



Conference on ENTERprise Information Systems / International Conference on Project
MANagement / Conference on Health and Social Care Information Systems and Technologies,
CENTERIS / ProjMAN / HCist 2015 October 7-9, 2015

Selecting an Open-Source Framework: A practical case based on software development for sensory analysis

Leonor Teixeira^{a,b,*}, Ana Raquel Xambre^{a,c}, Helena Alvelos^{a,c}, Nelson Filipe^b, Ana Luísa
Ramos^{a,d}

^aDepartment of Economics, Management and Industrial Engineering, University of Aveiro, 3810-193 Aveiro, Portugal

^bIIEETA - Institute of Telematics and Electronic Engineering of Aveiro, University of Aveiro, 3810-193 Aveiro, Portugal

^cCIDMA - Center for Research and Development in Mathematics and Applications, University of Aveiro, 3810-193 Aveiro, Portugal

^dGOVCOPP - Governance, Competitiveness and Public Policies, University of Aveiro, 3810-193 Aveiro, Portugal

Abstract

The use of frameworks based on Free Open-Source Software (FOSS) has become a viable alternative in the software development process, when compared with Proprietary Software or Closed Source Software. Given that the quality of Open-Source Software (OSS) products varies widely, the careful evaluation of such frameworks, according to a set of requirements, is an important step in the software development process. This work presents the evaluation of some open-source frameworks in order to find the most suitable one for developing a Decision Support System (DSS) to use in Sensory Analysis. This DSS is being designed to support the evaluation of Tasting Panels in sectors where Sensory Analysis is used to assess products' quality. The methodology used, based on content analysis in Online Collaborative Spaces, proved to be appropriate to achieve the objectives of this study and therefore, can be extended to select OSS in other areas. Thus, the result of this work is valuable not only for researchers on trends in the OSS area, but also for software developers that intend to implement DSS using FOSS.

© 2015 The Authors. Published by Elsevier B.V. This is an open access article under the CC BY-NC-ND license

(<http://creativecommons.org/licenses/by-nc-nd/4.0/>).

Peer-review under responsibility of SciKA - Association for Promotion and Dissemination of Scientific Knowledge

Keywords: Development Software, Open-source, Web Framework, Sensory Analysis

*Corresponding author. Tel.: +351-234-370361; fax: +351-234-370215.

E-mail address: lteixeira@ua.pt

1. Introduction

Sensory Analysis is a scientific area that uses statistical techniques to analyze the results of the evaluation of consumer products by the human senses (sight, smell, taste, touch and hearing) [1]. Sensory Analysis is carried out by tasting panels that evaluate sensorial characteristics of products' samples, using different types of scales. In this process, the resulting data is numerous and, as such, its interpretation requires complex statistical analysis. Due to the complexity of the process, it is very important to ensure the consistency of data entry, the correct processing of information, as well as the appropriate visualization format, using modern computer-based information systems. Decision Support Systems (DSS) are a specific class of computer-based information systems that support decision-making activities through the use of analytical models (including knowledge-based systems) and specialized databases [1].

The work presented in this paper is part of a broader project which aim is to create and test a Decision Support System (DSS) prototype to be used in Sensory Analysis. The data that result from the evaluation of products through tasting panels is used in the decisions regarding the products and can also be used to evaluate the tasters that belong to the panel.

Pertaining to the models and the data architecture, the proposed DSS comprises two modules: 'Tasting Process' and 'Taster Performance'. The first module refers to the tasting procedure and related data that derives from tasting session. The second one is responsible for grouping all procedures and statistical methods for assessing tasters' and Tasting Panels' performance. The combined information from these two modules will help the responsible make the final decision about the products based on a consistent evaluation process.

Technologically, the DSS prototype was planned as a web-application that runs in a secure web server. The user (client) performs requests to the web server and receives information via a web browser. This technological solution implies two benefits: on one hand, the users only need a web browser to use the application, because all the specific computations take place on the server side; and, on the other hand, the application can be accessed from any location, depending only of an internet connection.

Data is introduced by the users using user-friendly interfaces, stored on a central database, and visualized in a graphical format.

Regarding the DSS development life-cycle, in a first stage, a rigorous requirements engineering process was performed, using direct observation, documental analysis and informal interviews with the domain experts. The relevant data was organized and integrated in a conceptual model [2], represented using the Unified Modeling Language (UML) notation [3]. Since the system has a large component of interactivity, before the technological solution implementation, all interfaces were designed using a Lumzy[®] prototyping tool [4].

In order to create the web application it was necessary to select the ground technology and frameworks that would enable a swift and reliable development of the prototype. Given the current state of open-source technologies it was decided to utilize Free Open-Source Software (FOSS) as the basis for the working prototype.

Since the prototype is to be created from scratch, full-stack web frameworks were considered good candidates for the underlying technology. Full-stack web frameworks encompass the full development stack from storing the data to the user interface, thus providing everything needed from the technological viewpoint to complete a functioning web application prototype.

This paper presents the process of selecting the most suitable open-source framework for developing the above mentioned Decision Support System (DSS) for Sensory Analysis. Several open-source technologies were analyzed and data was collected in order to determine their viability as a reliable base for the project.

2. Selection Process of an Open-Source Framework for developing a Decision Support System

2.1 About Open-Source Software

The Open-Source Software (OSS) is becoming increasingly popular. It can be considered as a multidisciplinary research topic that has received a growing attention during the last decade, attracting the interest of academics and practitioners. OSS represents a software development model, in which the source code is available and the users

(normally programmers) can view, read, modify and redistribute this code without the restrictions of the intellectual property rights [5].

According to Corbly [6], a software product can be considered open-source if it meets several criteria, namely: (i) the software must be available without costs; (ii) the program must be distributed as source code or compiled code; (iii) programmers and/or end-users may alter the program's code; and (iv) the modified source code must be redistributed under the same conditions as the license for the original version of the software.

Today, OSS solutions can be found in many typical business applications. However, ready-made OSS solutions rarely fully meet the end-users' needs, requiring, most of them, some level of adaptation.

The collective intelligence is the basic definition underlying the OSS solutions, representing the intelligence that emerges when a group of individuals do things together [5]. OSS projects are based on virtual support communities that use the software and, consequently participate in their development. The community members learn from each other and, as a result, the overall abilities and skills of the community are improved.

In this study several open-source frameworks are analyzed and data are collected to determine their viability as a reliable base for a project. Given the role of collective intelligence and the power of crowds in the OSS context, the methodology used in this project was based on content analysis in Online Collaborative Spaces.

2.2 The process of selecting an Open-Source Framework

A framework is a cohesive design and implementation artefact and represents a set of classes, interfaces and patterns to solve a group of problems, which is a popular method to improve the development efficiency and reduce the development cost [7]. It defines an abstract design, providing common behaviour, and allowing the user to insert its specific behaviour by sub-classing or plug-in in specific implementations. There are several free open-source full-stack web frameworks to aid in the development of web applications [8], [9], [10].

In this study, a group of popular frameworks was selected to be further analyzed. All these frameworks had to meet a few criteria: (i) be completely FOSS, (ii) encompass the full development stack, (iii) work with a popular language, (iv) have a robust Object-Relational Mapping (ORM) for extensive and easy data querying and manipulation, (v) focus on productivity, and (vi) have real use cases. Additionally, given that the DSS needs to manipulate a significant volume of data to calculate the relevant statistics, the underlying ORM, Data Base Management System (DBMS) and coding language should perform adequately. These criteria lead to the election of four web frameworks: Ruby on Rails, Django, Grails and Play.

Ruby on Rails was considered for being an early popular and highly productive full-stack web framework that heavily influenced other frameworks. Django was chosen for being a widely used web framework for Python. Grails and Play frameworks were chosen for their high productivity and performance and for working with Java, notably the most used programming language with strong presence in the industry.

All the selected frameworks provide rapid prototyping and adhere to a common modern set of philosophies and architectural patterns like the 'model-view-controller' (MVC) pattern, 'convention over configuration' (CoC) and 'don't repeat yourself' (DRY). This comes in part from the influence that Ruby on Rails popularity had on the development of the remaining selected frameworks.

This study was performed based on the analysis of the above mentioned four, well-known, web frameworks, briefly characterized in Table 1.

Table 1. Analyzed frameworks and main characteristics

Framework (Tested version)	First release	Language	Company	Current License
Django (1.6.5)	July, 2005	Python	Django Software Foundation	BSD License
Grails (2.4.2)	March, 2006	Groovy and Java	Pivotal	Apache License
Play (2.2.2)	May, 2008	Scala and Java	Zengularity and Typesafe	Apache License
Ruby On Rails (4.1.4)	July, 2004	Ruby	Rails Core Team	MIT License

Some of the development philosophies and paradigms of each framework are briefly stated by their founders:

- (i) Django is a high-level Python web framework that encourages rapid development and clean, pragmatic design [11].
- (ii) Grails is inspired by Ruby on Rails and “Grails is an Open-Source, full stack, web application framework for the JVM (Java Virtual Machine). It takes advantage of the Groovy programming language and convention over configuration to provide a productive and stream-lined development experience.” [12].
- (iii) Play is inspired by Ruby on Rails and Django and “Play is based on a lightweight, stateless, web-friendly architecture. Play provides predictable and minimal resource consumption (CPU, memory, threads) for highly-scalable applications.” [13].
- (iv) “Ruby on Rails® is an open-source web framework that’s optimized for programmer happiness and sustainable productivity. It lets you write beautiful code by favouring convention over configuration.” [14].

Selecting an open-source framework as the foundation for a project should take into account some mandatory aspects like: activity, reliability, and performance. Whereas for classical commercial software the reliability, performance and support are assured in a customer-seller relationship, in free open-source software these are often only assured by the community, with no warranties [15]. As such, it becomes increasingly necessary to study and understand the selected software to ensure it can perform as necessary and respond in a reliable manner to the project requirements. Understanding what makes a free open-source project viable for production usage is thus essential.

There are some characteristics considered crucial to ensure the reliability of a free open-source software, namely:

- (i) Strong and active community: a project with an active and invested community is alive and it will probably have rapid iterations in its development and debugging;
- (ii) Good documentation and tutorials: quality learning materials and detailed documentation lowers the entry barrier and reduces the learning curve for new developers, meaning they will be productive much sooner;
- (iii) Proven real use cases: when a large company creates a product or service making use of an open-source project it usually results in the company also investing in the framework itself. Furthermore, large companies often have the most demanding scenarios regarding reliability, performance, scalability and productivity being a good measure of how reliable a framework is.

“There’s rarely a reason to choose a framework without a great community, great documentation, and a mature set of companies using it to run successful production applications” [16].

In order to select an appropriate framework for the project in analysis (a DSS for sensory analysis), the following characteristics were analysed: (i) popularity and use cases, (ii) activity and trends, and (iii) job offers. All data were collected on June of 2014.

Open-source software needs a strong community to stay active, so popularity becomes an important factor when analyzing open-source frameworks. In addition, an open-source project must be proven to be useful, so production use cases are also important as they demonstrate the reliability of the framework in a real use case.

To gauge the popularity of the considered frameworks the number of users in the project mailing list and the number of followers in Twitter and StackOverflow.com are shown in Table 2. This table also presents the most notable uses of the frameworks under analysis.

Table 2. Framework popularity and use cases, Jun 2014

Framework	Mailing list users	Twitter followers	StackOverflow followers	Notable uses
Django	26.6k	21.0k	10.7k	Pinterest, Instagram, Mozilla, The Washington Times, Disqus, Public BroadCasting Service
Grails	-	11.2k	2.2k	Atlassian, sky, LinkedIn Recruiter, eHarmony, ESPN, Netflix
Play	11.3k	14.7k	1.6k	Linkedin, Coursera, theguardian, Klout, Gilt, ZapTravel
Ruby On Rails	26k	77.9k	15.7k	GitHub, Yammer, Scribd, Shopify, Hulu, Basecamp

StackOverflow is a Q&A site for programmers that is widely used, with more than 170 million visits per month [17]. It should be noted that StackOverflow doesn't necessarily represent all the frameworks equally. Some programming languages and frameworks may rely more in other discussion tools like community forums or equivalent. But given its high number of visitors it should give a reasonable general assessment of each framework users. For example, Django and Ruby on Rails are the older frameworks and clearly show a larger user base.

All frameworks show interesting real use case scenarios with examples of large companies using each of the elected open-source projects.

Community activity indicates how intensively a project is used. Strong user activity will usually lead to stronger development. A possible way to measure user activity is, for example, to look at the number of monthly questions in StackOverflow. Search trends indicate developers' interest and relevance of the open-source projects. Interest tendencies give some insight into how developers focus and preferences are evolving. Regarding these two characteristics, Figure 1 (left side) illustrates the StackOverflow tagged monthly questions and Figure 1 (right side) shows the Google search trends (search statistics were collected from Google pertaining to the period from January of 2004 to June of 2014). As the charts in Figure 1 show, Django and Ruby on Rails have the highest activity although Grails and Play are slowly growing. Older frameworks will tend to have larger user bases even if there are new better ones. Developers have a tendency to adopt a framework they like and keep with it for some time. Projects that have been completed need maintenance or further development, meaning the framework chosen at start will continue to be utilized.

Regarding search trends, Ruby on Rails had a huge demand when it first launched, probably because it was one of the first open-source full-stack web-framework to be released. It became widely adopted when it was bundled with Apple's OS X 10.5 in August 2006. As other frameworks were released and improved (2009 onwards) Ruby on Rails searches started to diminish because developers' interest shifted.

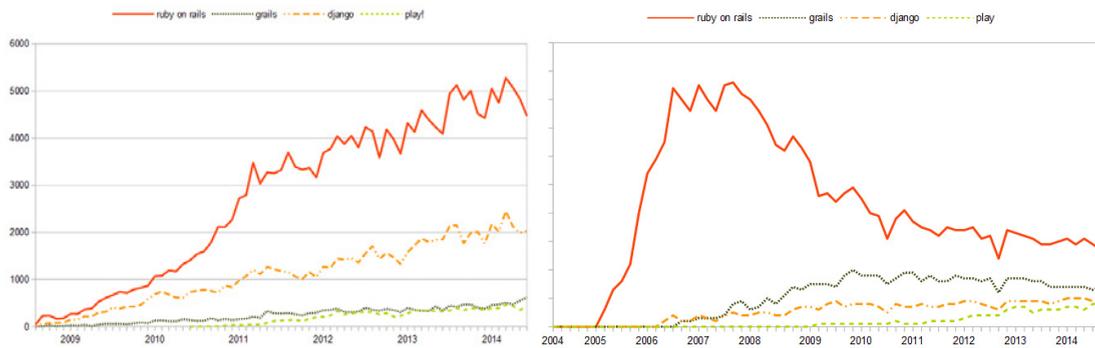


Fig. 1. StackOverflow tagged monthly questions (left) and Google search trends Jan 2004 – Jun 2014 (right)

The number of job offers that refer a given framework show the adoption level of the frameworks by the industry and indicate the volume of real use case scenarios. Job offers for the frameworks were recorded from three different sites: Dice.com, Indeed.com and LinkedIn.com and are shown in Figure 2. Play framework is not presented due to difficulties in search terms.

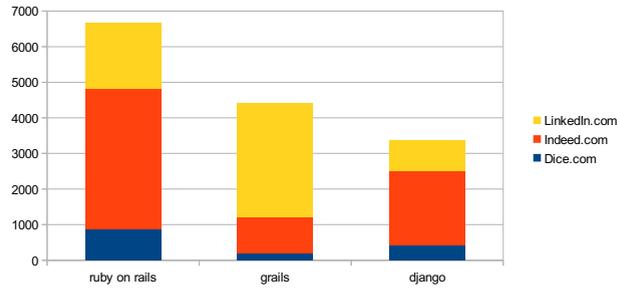


Fig. 2. Job offers in the US on December 2014

Ruby on Rails has the largest demand but all frameworks have similar strong magnitudes of job offers, indicating a relevant interest and adoption by the software industry.

On the top of this analysis and, in order to choose the appropriate framework for the system in analysis, the initial tutorials were followed and a basic project was built to better understand the workflow and productivity of each framework.

It was found that all the frameworks offer similar functionalities and organization, which probably results from sharing the same development philosophies and patterns (Grails and Play were heavily influenced by Ruby on Rails). Django is a good framework, competent in all aspects. Grails is very easy to use with very good documentation and it has a bundled IDE (Groovy/Grails Tool Suite) which really simplifies its utilization. The Groovy language used in Grails is a syntax light language, easy to learn, and highly productive, and the ability to use Java interchangeably with Groovy allows a developer with Java knowledge to be productive, while learning Groovy, which is ideal for shorter projects. Play framework is a highly performing and light-weight framework, the tutorials for Java however were not very detailed as Java, when compared to the Scala language. Ruby on Rails seems quite robust but not as user friendly as the other frameworks.

2.3 Results

Each of the studied frameworks revealed a strong and active community with frequent development iterations. A strong community is one of the most important assets for an Open-Source Project. It generates a wealth of information in the form of questions and answers in sites like StackOverflow or in forums for developers. This exchange of information and discussion of ideas can greatly help a framework to move forward and become more robust, polished and easier to use. A large community can also create a large ecosystem around a framework which in turn can generate more plugins, tools and tutorials that can enhance the utilization of the framework.

There is demand in the industry for developers with experience in all of these frameworks, which means that it is positive for any developer to invest time in learning them and gain experience by use them. Challenging real use case projects exist for each of the studied frameworks (e.g., GitHub, LinkedIn, and Instagram) which indicate good reliability and performance.

Through the data analyzed in this work the four frameworks were shown to be viable candidates with healthy communities, industry demand and proven real use cases.

In terms of popularity Ruby on Rails largely exceeds the others frameworks, due to its age. Recently Ruby on Rails search trends has been declining, as Grails and Django are increasing. A possible explanation for this, can be that developers are transitioning from Ruby on Rails to newer frameworks that offer similar high productive environments inspired by the same philosophies prevalent in Ruby on Rails. Grails has a strong demand even when compared to Ruby on Rails and taking into account the difference in their popularity, indicating the possibility of Grails demand in the industry being growing since it is being adopted by more developers. Although Ruby on Rails is still the most popular of the frameworks its advantage has been diminishing in favour of newer frameworks.

Grails growing search trends and strong demand in the industry together with its high productivity, ease of use and great documentation make it the ideal choice for this project.

3. Conclusions

In any project, the selection process of a web framework is somehow subjective depending, largely, on the complexity of the project as well as on the needs and the preferences of the developer. As such, some of the factors to consider, when selecting a web framework, include the programming experience of the developer, the available supporting libraries, the size or scale of the project, and the functionalities required by the project [18].

The present work described the process of evaluating some of the most popular open-source frameworks in order to find the most suitable one for developing a Decision Support System (DSS) for Sensory Analysis.

The popular frameworks Django, Grails, Play, and Ruby on Rails were studied regarding characteristics like activity, reliability, and performance. All frameworks reveal strong and healthy communities, good documentation and tutorials, and proven real use cases.

In this particular case Grails was the chosen framework. The strong demand and search trends that were observed indicated Grails as a solid choice to many developers. Although Ruby on Rails has still higher popularity, search trends indicate a retreating curve and the demand gap between them is closing. Additionally, the tutorials done in each framework showed Grails to be more productive and easy to use, which results, in part, from the well structured documentation and information created by its active community. These aspects contributed to an informed choice and led to the election of Grails as the appropriate web framework for the project-at-hand.

Acknowledgement

This work was supported by:

- COMPETE (Operational Programme Factors of Competitiveness) and by National Funds through FCT (Foundation for Science and Technology) in the context of the project FCOMP-01-0124-FEDER-041776.
- National Funds through FCT - Foundation for Science and Technology, in the context of the project PEst-OE/EEI/UI0127/2014.
- Portuguese funds through the CIDMA - Center for Research and Development in Mathematics and Applications, and the Portuguese Foundation for Science and Technology, within project UID/MAT/04106/2013.
- COMPETE (Operational Programme Factors of Competitiveness) and by National Funds through FCT (Foundation for Science and Technology) in the context of the project PEst-C/CJP/UI4058/2011.

References

1. Arnott D, Pervan G. A critical analysis of DSS revisited. *J Inf Technol.* 2014; 29 (4) 269–293.
2. Teixeira T, Ramos AL, Xambre AR, Alvelos H. Designing a Decision Support System for Tasting Panels. *Procedia Technol.* 2014; 16: 440–446.
3. Arlow J, Neustdt I. *UML 2 and The Unified Process: practical object-oriented analysis and design*, 2nd ed. Massachusetts: Pearson Education, 2005.
4. Xambre A R, Ramos A L, Teixeira L , Filipe N, Alvelos A. Interface Design for a Sensory Analysis Decision Support System. *8th IADIS International Conference Information Systems*, 2015, 239–442.
5. Martinez-Torres M R, Diaz-Fernandez MC. Current issues and research trends on open-source software communities. *Technol. Anal. Strateg. Manag.* 2013; 26: 55–68.
6. Corby J. E. The Free Software Alternative: Freeware. *Open-source software , and Libraries.* 2014, 65–76.
7. Tang Z, Zhang W, Wu P. A holistic framework for engineering simulation platform development gluing open-source and home-made software resources. *Adv. Eng. Softw.*, 2014; 76: 99–109.
8. Bonaccorsi A, Rossi, C. Why Open Source software can succeed. *Res. Policy*, 2003; 32(7): 1243–1258.
9. Lakhani K R, Von Hippel E. How open source software works: ‘ free ’ user-to-user assistance. *Res. Policy*, 2003 (32): 923–943.
10. Guang W, Yang Z, Wang Z. Multi-level framework of open source software adoption. *J. Bus. Res.*, 2011; 64(9) 997–1003.
11. Django Software Foundation, The Web Framework for perfectionists with deadline. 2014. [Online]. Available: <https://www.djangoproject.com/>. [Accessed: 10-Apr-2015].
12. GoPivotal I. The search is over. 2014. [Online]. Available: <https://grails.org>. [Accessed: 10-Apr-2015].
13. Zengularity T. Play Framework - Build Modern & Scalable Web Apps with Java and Scala. 2014. [Online]. Available: <https://www.playframework.com/>. [Accessed: 10-Apr-2015].

14. Rails Core Team. Ruby on Rails. 2014. [Online]. Available: <http://rubyonrails.org/>. [Accessed: 10-Apr-2015].
15. Stol K, Babar MA. A Comparison Framework for Open Source Software Evaluation Methods. *IP Adv. Inf. Commun. Technol.*, 2010; 319:389–394.
16. ClearlyTech, Choosing a Web Framework - ClearlyTech. 2014. [Online]. Available: <http://www.clearlytech.com/2013/12/01/choosing-web-framework/>. [Accessed: 10-Apr-2015].
17. Quantcast Corporation, Quantcast. 2014. [Online]. Available: <https://www.quantcast.com/stackoverflow.com>. [Accessed: 10-Apr-2015].
18. Swain NR, Latu K, Christensen SD, Jones NL, Nelson EJ, Ames DP, Williams GP. A review of open source software solutions for developing water resources web applications. *Environ. Model. Softw.*, 2015; 67: 108–117.