Alejandro Muñoz Crespo
Jonathan E. Benavides Vallejo
Francisco J. Reina

DOCUMENTACIÓN iOS

1. Entorno de programación
2. Interfaz Gráfica
    1. Tablet
    2. Teléfono
3. Estructura del código
4. Notificaciones PUSH con Parse
5. CoreData

## 1. Entorno de programación

La programación en iOS se realiza con el IDE (Integrated development environment de sus siglas en inglés) Xcode proporcionado por Apple de forma completamente gratuita. Es un software privativo disponible sólo para el sistema operativo OS X de la propia compañía.

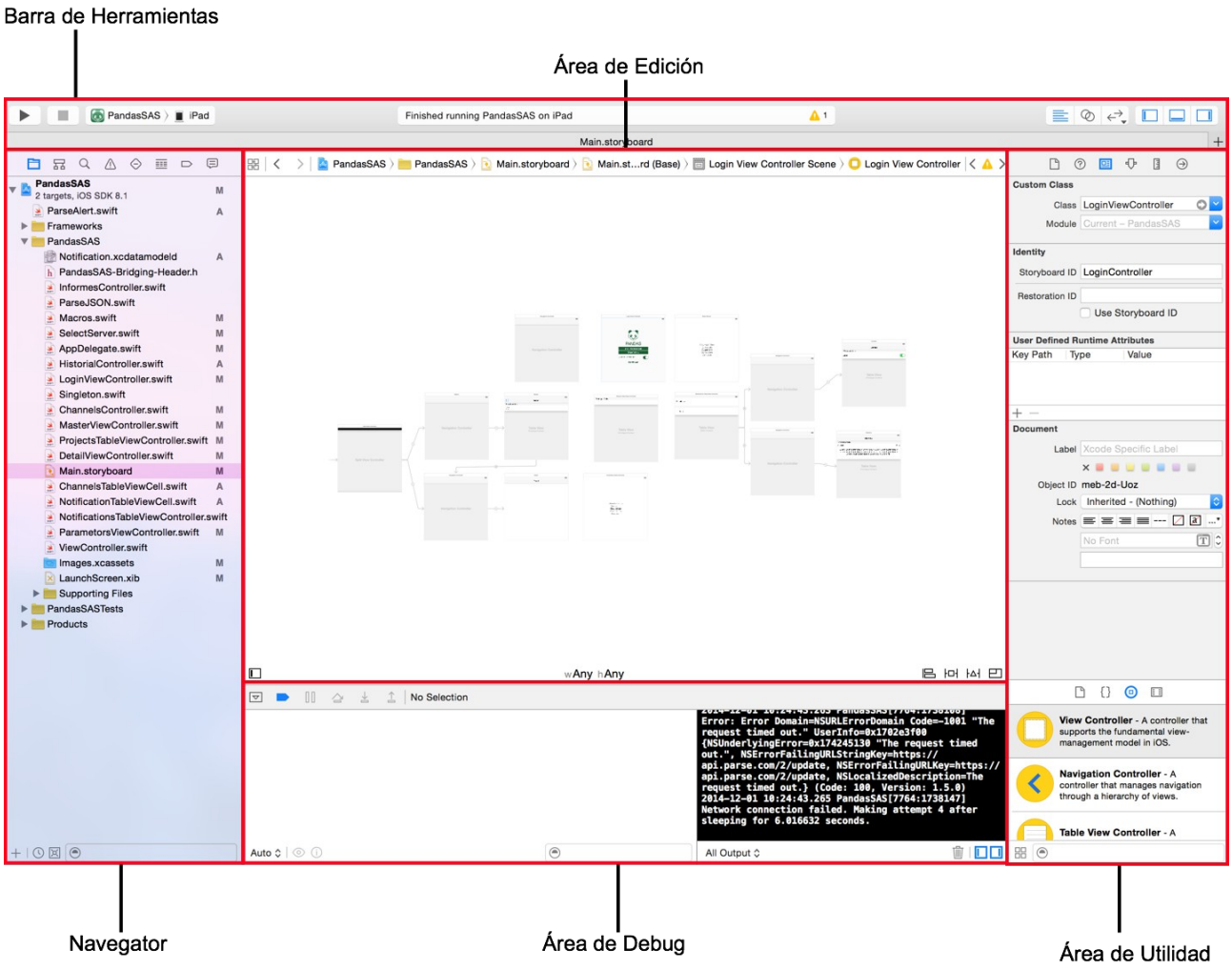



Fig 1.

El entorno trae un simulador, el cual nos permite probar los proyectos en todos los dispositivos de la compañía. Si se quiere probar en un dispositivo físico es necesario registrarse como desarrollador de la compañía con un pago de 80€ anuales.

1. INTERFAZ DE USUARIO
    1. Tablet


Fig 2. Inicio de sesión

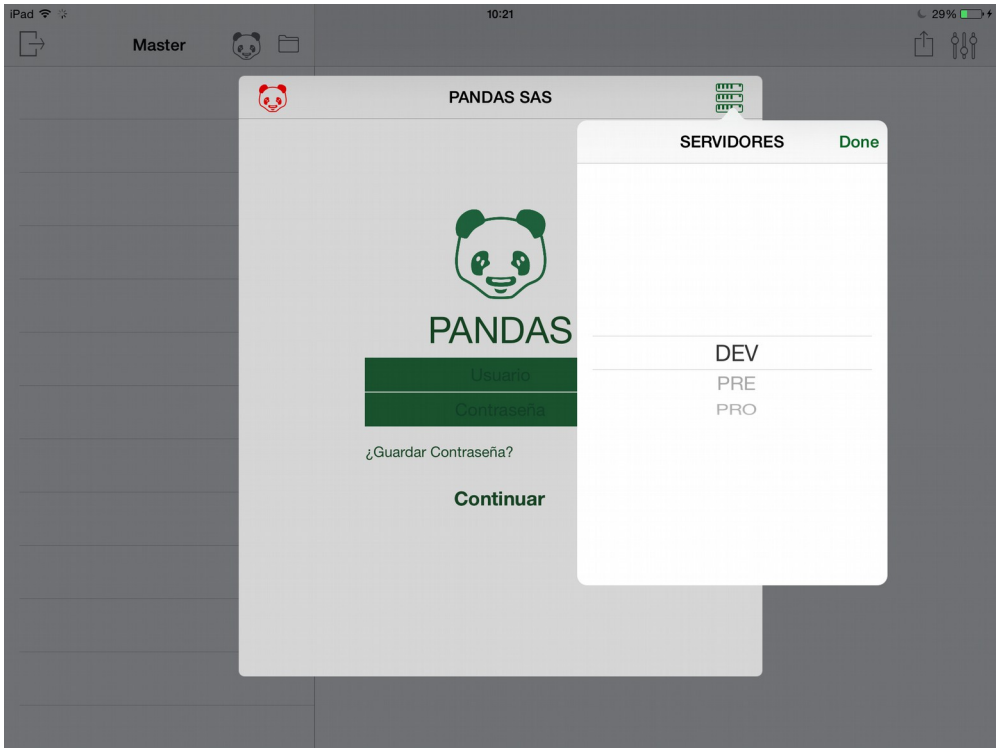



Fig 3. Selección de servidor

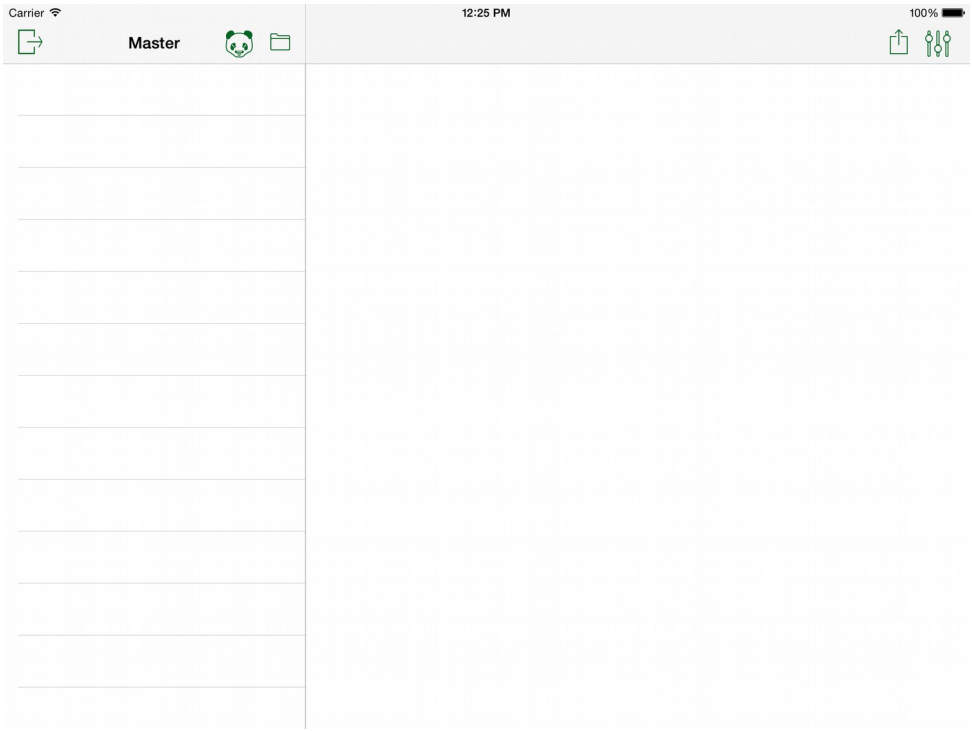



Fig 4. Pantalla principal

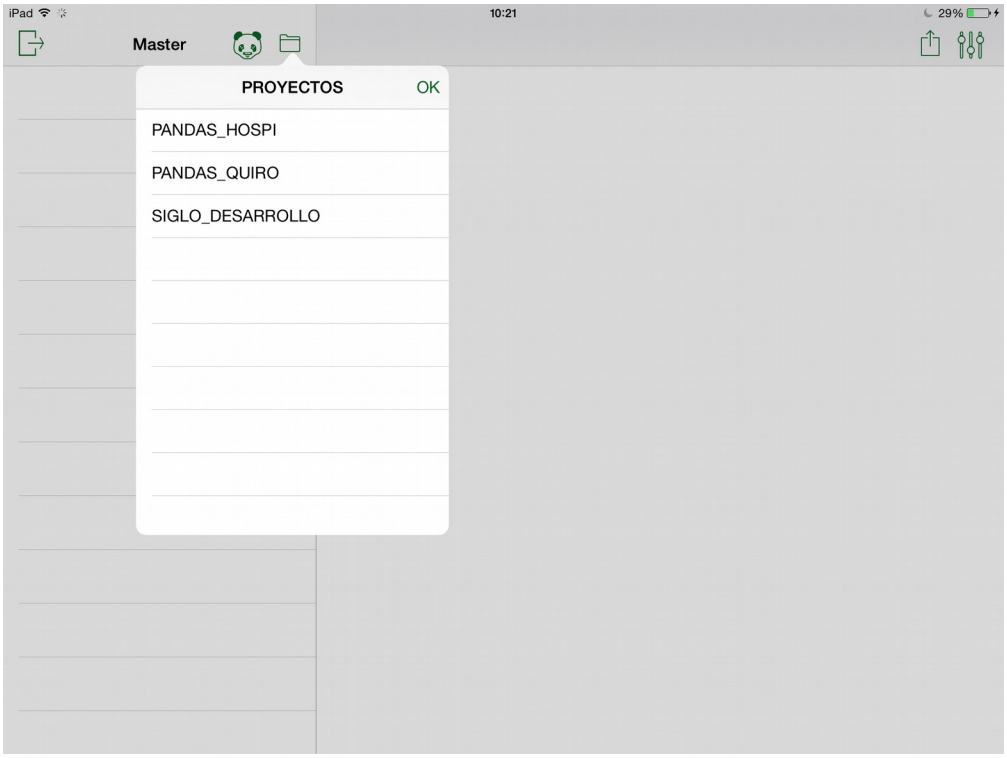



Fig 5. Proyectos del servidor

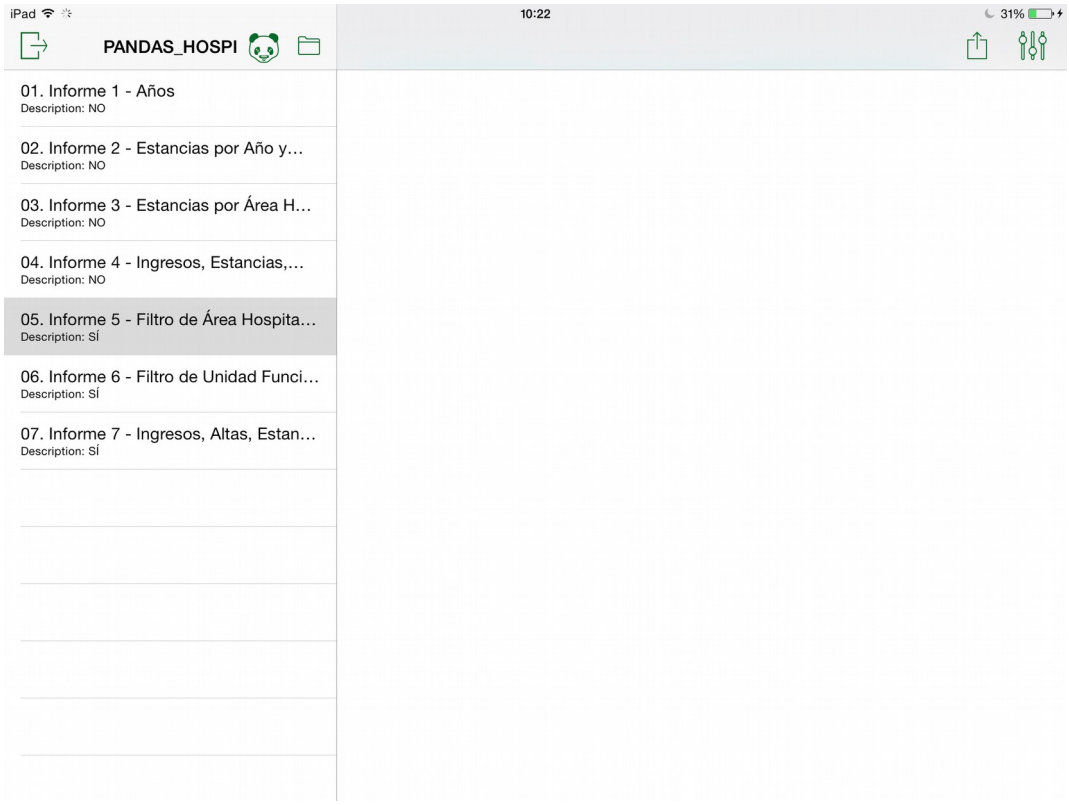



Fig 6. Informes de un proyecto

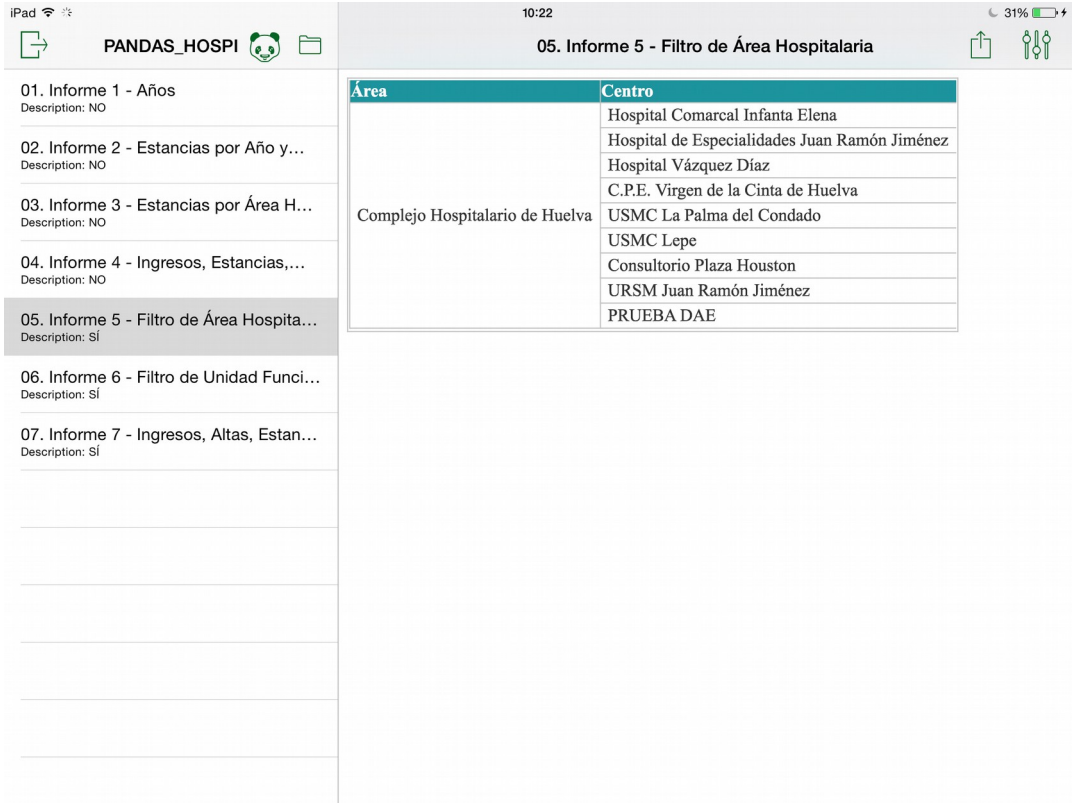



Fig 7. Informe cargado

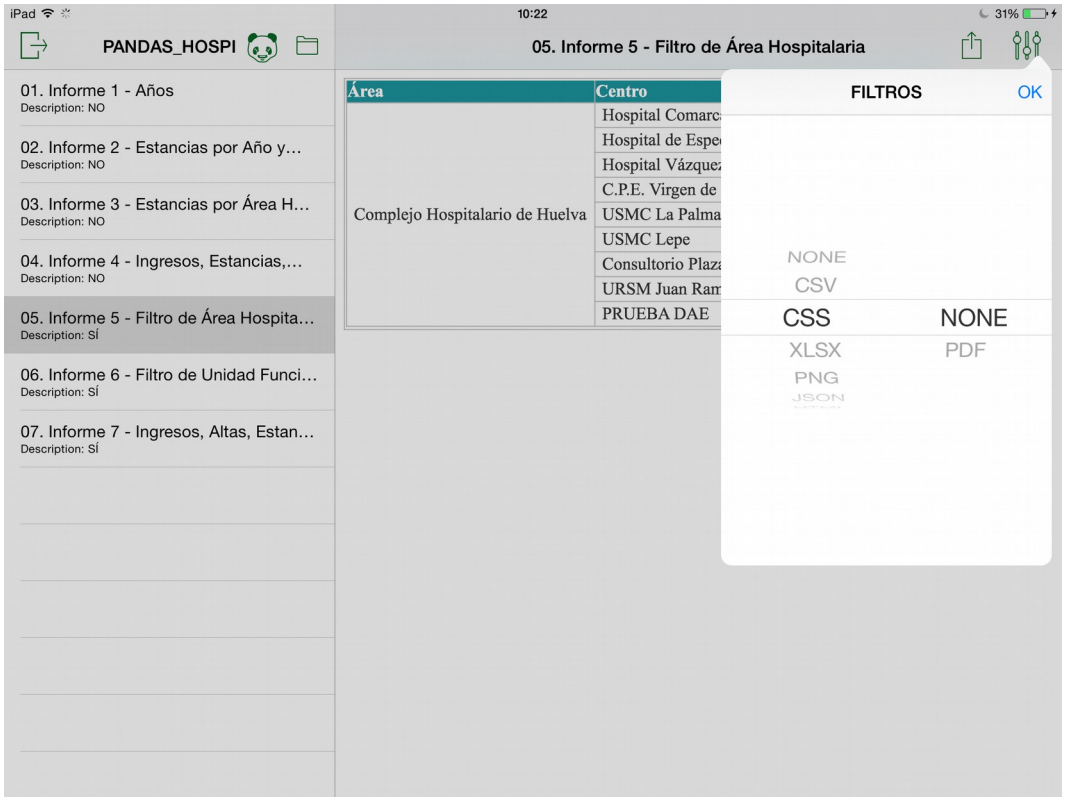



Fig 8. Filtros

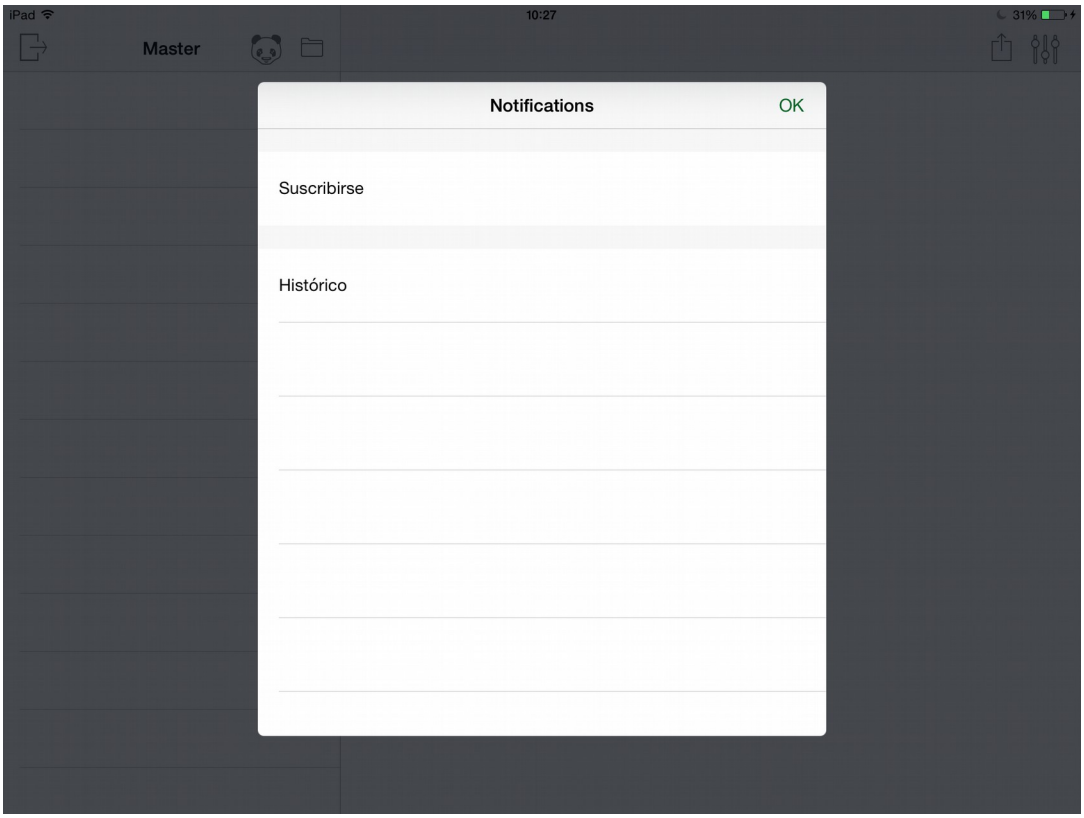



Fig 9. Notificaciones (Pantalla principal)

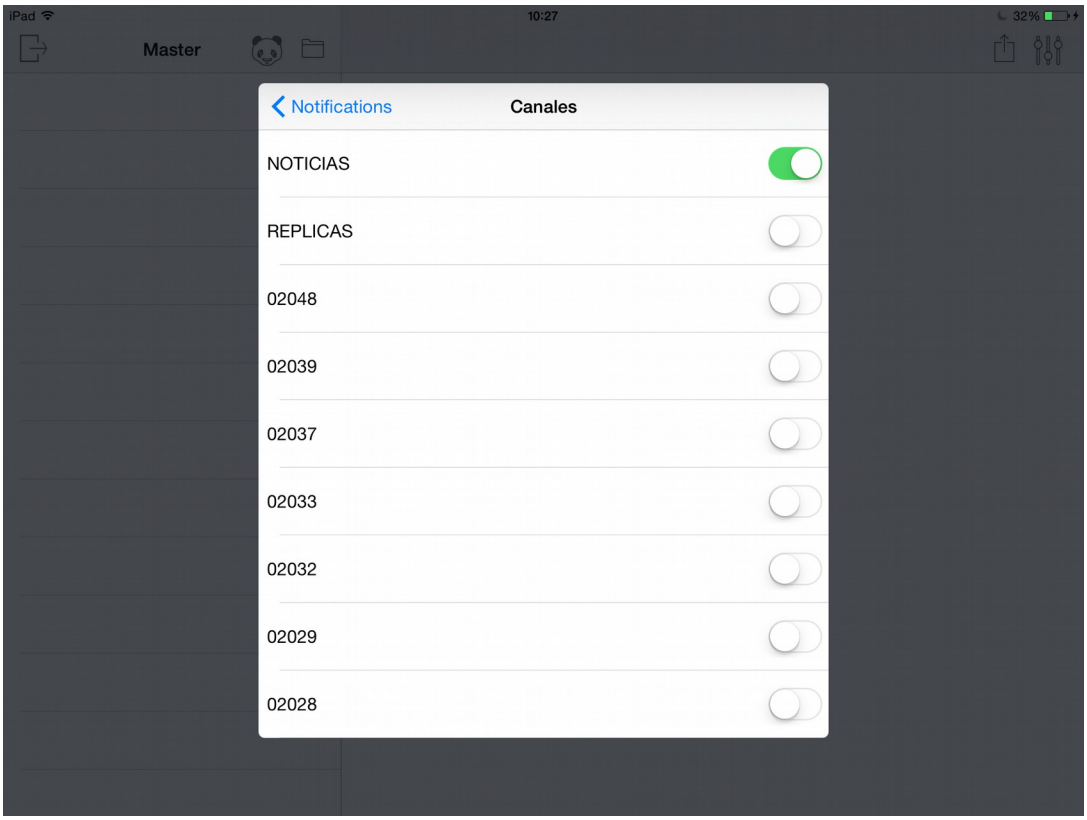



Fig 10. Suscripción a canales

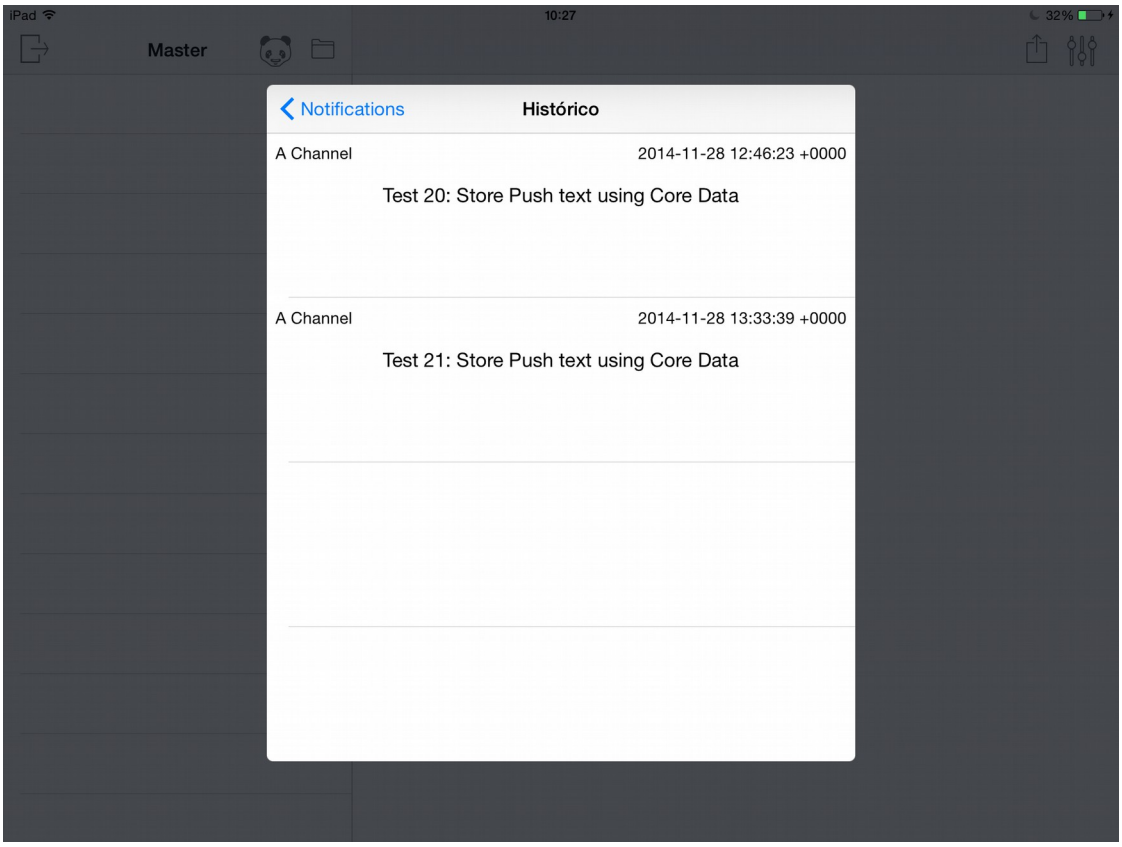



Fig 11. Histórico de notificaciones

## 2. Teléfono


Fig. 12 Inicio de sesión


Fig. 13 Selección de servidor

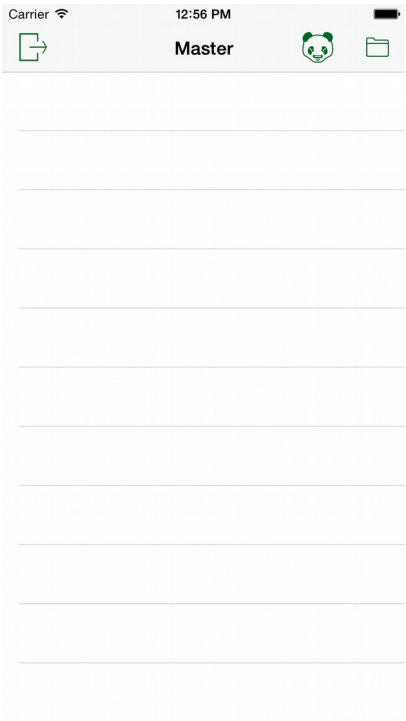



Fig. 14 Principal


Fig. 15 Proyecto

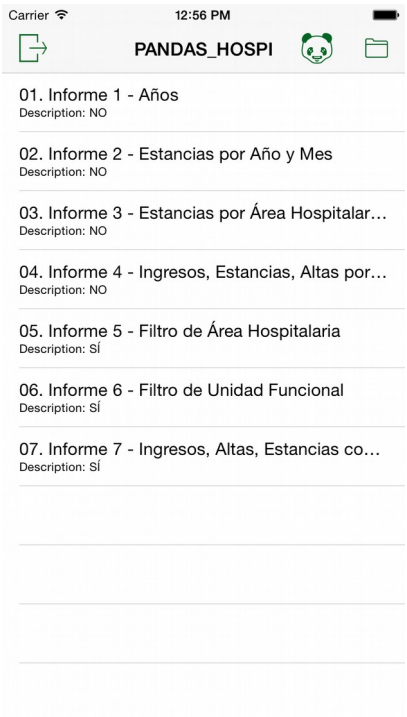



Fig. 16 Informes

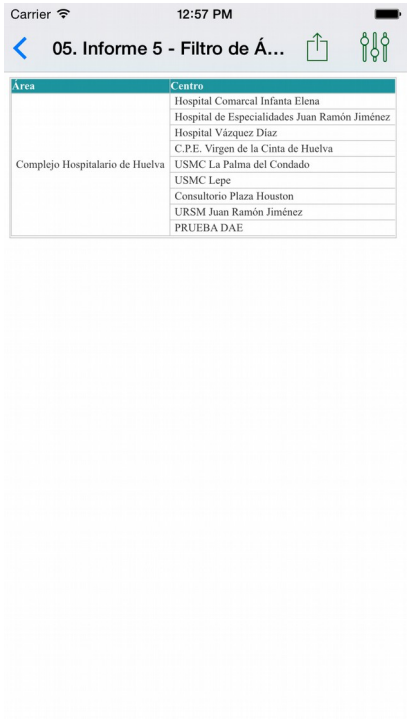



Fig.17 Informe cargado

### 3. Estructura del código

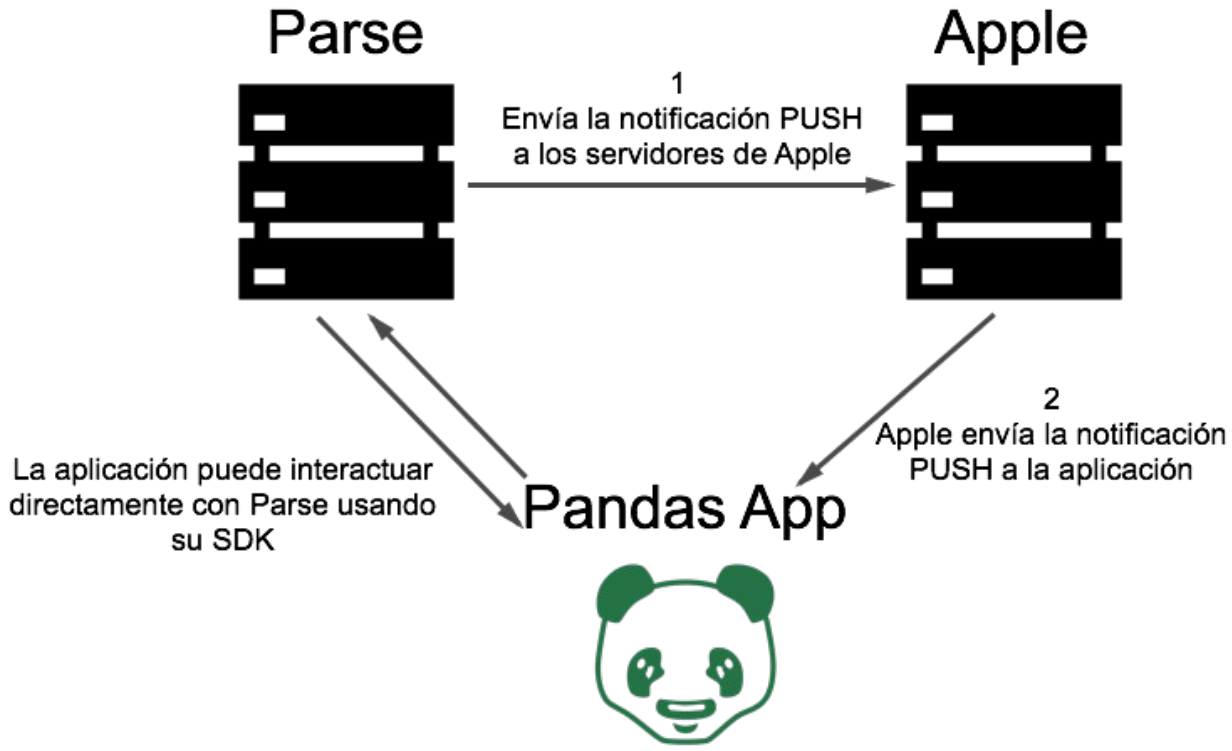
Se ha seguido como patrón de arquitectura de software el modelo-vista-controlador (MVC), en donde por cada pantalla o vista de la interfaz se usa un controlador, además se ha utilizado la plantilla **SplitView** para mostrar el nombre de los informes en el tableView y el informe en un WebView y dicho esto el proyecto cuenta de los siguientes:

- LoginViewController
  - Se envían las credenciales al servidor mediante autenticación NTLM
  - Delegado/s: userDidLoginWithServer(server: String)
    - Delega el nombre del servidor seleccionado
- SelectServerViewController
  - Permite la selección de uno de los servidores (DEV, PRO o PRE)
  - Delegado/s: userSelectServer(server: String)
  - Implementa el Delegado SelectServerDelegate, recibe el nombre del servidor
- MasterViewController
  - Es el controlador principal, se encarga de llamar al login, mostrar los proyectos, los informes, las notificaciones.
  - Delegado/s: No contiene, pero implementa los delegados:
    - LoginDelegate: recibe el nombre del servidor
    - ProjectsDelegate: recibe el nombre del proyecto seleccionado
- DetailViewController
  - ParametrosDelegate: Filtro de como mostrar los informes
- ProjectsTableViewController
  - Carga los proyectos de un servidor en un tableView
- ParametrosViewController (Filtros/Formatos de informe)
  - PickerView con el filtro de los informes
- HistorialViewController
  - Histórico de notificaciones recibidas (TableView)
- NotificationsTableViewController
  - TableView con celdas estáticas de suscripción e Histórico

## 4. Notificaciones PUSH con Parse

Las notificaciones PUSH en iOS están gestionadas por la propia Apple, por este motivo es necesario decirle a Apple que nuestra aplicación va a hacer uso de dicha característica.

El funcionamiento es el siguiente




Pasos:

1) Crear certificado para que Apple envíe las notificaciones PUSH a nuestra aplicación, seguir los pasos: https://www.parse.com/tutorials/ios-push-notifications
2) Activar las notificaciones PUSH de PARSE en nuestro aplicación siguiendo los siguientes pasos: https://www.parse.com/apps/quickstart#parse_push/ios/existing

Problemas encontrados

Al realizar la configuración PUSH por primera vez he detectado que los servidores de PARSE no detectan los dispositivos instantáneamente, muchas veces he tenido que esperar 1 hora al menos.

5. Core Data

Se ha utilizado para almacenar el texto de las notificaciones PUSH, los canales a los que esta suscrito y en un futuro para almacenar preferencias del usuario.

El archivo Core Data esta formado de 2 instancias:

- Notifications
- Channel

Core Data esta presente en los archivos:

- AppDelegate:
  - Para la lectura de los canales predefinidos por el usuario y configurarlos en Parse en el momento de abrir la aplicación, realizado por el método: func subscribedChannelsCoreData()
  - Almacenamiento de las notificaciones PUSH recibidas y guardarlas como historial, el método encargado de esto es: func application(application: UIApplication, didReceiveRemoteNotification userInfo: [NSObject : AnyObject] ) { PFPush.handlePush(userInfo)
  - ChannelsController:
    - A lo largo del código para la lectura y escritura de los canales configurados.
  - HistorialController:
    - A lo largo del código para la lectura y escritura de las notificaciones PUSH.