

UNIVERSIDAD AUTÓNOMA METROPOLITANA

UNIDAD AZCAPOTZALCO

DIVISIÓN DE CIENCIAS BÁSICAS E INGENIERÍA

---

**Aplicación para el Sistema Operativo  
iOS para descubrir la UAM unidad  
Azcapotzalco**

---

REPORTE DE PROYECTO TERMINAL

*Presenta*

Elias Navarro Juan José  
Sosa Sánchez Marco Antonio

*Asesor:*

M. en C. Arturo Zúñiga López  
Profesor Asociado

DICIEMBRE 2012



# Resumen

En los últimos años se ha producido un crecimiento exponencial en el uso los smartphones (teléfono móvil con una mayor capacidad de capacidad de computo y conectividad que un teléfono convencional), principal motivo por el que las aplicaciones móviles se ha convertido en un mercado emergente con grandes posibilidades. El presente trabajo expone el desarrollo de una aplicación para un dispositivo móvil, el cual tiene como principal objetivo solucionar el problema de ser ajeno a la UAM unidad Azcapotzalco, es decir, no tener conocimiento de la ubicación e información de los edificios con los que cuenta la unidad.



# Índice general

<b>1. Introducción</b>	<b>1</b>
1.1. Antecedentes y Estado de Arte . . . . .	2
1.1.1. Referencias Internas . . . . .	2
1.1.2. Referencias Externas . . . . .	2
1.2. Justificación . . . . .	2
1.3. Objetivos . . . . .	3
1.4. Estructura del Reporte . . . . .	4
<b>2. Fundamentos Teóricos</b>	<b>5</b>
2.1. Sistema Operativo iOS . . . . .	5
2.1.1. Xcode . . . . .	6
2.2. Google Maps . . . . .	7
2.3. SQLite . . . . .	9
<b>3. Desarrollo</b>	<b>11</b>
3.1. Investigación Previa . . . . .	11
3.1.1. Framework . . . . .	11
3.1.2. Tipos de Archivos en iOS . . . . .	12
3.1.3. Tipos de Proyectos . . . . .	12
3.2. Recursos . . . . .	13
3.3. Módulos de Programación . . . . .	14
3.3.1. Obtener ubicación . . . . .	15
3.3.2. Búsqueda en Mapa . . . . .	18
3.3.3. Perspectiva del dispositivo . . . . .	20
3.3.4. Cámara . . . . .	22
3.3.5. Información de Edificios . . . . .	23
<b>4. Pruebas y Resultados</b>	<b>25</b>
4.1. Principales Opciones . . . . .	25

4.2. Pruebas . . . . .	25
4.2.1. Mapa . . . . .	25
4.2.2. Cámara . . . . .	30
<b>5. Conclusión</b>	<b>31</b>
<b>Bibliografía</b>	<b>33</b>
<b>A. Manual de Usuario</b>	<b>35</b>
A.0.3. Acerca de . . . . .	35
A.0.4. Mapa . . . . .	35
A.0.5. Mapa Desplazable . . . . .	36
A.0.6. Búsqueda de Edificios . . . . .	36
A.0.7. Edificios . . . . .	36
A.0.8. Ubicación . . . . .	37
A.0.9. Como llegar . . . . .	37
A.0.10. Cámara . . . . .	39

# Índice de figuras

2.1. Capas de Abstracción en iOS . . . . .	6
2.2. Xcode versión 4.2 . . . . .	7
2.3. Google Maps en iOS . . . . .	9
2.4. Google Maps en iOS . . . . .	10
3.1. Frameworks nativos en iOS 5 . . . . .	11
3.2. Módulos de programación de nuestra aplicación. . . . .	15
3.3. Librerías para el uso de mapa y localización. . . . .	15
3.4. Agregando un Map View en la interfaz . . . . .	16
3.5. Variables de instancia para ubicación y mapa. . . . .	17
3.6. Código ubicar el mapa en la UAM Azcapotzalco. . . . .	17
3.7. Código para obtener la localización del usuario. . . . .	18
3.8. Agregando barra de búsqueda en interfaz . . . . .	19
3.9. Variable de instancia de la barra de búsqueda . . . . .	19
3.10. Código para obtener la búsqueda deseada . . . . .	19
3.11. Código para realizar la búsqueda en la base de datos . . . . .	20
3.12. Código para obtener la perspectiva o heading en iphone. . . . .	20
3.13. Algoritmo para mostrar botones de los edificios basado en la perspectiva. . . . .	21
3.14. Código para obtener la imagen de la cámara. . . . .	22
3.15. Librería para el uso de la base de datos. . . . .	23
3.16. Variable de Instancia de la base de datos. . . . .	23
3.17. Inicialización de la Base de datos. . . . .	23
4.1. Prueba del botón de <i>Edificios</i> . . . . .	26
4.2. Posición para la prueba del botón <i>Ubicación</i> . . . . .	27
4.3. Prueba del botón de <i>Ubicación</i> en dos dispositivos. . . . .	28
4.4. Edificios tomados para ejemplo del botón <i>Como Llegar</i> . . . . .	28
4.5. Prueba del botón <i>Como Llegar</i> . . . . .	29

4.6. Pruebas de <i>Búsqueda de Edificios</i> . . . . .	29
4.7. Posición para la prueba de la opción de <i>Cámara</i> . . . . .	30
4.8. Pruebas al presionar el botón de <i>Mostrar Edificios</i> . . . . .	30
A.1. Pantalla inicial de la aplicación. . . . .	35
A.2. Vistas del mapa . . . . .	37
A.3. Búsqueda de Edificios . . . . .	38
A.4. Marcadores e información de Edificios . . . . .	39
A.5. Ubicación del Usuario . . . . .	40
A.6. Ejemplo de opción <i>como llegar</i> . . . . .	41
A.7. Pruebas de la opción cámara realizadas en un iPhone 4G . . . . .	41
A.8. Pruebas de la opción cámara realizadas en un iPod 4G . . . . .	42



# Capítulo 1

## Introducción

Una aplicación es un programa informático diseñado como herramienta para permitir a un usuario realizar uno o diversos tipos de trabajos. Esto lo diferencia principalmente de otros tipos de programas como los sistemas operativos (que hacen funcionar a la computadora), las utilidades (que realizan tareas de mantenimiento o de uso general) y los lenguajes de programación<sup>1</sup>. En los últimos años se ha producido un crecimiento exponencial en el uso de los smartphones<sup>2</sup>, motivo por el cual, el desarrollo de las aplicaciones móviles, se ha convertido en un mercado emergente con grandes posibilidades.

En este reporte se presenta un aplicación mediante la cual podremos descubrir la UAM unidad Azcapotzalco, es decir, con el uso de esta aplicación podremos conocer la ubicación e información de cada edificio en la unidad así como marcar rutas de como llegar de un punto a otro. Esto a través de un dispositivo con sistema operativo iOS<sup>3</sup> (con versión 5 o superior). Esta aplicación se basa en la geolocalización, técnica a través de la cual hace referencia al posicionamiento de un objeto espacial (representado por un punto, vector, área y volumen) en un sistema de coordenadas. Una vez obtenida la posición actual del usuario (obtenida mediante el dispositivo con sistema operativo iOS: iPod<sup>4</sup> o iPhone<sup>5</sup>) podremos saber cuales son los edificios más cercanos y así poder ofrecer rutas de como llegar, e información relevante de cada edificio.

A continuación se proporciona una breve revisión del estado del arte y la justi-

---

<sup>1</sup>Idioma artificial con el cual se crean los programas informaticos

<sup>2</sup>Teléfono móvil con una mayor capacidad de computo y conectividad.

<sup>3</sup>Sistema operativo móvil de Apple.

<sup>4</sup>Reproductor de audio digital diseñado por Apple.

<sup>5</sup>Teléfono inteligente diseñado por Apple.

ficación de este trabajo.

## **1.1. Antecedentes y Estado de Arte**

Algunas de las aplicaciones móviles similares al objetivo de esta tesis son:

### **1.1.1. Referencias Internas**

Proyectos similares realizados dentro de la UAM Azcapotzalco:

- “Aplicación de realidad aumentada: Marcador de señales de tránsito”, él cual fue propuesto en el trimestre 09-0, y como su nombre lo indica es una aplicación la cual usa realidad aumentada para poder reconocer señales de tránsito.
- “Sistema móvil de reconocimiento de objetos y posicionamiento en una exposición basado en realidad aumentada”, él cual fue propuesto en el trimestre 11-P, como su nombre lo indica usa la realidad aumentada para el reconocimiento de objetos, específicamente para exposiciones de arte en un museo.

### **1.1.2. Referencias Externas**

Algunas de las aplicaciones más conocidas son:

- UNAM 360: Aplicación que permite descubrir el campus de Ciudad Universitaria por medio del GPS y realidad aumentada.
- CheckIn +: Aplicación con realidad aumentada por medio de la cual puedes descubrir los puntos más destacados que te rodean.
- Tom Tom: Aplicación de navegación, la cual ofrece diferentes opciones de ruta, e información detallada de los lugares más conocidos.

## **1.2. Justificación**

Uno de los problemas al entrar por primera vez a un lugar es desconocer la ubicación e información relevante de dicho lugar, es por esto que se ha llevado a cabo la tarea de crear aplicaciones las cuales faciliten esta tarea a todo tipo de usuarios. Hoy en día algunas de estas aplicaciones resuelven este problema mediante el uso

de la realidad aumentada<sup>6</sup> y otras mediante el uso de la geolocalización<sup>7</sup>.

Pensando en esta necesidad y tomando en cuenta el gran uso de los smartphones, hemos decidido desarrollar una aplicación la cual permita al usuario descubrir de manera fácil y rápida el campus de la UAM Azcapotzalco.

### 1.3. Objetivos

El objetivo general de este proyecto es:

- Diseñar e implementar una aplicación para el sistema operativo iOS, la cual permitirá a un usuario mostrar el campus de la UAM Azcapotzalco así como la ubicación e información de lugares.

Para ello se plantearon los siguientes objetivos específicos:

1. Diseñar e implementar el módulo de mapas, el cual consiste en delimitar y marcar los edificios de la UAM-Azc con ayuda de la API<sup>8</sup> de Google Maps<sup>9</sup>.
2. Diseñar e implementar la base de datos del sistema (almacenada en la memoria del dispositivo) por medio de la herramienta SQLite<sup>10</sup>, la cuál contendrá toda la información acerca de los edificios de la unidad.
3. Diseñar e implementar un módulo para acceder a la cámara del dispositivo y re-dibujar etiquetas.
4. Diseñar e implementar un modulo para poder obtener la ubicación del dispositivo en base a su conexión (Wi-Fi<sup>11</sup> o GPS<sup>12</sup>)
5. Diseñar e implementar un modulo para poder saber la perspectiva del dispositivo en base al giroscopio<sup>13</sup>.

---

<sup>6</sup>Visión directa o indirecta de un entorno físico del mundo real, cuyos elementos se combinan con elementos virtuales para la creación de una realidad mixta en tiempo real.

<sup>7</sup>Mostrar información relevante en base a la posición actual del usuario

<sup>8</sup>Interfaz de programación de aplicaciones

<sup>9</sup>Servicio de Mapas en la web

<sup>10</sup>Sistema de Gestión de Base de Datos Relacional

<sup>11</sup>Mecanismo de conexión de dispositivos electrónicos de forma inalámbrica.

<sup>12</sup>Sistema de Posicionamiento Global.

<sup>13</sup>Dispositivo mecánico que gira alrededor de su eje de simetría.

## 1.4. Estructura del Reporte

El presente trabajo está estructurado en una serie de capítulos, cuyo contenido se describe a continuación:

- En el Capítulo 2 se habla con detalle de los fundamentos teóricos necesarios para el desarrollo de la aplicación.
- En el capítulo 3 se presentan las metodologías diseñadas para desarrollar la aplicación propuesta.
- En el capítulo 4 se muestran los resultados obtenidos al finalizar la implementación de todos los módulos de programación.
- Finalmente en el capítulo 5 se presentan las conclusiones del trabajo y los trabajos a futuro.

## Capítulo 2

# Fundamentos Teóricos

En este capítulo se brinda una pequeña introducción y un panorama de los principales conocimientos teóricos a usar en este proyecto (sistema operativo iOS y las librerías de Google Maps, SQLite).

### 2.1. Sistema Operativo iOS

Sistema operativo desarrollado por Apple que originalmente fue desarrollado en el año 2007 para la primera versión de su teléfono móvil “*Iphone*”, dicho sistema operativo esta basado en MAC OS X<sup>1</sup>(sistema basado en un sistema unix<sup>2</sup>), programado en C<sup>3</sup>, Objective-C<sup>4</sup> y C++<sup>5</sup>.

iOS cuenta con cuatro capas de abstracción<sup>6</sup> (figura 2.1):

1. **Capa del núcleo del sistema operativo** - proporciona características de bajo nivel así como un *framework* para la seguridad y la interacción con el *hardware* externo.
2. **Capa de Servicios Principales (Core Services)** - capa que provee los servicios requeridos por las capas superiores.

---

<sup>1</sup>Sistema operativo de Apple usado en su gama de computadoras.

<sup>2</sup>Sistemas operativo desarrollado en 1969.

<sup>3</sup>Lenguaje de programación creado en 1972 por Dennis M. Ritchie.

<sup>4</sup>Lenguaje de programación orientado a objetos creado como un superconjunto de C.

<sup>5</sup>Lenguaje de programación diseñado a mediados de los años 1980 por Bjarne Stroustrup

<sup>6</sup>Capa o nivel de abstracción, es una forma de ocultar los detalles de implementación de ciertas funcionalidades.

3. **Capa de Medios (Media Layer)** - proporciona las tecnologías necesarias para gráficos, audio y video.
4. **Capa “Cocoa Touch”** - lugar donde se encuentran los *frameworks*<sup>7</sup>, que usualmente son usados cuando se crea una aplicación.

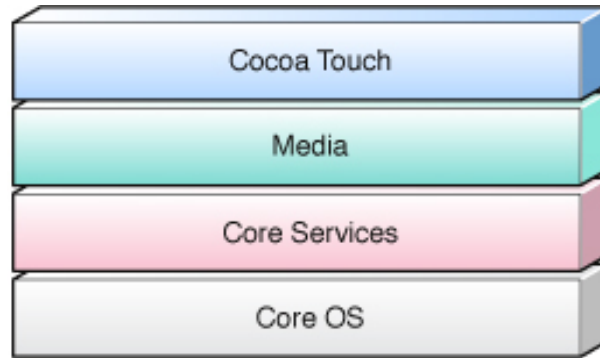


Figura 2.1: Capas de Abstracción en iOS

### 2.1.1. Xcode

Para desarrollar aplicaciones en iOS se usa el SDK <sup>8</sup>(*Software Development Kit*) de Apple, “Xcode” (figura 2.2), el cual incluye herramientas e interfaces para desarrollar, instalar y probar aplicaciones. Xcode incluye la colección de compiladores<sup>9</sup> del proyecto GCC<sup>10</sup> y puede compilar código C, C++, Objective-C, Java<sup>11</sup>, AppleScript mediante una amplia gama de modelos de programación. Una de las principales características de Xcode es la tecnología para distribuir el proceso de construcción a partir de código fuente en varias computadoras “*Bonjour*”.

<sup>7</sup>Conjunto de librerías para agregar un función específica a un programa.

<sup>8</sup>Conjunto de herramientas de Desarrollo de software que le permite al programador crear aplicaciones.

<sup>9</sup>Programa que traduce un lenguaje de programación a otro lenguaje de programación para que sea interpretado por una maquina.

<sup>10</sup>Conjunto de compiladores creados pro el proyecto GNU.

<sup>11</sup>Lenguaje de programación de alto nivel orientado a objetos.

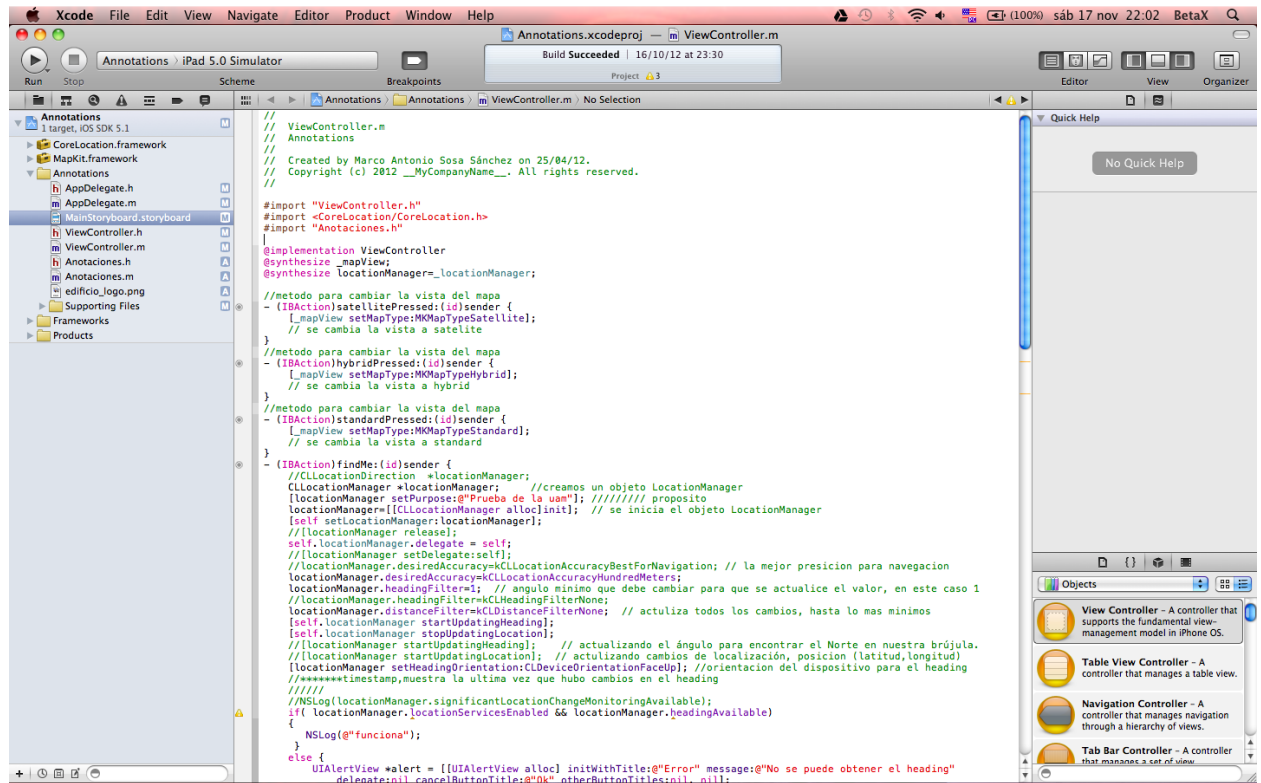


Figura 2.2: Xcode versión 4.2

## 2.2. Google Maps

Google Maps es un servicio de Google Inc. que ofrece tecnología cartográfica potente y fácil de usar. En el año 2008 Google Maps presentó su primera aplicación para dispositivos móviles, destinada para funcionar en cualquier dispositivo con Java. Algunas de los principales servicios de Google Maps son:

- Resultados de lugares integrados en la búsqueda ( Información de empresas así como de su contacto en un mismo sitio, todo ello integrado en el mapa).
- Imágenes por satélite (Imágenes obtenidas por satélite).
- Vista de Calle (Fotografías panorámicas a nivel de calle)
- Rutas detalladas (Opción capaz de señalar una ruta entre dos o más ubicaciones).

- Mapas que se pueden arrastrar (Mapas a disposición del usuario con la opción de moverse a la ubicación que se desee).
- Servicio de Localización (Servicio capaz de ubicar la posición del usuario ya sea por GPS o mediante la triangulación de antenas transmisoras de telefonía celular.)

Actualmente este servicio se encuentra disponible para las siguientes plataformas<sup>12</sup>:

- Android<sup>13</sup>
- iOS(figura 2.3)
- Windows Mobile<sup>14</sup>
- Symbian OS<sup>15</sup>
- Palm OS<sup>16</sup>
- BlackBerry<sup>17</sup>

---

<sup>12</sup>Referente al sistema operativo o a sistemas complejos que a su vez sirven para crear programas

<sup>13</sup>Sistema operativo basado en Linux para dispositivos móviles.

<sup>14</sup>Sistema operativo para telefonos diseñado por la compañía de Microsoft.

<sup>15</sup>Sistema operativo para telefonos diseñado por Accenture.

<sup>16</sup>Sistema operativo móvil desarrollado inicialmente por Palm Inc.

<sup>17</sup>Marca de dispositivos móviles y servicios desarrollados por Research In Motion (RIM).





Figura 2.3: Google Maps en iOS

### 2.3. SQLite

SQLite (figura2.4) es un sistema de gestión de base de datos relacional que implementa un motor de base de datos SQL autónomo, sin servidor, configuración rápida y transaccional, todo esto contenido en una pequeña biblioteca de C. A diferencia de los sistemas de gestión de bases de datos cliente-servidor, el motor de SQLite no es un proceso independiente con el que el programa principal se comunica. En lugar de eso, la biblioteca SQLite se enlaza con el programa pasando a ser parte integral del mismo. El programa utiliza la funcionalidad de SQLite a través de llamadas simples a subrutinas y funciones.



Figura 2.4: Google Maps en iOS

Esto reduce la latencia<sup>18</sup> en el acceso a la base de datos, debido a que las llamadas a funciones son más eficientes que la comunicación entre procesos. El conjunto de la base de datos (definiciones, tablas, índices, y los propios datos), son guardados como un sólo archivo estándar en la máquina *host*<sup>19</sup>. Este diseño simple se logra bloqueando todo el archivo de base de datos al principio de cada transacción.

---

<sup>18</sup>Suma de retardos producidos en una red.

<sup>19</sup>Computadora que funciona como el punto de inicio y final de las transferencias de datos.

# Capítulo 3

## Desarrollo

En este capítulo se muestran los principales pasos a seguir para el desarrollo de esta aplicación.

### 3.1. Investigación Previa

#### 3.1.1. Framework

Un *framework* es un conjunto de librerías que nos permiten añadir una funcionalidad concreta a una aplicación. Por defecto cuando creamos un proyecto tenemos los *framework's* esenciales (figura 3.1) para el funcionamiento básico.

<b>Audio and Video</b> Core Audio OpenAL Media Library AV Foundation	<b>Graphics and Animation</b> Core Animation OpenGL ES Quartz 2D	<b>User Applications</b> Address Book Core Location Map Kit Store Kit
<b>Data Management</b> Core Data SQLite	<b>Networking and Internet</b> Bonjour WebKit BSD Sockets	

Figura 3.1: Frameworks nativos en iOS 5

### 3.1.2. Tipos de Archivos en iOS

Algunos de los principales tipos de archivos a usar son:

- **info.plist** - XML<sup>1</sup> constituido de las propiedades de configuración (en UTF-8<sup>2</sup>)
- **Archivos.h** - Archivo el cual contiene las declaraciones de las interfaces, clases y atributos.
- **Archivos.m** - Este tipo de archivos contiene la implementación de las clases y métodos definidos en los *archivos.h*, no necesariamente requieren un archivo de este tipo.
- **Archivos.storyboard** - Archivo en donde se diseña toda la parte visual de la aplicación.
- **Archivos.xib** - Este tipo de archivos son los que se utilizan para crear la interfaz de usuario mediante la *interface builder* que nos provee Xcode.

### 3.1.3. Tipos de Proyectos

- **Navigation-based** - Proyecto basado en páginas de aplicación (múltiples vistas), implementa un controlador de vistas (*page view controller*).
- **Open GL** - Proyecto para crear juegos a través del framework de OpenGL.
- **Split View-based(Master-Detail)** - Proporciona una interfaz de usuario con un controlador de navegación para mostrar una lista de elementos con la capacidad de dividir la vista (solo en el iPad<sup>3</sup>).
- **Tab Bar (Tabbed)** - Presentan un menú conformado por una barra de pestañas y controladores de vista de diversos elementos.
- **Utility** - Aplicaciones que presentan una vista principal y permiten que el usuario acceda a otra vista para realizar personalizaciones básicas.
- **View-based(Single View)** - Aplicación con una vista simple para implementar la interfaz.
- **Window-based(Empty)** - Aplicación que sólo incluye un delegado de aplicación y una ventana.

---

<sup>1</sup>Lenguaje de marcas desarrollado por World Wide Web Consortium(W3C).

<sup>2</sup>Formato de codificación de caracteres creado por Robert C. Pike y Kenneth L. Thompson.

<sup>3</sup>Dispositivo electrónico tipo tableta desarrollado por Apple Inc.

## 3.2. Recursos

Para poder llevar a cabo el desarrollo de esta aplicación se utilizarán los siguientes dispositivos:

1. Una computadora portátil:

**Nombre del Modelo** MacBook Pro

**Procesador** Intel Core Duo

**Memoria** 2 GB DDR2 667 MHZ

**Disco Duro** 250 GB

**Sistema operativo** Mac OS X 10.6.8 (Snow Leopard)

2. Una computadora de escritorio:

**Nombre del Modelo** iMac

**Procesador** Intel Quad-Core

**Memoria** 8 GB DDR3 1600 MHZ

**Disco Duro** 2.25 TB

**Sistema operativo** Mac OS X 10.8 (Mount Lion)

3. Un telefono inteligente (*smartphone*):

**Nombre del Dispositivo** iPhone 4G

**Conectividad** GSM/EDGE, Wi-Fi (802.11n 2.4GHz), Bluetooth 2.1

**Sensores** Acelerómetro, Sensor de luz ambiental, Giroscopio

**Ubicación** Brújula Digital, GPS, Wi-Fi, Celular

**Capacidad** 16GB

**Sistema operativo** iOS 5.1

4. Reproductor portátil de audio digital:

**Nombre del Dispositivo** iPod 4G

**Conectividad** Wi-Fi (802.11n 2.4GHz), Bluetooth 2.1

**Sensores** Acelerómetro, Sensor de luz ambiental, Giroscopio

**Capacidad** 32GB

**Sistema operativo** iOS 6.0

### 3.3. Módulos de Programación

Para el desarrollo de esta aplicación se planeó el diseño de cuatro principales módulos de programación (figura 3.2):

#### **Obtener ubicación**

Módulo capaz de obtener la ubicación del usuario (del dispositivo) en base a la combinación de datos móviles Wi-Fi y GPS.

#### **Búsqueda en Mapa**

Módulo capaz de realizar búsquedas de los edificios de la unidad en un mapa.

#### **Perspectiva del dispositivo**

Módulo capaz de obtener la perspectiva<sup>4</sup> del usuario (del dispositivo).

#### **Cámara**

Módulo capaz de obtener la imagen de la cámara y sobreponer botones de los edificios observados, en base a la posición y perspectiva del usuario (del dispositivo).

#### **Información de Edificios**

Módulo en el cual estará contenida toda la información de cada edificio de la unidad.

Estos módulos corresponden a los objetivos particulares del proyecto.

---

<sup>4</sup>Visión, que viene favorecida por observación ya sea distante, parcial o temporal de cualquier hecho o fenómeno.

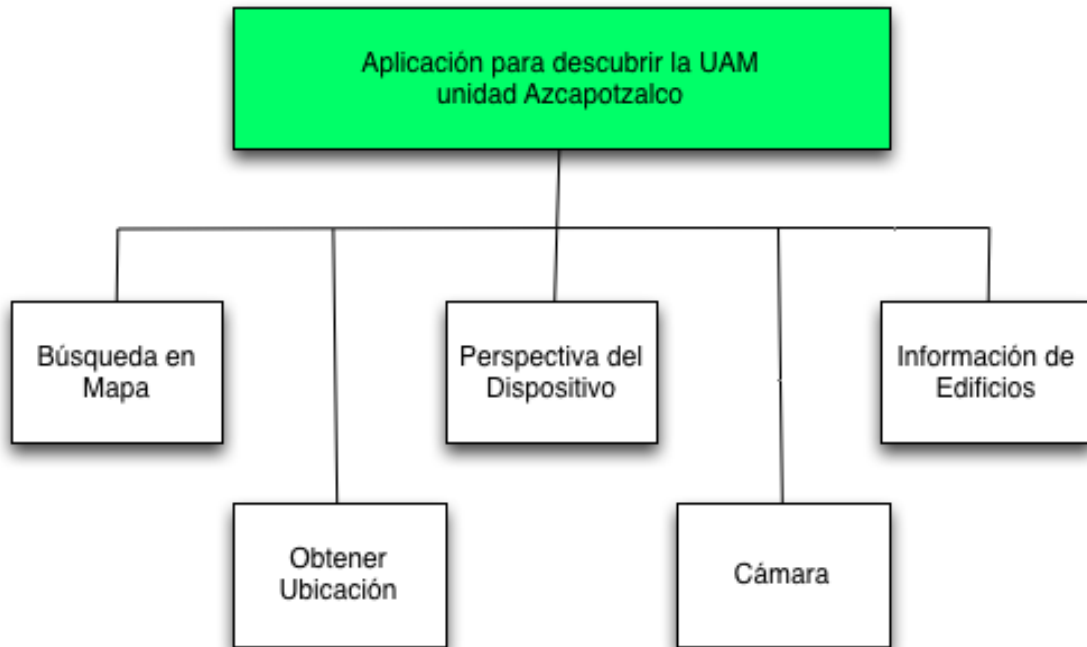


Figura 3.2: Módulos de programación de nuestra aplicación.

### 3.3.1. Obtener ubicación

Para el desarrollo de este módulo añadimos los siguientes *framework's*: **Core-Location** (Permite determinar la ubicación actual o partida asociada del dispositivo, esto a través del hardware disponible, GPS o Wi-Fi) y **MapKit** (Proporciona una interfaz la cual incorpora mapas, Google Maps, directamente en ventanas o vistas). Después importamos las librerías en todos los archivos que utilicen estas clases (especialmente en los archivos **.h**) como se muestra en la figura 3.3.

```
#import <MapKit/MapKit.h>
#import <CoreLocation/CoreLocation.h>
```

Figura 3.3: Librerías para el uso de mapa y localización.

Ahora en nuestro *Main.storyboard* (no necesariamente el principal) arrastramos un *Map View* (vista de mapa) a la interfaz de nuestra aplicación como se ve en la figura 3.4.

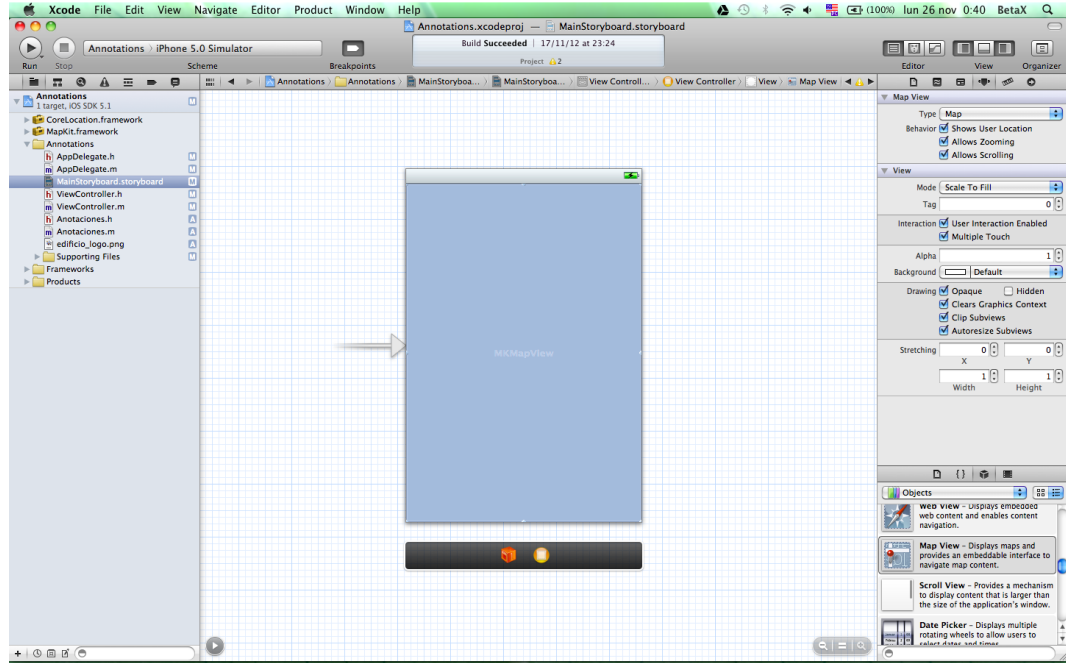


Figura 3.4: Agregando un Map View en la interfaz .



Ahora declaramos las variables de instancia como se muestra en la figura 3.5.

```
@property(weak, nonatomic) IBOutlet MKMapView *viewMap;  
@property(strong, nonatomic) CLLocationManager *locationManger;
```

Figura 3.5: Variables de instancia para ubicación y mapa.

Por ultimo agregamos los métodos para obtener la *ubicación* y delimitar el mapa inicialmente en la UAM Azcapotzalco (figuras 3.6 y 3.7).

```
//creamos un objeto de tipo localizacion  
CLLocationCoordinate2D zoomLocation;  
//se establece latitud del mapa deseado  
zoomLocation.latitude = 19.503553;  
//se establece longitud del mapa  
zoomLocation.longitude= -99.186679;  
//se ajusta al mapa  
MKCoordinateRegion viewRegion = MKCoordinateRegionMakeWithDistance  
(zoomLocation, 0.35*METERS_PER_MILE, 0.35*METERS_PER_MILE);  
MKCoordinateRegion adjustedRegion = [viewMap regionThatFits:viewRegion];  
//se establece el tipo de mapa  
[viewMap setMapType:MKMapTypeSatellite];  
//se muestra la region establecida  
[viewMap setRegion:adjustedRegion animated:YES];
```

Figura 3.6: Código ubicar el mapa en la UAM Azcapotzalco.

```
//se inicia el objeto locationManager  
locationManager=[[CLLocationManager alloc]init]  
//se establece el delegado  
self.locationManager.delegate = self;  
// filtro para notificar todos los cambios, hasta lo mas minimos  
locationManager.distanceFilter=kCLLocationDistanceFilterNone  
//empezamos a obtener la ubicacion  
[self.locationManager startUpdatingHeading];  
//se crea una variable de tipo localizacion  
CLLocationCoordinate2D userLocation;  
//se obtiene la latitud y longitud del dispositivo  
userLocation.latitude=locationManager.location.coordinate.latitude;  
userLocation.longitude=locationManager.location.coordinate.longitude;  
//agregamos una etiqueta a la ubicacion  
viewMap.userLocation.title=@"Tu estas aqui";  
//se muestra la ubicacion en el mapa  
viewMap.showUserLocation = TRUE;
```

Figura 3.7: Código para obtener la localización del usuario.

### 3.3.2. Búsqueda en Mapa

Para desarrollar esta opción primero agregamos la barra de búsqueda en nuestra interfaz como se muestra en la figura 3.8, para esto solo tenemos que arrastrar el objeto de la barra de búsqueda de la barra de herramientas a nuestra interfaz.

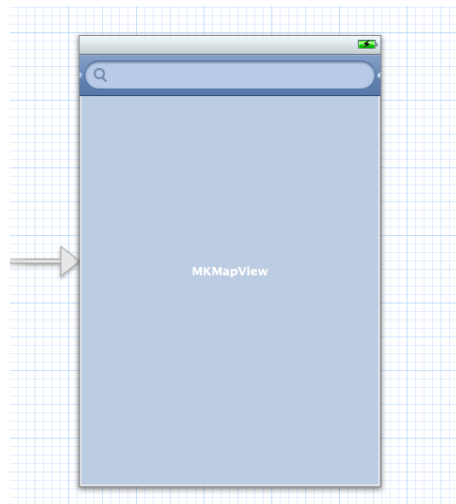


Figura 3.8: Agregando barra de búsqueda en interfaz

Después declaramos la variable de instancia como se muestra en la figura 3.9.

```
@property(strong, nonatomic) IBOutlet UISearchBar *searchBar;
```

Figura 3.9: Variable de instancia de la barra de búsqueda

Una vez realizado esto, obtenemos el texto de búsqueda ingresado por el usuario (figura 3.10).

```
//obtnemos el texto de la busqueda en su forma normal y en mayusculas  
NSArray *res = [util getInfoEdif:searchBar.text.uppercaseString info:searchBar.text];
```

Figura 3.10: Código para obtener la búsqueda deseada

Ahora agregamos el código para realizar la búsqueda (figura 3.11) en la base de datos de la aplicación, la cual se mostrará posteriormente.

```

//creamos un query de busqueda
NSString *query = [NSString stringWithFormat:@"select *
from Edificio WHERE Nombre=' %@'",edif];
//guardamos el resultado en un arreglo
NSArray *res = [_conexion execSelect:query];
//si el arreglo es mayor a cero regresamos que si hay un resultado
if ([res count]>0)
return res;
//si el contenido es corto, se toma como nombre de edificio
if (edif.length <3)
return NULL;
query = [NSString stringWithFormat: @"select * from
Edificio WHERE Descripcion like ' % % % @ % % '",inf];
//regresamos el resultado
return [_conexion execSelect:query];

```

Figura 3.11: Código para realizar la búsqueda en la base de datos

### 3.3.3. Perspectiva del dispositivo

Para obtener la perspectiva del dispositivo (figura 3.12), en el caso del iPhone, usaremos el mismo objeto que creamos para obtener la ubicación como se mostró en la figura 3.7.

```

// angulo minimo que debe cambiar para que se actualice el valor
locationManager.headingFilter=15;
//orientacion del dispositivo para el heading
[locationManager setHeadingOrientation:CLDeviceOrientationFaceUp];
//se comienza a registrar los cambios en el heading
[self.locationManager startUpdatingHeading];

```

Figura 3.12: Código para obtener la perspectiva o heading en iPhone.

Una vez que obtenemos la perspectiva (heading) del dispositivo usamos el siguiente algoritmo (figura3.13) para mostrar los edificios a través de la cámara.

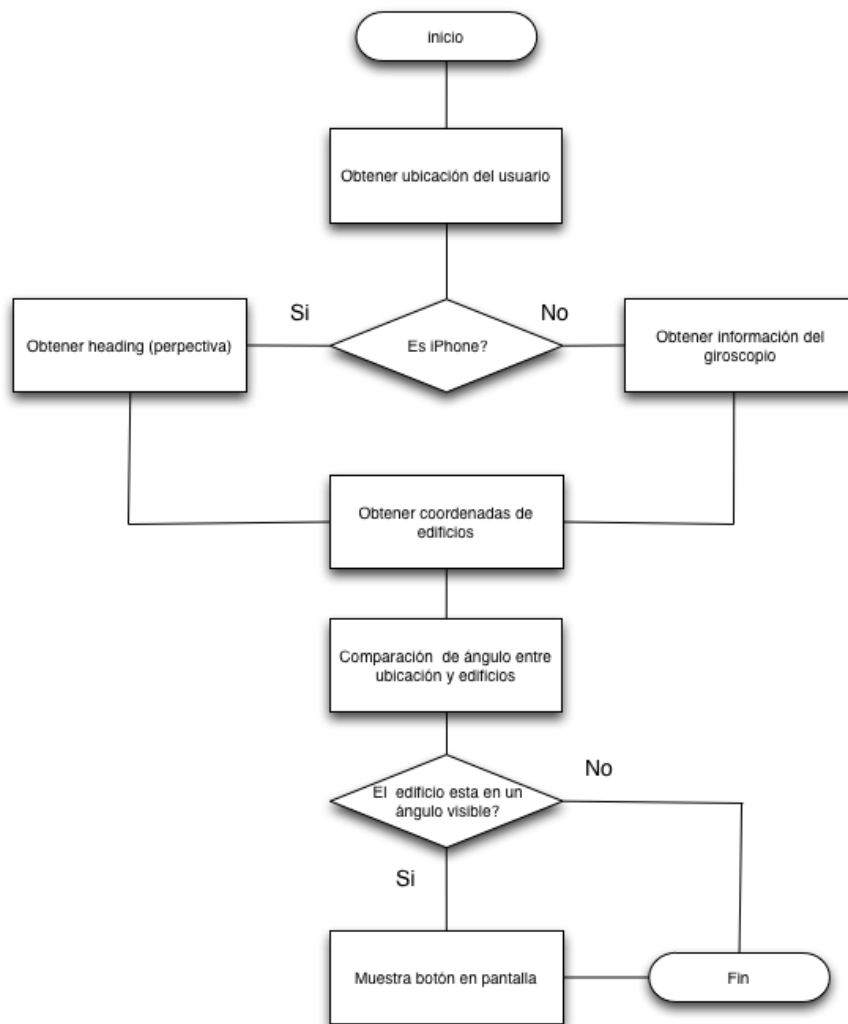


Figura 3.13: Algoritmo para mostrar botones de los edificios basado en la perspectiva.

### 3.3.4. Cámara

En la figura 3.14 se muestra el código que se usó para obtener la imagen de la cámara.

```
//creamos un objeto AVCaptureSession
self.session= [[AVCaptureSession alloc]init];
//definimos la resolucio de captura
self.session.sessionPreset =AVCaptureSessionPreset352x288;
//tipo de grabacion
self.videoDevice=[AVCaptureDevice defaultDeviceWithMediaType:
AVMediaTypeVideo];
//delegado
self.videoInput=[AVCaptureDeviceInput deviceInputWithDevice:
self.videoDevice error:nil];
//iniciamos la captura de video
self.frameOutput=[[AVCaptureVideoDataOutput alloc]init];
self.frameOutput.videoSettings= [NSDictionary dictionaryWithObject:
[NSNumber numberWithInt:kCVPixelFormatType_32BGRA] forKey:
(id)kCVPixelBufferPixelFormatTypeKey];
//mostramos la imagen obtenida
[self.session addInput:self.videoInput];
[self.session addOutput:self.frameOutput];
[self.frameOutput setSampleBufferDelegate:self queue:dispatch_get_main_queue()];
[self.session startRunning];
```

Figura 3.14: Código para obtener la imagen de la cámara.

### 3.3.5. Información de Edificios

Para el desarrollo de este módulo se agrego el framework *SQLite* (él cual permite gestionar una base de datos en el dispositivo). Después importamos las librerías en todos los archivos que utilicen estas clase (figura 3.15).

```
# import <sqlite3.h>
```

Figura 3.15: Librería para el uso de la base de datos.

Después declaramos la variable de instancia (figura 3.16).

```
@property(strong, nonatomic) IBOutlet NSString *BDName;
```

Figura 3.16: Variable de Instancia de la base de datos.

Para finalizar creamos la base de datos (figura 3.17) la cual contendrá la información de los edificios (nombre, ubicación, imágenes y contenido).

```
[self execQuery:@"CREATE TABLE IF NOT EXISTS Edificio (id int not null, Nombre varchar not null, Descripcion varchar, latitud float, longitud float, imagen1 varchar, imagen2 varchar, primary key(id))"];
```

Figura 3.17: Inicialización de la Base de datos.





## Capítulo 4

# Pruebas y Resultados

En este capítulo se muestra las pruebas y resultados después de terminar la implementación de todos los módulos de programación. Una vez unidos todos los módulos se realizaron varias pruebas para comprobar el funcionamiento de la aplicación, dichas pruebas se realizaron en un iPod y en un iPhone, antes mencionados en los recursos del proyecto. A continuación se muestran algunas imágenes de los resultados obtenidos.

### 4.1. Principales Opciones

La aplicación cuenta con tres opciones principales:

**Acerca de** En esta opción podemos encontrar una breve información del uso la aplicación así como el nombre de los desarrolladores.

**Cámara** En Esta opción el usuario podrá la dirección en que se encuentran los edificios mas cercanos a su posición y a su perspectiva, esto a través de la cámara de su dispositivo.

**Mapa** Esta opción cuenta con el el mapa de la unidad (con diferentes tipos de vista), así como las opciones de rutas, mostrar los edificios de la unidad en el mapa, ubicación del usuario y la búsqueda de información/ Edificios.

### 4.2. Pruebas

#### 4.2.1. Mapa

Para empezar realizamos pruebas de todas las funciones contenidas dentro de la opción de mapa.

## Edificios

Para empezar probamos el botón de *Edificios*, el cual muestra la ubicación de todos los edificios mediante marcadores (pines) en el mapa (en la figura 4.1 se muestran los resultados obtenidos).



Figura 4.1: Prueba del botón de *Edificios*.

## Ubicación

Al realizar esta prueba nos ubicamos en el edificio G como se muestra en la figura 4.2 y se obtuvieron los resultados mostrados en la figura 4.3 (para esta prueba se uso la red interna de la UAM *WiFi UAM Azc*).



Figura 4.2: Posición para la prueba del botón *Ubicación*.

Las principales diferencias al realizar esta prueba en un iPod y en un iPhone son:

- **Mapas** - La diferencia de los mapas (mostrados en la figura 4.3) es debido a la versión del sistema operativo, en la versión 5 (iPhone) o anteriores están proporcionados el servicio de Google Maps y de la versión 6 (iPod) en adelante están proporcionados por el servicio de Apple.
- **Presición** - La presición de la ubicación varia ya que esta es obtenida de diferentes formas, en el caso del iPod es mediante Wi-Fi y en el iPhone mediante GPS.

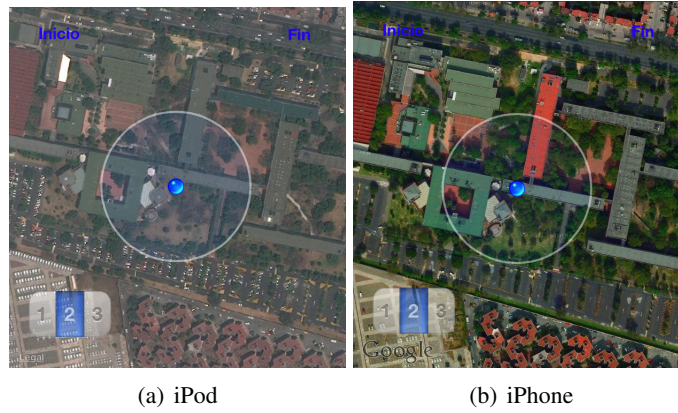


Figura 4.3: Prueba del botón de *Ubicación* en dos dispositivos.

### Como llegar

Antes de comenzar esta prueba, tomamos la ubicación obtenida en la prueba anterior, esto debido que como requerimiento para el uso del botón *Como Llegar* tenemos que usar la ubicación del usuario para posicionar inicialmente los marcadores (pines) de **Inicio** y **Fin**. Diseñamos una ruta entre los edificios H y F (figura 4.4) y obtuvimos los siguientes resultados (figura 4.5).



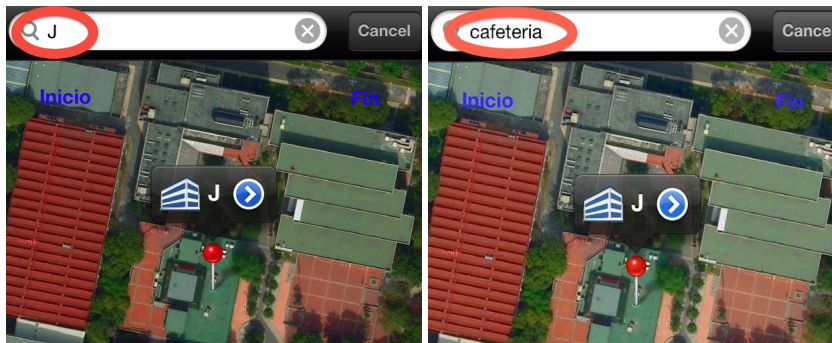
Figura 4.4: Edificios tomados para ejemplo del botón *Como Llegar*.



Figura 4.5: Prueba del botón *Como Llegar*.

### Búsqueda De Edificios

En esta prueba realizamos la búsqueda de un edificio tanto por su nombre así como el contenido de la información almacenada (figura 4.6), esto a través de la *Barra de Búsqueda* ubicada en la parte superior de pantalla.



(a) Nombre de Edificio

(b) Contenido

Figura 4.6: Pruebas de *Búsqueda de Edificios* .

### 4.2.2. Cámara

Antes de realizar las pruebas de esta función primero nos ubicamos en el Edificio I con vista hacia el norte (como se muestra en la figura 4.7), después presionamos el botón de *Mostrar Edificios* y obtuvimos el siguiente resultado (figura 4.8).



Figura 4.7: Posición para la prueba de la opción de *Cámara*.

Una de las diferencias visibles en esta opción entre un iPhone y un iPod (figura 4.8), son los *Botones de calibración*, los cuales se muestran solamente en el iPod debido a que este dispositivo no cuenta con una brújula, a diferencia del iPhone, por lo tanto es necesario que el dispositivo se encuentre inicialmente apuntando en dirección al Norte antes de presionar el botón de *Mostrar Edificios*, una vez hecho esto se podrá usar de forma natural (girando el dispositivo) o mediante los botones.



Figura 4.8: Pruebas al presionar el botón de *Mostrar Edificios*.

## Capítulo 5

# Conclusión

El principal objetivo de este trabajo era diseñar una aplicación para mostrar el campus de la UAM Azcapotzalco a cualquier tipo de usuario, proponiendo realizar esta aplicación en el sistema operativo iOS. Durante el desarrollo de este proyecto se observó que existen diferencias en la implementación del código para diversas funciones dependiendo la versión del sistema operativo en la cual se elabore dicho proyecto. Se observó como el desarrollo a pesar de ser en la misma plataforma tuvo diferencias en los distintos dispositivos que se usaron para la implementación de este proyecto, aún con estas diferencias se pudo desarrollar una aplicación capaz de funcionar en ambos dispositivos, los cuales cuentan con diferentes versiones de sistema operativo.

Durante el desarrollo del módulo de la mapa se observaron algunas de las diferencias antes mencionadas, por ejemplo al obtener la ubicación mediante Wi-Fi el margen de error es mayor comparado con el GPS ya que este hace uso de la mayor cantidad de satélites disponibles para una mejor triangulación y la ubicación obtenida por Wi-Fi sólo se basa en la ubicación de los *routers*<sup>1</sup> previamente almacenada, como consecuencia al no contar con una buena precisión de la ubicación la perspectiva del dispositivo se ve afectada.

Una dificultad para el desarrollo del módulo de la cámara fueron diferencias en el hardware utilizado, en el caso del iPod no cuenta con la brújula por lo cuál se tuvo que implementar en base al giroscopio, produciendo un

---

<sup>1</sup>Dispositivo que proporciona conectividad a nivel de red o nivel tres en el modelo OSI.

mayor margen de error tomando en cuenta que este dispositivo no cuenta con la referencia geográfica, lo cual ocasiona un inconveniente para el usuario debido a que este debe de conocer la orientación geográfica.

Los trabajos a futuro para esta aplicación son:

- **Agregar la función de rutas sobre la opción de cámara** - mostrar señalizaciones de como llegar a una ubicación usando la camara del dispositivo.
- **Modificar en tiempo real las rutas** - en caso de salir de la ruta marcada volver a calular una nueva ruta.
- **Rutas alternas** - mostrar mas de una ruta como resultado de la opción *Como Llegar*.
- **Integración de los Servicios de la UAM** - añadir los servicios proporcionados por la UAM (Módulo escolar, consulta de libros en la biblioteca, inscripción a idiomas).
- **Horario** - mostrar el horario del alumno en base a su matricula.



# Bibliografía

- [1] F. Ayala. *Aplicación de realidad aumentada: Marcador de señales de tránsito*. PhD thesis, UAM unidad Azcapotzalco, 2009.
- [2] Nick Broughall. The history of ios [en línea]. "Website", Septiembre 2010. <http://www.gizmodo.com.au/2010/09/the-history-of-ios/>.
- [3] Dieter Bohn. ios: A visual history [En línea]. "Website", December 2011. <http://www.theverge.com/2011/12/13/2612736/ios-history-iphone-ipad>.
- [4] C. A. Sánchez. *Sistema móvil de reconocimiento de objetos y posicionamiento en un exposición basado en realidad aumentada*. PhD thesis, UAM unidad Azcapotzalco, 2011.
- [5] Definición ABC. Definición de geolocalización [En línea]. "Website", Noviembre 2012. <http://www.definicionabc.com/geografia/geolocalizacion.php>.
- [6] TomTom International BV. App tomtom navegación para iphone [En línea]. "Website", Noviembre 2012. [http://www.tomtom.com/es\\_mx/products/mobile-navigation/index.jsp](http://www.tomtom.com/es_mx/products/mobile-navigation/index.jsp).
- [7] Apple Inc. ios features [En línea]. "Website", Noviembre 2012. <https://developer.apple.com/technologies/ios/features.html>.
- [8] Apple Inc. iphone app programming guide [En línea]. "Website", Noviembre 2012. [http://manuals.info.apple.com/en/iphone\\_programing\\_guide.pdf](http://manuals.info.apple.com/en/iphone_programing_guide.pdf).
- [9] Apple Inc. Xcode [En línea]. "Website", Noviembre 2012. <https://developer.apple.com/xcode/>.

- [10] Google Inc. Acerca de google maps [En línea]. "Website", Noviembre 2012. <http://support.google.com/maps/bin/answer.py?hl=esanswer=7060>.
- [11] Google Inc. Versión 3 de google maps [En línea]. "Website", Noviembre 2012. <http://code.google.com/intl/es/apis/maps/documentation/javascript/basics.html>.
- [12] Cory Janssen. ios [En línea]. "Website", Noviembre 2012. <http://www.techopedia.com/definition/25206/ios>.
- [13] SHAPE. Checkin+ for iphone [En línea]. "Website", Noviembre 2012. <http://www.shape.ag/en/products/apps.php?platform=iphone>.
- [14] SQLite. About sqlite [En línea]. "Website", Noviembre 2012. <http://www.sqlite.org/about.html>.
- [15] UNAM. Unam 360 [En línea]. "Website", Noviembre 2012. <http://m.unam.mobi/proyectos>.
- [16] Inc. Wikimedia Foundation. ios version history [En línea]. "Website", 2012. [http://en.wikipedia.org/wiki/IOS\\_version\\_history](http://en.wikipedia.org/wiki/IOS_version_history).
- [17] Inc. Wikimedia Foundation. Sqlite [En línea]. "Website", Noviembre 2012. <http://en.wikipedia.org/wiki/SQLite>.
- [18] Wei-Meng Lee. *Beginning iOS 5 Application Development*. O'Reilly Media, 2011.
- [19] Matt Neuburg. *Programming iOS 5*. O'Reilly Media, 2011.

## Apéndice A

# Manual de Usuario

### A.0.3. Acerca de

Esta opción es inicialmente cargada, en ella podemos observar un mensaje de bienvenida con el nombre de nuestro dispositivo (en nuestro caso es Sosasoft), así como el título de la aplicación *iUAM-a* (el cual es un enlace directo a la página web de la UAM unidad azcapotzalco), las principales opciones de la aplicación se encuentran ubicadas en un menú ubicado en la parte inferior de la pantalla como se muestra en la figura A.1):



Figura A.1: Pantalla inicial de la aplicación.

### A.0.4. Mapa

Dentro de esta opción podemos encontrar las siguientes funciones:

- **Mapa desplazable** (mapa inicialmente posicionado en la UAM Azcapotzalco, con la opción de diferentes tipos de vistas, figura A.2.)
- **Búsqueda de Edificios** (opción para buscar edificios así como información relevante a ellos)
- **Como llegar** (opción para trazar rutas mediante el uso de dos marcadores; inicio y fin)
- **Edificios** (opción capaz de mostrar marcadores de todos los edificios en el mapa)
- **Ubicación**( opción para conocer la posición actual del dispositivo)

#### A.0.5. Mapa Desplazable

Mapa con la capacidad de desplazarse a deseo del usuario (posicionado inicialmente en la UAM unidad azcapotzalco), con la opción de tres diferentes vistas como se muestra en la figura A.2.

#### A.0.6. Búsqueda de Edificios

Una forma rápida y sencilla para buscar edificios es a través de la barra de búsqueda, ubicada en la parte superior de la pantalla, mediante la cual podremos realizar una búsqueda por medio del teclado (por nombre o información del edificio), en caso de éxito se pondrá marcador (pin) en el mapa, en caso contrario se mostrara un mensaje de error. *Es importante recordar que se solo encuentran almacenados los edificios más relevantes para los alumnos de CBI, como se menciona en los objetivos de este proyecto.* (Todos los marcadores además de contar con una etiqueta con el nombre el edificio cuentan un link ,botón color azul capaz de mostrar información detallada de cada edificio, como se observa en la figura A.4).

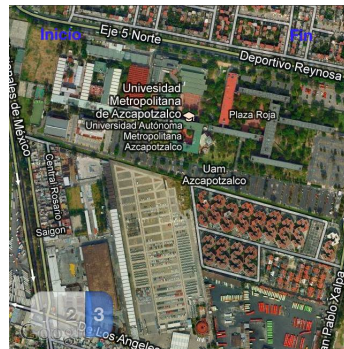
#### A.0.7. Edificios

Este botón despliega en el mapa marcadores (pines) de todos los edificios en el mapa, como se observa en la figura A.4.



(a) Standart

(b) Satellite



(c) Hybrid

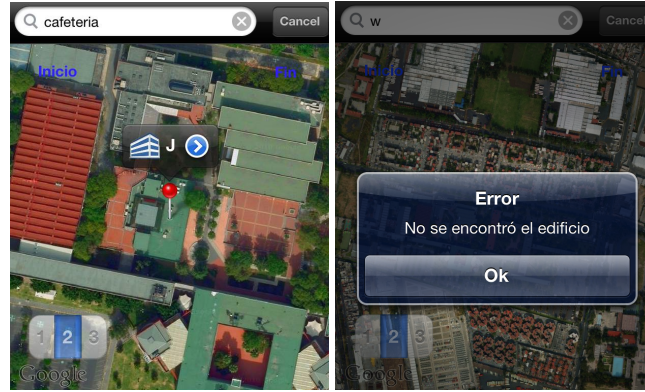
Figura A.2: Vistas del mapa

### A.0.8. Ubicación

Este botón muestra en el mapa la ubicación del usuario mediante un marcador color azul, con la etiqueta "Tu estas aquí", capaz de desplazarse según tu posición( en caso de no tener los permisos de localización nos mostrara un mensaje de error).

### A.0.9. Como llegar

Antes de usar esta opción primero tenemos que obtener la ubicación del usuario (mediante el botón de *Ubicación*), en caso de no hacer esto se mostrará una advertencia (como se muestra en la figura A.5), una vez realizado esto dar clic en la etiqueta *inicio* (Esquina superior izquierda del



(a) Búsqueda

(b) Mensaje de Error

Figura A.3: Búsqueda de Edificios

mapa), con lo cual se pondrá un marcador ( color morado, con la etiqueta de inicio) en el mapa con la ubicación actual del usuario, con la posibilidad de cambiar dicha posición al gusto del usuario, después haremos lo mismo para el marcador de *fin*. Al terminar esto dar clic en el botón de **Como llegar** y se trazará la ruta en color azul como se ve en la figura A.6.



Figura A.4: Marcadores e información de Edificios

#### A.0.10. Cámara

Dentro de esta opción podemos ver en la pantalla la imagen obtenida a través de la cámara del dispositivo y un botón: *Muestra Edificios* (localizado en la parte superior de la pantalla), una vez presionado este botón se mostrarán las etiquetas de los edificios más cercanos en base a la posición actual y a la perspectiva del usuario, lo cuales cuentan con la opción de dar click para mostrar la información del edificio como se mostró en la figura A.4.



Figura A.5: Ubicación del Usuario

Un caso diferente ocurre en el iPod (cuarta generación), que a diferencia del iPhone (4G o superior), no cuenta con una brújula, por lo tanto al dar clic **Muestra Edificios** aparecerá un mensaje con la siguiente leyenda: *Este dispositivo no cuenta con una brújula, para un mejor uso orientar el iPod hacia el norte (la precisión puede variar)*. El usuario podrá ir moviendo el dispositivo de manera natural o por medio de dos botones de calibración (flechas ubicadas en la parte superior de la pantalla).



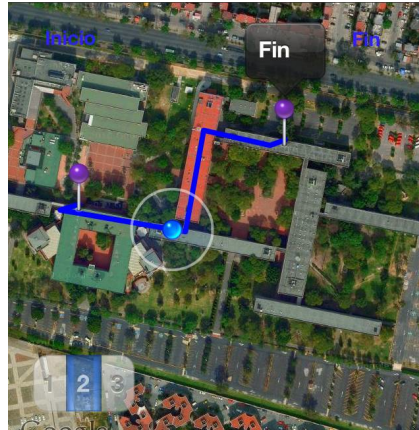
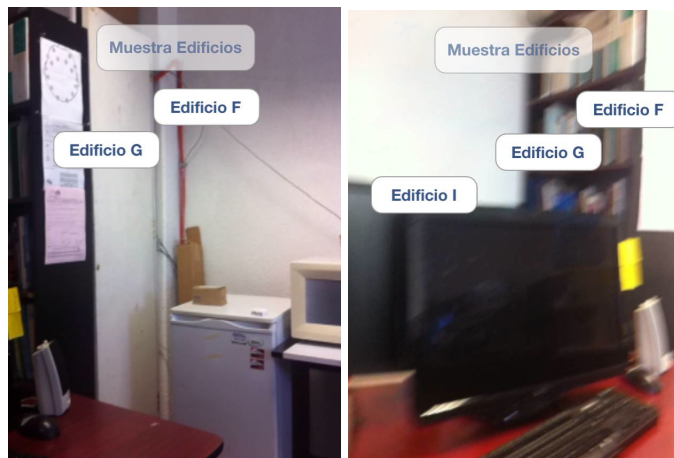


Figura A.6: Ejemplo de opción *como llegar*.



(a) Prueba uno

(b) Prueba dos

Figura A.7: Pruebas de la opción cámara realizadas en un iPhone 4G



(a) Mensaje

(b) Calibración

Figura A.8: Pruebas de la opción cámara realizadas en un iPod 4G