quinto y sexto botón son para realizar acercamientos y alejamientos sobre el mapa respectivamente.

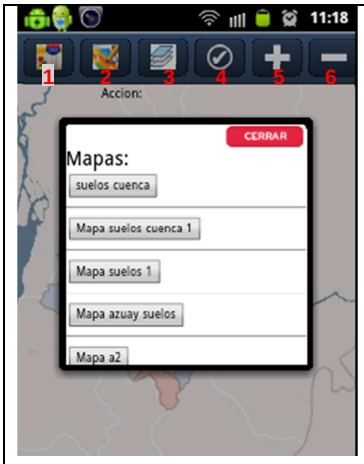

Figura 17. Opción de recuperación de mapas WMC almacenados

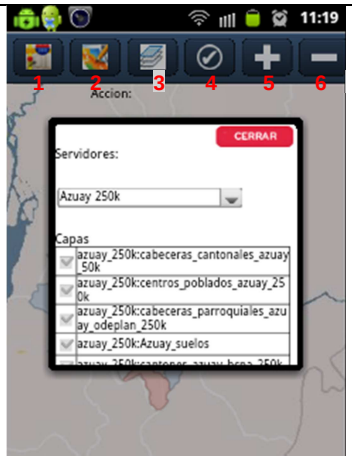

Figura 18. Servicios WMS y capas disponibles de cada servicio

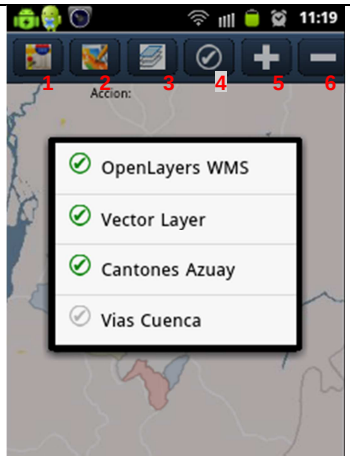

Figura 19. Opción para activar/desactivas capas del visor

3.1.3.Metadatos

La interfaz de metadatos permite realizar búsquedas de información por palabras clave (figura 18.a). Las búsquedas se realizan en los elementos de metadatos de título, palabras claves y resumen del metadato. La respuesta a la consulta es el número de registros encontrados y un listado de la información básica del metadato, es decir título, resumen, fecha de creación, palabras claves, responsables y códigos de identificación del metadato en el catálogo (figura 18.b).

(a) (b)

Figura 20. Componente de búsqueda de metadatos. a) Interfaz de búsqueda y b) interfaz de recuperación de resultados

Adicionalmente, se muestra el marcador del metadato original para que el metadato completo pueda ser recuperado desde la interfaz web (Figura 21). El metadato completo puede ser accedido y consultado desde el interfaz de Geonetwork y allí se puede revisar toda la información del mismo como información suplementaria, puntos de contacto, limitaciones de acceso y uso de la información, entre otros.

Figura 21. Vista del metadato en el catálogo Geonetwork

3.1.4.Registro y acceso de usuarios

Tanto la IDE web como la móvil presentan la opción de crear una cuenta de usuario, la misma permitirá la creación de contenidos personalizados de usuario para que se ajuste a las necesidades de cada persona (Figura 22).

Figura 22. Registro de usuarios

El panel de control desplegado para cada usuario registrado se muestra en la figura 23. La primera de las opciones es el visor basado en OpenLayers (figura 25) que permitirá visualizar y generar mapas con estándar WMC para que puedan ser accedidos en la web y en el móvil (apartado 3.1.5, Figura 24). Este mantiene sincronía con los mapas generados desde el visor móvil. La segunda opción son los marcadores, es decir los vínculos que permitirán visualizar los contenidos consultados en el móvil desde una vista web hacia la información completa. Esta opción se detallara en profundidad en el apartado 3.1.5. La tercera opción es el generador de códigos QR de los marcadores que permitirá colocar los marcadores de forma pública con la finalidad de compartir estos contenidos en la web a través de fuentes impresas o redes sociales en formato de código de respuesta rápida QR. Esta opción se detallara en profundidad en el apartado 3.1.6.

Figura 23. Panel de control de usuario

3.1.5.Marcadores

La opción marcadores despliega el listado de todos los marcadores que han sido almacenados en la versión móvil, clasificados según su tipo (Mapa o Metadato) (Figura 24). En caso de acceder a un marcador tipo metadato, el hipervínculo re-direccionará al catálogo de Geonetwork donde se obtiene el metadato completo (Figura 21). En el caso de seleccionar un mapa se cargará el visor OpenLayers con el mapa seleccionado (Figura 25).

Al momento de realizar la búsqueda de contenidos por cada uno de ellos se presenta la opción para almacenarlo en la base de datos.

(a) (b)

Figura 24. Administración de marcadores de la IDE UDA móvil. a) Visualización en la web de los marcadores de usuarios y b) visualización de marcadores en el móvil

Figura 25. Sincronización de mapas entre el móvil y la web

3.1.6.Lector de códigos QR

Desde el aplicativo Android podemos utilizar la cámara como lector de códigos QR (funcionalidad semejante a los códigos de barra). Estos códigos QR se codificarán en formato de URL enviando la información y parámetros necesarios para el caso de las búsquedas envíe el criterio usado para la consulta o en caso de un hipervínculo normal abrirá el mismo en el navegador web del móvil. El lector re-direccionará al mapa o metadato según el código en su versión web para que contenga toda la información necesaria (Figura 21).

En primer lugar los marcadores creados por el usuario se visualizan automáticamente como código QR en la opción Generador QR del panel de control de usuario. Dentro de este mismo panel se agregó la opción de compartir estos códigos en redes sociales (Facebook) con la finalidad que otros usuarios vean nuestros datos. Cabe recalcar que al momento de compartir esta información el marcador asume un estado de público y puede acceder cualquier persona.

Este marcador puede distribuirse de forma digital o impresa. Estos se codifican de tal forma que si son leídos por el escáner QR de la aplicación de la IDE UDA móvil se re-direccionará a la opción más adecuada.

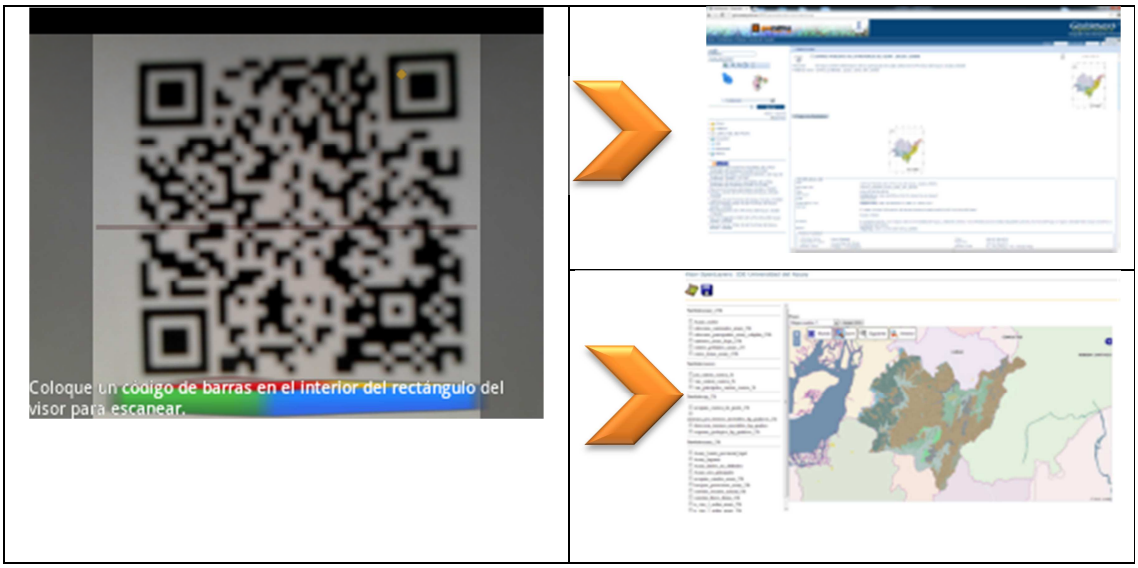

Figura 26. Marcadores en sincronía con metadatos y mapas

3.1.7.Noticias

La figura 27 muestra la sección de últimas noticias, que son un resumen de las contenidas en la versión web.

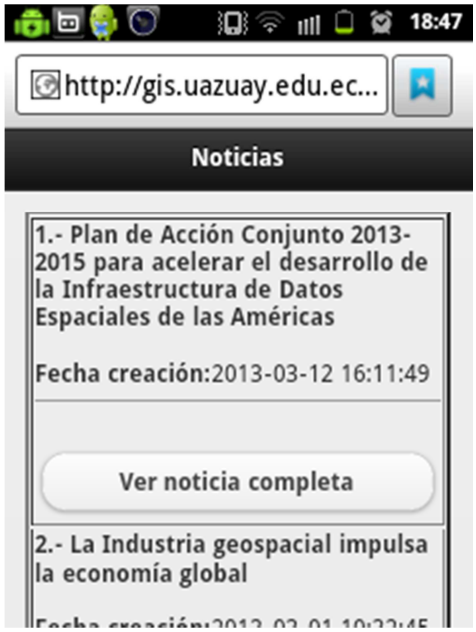

Figura 27. Sección de noticias vista desde el móvil

4. CONCLUSIONES

El objetivo de esta tesis ha sido la creación de un aplicativo para acceder a la IDE de la Universidad del Azuay desde un dispositivo móvil inteligente, para que los usuarios puedan explotar y extender las funcionalidades de acceso a la geoinformación. El acceso se realiza desde la web o en su defecto desde el aplicativo que puede ser instalado en los móviles con sistema Android.

El aplicativo móvil contiene siete componentes fundamentales: 1) directorio de servicios WMS, 2) visor de mapas y 3) catálogo de metadatos. Adicionalmente, también se incluyeron componentes de 4) registro y gestión de usuarios, 5) marcadores, 6) noticias y 7) lector de códigos QR disponible solo en la versión Android. Como principales ventajas de la implementación se puede mencionar la modularidad de su arquitectura, pudiendo fácilmente personalizar el aplicativo, eliminar componentes no deseados o incluir nuevos componentes que se desarrollen a futuro. El componente de búsqueda y consulta de metadatos es un componente innovador y no se ha localizado en la literatura otros componentes similares en dispositivos móviles. Este concepto podría ser adoptado por varias instituciones públicas que no desean utilizar el interfaz de Geonetwork sino han personalizado con la finalidad de simplificar el uso del mismo y permitir que esta herramienta ya no sea de uso exclusivo de personal especializado.

La utilización de vistas y dblink en la base de datos permite que cualquier contenido pueda ser capturado (sin importar su ubicación) y formateado para un móvil con la finalidad de resumir el mismo. Esto tiene una importancia vital para controlar la limitación de ancho de banda y el volumen información a transmitir versus la capacidad de ancho de banda que soporta un móvil en la actualidad. Un ejemplo de ello se observa en la sección de noticias. También ligado a la limitación de ancho de banda, la implementación del visor de mapas con GeoExt Móvil ha resultado satisfactoria dado que su interfaz es muy ligera y amigable.

La implementación ha desarrollado componentes del lado del cliente. Es decir, se ha respetado la arquitectura y configuración de la IDE tradicional. Esto permite que, utilizando los desarrollos de esta tesis, una IDE tradicional pueda sea fácilmente extendida con una versión móvil.

Algunas limitaciones se observan en cuanto el aplicativo (IDE_APP.apk) solo puede utilizarse en plataformas Android. Esta limitación es superada por la versión web ya que se puede utilizar en otros navegadores para móviles e incluso en ciertos navegadores web de PC. Adicionalmente, para la construcción del aplicativo Android se utilizó App Inventor, que si bien

es de fácil uso presenta limitaciones en comparación al uso de lenguajes de programación nativos de Android.

Como futuros desarrollos y para continuar expandiendo las funcionalidades del aplicativo se ampliará la interacción con el dispositivo GPS y sensores que puedan capturar información de su entorno, se puede crear visores con información específica según las necesidades del usuario. Finalmente, los resultados de esta tesis contribuyen con un acceso a la información geográfica más rápida y oportuna utilizando una interfaz flexible y de fácil uso.

5. REFERENCIAS BIBLIOGRÁFICAS

- App Inventor. (n.d.). Retrieved from http://appinventor.mit.edu/
- Brinkhoff, T. (2008). Supporting Mobile GIS Applications by Geospatial Web Services. 21st Congress of the International Society for Photogrammetry and Remote Sensing, Beijing.
- Brähler, S. (2010). Analysis of the Android Architecture.
- CONAGE. (2010). Perfil Ecuatoriano de metadatos R. O. (28/09/2010). Retrieved from http://www.sni.gob.ec/c/document_library/get_file?uuid=99849f75-38ed-430f-8a56-66d0a9fa8ff7&groupId=10156
- Castelein, W., Grus, L., Crompvoest, J., & Bregt, A. (2010). A characterization of volunteered Geographic Information. Guimaraes, Portugal.
- Conage. (2010). Políticas Nacionales de Información Geoespacial. Retrieved from http://www.sni.gob.ec/c/document_library/get_file?uuid=dd46b294-75e1-4732-a9e0-037eb0f23360&groupId=10156
- Conage. (2012). Estrategias para la aplicación de las Políticas Nacionales de Información Geoespacial. Retrieved from http://www.sni.gob.ec/c/document_library/get_file?uuid=600aac2c-2b6c-4899-9347-d8b003704d2c&groupId=10156
- Delgado, O., & Ochoa, P. (2011). Tutorial de prácticas ArcGis Versión 9.2. Cuenca: Universidad del Azuay.
- ESRI. (n.d.). Arcgis. Retrieved from http://www.esri.com/software/arcgis
- ESRI. (n.d.). ArcCatalog. Retrieved April 15, 2013, from http://help.arcgis.com/es/arcgisdesktop/10.0/help/index.html#//006m00000069000000
- Foerster, T., Nüst, D., Bröring, A., & Jirka, S. (n.d.). Discovering the Sensor Web through Mobile Applications.
- G, & Gil, B. T. (2013). Aplicaciones, Android: Programación de. Retrieved April 1, 2013, from http://miriadax.net/web//android_programacion
- Ganti, R. K., Ye, F., & Lei, H. (2011). Mobile crowdsensing: Current state and future challenges. Communications Magazine, IEEE, 49(11), 32-39. IEEE.

- Geonetwork. (n.d.). Retrieved April 1, 2013, from http://geonetwork-opensource.org/
- Goodchild, M. F. (2007). Editorial : Citizens as Voluntary Sensors : Spatial Data Infrastructure in the World of Web 2 . 0. International Journal, 2(2), 24-32. doi:10.1016/j.jenvrad.2011.12.005
- INEC. (2011). Reporte anual de estadísticas sobre tecnologías de la información y comunicaciones (TIC's). Retrieved from www.inec.gob.ec/sitio_tics/presentacion.pdf
- Locus Map. (n.d.). Retrieved from http://www.locusmap.eu/
- OpenLayers mobile. (n.d.). Retrieved April 15, 2013, from www.openlayers.org
- Oruxmaps. (n.d.). Retrieved from http://www.oruxmaps.com/
- Pacheco, D., Ballari, D., & Delgado, O. (2012). Infraestructura de Datos Espaciales de la Universidad del Azuay. LatinOSGIS 2012. First Latin American Congress of Free and Open Source GIS. Retrieved from http://www.fiec.espol.edu.ec/LatinOSGIS/index.php/es-descargas
- Quantum Gis for Android. (n.d.). Retrieved April 15, 2013, from http://hub.qgis.org/projects/android-qgis
- Terrestris GmbH & Co. KG. (n.d.). GeoExt Mobile (GXM). Retrieved April 8, 2013, from http://www.terrestris.de/en/open-source/byterrestris/gxm/
- Wolber, D. (2011). App Inventor and Real-World Motivation.
- gvSIGmini. (n.d.). Retrieved from https://confluence.prodevelop.es/display/GVMN/Home

6. ANEXOS

6.1. Fragmento código de bloques escáner códigos QR (App inventor)

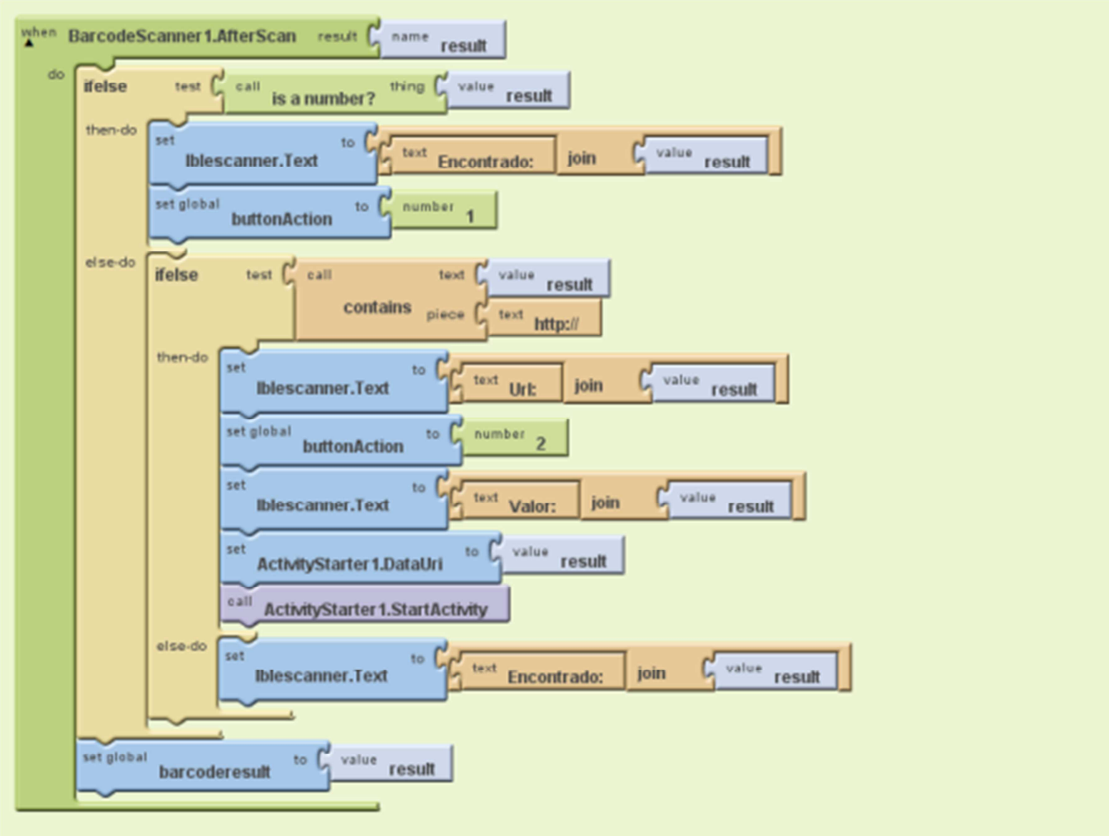

Figura 28. Fragmento código de bloque de App Inventor para lectura de códigos QR

6.2. Anexo Código fuente mobile-base.js

```
// API key for http://openlayers.org. Please get your own at
// http://bingmapsportal.com/ and use that instead.
var apiKey = "AqTGBsziZHIJYYxgivLBf0hVdrAk9mWO5cQcb8Yux8sW5M8c8opEC2lZqKR1ZZXf";

// initialize map when page ready
var map;
var gg = new OpenLayers.Projection("EPSG:4326");
var sm = new OpenLayers.Projection("EPSG:900913");

var init = function (onSelectFeatureFunction) {

    var vector = new OpenLayers.Layer.Vector("Vector Layer", {});
            var cazuay = new OpenLayers.Layer.WMS("Cantones Azuay",
                                                  "http://gis.uazuay.edu.ec:8080/geoserver/azuay_250k/wms?",
                                                  {'layers':    'azuay_250k:cantones_azuay_hcpa_250k',    transparent:
false}
                                                   );
            var lb= new OpenLayers.Layer.WMS(
                                                  "Provincias",
                                                  "http://www.geoportaligm.gob.ec/nacional/wms",
                                                  {'layers': 'igm:provincias', transparent: true, visibility: true}
                                                  );
```

```
            var wmsOverlay = new OpenLayers.Layer.WMS( "OpenLayers WMS",
   "http://vmap0.tiles.osgeo.org/wms/vmap0",
   {layers: 'basic', isBaseLayer:false});

            var base= new OpenLayers.Layer.WMS(
                                                        "Vias",
                                                        "http://www.geoportaligm.gob.ec/nacional/wms",
                                                        {'layers': 'igm:vias', transparent: true, visibility: true}
                                                        );
//    {type: G_SATELLITE_MAP, sphericalMercator: true}


            var layer0 = new OpenLayers.Layer.WMS(
           "Cuenca",
           "http://gis.uazuay.edu.ec:8080/geoserver/cuenca/wms",
           {layers: "cuenca:vias_canton_cuenca_5k",format: 'image/png', transparent: true},
                                    {visibility: false}
    );

    var sprintersLayer = new OpenLayers.Layer.Vector("Sprinters", {
        styleMap: new OpenLayers.StyleMap({
            externalGraphic: "img/mobile-loc.png",
            graphicOpacity: 1.0,
            graphicWidth: 16,
            graphicHeight: 26,
            graphicYOffset: -26
        })
    });

    var sprinters = getFeatures();
    sprintersLayer.addFeatures(sprinters);

    var selectControl = new OpenLayers.Control.SelectFeature(sprintersLayer, {
        autoActivate:true,
        onSelect: onSelectFeatureFunction});

    var geolocate = new OpenLayers.Control.Geolocate({
        id: 'locate-control',
        geolocationOptions: {
            enableHighAccuracy: false,
            maximumAge: 0,
            timeout: 7000
        }
    });
    // create map

                        /*
                         options = {
            projection: new OpenLayers.Projection("EPSG:900913"),
                                    displayProjection: new OpenLayers.Projection("EPSG:4326"),
            units: "m",
            numZoomLevels: 22,
            maxResolution: 156543.0339,
            maxExtent: new OpenLayers.Bounds(-20037508, -20037508,
                                    20037508, 20037508.34)
        };
                        */

    map = new OpenLayers.Map({
        div: "map",
        theme: null,
        //projection: sm,
                        projection: gg,
                        //displayProjection: sm,
        units: "m",


                        // maxResolution: 156543.0339,
                         //maxExtent: new OpenLayers.Bounds(-5, 35, 15, 55);
```

```
            //              maxExtent: new OpenLayers.Bounds(-20037508, -20037508, 20037508, 20037508.34),
        numZoomLevels: 22,
        controls: [
            new OpenLayers.Control.Attribution(),
            new OpenLayers.Control.TouchNavigation({
                dragPanOptions: {
                    enableKinetic: true
                }
            }),
            geolocate,
            selectControl
        ],
        layers: [
         /*   new OpenLayers.Layer.OSM("OpenStreetMap", null, {
                transitionEffect: 'resize'
            }),
            new OpenLayers.Layer.Bing({
                key: apiKey,
                type: "Road",
                // custom metadata parameter to request the new map style - only useful
                // before May 1st, 2011
                metadataParams: {
                    mapVersion: "v1"
                },
                name: "Bing Road",
                transitionEffect: 'resize'
            }),*/
                                        wmsOverlay,
                                        lb,
                                        base,
            vector,
                                        cazuay,
                                        layer0

        ],
       // center: new OpenLayers.LonLat(-8830000,-350000),
                center: new OpenLayers.LonLat(-79.00443,-2.89751),

        zoom: 9
    });


            //transformacion
            //OpenLayers.Projection.addTransform("EPSG:4326",                                    "EPSG:900913",
OpenLayers.Layer.SphericalMercator.projectForward);
    //OpenLayers.Projection.addTransform("EPSG:900913",                                    "EPSG:4326",
OpenLayers.Layer.SphericalMercator.projectInverse);


    var style = {
        fillOpacity: 0.1,
        fillColor: '#000',
        strokeColor: '#f00',
        strokeOpacity: 0.6
    };
    geolocate.events.register("locationupdated", this, function(e) {
        vector.removeAllFeatures();
        vector.addFeatures([
            new OpenLayers.Feature.Vector(
                e.point,
                {},
                {
                    graphicName: 'cross',
                    strokeColor: '#f00',
                    strokeWidth: 2,
                    fillOpacity: 0,
                    pointRadius: 10
                }
            ),
            new OpenLayers.Feature.Vector(
```

```
            OpenLayers.Geometry.Polygon.createRegularPolygon(
                new OpenLayers.Geometry.Point(e.point.x, e.point.y),
                e.position.coords.accuracy / 2,
                50,
                0
            ),
            {},
            style
        )
    ]);
    map.zoomToExtent(vector.getDataExtent());
});

function agregar_capa(capa)
{
var tmp;
var tmp1;
tmp = server_seleccionado.split(",");
tmp1 = tmp[1].split("?");
var c1 = new OpenLayers.Layer.WMS(capa.value,
                                                    tmp1[0] + "?",
                                                    {'layers': capa.value, transparent: true},
                                                    {isBaseLayer: false,units: "dd", projection: sm}  );
map.addLayer(c1);
}

function gup( name ){
        var regexS = "[\\?&]"+name+"=([^&#]*)";
        var regex = new RegExp ( regexS );
        var tmpURL = window.location.href;
        var results = regex.exec( tmpURL );
        if( results == null )
                return"";
        else
                return results[1];
}
```

6.3. Anexo Código fuente index.html del visor GXM

```
<!DOCTYPE html>
<html>
  <head>
    <meta name="viewport" content="width=device-width, initial-scale=1.0, maximum-scale=1.0, user-scalable=0"/>
    <meta name="apple-mobile-web-app-capable" content="yes"/>
    <meta http-equiv="Content-Type" content="text/html; charset=utf-8">
    <title>MINIIDEA :: UDA</title>
                <script src="http://www.openlayers.org/api/2.11/OpenLayers.js"></script>
    <!--<script src="http://openlayers.org/dev/OpenLayers.mobile.js"></script>-->

    <link rel="stylesheet" href="./theme/default/style.mobile.css" type="text/css">
    <link rel="stylesheet" href="http://cdn.sencha.io/touch/1.1.0/resources/css/sencha-touch.css">
    <script src="http://cdn.sencha.io/touch/1.1.0/sencha-touch.js"></script>
    <script src="mobile-sencha.js"></script>
    <script src="mobile-base.js"></script>
                <script src="./jquery-1.5.min.js" type="text/javascript"></script>
                <script src="./jquery.mobile-1.0a3.min.js" type="text/javascript"></script>

    <style>
      .searchList {
        min-height: 150px;
      }

      .close-btn {
        position: absolute;
        right: 10px;
        top: 10px;
      }
```

```
    img.minus {
        -webkit-mask-image: url(img/minus1.png);
    }

    img.layers {
        -webkit-mask-image: url(img/list.png);
    }

    .gx-layer-item {
        margin-left: 10px;
    }

    #map {
        width: 100%;
        height: 100%;
    }

    .olControlAttribution {
        font-size: 10px;
        bottom: 5px;
        right: 5px;
    }

    #title, #tags, #shortdesc {
        display: none;
    }
</style>
<script>
                var server_seleccionado="";
                var format = new OpenLayers.Format.WMC({'layerOptions': {buffer: 0}});
                OpenLayers.IMAGE_RELOAD_ATTEMPTS = 2;
                var doc, context;
                var tmpu=gup('usuario');
                if (tmpu=='..' || tmpu=='')
                {alert("Recuerde que para almacenar sus mapas debe estar logueado.");}
var app = new Ext.Application({
    name: "ol",
    launch: function() {
        this.viewport = new Ext.Panel({
            fullscreen: true,
            dockedItems: [{
                dock: "top",
                xtype: "toolbar",
                ui: "light",
                layout: {
                    pack: "center"
                },
                items: [{
                                                                    iconMask: true,
                                                                        icon: './img/load.png',
                    handler: function() {
                                                                        mapas_server();
                    }
                },
                                                                {
                    iconMask: true,
                                                                        icon: './img/save.png',
                    handler: function() {
                                                                        leer_wmc();
                    }
                }, {
                    iconMask: true,
                                                                        icon: './img/capas.png',
                    //iconCls: "servers",
                    handler: function() {
                        capas_server();
                                                                                }
                }
                                                                , {
                    iconMask: true,
```

```
            iconCls: "layers",
            handler: function() {
                if (!app.popup) {
                    app.popup = new Ext.Panel({
                        floating: true,
                        modal: true,
                        centered: true,
                        hideOnMaskTap: true,
                        width: 240,
                        items: [{
                            xtype: 'app_layerlist',
                            map: map
                        }],
                        scroll: 'vertical'
                    });
                }
                app.popup.show('pop');
            }
        }, {
            xtype: "spacer"
        }, {
            iconMask: true,
            iconCls: "add",
            handler: function() {
                map.zoomIn();
            }
        }, {
            iconMask: true,
            iconCls: "minus",
            handler: function() {
                map.zoomOut();
            }
        } ]
    }],
    items: [
        {
            xtype: "component",
            scroll: false,
            monitorResize: true,
            id: "map",
            listeners: {
                render: function() {
                    var self = this;
                    init(function(feature) {
                        var htmlContent = "";
                        for (var property in feature.data) {
                            if (feature.data[property] != 'undefined') {
                                htmlContent = htmlContent + feature.data[property] + "<br>";
                            }
                        }
                        if (self.featurePopup) {
                            self.featurePopup.destroy();
                        }
                        self.featurePopup = new Ext.Panel({
                            floating: true,
                            modal: true,
                            centered: true,
                            hideOnMaskTap: true,
                            width: 240,
                            html: htmlContent,
                            scroll: 'vertical'
                        });
                        self.featurePopup.show();
                    })
                },
                resize: function() {
                    if (window.map) {
                        map.updateSize();
                    }
                },
```

```
                    scope: {
                        featurePopup: null
                    }
                }
            }
        ]
    });
}
});


                    function capas_server()
{
                    if (!app.popup2) {
    app.popup2 = new Ext.Panel({
    floating: true,
    modal: true,
    centered: true,
    hideOnMaskTap: true,
    width: 240,
                                        height: 240,
    items: [{
    html: '<img src="./img/cerrar.png" style="float:right;" onClick="app.popup2.hide();"><br><div style="font-size:12px;">Servidores:</div><br> <div class="x-panel-header" id="wms_servers" style="font-size:11px;"></div><br style="font-size:12px;"><div style="font-size:12px;">Capas</div><div class="x-panel-header" style="font-size:11px;" id="capas_server"></div>'
    }],
    scroll: 'vertical'
    });
}
                                callAjax();
    app.popup2.show('pop');

}

                    function mapas_server()
{
                    if (!app.popup1) {
    app.popup1 = new Ext.Panel({
    floating: true,
    modal: true,
    centered: true,
    hideOnMaskTap: true,
    width: 240,
                                        height: 240,
    items: [{
    html: '<img src="./img/cerrar.png" style="float:right;" onClick="app.popup1.hide();"><br>Mapas:<br> <div class="x-panel-header" id="mapas_usuarios" style="font-size:11px;"></div>'
    }],
    scroll: 'vertical'
    });
}
                                cargar_mapas();
    app.popup1.show('pop');

}


    function listar_capas(obj)
    {
    server_seleccionado=obj.value;
    $.ajax({
type: 'GET',
url: './ajax.php?id=113&criterio=' + obj.value,
success: function(result) {
    var s= document.getElementById("capas_server");
    s.innerHTML="";
  $('#capas_server').append(result);
}
});
```

```
            //var res = document.getElementById("capas_server");
            //res.innerHTML=obj.value;
            }

function callAjax() {
  $.ajax({
    type: 'GET',
    url: './ajax.php?id=112',
    success: function(result) {
                        var s= document.getElementById("wms_servers");
                        s.innerHTML="";
      $('#wms_servers').append(result);
    }
 });
 }

 function cargar_mapas() {
 var usuario;
 usuario=gup('usuario');
            var tmpurl='./ajax.php?id=115&criterio=' + usuario;
  $.ajax({
    type: 'GET',
    url: tmpurl,
    success: function(result) {
                        var s= document.getElementById("mapas_usuarios");
                        s.innerHTML="";
      $('#mapas_usuarios').append(result);
    }
 });
 }

function leer_wmc()
{
//genero wmc
OpenLayers.IMAGE_RELOAD_ATTEMPTS = 2;
      OpenLayers.Util.onImageLoadErrorColor = "transparent";

      var layerOptions = {
         isBaseLayer: false,
         singleTile: true,
         buffer: 0,
         ratio: 1
         };
//usuario
var usuario;
usuario=gup('usuario');


try {
            var text = format.write(map);
            document.getElementById("wmc").value = text;
         } catch(err) {
            document.getElementById("wmc").value = err;
         }

//mensaje grabar en la base de datos
            if (usuario==" || usuario==undefined)
            {alert("Debe estar logueado para guardar el mapa.");return;}
            else
            {
                        //titulo del mapa

                        //
                        var wmc= document.getElementById("wmc").value;
                        var answer = confirm ("Esta seguro que desea almacenar el mapa.");
                        if (answer)
                        {
                        var titulo= window.prompt("Ingrese titulo del mapa","Mapa");
                                    $.ajax({
```

```
                                type: 'GET',
                                contentType: "application/x-www-form-urlencoded;charset=ISO-8859-1",
                                url: './ajax.php?id=114&usuario=' + usuario + '&wmc=' + wmc +  "&titulo=" + titulo,
                                success: function(result) {
                                        document.getElementById("rwmc").innerHTML="";
                        $('#rwmc').append(result);
                        }});


                        }
                        else
                        {
                        document.getElementById("rwmc").innerHTML="";
                        $('#rwmc').append("Accion cancelada por el usuario.");}

                }
}

        function readWMC(merge) {
            var text = document.getElementById("wmc").value;

            if(merge) {
                try {
                    map = format.read(text, {map: map});
                } catch(err) {
                    document.getElementById("wmc").value = err;
                }
            } else {
                map.destroy();
                try {
                    var jsonFormat = new OpenLayers.Format.JSON();
                    var mapOptions = jsonFormat.read(OpenLayers.Util.getElement('mapOptions').value);
                    map = format.read(text, {map: mapOptions});
                    map.addControl(new OpenLayers.Control.LayerSwitcher());
                } catch(err) {
                    document.getElementById("wmc").value = err;
                }
            }
        }


        </script>
    </head>
    <body>
        <h1 id="title">MiniIDE UDA</h1>
        <div id="tags">
            mobile, sencha touch, WMS, UDA, AZUAY, CUENCA, ECUADOR
        </div>
        <p id="shortdesc">
            Using Sencha Touch to display an OpenLayers map.
        </p>
                        <div id="servers" style="position:absolute;top:0px;left:50px; zindex:50;"> servidores</div>
                        <div id="rwmc" style="position: absolute; z-index: 1004;top:50px;left:80px;font-size:12px;">
Accion:</div>
                        <textarea value='wmc' name='wmc' id='wmc' style="position: absolute; z-index: -4;top:50px;
visibility:hidden; "></textarea>
                        <input type="text" id="mapOptions" style='visibility:hidden;' value='{"div": "map", "allOverlays":
true}'/>
                        <!--<img src="./img/save.png" onClick="leer_wmc();" title='WMC' style="position: absolute; z-index:
1004;top:45px;">-->
                        <!--<input type='button' onClick="leer_wmc();" value='WMC' style="position: absolute; z-index:
1004;top:150px;">-->
    </body>
</html>
```

6.4 Anexo generación de códigos QR con librería Php.

```
<?php
$usuario=str_replace("@", " ", $_GET["usuario"]);
$usuario=str_replace("\\", "", $usuario);

header('Content-Type: text/html; charset=ISO-8859-1');

include('../libuda/cnn.php');
include('./tantaqrcode.php');
$conexion=$dbconn;

$sqlrec="select * from mini_marcadores where usuario='$usuario' order by
tipo";
$resultado = pg_Exec($conexion,$sqlrec);
$filas=pg_NumRows($resultado);
$xml="<h2>Codigos QR de marcadores </h2><hr><br>";
if ($filas==0)
{echo "No se puede generar codigos QR. Verifique la existencia de
marcadores.";exit;}
else
   {
      while($fila = pg_fetch_array($resultado, null, PGSQL_ASSOC)) {
       $qr = new tantaQRCode();
       $qr->url($fila['link']);
       $code =
sha1(mt_rand().time().mt_rand().$_SERVER['REMOTE_ADDR']);
       $nombre="$usuario$code ";
       $filename = $qr->draw( 150, $nombre, 'png' );
       $xml .= "<hr><b>Marcador: </b>" . $fila['tipo'] . "<hr> Título
: " .  $fila['descripcion'] . "<br>Creado: " . $fila['creado']. "<br>Url: <a
href="' . $fila['link'] . "' target='_blank'>" . $fila['link'] . "</a>";
       $xml .= "<br><img src='./qrs/$filename.png'>";
       $xml.= "<a href='http://www.facebook.com/sharer.php?
u=http://gis.uazuay.edu.ec/ide/visor/GeoExt/examples/qr/qrs/$filename.png&t="
. $fila['descripcion'] . "IDE Universidad del Azuay
(http://gis.uazuay.edu.ec)' target='_blank'><img src='./fb.png'></a> " ;
       }
       echo $xml;
   }
       exit;

?>
```