USING COURSEBUILDER AS A MOOC PLATFORM

Martín Candela Calabuig[1], Carlos Carrillo Boj[1], Adrián González Martín[1], Pedro Pernías Peco[1], Sergio Luján-Mora[2]

[1] Institute of International Economy, University of Alicante (SPAIN)
[2] Department of Software and Computing Systems, University of Alicante (SPAIN)

Abstract

Massive Open Online Courses (MOOCs) are online courses with the option of free and open registration that are usually followed by thousands of students from all over the world at the same time. MOOCs are usually created and offered by higher education institutions, such as universities. The structure and content of MOOCs usually mimics traditional online courses: a syllabus, a calendar, educational materials (mainly videos), some activities or projects, quizzes (usually multiple choice questions) to assess students' learning, and a forum to discuss with instructors and other students.

This paper focuses on the development of MOOCs (Massive Open Online Courses) based on Google CourseBuilder. Firstly, we discuss the current available platforms aimed to develop MOOCs. Then, we explore the main features of Google CourseBuilder. Then, we explain how to create a MOOC with CourseBuilder. Finally, we provide some advices to improve the performance of CB and to avoid some common problems.

Keywords: LMS, MOOC, e-learning, platform, innovation, technology.

1 INTRODUCTION

MOOCs (Massive Open Online Courses) have been around us since 2008, when around 2,300 students took part in a course called "Connectivism and Connective Knowledge", organized by the University of Manitoba (Canada). However, 2012 was widely recognized as "The year of the MOOC" [1], because some MOOC initiatives, such as Coursera, Udacity, or edX, gained a world-wide popularity. Many experts consider MOOCs a "revolution in education" [2]. However, other experts think is too soon to make such a claim and MOOCs still have to prove their real value [3].

MOOCs are the natural evolution of OpenCourseWare (OCW), but in contrast with them, MOOCs offer the opportunity to interact with other students and instructors, and they typically provide a sort of certification at the completion of the course.

So far, MOOCs do not differ much from online courses that have existed among us for years: a syllabus (with course objectives and expected outcomes), a calendar, some educational materials quizzes and exams (usually multiple choice exercises) to assess students' learning, and a forum to discuss with the teacher and other students.

A key factor in the deployment and success of a MOOC is the scalability. Some systems start to work slowly when the number of users increases. Scalability is the ability of a system to handle a growing amount of users (workload) in a capable manner because it can be enlarged to accommodate that growth. A system without this ability cannot be enlarged to accommodate a huge amount of users, or it can be enlarged but the cost is prohibitive because adding new resources to support new users is more expensive each time. Some popular Learning Management Systems (LMS), such as Moodle or Sakai, have been used to provide MOOCs. However, these LMS present some scalability problems because they were not designed to support thousands of students at the same time.

Google CourseBuilder (GCB) is an open source project of Google [4]. It is important to clarify that GCB is not a Google product. CB provides the capability for anyone to create a MOOC. GCB runs on Google App Engine (GAE), Google's could computing platform for developing and hosting web applications in Google's data centers. The most important feature of GAE is the high level of scalability it offers. But GAE provides additional capabilities that are essential for the creation of a MOOC.

In this paper, based on our experience in the creation of some MOOCs, we discuss how to use GCB as a MOOC platform. Firstly, we analyze and compare the different platforms that can be used to develop a MOOC nowadays. Then, we explain how to create a MOOC with GCB and finally we highlight the advantages and disadvantages that GCB offers.

2 MOOC PLATFORMS

The current available platforms aimed to develop MOOCs can be classified based on different dimensions:

- Open source o proprietary.
- Cloud computing or traditional web hosting.
- Software provided or software as a service.

On the one hand, there are some online education websites, such as Coursera, edX, and Udacity, that offer many MOOCs. Unfortunately, these websites do not provide their software, courses have to be hosted in their platforms, and the offering of a course must be negotiated with those platforms (and a contract must be signed).

On the other hand, other online education websites, such as P2PU (based on Lernata), Canvas Network (based on Canvas), and CourseSites (based on Blackboard) are similar to Coursera, edX, and Udacity, but they allow higher education institutions to offer MOOCs without any cost. The publication of courses is almost immediate and they do not impose strong restrictions.

On the other hand, some popular LMS (Learning Management Systems), such as Moodle or Sakai, have been used to provide some MOOCs. However, these LMS present some scalability problems because they were not designed to support thousands of students at the same time.

- Sakai it's an open source solution for Virtual Learning Environments (VLE). It is used by many institutions and it is very modular and customizable. So, where is the problem? It wasn't developed with MOOCs in mind (they did not even exist), therefore it has some scalability problems. Remember than we are not dealing with a classroom of 200 students, but an audience of 25,000 students, so the solution must handle this. There are some initiatives to bring this functionality to Sakai (including peer reviews, better analytics, etc) but there are not mature enough.
- Moodle it's another open source e-learning platform, similar to Sakai. So, it has a similar problem: it is not built for supporting more than 25000 students. There are movements stating that Moodle can be used for MOOCs and even trying to, but these attempts never go more than 2000 students so they are not reliable.

Both Sakai and Moodle are great for "small" virtual learning environments like traditional classrooms (from 50 to 200 students) but when we go bigger scalability problems appear.

Fortunately, a new breed of LMS has appeared in recent months with the aim of providing tools to create MOOCs: edX, OpenMOOC, and Google CourseBuilder (GBC).

- edX is based on XBlock, a component architecture that enables developers to create independent course components, or XBlocks. The creation of a MOOC is made with the combination of XBlocks from a variety of sources, such as, text, video, wiki, and so on. XBlock was released in March 2013.
- OpenMOOC is an "an open access platform based on free source components to build a connectivist environment". Unfortunately, the identification module of OpenMOOC doesn't accept standard user accounts, such as Google, Facebook, or Twitter.
- CourseBuilder is a platform specifically for MOOC development. It started as a Google internal experiment, but the google's developers soon saw its potential, and began to use it for their training courses. In September 2012, Google announced the availability of the source code for anyone interested in the MOOC movement. To deploy a course all we have to do is prepare the content, download the source code and the launcher, create an application in the App Engine and deploy the code with the launcher to that specific application. Once the course is live we can start to add more content to it through the CMS that comes with GBC. What we have then is an easy to use, scalable application that can serve to thousands of students without batting an eye. However, many other solutions offer these features, so why choose GBC over all of them? The reason is that while we did need all those features, what we're really wanted was to add our own suite of characteristics to the course, and in that aspect GBC shines over every other option. Its architecture is easy enough to understand so that adding a new feature doesn't take more than it actually needs to. This allowed us to develop key features that made our course stand out among the rest, like an educational social network or a custom admin panel,

which was crucial for the development of our course, since it let us see statistics about our users usage of the course in real time, and adapt in response to that information.

3 GOOGLE COURSEBUILDER

As we have stated, Google CourseBuilder is an open source project of Google, not a Google product. GCB offers a lightweight way to publish course material online, track student engagement, and evaluate students' performance. An international community of GCB users is growing rapidly. GCB is a suitable solution for institutions looking for an in-house platform. Besides, as the source code is available and can be modified, institutions can develop their own customizations.

GCB's website offers a lot of information for MOOC authors. It is worth noting the section on evaluating course efficacy. The documentation explains how to collect and analyze data of various sorts.

A key factor in the deployment and success of a MOOC is the scalability. Some systems start to work slowly when the number of users increases. Scalability is the ability of a system to handle a growing amount of users (workload) in a capable manner because it can be enlarged to accommodate that growth. A system without this ability cannot be enlarged to accommodate a huge amount of users, or it can be enlarged but the cost is prohibitive because adding new resources to support new users is more expensive each time.

GCB runs on Google App Engine (GAE), Google's cloud computing platform for developing and hosting web applications in Google's data centres. The most important feature of GAE is the high level of scalability it offers. However, GAE has an important disadvantage: GAE is a fee service. A GAE application can consume a certain level of computing resources for free (blobstore, channels, datastore operations, mails sent, etc.), controlled by a prefixed set of limits, over a 24 hour period; once one of these resources has been depleted, the resource becomes unavailable until the resource is replenished and the application will stop working. Anyway, the costs associated with the use of this service are low compared to a traditional web hosting. Therefore, the use of GCB and GAE implies a cost than must be evaluated before using this LMS.

GCB is programmed in Python language. GCB can be changed and extended easily. Besides, latest version of GCB offers the possibility of adding custom components and modules. For example, GCB documentation explains how to develop a Khan Academy module to import Khan Academy exercises into GCB.

4 HOW TO CREATE A MOOC USING GBC

4.1 Recommend Knowledge

To setting up a simple course with CourseBuilder, the only acknowledgment necessary is a little bit of JavaScript for creating the activities and assessments but you don’t need to have knowledge about web programming or how to set up set up the App Engine. However, it is recommended knowing basic concepts about cloud computing, for the correct use of App Engine and personalize the configuration to reduce the cost of it. In addition for the use of your own domain in the App Engine you need to know basic concepts of how to set up a domain.

Once you have your course in the App Engine you can configure it completely, the number of instances you want, the maximum and minimum latency you want, the amount of logs the App Engine, etc. For this reason knowing as much as you can about App Engine is highly recommended for the maintenance of the course in the cloud.

In the other hand, if you want personalize the course appearance or adding extra functionality or extra components, the necessary knowledge is greater. If you want to change the appearance of your course, to make it more similar for the style of your own website you only need to know HTML and CSS, because the views and styles are separated from the application core code. But if you want to go farther adding extra functionality you need to know python, because Google CourseBuilder is written in python. In addition, Google CourseBuilder use the webapp2 framework and follow the MVC pattern so for writing new functionality or new modules it’s recommended follow this pattern and use webapp2.

4.2 Using Google App Engine

4.2.1 Creating an account

To create an account on Google App Engine (GAE), we'll need a Google account (Gmail). You can access the App Engine services through the Cloud Platform portal (https://cloud.google.com/).

We can find the different products Google offers in the Products tab, one of which is Google Cloud, although we'll only use the App Engine: By clicking the "Try it now" button and accepting the terms, we will be able to make use of the services offered for free in up to ten applications. On top of the 10 free applications, Google has set a free quota for every type of resource. This allows us to deploy simple applications or services without spending a dime. If we consume this free quota for one particular resource our application won't be able to use that until the next reset, unless we activate billing in that application.

4.2.2 Features of GAE

The App Engine offers a wide range of features that will allow us to develop, deploy and maintain any kind of application, here are listed a small portion of those:

- NoSQL: The App Engine uses a NoSQL database (known as the Datastore) to save your application's data, but if you have to use a relational database you can do so, since Google offers us an API that'll let you create fully functional relational databases hosted on Google Cloud SQL.
- Files: the file system stands out, since it doesn't let you create or write to local files. This is due to the distributed nature of the applications in the App Engine. Any kind of persistent data must be written to the Datastore. To circumvent this you can create files and store them as blobs (now a deprecated feature) or use Google's Cloud Storage.
- Memcache: a distributed in memory cache that'll let us significantly improve our applications response time by caching our most used data, as long as it doesn't change very often.
- Logs: as any other system, logging is crucial for the development and maintainability of our application. To cover this we have the log system, which will let us sort through the logs generated by all the traffic that comes to our application, as well as our own logs, separated by the usual priority levels.
- Taskqueue: we can also schedule tasks (in the form of web requests) to be executed at a specific time, or configure a queue to execute it's items in order, at a fixed rate.
- Mail: to let us communicate to our users, the App Engine offers an email system. We can send emails of any kind, and so so from any of the emails registered in the application.
- MapReduce: a popular feature for computation over large sets of data is the MapReduce, and Google offers a custom version for their App Engine users.

As we can see the feature set offered by the App Engine is really complete, and it'll greatly help us in the process of making our application, however to make use of its full potential we'll need to enable billing in our application, since there are a couple of resources that'll deplete very quickly. For example, one of the new features we introduced in our CourseBuilder version was an email confirmation when a new student registers for the course. We soon realized that the App Engine only allows you to send a hundred emails for free, even though the traffic generated by those people didn't use completely any other resource. However, even though we had enabled billing in our application, the emails weren't being sent, since that particular service didn't unlock until we made a payment, which are collected weekly.

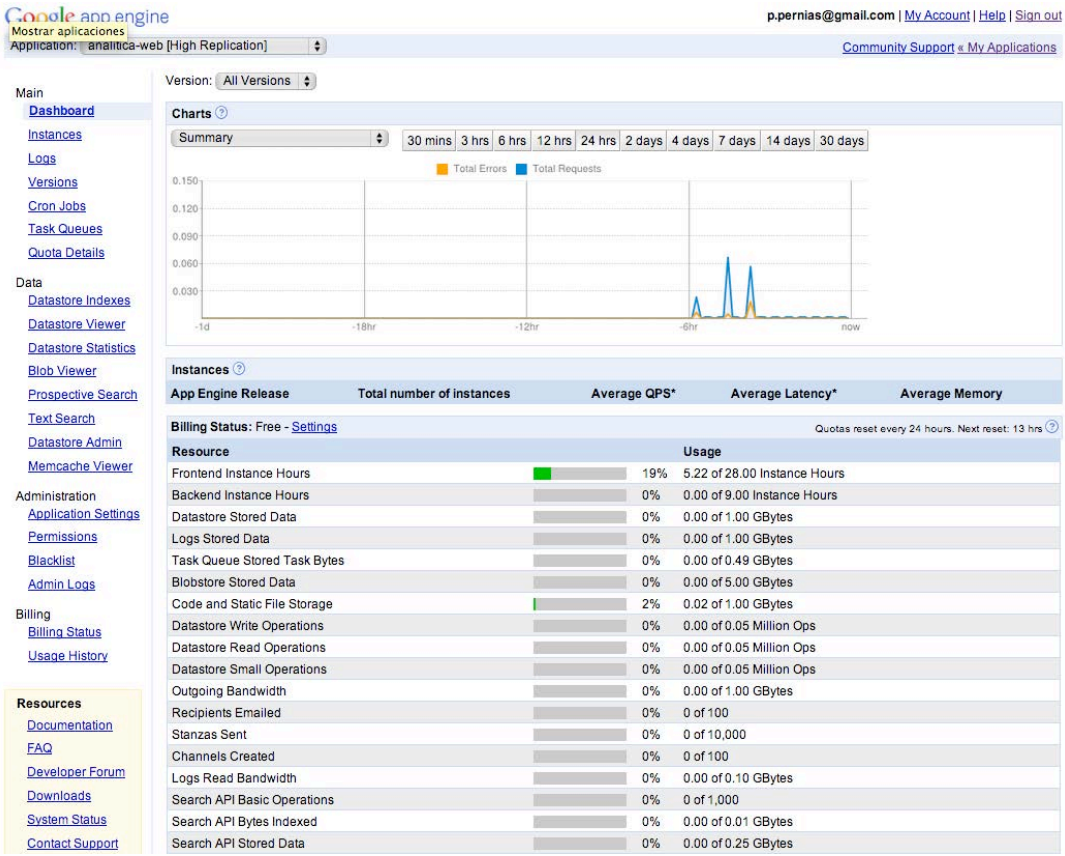

Fig. 1: Google's App Engine main screen

Besides the *Billing* menu option, there are a number of features we can (and should) configure in our application, and perhaps the most important one is located in the 'Performance' tab, under 'Application Settings'. Here we have total control over the way our application behaves under heavy traffic and in calm moments. Among other things we can change the amount of idle instances, the time to wait until a new instance is brought up and the type of memcache we want to use. However, it would be impossible to tell if our application is responding well to our configuration if there wasn't any way of monitoring our application, and for that purpose there is the Logs section. Here we can see the latest logs generated by our application, we can also filter by application version (or between the frontend application and the backends), minimum severity, date, etc. If we want to persistently store the logs we'll have to program it, since the App Engine will delete old logs as new ones get created when the space available for them gets filled up. We can also get quite a lot of information from the Dashboard section, where we'll find our current estimated cost (since the last reset) broken down by resource. We can also see the URLs that are getting the most requests (with information about their runtime and latency) as well as the which URLs are producing errors and how many of them.

To sum up, the Google App Engine proves to be an extremely efficient and easy to use platform for the development of state of the art web applications, and lets us take advantage of the latest technologies to reach to our users and give them the best experience.

4.2.3 Configuring Google App Engine

There are three basic parameters that we can tweak to adjust the App Engine to our needs. The default values are fine, but to really get out the most of it we need to modify them.

The first thing we have to make is open the administration console, which is located at appengine. google.com, and there we'll select our application. In the side menu you have to click on 'Application Settings'.

The first parameter we can change is the type of processor. There are a couple of different options, and they differ in the amount of RAM available and the potency of the processor. By default the most basic machine is selected. If we want to minimize cost we'll leave it as it is.

The next option we can modify is the idle instances, this parameter lets us choose how many instances are waiting to answer requests at any given time, both a minimum and a maximum. By default it's automatic, so as more and more request come in, more instances will be started and set to wait, however this can increase the cost. The bigger the value the faster our application's response time will be.

Finally, the last parameter we can change is the max idle time. This corresponds to how long will have to wait a request until an idle instance responds to it. By default request a wait minimum of 10ms. We can increase this value, which will mean users will experience a slower page load when there's heavy traffic. A small value will mean that users get answered earlier, but more instances will be started.

As we can see, the GAE allows us to configure our application behavior so that it responds how we want it to, and it spends as much as we tell it to.

4.3 How to install and use Google CourseBuilder script

4.3.1 Preparing the local installation of GBC

GBC is hosted by Google App Engine. This means that we have not to worry about rent / buy servers, install databases, load balancing, etc. App Engine manages everything for us automatically. Besides of it, it starts our local server and also gives us a simple way to try everything we add to the course in our own machine without having to upload it to the cloud, which is very useful when we add new content and need to test them before.

The process to have a GBC running in the Google App Engine starts by installing all the software necessary locally. It is necessary to install:

- Python 2.7
- AppEngine SDK and
- Google Appengine Launcher

The installation process is different depending on the local operative system but it is not difficult in any case.

Once the installation of this software has been completed, the next step is to start the GoogleAppengineLauncher application. *(We will use the Mac OS Version to illustrate the following steps)*

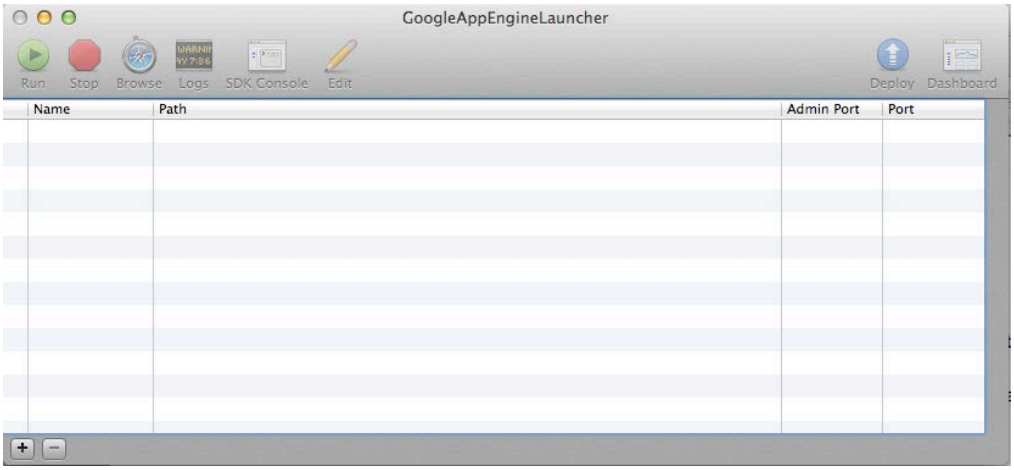

Fig. 2: GoogleAppEngineLauncher application main screen

The central window will show all the applications that we will have activated in our computer, not necessarily uploaded to the cloud system.

At the top row we can see a set of buttons. The most important buttons are:

- RUN: activate and play the selected application
- STOP: stop the execution of the selected application

- BROWSE: to use the default browser to access to the local server and have a look of the course we are building.
- EDIT: edit the app.yaml file with the main configuration data for deployment.
- DEPLOY: upload the course to the App Engine cloud account.

The next step is to download the last version of CourseBuilder (it is a zip file) and to unzip it on our computer.

Using the GoogleAppEngineLauncher, we click on “File>Add existing Application” and we select the folder we have unzipped with the GBC files. Finally we click on “ADD” and the files of the GBC will be activated in our local GoogleAppEngineLauncher.

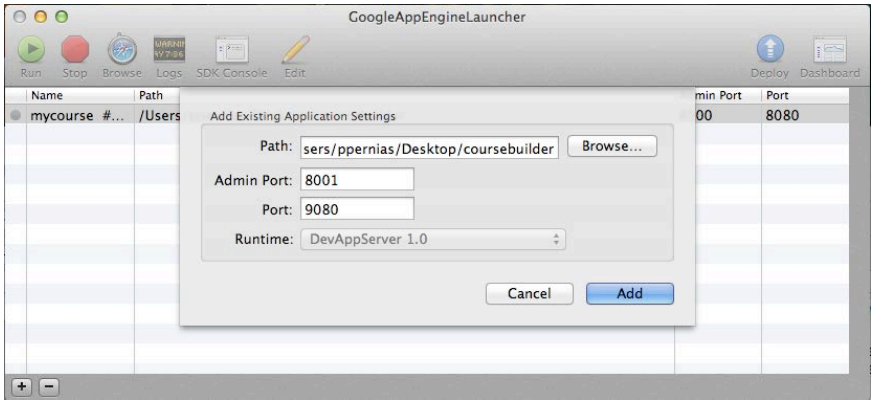

Fig. 3: adding course to GoggleAppEngineLauncher

Now, we click on RUN and BROWSE to see the example course running in our local server.

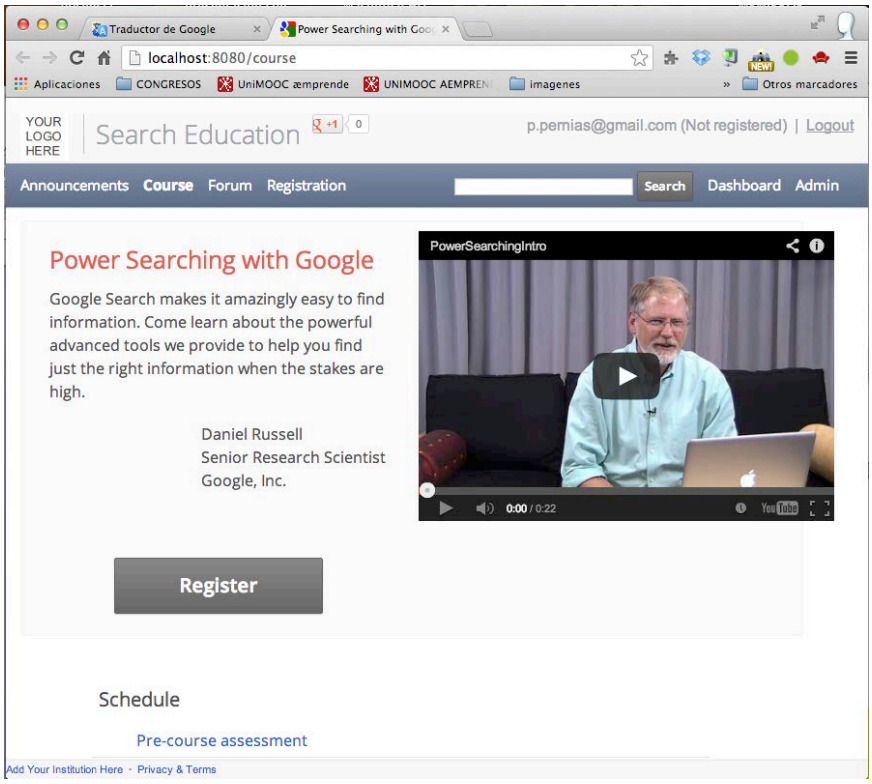

Fig. 4: example course running in local server mode

4.3.2 Personalization

In order to customize the course for our needs, we need to make changes in two separated files:

- course.yaml
- course_template.yaml

The file course.yaml is where we find all our static information required for the course. In this file we found several distinct sections.

First, we have the "*General Course Information*" section, where you can specify all the information of the course, as the title, the main message ... Fields can find in this section are:

- admin_users_emails: To specify emails from administrators of the course. The mails must be enclosed in brackets and separated by spaces.
- title: in order to modify the course title.
- blurb: The message shown on the main page, so you can use multiple lines in this field must begin with the character | and start on a new line. Also, we must also leave message tabulated using spaces as it is the format used files. yaml.
- instructor_details: for the information of the person who organizes the course.
- main_video and main_image: The initial video course we can configure in this section, we can also choose to use a video or simply specify an image.
- start_date: The date the course will be available to students
- forum_url and forum_embed_url: here you can specify the url to our forum, if you have created a forum for the course.
- locale: finally found the choice of language, here we indicate the current language to translate all links and different elements of the course.
- now_available: "true" if the course is open for registration
- browsable: whether the course material can be accessed without requiring users to log
- in or register

The followings sections allows to enter descriptive html code in order to complete the information available to the students at different moments of the course, like preview course page, registration page, post assessment page and units and lessons pages.

The file course_template.yaml allow us to change information about our institution and how the corporative information will appears in all CourseBuilder pages. It has too some sections:

Information about your institution section. This section, though short because as it only has three fields, is very important. Here is where we can change our course logo and the link to our institution in the footer page.

- name: the name of our institution is shown in the footer of the course.
- url: the web address of our institution, where we will link both the logo and the foot of the course.
- logo: url to our logo and other descriptive text about it.

And the "base page template" section, where we also found four fields to modify:

- show_gplus_button: whether or not to display the google plus button.
- nav_header: the title that appears next to the logo.
- privacy_terms_url: link to disclaimer page and related.
- locale: for internationalization of explorer pages.

Also, in this section, we can add tags and html code for learning analytics purposes. Now, we are ready to enter the course content.

4.3.3 Adding new content

To create a course, you have to add the contents (lessons, assessments, units, etc) to the platform. CourseBuilder has recently added a simple Content Management System but it has its own drawbacks: it needs to be edited on the platform (not offline), it slows the uploading process of adding new content (specially when you need to upload a huge number of resources), etc. So we are currently uploading content by the "old way", i.e. editing by hand some files which are later loaded to database.

For adding units and lessons, you need to edit two CSV (Comma Separated Values) files: unit.csv and lesson.csv. This is usually faster than adding a new lesson on the CMS, waiting for it to be uploaded, add another one, etc.

On the other hand, to add assessments and activities (sets of questions with minor importance), you need to create JSON files which have the information about the kind and text of each question. This can be sometimes frustrating but, as stated above, it is faster than using the CMS.

4.3.4 Deploying the course

The process of deploying a course made with GBC is quite simple. Google provides a launcher that allows us to deploy our source code to any App Engine application easily. To deploy an application first we need to download and install the App Engine launcher. Then, we'll need the CourseBuilder code in a folder in our machine. Finally we need to set up an account in the App Engine and, using that account, we will create a new application.

The first step now is to open the launcher and add our source code through the File > Add Existing Application menu. Now we need to edit the app.yaml file in our source code and change the application field to the id of the application we created, it should look like "application: myappid". Once we have our application added in the launcher we have to click on the Deploy button, this will bring up a window that asks for your email and password. This account is the one you used to create the application. Then, once you've filled both fields and clicked OK a new window with the deployment status will pop up. Once the deployment finishes (you'll see a message telling you that you can close the window) your application is ready to use: head to myappid.appspot.com (substituting myappid with your app ide) and see the changes you've made live.

4.3.5 Monitorizing the course

Once the course is running it is possible to monitor how the App Engine resources are being used. It is very useful to control the cost and to optimize the speed and the fluency of the course. Also, it is possible to have a log of the errors on sending files to the users, allowing stay alert to incidences of the course (Fig. 1).

5 CONCLUSIONS AND RECOMENDATIONS

Between 2012 and 2013, we have run a very big MOOC using GBC as platform. It is called UNIMOOC Aemprende and it is focused on fostering entrepreneurism in the context of the digital economy. At present it has 27 different units each one with his own assessment and accreditation. More than 25,000 students have been registered and more than 27,000 assessments have been accomplished by them [5].

From our experience, Google CourseBuilder has been a very robust system and the off-line time is near zero. Only some bugs due to typos in urls in the course content produced some difficulties.

Summarizing the positive aspect of using Google CourseBuilder we highlight:

- Very powerful control of the cost of running the course in the App Engine cloud system. It is possible to limit the daily expense.
- Flexibility to adapt to our growing needs. Using the Google App Engine Cloud system we have paid only the resources what we have used.
- The CourseBuilder python script is very easy to modify. The learning curve for developers is very fast and the developing of new features is very easy.
- The learning analytics that CourseBuilder provides are far enough for improving the learning process during running time.

On the other side, the main disadvantages on using CourseBuilder are:

- It is hard for teachers non-technically skilled to add educational content, specially activities and assessments. The content management system included in GBC is in a very early stage of development and the CSV files editing method implies to use JavaScript language not friendly for non programmers.
- Unless there exists a developer team, it is not possible to make changes to the shape of the course beyond the use of structure of unit and lessons. Also, the default configuration only allows to have three assessments and to increase the number of them is possible but requires deeper modifications. Some courses will not fit in this way to present the content.

Our final conclusion has been that Google's CourseBuilder is an excellent piece of software that allows us to configure a very robust system able to support a huge number of students at a reduced cost.

It is very easy to configure and to adapt to the needs of the MOOC's organizers and the possibility to extend the features of the software by creating new modules is very interesting for developer teams.

If the limitations of the CMS are not important for a specific case and can be overcome, it is a very recommendable software platform to create and implement a MOOC.

REFERENCES

[1] Pappano, L. (2012). The Year of the MOOC, The New York Times, November 2. [Online]. Available: http://www.nytimes.com/2012/11/04/education/edlife/massiveopen-online-courses-are-multiplying-at-a-rapid-pace.html

[2] Friedman, T. L. (2013). Revolution Hits the Universities, The New York Times, January 26. [Online]. Available: http://www.nytimes.com/2013/01/27/opinion/sunday/friedmanrevolution-hits-the-universities.html

[3] Kolowich, S. (2012). The MOOC 'Revolution' May Not Be as Disruptive as Some Had Imagined, The Chronicle of Higher Education, August 8. [Online]. Available: http://chronicle.com/article/MOOCs-May-Not-Be-So-Disruptive/140965/

[4] Google (2013). Course Builder. [Online] Available: https://code.google.com/p/course-builder/

[5] Unimooc Web Page (2013). [Online]. Available: http://unimooc.com/landing/?page=quequienes.html