

iDashBoardA

pp

Cuadro de Mando Operacional

Ángel Luis Peñalosa Roiz
Aplicaciones multimedia de nueva generación
ITT - Telemática



Introducción

- Se pretende desarrollar una aplicación Ad hoc para la plataforma iOS de Apple.
- Una herramienta de control y gestión bajo una interfaz grafica de usuario, denominada cuadro de mando.
- Una solución de visibilidad para los datos del negocio accesible desde cualquier lugar con conexión a internet.



Objetivos

- Desarrollar la aplicación en Swift, el nuevo lenguaje programación para aplicaciones iOS de Apple.
- Diseñar la aplicación con gráficos vectoriales realizados con Sketch y según la Graphical User Interface predefinida por Apple.
- Buscar flexibilidad con la carga de datos desde un fichero XML en la nube, desacoplándola de cualquier modelo de datos existente.
- Transformar los datos en información y esta en conocimiento para el negocio.



Diseño e Implementación

Las fases empleadas en el desarrollo de la aplicación han sido las siguientes:

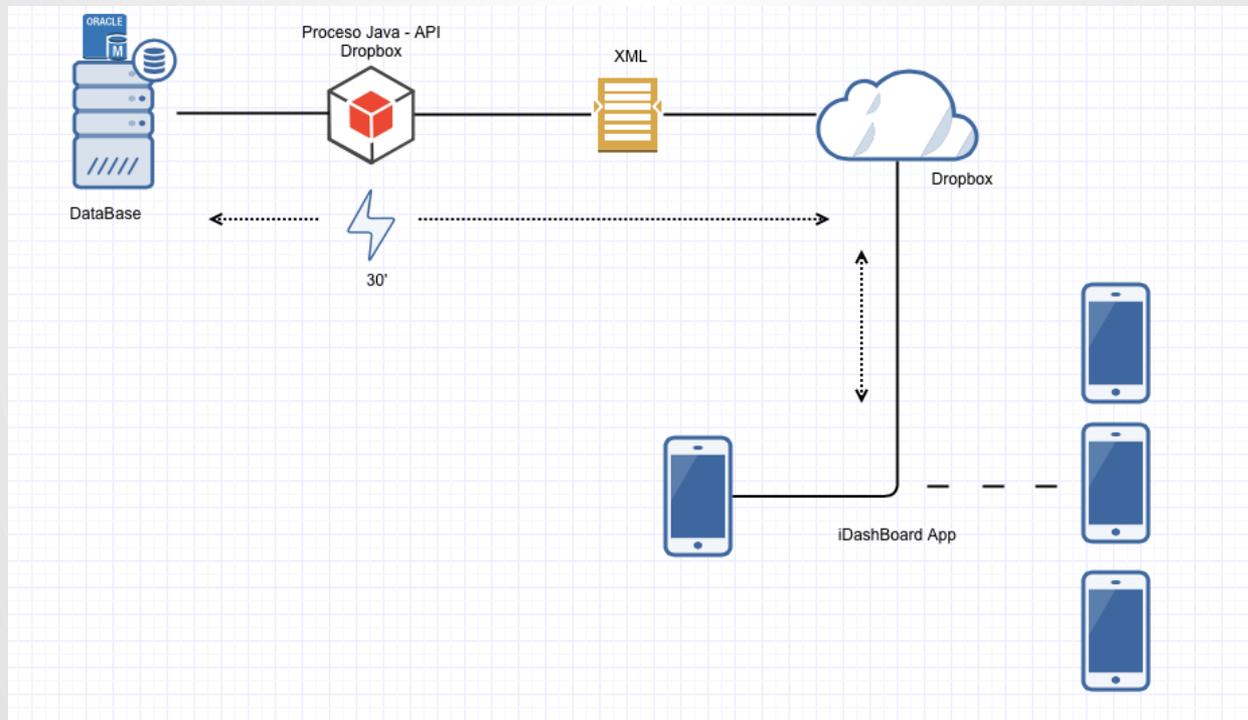
- ✧ **Estructura.** Estructura de los elementos y contenidos de la App, definición del fichero XML.
- ✧ **Diseño Interfaz.** Diseño gráfico de la aplicación y navegabilidad.
- ✧ **Implementación.** Desarrollo del código fuente de la aplicación.

✧ Estructura

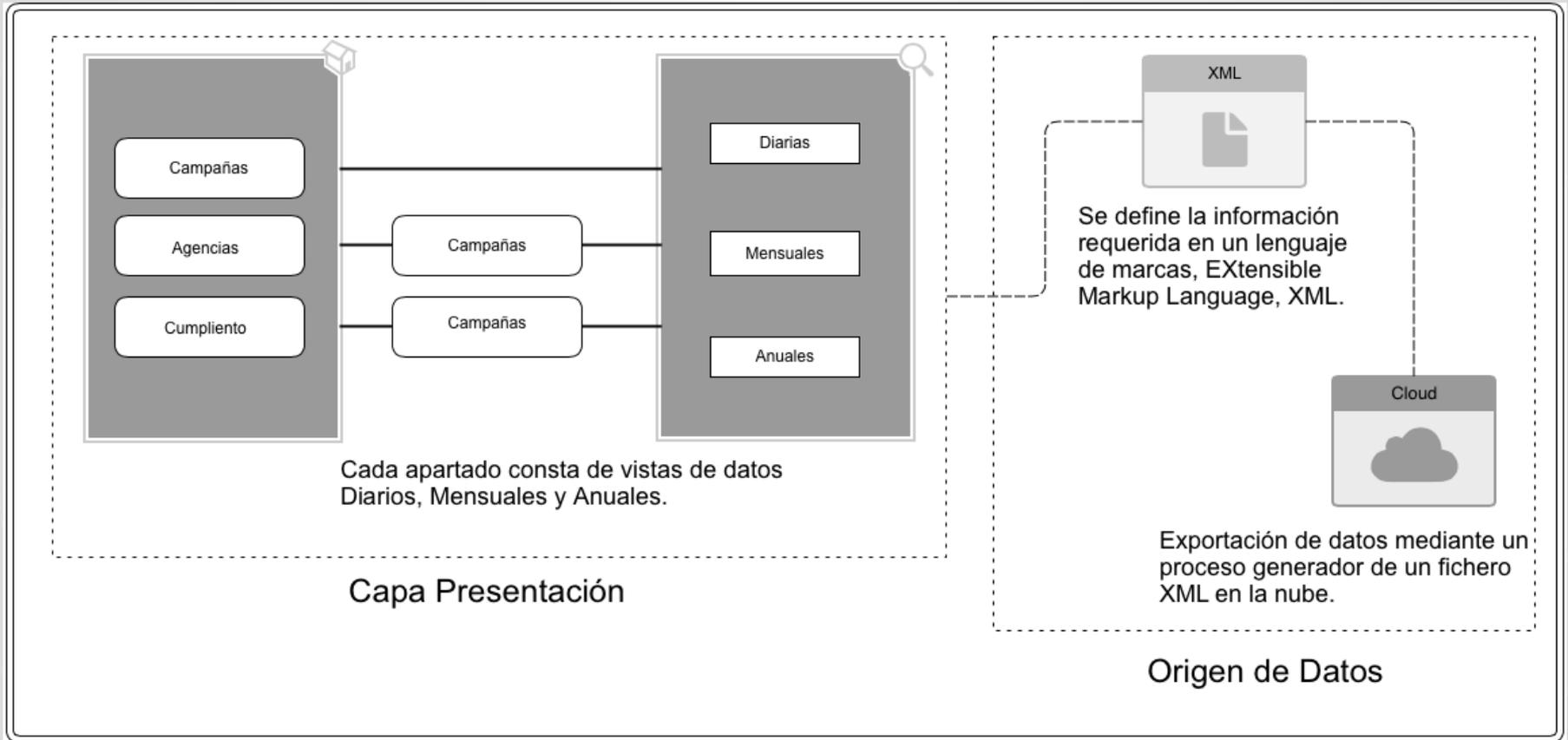
- Organización de la aplicación partiendo de un entorno empresarial comercial, modelo Ad Hoc.
- Está compuesta de los siguientes elementos.
 - Campañas. Selección de campañas comerciales lanzadas por la empresa para captación de clientes.
 - Agencias. Selección de agencias comerciales que explotan comercialmente las campañas ofertadas.
 - Cumplimiento. Objetivos y cumplimientos de ventas por campaña.

- Esquema Arquitectura Global

- Un proceso local generará un fichero de datos XML periódicamente que lo subirá a la nube.
- La App consumirá del fichero para el tratamiento de la información.



- Esquema de carga y muestra de datos



•Ofrece la información detallada y relevante del negocio para facilitar la toma de decisiones.

- Definición Datos App-XML
 - Al estar desacoplado de un modelo de datos específico (bases de datos), el origen de datos se realiza mediante la carga de datos realizando un parseo del fichero XML definido por ambas partes.
 - Se muestran los Structs definidos para el tratamiento de datos.

Campana
nombre: NSString
ventas: NSString
colorVentas: NSString
ventasHora: NSString
colorVentasHora: NSString
conversion: NSString
colorConversion: NSString
eurVenta: NSString
colorCoste: NSString
contactosHora: NSString
colorContactosH: NSString
objetivosVentas: NSString
cumplimientoVentas: NSString

AgenciaCampana
nombre: NSString
ventas: NSString
colorVentas: NSString
ventasHora: NSString
colorVentasHora: NSString
conversion: NSString
colorConversion: NSString
eurVenta: NSString
colorCoste: NSString
contactosHora: NSString
colorContactosH: NSString
productividad: NSString
colorProductividad: NSString
asesores: NSString

Objetivo
nombre: NSString
ventas: NSString
objetivoVentas: NSString
cumplimientoVentas: NSString
ventasHora: NSString
objetivoVentasHora: NSString
cumplimientoVentasHora: NSString
conversion: NSString
objetivoConversion: NSString
cumplimientoConversion: NSString
euroVenta: NSString
objetivoEuroVenta: NSString
cumplimientoEuroVenta: NSString

Agencia
nombre: NSString
postsAgenCamp: [AgenciaCampana] = []

✧ Diseño Interfaz

- Se realiza un skin específico a partir del editor de imágenes vectoriales Sketch.
- Se emplean técnicas de navegabilidad mediante un menú principal y deslizamiento lateral en ambos sentidos.
- Interfaz limpia e intuitiva que permita visualizar la información rápidamente.



- Screencast de Navegabilidad

[Visualización de Video]



✧ Implementación

- Principales objetos que controlan y configuran los rasgos de la aplicación.

- AppDelegate.swift

Se especifica determinadas características y acciones que queremos estén presentes en el conjunto de la App.

- Visual Controllers (VC)

Determinan el comportamiento de las clases que se configuran en la parte visual. Cada componente tiene su parte visual relacionada con su parte de código.

MenuScreenViewController – Es la pantalla del menú principal y cada elemento está definido dentro del correspondiente Storyboard.

- **Menu (MenuTableViewCell.swift)**

Se utiliza para crear tablas con un aspecto personalizado, que difiere del establecido por defecto en iOS.

- **Situación (ViewController y PageViewController)**

Controla las vistas de interacción entre las paginaciones de las vistas diaria, mensual y anual.

- **Parsing con NSXMLPARSE**

Permite la lectura del documento XML, la clase encargada es NSXMLParser de Swift.

Se encarga de recorrer todos los elementos del fichero. Para el almacenamiento de datos se ha elaborado un algoritmo de control de los elementos parseados, estos serán cargados sobre structs previamente definidos.

- **Refresco de Vistas**

El método `viewDidLoad` se ejecuta al cargarse una vista, aquí se define el código necesario para refrescar los datos de una tabla realizando el gesto de deslizamiento hacia abajo, conocido como "pullToRefresh".

○ Compartido (Shared)

Las clases, Agencia, Campaña, Objetivo y UIColorExtension, son utilizadas a lo largo de la App. Son clases compartidas que definen las propiedades que tendrán los objetos.

○ Storyboards

Se define la interfaz visual de la aplicación, la parte correspondiente a la Vista dentro del patrón MVC utilizado a la hora de desarrollar una App para iOS.

A través de AutoLayout nos facilita la adaptación de las interfaces a los diferentes tamaños de pantalla.

○ Assets

Incluye todos los iconos, logotipos e imágenes que se vayan a incluir en el proyecto. Este catálogo debe cubrir las diferentes resoluciones necesarias para el conjunto de dispositivos iOS, @1x, @2x, @3x.



- Info.plist

En este fichero se configura partes importantes de la App, como orientaciones soportadas, nombre del paquete de la App, estilo de la barra de estado...etc.

- Gradientes

Los colores con estilo degradado no están incluidos por defecto dentro del SDK de iOS, por lo que hay que crearlos insertando una subcapa de tipo "CAGradientLayer".

- Pods

Librerías de terceros que permitirá dotar de animaciones a la App, el más popular instalador de paquetes se llama Cocoapods.

Las animaciones que se han utilizado son de "Spring" (<https://github.com/MengTo/Spring>) y en la propia web ya nos indica el texto a introducir dentro del Podfile.

Test y Mejoras

- La aplicación ha sido sometida a test reales en dispositivos físicos de algun@s compañeros.
- El resultado ha sido bastante satisfactorio pero se han identificado diferentes mejoras para un futuro.
 - Panel de configuración.
 - Parametrización de campos.
 - Login de usuario
 - Envío por correo electrónico de la información, así comunicar e interactuar con las tomas de decisiones sobre el negocio.

Conclusiones

- Los objetivos previstos inicialmente se han cumplido satisfactoriamente.
- La aplicación tiene muchas posibilidades de ampliación por lo que es escalable
- Me ha permitido entender los procesos a desarrollar dentro de un proyecto.
- Aprender un nuevo lenguaje y poner en prácticas habilidades adquiridas en la carrera.
- Satisfacción de haber realizado un buen proyecto ante duras adversidades.



