Universidade Federal de Santa Catarina
Centro Tecnológico
Curso de Sistemas de Informação

Rafael Strecker Coelho de Souza

Avaliação de Ferramentas CMS Ruby

Florianópolis
2010

Rafael Strecker Coelho de Souza

Avaliação de Ferramentas CMS Ruby

Monografia apresentada ao Curso de Sistemas de Informação da UFSC, como requisito para a obtenção parcial do grau de BACHAREL em Sistemas de Informação.

Orientador: Lúcia Helena Martins Pacheco

Doutora em Engenharia

Florianópolis

2010

Coelho de Souza, Rafael

Avaliação de Ferramentas CMS Ruby / Rafael Coelho de Souza - 2010

xx.p

1.Availiação 2. CMS.. I.Título.

CDU 536.21

Rafael Strecker Coelho de Souza

Avaliação de Ferramentas CMS Ruby

Monografia apresentada ao Curso de Sistemas de Informação da UFSC, como requisito para a obtenção parcial do grau de BACHAREL em Sistemas de Informação.

Aprovado em 14 de outubro de 2010

BANCA EXAMINADORA

Lúcia Helena Martins Pacheco

Doutora em Engenharia

Eduardo Bellani

Bacharel em Sistemas de Informação

Aos meus pais, amigos e minha namorada que me aguentaram e pressionaram este tempo todo para a conclusão do trabalho

Agradecimentos

Agradeço ao meu amigo, colega de curso, parceiro de trabalhos e orientador Eduardo Bellani, pelo encorajamento, apoio e seus ricos conselhos sobre o melhor direcionamento deste trabalho.

A professora Lúcia Helena Martins Pacheco pela orientação, amizade, e pela paciência, sem a qual este trabalho não se realizaria.

Todos os meus amigos que incentivaram e compreenderam a minha ausência nesse período de corrida atrás do objetivo de concluir o curso.

Sumário

Lista de Figuras **4**

Lista de Tabelas **5**

1 Introducao **6**

2 Justificativa **7**

3 cms **8**

3.0.1 O que é Conteúdo 8

3.0.2 Tipos de CMS 9

3.0.3 Ferramentas Avaliadas 12

4 Ruby **13**

5 Ruby On Rails **17**

Lista de Figuras

5.1 Arquitetura MVC 18

Lista de Tabelas

1 Introducao

Nos últimos anos, tanto indivíduos quanto organizações multiplicaram a quantidade de conteúdo produzido e compartilhado. Ferramentas CMSforam criadas com o intuito de organizar e gerenciar melhor estas informações. Este trabalho definirá claramente o escopo do tema abordado, assim como os objetivos e as limitações da área pesquisada devidamente justificadas.

Neste trabalho será feita uma análise comparativa sobre ferramentas CMS, e por fim uam classificação das mesmas de acordo com critérios pré estabelecidos dentro do projeto. Em virtude da vasta abrangência de ferramentas CMS, este trabalho limitara-se-a avaliar as ferramentas escritas na linguagem de programação Ruby. Ruby é uma linguagem relativamente pouco conhecida, mas que nos últimos anos tem seu uso aumentado, principalmente na web.

2 Justificativa

A quantidade de conteúdo digital não para de crescer, nos últimos tempos tem crescido de forma assustadora. *Segundo o IDC o crescimento de conteudos digitais cresceu 62% em 2009, em um número aproximado de 0,8 zetabyte.* Diante de tanto conteúdo gerado surge a necessidade de gerenciar e organizar essas informações, tanto por indivíduos como também organizações.

3 cms

CMS é a sigla para Content Management System, ou sistema gerenciador de conteúdo. É a idéia de um gerenciamento da informação de organizações que produzem muito conteúdo. São baseados na web, auxiliando em vários aspectos a publicação de conteúdo, desde sua forma de apresentação ao seu controle de versão. Estes CMS servem para a publicação de conteúdo,gerenciamento de transação de e-commerces, Wikis, gerenciamento de documentos, entre outras atividades. Sistema de Gerenciamento de Conteúdo podem de maneira simples e óbvia serem definidos por sua sigla, um sistema que gerencia conteúdos. Para uma melhor definição de CMS - Content Management Systems, o assunto será abordado a teoria de conteúdo e gestão dos mesmos.

Podemos dizer que um CMS é um framework, "um esqueleto"de website pré-programado, com recursos básicos e de manutenção e administração já prontamente disponíveis. É um sistema que permite a criação, armazenamento e administração de conteúdo de forma dinâmica, através de uma interface de usuário via Internet. Um CMS permite que a empresa tenha total autonomia sobre o conteúdo e evolução da sua presença na internet e dispense a assistência de terceiros ou empresas especializadas para manutenções de rotina. Nem mesmo é preciso um funcionário dedicado (webmaster), pois cada membro da equipe poderá gerenciar o seu próprio conteúdo, diminuindo os custos com recursos humanos. A habilidade necessária para trabalhar com um sistema de gerenciamento de conteúdo não vai muito além dos conhecimentos necessários para um editor de texto. **?**

Linguisticamente, (CMS) significa qualquer sistema que auxilia no gerenciamento de conteudo - criação, armazenamento, indexação, arquivamento, publicação e distribuição do conteúdo. **?**[1]

3.0.1 O que é Conteúdo

Há várias definições para Conteúdo, no contexto de CMS, uma visão mais simples de conteúdo vem de **?** "O termo conteúdo representa qualquer conteúdo eletrônico, incluindo registros,dados e metadados, como também documentos e websites". A definição de **?**

[1] Tradução livre do autor

"Conteúdo,protanto, é a informação que você rotula com dados para então um computador porder organizar e sistematizar a coleção,gerenciamento e publicação". Estas definições apresentam algumas diferenças, mas nota-se que sem o auxílio de uma ferramenta específica para gerenciar conteúdos, seria uma tarefa deveras árdua.

3.0.2 Tipos de CMS

Como CMS é um conceito amplo, existem várias classificações entre os Gerenciadores de Conteúdo. Alguns podem ser mais específicos como para o funcionamento de blogs ou wikis, outros mais generalistas como os Web Content Management Systems (WCMS).

Segundo **?** eis as seguintes classificações

Portais ou CMS genéricos ou WCMS

Estes CMS são muito populares, geralmente encontrados na confecção de sites corporativos, de pequeno até grande porte no acso de Portais. Eliminam em grande parte a complexidade de o site ser administrado por uam pessoa técnica, conhecida como webmaster. Com este tipo de CMS pessoas com pouco conhecimento técnico podem publicar conteúdo.

Principais Características

- Criar a gerenciar seções de conteúdos.
- Criar páginas e adicionar conteúdos de textos ou imagens.
- Editar conteúdo publicado.
- Administração por múltiplos usuários.
- Versionamento de conteúdo.
- Gerenciamento de Workflow.

Exemplos de WCMS

- BrowserCMS

- RadiantCMS
- RefineryCMS
- ZenaCMS
- Drupal
- Joomla
- Liferay

Blog CMS

Blogs também são considerados CMS pois são autorais, publicam conteúdo de texto,imagem,vídeos.

Principais Características

- Criar posts.
- Categorizar Posts.
- Gerenciar comentários.
- Adicionar imagens,videos, textos.

Exemplos de Blog CMS

- WordPress
- Mephisto
- Typo
- Blogger

Wiki CMS

Wiki é uma página ou coleção de páginas web projetadas para permitir o acesso, a qualquer usuario, para contribuir ou modificar o conteúdo (excluíndo os usuários bloqueados),

utilizando uma linguagem de marcação simplificada. Wikis são geralmente usados para criarem sites colaborativos e amplificar sitems de comunidades. ?página 22[2]

Principais Características

- Facilidade de criar páginas, chamdas de wikiweb.
- Linguagem de marcação simples.
- Criação de links automatizadas, ainda que o link ainda não exista.
- Sistema completo de versionamento.
- Pode ter acesso restrito à usuarios ou grupos.

Exemplos de Wiki CMS

- MediaWiki
- TWiki
- Typo
- Blogger

eLearning CMSs

eLearning CMS são gerenciadores de conteúdos especializados em ensino a distância. Também conhecidos como LMS Learning Management Systems, eles tem responsabilidade pela administração,documentação,registro e relatório de cursos.

Principais Características

- Gerenciar cursos, estudantes, professores.
- Criar um curso e um programa de aprendizado.
- Criar documentos, testes, discussões e anúncios.
- Sistema de chat, forum, blogs, etc.

[2]Tradução livre do autor

Exemplos de eLearning CMS

- Dokeos
- Moodle
- LAMS

?[3]

3.0.3 Ferramentas Avaliadas

As ferramentas avaliadas pertencem ao grupo dos Web Content Management Systems, e dentro deste contexto elas serão avaliadas.

RadiantCMS

Radiant é um sistema de gerenciamento de conteúdo aberto, simples e projetado para pequenas equipes **?**[4]

Radiant é um dos CMS mais antigos disponíveis para Ruby, lançado em 2006. De instalação e uso simples tem uam grande quantidade de plugins disponíveis e uma grande comunidade mantenedora da aplicação.

Características

- Interface elegante intuitiva
- Bom sistema de extensões e plugins.
- Sistema de usuários e permissões simples.

Radiant

[3]Tradução livre do autor
[4]Tradução livre do autor

4 Ruby

Ruby é uma linguagem de programação de código aberto e orientada a objetos criado por Yukihiro 'Matz' Matsumoto. A primeira versão foi lançada no Japão em 1995. Ruby tem ganhado aceitação ao redor do mundo como uma linguagem fácil de aprender, poderosa, e expressiva, especialmente desde o advento do Ruby on Rails, um framework para aplicações voltadas para Web escrito em Ruby[1]*. O núcleo de Ruby é escrito na linguagem de programação C e roda na maioria das plataformas. É uma linguagem interpretada e não compilada.* **?**página 1[2]

Uma descrição pouco mais detalhada sobre Ruby: *Ruby é uma linguagem de programação dinâmica com uma complexa, porém expressiva gramática e um núcleo de biblioteca de classes com uma rica e poderosa API. Ruby tem inspirações em Lisp,SmallTalk e Perl, mas usa uma gramática que facilita o entendimento de programadores Java e C. Ruby é uma linguagem orientada à objetos pura, mas também suporta estilos de programação funcional e procedural.Ela inclui uma poderosa capacidade de metaprogramação e pode ser usada para criar linguagens de domínio específico ou DSLs.* **?**página 1[3]

Ruby é escrita em C e atualmente está disponível para as mais diversas plataformas UNIX, Mac OS X, Windows 95/98/Me/NT/2000/XP/Vista/Seven, DOS, BeOS, OS/2, .NET, Solaris (**?**).

O criador de Ruby, Yukihiro 'Matz' Matsumoto, teve a idéia de criar uma linguagem de programação em 1993, durante uma conversa um um colega sobre linguagens de script. Através da conversa, o seu interesse cresceu muito sobre as linguagens de script, onde ele ficou impressionado pelo poder e pelas possibilidades. Como fora um fã de longa data de programação orientada à objetos, lhe pareceu , a orientação à objetos, muito adequada para linguagens de script. Então passou olhar pela rede e encontrou Perl 5, que ainda não havia sido lançada, contendo algumas características de orientação à objetos, mas ainda não era o que ele queria. Então decidiu abandonar Perl como uma linguagem de script orientada à objetos. Então ele procurou Python, era uma linguagem orientada

[1] http://www.rubyonrails.org

[2] Tradução livre do autor

[3] Tradução livre do autor

à objetos interpretada. Mas ele não teve a sensação que era uma linguagem de script, era uma linguagem híbrida, procedural e orientada à objetos. Matz queria uma linguagem de script que seria mais poderosa que Perl e mais orientada à objetos que Python. Então ele decidiu criar a própria linguagem. O nome Ruby veio de uma brincadeira com um amigo durante o projeto de desenvolvimento da linguagem, ele queria um nome de uma pedra preciosa, a là Perl, e então o amigo sugeriu Ruby, que depois se tornou o nome oficial da linguagem. (**?**).

A filosofia de Ruby segundo o Yukihiro 'Matz' Matsumoto: *Ruby foi projetado para fazer programadores mais felizes* **?**página 1[4]

Abaixo um exemplo de uma classe Ruby

No exemplo que segue podemos visualizar como Ruby define uma classe e cria um objeto. A definição de classe se dá pelo comando *class*. No exemplo o nome da classe é *Pessoa*. Na linha 2 há uma declaração de um atributo da classe $attr_acessorestadeclaraofazcomqueoR$

Nas linhas 15 e 16 criaremos 2 objetos da classe pessoa passando argumentos diferentes para cada um deles. Nas linhas subsequentes iremos escrever na tela do console o valores retornados pelo método agradecer que vai retornar o agradecimento correto para homem *e* mulher.

exemplo_ruby.rb

```
1  class Pessoa
2    attr_accessor :nome
3
4    def initialize(nome = "Rafael Strecker Coelho de Souza")
5      @nome = nome
6    end
7  end
8
9  pessoa = Pessoa.new
10 p pessoa.nome
11 pessoa.nome = 'Rafael'
12 p pessoa.nome
```

Como resultado teremos na tela os valores do método invocado pelos objetos

[4]Tradução livre do autor

homem e mulher:

$ ruby exemplo-criacao-objetos.rb

"Homem: Obrigado"

"Mulher: Obrigada"

Em Ruby, como é uma linguagem orientada à objeto pura, tudo é um objeto. Desde números aos valores booleanos true e falseao nil (versão de Ruby para null, indicação de falta de um valor).(

O exemplo abaixo demonstra que valores numéricos, valores booleanos, e o nil podem invocar o método class *que retorna a sua classe e imprimiremos o resultado na tela do console.*

exemplo_objetos.rb

```
1 p 1.class
2 p true.class
3 p nil.class
```

O resultado gerado na tela pelo exemplo é:

$ ruby exemplo_objetos.rb

Fixnum

TrueClass

NilClass

Cada linha é a resposta do nome da classe ao qual cada valor pertence. Por exemplo o número 1 é da classe Fixnum*, o número 1.1 pertence a classe* Float *e assim por diante.*

O interpretador mais usado e conhecido de Ruby é o MRI, ou Matz Ruby Interpreter, escrito em C e a sua versão atual é 1.9.2. Além dele há outros interpretadores como JRuby,Rubinius,Maglev,MacRuby,IronRuby.

JRuby é um interpretador Ruby rodando sobre a maquina virtual Java (JVM). Ele está na versão 1.5.2 e esta versão é compatível com o MRI 1.8.7.

Rubinius que foi originalmente idealizado para ser o interpretador de Ruby feito em Ruby, mas em virtude da compatibilidade com o MRI parte do seu código é escrito

em C++. Está na versão 1.0.1 e é compatível ao MRI 1.8.7 Um grande aspecto de linguages populares como Java e C, é que a maioria das funcionalidades disponíveis para os programadores é escrito na própria linguagem. O objetivo de Ribinius é adicionar Ruby à estas linguagens. Desta forma Rubystas podem facilmente adicionar funcionalidades à linguagem, arrumar defeitos, e aprender como a linguagem funciona. Aonde é possível Rubinius é escrito em Ruby. Onde (ainda) não é escrito em C++. **?**[5]

*Maglev é o um interpretador Ruby desenvolvido pela empresa Gemstone. Conhecida pelo seu interpretador de Smalltalk, a empresa se lançou como uma plataforma alternativa, já que inclui um mecanismo de persistencia de objetos distribuidos. Está em versão Alpha ainda.*Maglev é uma implementação de Ruby rápida,estável com uma integrada persistência de objetos e cache compartilhado e distribuído. **?**[6]

Iron Ruby que é um interpretador Ruby feito sobre a plataforma .NET. Ele era patrocinado pela Micrsoft até Julho de 2010, onde a empresa cortou o último desenvolvedor remanescente do projeto. Está na versão 1.0 e com um futuro incerto.

[5]Tradução livre do autor

[6]Tradução livre do autor

5 Ruby On Rails

Ruby on Rails é um framework projetado para escrever apliacações web de forma rápida e simples. Extraído do Basecamp, ferramenta de gerenciamento de projetos da 37 Signals, o criador David Heinemeier Hanson escreveou Rails com soluções já praticadas para problemas comuns do mundo real. Estas soluções e decisões são o que torna o Rails prazeroso de usar - as chatas, tarefas servis, ja estão feitas, deixando você apenas para concentrar no seu problema. **?***página 1*[1]

Ruby on Rails é um framework escrito em Ruby, lançado em 2004, se tornou popular em 2005 à partir de uma palestra feita no Fórum Internacional de Software Livre pelo seu criador David Heinemeier Hanson, onde mostrou como criar uma weblog usando Rails em 15 minutos.

Rails foi extraído de uma aplicação real, o Basecamp, uma ferramenta de gerenciamento de projetos da empresa 37 Signals. Seu criador teve base em outros frameworks web, utilizando idéias de cada um e combinado à expressividade da linguagem Ruby, uma boa divulgação, atingiu uma grande popularidade. O framework introduziu a linguagem Ruby para o ocidente, e trouxe bastante atenção de programadores interessados nas facilidades providas pelo framework.

Uma de suas filosofias é a convenção ao invés de configuração (CoC-Convention over Configuration). David era contra os frameworks que abusavam de configurações em arquivos XML, então ele partiu para abolir ao máximo possível arquivos de configuração. Mesmo os arquivos de configuração de Rails não utilizam XML e sim um outro formato de serialização chamdo YAML (Yet Another Markup Language). Outra filosofia adotada é o DRY (Don't Repeat Yourself), onde dita que a informação deve ser localizada em um único local.

Uma característica da arquitetura é o padrão MVC, Model View Controller, que ele utiliza. Este padrão é seguido pelos frameworks web mais populares. A figura abaixo 5.1 demonstra o funcionamento do MVC.

Podemos observar na figura que:

[1] Tradução livre do autor

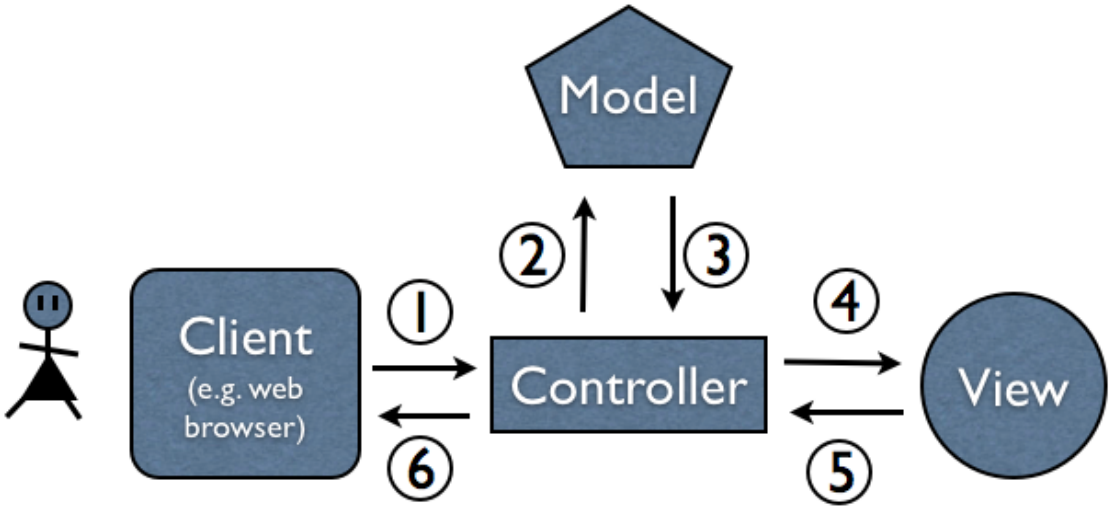

Figura 5.1: Arquitetura MVC

1. *O cliente à partir do seu browser faz uma requisição para uma página.*
2. *Esta requisição é encaminhada ao controller, que tem a responsabilidade de determinar o destino da requisição. Ele deve pedir ao model alguns dados necessários para completar a requisição.*
3. *O model que é responsável pela comunicação com a base de dados, então faz uma chamada SQL para buscar os dados requisitados pelo controller e os encaminha.*
4. *O controller então encaminha para a view estes dados oriundos do model.*
5. *A view detém a responsabilidade de combinar estes dados com um modelo de html e css, gera a página html de resposta para o browser do cliente.*

A estrutura do framework vem separada em diversos módulos, são eles:

- *ActiveRecord - Módulo de Mapeamento Objeto Relacional, usado para facilitar o acesso e a manipulação da base de dados.*
- *ActionController - Módulo responsável pelo controle, usado para gerenciar todo o ciclo de vida de uma requisição.*
- *ActionView - Módulo responsável pela renderização dos dados, é a parte visível para o browser.*
- *ActionMailer - Módulo de envio de emails, age similar ao ActionView, lidando com emails.*

- *ActiveResource - Módulo de web services, auxilia a gerar e consumir web services RESTful.*

Além destes módulos Rails conta um uma arquitetura de plugins que fomenta a criação e extensão do framework. A partir desta arquitetura é possível extender, modificar e criar novas funcionalidades para o framework. Esta arquitetura fácil levou a criação de milhares de plugins para Rails.

Uma característica que faz do Rails uma opção popular são os seus geradores. São pedaços de códigos gerados automaticamente a partir de comandos enviados pelo console. Os geradores tornaram bem simples a geração de cadastros CRUD (Create Retrieve Update Delete), uma das atividades mais repetitivas do desenvolvimento web.