DJANGO'S ARCHITECTURE

October 22, 2018

Luis Cárabe
Blanca Martín

INTRODUCTION

Django is an example of a MVC (Model View Controller) pattern. That is, the logic, the representation and the application work-flow are separated into different modules.

A Django project consists of one or more Django applications. The combination of the project and each application forms the MVC pattern.

MODEL

The Model represents the application data, and in Django's case it's given by the file *models.py* inside each Django application and its interaction with the project's database. This file contains the functionality of the application, and will be accessed by the application to update the status, or just to check it.

VIEW

The View accesses the Model and represents its data, adapting the output to the client and the platform it's been viewed on. It is composed inside the Django application by the set of *html* files that makes up the *templates* directory that rests in the project.

CONTROLLER

The Controller takes the user interactions with the View and notifies the Model. Also, it decides which output to show the client. So, in our Django project the Controller is the file *urls.py*, and inside the Djando application would be the files *urls.py* and *views.py*.

BASIC WORK-FLOW

When the Server inside which is placed the Django project receives a request for an URL, the URL's pattern is searched in the file *urls.py* of the Django project, that redirects the request to a Django application's *urls.py* file inside the project.

The *urls.py* file inside the application, calls a function of *views.py* based on the given URL. That function analyzes the request, accessing to the database (created based on *models.py*) if

it is necessary, and selects a representation (*templates* directory) for the solicited data.

This, however, only shows a one-way channel in the communication between the parts that composed the MVC pattern. Also, there could be a scenario where the client interaction was not to request data but to send it. That is, the form filling case. In this situation, the process would be exactly the same, but the database access would be done to update the status of the data, and the *templates* will be notified of such change once the Controller selects a View.