



**UNIVERSIDAD TECNOLÓGICA METROPOLITANA
FACULTAD DE INGENIERÍA
ESCUELA DE INFORMÁTICA**

IMPLEMENTACIÓN DE API DE GOOGLE MAPS PARA VISUALIZAR MAPAS EN TV DIGITAL

**TRABAJO DE TITULACIÓN PARA OPTAR AL TÍTULO DE
INGENIERO EN INFORMÁTICA**

PROFESOR GUÍA: MAURO CASTILLO VALDÉS

ALUMNO : JUAN FERNÁNDEZ ARRAÑO

**SANTIAGO - CHILE
2013**

NOTA OBTENIDA:

Firma y timbre de la autoridad responsable

RESUMEN

En este trabajo, se presenta un nuevo proyecto para los clientes de televisión que tengan este servicio contratado de la empresa GTD Manquehue, que permite visualizar mapas para TV digital usando la API de Google Maps, cuyas características fundamentales consisten en ser útil, rápido y eficaz, en el cual el usuario de televisión digital se puede orientar fácilmente sin costos adicionales a los servicios ya contratados, a fin de optimizar los tiempos y las distancias en ubicar la dirección requerida de manera simple y de fácil uso para cualquier persona de cualquier edad, sexo o condición social.

Para poder llevar a cabo esto, GTD Manquehue necesitó implementar una aplicación que utilice la API de Google Maps, esta implementación utiliza las distintas interfaces de comunicación para la televisión digital. Es decir, por medio de un decodificador conectado a la televisión, en la cual se despliega el portal interactivo de GTD Manquehue, que muestra las aplicaciones o servicios al usuario, que a través de un control remoto puede seleccionar para visitar y así utilizar la aplicación alojada en los servidores de GTD Manquehue, en el navegador web "Galeon" que se encuentra en el decodificador.

ABSTRACT

This work presents a new project for television customers who have this contracted firm service GTD Manquehue, that allows to view maps for digital TV using the Google Maps API, whose fundamental characteristics consist in being useful, fast and effective, in which the user of digital television can easily Orient without additional costs to the services already contracted in order to optimize the time and distances to locate the direction required simple and easy to use for anyone of any age, sex or social status.

To be able to carry out this, GTD Manquehue needed to deploy an application that uses the Google Maps API, its implementation uses various communication interfaces for digital television. I.e. through a decoder connected to the television, which unfolds the interactive portal of GTD Manquehue, showing the applications or services to the user, who can select to visit and use the application hosted in servers of GTD Manquehue, web browser "Galleon" located in the decoder through a remote control.

ÍNDICE

INTRODUCCIÓN.....	1
CAPÍTULO 1: ANTECEDENTES GENERALES	3
1.1 MOTIVACIÓN	4
1.2 OBJETIVOS	6
1.3 ALCANCES Y LIMITACIONES	7
CAPÍTULO 2: GOOGLE MAPS	9
2.1 ¿CÓMO FUNCIONA GOOGLE MAPS?	12
2.2 FUNCIONES.....	12
2.2.1 MAP OPTIONS.....	12
2.2.2 GETTERS y SETTERS.....	13
2.2.3 CODIFICACIÓN GEOGRÁFICA	13
2.2.4 CREAR MARCADOR	24
CAPÍTULO 3: “SET TOP BOX”	26
3.1 ESQUEMA DE BLOQUES DE UN SET-TOP BOX	28
3.2 ARQUITECTURA SET-TOP BOX	29
3.3 CARACTERÍSTICAS COMUNES DE UN SET-TOP BOX.....	30
CAPÍTULO 4: METODOLOGÍA	33
4.1 DESARROLLO ÁGIL DE SOFTWARE.....	34
4.2 FEATURE DRIVER DEVELOPMENT	37
4.2.1 PROCESO.....	38
4.3 ROLES Y RESPONSABILIDADES	40
4.3.1 KEY ROLES / ROLES CLAVES	40
4.3.2 SUPPORTING ROLES / ROLES DE SOPORTE	42
4.3.3 ADDITIONAL ROLES / ROLES ADICIONALES.....	43
4.4 COMPARACIONES RESPECTO A OTRAS METODOLOGÍAS ÁGILES ..	44
4.5 VENTAJAS	46
4.6 DESVENTAJAS	47
CAPÍTULO 5: CONSTRUCCIÓN.....	48
5.1 BARRA DE BÚSQUEDA.....	49
5.2 WEB SERVICE	50

5.3 API DE GOOGLE MAPS	51
5.4 DIAGRAMAS USADOS PARA LA CONSTRUCCIÓN DEL SISTEMA.....	56
5.4.1 ESQUEMA GENERAL DE LA APLICACIÓN	56
5.4.2 ESQUEMA DEL USUARIO	57
5.4.3 ESQUEMA DEL DECODIFICADOR	58
CAPÍTULO 6: PRUEBAS.....	60
6.1 PRUEBAS REALIZADAS EN EL COMPUTADOR	61
6.2 PRUEBAS REALIZADAS EN EL DECODIFICADOR	62
CAPÍTULO 7: PUESTA EN MARCHA	65
CONCLUSIONES.....	69
BIBLIOGRAFÍA.....	71
WEBGRAFÍA	72
ANEXO	75
GLOSARIO DE TÉRMINOS.....	75

ÍNDICE DE FIGURAS

Figura N° 1: Empresas que conforman el grupo GTD.	5
Figura N° 2: Atributos del objeto GeocodeRequest.	15
Figura N° 3: Solicitud de búsqueda.	16
Figura N° 4: Campos del objeto GeocoderResults.	17
Figura N° 5: Set Top Box.	30
Figura N° 6: Procesos de la metodología FDD.	38
Figura N° 7: Metodología FDD versus otras metodologías ágiles.	45
Figura N° 8: Barra de búsqueda.	49
Figura N° 9: Funcionamiento de Web Service.	50
Figura N° 10: Script para cargar API Google Maps.	51
Figura N° 11: Estilo CSS del div map_canvas.	52
Figura N° 12: Características por defecto del mapa.	52
Figura N° 13: Script para cargar el objeto de mapa.	52
Figura N° 14: Funciones de setZoom(), panBy y setMapTypeId.	54
Figura N° 15: Esquema global de la aplicación.	56
Figura N° 16: Representación del sistema para el usuario final.	57
Figura N° 17: Aplicación desde el punto de vista del decodificador.	58
Figura N° 18: Error de la función xsystem.	62
Figura N° 19: Alerta cuando se ingresa una dirección inexistente.	64
Figura N° 20: Carta Gantt.	67

INTRODUCCIÓN

Dada la gran cantidad de aplicaciones existentes, las cuales permiten hacer búsquedas, tales como buscar una dirección o cualquier información de interés para el usuario, que han sido desarrolladas para teléfonos móviles o computadoras. Como es el caso de Google, que tiene dentro de sus servicios, YouTube, Gmail, Google Maps, entre otros. En este caso se tratará específicamente sobre la implementación de la API de Google Maps para los usuarios de televisión digital, ya que por su funcionalidad permitirá a las distintas personas que interactúan con esta aplicación de la API, localizar diversas ciudades, países, lugares turísticos o alguna dirección en particular, además de poder observar en el mapa con mayor o menor detalle.

La interactividad que ofrece la televisión digital adquiere diferentes niveles que van desde una interactividad básica, como por ejemplo la realizada con el televisor (modificar el volumen o cambiar de canal), pasando por una interactividad media como sería solicitar una película en un determinado momento (vídeo bajo demanda) o hasta un nivel alto de interactividad, en el cual el telespectador puede influir directamente en el

desenlace de un programa en emisión, por ejemplo mediante una votación realizada en muchos casos a través de un canal de retorno.

Con el fin de comunicar al usuario la posibilidad de acceso a aplicaciones interactivas, los operadores o canales de televisión presentan un pequeño menú o portal interactivo, en el cual se muestra una lista al usuario de qué aplicación o grupo de aplicaciones interactivas puede visitar. Estas aplicaciones comúnmente también se clasifican según el servicio que realicen o desarrollen, por ejemplo, ver el tiempo actual en la ciudad de Santiago.

En el presente trabajo se implementará una aplicación interactiva, que consiste en que el usuario de televisión pueda visualizar mapas a través de un TV conectado al decodificador usando la API de Google Maps, este sería un programa adicional a los contenidos de televisión que tiene la empresa GTD a los que se puede acceder y ejecutar en un decodificador interactivo. El usuario es quien decide si ver o no las aplicaciones interactivas mediante una acción simple con el control remoto. Para llevar a cabo el objetivo de la implementación de esta aplicación, se tuvo que participar en las distintas etapas en conjunto con la empresa GTD.

A la fecha la aplicación se encuentra en fase de mantención.

CAPÍTULO 1: ANTECEDENTES GENERALES

1.1 MOTIVACIÓN

El presente trabajo, trata de la implementación de una aplicación para la empresa de telecomunicaciones llamada GTD Manquehue, sus servicios que ofrece son: la telefonía (tradicional y por IP), internet (ADSL y FTTH¹), televisión (digital y HD).

El compromiso que tiene hacia los clientes, es que ellos han decidido confiar en GTD, lo que obliga a satisfacer sus necesidades de comunicación y ofrecerles un servicio de excelencia. Hacia el personal, tener personas totalmente dispuestas a trabajar en equipo que le permite a GTD ser un grupo especialmente exitoso. Creemos en el valor y la dignidad del trabajo bien realizado.

Respecto a la excelencia e integridad, está el constante esfuerzo por alcanzar la excelencia en cada cosa que se hace. Se desea ser la mejor opción para los clientes, la mejor empresa para trabajar en ella así como entregar los mejores resultados hacia sus accionistas.

El Grupo GTD es un Holding de empresas especialistas en telecomunicaciones y las empresas que lo componen son:

¹ Ver anexo, glosario de términos, pág. 77



Figura N° 1: Empresas que conforman el grupo GTD.

En el año 2008 GTD Manquehue como un paso adelante en su estrategia comercial, sumó a sus servicios, con lo que aumentó su participación como un actor relevante en el mercado de las telecomunicaciones.

Esta aplicación de visualización de mapas mediante la funcionalidad de la API de Google Maps, se desarrolló para la empresa GTD Manquehue, en el área de GTD imagen, con fin de que se implemente en el área de televisión, de modo que el usuario de televisión pueda visualizar mapas en televisión digital mediante el uso de la API² de Google Maps.

² Ver anexo, glosario de términos, pág. 76

La necesidad de la implementación de este proyecto, tiene el objetivo de ser una estrategia de mercado, puesto que al ofrecer un nuevo servicio a los clientes que tengan contratado los servicios de televisión de GTD, va a permitir tener a su disposición el servicio de búsqueda y visualizar mapas en TV digital que provee la API de Google Maps.

1.2 OBJETIVOS

1.2.1 OBJETIVO GENERAL

Implementar una aplicación computacional para televisión digital, que mediante un decodificador, permita al usuario interactuar con las funcionalidades que provee Google Maps.

1.2.2 OBJETIVOS ESPECÍFICOS

Para llevar a cabo el desarrollo de la aplicación se deben cubrir las siguientes etapas:

- Revisión de requerimientos (interfaz y funciones), para ello se realizan controles de avance diario, es decir, se hace un seguimiento del estado de la aplicación.
- Implementar las funciones de la barra de búsqueda para que el usuario pueda ingresar la dirección que desea buscar en Google Maps.
- Implementar las funciones de Web Service y asociar con las funciones de la API de Google Maps.
- Realizar pruebas al funcionamiento de la aplicación en la televisión mediante el decodificador.

1.3 ALCANCES Y LIMITACIONES

Puesto que Google Maps está desarrollado para múltiples plataformas, por lo que puede ser usado para cualquier sistema que quiera disponer de esta funcionalidad, en este caso se usará para que el usuario pueda ver y utilizar Google Maps en televisión digital o interactiva mediante un receptor de televisión o conocido comúnmente como decodificador, dado que presenta una arquitectura similar a la de un computador de escritorio o portátil.

Esta aplicación sólo estará disponible para aquellas personas que tengan contratado el servicio de televisión que provee GTD Manquehue, sea TV digital o en “High Definition” (HD). Dado que, los usuarios disponen de un decodificador, el cual tiene funciones desarrolladas por el fabricante (XAVi), como la de tener la MAC³ (Media Access Control) del usuario, que en este caso se usará para el Web Service⁴.

Las limitaciones que se presentan en esta situación, es que si bien el decodificador tiene una arquitectura similar a la de un computador, no es recomendable tener aplicaciones demasiado “pesadas” puesto que el tiempo de respuesta de un computador es más rápido versus el tiempo de respuesta de un STB o decodificador.

³ Ver anexo, glosario de términos, pág. 79

⁴ Ver anexo, glosario de términos, pág. 82

CAPÍTULO 2: GOOGLE MAPS

Consiste en un servidor de aplicaciones de mapas en la Web. Originalmente soportaría solo a los usuarios de Internet Explorer y Mozilla Firefox, pero el soporte para Opera y Safari fue agregado el 25 de febrero de 2005. El software estuvo en su fase beta durante 6 meses antes de convertirse en parte de Google Local, el 6 de octubre de 2005.

Es muy similar a Google Earth, que es una aplicación para múltiples sistemas operativos como Windows, Mac, Linux, con la cual se observan distintos tipos de vistas del globo terráqueo, sea de día o de noche, pero que no es fácil de integrar a páginas Web. Está disponible para Android y Java ME.

Dispone de imágenes de mapas desplazables, así como fotos satelitales del mundo e incluso la ruta entre diferentes ubicaciones o imágenes a pie de calle como Google Street View.

También, permite la posibilidad de que cualquier propietario de una página Web integre muchas de sus características a su sitio, vale decir, usar las funcionalidades de la API, como mostrar a los usuarios las ubicaciones de las sucursales de la empresa en un tipo de mapa determinado, por ejemplo, un mapa satelital.

Dado que se utilizan una gran cantidad de archivos de Javascript en las aplicaciones webs de Google, con estos archivos se crea Google Maps. Puesto que, el usuario puede mover el mapa, la visualización del mismo se baja desde el servidor. Cuando un usuario busca un negocio, la ubicación es marcada por un indicador en forma de pin, el cual es una imagen PNG transparente sobre el mapa. Para lograr la conectividad sin sincronía con el servidor, Google aplicó el uso de AJAX dentro de esta aplicación. Es una aplicación para el desarrollo de mapas.

Google Maps fue desarrollado originalmente por dos hermanos Daneses, Lars y Jens Rasmussen, co-fundadores de Where 2 Technologies una empresa dedicada a la creación de soluciones de mapeo. La empresa fue adquirida por Google en octubre de 2004, y los dos hermanos luego crearon Google Maps (también son los que están detrás de Google Wave).

Antes de que hubiera una API pública, algunos desarrolladores descubrieron la manera de vulnerar Google Maps para incorporar los mapas en sus propios sitios web. Esto llevó a Google a la conclusión de que había una necesidad de una API pública, y en junio de 2005 fue lanzado públicamente.

2.1 ¿CÓMO FUNCIONA GOOGLE MAPS?

Es HTML⁵, CSS⁶ y JavaScript trabajando junto. Los mapas son solo imágenes que se cargan en el fondo a través de peticiones ejecutadas por la tecnología de AJAX⁷, y se insertan en un <div> en la página HTML. Mientras se navega en el mapa, el API envía información acerca de las nuevas coordenadas y los niveles de “zoom” del mapa a través de AJAX y esto retorna las imágenes. El API consiste de archivos Javascript que contienen las clases, métodos y propiedades que se usan para el comportamiento de los mapas.

2.2 FUNCIONES

2.2.1 MAP OPTIONS

MapOptions contiene la información de cómo queremos ver el mapa y cómo queremos que se comporte. Se requiere tres propiedades:

- Zoom: Define el “zoom” inicial. Debe ser un número entre el 1 y el 23. El 1 es la vista del mapa completo y el 23 es la vista del lugar con mayor detalle.

⁵ Ver anexo, glosario de términos, pág. 77

⁶ Ver anexo, glosario de términos, pág. 77

⁷ Ver anexo, glosario de términos, pág. 75

- Center: Define el centro del mapa con las coordenadas. Las coordenadas debe indicarse usando el método `google.maps.LatLng` (latitud, longitud).
- MapTypeId: Define qué tipo de mapa se desea mostrar al inicio.

2.2.2 GETTERS y SETTERS

Existen métodos que nos permiten solo modificar y/o obtener los resultados de las propiedades que son requisitos, tales como:

- `getZoom()`
- `setZoom(1-23)`
- `getCenter()`
- `setCenter(google.maps.LatLng(latitud, longitud))`
- `setMapTypeId(google.maps.MapTypeId.*)`

2.2.3 CODIFICACIÓN GEOGRÁFICA

La codificación geográfica es el proceso de transformar direcciones (como "1600 Amphitheatre Parkway, Mountain View, CA") en coordenadas geográficas (como 37.423021 de latitud y -122.083739 de longitud), que se pueden utilizar para colocar marcadores o situar el mapa.

El API de Google Maps proporciona una clase llamada geocoder que permite codificar de forma geográfica las direcciones de manera dinámica a partir de los datos introducidos por el usuario. Estas solicitudes son limitadas para evitar un abuso del servicio. Si por el contrario se desea codificar de forma geográfica direcciones conocidas y estáticas, se puede consultar la documentación del API de codificación geográfica.

Solicitudes de codificación geográfica

El acceso al servicio de asignación de identificadores geográficos tiene lugar de forma asíncrona, dado que el API de Google Maps necesita realizar una llamada a un servidor externo. Por ese motivo, se debe incluir un método de devolución de llamada para que se ejecute al completar la solicitud. Este método de devolución de llamada procesará los resultados. Por lo que se debe tener en cuenta que el geocoder puede devolver más de un resultado.

Se debe acceder al servicio de asignación de identificadores geográficos del API de Google Maps desde el código a través del objeto `google.maps.Geocoder`. El método `Geocoder.geocode()` inicia la solicitud al servicio de asignación de identificadores geográficos y transmite un

objeto literal GeocodeRequest que contiene los términos de entrada y un método de devolución de llamada que se ejecutará al recibir la respuesta.

El objeto literal GeocodeRequest contiene los siguientes campos:

```
{  
  address: string,  
  latLng: LatLng,  
  bounds: LatLngBounds,  
  region: string  
}
```

Figura N° 2: Atributos del objeto GeocodeRequest.

Ahora, se menciona en qué consiste cada campo del objeto llamado GeocodeRequest:

- address (obligatorio): es la dirección que deseas codificar de forma geográfica.
- latLng (obligatorio): la latitud y la longitud (LatLng) para las que deseas obtener la dirección interpretable por humanos más cercana.
- bounds (opcional): los límites de latitud y de longitud (LatLngBounds) dentro de los que quieres predeterminedar los resultados de codificación geográfica con mayor importancia. Para

obtener más información, se consulta la sección Cómo especificar la ventana gráfica que encontrarás a continuación.

- **region** (opcional): el código de región, especificado como una sub-etiqueta región del lenguaje IANA. En la mayoría de los casos, estas etiquetas se asignan directamente a valores de dos caracteres ccTLD ("dominio de nivel superior") ya conocidos.

Los parámetros **bounds** y **region** solo influirán en los resultados del geocoder, no los restringirán por completo.



Figura N° 3: Solicitud de búsqueda.

En la figura la solicitud realizada indica la ciudad de Sydney en el estado de New South Wales.

Respuestas de codificación geográfica

Para el servicio de asignación de identificadores geográficos se necesita un método de devolución de llamada que se ejecute al recuperar los resultados del geocoder. Esta devolución de llamada debe transmitir dos parámetros que alojen (**results**) y un código (**status**), en ese orden. Puesto

que el geocoder puede devolver más de una entrada, el objeto literal `GeocoderResults` es una matriz.

Resultados de codificación geográfica



El objeto literal `GeocoderResults` representa un único resultado de codificación geográfica y presenta la siguiente forma:

```
results[]: {  
  types[]: string,  
  formatted_address: string,  
  address_components[]: {  
    short_name: string,  
    long_name: string,  
    types[]: string  
  },  
  geometry: {  
    location: LatLng,  
    location_type: GeocoderLocationType  
    viewport: LatLngBounds,  
    bounds: LatLngBounds  
  }  
}
```

Figura N° 4: Campos del objeto `GeocoderResults`.

A continuación, se explican en qué consisten los campos mostrados anteriormente:

- `types[]` es una matriz que indica de qué tipo son los resultados obtenidos. Esta matriz contiene una o más etiquetas que identifican el tipo de elemento que se ha obtenido en el resultado. Por ejemplo, un código geográfico de "Chicago" devuelve "locality", lo que indica que "Chicago" es una ciudad y además devuelve "political", lo que significa que es una entidad política.
- `formatted_address` es una cadena que contiene la dirección interpretable por humanos de la ubicación. Esta dirección suele corresponder con la "dirección postal", que a veces difiere de un país a otro. (Se debe de considerar que en algunos países, como Gran Bretaña, no se permite la distribución de direcciones postales reales debido a restricciones de licencia). Esta suele estar formada por uno o varios componentes de la dirección. Por ejemplo, la dirección "111 8th Avenue, New York, NY" contiene componentes independientes para "111 8th Avenue" (una dirección postal), "New York" (la ciudad) y "NY" (el estado de EE.UU.). Los componentes de la dirección se describen más abajo. Para obtener más información sobre los diferentes tipos, consulta la sección Tipos de componentes de la dirección que encontrarás a continuación.

- `address_component[]` es una matriz que contiene los diferentes componentes de la dirección, tal como se ha explicado anteriormente.
- `geometry` contiene la siguiente información:
 -  `location` contiene el valor de latitud y longitud codificado de forma geográfica. Ten en cuenta que este valor se devuelve como un objeto `LatLng`, no como una cadena con formato.
 -  `location_type` almacena datos adicionales sobre la ubicación especificada. Actualmente se admiten los siguientes valores:
 - `google.maps.GeocoderLocationType.ROOFTOP` indica que el resultado obtenido refleja un código geográfico preciso.
 - `google.maps.GeocoderLocationType.RANGE_INTERPOLATED` indica que el resultado obtenido refleja una aproximación (normalmente en una calle o carretera) interpolada entre dos puntos precisos (por ejemplo, intersecciones). Los resultados interpolados se suelen obtener cuando no hay ningún código geográfico preciso disponible para una dirección postal.
 - `google.maps.GeocoderLocationType.GEOMETRIC_CENTER` indica que el resultado obtenido corresponde al

centro geométrico de un resultado del tipo polilínea (por ejemplo, una calle) o polígono (una región).

- `google.maps.GeocoderLocationType.APPROXIMATE`

indica que el resultado obtenido es aproximado.

- `viewport` almacena la ventana gráfica recomendada para el resultado obtenido.
- `bounds` (se devuelve de forma opcional) almacena los límites de latitud y de longitud (`LatLngBounds`) que pueden contener por completo el resultado obtenido. Ten en cuenta que estos límites pueden no coincidir con la ventana gráfica recomendada (Por ejemplo, San Francisco incluye Los Farallones, que técnicamente forman parte de la ciudad pero no se muestran en la ventana gráfica).

El geocoder devuelve las direcciones con la configuración de idioma preferida del navegador, o en el idioma especificado al cargar el código JavaScript del API mediante el parámetro `language`.

Tipos de componentes de la dirección

La matriz `types[]` del resultado obtenido indica el tipo de dirección. Estos tipos también pueden aparecer dentro de las matrices `address_components[]` para indicar el tipo del componente de la dirección en concreto. Las direcciones del geocoder pueden tener diversos tipos; los tipos se pueden considerar "etiquetas". Por ejemplo, muchas ciudades pueden estar clasificadas con los tipos `political` y `locality`.

El geocoder HTTP⁸ admite y devuelve los siguientes tipos:

- `street_address` indica una dirección postal precisa.
- `route` indica una carretera identificada (por ejemplo, "US 101").
- `intersection` indica una intersección importante, normalmente de dos carreteras o calles importantes.
- `political` indica una entidad política. Este tipo suele indicar un área poligonal de alguna administración civil.
- `country` indica la entidad política nacional y normalmente, es el tipo de mayor escala que devuelve el geocoder.
- `administrative_area_level_1` indica una entidad política de primer nivel por debajo del nacional. En Estados Unidos, las entidades

⁸ Ver anexo, glosario de términos, pág. 78

que corresponden a este nivel administrativo son los estados. No todos los países cuentan con este nivel administrativo.

- `administrative_area_level_2` indica una entidad política de segundo nivel por debajo del nacional. En Estados Unidos, las entidades que corresponden a este nivel administrativo son los condados. No todos los países cuentan con este nivel administrativo.
- `administrative_area_level_3` indica una entidad política de tercer nivel por debajo del nacional. Este tipo indica una división administrativa de menor tamaño. No todos los países cuentan con este nivel administrativo.
- `colloquial_area` indica un nombre alternativo usado con frecuencia para designar la entidad.
- `locality` indica una entidad política equivalente al municipio.
- `sublocality` indica una entidad política de primer nivel por debajo del municipal.
- `neighborhood` indica un barrio identificado.
- `premise` indica una ubicación identificada. Normalmente se trata de un edificio o de un complejo de edificios con un nombre común.
- `subpremise` indica una entidad de primer nivel por debajo del de ubicación identificada. Normalmente se trata de un edificio individual dentro de un complejo de edificios con un nombre común.

- `postal_code` indica un código postal utilizado para enviar correo postal dentro del país.
- `natural_feature` indica un paraje natural destacado.
- `airport` indica un aeropuerto.
- `park` indica un parque identificado.

Además de los tipos anteriores, los componentes de la dirección pueden presentar los siguientes:

- `post_box` indica un apartado postal específico.
- `street_number` indica la calle y el número exacto del edificio.
- `floor` indica la planta del edificio.
- `room` indica la puerta dentro del edificio.

Códigos de estado

El código `status` puede devolver uno de los siguientes valores:

- `google.maps.GeocoderStatus.OK` indica que la codificación geográfica se ha realizado correctamente.
- `google.maps.GeocoderStatus.ZERO_RESULTS` indica que la codificación geográfica se ha realizado correctamente pero no ha

devuelto ningún resultado. Esto puede ocurrir si en la codificación geográfica se incluye una dirección (address) inexistente o un valor de latng en una ubicación remota.

- `google.maps.GeocoderStatus.OVER_QUERY_LIMIT` indica que se ha excedido el cupo de solicitudes.
- `google.maps.GeocoderStatus.REQUEST_DENIED` indica que se ha denegado la solicitud por algún motivo.
- `google.maps.GeocoderStatus.INVALID_REQUEST` normalmente indica que no se ha especificado la solicitud (address o latLng).

2.2.4 CREAR MARCADOR

El uso más común del mapa es visualizar la posición geográfica de algún lugar de interés para el usuario, los marcadores son ideales para este objetivo. Un marcador es básicamente una pequeña imagen posicionada en un lugar específico del mapa.

Para crear un marcador, necesitas usar el objeto `google.maps.Marker`. Este toma un solo argumento y es `google.maps.MarkerOptions`. `MarkerOptions` tiene varias propiedades que puedes usar para hacer que el marcador se vea y comporte de diversas formas. Sólo dos propiedades son requisitos:

- Position: Define las coordenadas donde el marcador va a estar posicionado. Toma las coordenadas usando el método `google.maps.LatLng`.
- Map: Es una referencia al mapa donde quieres añadir el marcador.

Cómo cargar el API de Google Maps

```
<script type="text/javascript"
src="http://maps.googleapis.com/maps/api/js?key=YOUR_API_KEY&sens
or=SET_TO_TRUE_OR_FALSE"></script>
```

La URL incluida en la etiqueta script indica la ubicación de un archivo JavaScript que carga todos los símbolos y las definiciones que necesitas para utilizar el API de Google Maps. La etiqueta script es obligatoria.

El parámetro key incluye la clave de API de la aplicación.

El parámetro sensor de la URL es obligatorio e indica si esta aplicación utiliza un sensor (por ejemplo, un localizador GPS) para determinar la ubicación del usuario. En este ejemplo se ha dejado el parámetro como la variable `set_to_true_or_false` para hacer hincapié en que se debe definir este valor en true o false de forma explícita.

CAPÍTULO 3: “SET TOP BOX”

El “Set Top Box” (STB) o como se conoce en términos simples decodificador, es el hardware que permite la televisión interactiva, dado que por su arquitectura que es similar a la de un computador, pero con menor capacidad, se conecta mediante fibra óptica hasta el hogar (FTTH). Presenta dos tipos de señales, una destinada para la televisión y la otra que es la señal de IP, que es el retorno donde está el protocolo HTTP (HiperText Transfer Protocol).

Se encarga de recibir una señal digital, en alguno de los estándares (cable, satélite, terrestre, IPTV), y de comprobar que se tenga permiso para ver esta señal. Posteriormente lo desmodula y lo envía al televisor. También permite disfrutar de todo el conjunto de ventajas que ofrece la nueva televisión digital, como pueden ser: Acceso condicional, televisión interactiva (MHP) o la televisión en alta definición.

Actualmente un STB puede ofrecer muchos servicios, desde utilizarlo como grabador (DVR) en los STB que incorporen disco duro, como utilizarlos para hacer consultas meteorológicas, hacer la reserva de una visita médica, o hacer compras en los que disponen de interactividad. También muchos de ellos nos dan la opción de conectarles dispositivos externos como podrían ser videocámaras, impresoras, entre otros.

3.1 ESQUEMA DE BLOQUES DE UN SET-TOP BOX

Los pasos que sigue un STB en su funcionamiento, son los siguientes:

Lo primero que hace es sintonizar una señal digital, la cual incluirá tanto información de video (MPEG2, o MPEG4 para señales en alta definición), información de audio e información de datos (DVB-SI).

El siguiente paso es separar los tres tipos de información que recibimos para tratarlos por separado.

A continuación, el sistema de acceso condicional decidirá cuales son los permisos que tiene el suscriptor para poder ver los contenidos que está recibiendo. Si tiene permiso descifrára esa información. Una vez descifrados, los paquetes de video y audio son enviados al televisor.

Los paquetes de datos que hemos recibido junto con los de video y audio, se ejecutarán en caso de ser necesarios o solicitados por el consumidor.

El STB puede poseer un canal de retorno por donde enviar datos a la cabecera (Back Channel).

3.2 ARQUITECTURA SET-TOP BOX

Para poder ejecutar los datos o programas descargados de la señal de datos, se necesitan una serie de elementos. Estos se pueden describir por el siguiente esquema de capas muy parecido al de un computador.

- Capa de Hardware: Son todos los componentes físicos que forman un STB (CPU, Memoria, acceso condicional, decodificador MPEG⁹).
- Sistema operativo: Al igual que en un computador, un STB también necesita de un sistema operativo para su funcionamiento. La diferencia básica sería en que un STB, necesita de un sistema operativo en tiempo real, ya que, operaciones como la decodificación MPEG necesitan que se realicen al instante. Algunos ejemplos de sistema operativo serian: Linux, Windows CE o Psos.
- La plataforma o Middleware: Se trata de una capa intermedia entre la capa hardware y la capa de software. Se trata de un conjunto de módulos que permiten un desarrollo más eficiente de las aplicaciones. El middleware proporciona un API (Application Programing Interface) para cada uno de los tipos de programación que soporta. De los diferentes lenguajes de programación que

⁹ Ver anexo, glosario de términos, pág. 80

puede soportar un STB, el que sería más destacable, sería DVB-J (DVB-Java), que es utilizado para las aplicaciones interactivas (MHP).

- Capa de aplicaciones: Aquí es donde encontraremos las aplicaciones, que una vez descargadas se podrán ejecutar (algunas aplicaciones podrían ser: EPG, anuncios interactivos, entre otras). A diferencia de las demás capas, este no debe de estar operativa en todo momento, pues simplemente se ejecutará cuando el consumidor lo solicite.

3.3 CARACTERÍSTICAS COMUNES DE UN SET-TOP BOX



Figura N° 5: Set Top Box.

Las características más usuales de un STB son:

- Procesador 32 bits a 125MHz.
- Un mínimo de 16MB de memoria RAM.
- 2KB de memoria EEPROM.
- 8 MB de memoria Flash.
- Mando a distancia.
- Disco duro (no todos disponen del mismo, y su capacidad oscila entre los 20GB y los 250GB).
- Canal de retorno (módem interno, tarjeta Ethernet, etc.)
- Ranura de lectura de Smart Cards.

Las aplicaciones interactivas se dividen en dos grandes categorías:

- 1) Las que utilizan interactividad primaria, vale decir que el usuario interactúa con la aplicación mediante el control remoto, sin recibir o enviar información hacia el exterior ya que no usan una conexión a la banda ancha. Ejemplo: un juego, ver en pantalla información complementaria que incluye la misma aplicación ya sean textos o imágenes.

2) Las aplicaciones interactivas avanzadas, son las que se desarrollan teniendo en cuenta que el STB (o el TV con sintonizador digital) tiene una conexión a Internet de banda ancha. Entonces es posible que por ella se reciban y envíen datos hacia el exterior. Como ejemplo podemos citar ver el pronóstico del tiempo en cualquier momento; hacer que el anunciante de un producto que nos interesa nos llame por teléfono ó establecer con él una sesión de chat o enviarle un SMS. También, participar en encuestas en línea, trivias, concursos, etc.

CAPÍTULO 4: METODOLOGÍA

4.1 DESARROLLO ÁGIL DE SOFTWARE

El desarrollo ágil de software son métodos de ingeniería del software basado en el desarrollo iterativo e incremental, donde los requerimientos y soluciones evolucionan mediante la colaboración de grupos auto organizado y multidisciplinario. Existen muchos métodos de desarrollo ágil; la mayoría minimiza riesgos desarrollando software en lapsos cortos. El software desarrollado en una unidad de tiempo es llamado una iteración, la cual debe durar de una a cuatro semanas. Cada iteración del ciclo de vida incluye: planificación, análisis de requerimientos, diseño, codificación, revisión y documentación. Una iteración no debe agregar demasiada funcionalidad para justificar el lanzamiento del producto al mercado, pero la meta es tener una «demo» (sin errores) al final de cada iteración. Al final de cada iteración el equipo vuelve a evaluar las prioridades del proyecto.

Los métodos ágiles enfatizan las comunicaciones cara a cara en vez de la documentación. La mayoría de los equipos ágiles están localizados en una simple oficina abierta, a veces llamadas "plataformas de lanzamiento". La oficina debe incluir revisores, escritores de documentación y ayuda, diseñadores de iteración y directores de proyecto. Los métodos ágiles también enfatizan que el software funcional es la primera medida del progreso. Combinado con la preferencia por las

comunicaciones cara a cara, generalmente los métodos ágiles son criticados y tratados como "indisciplinados" por la falta de documentación técnica.

La definición moderna de desarrollo ágil de software evolucionó a mediados de la década de 1990 como parte de una reacción contra los métodos muy estructurados y estrictos, extraídos del modelo de desarrollo en cascada. El proceso originado del uso del modelo en cascada era visto como burocrático, lento, degradante e inconsistente con las formas de desarrollo de software que realmente realizaban un trabajo eficiente.

Los métodos de desarrollos ágiles e iterativos pueden ser vistos como un retroceso a las prácticas observadas en los primeros años del desarrollo de software (aunque en ese tiempo no había metodologías formales). Inicialmente, los métodos ágiles fueron llamados métodos de "peso liviano".

En el año 2001, miembros prominentes de la comunidad se reunieron en Snowbird, Utah, y adoptaron el nombre de "métodos ágiles". Poco después, algunas de estas personas formaron la "alianza ágil", una organización sin fines de lucro que promueve el desarrollo ágil de aplicaciones. Muchos métodos similares al ágil fueron creados antes del

2000. Entre los más notables se encuentran: Scrum (1986), Crystal Clear (cristal transparente), programación extrema (en inglés eXtreme Programming o XP, 1996), desarrollo de software adaptativo, feature driven development, Método de desarrollo de sistemas dinámicos (en inglés Dynamic Systems Development Method o DSDM, 1995).

Kent Beck creó el método de Programación Extrema (usualmente conocida como XP) en 1996 como una forma de rescatar el proyecto del Sistema exhaustivo de compensaciones de Chrysler (C3). Mientras Chrysler cancelaba ese proyecto, el método fue refinado por Ron Jeffries.

Métodos ágiles

Algunos métodos ágiles de desarrollo de software:

- Adaptive Software Development (ASD).
- Agile Unified Process (AUP).
- Crystal Clear.
- Essential Unified Process (EssUP).
- Feature Driven Development (FDD).
- Lean Software Development (LSD).
- Kanban.
- Open Unified Process (OpenUP).

- Programación Extrema (XP).
- Método de desarrollo de sistemas dinámicos (DSDM).
- Scrum.

4.2 FEATURE DRIVER DEVELOPMENT

- Es un proceso ágil para el desarrollo de sistemas.
- Fue diseñado por Peter Coad, Eric Lefebvre y Jeff DeLuca.
- No hace énfasis en la obtención de los requerimientos sino en cómo se realizan las fases de diseño y construcción.
- Se preocupa por la calidad, por lo que incluye un monitoreo constante del proyecto
- Se basa en un proceso iterativo con iteraciones cortas que producen un software funcional que el cliente y la dirección de la empresa pueden ver y monitorear.
- Define claramente entregas tangibles y formas de evaluación del progreso del proyecto.

4.2.1 PROCESO

El proceso consiste de cinco pasos secuenciales durante los cuales se diseña y se construye el sistema (ver figura N° 6):

- Desarrollo de un modelo global.
- Construcción de una lista de funcionalidades.
- Planeación por funcionalidad.
- Diseño por funcionalidad.
- Construcción por funcionalidad.

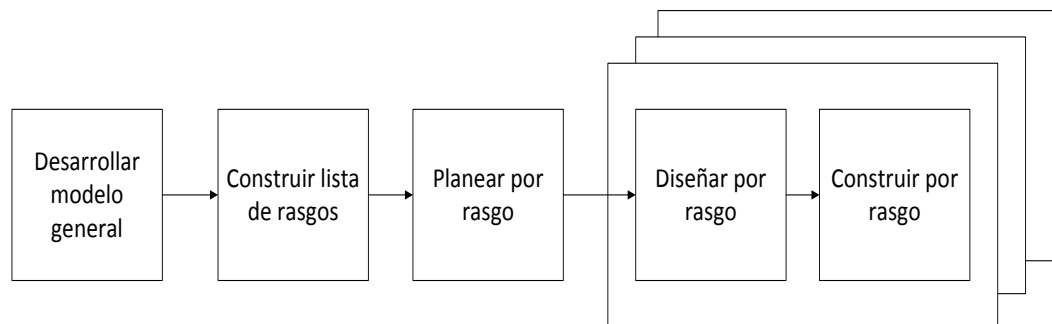


Figura N° 6: Procesos de la metodología FDD.

Desarrollo de un modelo global

Cuando comienza el desarrollo, los expertos del dominio están al tanto de la visión, el contexto y los requerimientos del sistema a construir.

Se divide el dominio global en áreas que son analizadas detalladamente.

Los desarrolladores construyen un diagrama de clases o de objetos por cada área.

Se construye un modelo global del sistema.

Construcción de una lista de funcionalidades

Una funcionalidad es un ítem útil a los ojos del cliente.

Se elabora una lista de funcionalidades que resuma la funcionalidad general del sistema.

La lista es elaborada por los desarrolladores y es evaluada por el cliente.

Se divide la lista en subconjuntos según la afinidad y la dependencia de las funcionalidades.

La lista es finalmente revisada por los usuarios y los responsables para su validación y aprobación.

Planeación por funcionalidad

En este punto se procede a ordenar los conjuntos de funcionalidades conforme a su prioridad y dependencia, y se asigna a los programadores jefes.

4.3 ROLES Y RESPONSABILIDADES

Categorías:

- Key Roles / Roles claves
- Supporting Roles / Roles de soporte
- Additional Roles / Roles adicionales

4.3.1 KEY ROLES / ROLES CLAVES

Project Manager / Director del Proyecto:

- * Líder administrativo y financiero del proyecto.
- * Protege al equipo de situaciones externas.

Chief Architect / Arquitecto jefe:

- * Diseño global del sistema.
- * Ejecución de todas las etapas.

Development Manager / Director de desarrollo

- * Lleva diariamente las actividades de desarrollo.
- * Resuelve conflictos en el equipo.
- * Resuelve problemas referentes a recursos.

Chief Programmer / Programador Jefe

- * Analiza los requerimientos.
- * Diseña el proyecto.
- * Selecciona las funcionalidades a desarrollar de la última fase del FDD.

Class Owner / Propietario de clases

- * Responsable del desarrollo de las clases que se le asignaron como propias.
- * Participa en la decisión de que clase será incluida en la lista de funcionalidades de la próxima iteración.

Expertos de dominio

- * Puede ser un usuario, un cliente, analista o una mezcla de éstos.
- * Poseen el conocimiento de los requerimientos del sistema.
- * Pasa el conocimiento a los desarrolladores para que se asegure la entrega de un sistema completo.

4.3.2 SUPPORTING ROLES / ROLES DE SOPORTE

Domain Manager

- * Lidera al grupo de expertos del dominio.
- * Resuelve sus diferencias de opinión concernientes a los requerimientos del sistema.

Release Manager

- * Controla el avance del proceso mediante la revisión de los reportes del Chief Programmer.
- * Reporta resultados obtenidos semanalmente al gerente, al cliente donde incluye el porcentaje de avance de cada feature.

Language Lawyer / Guru del Lenguaje

- * Responsable de poseer un vasto conocimiento en, por ejemplo, un lenguaje específico de programación o tecnología.
- * Es muy importante cuando se trabaja una nueva tecnología.

Build Engineer / Ingeniero de construcción

- * Responsable de preparar, mantener y correr el proceso de construcción.
- * Realiza el mantenimiento de las versiones y la publicación de la documentación.

Toolsmith / Herramentista

- * Rol para la construcción de herramientas específicas para el desarrollo, conversión de datos y testeo.

- * Puede trabajar en la preparación y mantenimiento tanto de bases de datos o sitios web destinados al proyecto.

System Administrator / Administrador del sistema

- * Configura, administra y repara los servidores, estaciones de trabajo y equipos de desarrollo y testeo utilizados por el equipo.

4.3.3 ADDITIONAL ROLES / ROLES ADICIONALES

Tester

- * Verifica que el sistema recién creado cumpla con los requerimientos del cliente.

- * Puede llegar a ser una persona independiente del equipo del proyecto.

Deployer

- * Es el encargado de convertir la información existente requerida por el nuevo sistema.

- * Participa en el lanzamiento de los nuevos productos.

Technical Writer / Escritores de documentos técnicos

* Prepara la documentación para los usuarios, que pueden formar parte o no del equipo del proyecto.

4.4 COMPARACIONES RESPECTO A OTRAS METODOLOGÍAS ÁGILES

Puesto que todos los procesos se centran en la producción de software es deseable una comparación, no en su conjunto, sino según los medios que emplean y sus resultados. Por lo que se puede realizar una comparación entre FDD, RUP y XP, según los siguientes criterios:

Criterio	Análisis
Tamaño de los equipos	RUP está pensado para proyectos y equipos grandes, en cuanto a tamaño y duración. FDD y XP se implementan mejor para proyectos cortos y equipos más pequeños, siendo quizás FDD más escalable que XP.
Obtención de requisitos	RUP y XP crean como base UseCases y UserStories, por lo contrario FDD no define explícitamente esa parte del proyecto sobre la adquisición de requisitos.
Evaluación del estado del proyecto	FDD es posiblemente el proceso más adecuado para definir métricas que definan el estado del proyecto, puesto que al dividirlos en unidades pequeñas es bastante sencillo hacer un seguimiento de las

	<p>mismas.</p> <p>XP también define esos componentes pequeños. RUP por su parte, es tan grande y complejo en este sentido como en el resto, por lo que manejar el volumen de información que puede generar requiere mucho tiempo.</p>
Carga de trabajo	<p>XP es un proceso ligero, esto es, que los creadores del proceso han tenido cuidado de no poner demasiadas tareas organizativas sobre los desarrolladores. RUP es un proceso pesado, basado mucho en la documentación, en la que no son deseables todos esos cambios volátiles. FDD es por su parte un proceso intermedio, en el sentido de que genera más documentación que XP pero menos que RUP.</p>
Relación con el cliente	<p>Con RUP se presentarán al cliente los artefactos del final de una fase, en contrapartida, la aseguración de la calidad en XP y FDD no se basa en formalismos en la documentación, si no en controles propios y una comunicación fluida con el cliente.</p>
Conocimiento sobre la arquitectura	<p>En RUP se intentará reducir la complejidad del software a producir a través de una planificación intensiva. En XP se conseguirá a través de la programación a pares que ya en la creación del código se puedan evitar errores y malos diseños. En FDD sin embargo se usan las sesiones de trabajo conjuntas en fase de diseño para conseguir una arquitectura sencilla y sin errores.</p>

Figura N° 7: Metodología FDD versus otras metodologías ágiles.

4.5 VENTAJAS

- El equipo de desarrollo no malgasta el tiempo y dinero del cliente desarrollando soluciones innecesariamente generales y complejas que en realidad no son un requisito del cliente.
- Cada componente del producto final ha sido probado y satisface los requerimientos.
- Rápida respuesta a cambios de requisitos a lo largo del desarrollo.
- Entrega continua y en plazos cortos de software funcional.
- Trabajo conjunto entre el cliente y el equipo de desarrollo.
- Minimiza los costos frente a cambios.
- Importancia de la simplicidad, al eliminar el trabajo innecesario.
- Atención constante a la excelencia técnica y al buen diseño.
- Mejora continua de los procesos y el equipo de desarrollo.
- Evita malentendidos de requerimientos entre el cliente y el equipo.

4.6 DESVENTAJAS

- FDD presenta su principal debilidad en la necesidad de tener en el equipo miembros con experiencia que marquen el camino a seguir desde el principio, con la elaboración del modelo global, puesto que no es tan ágil como podría serlo XP.
- Falta de documentación del diseño. El código no puede tomarse como una documentación. En sistemas de tamaño grande se necesitan leer los cientos o miles de páginas del listado de código fuente.
- Problemas derivados de la comunicación oral. Este tipo de comunicación resulta difícil de preservar cuando pasa el tiempo y está sujeta a muchas ambigüedades.
- Fuerte dependencia de las personas. Como se evita en lo posible la documentación y los diseños convencionales, los proyectos ágiles dependen críticamente de las personas.
- Falta de reusabilidad. La falta de documentación hacen difícil que pueda reutilizarse el código ágil.

CAPÍTULO 5: CONSTRUCCIÓN

Para poder llevar a cabo esta etapa, se debieron utilizar las siguientes funciones:

5.1 BARRA DE BÚSQUEDA

Es el formulario que permite interactuar con las funciones de la aplicación, está diseñada con HTML y CSS, esta barra de búsqueda permite al usuario ingresar la solicitud del lugar que desea buscar, el cual puede ser una dirección en particular, una ciudad, un país, entre otros. Esta petición por parte del usuario se envía al pulsar el botón OK para el script `codeAddress()` implementado en Javascript.

Esta función recibe la solicitud que envía el usuario, con lo cual Google Maps busca dentro de su base de datos la ubicación, vale decir, las coordenadas geográficas (latitud y longitud) más apropiadas de acuerdo a la dirección requerida por el usuario.

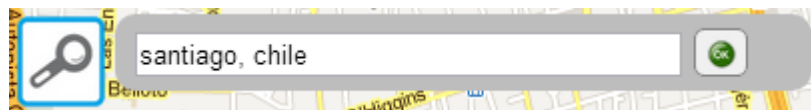


Figura N° 8: Barra de búsqueda.

5.2 WEB SERVICE

Servicio implementado en la aplicación que mediante la función `xsystem.hwMacAddr` que provee XAVi, que es el fabricante del STB y que permite obtener la MAC del cliente, para extraer la información del cliente, específicamente la dirección de él y geo-referenciar al inicio de la aplicación de Google Maps.

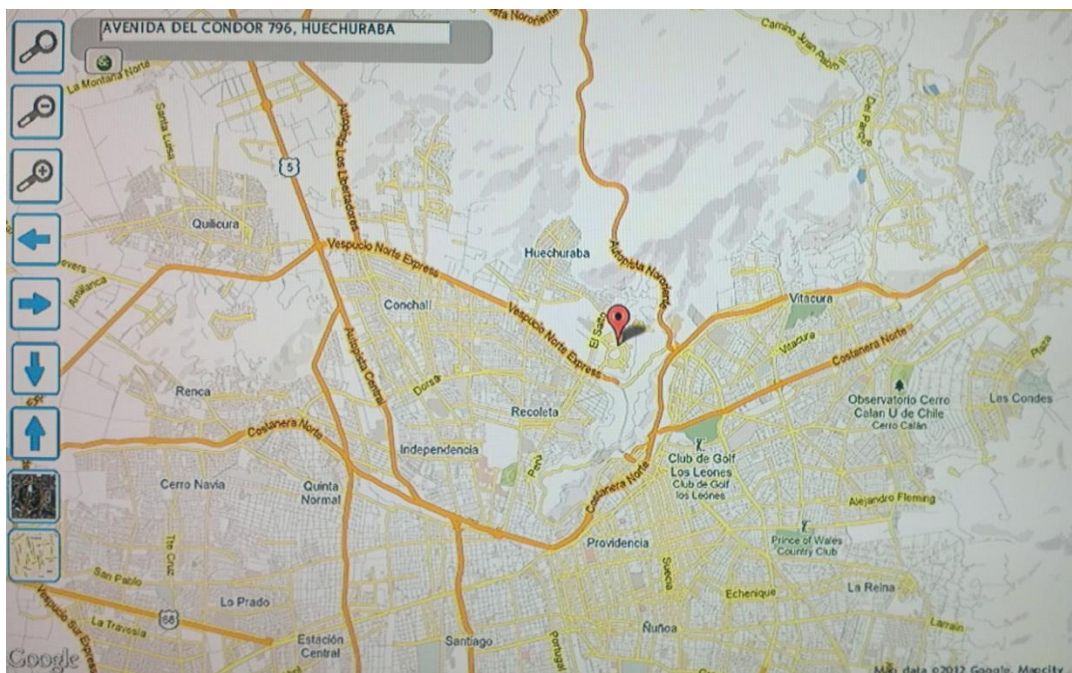


Figura N° 9: Funcionamiento de Web Service.

En la figura N° 9, se aprecia la geo-referenciación de la dirección del cliente, al obtener la MAC del decodificador (STB) asociado al cliente, este procedimiento se realiza mediante un WSDL¹⁰ que es un enlace

¹⁰ Ver anexo, glosario de términos, pág. 85

determinado, el cual cumple una función determinada en el servidor y que al consumirlo en un programa SOAP¹¹ se acceden a los datos del cliente en un formato XML¹², de los cuales se extraen la dirección, el número y la comuna.

5.3 API DE GOOGLE MAPS

Previo a definir las funciones que se van a utilizar en conjunto con el API de Google Maps, se debe tener en cuenta:

- Incluir el código JavaScript del API de Google Maps mediante la etiqueta script. Como se muestra a continuación en la figura N° 10:

```
<html>
  <head>
    <script type="text/javascript"
      src="http://maps.googleapis.com/maps/api/js?key=YOUR_API_KEY&sensor=SET_TO_TRUE_OR_FALSE">
    </script>
```

Figura N° 10: Script para cargar API Google Maps.

¹¹ Ver anexo, glosario de términos, pág. 81

¹² Ver anexo, glosario de términos, pág. 85

- Crear un elemento div denominado “map_canvas” que aloja el mapa. En la figura N° 11 se muestra el estilo CSS del div:

```
<style type="text/css">
  html { height: 100% }
  body { height: 100%; margin: 0; padding: 0 }
  #map_canvas { height: 100% }
</style>
```

Figura N° 11: Estilo CSS del div map_canvas.

- Crear un objeto JavaScript literal para alojar una serie de propiedades de mapa. Escribir una función JavaScript para crear un objeto de mapa. (Ver figura N° 12).

```
function initialize() {
  var mapOptions = {
    zoom: 8,
    center: new google.maps.LatLng(-34.397, 150.644),
    mapTypeId: google.maps.MapTypeId.ROADMAP
  }
  var map = new
google.maps.Map(document.getElementById("map_canvas"),
mapOptions);
}
```

Figura N° 12: Características por defecto del mapa.

- Inicializar el objeto de mapa desde el evento onload de la etiqueta body. Como se muestra en la siguiente figura N° 13.

```
<body onload="initialize()">
```

Figura N° 13: Script para cargar el objeto de mapa.

Funciones para interactuar con Google Maps:

- `codeAddress()`: script en el cual se obtiene la dirección del formulario, después, se realiza la llamada para el objeto geocoder y se verifica el estado de esta llamada, de ser exitoso, se realiza la actualización de la ubicación y se coloca el marcador en esa posición del mapa.
- `code_Address(dir)`: realiza las mismas acciones que la función descrita en el punto anterior, pero en este caso recibe como parámetro la dirección recibida por el Web Service.

Funciones asociada a íconos (Figura N° 14):



Figura N° 14: Funciones de setZoom(), panBy y setMapTypeld.

- getZoom(): obtiene el nivel de zoom del mapa.
- setZoom(): modifica el nivel de zoom para el mapa, el nivel de zoom está determinado desde el mínimo que es el nivel 1 hasta el máximo que es el nivel 23.

La combinación de estas dos funciones permite ver el mapa con mayor o menor detalle, por lo cual se usan dos botones, uno para

aumentar y otro para disminuir el zoom del mapa. Para poder aumentar o disminuir el zoom, se obtiene el nivel de zoom con la función `getZoom()`, que retorna un valor de entre 1 y 23, que son los límites del nivel de zoom para el mapa.

- `setMapTypeId()`: permite modificar el tipo de mapa que se quiera mostrar en la aplicación, en este caso los tipos son ROADMAP o HYBRID. Para ello se usan dos botones distintos, uno para el tipo HYBRID y otro para el tipo ROADMAP.
- `panBy()`: especifica la dirección y distancia del desplazamiento a través del mapa, es decir, cuanto espacio para moverse hacia arriba, abajo, izquierda o derecha. Para llevar a cabo esta función se disponen de cuatro botones (que tienen como imagen una flecha), que indican hacia qué dirección del mapa desea moverse el usuario.

5.4 DIAGRAMAS USADOS PARA LA CONSTRUCCIÓN DEL SISTEMA

5.4.1 ESQUEMA GENERAL DE LA APLICACIÓN

A continuación en la siguiente figura N° 15, se representa el sistema desde un punto de vista general, donde están las distintas interfaces, procesos y bases de datos que participan en el sistema.

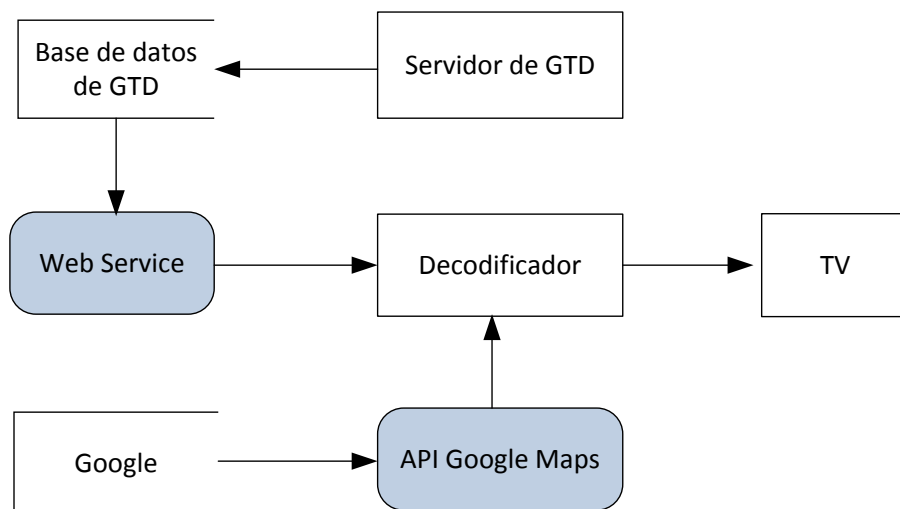


Figura N° 15: Esquema global de la aplicación.

5.4.2 ESQUEMA DEL USUARIO

En la figura N° 16, se representa como el usuario interactúa con la aplicación de Google Maps mediante las distintas interfaces de comunicación como la interfaz del programa y la de la televisión usando para ello el decodificador.

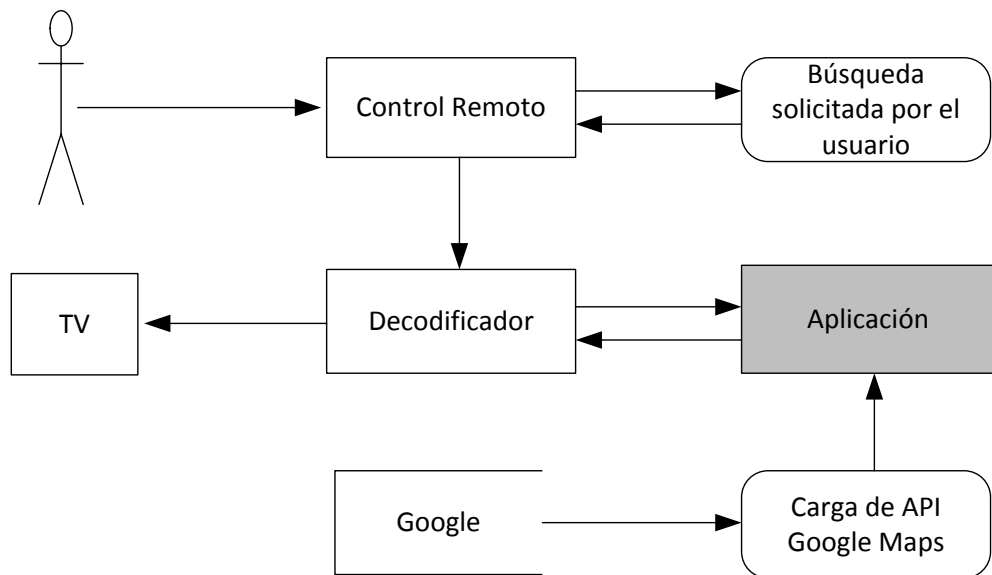


Figura N° 16: Representación del sistema para el usuario final.

5.4.3 ESQUEMA DEL DECODIFICADOR

En la figura N° 17, se representa como el decodificador se relaciona con la aplicación mediante el navegador web llamado Galeon que se caracteriza por ser un navegador libre basado en el motor de renderizado Gecko que es el mismo que utiliza Mozilla Firefox. Otras de sus características son: posee entrada de video y HDMI, usa cable coaxial para la conexión de los servicios contratados por el cliente, además de tener un disco duro de 320 GB para guardar series de televisión, películas, etc.

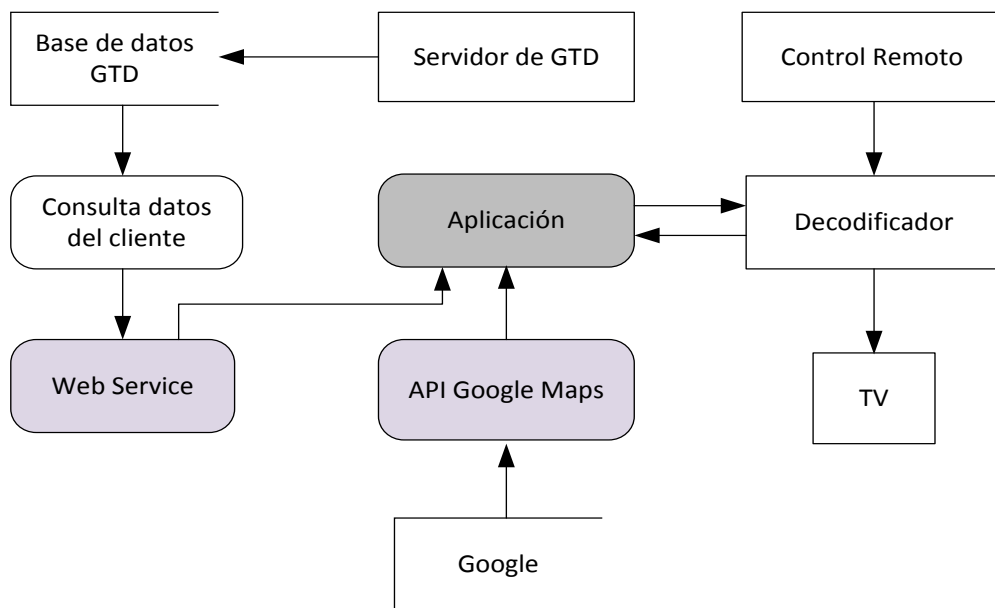


Figura N° 17: Aplicación desde el punto de vista del decodificador.

Como resultado final queda una aplicación, principalmente porque el usuario puede hacer una búsqueda de cualquier dirección a través de Google Maps.

CAPÍTULO 6: PRUEBAS

A continuación se detallan los tipos de pruebas realizadas en conjunto con el desarrollo del sistema de Google Maps.

6.1 PRUEBAS REALIZADAS EN EL COMPUTADOR

Este tipo de pruebas se hicieron para ver cómo iba el funcionamiento de la aplicación previo a realizar una prueba en el decodificador, con este tipo de pruebas se podían tener los errores de codificación o problemas con el diseño de la interfaz gráfica y posteriormente proceder a corregirlos.

Las pruebas se realizaron en el navegador web Mozilla Firefox, usando un complemento llamado “Firebug”, con el cual se podían saber los errores de sintaxis que podía tener algún script o ver cuáles son las modificaciones que se deben hacer al diseño web definido en el código CSS de la página web.

En la figura N°18, el error que se aprecia de que la función `xsystem.hwMacAddr` no está definida, ocurre porque sólo se encuentra disponible para ser utilizada en el decodificador y no en un computador, ya que es una función implementada por el fabricante XAVi en Javascript para el STB.

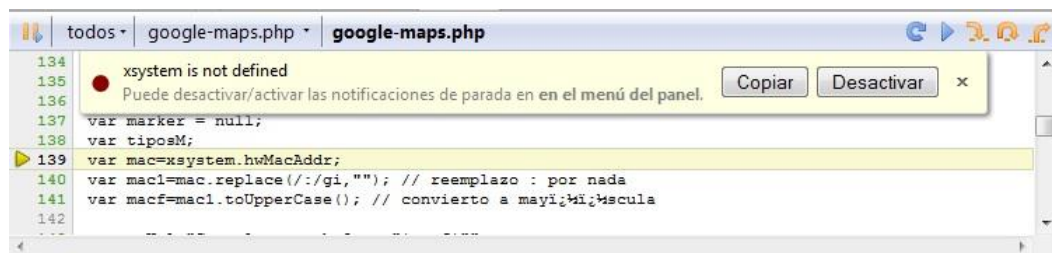


Figura N° 18: Error de la función xsystem.

6.2 PRUEBAS REALIZADAS EN EL DECODIFICADOR

Con estas pruebas realizadas después de haber hechos las pruebas respectivas en un computador, se podían observar que detalles a la interfaz gráfica o algún error de codificación que no fuera detectado en las pruebas realizadas previamente.

Errores detectados en log del decodificador en las pruebas realizadas, que después fueron corregidos una vez finalizada las pruebas a la aplicación. Estos errores se encontraron en algunos script realizados en JavaScript que presentaban algún error de sintaxis al ver el log del STB:

- (none) local1.err syslog: [JS] script error: Attempt to convert null or undefined value latLng to Object
at http://172.25.24.50/googlemaps/google-maps.php:Line 338 Col 8

- (none) local1.info syslog: [HTTP]
open http://maps.gstatic.com/cat_js/intl/en_us/mapfiles/api-3/9/19/%7Bcommon,util,geocoder,map,marker%7D.js
- (none) local1.err syslog: [JS] script error: b.clearAttributes() is not callable
athttps://ajax.googleapis.com/ajax/libs/jquery/1.5.2/jquery.min.js:Line 16 Col 2924
- (none) local1.err syslog: [JS] script error:
this.b.insertAdjacentHTML() is not callable at internal://4257/:Line 33 Col 315

En las pruebas realizadas, no fue necesario el realizar un método o una función de validación para los datos que ingrese el usuario, ya que la API de Google Maps presenta un conjunto de estados, por lo cual, si el usuario ingresa una dirección no válida devuelve el valor ZERO_RESULTS, como se muestra en la figura N° 19.



Figura N° 19: Alerta cuando se ingresa una dirección inexistente.

CAPÍTULO 7: PUESTA EN MARCHA

Una vez concluido el proceso de construcción, codificación y las pruebas, se procedió con el último paso que es poner el proyecto finalizado en marcha.

El proceso previo a la puesta en marcha fue de dos meses, detallados en la siguiente planificación:

Nombre de tarea	Duración	Comienzo	Fin
Reunión con la empresa GTD para ver tema	1 día	mié 26-09-12	mié 26-09-12
Levantamiento de requerimientos	4 días	mié 26-09-12	lun 01-10-12
Implementación barra de búsqueda	9 días	lun 01-10-12	jue 11-10-12
Asociar barra de búsqueda con las funciones de la API de Google Maps	3 días	lun 15-10-12	mié 17-10-12
Prueba en el computador	1 día	mié 17-10-12	mié 17-10-12
Prueba en el decodificador	1 día	mié 17-10-12	mié 17-10-12
Implementar barra de herramientas (zoom, cambio de mapa, navegar en el mapa)	3 días	jue 18-10-12	lun 22-10-12
Pruebas en el computador	1 día	lun 22-10-12	lun 22-10-12
Pruebas en el decodificador	1 día	lun 22-10-12	lun 22-10-12
Modificaciones al diseño (estilos CSS) de la aplicación	4 días	lun 22-10-12	jue 25-10-12
Implementación de Web Service	4 días	lun 12-11-12	jue 15-11-12
Pruebas en el computador	1 día	lun 19-11-12	lun 19-11-12
Pruebas en el decodificador	1 día	lun 19-11-12	lun 19-11-12
Modificaciones del diseño de la aplicación	4 días	lun 19-11-12	jue 22-11-12
Pruebas en el computador	1 día	lun 26-11-12	lun 26-11-12
Pruebas en el decodificador	1 día	lun 26-11-12	lun 26-11-12

Modificación tamaño barra de búsqueda	1 día	mar 27-11-12	mar 27-11-12
Pruebas en el computador	1 día	mar 27-11-12	mar 27-11-12
Pruebas en el decodificador	1 día	mar 27-11-12	mar 27-11-12
Envío de la aplicación de GTD Imagen al área de producción	1 día	mié 28-11-12	mié 28-11-12
Puesta en marcha del proyecto para los usuarios de televisión	2 días	mié 28-11-12	jue 29-11-12

Figura N° 20: Carta Gantt

Para la puesta en marcha, GTD Imagen realizó las siguientes acciones:

- ✓ Verificar el correcto funcionamiento de la aplicación, vale decir, no tener errores de Javascript, funcionamiento del Web Service y presentar un buen diseño de la página con los estilos CSS, para así poder mandarla al área de producción, que se encarga de gestionar las nuevas aplicaciones desarrolladas para el portal interactivo de GTD Manquehue.

- ✓ Crear un anuncio publicitario para la nueva aplicación en el panel interactivo de información desplegado en la pantalla de la televisión. Invitando a los usuarios de televisión a probar la aplicación de Google Maps.

- ✓ Comprobar que la aplicación de Google Maps fue visitada y utilizada por parte de los clientes que tengan el servicio de televisión de la empresa GTD Manquehue, para ello se llevó un registro de los usuarios que la visitaron mediante el portal interactivo de GTD Manquehue o por el panel interactivo donde se despliega la aplicación de Google Maps desarrollada recientemente, según las consultas hechas a la base de datos de GTD Manquehue fueron alrededor de 5300 usuarios los que usaron la aplicación.

Con la aplicación funcionando para los usuarios de televisión digital, dado que la aplicación al estar en funcionamiento en el STB ocupaba un 99.9% de utilización de memoria RAM, por este motivo se decidió hacer una mantención al desempeño del servicio. Puesto que, es una estrategia de negocio para los clientes que tengan contratado los servicios de televisión, por esta razón en el área de GTD Imagen se optó por realizar las mejoras pertinentes a la aplicación para poder volver a ponerla en marcha.

CONCLUSIONES

De lo expuesto anteriormente, se concluye que:

En lo profesional, cumpliendo el rol de ingeniero para la implementación del proyecto de Google Maps para el área de televisión de la empresa GTD Manquehue, se cumplieron con los objetivos planteados en los requerimientos de la empresa GTD Manquehue, que consistía en implementar un sistema que utilice la API de Google Maps para visualizar mapas en televisión y realizar una búsqueda de dirección, ciudad, país o algún lugar en particular como un museo, hotel, etc. para el usuario usando de por medio del decodificador.

Con la puesta en marcha del proyecto, GTD Manquehue, específicamente del área de televisión avanza con sus estrategias de mercado al presentar una aplicación útil y novedosa exclusivamente para los clientes que tengan contratados los servicios de televisión, sin costos adicionales para los usuarios mientras tengan contratado los servicios de la empresa.

En el ámbito académico, el desarrollo del trabajo de título en un comienzo se hizo difícil, ya que con la inexperiencia laboral junto con el escaso dominio de los lenguajes de programación hicieron dudar si podía llegar a

cumplir con los objetivos y los plazos establecidos para la implementación del proyecto. Aunque, con el paso de los días fue cambiando al ir teniendo avances en la aplicación, funcionamiento y diseño, para poder ir resolviendo las dudas que fueran surgiendo respecto al desarrollo de la aplicación, y así poder cumplir con la planificación.

En el ámbito personal, fue un proceso complejo en un principio, porque el hecho de participar en un proyecto en una gran empresa como GTD Manquehue, influyó en un comienzo en la actitud tímida que tenía en un comienzo, además de la inexperiencia de estar en el mundo laboral. Pero con el transcurso de los días se fue superando esta limitación, lo que permitió llegar al cumplimiento del objetivo.

BIBLIOGRAFÍA

Francesc Tarrés Ruiz, “Sistemas Audiovisuales - TV Analógica y Digital”, Edicions UPC, España, 2000

Gabriel Svennerberg, “Beginning Google Maps API 3”, Apress, United States of America, 2010

Roger S. Pressman, “Ingeniería de Software, Un enfoque práctico”, McGraw Hill, Sexta Edición.

Mary Poppendieck, Tom Poppendieck, “Lean Software Development: An Agile Toolkit”, Addison-Wesley, 2003.

Stephen R. Palmer, John M. Felsing, “A Practical Guide to: Feature-Driven Development”, TogetherSoft Corporation, 2002.

Juan Diego Gauchat, “El gran libro de HTML5, CSS3 y JavaScript”, Marcombo S.A., 2012.

WEBGRAFÍA

GTD Manquehue. Disponible en:

<http://www.gtdmanquehue.com/quienes-somos/historia/> [Consultado el 3/12/2012]

TV Digital y STB's Actuales (Publicado el 13/11/2007). Disponible en:

<http://ctellezg.blogspot.com/2007/11/tv-digital-y-stbs-actuales.html>
[Consultado el 15/11/2012]

Galeon es un navegador libre creado para el proyecto GNOME.
(Publicado el 17/5/2011). Disponible en:

<http://www.linux-es.org/node/1883> [Consultado el 16/11/2012]

Introducción - Versión 3 del API de JavaScript de Google Maps.
Disponible en:

<https://developers.google.com/maps/documentation/javascript/tutorial?hl=es> [Consultado el 16/10/2012]

Servicio de asignación de identificadores geográficos - Versión 3 del API
de JavaScript de Google Maps. Disponible en:

<https://developers.google.com/maps/documentation/javascript/geocoding?hl=es> [Consultado el 16/10/2012]

Google Maps API V3 introducción y primeros pasos (Publicado el 4/4/2011). Disponible en:

<http://www.maestrosdelweb.com/editorial/google-maps-api-v3-introduccion-y-primeros-pasos/> [Consultado el 7/11/2012]

Marcadores, posicionar una imagen en el mapa (Publicado el 13/4/2011).

Disponible en:

<http://www.maestrosdelweb.com/editorial/guia-mapas-marcadores/>
[Consultado el 8/11/2012]

Feature Driven Development (FDD) - Ingeniería de Soporte Lógico.

Disponible en:

<https://sites.google.com/site/ingsoportelogico/home/feature-driven-development-fdd> [Consultado el 5/12/2012]

Metodología FDD – Feature Driven Development / Desarrollo Basado en Funciones (Publicado el 12/7/2012). Disponible en:

<http://metodologiafdd.blogspot.com/2012/06/metodologia-fdd.html>
[Consultado el 5/12/2012]

FDD: Feature Driven Development Desarrollo Basado en Funcionalidades. Disponible en:

https://docs.google.com/viewer?a=v&q=cache:ROdxGx-cvgJ:pisic.unalmed.edu.co/cursos/material/3004582/1/PresentacionFDD.ppt+fdd+metodologia&hl=es&gl=cl&pid=bl&srcid=ADGEESgWMRafA7slyvwzAQSEp58xENbZ529_vl-p-m8hAlr9XueWjbtogae4d_P81g0iWn6xtvjXR9RUIOJI7ILkZ5LB9U_c0IEqvxTd73fsFlpdCexP0-rx0PAxdNmtxtxOHN17Q&sig=AHIEtbQmbrURZKIT8yuvjYzVs50Ma9H_9A

[Consultado el 7/12/2012]

ANEXO

GLOSARIO DE TÉRMINOS

AJAX (Asynchronous JavaScript And XML): es una forma de desarrollo web para crear aplicaciones interactivas. Estas aplicaciones se ejecutan en el cliente (en este caso el navegador de los usuarios), y mantiene comunicación asíncrona con el servidor en segundo plano. De esta forma es posible realizar cambios sobre la misma página sin necesidad de recargarla. Esto significa aumentar la interactividad, velocidad y usabilidad en la misma.

Combina las tecnologías:

- * XHTML (o HTML) y hojas de estilos en cascada (CSS) para el diseño que acompaña a la información.

- * Document Object Model (DOM) accedido con un lenguaje de scripting por parte del usuario, especialmente implementaciones ECMAScript como JavaScript y JScript, para mostrar e interactuar dinámicamente con la información presentada.

* El objeto XMLHttpRequest para intercambiar datos asincrónicamente con el servidor web. En algunos frameworks y en algunas situaciones concretas, se usa un objeto iframe en lugar del XMLHttpRequest para realizar dichos intercambios.

* XML es el formato usado comúnmente para la transferencia de vuelta al servidor, aunque cualquier formato puede funcionar, incluyendo HTML pre-formateado, texto plano, JSON y hasta EBML.

API (Application Programming Interface): Grupo de rutinas (conformando una interfaz) que provee un sistema operativo, una aplicación o una biblioteca, que definen cómo invocar desde un programa un servicio que éstos prestan. En otras palabras, una API representa un interfaz de comunicación entre componentes software.

El software que provee la funcionalidad descrita por una API se dice que es una implementación del API. El API en sí mismo es abstracto, en donde especifica una interfaz y no da detalles de implementación.

Un API a menudo forma parte del SDK (Kit de desarrollo de software).

CSS (Cascade Style Sheet): Hoja de Estilo en Cascada. Conjunto de instrucciones escritas en HTML que definen las apariencias de una página web con el objetivo de que sus estilos se parezcan.

FTTH (Fiber To The Home): Es una tecnología que utiliza cable de fibra óptica y sistemas de distribución ópticos adaptados a esta tecnología para la distribución de servicios avanzados, como telefonía, internet de banda ancha, televisión o videojuegos.

HTML (Hyper Text Mark-up Language): Lenguaje desarrollado por el CERN que sirve para modelar texto y agregarle funciones especiales (por ej. hipervínculos). Es la base para la creación de páginas web tradicionales.

El texto se modela a partir del uso de etiquetas o tags. También se pueden agregar scripts al código fuente HTML (generalmente JavaScript, PHP, etc.).

HTTP (HyperText Transfer Protocol): Protocolo usado para acceder a la Web (WWW). Se encarga de procesar y dar respuestas a las peticiones para visualizar una página web.

Además sirve para el envío de información adicional como el envío de formularios con mensajes, etc.

Luego de finalizada la transacción, HTTP no guarda ninguna información sobre la misma, por lo tanto es considerado un protocolo "sin estado". Para guardar la información entre distintas peticiones, se suelen utilizar cookies o pasos de parámetros.

El protocolo HTTP generalmente utiliza el puerto 80.

El HTTP está basado en el modelo cliente-servidor, en donde un cliente HTTP (un navegador por ejemplo) abre una conexión y realiza una solicitud al servidor. Este responde a la petición con un recurso (texto, gráficos, etc.) o un mensaje de error, y finalmente se cierra la conexión. Uno de los mensajes de error HTTP es el 404: Not found.

MAC (Media Access Control): consiste en un identificador realizado con la combinación de 6 bloques hexadecimales que da como resultado una identificación mediante 48 bits, que se asigna en forma exclusiva a un dispositivo que permite la conexión a la red, o bien a través de una tarjeta de expansión exclusiva para la conectividad, por lo que gracias a esta especificidad se le suele considerar como una dirección física del equipo conectado a la red.

Esta determinación se da por el valor que es conocido como IEEE y representa a los últimos 24 bits de la dirección MAC, mientras que los primeros 24 bits se conocen como Valores de Fabricante, que están predefinidos por valores bajo la normativa del Organizationally Unique Identifier.

Si bien no todos los protocolos de comunicación llevan como reconocimiento una Dirección MAC, lo cierto es que esto no es exclusivamente necesario ya que no precisamente en todas las redes se requiere de un identificador global único como lo es este sistema.

Cada hardware tiene su asignación específica de una dirección MAC, y esta variabilidad está dada en el momento de la fabricación, por lo que se trata de un identificador único en todo el mundo para cada equipo,

mediante un proceso que es conocido en idioma inglés como Burned-In Addresses, que comprende a la elaboración de los 48 bits únicos de la dirección presente en el dispositivo de red.

MPEG (Moving Pictures Expert Group): Grupo de trabajo de ISO/IEC encargado del desarrollo de estándares de codificación de video y audio.

Su primer encuentro fue en Mayo de 1988 en Ottawa (Canadá) con unos pocos miembros. Actualmente incluye más de 350 miembros por reunión provenientes de industrias, universidades e institutos de investigación.

El nombre oficial de MPEG es ISO/IEC JTC1/SC29 WG11.

Algunos de los formatos de compresión que ha estandarizado MPEG son:

- MPEG-1: Estándar inicial para la compresión de video y audio. Usado como estándar en Video CD e incluido en el formato de audio MP3 (Layer 3).
- MPEG-2: Estándar para la transmisión de televisión. Usado para la televisión digital ATSC, DVS y ISDB, señales digitales de televisión por cable, y (con pequeñas modificaciones) para DVD.

- MPEG-3: Originalmente fue diseñado para la televisión de alta definición (HDTV), fue abandonado cuando descubrieron que el MPEG-2 (con extensiones) era suficiente para la HDTV.
- MPEG-4: Expande el MPEG-1 para soportar objetos video/audio, contenido 3D, soporte para Digital Rights Management, y codificación de bajo bitrate. Existen varias versiones, la más importante es la MPEG-4 Part 10 (o Advanced Video Coding o H.264). Es usado en HD-DVD y discos Blu-ray.

SOAP: es un protocolo para el intercambio de mensajes sobre redes de computadoras, generalmente usando HTTP. Está basado en XML, a diferencia de DCOM y CORBA que son binarios; esto facilita la lectura por parte de los humanos, pero también los mensajes resultan más largos y, por lo tanto, considerablemente más lentos de transferir.

Existen múltiples tipos de modelos de mensajes en SOAP pero, por lejos, el más común es el RPC, en donde un nodo de red (el cliente) envía un mensaje de solicitud a otro nodo (el servidor) y el servidor inmediatamente responde el mensaje al cliente.

Los mensajes SOAP, son independientes del sistema operativo, y pueden transportarse en varios protocolos de internet como SMTP, MIME y HTTP.

SOAP al principio significaba Simple Object Access Protocol, luego fue Service Oriented Architecture Protocol, pero actualmente es simplemente SOAP. El acrónimo inicial, fue dejado de lado en la versión 1.2, cuando se volvió una recomendación de la W3C el 24 de junio de 2003, porque su nombre daba a confusión.

SOAP fue diseñado por David Winer, Don Box, Bob Atkinson y Mohsen Al-Ghosein en 1998 con respaldo de Microsoft. Actualmente, la especificación SOAP es mantenida por el XML Protocol Working Group de la W3C.

Web Service: La W3C lo define como un sistema de software diseñado para permitir interoperabilidad máquina a máquina en una red. En general, los servicios web son sólo APIs Web que pueden ser accedidas en una red, como internet, y ejecutadas en un sistema de hosting remoto.

En términos sencillos, un servicio web es cualquier sistema de software diseñado para soportar interacción máquina a máquina sobre una red.

Esta amplia definición abarca múltiples y diferentes sistemas, pero en general "servicio web" suele referir a clientes y servidores que se comunican usando mensajes XML que siguen el estándar SOAP.

En definitiva, permite comunicación entre diferentes máquinas, con diferentes plataformas y entre programas distintos. Esta comunicación se logra a través de la adaptación de diversos estándares abiertos.

El organismo WS-I se creó para mejorar la interoperabilidad entre las distintas implementaciones de los servicios web.

Algunas especificaciones que define el servicio web

Las especificaciones que definen los servicios web son intencionalmente modulares y esto resulta en que no hay ni un solo documento que las contenga a todas ellas. Tampoco no hay un solo conjunto estable de especificaciones.

Ventajas de los servicios web

- Aumenta la interoperabilidad entre programas independientemente de la plataforma en donde están instalados.
- Aumenta la interoperabilidad entre servicios y programas de diferentes compañías y ubicados en diferentes lugares geográficos.
- Fomentan los estándares y protocolos basados en texto, haciendo más fácil acceder y entender su contenido y funcionamiento (pero, en general, produciendo una baja en su rendimiento).
- Al emplear HTTP, pueden utilizar un sistema firewall sin cambiar las reglas de filtrado.

Desventajas de los servicios web

- No son tan desarrollados para realizar transacciones comparadas a otros sistemas como CORBA (Common Object Request Broker Architecture).
- Su rendimiento es bajo comparado con otros sistemas como CORBA, DCOM o RMI, especialmente por el uso de protocolos y estándares basados en texto.

WSDL (Web Services Description Language): Lenguaje basado en XML para describir servicios web. Permite describir la interfaz pública de los servicios web; eso significa que detalla los protocolos y los formatos de los mensajes necesarios para interactuar con los servicios listados en su catálogo. Las operaciones y mensajes que soporta se describen en abstracto y se ligán después al protocolo concreto de red y al formato del mensaje. WSDL se utiliza a menudo junto con SOAP y XML Schema.

Un programa cliente se conecta a un servicio web y puede leer el WSDL, determinando así las funciones disponibles en el servidor.

WSDL fue creado por Microsoft e IBM.

XML (eXtensible Markup Language): Desarrollado por el World Wide Web Consortium (W3C). Su objetivo es conseguir páginas web más semántica. XML separa la estructura del contenido y permite el desarrollo de vocabularios modulares. Se trata de un formato abierto.

Al igual que el HTML, se basa en un texto plano y etiquetas, con la diferencia de que XML definen las etiquetas en función al tipo de dato que

está describiendo y no, como en HTML, a la apariencia final que tendrán en pantalla.

XML (al igual que HTML) deriva de SGML. XML es una simplificación de SGML para aplicaciones de propósito general, como la web semántica.

XML ha sido usado para un gran número de aplicaciones como ser XHTML, RSS, Atom, XML-RPC y SOAP.

Algunas tecnologías para el procesamiento de XML son:

- Document Object Model
- XSL Transformations (XSLT)
- SAX
- VTD-XML
- Streaming Transformations para XML (STX)