Ruby on Rails

Group Presentation
Bram Wiebe
Trevor Jones
Wanying Luo
Howard Chung

CMPUT 410
Osmar Zaine
November 16 2007

Table of Contents

The Ruby on Rails team describes its project as “an open-source web framework that's optimized for programmer happiness and sustainable productivity.” Both developer happiness and productivity are influenced by several factors. In the following report we will attempt to outline where Ruby on Rails (RoR) provides an advantage or disadvantage in these factors which play such an important role in today’s development environments.

What is Ruby on Rails?

RoR as it is commonly referred to as, is a web framework built upon the Ruby language. It integrates Model-View-Controller style connectivity between the data storage, business logic, and interface. It is designed to provide a common, and linear view of the web application such that there is very little of the “oh what does that code do” issues found in many other languages and their accompanying frameworks such as perl, java, and php. Much of this is due to the Ruby Language which is “[reflectively] Object Oriented” such that it has no scalars; even the number 3 is an object with methods! Ruby “rectifies the class concept not attaching it necessarily to a type but to the behaviour” allowing, when coupled with its reliance on the compiler, the developer to write elegant code which solves the primary problem at hand, instead of dealing with underlying complications of the language. (Ref: 1)

Learning Curve

Switching to a new development system can be an arduous task for some and less for others. Ruby on Rails has a disadvantage when we look at transitioning over to the framework in comparison to adopting other frameworks utilizing the current languages. The disadvantage lies in the advanced nature of the language the framework uses, in that it is not a typical C-like language much like java, php, and other more traditional languages which are commonly known and used. With a new language to learn, which is very different from other languages, coupled with learning the language of Rails, this framework poses a potential downtime while developers are retrained as opposed to choosing a framework which uses the traditional languages and thus requires very little training.

Technical reasons for RoR's slightly higher learning curve were very difficult to find. So we decided to look into the matter ourselves by writing a test application to find what were the most troublesome aspects of learning RoR from scratch. We found that programming in Ruby was quite different from how one develops in conventional languages due to the truly object oriented nature of the language. And yet while dealing with learning a different thought process, one constantly stumbles over the Rails conventions: attempting to remember if database table names are to be pluralized or not, what should be camelcased or what the scripts will generate for you and what you need to do yourself. Yet once one grasps the concepts it seems to become very easy in very short order to produce working, database driven apps very quickly and painlessly. One could relate this to the metaphorical “ripping off the bandaid”.

Support

The Ruby on Rails team has not been lapse when developing tools to assist in learning, developing, testing, debugging, and maintaining your Ruby on Rails projects. The official website includes a complete Ruby, Ruby Standard Libraries and Ruby on Rails APIs; lists several Ruby on Rails reference books from *Ruby on Rails for DUMMIES* to *Beginning Ruby on Rails E-Commerce*; several step by step tutorials and user submitted How-Tos. And that is all official Ruby on Rails approved documentation.

In addition to the official documentation, the Ruby on Rails team has provided several means for Ruby on Rails users from professionals to beginners to share information on specific issues. This includes mailing lists, a Wiki, an IRC channel and a community forum, all of which can be considered when looking for Ruby on Rails support. Neither the official nor community support options appear to include basic information about the Ruby language itself – people unfamiliar with RoR will have to contend with the Ruby support as well.

The combination of official and community support options does appear to be quite large, however the size leads to the challenge of finding the appropriate resource for your situation. For example, in a relatively quick (5 minutes) search of the official support sources I was unable to find an example, tutorial or code snippet that would help me determine the proper syntax for a For Loop, but I did manage to find out How To Avoid Cross Site Request Forgeries. One can draw form this that in regard to online sources, all the information is there to be found, and for the most part for free, but finding it may not be easy.

It is to be noted that when looking for documentation from offline sources, RoR does not fall short when compared to other frameworks. Searches on Amazon for books on RoR produces 182 results compared to apache struts (189), cakephp (7), and symfony (9). Clearly in its short rise to fame RoR has amassed a comparable collection of books to one of the most prominent web framework rivals.

Maintainability

The Ruby on Rails team has been working hard to make the code produced by the framework concise, easy-to-follow, and thus easy to maintain. Ruby on Rails generates readable code for the user when the user asks it to do so. A programmer using this framework does have the freedom to choose not to use code generation. In fact, experience programmers and serious development typically don't rely heavily on scaffolding.

Some users worry about the maintenance nightmare brought by Ruby on rails' code generation mechanism, but users should be aware that maintainability issues always exist in code-generation frameworks, this is not something unique to or seriously wrong inside Ruby on Rails itself. Ruby on rails has a large user base, major common bugs have already been eradicated, so we should have equal confidence on the generated code and our own coding skills. Before worrying about the maintainability issue, why don't we step backward and think about why would a programmer want to modify the generated code in the first place. If indeed the structure or logic of the generated code is unsatisfactory, it is more reasonable to encourage RoR team to improve generator itself in order to obtain good quality

of generated code and benefit from it in the long run, rather than modify the generated code manually over and over again. (ref: 2,3,4)

OS/Web Server/DBMS Support

Ruby on Rails is available for Linux, Windows, and MacOS X and can be installed on various web servers such as Apache, LightSpeed, Lighttpd. Support for FastCGI, SCGI, or Mongrel is recommended, as Rails tends to run slowly otherwise. While Rails can be installed on different web servers, there aren't as many hosts available with Rails support compared with other more popular technologies such as PHP, Perl, etc. (ref: 5,6)

Several database drivers exist for Ruby. Included among them are drivers for the more well known DBMSs, such as Oracle, MySQL, and PostgreSQL. With support for the more popular DBMSs, it should reduce the instances where a new DBMS would need to be installed for use with Ruby on Rails. However, if an unsupported DBMS is being used and migration to a supported DBMS is not possible, or highly cumbersome, Ruby on Rails may not be an attractive choice for developing database driven web applications. Since Ruby on Rails follows convention over configuration, if certain naming schemes are not followed in existing databases, integrating them with Rails will require some extra work. There are some database adapters in development and requests have been made for support for many other DBMSs, but whether or not adapters for those DBMSs will be developed is hard to say. (ref: 7)

Overall, since Ruby on Rails can be used with the popular operating systems, web servers, and DBMSs, it should be useable with most existing systems, although configuration settings may need to be adjusted somewhat depending on the setup.

Web services/HTTP Communication Protocols

The ActionWebService module provides support for web services. Services using the common protocols SOAP, XML-RPC, and REST are fairly simple to implement using Rails.

Rails appears to be leaning toward the REST method as indicated by a preview of the next release at http://weblog.rubyonrails.com/2007/9/30/rails-2-0-0-preview-release.

Like SOAP and XML-RPC, REST uses HTTP as the communication medium. However, while SOAP and XML-RPC are based on the concept of remote procedure calls, REST uses the concept of remote resources. REST is used to perform the most common actions on a resource and regular HTTP commands are used to indicate what action should be performed on a resource. These actions include creating (HTTP put), read (HTTP get), updating (HTTP post), and deleting (HTTP delete). With this method, simple URLs can be used to access a resource and execute basic operations.

Given the REST methodology and the purpose of Rails (i.e. creating database driven web applications) it makes sense that Rails is favoring REST, as the use of databases (remote resources) and the actions performed on them mesh quite well. (ref: 8,9,10)

Integration with other technologies

Aside from using web services to work/communicate with applications developed using different technologies, a few plugins/libraries exist which facilitate this using different mechanisms.

A plugin called WebORB allows communication between Rails applications and Flex and Flash remoting clients. Thus, instead of using the more typical HTML/Javascript frontend with the Rails backend, you can enhance the client using Flex/Flash. The WebORB plugin exposes Ruby classes and the Flex/Flash clients communicate with the WebORB plugin via AMF0 and AMF3 protocols. (ref: 11,12)

An implementation of the Ruby interpreter called JRuby also allows integration of Ruby and Java applications. It also provides some support for Rails, but access to extensions written in C is unsupported. (ref: 13)

Another Ruby/Java integration option is a library called Ruby Java Bridge (RJB). This library provides Ruby applications access to Java classes, but using the native Ruby interpreter. Thus, it should provide full access to C extensions and Rails. (ref: 14,15)

Rails also comes with the Prototype and script.aculo.us Javascript libraries for Ajax support and the scaffolds help with some of the code generation. (ref: 16,17)

The availability of options to integrate with other technologies provides added flexibility to creating applications with Ruby on Rails. Developers can take advantage of existing Java classes or spice up the client using Flex or Flash.

Comparison with Other Frameworks

Project	Language	Ajax	MVC framework	MVC Push/Pull	i18n & l10n?	ORM	Testing framework(s)
Ajile	JavaScript	Yes	Yes	Push & Pull	Yes		Yes, jsUnit
Akelos PHP Framework	PHP	Yes, Prototype, script.aculo.us	Yes, ActiveRecord	Push	Yes	Yes, ActiveRecord	Yes, Unit Tests
Apache Struts	Java	Yes		Push	Yes	Yes	Yes, Unit Tests
Apache Struts 2 (ex. WebWork)	Java	Yes		Push & Pull	Yes	Yes	Yes, Unit Tests
Aranea MVC	Java	Yes		Pull	Yes	Yes	
CakePHP	PHP	Yes, Prototype, script.aculo.us	ActiveRecord	Push	Yes, Development branch	Yes, ActiveRecord	Yes, Unit Tests
Camping	Ruby	No	Yes	Push	No	Yes, ActiveRecord	Yes, via Mosquito
Catalyst	Perl	Yes, multiple (Prototype, Dojo...)	Yes	Push in its most common usage	Yes	Yes, multiple (DBIx::Class, Rose::DB...)	Yes[1]
CodeIgniter	PHP	Yes, Framework extension	Yes, Modified ActiveRecord	Push	Yes	Yes, framework extension	Yes, Unit Tests
Django	Python	Yes	Yes	Push	Yes	Yes, Django ORM, SQLAlchemy	Yes
DotNetNuke	.NET	Yes		Pull	Yes	Yes, SubSonic, NHibernate	
Fusebox	ColdFusion, PHP	Yes	Yes, but not mandatory	Push	No, custom	Yes, via lexicons for Transfer and Reactor	Yes, CFUnit, CFCUnit
Gluon	Python	Yes	Yes	Push	Yes	Yes	No
Grails	Groovy	Yes	ActiveRecord	Push	Yes	Yes, GORM, Hibernate	Yes, Unit Test
JBoss Seam	Java	Yes		Pull	Yes	Yes, JPA, Hibernate	Yes, JUnit, TestNG
Mach-II	ColdFusion	Yes, via CF or any JavaScript Library	Yes	Push	Yes, via custom plugin	Yes Transfer, Reactor, Hibernate	Yes, CFUnit, CFCUnit
MonoRail	.NET	Yes, Prototype	ActiveRecord	Push	Yes	Yes, ActiveRecord	Yes, Unit Tests
Nitro	Ruby	Yes, jQuery	Yes	Push	Yes	Yes, Og	Yes, RSpec
PRADO	PHP5	Yes, Active Controls	Yes	Push	Yes	Yes, ActiveRecord, SQLMap	Yes, PHPUnit, SimpleTest, Selenium
Pylons	Python	Yes, helpers for Prototype and script.aculo.us	Yes	Push	Yes	Yes, SQLObject, SQLAlchemy	Yes, via nose

Features:

http://en.wikipedia.org/wiki/Comparison_of_web_application_frameworks#Comparison_of_features

Performance:

Based on benchmarks, it appears that Ruby on Rails performs a bit on the slower side compared to some of the other frameworks, although it isn't the slowest. In fact RoR performs ~30% faster in comparison to Java Springs according to tests run by Justin Gehtland author of "Springs: A Developers Notebook". (ref: 18)

Development Time:

In terms of development time, a comparison between Java using Spring and Ruby on Rails by Justin Gehtland as above produced the following statistics when implementing

the same project:

	Lines of Code	Number of Classes	Number of Methods	Configuration Lines
Rails	1164	55	126	113
Java	3293	62	549	1161

Clearly the results are staggering. By reducing the work done by developers, not only do you gain a speed increase in development, but give faster job satisfaction to the developers which translates into less frustration and more happiness.

An easily overlooked benefit of RoR which certainly increases development time is the ability to do rapid prototyping using its scaffolding features. It is very beneficial to developers if they can put stub code in place which not only gives them something to work against but actually performs the basic duties of the eventual code (think database administration tool for the backend of a website).

A comparison of some frameworks was also done by Sean Kelly, demonstrating how much less development time was needed to develop a simple web application using various frameworks compared to a Java setup. While a full blown framework was not used for the Java version, we can see how much less work is required when using a framework, and we can see how Ruby on Rails compares to a few other frameworks. Some results are summarized below:

JSP + servlets + Hibernate + MySQL	Ruby on Rails	Zope/Plone	Turbo Gears	Django
218 lines Java code 59 lines JSP code 222 lines XML config	6 lines Ruby code 23 lines view code 0 lines XML config	0 lines Python code 0 lines page templates 0 lines XML config	49 lines Python code 53 lines page templates 0 lines XML config	19 lines Python code 0 lines page templates 0 lines XML config

	Time (min)	Easy	Form Validation	Authentication/ Authorization	Templates	Docs	Internationalization (i18n)
Rails	13	0.3	Yes	No	Yes	0.9	No
Zope/Plone	7	0.9	Yes	Yes	Yes	0.9	Yes
Turbo Gears	59	0.4	Yes	No	Yes	0.4	No
Django	16	0.8	Yes	Yes	No	0.3	No

The analysis done by Sean Kelly seems to indicate that, of the frameworks examined, Rails was one of the more diffcult ones to use, which seems to further support the findings in the “Learning Curve” section of this report. As well, the set of built-in features provided are not quite the same across different frameworks. However, there appear to be many plugins available for the various frameworks to add features that may not be built-in. For Rails, there is a localization plugin to help with internationalization and various methods of authentication described on the Ruby on Rails wiki. (ref: 19,20,21,22)

Case Studies

> *"After researching the market, Ruby on Rails stood out as the best choice. We have been very happy with that decision. We will continue building on Rails and consider it a key business advantage."*
>
> *-Evan Williams, Creator of Blogger and ODEO*

> *"Ruby on Rails is astounding. Using it is like watching a kung-fu movie, where a dozen bad-ass frameworks prepare to beat up the little newcomer only to be handed their asses in a variety of imaginative ways."*
>
> *-Nathan Torkington, O'Reilly Program Chair for OSCON*

> *"Rails is the most well thought-out web development framework I've ever used. And that's in a decade of doing web applications for a living. I've built my own frameworks, helped develop the Servlet API, and have created more than a few web servers from scratch. Nobody has done it like this before."*
>
> *-James Duncan Davidson, Creator of Tomcat and Ant*

> *"Ruby on Rails is a breakthrough in lowering the barriers of entry to programming. Powerful web applications that formerly might have taken weeks or months to develop can be produced in a matter of days."*
>
> *-Tim O'Reilly, Founder of O'Reilly Media*

As a rather new web framework, Ruby on Rails has not yet seen the popularity of many of today's current standards such as the many Java or .NET based frameworks. This does not mean that Ruby on Rails is incapable of handling real world applications, as it can and has been used for a wide range of dynamic web-projects. For example, the website www.shopify.com is an e-commerce tool that allows users to easily create and manage their own online store; www.strongspace.com provides a means to store, backup, or share any type of files on a secure off-site location allowing access around the world; www.alistapart.com is an online magazine for website developers. Many other websites also use Ruby on Rails for other purposes such as "collaboration, community, e-commerce, content management, statistics, management, you name it"

Ruby on Rails is not limited to the small scale websites either; many big names on the Internet have implemented RoR into sections of their web applications. www.amazon.com has created their 'unspun' project exclusively in Ruby on Rails. Although in no way connected to Amazon's e-commerce site, the unspun site is a collection of ranked lists i.e. community opinions. For example on the list of *Favorite Thing to Put on a Hot Dog*, mustard, ketchup and relish top the list, with the least favorite being cabbage (at #22); Little Caesar's is the number one *Best Pizza Joint in Seattle*; and periwinkle is the ninth *Favorite Color for Classic Star Trek Uniforms*.

Other big names online have re-written their entire website using Ruby on Rails including www.penny-arcade.com, www.yellowpages.com and www.twitter.com . All three of these websites are very popular and active as illustrated below. All three of these sites provide a wide range of functions in

and of themselves. Penny Arcade provides a blog, webcomic, and forum; twitter provides a blog and social networking tool, and Yellow Pages.com provides a personal and business telephone directory, as well as mapping and directions.

	Traffic Rank www.alexa.com	Page Views per Day www.statbrain.com	Avg Load Time www.alexa.com
Twitter	602	3,439,579	2.5s
Yellow Pages	1766	830,204	4.8s
Penny-Arcade	2020	196,990	2.3s

Retrieved on February 12 2007 13:22

Ruby on Rails is marketed for “everyone from startups to non-profits to enterprise organizations” and that is definitely the case – it appears capable of handling any task chosen for it, and capable of handling large traffic volumes for enterprise companies, and simple enough for the smaller companies to effectively use it. Through its elegant coding, rapid development capabilities, large compatibility list with third party tools, and user friendly Ajax enabled code generation engine Ruby on Rails is certainly a development tool worth considering to decrease development costs and increase developer happiness and productivity.

Online Resources

1. http://holoflux.wordpress.com/2006/08/24/ruby-result-of-a-japanese-guy-that-just-wanted-to-be-happy-programming-and-made-a-revolution/
2. http://www.incutio.com/services/bespoke-development/ruby-on-rails/
3. http://journal.bitshaker.com/articles/2005/12/21/is-ror-maintainable
4. http://ask.slashdot.org/askslashdot/05/12/21/153252.shtml?tid=156&tid=4
5. http://wiki.rubyonrails.com/rails/pages/HowtosInstallation
6. http://www.railshosting.org/#10_things_to_look_for
7. http://wiki.rubyonrails.org/rails/pages/DatabaseDrivers
8. http://www.scribd.com/doc/202103/Web-Services-with-Rails
9. http://www.ibm.com/developerworks/java/library/j-cb08016/
10. http://www-128.ibm.com/developerworks/webservices/library/ws-restvsoap/
11. http://weblog.rubyonrails.org/2006/8/28/power-flash-and-flex-from-rails
12. http://themidnightcoders.com/weborb/rubyonrails/index.htm
13. http://www.netbeans.org/kb/60/ruby/java-ruby.html
14. http://rjb.rubyforge.org/
15. http://depth-first.com/articles/2006/08/26/scripting-java-libraries-with-ruby-java-bridge
16. http://www.webmonkey.com/webmonkey/05/45/index0a_page2.html?tw=programming
17. http://www.onlamp.com/pub/a/onlamp/2005/06/09/rails_ajax.html
18. http://www.thespringexperience.com/blog/justin_gehtland/2005/04/some_numbers_at_last.html
19. http://seankelly.tv/videos/better-web-app-development
20. http://wiki.rubyonrails.org/rails/pages/Internationalization
21. http://wiki.rubyonrails.org/rails/pages/Localization+Plugin
22. http://wiki.rubyonrails.org/rails/pages/Authentication