Metrics for Web Programming Frameworks

Daniel Walker and Ali Orooji
Dept. of Electrical Engineering and Computer Science
University of Central Florida
Orlando, FL 32816
polyesterhat@gmail.com
orooji@eecs.ucf.edu
Phone: (407)823-2341
Fax: (407)823-5419

Abstract - *Many languages and techniques exist under the umbrella of programming. Web development comprises a small subset of the whole. Without JavaScript, CSS, HTML, Databases like MySQL, and several server side languages such as PHP, the internet would be far less user friendly. Since a vast amount of code is duplicated between projects, web application frameworks were born. Some of these frameworks are basic and others are very feature-rich. This paper provides some metrics for evaluating the MVC (model-view-controller) based frameworks. More specifically, four frameworks are examined: CakePHP, Django, Ruby on Rails, and ASP.NET MVC.*

Keywords: Web Programming Frameworks; Model-View-Controller (MVC); Apache, Python, Ruby, PHP, and MySQL; CakePHP, Django, ASP.NET MVC

1 Why MVC?

The basic concept/idea in MVC is that there are three layers to a dynamic website. The database is the bottom layer which contains records modeling a certain type of data. The layer which the user sees is the view. The controller is the logical glue between the model and view, determining what data to send from the database to the view. The controller can be thought of as a two-way layer, and is divided into actions; each representing an actual page which a user can visit.

Building a modern website can be related to building a cargo ship in the olden days. The ship has many inherent expectations, like the ability to float and store food and supplies. Building the entire ship by hand would be labor intensive; a builder would struggle with even the simplest task like crafting 700,000 nails, or a set of cast iron cannons. However, he could instead hire a company which specializes in this field. This company employs experts with specific tools and materials optimized for the task. The company meets these standards through optimized methodology. Cannons must be molded from cast iron; this is a standard. If a company did not follow this standard, instead using hollowed out palm trees, havoc would ensue. A well-built ship is a vast combination of engineering from multiple sources and industries.

Similarly, a modern website developer needs to employ the right web framework. If the wrong company builds the ship, it may sink. Many web standards exist, and a framework should assist the developer in meeting these standards. A developer will not use “Old Joe’s Framework” if Old Joe did not meet standards in the framework. MVC organizes a web project very nicely; when something better comes along, MVC communities may shift their focus.

2 Installation

In general, a framework aims to assist the developer rather than supply a list of chores; a framework needs to be easy to install. A painless install means project specific code begins right away. In the early days of programing, tasks such as installation required a computer’s command line. Though many developers still prefer a command line, there are now more user-friendly methods. CakePHP, Django, and ASP.NET MVC are very easy to install. For example, to install CakePHP, a developer downloads and moves the folder structure to a virtual host. Django is installed the same way as CakePHP. For ASP.NET, simply, install a Microsoft IDE (integrated development environment). The trickiest part is installing the dependencies. The table below provides the major dependencies of each framework.

Framework	Dependencies
CakePHP	Apache PHP Database
Django	Apache Python Database
Ruby on Rails	Apache with

	Phusion Passenger, etc. Ruby RubyGems MySQL or PostgreSQL
ASP.NET MVC	Visual Studio .NET or Visual Web Developer Express Microsoft SQLServer or MySQL

Table 1: Framework Dependencies

Ruby on Rails requires Ruby, the language, and RubyGems. RubyGems is a module installation system for Ruby that is used to download and install Rails (the web framework). It is important to remember that Ruby and Python are standalone programming languages; PHP and ASP.NET were specifically designed for the web. Mac OS X comes with PHP, Ruby, and Python installed; third party software such as Mono can be used to program ASP.NET on a Unix or Linux environment. An apple computer can use a Bootcamp partition to utilize MS Visual Studio. A windows' user would need to install a server, PHP, Ruby, Python and the ASP.NET IDE manually. Each framework has a website with detailed installation instructions, and each has easy as well as tricky parts. Once the dependencies are installed, steps become simpler.

3 Learning Curve

Unless a developer feels like learning a whole new language, picking up a web framework which uses a language he already knows would be the easiest. If a developer does not know any web languages, then he should learn one before beginning to use a framework. Frameworks make complex things easier, but as soon as something abnormal is required, a good knowledge of general programming is required. There is an extra learning curve when it comes to a web framework.

Frameworks contain a series of shortcuts and methodologies built on top of the language itself. For example, in an MVC, the URL of a webpage is simply the name of a function in the controller object; whereas, without a framework, a file with that name, containing its own HTML, server-side logic, and database queries would be required; there would be a similar file for every page on the entire site. A framework speeds things up, and has many functions developers must become familiar with. Within a few weeks, one would be comfortable with a framework.

The frameworks covered in this paper are the best candidates for each language in terms of their popularity and their meeting of the standard MVC features. However, a good website built in Zend PHP could still be better than a website built in CakePHP. There are many frameworks because, as soon as a developer is unhappy with an existing one, he makes his own. For an expert at CakePHP, it would be easy to learn ASP.NET MVC. Django and ruby are closely related in terms of syntax; as are PHP and ASP.NET. ASP.NET is the strictest when it comes to OOP because it is most like Java. PHP is generally considered a scripting language, not a full blown OO language. So, a developer who does not know OOP may benefit from PHP.

4 Core Library

If someone is looking for a book on a certain topic, they will probably go to the largest library nearby. A larger library is more capable of getting the needed book. Similarly, every web framework has a library containing convenience methods for the developer. Generally, a larger core library is more capable of helping a task. Many of these methods are originally written for use by the developers who made the framework; but included for public use within the framework. For example, CakePHP has a utility library called Set, which adds union and intersection ability for arrays. CakePHP also has a core helper library called HtmlHelper, which assists in generating html code for a view.

A core library on an MVC is built on top of the library functions of the language itself, and all successful web languages have vast libraries. For example, ASP.NET has many built-in functions similar to Java allowing string, array and file manipulations. ASP.NET and Django both have an interesting example of a core library; they auto generate an administrative section for a website project. This section is a web page which allows the managing of users and groups as well as access control for the actions within the project. The Django automatic admin page also allows editing of all content in the database of the project. In other words, it looks at the database schema and generates "scaffolded" controllers and views to edit that data; since it generates the code, a developer can then customize it. This adds a huge jump start to any project in Django. CakePHP does something similar with a "bake script," but is far more basic than Django. This bake script reads the database configuration and builds out "scaffolded" code for the entire website; which can then be edited.

Since there are so many functions already in web languages, a framework only needs to add a higher level functionality such as html helpers, database interaction libraries, and authentication libraries.

5 Object Relational Mapping

Frameworks use a combination of object relational mapping (ORM) and a database abstraction layer (DBAL) to completely eliminate the need for custom SQL. The framework queries a database table for the schema, then dynamically builds objects within the programming language for the developer which model the schema of the database object. The ORM is the set of functions which represent a record in the database. The DBAL is the set of properties given to the ORM object allowing it to perform CRUD operations (create, read, update, delete) on itself. For example, in CakePHP one could perform the following:

```
// this is DBAL
$s1 = $this->Student->findByName('Bob');
// this is ORM
$s1['Student']['name'] = 'Bobby';
$this->Student->save($s1);
// name of Bob is changed and saved
```

Figure 1: ORM / DBAL Example

The process is similar in each of the four frameworks. Without a framework, building a query is a very manual process; not to mention processing through the returned data. As Django's documentation says "A model class represents a database table, and an instance of that class represents a particular record in the database table. To create an object, instantiate it using keyword arguments to the model class, then call save() to save it to the database." A developer can also use plain SQL at times in any of these frameworks if they prefer.

6 Unit Testing

Test driven development means using small asserted statements to test a controller or model for expected output. In fact, using unit testing, a developer could make all the controller actions of a website before making any views. Many developers test simply by refreshing a webpage, but these frameworks allow a more direct approach. ASP.NET programmers may have it the easiest, as unit testing is simply a menu item within Visual Studio .NET. Rails has an easy guide here: http://guides.rubyonrails.org/testing.html. Rails, Django and CakePHP use "Fixtures" which are sample data sets used for the tests which are easy to make. Each of the four frameworks allow third party unit testing libraries, but Django and Ruby may be easier for this than CakePHP and ASP.NET. CakePHP considers a third party class a vendor; and including them into a project can be a hassle. However, CakePHP's built-in unit testing is beyond satisfactory.

7 JavaScript Included

Currently, the most popular JavaScript library online is jQuery. Whether or not a framework comes with some prepackaged JavaScript modules like Ajax loading or effects has no bearing on whether jQuery or another library can be used. Using a JavaScript library is easy in any framework. Each framework has a folder for web resources, CakePHP calls it "webroot", Django calls it "scripts", etc. If a developer does not want to learn JavaScript, he should still stay away from the preconfigured JavaScript modules within web frameworks; custom functionality requires much custom code. In other words, some frameworks come with pre-built support for prototype.js, or Mootools or jQuery, but the point is: no amount of pre-built JavaScript modules will make for a good website. If it needs to be functional on the client-side, much custom JavaScript will be required.

8 Documentation

For good learning, there must be a good teacher. Each of these frameworks have thorough documentation, detailing each aspect of developing with the framework. The documents backing CakePHP, Rails, and Django have similar documentation; all are very high quality. ASP.NET MVC documentation is very different. ASP.NET MVC was built when Microsoft decided to make its already popular web framework ASP.NET into an MVC model; only adding enough code to force it into the MVC pattern. The bulk of the core library documentation is within the documentation for ASP.NET, not ASP.NET MVC.

Since each of these frameworks sits on a popular language, their documentation also sits on the documentation of the language. In other words, something a developer needs in Django, may already be part of Python. Familiarity with the language is a must. Each of these frameworks has two types of docs, the API or api and the tutorials. The api lists every method of each library within the framework while the tutorials show examples of common tasks. Programming languages generally have the api only; but many supporting community websites. CakePHP has a third type of documentation called the Bakery which lists third party tutorials. Also, so many blogs exist online that every aspect of every common task of all four frameworks is thoroughly hammered out. Going purely by aesthetics, the ASP.NET MVC documentation website gets less praises compared to the Rails, CakePHP and Django website.

9 Community

The community is the backbone of a popular framework; the community encourages growth for the framework while maintaining standards. There are major differences between the communities of large frameworks; sports teams have similar fan-bases. The more someone uses a framework, or the less they have used other frameworks, the more their preferred framework is elevated in their mind. With every framework, some people come and go, while others would die before leaving. However, these people are the reason beginners' questions do not go unanswered. The community of supporters for a framework is made of the people who interact and communicate about it on blog websites. It seems that a larger percent of the community of ASP.NET are in foreign countries.

All communities have knowledgeable members. However, it does seem that more people who use Rails and CakePHP like to talk about what they made, more so than people who use Django. People who use ASP.NET seem to only use it for work and therefore only talk about the problems they are having with it. One thing the community loves to do is show off their code. The ASP website is mostly in-house examples. Here are some galleries of framework code:

Framework	Website
CakePHP	http://bakery.cakephp.org/
Django	http://djangopackages.com/
Rails	http://rubygems.org/
ASP.NET MVC	http://www.asp.net/mvc

Table 2: Framework Websites

10 Updates to the Framework

Each of the four frameworks studied in this paper are updated regularly. In fact, regular updates was a required metric for inclusion within our study. It does not matter which framework has the highest version, or even the number of previous versions. The most important aspect is whether or not there has been a steady improvement upon the framework. Also, the language which the framework is built on needs to be regularly maintained and updated. The real reason a developer would want an update is for future projects. Sometimes, updating an old project with a new version of a framework can waste time. CakePHP just released 1.3 which deprecated several old functions and methodologies which would break older projects. Django has a 1.3 release candidate (http://docs.djangoproject.com/en/dev/releases/1.3/). Release candidates are great because developers commit fixes and features before the alpha release.

11 Reusability of Parts (Plugins, Helpers, etc.)

The idea behind web frameworks is that a developer won't have to retype all kinds of common code between projects. What about the custom code that he wants to share between projects or with other developers? The table below provides, for each framework, the different reusable code formats.

Framework	Module Names
CakePHP	Component, Behavior, Helper, Element, Plugin (all encompassing)
Django	Egg, App, Extension, Middleware
Rails	Gems (Sometimes called Plugins)
ASP.NET	Plugins

Table 3: Framework Modularity

CakePHP has many options for breaking up the reusable functionality. Each of the names in the table is essentially a directory within an app. So, sharing with other developers becomes easy. Each framework has different docs outlining how to use and make reusable parts within the framework. Python itself has eggs, similar to how Ruby has gems. Eggs and Gems are not part of the framework, but part of the language. Django has extensions and middleware. There are slight distinctions between all of these items, yet they only become relevant when a programmer finds he is knee deep in a project.

12 Conclusions

There are nearly 100 popular frameworks for about 10 popular web programming languages. The metrics offered in this paper should be a mere starting point for choosing a framework. However, the only true metric will be whether a developer is satisfied with the framework after spending time with it. The MVC methodology should be considered in high regard; frameworks which do not follow it get left behind by communities. When all is said and done, knowing a language is the most important part to using a framework.

13 References

[1] CakePHP http://cakephp.org/
[2] Django http://www.djangoproject.com/
[3] Ruby on Rails http://rubyonrails.org/
[4] ASP.NET MVC http://www.asp.net/mvc/
[5] ASP.NET MVC Overview Tutorial http://www.asp.net/mvc/tutorials/asp-net-mvc-overview-cs
[6] ASP.NET MVC Music Store Tutorial http://www.asp.net/mvc/tutorials/mvc-music-store-part-1
[7] Django Applications http://en.wikipedia.org/wiki/Django_(web_framework)#Applications_built_on_Django