Institiúid Teicneolaíochta Cheatharlach

INSTITUTE of TECHNOLOGY CARLOW

At the Heart of South Leinster

Research Document:
Code Editing in the Cloud

Author:

Alejandro Borrego Delgado (C00132731)

Final Project,

4th Bach (Honours) in Software Engineering

IT Carlow, 2009/2010

INDEX:

1. Introduction:

1.1. Definitions:

A **code editor** is a software application designed specifically for editing source code. Source codes editors usually implements different features to make easier for the programmer to implement new code, like syntax highlighting, auto-complete, etc.

Examples: Notepad++[1], SciTE[2], Emacs[3]...

An **IDE (integrated design environment)** is a software application that integrates a source code editor, among with other facilities for software developers like a file explorer, a compiler or a debugger.

Examples: Eclipse[4], Microsoft Visual Studio[5], anjuta[6]...

A **Software Version Control System** is a software application designed to manage a certain application version system, providing the software developers with a record of the source files from the different versions and a system for all the developers of a team to maintain the source code for their application updated.

Examples: CVS[7], SVN (Subversion)[8]...

Cloud Computing is an internet-based technology for software developing and use. This means that a software application would be executed on “the cloud” (on one or more remote machines) and the results for this execution would be displayed in a web browser. As web browsers are provided for every platform, Cloud Computing provides users with **platform free** applications.

1.2. What would be the application about?

The aim of this project is to build an IDE in the cloud. This IDE would include a code editor, a file explorer, and a compiler. Providing to the user with a tool that would help a team of developers to create a software project where everyone could work, and there is no need for Software Version Control System as all the files would be store and modified on the cloud. It would also build the application avoiding a need for installing a tool for that in a local or locals machines.

1.3. Who is this application destined to?

This application would be useful for software developers and engineers teams, that usually would need an IDE install in every machine, and a software version control system, with clients in every machine, and a central server. With our tool, the need for these tools would be removed, in stead, there would only be needed our application install in the server and a web browser in every machine.

2. Similar applications

2.1. CodeMirror

2.1.1. What is CodeMirror?

CodeMirror[9] is a **JavaScript library**, that allows developers to **embed** a code editor in any JavaScript application. It can be configured to fit the look and characteristics of the application. Code highlighter is provided for different languages:

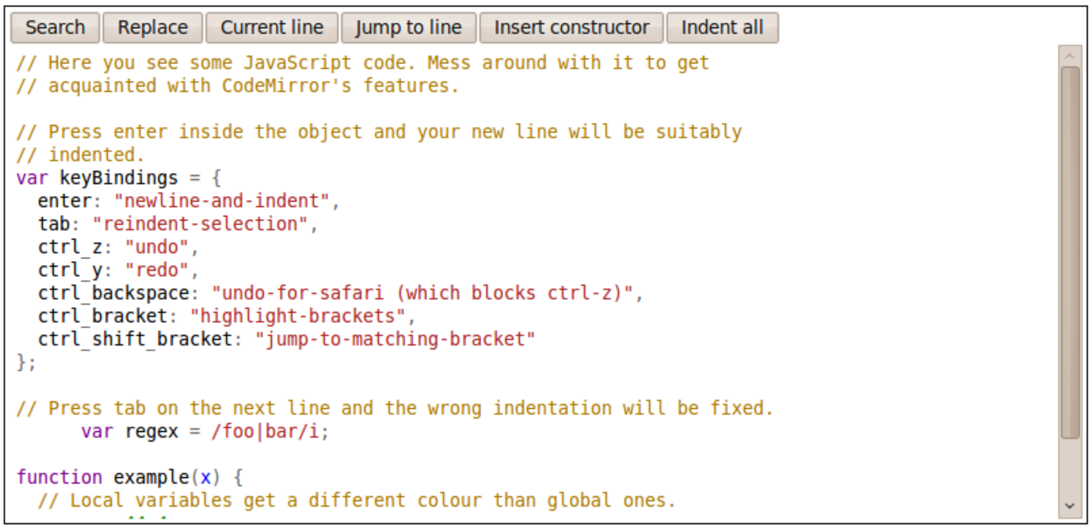
- JavaScript
- XML/HTML
- CSS
- SPARQL
- HTML & JavaScript
- HTML & PHP
- Python
- Lua
- Ruby
- SQL

2.1.2. How it is related to our application?

CodeMirror provides any web application with an editor. Our application would integrate a code editor in a IDE, so CodeMirror then would satisfy one of the requirements for the application.

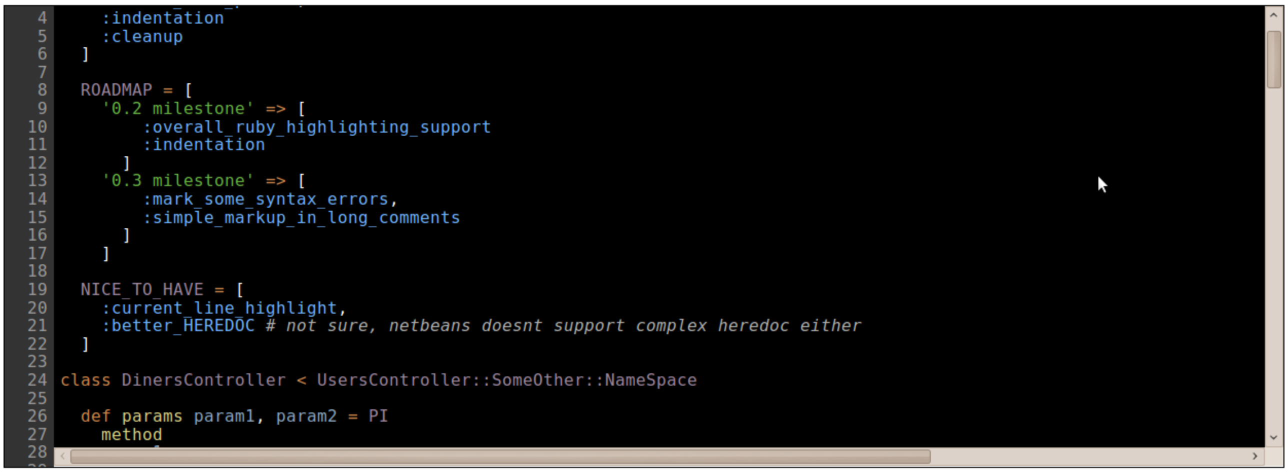
2.1.3. Examples:

The following images represents examples on some CodeMirror parsers and and application using CodeMirror embedded on it.



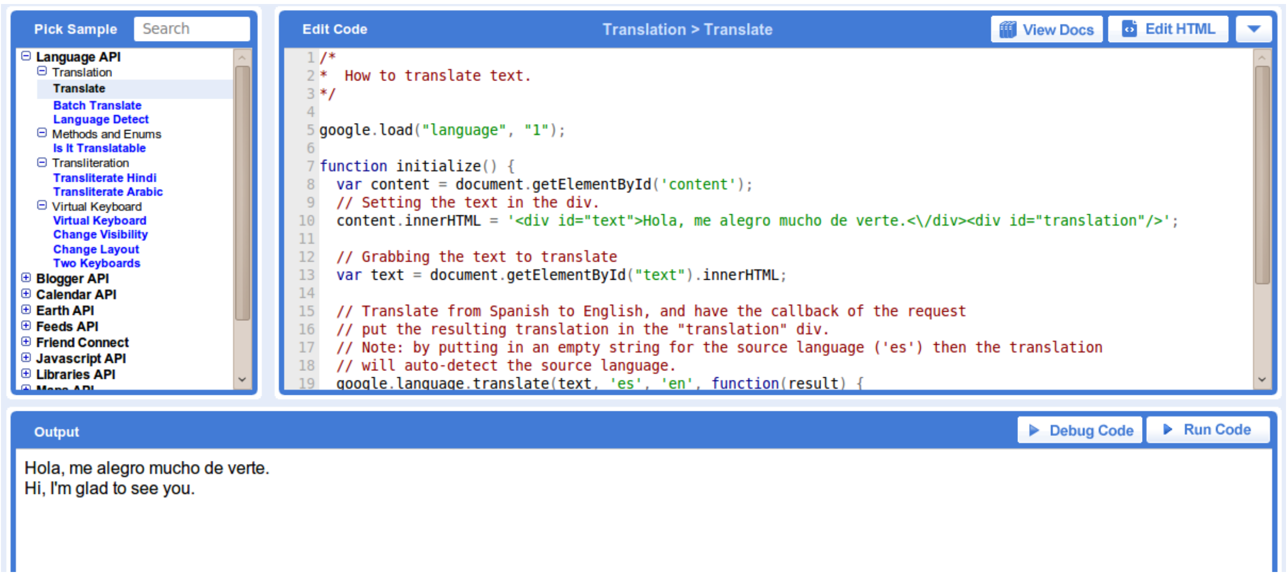
```
// Here you see some JavaScript code. Mess around with it to get
// acquainted with CodeMirror's features.

// Press enter inside the object and your new line will be suitably
// indented.
var keyBindings = {
  enter: "newline-and-indent",
  tab: "reindent-selection",
  ctrl_z: "undo",
  ctrl_y: "redo",
  ctrl_backspace: "undo-for-safari (which blocks ctrl-z)",
  ctrl_bracket: "highlight-brackets",
  ctrl_shift_bracket: "jump-to-matching-bracket"
};

// Press tab on the next line and the wrong indentation will be fixed.
      var regex = /foo|bar/i;

function example(x) {
  // Local variables get a different colour than global ones.
```

Figure 2.1.3.1: CodeMirror Highlighting JavaScript code.[10]



```
    :indentation
    :cleanup
  ]

  ROADMAP = [
    '0.2 milestone' => [
        :overall_ruby_highlighting_support
        :indentation
      ]
    '0.3 milestone' => [
        :mark_some_syntax_errors,
        :simple_markup_in_long_comments
      ]
    ]

  NICE_TO_HAVE = [
    :current_line_highlight,
    :better_HEREDOC # not sure, netbeans doesnt support complex heredoc either
  ]

class DinersController < UsersController::SomeOther::NameSpace

  def params param1, param2 = PI
    method
```

Figure 2.1.3.2: CodeMirror Highlighting ruby code.[11]



```
/*
*  How to translate text.
*/

google.load("language", "1");

function initialize() {
  var content = document.getElementById('content');
  // Setting the text in the div.
  content.innerHTML = '<div id="text">Hola, me alegro mucho de verte.<\/div><div id="translation"/>';

  // Grabbing the text to translate
  var text = document.getElementById("text").innerHTML;

  // Translate from Spanish to English, and have the callback of the request
  // put the resulting translation in the "translation" div.
  // Note: by putting in an empty string for the source language ('es') then the translation
  // will auto-detect the source language.
  google.language.translate(text, 'es', 'en', function(result) {
```

```
Hola, me alegro mucho de verte.
Hi, I'm glad to see you.
```

Figure 2.1.3.3: CodeMirror embedded in "Code Playground" application.[12]

2.2. CodeTextArea – Notapad

2.2.1. What is CodeTextArea and Notapad?

CodeTextArea as well as CodeMirror provides the developers with an **interface** to **edit code** in their applications.[13]

Notapad is a web application using CodeTextArea to provide an **IDE**. **File save** and **export** are provided with Notapad

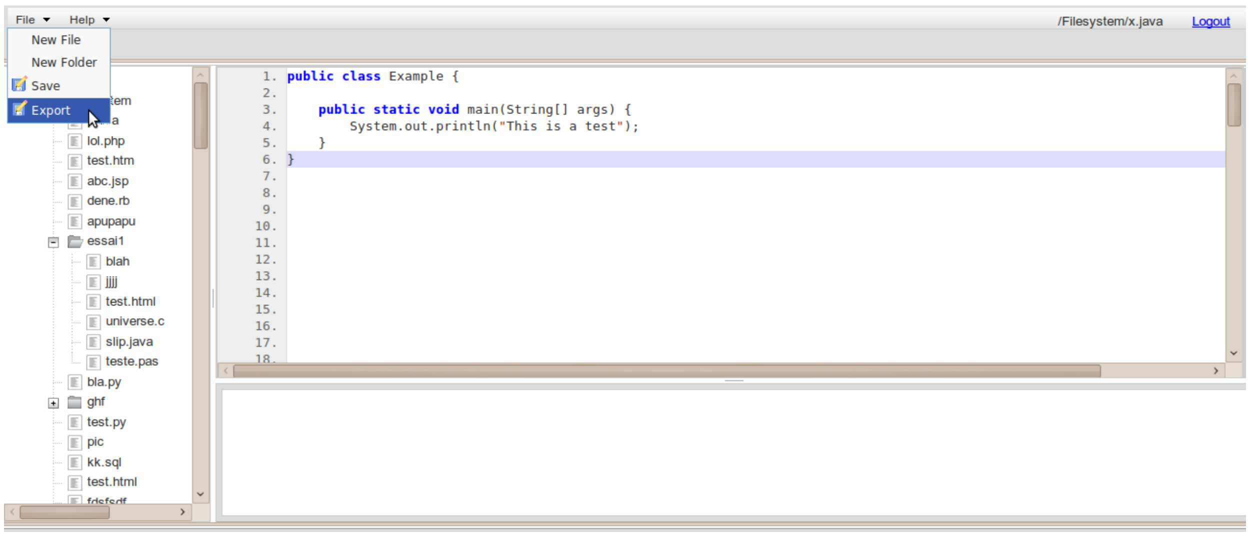
The quality of the code highlighter is not very good, but still, Notapad can gave as an idea of how an IDE on the cloud could be.

2.2.2. How it is related to our application?

Notapad is an IDE, it would perform some of the operations that are expected for our application, other than projects managing and compiling operations.

2.2.3. Example:

An example of Notapad is provided:



```java
public class Example {

    public static void main(String[] args) {
        System.out.println("This is a test");
    }
}
```

Figure 2.2.3.1: Notapad View [14]

2.3. Bespin

2.3.1. What would Bespin be?

Bespin is an "*open extensible web-based framework*" currently under development, "*for code editing that aims to increase developer productivity*"[16]

Bespin aim is to produce a development environment, with the following characteristics: "*Ease of Use*", "*Real-time Collaboration*", "*Integrated Command-Line*", "*Extensible and Self-Hosted*", "*Wicked Fas*", "*Accessible from Anywhere*". [15]

Mozilla Developer Tools Lab responsible for the development of Bespin has currently released an initial prototype "v0.1", that has provides software developers and engineers with the support for basic editing features, such as syntax highlighting, large file sizes, undo/redo, previewing files in the browser, importing/exporting projects, etc.

Bespin could be adapted to use with lost of technologies, and as is open source it is expected to be adapted to almost every web designed technology. [15][16][17][18]

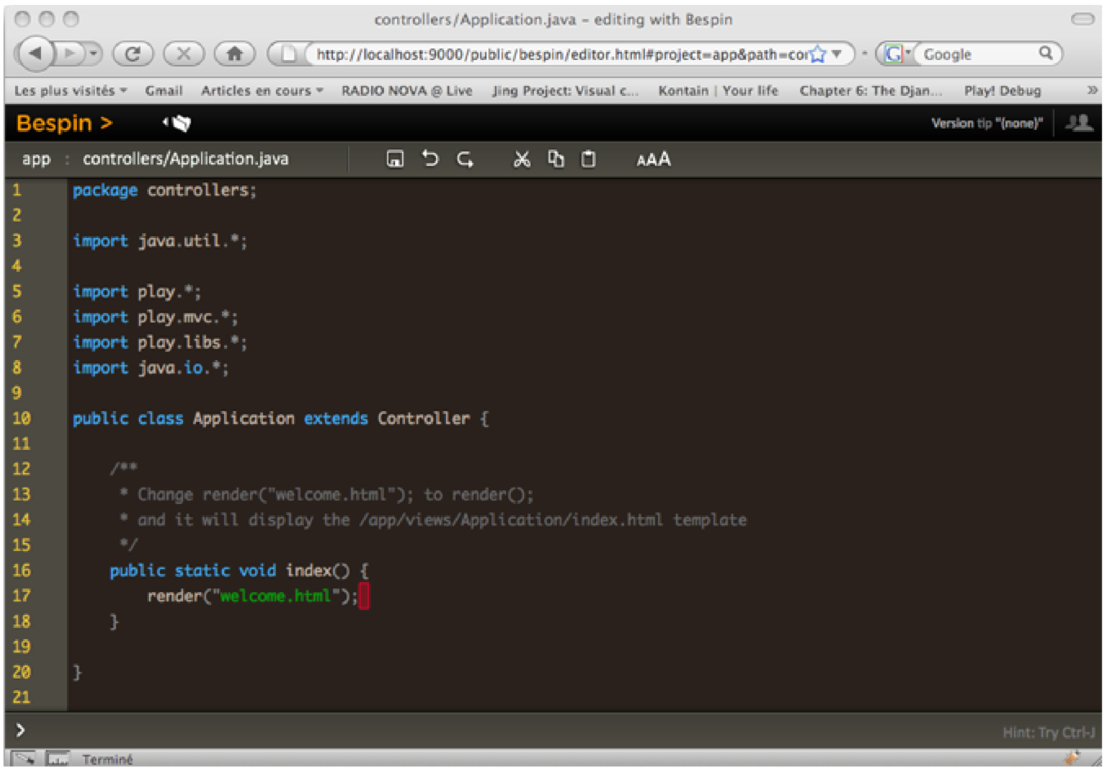
2.3.2. How it is related to our application?

By the time that Bespin would be completed, it will provide the developers with a very good framework to create web applications capable of editing and maintaining projects on the cloud.

Our application, would have the same features that Bespin is intending to achieve. Other than the real-time collaboration under the same file, which is not intended to be provided in a first approach to the problem. Adding to the solution a compilation tool.

2.3.3. Example:

The current version can be tested and some application snapshots are provided:


controllers/Application.java – editing with Bespin
http://localhost:9000/public/bespin/editor.html#project=app&path=co
Bespin >
Version tip "(none)"
app : controllers/Application.java
Hint: Try Ctrl-J
Terminé

```java
package controllers;

import java.util.*;

import play.*;
import play.mvc.*;
import play.libs.*;
import java.io.*;

public class Application extends Controller {

    /**
     * Change render("welcome.html"); to render();
     * and it will display the /app/views/Application/index.html template
     */
    public static void index() {
        render("welcome.html");
    }

}
```

Figure 2.3.3.1: Bespin Edit view.[16]

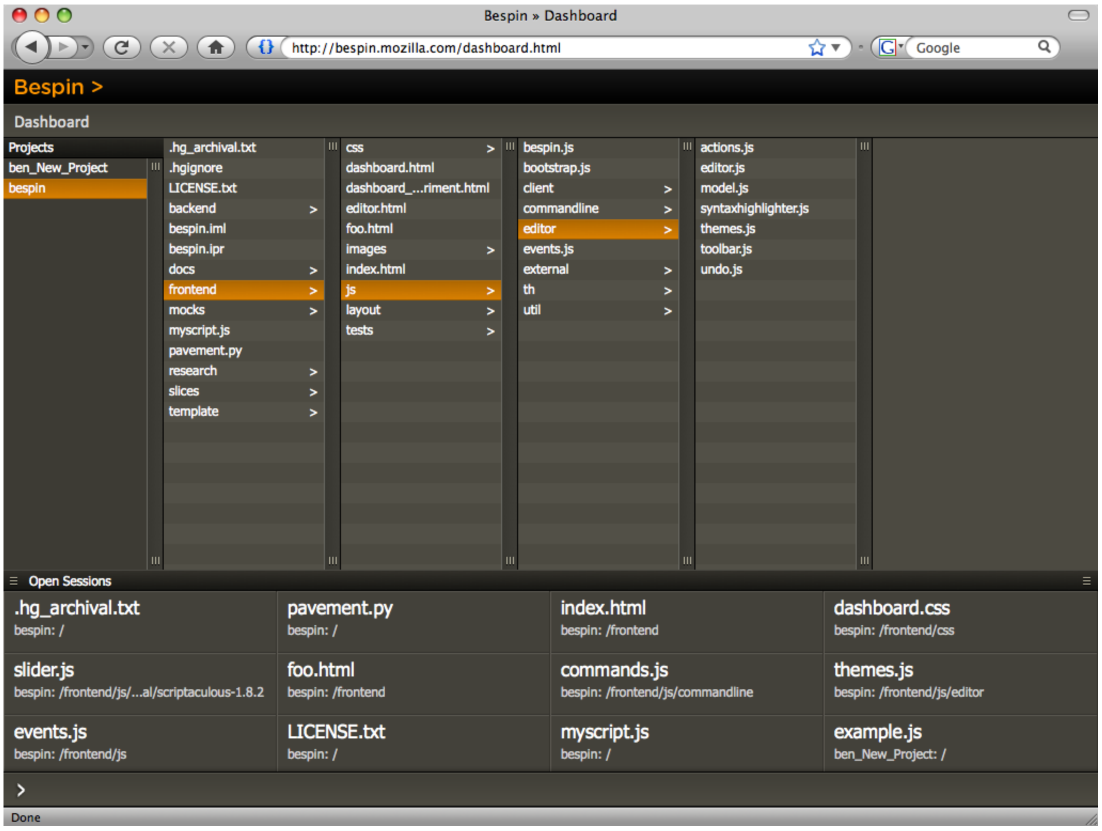


Figure 2.3.3.1: Bespin dashboard. [16]

3. Technologies

3.1. OpenUp:

3.1.1. UP / RUP

UP and **RUP** are iterative and incremental development process divided in **phases** (Inception, Elaboration, Construction and Transition). These phases are divided on **Iterations**. UP and RUP are **adaptive** models, as a result of that there is lots of variations from the originals RUP and UP. [19] [20]

3.1.2. Agile Software Development

Agile Software Development refers to those methodologies that follows the ideas From the **Manifesto For Agile Software Development**:

"*We are uncovering better ways of developing software by doing it and helping others do it. Through this work we have come to value:*

- *Individuals and interactions over processes and tools*
- *Working software over comprehensive documentation*
- *Customer collaboration over contract negotiation*
- *Responding to change over following a plan*

That is, while there is value in the items on the right, we value the items on the left more." [21]

The Agile Manifest encourage software engineers to develop software in a new way, forgetting about classic waterfall models, and getting better results.

3.1.3. OpenUP:

OpenUP is an **agile** methodology based on **UP** applying iterative and incremental approaches to software development that applies iterative and incremental approaches in a structured lifecycle. OpenUP can be **extended** to suit a broad variety of project types.

The project lifecycle is divided on **4 phases**: Inception, Elaboration, Construction, and Transition.

- **Inception:** "*The purpose in this phase is to achieve conjunction among all stakeholders on the lifecycle objectives for the project.*"[22]
- **Elaboration: "***The purpose of this phase is to establish the baseline of the architecture of the system and provide a stable basis for the bulk of the development effort in the next phase.*"[22]
- **Construction:** "*The purpose in this phase is to complete the development of the system based on the architecture defined in previous phases.*"[22]
- **Transition:** "*The purpose in this phase is to ensure that the software is ready for delivery to users.*"[22]

Each phase would be split in different **iterations**. An **iteration** is a "*set period of time within a project in which you produce a **stable**, executable **version** of the product, together with any other supporting documentation, install scripts, or similar, necessary to use this release*"[22]. Also referred to as a cycle. Every iteration is **planned** in advance so every objective would be fulfilled before the deadline.

Personal effort on an OpenUP project is organized in **micro-increments**. A micro-increment is a "***small, measurable step** towards reaching the goals of an iteration*"[22]. "*A micro-increment represents the outcome of a few hours to a few days of work performed by one person (or typically a few people collaborating) to reach a goal*"[22].

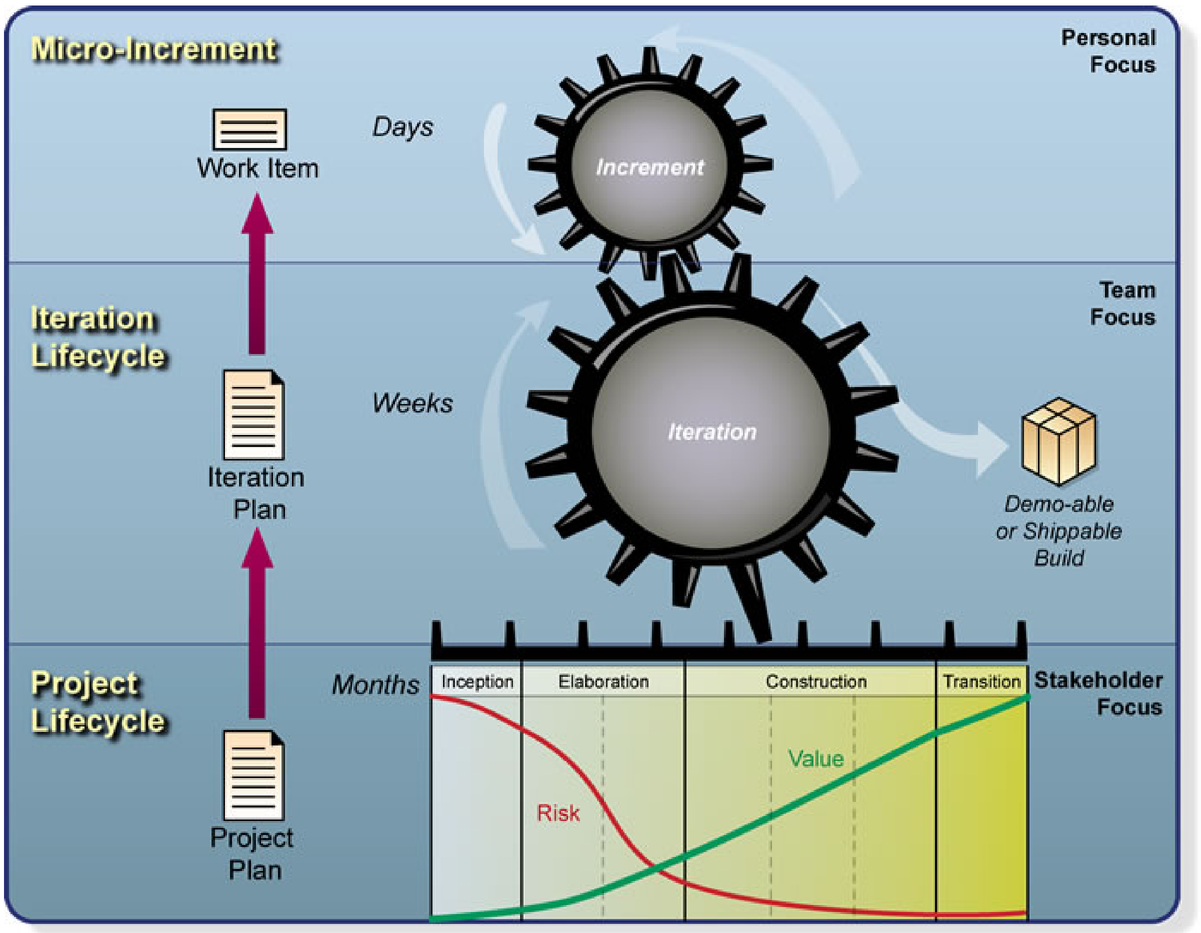


Figure 3.1.3.1: OpenUP layers: micro-increments, iteration life-cycle and project life-cycle.[22]

3.2. Ruby on Rails:

3.2.1. Ruby:

Ruby is a **general purpose object-oriented** programming language, originally inspired on perl and influenced by many programming languages: Smalltalk, Perl, Python, etc.

"Since its public release in 1995, Ruby has drawn devoted coders worldwide. In 2006, Ruby achieved mass acceptance". "With active user groups formed in the world's major cities and Ruby-related conferences filled to capacity".[23]

"*The TIOBE index, which measures the growth of programming languages, ranks Ruby as #10 among programming languages worldwide*". "*Much of the growth is attributed to the popularity of software written in Ruby, particularly the Ruby on Rails web framework*".[23][71]

As new programming languages ruby is interpreted, not compiled. The aim of new programming languages are to produce code in shorter time. Every day new technologies provides faster and more powerful equipment and software developers time are a valuable resource. Joining this two things, ruby language try to make things easier for software developers, so **programmers productivity** is improved by making a sacrifice in application use of resources, as computers would be more powerful everyday.

Ruby is seen as a flexible language, since it allows its users to freely alter its parts. Essential parts of Ruby can be removed or redefined, at will. Existing parts can be added upon. Ruby tries not to restrict the coder. Ruby incorporates a **gem** provider called **rubygems**. Gems are packaged software that would help us developing our application, and includes libraries, frameworks, etc. [23][24]

3.2.2. Rails:

"*Rails* *is an open source web application* ***framework*** *for Ruby [...] Rails was created in 2003 by David Heinemeier Hansson and has since been extended by the Rails core team, more than 1,400 contributors, and supported by a vibrant ecosystem.*"[25]

It is defined to be used within an agile programming methodology and follows **Model-View-Controller (MVC)** architecture. MVC split an application into three main parts: **Model**, that would control the data; **View**, the visual interfaces; and **Controller**, the logic of the application.[25][26]

3.2.3. Ruby on Rails (RoR):

Rails is provided by rubygems. Ruby on Rails philosophy is based on two principals: ***"Convention over configuration"***[25] and ***"don't repeat yourself"***.[25]

Convention over configuration means that if we use the framework as much as possible in a convent stile, it would only be needed to define those aspects that does not fit the conventional definition. In MVC development it is usually common to repeat some aspect of the code in Models, Views and Controllers. With RoR this information is only provided once.

"*Ruby on Rails features several tools intended to make commonplace development tasks easier --out of the box--*"[25]. Rails provides scaffolding that can automatically construct some of the models and views needed for a basic website. A simple ruby web server and Rake* build system are also included. By including these common tools with the Rails framework, a basic development environment is provided with all versions of the software. Ruby scripts are provided with RoR to generate Models Views and Controllers in an automatic way. Also scaffolding scripts are provided**.[27][28]

** "Rake, a simple ruby build program with capabilities similar to make.*

Rake has the following features:

- *Rakefiles (rake's version of Makefiles) are completely defined in standard Ruby syntax. No XML files to edit. No quirky Makefile syntax to worry about (is that a tab or a space?)*
- *Users can specify tasks with prerequisites.*
- *Rake supports rule patterns to synthesize implicit tasks.*
- *Flexible File lists that act like arrays but know about manipulating file names and paths.*

A library of prepackaged tasks to make building rakefiles easier."[32]

*** "Rails scaffolding is a quick way to generate some of the major pieces of an application. If you want to create the models, views, and controllers for a new resource in a single operation, scaffolding is the tool for the job".*[25]

3.3. Python and Django:

3.3.1. Python:

Python is a **general purpose object-oriented** programming language as well as ruby. Created in 1991, have evolved in one of the most used new programming languages. As others of these new programming languages, Python aim is similar to Ruby aim, increase developers productivity even at the expense of higher use of resources.

"*Python is a programming language that lets you work more quickly and integrate your systems more effectively [...] You can learn to use Python and see almost immediate gains in productivity and lower maintenance costs [...] Python is a multi platform programming language that runs on Windows, Linux/Unix, Mac OS X, and has been ported to the Java and .NET virtual machines*".[29]

Python Last version is 3.1.1[29]

3.3.2. Django:

"*Django* *is a high-level Python Web framework*".[30] It provides python programmers with a quick way of implementing web applications. Django is based on DRY principal (Don't Repeat Yourself) and could be compared with Rails as the have similar characteristics.

An example of Django power could be the following:

- "*Define your data models entirely in Python. You get a rich, dynamic database-access API for free (but you can still write SQL if needed).*"[30]
- *"Save yourself the tedious work of creating interfaces for people to add and update content. Django does that automatically."*[30]
- *"Design pretty, cruft-free URLs with no framework-specific limitations. Be as flexible as you like."*[30]
- *"Use Django's powerful, extensible and designer-friendly template language to separate design, content and Python code."*[30]
- *"Django has full support for multi-language applications, letting you specify translation strings and providing hooks for language-specific functionality".* [30]

3.3.3. Python and Django:

“*As a web developer, you want to program in a language that is powerful, clean, mature, and well-documented, and one that is equipped with a great standard library and a huge selection of high-quality third-party packages. You also want a web framework for that language--one with a vibrant, helpful community of users and developers, and one that is designed to function smoothly as an integrated stack, but whose components are loosely coupled so that you can make substitutions if circumstances require. In short, you want Python and you want Django*” [62]

The Conjunction of these two technologies could gave us a similar framework to the one provided by RoR. Python programs could be storage an used under google App engine*. Making easier to develop python programs and providing with features to use other technologies like AJAX.

* Google App engine “*Offers users the ability to build and host web applications on Google's infrastructure*”.

3.4. Google APIs

An **API (application programming interface)** is an interface that some software provides in order to allow developers interact with the application to build new software. APIs are provide for many of Google software, divided in two types:

Google Data APIs: Data APIs provides the developer with a simple standard protocol to read, write and modify data on the web. This are the currently data APIs provided by google: [34]

- **Google Apps:** "*provides business tools for communication and collaboration in the cloud. The Google Apps Administrative APIs enable you develop applications that automate administrative tasks and integrate with an existing infrastructure. This includes e-mail, documents, calendar, etc.*" [35]
- **Google Analytics:** *"It allows you to gather, view, and analyse data about your website traffic"*. [36]
- **Blogger, Google Base:** "*allows client applications to view and update Blog content*". [37]
- **Google Book Search:** "*it allows developers to integrate book search and book previews in a web site"*. [38]
- **Google Calendar:** "*Google Calendar offers many ways to create and share content on appointments*". [39]
- **Google Code Search:** "*web applications to search public source code for function definitions and sample code"*. [40]
- **Google Spreadsheets:** "*The Google Contacts APIs allow web applications to manage Google Contacts content"*. [41]
- **Google Document List:** "*allow web applications to upload and show documents to the cloud. The client application can request a list of documents form one user and show an existing document. Providing a web application with capabilities for managing text documents and spreadsheets to back up them and cooperate on line working throw the cloud. It also helps organizing the files by searching for documents base on keywords, etc."* [42]
- **Google Health:** "*allows applications to view and send health data about medical records, etc.*" [43]

- **Google Maps:** "*API allows client applications to view, store and update map data*". [44]
- **Google Notebook:** "*allows client applications to view public notebook content, public notebooks can be requested. It also helps keeping track of new content added to a public notebook that the user is interested in.*" [45]
- **Picassa Web Albums:** "*allows web applications to integrate photos and photo albums.*" [46]
- **Google spreadsheets:** "*allows to access worksheets data on the cloud.*" [47]
- **Webmaster Tools:** "*allows a client application to view sites, add and remove sites, verify site ownership, and submit and delete Sitemaps.*" [48]
- **YouTube API:** *"enable the develop to integrate YouTube's video content and functionality into a web application.*" [49]

AJAX APIs: This APIs provides the developer with the power to implement rich, dynamic web sites entirely in JavaScript and HTML. You can add a map to your site, a dynamic search box, or download feeds with just a few lines of JavaScript.

- **Google Maps API:** *"embed Maps as seeing on Google maps in your own web application with JavaScript."* [50]
- **AJAX Search API:** *"embed a simple, dynamic search box and display search results in your own web applications with JavaScript."* [51]
- **AJAX Feed API:** *"download any public Atom or RSS feed using only JavaScript."* [52]
- **Visualization API:** "Create visualizations and reporting applications that access structured data in a common format." [53]
- **AJAX Language API:** *"translate and detect the language of blocks of text within a web page using only Javascript"* [54]
- **AJAX Library API:** *"is a content distribution network and loading architecture for the most popular, open source JavaScript libraries."* [55]
- **Google Earth API:** *"embed a true 3D digital globe into your web application as seen on Google Earth."* [56]
- **Google AdSense API:** *"Provides Advertising to a web page."* [57]

3.4.1. Useful APIs for our project.

3.4.1.1. Google Document List:

"*Google Document List Data API allows client applications upload documents into Google Docs and show them as API feeds. [...] The Client Application can request a list of documents from an user and show an existing document content."*[42]

Examples of possible uses for the API would be the following:

- *"Upload text documents or spreadsheets in order to make a back up or cooperate on line to edit them."*[42]
- *"Search for all the documents that contains some specific key words, categories, or metadata."*[42]
- *"Create folders and move documents/folders in and out of folders."*[42]
- *"Modify the sharing permissions of documents and folders. The API allows sharing to individuals, group emails, or across an entire Google Apps domain."*[42]
- *Review and download a document's complete revision history."* [42]

Google provides Libraries for the following programming languages: Java, .Net, PHP, Python and objective-c; but it can also be used in other programming languages like Ruby or c++. [58] [59]

3.4.1.2. Google Apps:

App provides services for business such as Gmail, Google Calendar, Docs, etc. to be integrated in a web application. It also provides authentication control, user and groups managements and configuration facilities. For our application we would only be interested in Docs services.[60]

*"**Google Docs** provides the developers with on line documents and real-time collaboration Web-based documents that let users edit the same file at the same time so the last version is always available [...] Google Docs is securely powered by the web, giving you the flexibility to be productive from anywhere. Google Docs works in the browser on PC, Mac, and Linux computers. Administrators can manage file sharing permissions system-wide, and document owners can share and revoke file access at any time"*[61].

3.4.2. How to use them in our application

For our application, we could use both google APIs to store the files for a project, although it would be strange, because this two APIs are ready to store word processor formats, not plain text documents. The only facilities would be, storing files on Google servers instead on our own server and managing users accounts and log-in system (that would be provided by google).

3.5. HTML, JavaScript and XML: AJAX

3.5.1. HTML:

HTML (HyperText Markup Language) is a markup language for web pages. That means that it is a special kind of text document that is used by Web browsers to present text and graphics. HTML web pages can be written by hand but there is lots of HTML editors. The results can be displayed on a browser and if stored on the internet it can be accessed from anywhere.[63]

3.5.2. JavaScript:

It is an open source interpreted scripting language that was designed to add interactivity to HTML pages by embedding it directly into HTML pages. JavaScript is the most used scripting language. It provides significantly increased functionality to web pages allowing for a more interactive user experience, such us:

- "*Adding dynamic text into an HTML page. JavaScript can write a variable text into an HTML page."* [64]
- *"Scripts can be set to execute when as a react to an event like when a page has finished loading or when a user clicks on an HTML element."* [64]
- *"Use HTML elements in a dynamic way."* [64]
- *"Check data in order to validate it, without having to be processed on the server."* [64]
- *Manage Cookies"* [64] *

* "*Cookies are a general mechanism which server side connections (such as CGI scripts) can use to both store and retrieve information on the client side of the connection. The addition of a simple, persistent, client-side state significantly extends the capabilities of Web-based client/server applications.*

To put it more plainly, a cookie is a mechanism that allows a web site to record your comings and goings, usually without your knowledge or consent." [65]

3.5.3. XML:

"*Extensible Markup Language (XML) is a simple, very flexible text format developed by world wide web consortium. In opposition to other Markup languages like HTML, tags are not predefined, you must define your own tags*" [66]. XML is not a replacement for HTML. In most web applications, XML is used to transport data, while HTML is used to format and display the data.

If you need to display dynamic data in your HTML document, it will take a lot of work to edit the HTML each time the data changes. "*With XML, data can be stored in separate XML files. This way you can concentrate on using HTML for layout and display, and be sure that changes in the underlying data will not require any changes to the HTML. With a few lines of JavaScript, you can read an external XML file and update the data content of your HTML*". [66]

"With XML, data can easily be exchanged between incompatible systems. One of the most time-consuming challenges for developers is to exchange data between incompatible systems over the Internet. Exchanging data as XML greatly reduces this complexity, since the data can be read by different incompatible applications."[66]

"Upgrading to new systems (hardware or software platforms), is always very time consuming. Large amounts of data must be converted and incompatible data is often lost. XML data is stored in text format. This makes it easier to expand or upgrade to new operating systems, new applications, or new browsers, without losing data."[66]

Since XML is independent of hardware, software and application, XML can make your data more available and useful. Different applications can access your data, not only in HTML pages, but also from XML data sources. With XML, your data can be available to all kinds of "reading machines" (Handheld computers, voice machines, news feeds, etc), and make it more available for blind people, or people with other disabilities. [66]

3.5.4. AJAX

"*AJAX is not a new programming language, but a new technique for creating better, faster, and more interactive web applications using JavaScript and XML. With AJAX, a JavaScript can communicate directly with the server, trading data with it, without reloading the page. AJAX uses asynchronous data transfer between the browser and the web server, allowing web pages to request small bits of information from the server instead of whole pages. The AJAX technique makes Internet applications smaller, faster and more user-friendly.*" [67]

AJAX is based on JavaScript, XML, HTML and CSS.

"*To get or send information from/to a database or a file on the server with traditional JavaScript, you will have to make an HTML form, and a user will have to click the "Submit" button to send/get the information, wait for the server to respond, then a new page will load with the results. Because the server returns a new page each time the user submits input, traditional web applications can run slowly and tend to be less user-friendly. With AJAX, your JavaScript communicates directly with the server, a web page can make a request to, and get a response from a web server - without reloading the page. The user will stay on the same page, and he or she will not notice that scripts request pages, or send data to a server in the background.*"[67]

AJAX is platform independent, and can be used in many browsers.

AJAX can be easily integrated in many web application written in different programming languages, sometimes more obvious like Ruby on Rails or not like Django. [67][68][69]

3.6. Other probably useful technologies (Brief explanation):

3.6.1 Perl:

Perl is a general-purpose, interpreted programming language.

Pearl power with regular expressions is the most interesting feature for our project.

Code highlighting is one of the features for our application, and a technology that provides a strong way to manage regular expressions could be useful for the project. [33]

3.6.2. PHP:

"***PHP*** *which stands for "PHP: Hypertext Preprocessor" is a widely-used Open Source general-purpose scripting language that is especially suited for Web development and can be embedded into HTML. The main goal of the language is to allow web developers to write dynamically generated web pages quickly, but you can do much more with PHP.*" [70]

3.6.3. Google App Engine:

Google App Engine, is an application, that allows Python and Java users to develop and host web application inside google's infrastructure. App Engine solves the problem of finding a host to keep our application. [31]

3.6.4. Google PlayGround:

Google Playground is a tool to test Google API's and applications using them. [12]

4. Choice of technology for project

4.1. Text editing:

So far, for all the research done, the clearest thing is to use AJAX to display files on screen. AJAX can communicate with the server without having to refresh the screen, that means that code can be automatically highlighted without having to specifically press any button. In order to do this, AJAX would manage the event of writing an space, tab, etc. (ending to write a new word) and would highlight it property.

4.2. Storage of the files for the project.

The google APIs seen in this document that allows as to store documents on the cloud are Google App and Google Documents List. The aim of those APIs is not to store plain text but to store word processor. Storing a plain text fail in a remote machine (as a text fail, or as database tables) is not difficult. How can it be stored in a database using a Ruby on Rails application is part of the Appendix on this document.

For all this reasons the decision is not to use Google and store files in a database, improving in that way the security, being our application the only platform to access to the files.

4.3. Programming Language and Framework:

As google API's are not going to be used, Google playground is something that we won't need.

Perl provides a solution for a single problem, managing regular expressions, but make it difficult in other aspect, like interfaces.

PHP, Python and Django, and Ruby on Rails, are the most likely technologies to be used. As Django and Rails are frameworks that would make our job easier, PHP would be discarded

As AJAX is the choice of technology on displaying files on screen, it is natural to think that the best solution is using Ruby on Rails. AJAX can be easily integrated on Ruby on Rails, adding this to the fact that I have a little experience using Ruby on Rails, using this technology would reduce some risks.

By not using Python or Google APIs, Google App engine would not be needed.

5. APPENDIX: Prototype in how to store and generate a jar file using system calls on Ruby on Rails.

This is a simple prototype with a simple interface, generated with scaffolding technologies and editing it to show how to generate a jar file:

prototype.app.application_controller.rb

```
# Filters added to this controller apply to all controllers in the application.
# Likewise, all the methods added will be available for all controllers.

class ApplicationController < ActionController::Base
  helper :all # include all helpers, all the time
  protect_from_forgery # See ActionController::RequestForgeryProtection for details

  # Scrub sensitive parameters from your log
  # filter_parameter_logging :password
end
```

prototype.app.my_files_controller.rb

```
class MyFilesController < ApplicationController
  # GET /my_files
  # GET /my_files.xml
  def index
    @my_files = MyFile.all

    respond_to do |format|
      format.html # index.html.erb
      format.xml  { render :xml => @my_files }
    end
  end

  # GET /my_files/1
  # GET /my_files/1.xml
  def show
    @my_file = MyFile.find(params[:id])

    respond_to do |format|
      format.html # show.html.erb
      format.xml  { render :xml => @my_file }
    end
  end

  # GET /my_files/new
  # GET /my_files/new.xml
  def new
    @my_file = MyFile.new

    respond_to do |format|
      format.html # new.html.erb
      format.xml  { render :xml => @my_file }
    end
  end

  # GET /my_files/1/edit
  def edit
    @my_file = MyFile.find(params[:id])
  end

  # POST /my_files
  # POST /my_files.xml
  def create
    @my_file = MyFile.new(params[:my_file])

    respond_to do |format|
      if @my_file.save
        flash[:notice] = 'MyFile was successfully created.'
        format.html { redirect_to(@my_file) }
        format.xml  { render :xml => @my_file, :status => :created, :location =>
@my_file }
      else
        format.html { render :action => "new" }
        format.xml  { render :xml => @my_file.errors, :status => :unprocessable_entity }
      end
    end
  end

  # PUT /my_files/1
  # PUT /my_files/1.xml
  def update
    @my_file = MyFile.find(params[:id])

    respond_to do |format|
      if @my_file.update_attributes(params[:my_file])
        flash[:notice] = 'MyFile was successfully updated.'
        format.html { redirect_to(@my_file) }
        format.xml  { head :ok }
      else
        format.html { render :action => "edit" }
        format.xml  { render :xml => @my_file.errors, :status => :unprocessable_entity }
      end
    end
  end

  # DELETE /my_files/1
  # DELETE /my_files/1.xml
  def destroy
    @my_file = MyFile.find(params[:id])
    @my_file.destroy

    respond_to do |format|
      format.html { redirect_to(my_files_url) }
      format.xml  { head :ok }
    end
  end

  def compile
    @my_file = MyFile.find(params[:id])
    @title = @my_file.title
    @code = @my_file.code
    @f = File.open("public/generated/"+@title+".java",'w+')
    @f.write(@code)
    @f.close()
    @x = system("public/bash/script.sh "+@title)
  end
end
```

prototype.app.models.my_file.rb

```
class MyFile < ActiveRecord::Base
end
```

prototype.app.views.my_files.compile.html.rb

```
<%if @x then%>
<p>
  <b>File <%=h @title%>.jar generated</b>
</p>
<%else%>
<p>
  <b>Error generating <%=h @title %>.jar file</b>
</p>
<%end%>

<%= link_to 'Back', my_files_path %>
```

prototype.app.views.my_files.edit.html.rb

```
<h1>Editing my_file</h1>
```

```
<% form_for(@my_file) do |f| %>
  <%= f.error_messages %>

  <p>
    <%= f.label :title %><br />
    <%= f.text_field :title %>
  </p>
  <p>
    <%= f.label :code %><br />
    <%= f.text_area :code %>
  </p>
  <p>
    <%= f.submit 'Update' %>
  </p>
<% end %>

<%= link_to 'Show', @my_file %> |
<%= link_to 'Back', my_files_path %>
```

prototype.app.views.my_files.index.html.rb

```
<h1>Listing my_files</h1>


<table>
  <tr>
    <th>Title</th>
    <th>Code</th>
  </tr>

<% @my_files.each do |my_file| %>
  <tr>
    <td><%=h my_file.title %></td>
    <td><%=h my_file.code %></td>
    <td><%= link_to 'Show', my_file %></td>
    <td><%= link_to 'Edit', edit_my_file_path(my_file) %></td>
    <td><%= link_to 'Destroy', my_file, :confirm => 'Are you sure?', :method => :delete
%></td>
       <td><%= link_to 'Compile', :action => 'compile', :id => my_file.id %></td>
  </tr>
<% end %>
</table>

<br />

<%= link_to 'New my_file', new_my_file_path %>
```

prototype.app.views.my_files.new.html.rb

```
<h1>New my_file</h1>


<% form_for(@my_file) do |f| %>
  <%= f.error_messages %>

  <p>
    <%= f.label :title %><br />
    <%= f.text_field :title %>
  </p>
  <p>
    <%= f.label :code %><br />
    <%= f.text_area :code %>
  </p>
  <p>
    <%= f.submit 'Create' %>
  </p>
<% end %>

<%= link_to 'Back', my_files_path %>
```

prototype.app.views.my_files.show.html.rb

```
<p>
  <b>Title:</b>
  <%=h @my_file.title %>
</p>

<p>
  <b>Code:</b>
  <%=h @my_file.code %>
</p>

<%= link_to 'Edit', edit_my_file_path(@my_file) %> |
<%= link_to 'Back', my_files_path %>
```

prototype.config.database.yml

```
# SQLite version 3.x
#   gem install sqlite3-ruby (not necessary on OS X Leopard)
development:
  adapter: mysql
  database: prototype
  host: localhost
  username: prototype
  password: prototype
  encoding: utf8

# Warning: The database defined as "test" will be erased and
# re-generated from your development database when you run "rake".
# Do not set this db to the same as development or production.
test:
  adapter: sqlite3
  database: db/test.sqlite3
  pool: 5
  timeout: 5000

production:
  adapter: mysql
  database: prototype
  host: localhost
  username: prototype
  password: prototype
  encoding: utf8
```

prototype.config.20091105022210_create_my_files.rb

```
class CreateMyFiles < ActiveRecord::Migration
  def self.up
    create_table :my_files do |t|
      t.string :title
      t.text :code

      t.timestamps
    end
  end

  def self.down
    drop_table :my_files
  end
end
```

protoype.public.bash.script.sh

```
#!/bin/bash
cd public/generated
echo "Main-Class: $1" > manifest
javac "$1.java"
jar cfm "$1.jar" manifest "$1.class"
```

6. Bibliography and References

[1] Notepad home web page, http://notepad-plus.sourceforge.net

[2] Scite home web page, http://www.scintilla.org/SciTE.html

[3] GNU Emacs development page, *Greg Harvey,* http://www.gnu.org/software/emacs/

[4] Eclipse home web page, *The Eclipse Foundation*, http://www.eclipse.org/

[5] Microsoft Visual Studio, *Microsoft Corporation,* http://www.microsoft.com/spain/visualstudio/default.mspx

[6] Anjuta development web page, http://projects.gnome.org/anjuta/index.shtml

[7] CVS, *Derek Robert Price & Ximbiot*, http://www.nongnu.org/cvs/

[8] SVN (Subversion), *CollabNet*, http://subversion.tigris.org/

[9] CodeMirror Home page, http://marijn.haverbeke.nl/codemirror/

[10] CodeMirror JavaScript example, http://marijn.haverbeke.nl/codemirror/jstest.html

[11] CodeMirror Ruby example, http://d.hantl.cz/d/ruby-in-codemirror/

[12] Google Playground, http://code.google.com/apis/ajax/playground/

[13] CodeTextArea Development page, *Nicola Rizzo,* http://code.google.com/p/codetextarea/updates/list

[14] Notapad Test Page, http://www.notapad.org/notapad.php

[15] Bespin Introduction web page, *Mozilla*, https://mozillalabs.com/blog/2009/02/introducing-bespin

[16] Bespin Home page, *Mozilla*, https://mozillalabs.com/bespin/

[17] Bespin first preview, *Mozilla*, http://mozillalabs.com/bespin/2009/11/13/bespin-embedded-first-preview-release/

[18] Bespin on rails, *Mozilla*, http://github.com/provideal/bespin-on-rails

[19] Rational UP web page, http://www-01.ibm.com/software/awdtools/rup/

[20] "El Proceso Unificado de Desarrollo de Software", *Ivar Jacobson, Grady Booch and James Rumbaugh*; Pearson Addisson-Wesley.

[21] Manifesto for Agile Software Development, http://agilemanifesto.org/

[22] OpenUP wiki, *OpenUP*, http://epf.eclipse.org/wikis/openup/

[23] Ruby language web page, http://www.ruby-lang.org/en/

[24] Rubygems repository, http://docs.rubygems.org/

[25] Ruby on Rails home page, http://rubyonrails.org/

[26] Model-View-Controller in Java blueprints, *Sun Microsystems*, http://java.sun.com/blueprints/patterns/MVC-detailed.html

[27] “Ruby on Rails”, *Bruce A. Tate y Curt Hibbs*; Anaya-Multimedia O'Reilly.

[28] Ruby on Rails tutorial, http://guides.rubyonrails.org/getting_started.html#getting-up-and-running-quickly-with-scaffolding

[29] Python web page, *Python software foundation*, http://www.python.org/

[30] Django web page, *Django software foundation*, http://www.djangoproject.com/

[31] Google app engine, *Google*, http://code.google.com/intl/en/appengine/

[32] Rake documentation page, *Jim Weirich*, http://rake.rubyforge.org/

[33] Perl regular expressions tutorial, *Steve Litt*, http://www.troubleshooters.com/codecorn/littperl/perlreg.htm

[34] Google Data, *Google*, http://code.google.com/intl/en/apis/gdata/

[35] Google App documentation page, *Google,* http://code.google.com/intl/en/apis/apps/

[36] Google Analytics home page, *Google*, http://code.google.com/intl/en/apis/analytics/

[37] Google Blogger API home page, *Google*, http://code.google.com/intl/en/apis/blogger/

[38] Google Books API home page, *Google*, http://code.google.com/intl/en/apis/books/

[39] Google Calendar API home page, *Google*, http://code.google.com/intl/en/apis/calendar/

[40] Google Code Search API home page, *Google*, http://code.google.com/intl/en/apis/codesearch/

[41] Google Contacts API, *Google*, http://code.google.com/intl/en/apis/contacts/

[42] Google Documents List API, *Google*, http://code.google.com/intl/en/apis/documents/overview.html

[43] Google Health API, *Google*, http://code.google.com/intl/en/apis/health/

[44] Google Maps Data API, *Google*, http://code.google.com/intl/es/apis/maps/documentation/mapsdata/

[45] Google Notebook API, *Google*, http://code.google.com/intl/en/apis/notebook/

[46] Google Picassa Web Albums API, *Google*, http://code.google.com/intl/en/apis/picasaweb/overview.html

[47] Google Spreadsheet API, *Google*, http://code.google.com/intl/en/apis/spreadsheets/

[48] Google Webmaster Tools API, *Google*, http://code.google.com/intl/en/apis/webmastertools/

[49] Youtube API, *Google*, http://code.google.com/intl/en/apis/youtube/overview.html

[50] Google Maps API, *Google*, http://code.google.com/intl/en/apis/maps/

[51] Google AJAX Search API, *Google*, http://code.google.com/intl/en/apis/ajaxsearch/

[52] Google AJAX Feed API, *Google*, http://code.google.com/intl/en/apis/ajaxfeeds/

[53] Google Visualizations API , *Google*, http://code.google.com/intl/en/apis/visualization/

[54] Google AJAX Language API, *Google*,
http://code.google.com/intl/en/apis/ajaxlanguage/
[55] AJAX Libraries API, *Google*, http://code.google.com/intl/en/apis/ajaxlibs/
[56] Google Earth API, *Google*, http://code.google.com/intl/en/apis/earth/
[57] Google AdSense API, *Google*, http://code.google.com/intl/en/apis/friendconnect/
[58] Google Documents List Data API samples, *Google*,
http://code.google.com/intl/en/apis/documents/code.html
[59] Using Ruby with Google Data API, *Jochen Hartmann & Google*,
http://code.google.com/intl/en/apis/gdata/articles/using_ruby.html
[60] Google App, *Google*, http://www.google.com/apps/intl/en-GB/business/index.html
[61] Google App docs, *Google*, http://www.google.com/apps/intl/en/business/docs.html
[62] "Python Web Development with Django", *Jeff Forcier, Paul Bissex, and Wesley Chun*, Developer's Library.
[63] Getting started with HTML, *Dave Raggett*, http://www.w3.org/MarkUp/Guide/
[64] JavaScript Introduction, http://www.w3schools.com/JS/js_intro.asp
[65] Cookies, *EPIC*, http://epic.org/privacy/internet/cookies/
[66] Introduction to XML, http://www.w3schools.com/xml/xml_whatis.asp
[67] AJAX Introduction, http://www.w3schools.com/Ajax/ajax_intro.asp
[68] AJAX on Rails, *ONLap*, http://onlamp.com/onlamp/2005/06/09/rails_ajax.html
[69] Django and AJAX, *James Bennett*, http://www.b-list.org/weblog/2006/jul/02/django-and-ajax/
[70] PHP Manual, *PHP Documentation Group*, http://www.php.net/manual/en/
[71] TIOBE index, *TIOBE*,
http://www.tiobe.com/index.php/content/paperinfo/tpci/index.html