



TRABAJO DE FIN DE GRADO
GRADO EN INGENIERÍA INFORMÁTICA

iOrg2.0

Herramienta de apoyo a la docencia para el Departamento
de Organización de Empresas

Autor

Rogelio Gil García

Tutor

Dr. Pedro A. Castillo Valdivieso

Cotutora

Dña. María Magdalena Jiménez Barrionuevo



ESCUELA TÉCNICA SUPERIOR DE INGENIERÍAS INFORMÁTICA Y DE
TELECOMUNICACIÓN

Granada, 16 de agosto de 2017

iOrg2.0

Rogelio Gil García

Yo, **Rogelio Gil García**, alumno de la titulación **Grado en Ingeniería Informática** de la **Escuela Técnica Superior de Ingenierías Informática y de Telecomunicación de la Universidad de Granada**, autorizo la ubicación de la siguiente copia de mi Trabajo Fin de Grado (*iOrg2.0*) en la biblioteca del centro para que pueda ser consultada por las personas que lo deseen.

Este documento está desarrollado con LaTeX a partir de la plantilla facilitada por la Escuela Técnica Superior de Ingenierías Informática y de Telecomunicación, y la plantilla de LaTeX de Jaime Torres Benavente, publicada bajo licencia **Creative Commons Attribution-ShareAlike 4.0** (<https://creativecommons.org/licenses/by-sa/4.0/>) que se encuentra en <https://github.com/torresj/indi-android-ui/tree/master/documents>

La licencia de este trabajo es **Creative Commons Attribution-ShareAlike 4.0**(<https://creativecommons.org/licenses/by-sa/4.0/>)

Fdo: Rogelio Gil García

Granada, a 16 de agosto de 2017

Dr. Pedro A. Castillo Valdivieso, Profesor del **Departamento de Arquitectura y Tecnología de Computadores** de la **Universidad de Granada**.


Dña. María Magdalena Jiménez Barrionuevo, Doctora del **Departamento de Organización de Empresas** de la **Universidad de Granada**.

Informa:

Que el presente trabajo, titulado ***iOrg2.0***, ha sido realizado bajo su supervisión por **Rogelio Gil García**, y autoriza la defensa de dicho trabajo ante el tribunal que corresponda.

Y para que conste, expiden y firman el presente informe en Granada a 16 de agosto de 2017.

Los tutores:

 ../images/firma_tutores.png

Dr. Pedro A. Castillo Valdivieso

Dña. María Magdalena Jiménez Barrionuevo

Agradecimientos

A mi familia por darme la oportunidad de llegar hasta aquí.

A mis amigos por acompañarme y darme ánimos desde hace tantos años.

A mis tutores Dr. Pedro A. Castillo Valdivieso y Dña. María Magdalena Jiménez Barrionuevo, por su apoyo y asesoramiento en el proyecto.

Índice general

1. Resumen	1
1.1. Breve resumen y palabras clave	1
1.2. Extended abstract and key words	3
2. Introducción	5
2.1. Motivación y análisis del problema	5
2.1.1. Aplicaciones móviles híbridas	5
2.1.2. Análisis del problema	6
2.2. Estado del arte y elección de la tecnología	7
2.2.1. El lenguaje de programación	8
2.2.2. Frameworks web	9
2.2.3. Sistema para el control de versiones	11
Bibliografía	13

Índice de figuras

2.1. App híbrida	6
2.2. Top languages GitHub	8

Índice de tablas

2.1. Tabla de características.	9
--	---

Índice de fragmentos de código

Capítulo 1

Resumen

Breve resumen y palabras clave

Palabras clave: *Django, Python, Materializecss, aplicación web, desarrollo web, docencia.*

iOrg2.0 es un proyecto que nace a partir del trabajo realizado como becario ICARO¹ en el Departamento de Organización de Empresas de la Facultad de Ciencias Económicas y Empresariales de la Universidad de Granada y en colaboración directa con la Oficina de Software Libre. En dicho proyecto el Departamento de Organización de Empresas necesitaba desarrollar una aplicación móvil como apoyo a la docencia de sus asignaturas.

Con el objetivo de integrar las nuevas tecnologías en la docencia de estas asignaturas nace iOrg1.0. Esta aplicación se desarrolla como una aplicación móvil híbrida² dado el poco período de tiempo del que se disponía y los pocos recursos capaces de sostener. Ya que no era posible, por parte del departamento, mantener un servidor back-end y una base de datos, se decidió para una primera versión de la aplicación, desarrollar el proyecto en un entorno de aplicación móvil híbrida, que accede a Google Drive para servir los datos directamente sin almacenarlos.

iOrg2.0 pretende continuar con la idea original de explotar las virtudes de las nuevas tecnologías en el ámbito docente y a su vez solventar y mejorar situaciones de la primera versión híbrida, soluciones y decisiones que

¹El Portal de Gestión de Prácticas en Empresa y Empleo utilizado por las Universidades Públicas Andaluzas, la Universidad Politécnica de Cartagena y la Universidad Autónoma de Madrid.

²Combinación de tecnologías web que no son aplicaciones móviles nativas ni tampoco están basadas en Web, porque se empaquetan como aplicaciones para distribución y tienen acceso a las APIs nativas del dispositivo.

desarrollaremos en el siguiente capítulo. Sin embargo en esta versión se deja a un lado la aplicación móvil y se apuesta por el análisis y desarrollo de una solución que pasa por un servidor web.

Qué pretende de cara al alumno.

Qué pretende de cara a los docentes.

iOrg2.0: Herramienta de apoyo a la docencia para el Departamento de Organización de Empresas

Rogelio Gil García

Extended abstract and key words

Key words: *Django, Python, Materializecss, aplicación web, desarrollo web, docencia.*

Capítulo 2

Introducción

Motivación y análisis del problema

Como ya hemos hablado en el capítulo anterior, iOrg2.0 viene de una aplicación móvil híbrida en la que se detectaron carencias que probablemente provengan de la inexperiencia y el escaso tiempo de análisis del que se disponía. Vamos a exponer brevemente las características de las aplicaciones híbridas para entender los problemas que este proyecto pretende solventar.

Aplicaciones móviles híbridas

Con la llegada de los *smartphone* se ha impulsado enormemente el consumo y producción de apps móviles. En un principio, una aplicación móvil se desarrollaba como aplicación nativa con los contras que ello supone, como una curva de aprendizaje alta de un sistema concreto, en la mayoría de los casos desarrollo monoplataforma, y también los pros como un desarrollo y control mas específico del sistema, sistemas mas robustos, etc.

Ni que decir tiene que una aplicación compleja necesita las ventajas de un desarrollo específico y nativo de un sistema móvil, pero ¿qué pasa con las aplicaciones simples y livianas que, cada vez más, consumimos? Es aquí donde toman importancia las aplicaciones móviles híbridas.

Las principales ventajas de estas aplicaciones son:

- Bajo coste de desarrollo.
- Multiplataforma.
- Tiempo de desarrollo corto.
- Fácil distribución.

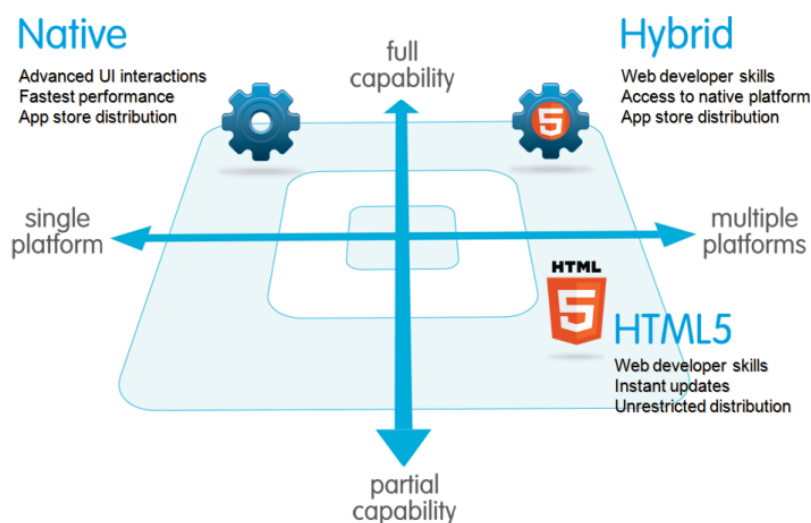


Figura 2.1: App móvil híbrida (<https://developer.salesforce.com>).

Y lo que más nos interesa para este proyecto, sus desventajas:

- Rendimiento pobre.
- Escalabilidad baja.
- Diseño sujeto al diseño web.
- Acceso a las características especiales del hardware limitado.

En un principio la aplicación que se desarrolló en el período de prácticas ICARO se planteó como una aplicación sencilla donde consultar conceptos y esquemas de las asignaturas. Los datos se consultarían desde una hoja de Google Drive común al departamento y se mostrarían en la aplicación directamente y por estas características, se .

A medida que avanzaba el desarrollo y acercándose la fase final, el “cliente” planteó nuevas funcionalidades, topándonos con el primer problema: **la escalabilidad** de estas aplicaciones.

Análisis del problema

Una vez desarrollada esta aplicación, aunque en general quedamos contentos con el resultado, acusamos los siguientes problemas:

- No disponíamos de un servidor, toda la aplicación se ejecutaba en el cliente.

- Falta de base de datos.
- No se usa la API de Google Drive
- Acceso a Google Drive en cada sección de la aplicación
- Lentitud de acceso de datos en cada sección de la aplicación
- Mala elección de framework CSS, señales de obsoleto.

Solventar estos problemas, con el fin de iniciar un proyecto mejor planificado, con miras a una mayor escalabilidad analizando los requisitos presentes y los que pudieran surgir en un futuro, es la principal motivación para este proyecto. Aprovecharemos para mejorar la experiencia de usuario, ya que la lentitud de carga entre secciones era uno de los grandes problemas que hacían tediosa la consulta del contenido de la asignatura.

El primer paso importante es olvidar la tecnología móvil híbrida, que originó muchos problemas en el proyecto anterior y dividirlo en dos, una parte de desarrollo web y una parte de desarrollo móvil capaces de escalar sin problema. Nuestro proyecto se centrará en la aplicación web.

Estado del arte y elección de la tecnología

Para abordar los problemas descritos y antes de realizar un análisis más profundo se hará un estudio de las tecnologías web que pueden ser útiles en proyectos de este tipo. Teniendo claro que vamos a desarrollar el proyecto bajo una arquitectura web y hecho el repaso de los inconvenientes en las tecnologías híbridas podemos sintetizar los siguientes puntos que necesita nuestro nuevo proyecto:

- Servir el contenido con un servidor web.
- Consultar el contenido de la hoja de Google Drive del Departamento.
- Almacenar el contenido en una base de datos.
- Gestión de usuarios.
- Establecer roles para los usuarios.

El lenguaje de programación

Lo primero que vamos a plantearnos es el **lenguaje** a usar, y que mejor forma de abrir un abanico de posibilidades que explorar los lenguajes más usados en la comunidad de GitHub¹ desde mediados de 2016 a mediados de 2017.

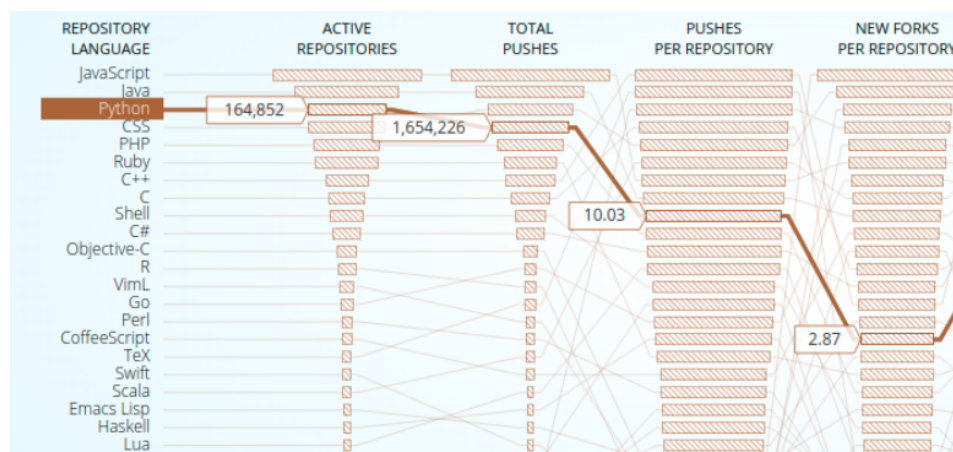


Figura 2.2: Top languages GitHub (<http://github.info/>).

Como podemos ver en la figura anterior, entre los 5 lenguajes más usados se encuentran: JavaScript, Java, Python, PHP y Ruby (y sí, está CSS y lo hemos obviado ya que es un lenguaje usado únicamente para definir los estilos de un documento HTML o XML).

JavaScript nos ofrece la alternativa de Node.js para el lado del servidor pero no está diseñado para aplicaciones complejas. Node.js es un lenguaje interpretado y aunque trabaja de forma asíncrona, está diseñado para ejecutarse en un sólo hilo, por lo que podría presentar ciertas limitaciones a la hora de escalar en aplicaciones complejas. Además las aplicaciones con Node.js permanecen en RAM, no siendo re-interpretadas con nuevas peticiones si no que se procesa directamente desde la memoria.

JavaScript no se estudió profundamente durante el Grado en Ingeniería Informática y aunque descubrir e indagar en un lenguaje nuevo podría ser un aliciente, sus propiedades y las “limitaciones” en cuanto a la escalabilidad,

¹GitHub es una plataforma de desarrollo colaborativo donde usted puede alojar proyectos propios así como contribuir en otros proyectos. GitHub usa el sistema de control de versiones Git.

que recuerdan al desarrollo en JavaScript de las aplicaciones híbridas, nos hacen descartar este lenguaje.

Java en cambio es un lenguaje que hemos profundizado más durante los años de estudio. Si es verdad que la mayoría del desarrollo que llevamos a cabo estos años era referente a aplicaciones de escritorio, pero la idea de un lenguaje nuevo o poco profundizado nos parecía más interesante en este proyecto.

Llegamos a los tres lenguajes que nos parecieron más interesantes para el proyecto **PHP Ruby** y **Python**. A una primera vista estos lenguajes eran candidatos para el desarrollo, vamos a señalar las características que se observaron para decidirnos por uno:

Características		
PHP	Ruby	Python
Curva de aprendizaje corta	Curva de aprendizaje compleja	Curva de aprendizaje corta
Legibilidad y estructuración baja	Legibilidad y estructuración media	Legibilidad y estructuración alta
Popularidad alta	Popularidad baja	Popularidad media
Comunidad media	Comunidad media	Comunidad alta

Tabla 2.1: Tabla de características.

Hemos visto que PHP es el más usado en cuanto a cantidad de aplicaciones totales desarrolladas, pero a lo largo del último año, como se aprecia en la figura 2.2, Python ha crecido enormemente en el número de contribuciones y en su comunidad. Viendo la evolución de este lenguaje y teniendo en cuenta sus características que ayudan a crear aplicaciones bien estructuradas, escalables, con módulos altamente reutilizables y teniendo una comunidad, tanto anglosajona como hispana, tan grande, Python será la opción elegida como lenguaje de iOrg2.0.

Decir también que durante el grado se hizo una breve introducción a Python y se usó para pequeñas y aisladas aplicaciones, sintiendo una gran inquietud por usar este lenguaje en un proyecto más completo.

Frameworks web

Un framework web es una colección de paquetes o módulos que permiten a los desarrolladores escribir aplicaciones o servicios web sin tener que manejar detalles de bajo nivel como protocolos, sockets o gestión de procesos.

Las aplicaciones web comúnmente utilizan y hacen uso de una serie de componentes, ya sea durante su desarrollo o en la etapa de producción, tales como un servidor HTTP, mecanismos de almacenamiento como podría ser una base de datos, un motor de plantillas, un conjunto de herramientas AJAX, etc. Estos framework agrupan y gestionan paquetes con estos componentes que ayudan en la abstracción del desarrollo de bajo nivel.

Ya que tenemos claro el lenguaje, vamos a hacer un breve estudio de los frameworks web existentes que nos permitirán un desarrollo ágil y mejor estructurado. Los tres frameworks web más importantes de Python por su volumen de proyectos, comunidad y completitud son:

- Django
- TurboGears2
- Web2Py

Entre ellos, **Django** es claramente el más usado y con una comunidad más grande. Se basa en el principio DRY² lo que propicia un desarrollo más rápido, con menos código y por consecuencia más limpio y pragmático. Django se centra en automatizar tanto como sea posible. En la documentación de Django se habla de que el framework usa un enfoque MVC³ y en la misma documentación también se habla de un enfoque MVT⁴. Esto es porque ambos enfoques están estrechamente relacionados, se habla de un MVT donde el controlador lo gestiona el propio framework trabajando el desarrollador directamente con los “Templates”.

TurboGears2 aparece con la idea de resolver las frustraciones de otros frameworks y con la filosofía de adoptar lo mejor de otros frameworks de características similares y resolver sus puntos flacos. Framework que destaca por la agilidad en el desarrollo de aplicaciones y servicios simples y la capacidad de escalar hasta aplicaciones completas. Utiliza un concepto de modularización que permite adaptar distintos motores de plantillas y webservers. Utiliza como controlador SQLAlchemy⁵, ORM muy bien considerado y afamado.

Por otro lado **Web2Py** es el más sencillo de los tres. No tiene archivos de configuración, no requiere instalación e incluso se puede ejecutar desde una unidad USB. Extendido y con una comunidad amplia está muy indicado para

²Don't Repeat Yourself

³Modelo Vista Controlador

⁴Modelo Vista Template

⁵Conjunto de herramientas Python SQL y ORM que ofrece al desarrollador toda la flexibilidad y funcionalidad de SQL

proyectos sencillos. Al igual que TurboGears2 utiliza la filosofía de agrupar lo mejor de otros frameworks. Utiliza la idea de MVC de Ruby On Rails y la filosofía de forms de Django por lo que para muchos desarrolladores será muy cómodo si conocen estos frameworks.

Para este proyecto el framework que hemos decidido utilizar es **Django**. Al ver su comunidad, volumen de proyectos y calidad de los mismos tales como Instagram, Mozilla o Pinterest creemos que es el indicado para desarrollar la aplicación. Django es compatible con Python 2 y 3 y muy recomendable para proyectos con miras a una gran escalabilidad. Personalmente y antes de desarrollar iOrg2.0 me parece interesante ya no sólo la gran comunidad de desarrolladores también la calidad de su documentación en cuanto a explicaciones y ejemplos. Creemos por esto que es la mejor apuesta para el TFG.

Sistema para el control de versiones

El control de versiones se dedica a la gestión de los cambios que se realizan sobre los ficheros o configuraciones. Una revisión o versión de un fichero es el estado en el que se encuentra este en cierto momento de su desarrollo.

Un sistema de control de versiones facilita la administración de las distintas versiones del código o el producto desarrollado, así como las posibles especializaciones realizadas, ya sea para una bifurcación del desarrollo en un punto del proyecto o para mantener una versión estable en producción mientras se desarrolla un proyecto.

Durante el Grado en Ingeniería Informática hemos trabajado con Git como sistema de control de versiones satisfaciendo totalmente las necesidades de los desarrollos. No encontramos razones para dejar Git y utilizar otros sistemas. Aún así se indagó en otras opciones comunes en el desarrollo de proyectos encontrando dos opciones en la mayoría de estos: Git y SubVersion. Las principales características son:

Git

- Control de versiones distribuido.
- Se trabaja sobre copias locales del repositorio.
- Autorización para la totalidad del repositorio.
- Sólo es necesaria conexión para la sincronización.

SubVersion

- Control de versiones centralizado.
- Repositorio central donde se generan copias de trabajo.
- Autorización sobre rutas concretas del repositorio.
- Necesaria conexión a la red con cada acceso.

Ya que necesitamos un repositorio donde poder trabajar con o sin conexión a internet, excepto cuando necesitemos sincronizar nuestro trabajo local, y no necesitamos un control de permisos sobre rutas específicas del repositorio, tendremos acceso al repositorio completo, seguimos pensando que Git es la mejor opción.

Aunque hablaremos de ello más adelante, vamos a tener la precaución de dividir los repositorios del proyecto en dos. Un repositorio será destinado al código fuente y otro a la documentación. Esto nos ayudará en el momento de desplegar la aplicación en un servidor de desarrollo ya que podremos subir los ficheros fuente independientemente sin tener que manejar los ficheros de la documentación en el mismo repositorio.

Bibliografía

- [1] “Web frameworks”, Wiki Python, Fecha de último acceso: 12/01/2017.
href<https://wiki.python.org/moin/WebFrameworks>
- [2] “Django”, Fecha de último acceso: 25/01/2017.
href<https://www.djangoproject.com/>
- [3] “TurboGears”, Fecha de último acceso: 25/01/2017.
href<http://turbogears.org/>
- [4] “Books and tutorials for Django”, Full Stack Python, Fecha de último acceso: 12/05/2017.
href <https://www.fullstackpython.com/django.html>

