Sistemas de Controle de Processos em RUBY ON RAILS

Marta Angela de Almeida Sousa Cruz
Rômulo Mendes Figueiredo
Rosângela Mourat da Rocha Ávila

RESUMO: *Mudanças no ambiente empresarial tornam essencial para as organizações o estímulo ao resgate do conhecimento organizacional para auxílio da decisão e busca de vantagens competitivas mediante ferramentas que possam auxiliá-las na gestão da informação. Estudos descritivos sobre o Rails mostram um framework de código aberto para desenvolvimento de sistema com vistas a atender necessidades específicas, de modo rápido. Através de análise documental e reuniões com membros do Departamento em que as rotinas foram desmembradas por questionamentos e acompanhamento direto aos funcionários que participam do processo, desenvolveu-se um sistema em Rails para ser utilizado como ferramenta de otimização da gestão dos processos que passam por um Departamento de Procuradoria Jurídica, de forma rápida e eficiente, sem custos de aquisição de software, incluindo sistema operacional. Observou-se que, mesmo com a existência de sistemas prontos, a especificidade da rotina do Departamento exigiu um novo sistema para atender as suas necessidades.*
Palavras-chave: *Tecnologia da Informação; Otimização; Ruby On Rails.*

ABSTRACT: *Changes in business environment make it essential for the organizations to stimulate the recovery of organizational knowledge to aid the decision and seek competitive advantages through tools that can assist them in managing information. Descriptive study on the Rails shows an open source framework for development system to meet specific needs on quick service. Through document analysis and meetings with members of the department where the routines have been disrupted by questions and direct accompaniments to employees who participate in the process, a system in Rails was developed to be used as a tool to optimize the management of processes across a Legal Department, quickly and efficiently, without acquisition costs of software, including operating system. It was observed that even with the existence of ready systems, the specificity of the routine of the department required a new system to meet its needs.*
Keywords: *Information Technology, Optimize; Ruby on Rails.*

INTRODUÇÃO

Vivemos num contexto mundial em que estamos permanentemente submetidos às implicações decorrentes de grandes transformações nos cenários político, econômico e tecnológico, trazendo como consequência direta a necessidade de estratégias mais adequadas a esse ambiente de constantes mudanças. Nesse sentido, a informação é utilizada como recurso essencial nas tomadas de decisão.

Atualmente, toda empresa tem acesso a variados tipos de informação e, para se manter no mercado de forma competitiva, garantindo o seu sucesso, precisa agregar valor a partir do acesso, do tratamento, da utilização e da disseminação da informação (Lehmkuhl *et al*., 2008).

Existem várias definições, abordadas por diferentes autores, que procuram diferenciar os conceitos de dado, informação e conhecimento. Porém, uma definição geral para essa diferenciação seria tomar o dado como registros, ou de características de coisas, ou de fatos que aconteceram sobre determinado evento; a informação, como dados contextualizados com conhecimento, ou seja, dados agregados de valor; e conhecimento, como a informação entendida e aplicada. Assim, a informação possibilita a redução da incerteza na tomada de decisão, tornando-se fator importante para determinação de escolhas com menores riscos (Freitas e Amaral, 2008).

A criação do conhecimento precisa do processamento de informações e do comprometimento pessoal para o desenvolvimento de um padrão de comportamento, envolvendo tanto idéias quanto iniciativas, dando origem à inovação.

A gestão do conhecimento necessita da disponibilidade da informação, agregada a experiência, contexto, negociação, interpretação e reflexão das pessoas para o qual essa informação faça sentido e tenha valor (Terra, 2001).

A criação de conhecimentos novos através de espaços físicos e/ou virtuais auxiliam a gestão do conhecimento e a aprendizagem organizacional, demonstrando que os conhecimentos não serão recursos estáticos acumulados em arquivos ou na cabeça dos indivíduos (Freitas e Amaral, 2008).

As empresas modernas têm como principal patrimônio o gerenciamento do seu conhecimento e, inseridas na sociedade da informação e do conhecimento, devem, com o auxílio de um novo modelo de tecnologia da informação, fazer a distribuição da informação e gestão, tornando-se competitivas no mercado (Cândido e Filho, 2003).

Para que a gestão de informação seja eficaz, *é necessário que se estabeleça um conjunto de políticas coerentes que possibilitem o fornecimento de informação relevante, com qualidade suficiente, precisa, transmitida para o local certo e no tempo correto* (Reis, 2004).

As organizações passam a ter a informação como um ativo, que precisa ser gerenciado (McGee e Prusak, 1994), de forma a tornar a informação útil para que as pessoas possam atuar com eficácia e garantir a competitividade organizacional.

MATERIAIS E MÉTODOS

Um setor de procuradoria jurídica do Rio de Janeiro solicitou um sistema de controle dos processos que chegam ao Departamento, para cadastrar pessoas envolvidas, registrar informações e acompanhar seu andamento, com controle de pendências, garantindo, assim, a organização das informações de forma otimizada, evitando retrabalho com perda de tempo e aumentando a rapidez na obtenção das informações.

Mediante reuniões sucessivas no departamento solicitante, identificaram-se as necessidades do sistema, a partir do entendimento de que processo é um documento composto de número, vara, local de abertura, ação, partes (pessoas envolvidas), procurador responsável e objeto da ação.

1) O sistema deveria suportar um cadastro, no qual cada pessoa poderia exercer diferentes posições: procurador, partes, usuários:

- Partes – pessoas que participam do processo, sem acesso ao sistema;
- Usuários – têm acesso ao sistema, através de senha, para cadastrar, atualizar, excluir e pesquisar dados;
- Procurador – recebe os processos e executa as mesmas funções dos usuários.

2) Uma pessoa pode estar envolvida em vários processos, podendo ser autor ou réu no processo;

3) Um procurador pode ser responsável por vários processos;

4) Uma mesma ação pode ser movida por diferentes pessoas (partes);

5) Uma mesma ação pode pertencer a vários processos;

6) O número inicial do processo é único, mas este pode ter vários números adicionados a ele;

7) Um objeto pode estar presente em vários processos.

Antes de começar a desenvolver o sistema, foi realizada uma pesquisa de sistemas já existentes no mercado. Embora não exista um conceito formado sobre o que vem a ser um ambiente computacional, a revisão bibliográfica seletiva sobre o tema possibilitou perceber que um ambiente computacional integra os recursos de um "Sistema de Informação". Esse Sistema, segundo O´Brien (2003), é formado por Dados, Redes, Hardware, Software e Pessoas. No conceito de Turban *et al*. (2003),

> *Um sistema de informação – SI – coleta, processa, armazena, analisa e dissemina informações com o propósito especifico. Como qualquer outro sistema, um sistema de informação abrange entradas (dados), saídas (relatórios, cálculos), processa essas entradas e saídas, e gera saídas que são enviadas para o usuário ou outros sistemas. É possível incluir um mecanismo de resposta – feedback – que controle a operação. E como qualquer outro sistema, um sistema de informação opera dentro de um ambiente.*

Segundo Rezende e Abreu (2000), um sistema de informação eficiente pode ter um grande impacto na estratégia corporativa e no sucesso da empresa. Entre os benefícios que as empresas buscam, estão: suporte à tomada de decisão, valor agregado ao produto, melhor serviço e vantagem competitiva, produtos de melhor qualidade.

Foram encontrados o ProcessMaker e o o Sistema de Protocolo Eletrônico de Documentos (SPED).

O ProcessMaker é um sistema de controle de processos gratuito e de código aberto, capaz de gerenciar o fluxo de um processo. O sistema possibilita extensa personalização do fluxo de documentos, em que cada etapa pode apresentar formulários próprios. É possível trocar informações via WebServices e consultar informações de base de dados externas para apresentar ao usuário do ProcessMaker durante a utilização do sistema.

O Sistema de Protocolo Eletrônico de Documentos (SPED) é um sistema WEB que surgiu da necessidade de integrar o controle na troca de documentos internos e externos das Organizações Militares do Exército. A partir dessa necessidade, o sistema foi desenvolvido pelo Exército para controlar o protocolo de documentos. Com o SPED, é possível enviar documentos eletronicamente, tais como memorandos, ofícios. A funcionalidade que chamou a atenção foi a possibilidade de assinar o documento digitalmente. O Portal do Software Público prevê a perspectiva de, no futuro, integrar-se com o correio corporativo para trâmite de documentos entre outras organizações que possuam o sistema instalado.

Ambos os sistemas encontrados apresentaram funcionalidades interessantes, mas exigiam dedicação de tempo para aprendizagem e para adaptação dos mesmos aos problemas detectados.

Devido ao crescimento de metodologias ágeis e sabendo que as informações e os sistemas de informações devem ser precisos, personalizados e imediatamente disponíveis em um formato que facilite o espaço e o uso (Terra, 2001), decidiu-se pelo desenvolvimento do sistema em uma linguagem que seguisse esse princípio; assim, o tempo despendido em aprender a ferramenta seria recompensado em novos projetos de necessidade da Instituição.

Considerada a decisão da elaboração de um sistema mediante a identificação das necessidades, realizou-se uma análise documental na qual foi definido o diagrama de classes do sistema, que listou todos os conceitos do domínio que foram implementados no sistema e as relações entre esses conceitos.

Linguagem Ruby on Rails

Ruby é uma linguagem de script interpretada para programação orientada a objetos. A linguagem foi criada no início da década de 90 no Japão e rapidamente ganhou popularidade no mundo inteiro, por sua filosofia deter foco nas pessoas (Antonio, 2008).

Ruby on Rails é um framework livre e de código aberto, também chamado de Rails ou RoR, para desenvolvimento de aplicações Web, implementado em Ruby (Williams, 2007). Segundo informações no próprio site do fabricante do *framework*, o Ruby on Rails foi projetado para: ser uma solução de desenvolvimento completa; ter as suas camadas se comunicando da forma mais transparente possível; ser uniforme, escrito totalmente apenas numa linguagem; e seguir a arquitetura MVC (Model-View-Controller). Essas características tornam o Ruby on Rails extremamente produtivo e mantêm baixa a curva de aprendizagem.

Segundo informações do fabricante, é possível instalar o Ruby no Linux, Mac OS X e Microsoft Windows. Após a instalação do Ruby, deve-se instalar o Ruby Gems, que é o gerenciador padrão de pacotes; com ele é possível instalar o Ruby on Rails e outros pacotes para diversas finalidades. Com um único comando, o Ruby Gems baixa e instala os pacotes solicitados.

Componentes

Na instalação do Ruby on Rails, é instalado também um conjunto completo de componentes para criação de aplicações web, tais como: Action Controller, Action View, Active Record, Action Mailer, Active Resource e Active Support.

Esses componentes diminuem o tempo de desenvolvimento de sistemas com o Ruby on Rails, fornecendo todas as ferramentas necessárias para a criação de aplicação web, incluindo persistência, lógica e apresentação. Existem dois conceitos que visam aumentar a produtividade do desenvolvedor: DRY e Convention over Configuration. Esses métodos estão implementados por todo o Ruby on Rails.

DRY – "Don't Repeat Yourself"

É o conceito por trás da técnica de definir nomes, propriedades e códigos sem repetição (Thomas e Hansson, 2006). Desta forma, o retrabalho pode ser evitado, já que esse código pode ser reaproveitado em outros lugares quando necessário.

Isso é possível devido a aplicação necessitar de uma escrita em um único lugar, conforme sugerido pela arquitetura MVC. Em alguns frameworks, uma mudança simples na estrutura da base de dados envolve a mudança de diversos arquivos; porém, com o Ruby on Rails esse impacto pode ser menor (Thomas e Hansson, 2006).

Convention over configuration

Sugere assumir valores-padrão baseados em convenções. Normalmente, cada equipe possui uma

padronização em que pode definir os nomes de tabelas, nomes de campos, nomes de arquivos. Esse procedimento reduz o tempo gasto para entender o código e aumenta a produtividade. No Ruby on Rails, além dessas vantagens, é possível escrever menos códigos de programação e de configuração. Por exemplo, o Rails convencionou que a chave primária de uma tabela chama-se "*id*"; é possível utilizar outro nome, mas dessa forma não é necessário escrever nenhuma linha de código.

Mapeamento objeto-relacional através do Active Record

O Active Record é o módulo que trata do acesso ao banco de dados, possuindo uma camada ORM – *Object-relational mapping*, ou mapeamento objeto-relacional, em português –, que trata de transformar os dados das tabelas do banco em objetos Ruby (Cunha Neto), onde as tabelas são mapeadas em classes, os registros em objetos, e as colunas em atributos de objetos (Thomas e Hansson, 2006).

Caso exista uma tabela *Processos*, teremos uma classe chamada *Processo*. Cada processo no banco de dados representa um objeto *Processo*. Junto com este objeto, são criados métodos para pegar e definir o valor de cada campo ou coluna da tabela.

CRUD (Create, Read, Update, Delete)

O Active Record torna fácil a implementação das quarto operações básicas em uma tabela do banco de dados: criar, ler, alterar e apagar (Thomas e Hansson, 2006).

O Ruby on Rails possui ferramentas para gerar parte do código necessário para manipular o modelo, ou seja, é criada a interface totalmente funcional, em que o usuário pode incluir, ver, alterar e deletar os registros. Essa interface pode ser adaptada conforme a necessidade do sistema.

MVC – *Models, Views e Controllers*

É um padrão utilizado pelo Ruby on Rails para o desenvolvimento em três camadas de software. Segundo Bezerra (2006), uma camada de software, ou simplesmente camada, é uma coleção de unidades de software, tais como programas ou módulos que podem ser executados ou acessados. Essas camadas normalmente recebem os seguintes nomes: camada de apresentação (*view*), camada de camada da lógica do negócio (*model*), camada de acesso (*controller*).

Camada de apresentação (*view*)

A camada de apresentação é composta de classes que constituem a funcionalidade para visualização dos dados pelo usuários e interface com outros sistemas (Bezerra, 2006). Também chamada de *view*, essa camada é responsável pelo desenvolvimento da interface com o usuário. No Rails, o módulo responsável por isso é o Action View.

Camada da lógica do negócio (*model*)

A camada da lógica do negócio é composta de classes que validam os dados da camada de apresentação e realizam computações com base nos dados armazenados ou nos dados de entrada (Bezerra, 2006).

No Ruby on Rails, a camada da lógica do negócio é chamada de model, ou modelo, em português. O modelo é implementado pelo Active Record, que além das responsabilidades de validação de dados e computações, é responsável em manter a persistência dos dados, ou seja, ele armazena as informações na base de dados.

Camada de acesso

Segundo Bezerra (2006), *a camada de acesso contém classes que se comunicam com outros sistemas para realizar tarefas ou adquirir informações*. Também chamada de *controller*, essa camada é responsável por gerenciar as requisições e a interação entre as camadas de negócio e de apresentação.

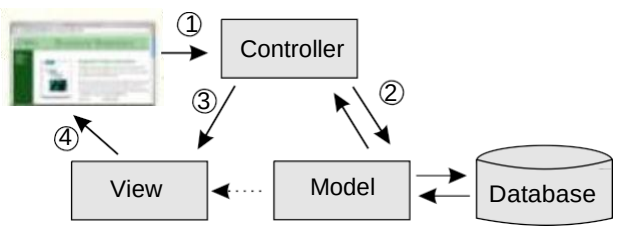

Figura 1
Arquitetura MVC
Fonte: Thomas e Hansson, 2006

A Figura 1 ilustra o fluxo de informações entre as camadas do MVC. As requisições do usuário ou de outros sistemas são recebidas pelo *controller*. Quando necessário, acessa o *model* para incluir, excluir, atualizar ou consultar informações armazenadas. Após todo o processamento necessário, o *controller* chama a *view* para exibir os dados processados. A *view* pode consultar o *model* para pegar informações adicionais.

Layout da aplicação

O módulo Action View encapsula toda a funcionalidade necessária para renderizar templates, mais comumente gerando código HTML e XML para o usuário. Pelo poder e simplicidade da linguagem Ruby, não é difícil imaginar que ela é utilizada nos templates em sua forma embutida (Raymond, 2006), ou seja, não é necessário aprender outra linguagem para os templates, pois ela é a mesma utilizada para desenvolver todo o resto da aplicação. Esse tempo economizado é importante, pois cada sistema precisa ser estudado antes de personalizar a interface da aplicação.

DESENVOLVIMENTO DO SISTEMA

Após a escolha da linguagem de programação e levantamento de requisitos, foi construído o diagrama de classes para representar o que deve ser armazenado pelo sistema e a interação entre os elementos que compõem os dados. Este diagrama foi elaborado seguindo a filosofia *convention over configuration* do Ruby on Rails. Isso potencializou o resultado final, pois aproveitou de forma eficiente as ferramentas do Ruby on Rails responsáveis pelo CRUD (create, read, update, delete). Foi utilizado o *script scaffold* para construir esta estrutura necessária para as principais operações. O diagrama de classes foi usado durante toda a primeira fase de desenvolvimento e evitou o retrabalho, pois o *script scaffold* foi utilizado uma única vez. Por apresentar uma estrutura básica, foi necessário alterar alguns dos arquivos gerados automaticamente, ou seja, mesmo assim, a interação de um especialista faz-se necessária.

Para exemplificar a clareza e a redução do código desenvolvido em Ruby on Rails foi incluída a classe Andamento.

1. *class* Andamento < *ActiveRecord::Base*
2. *belongs_to* :processo
3. *validates_presence_of* :data, :observacao
4. *end*

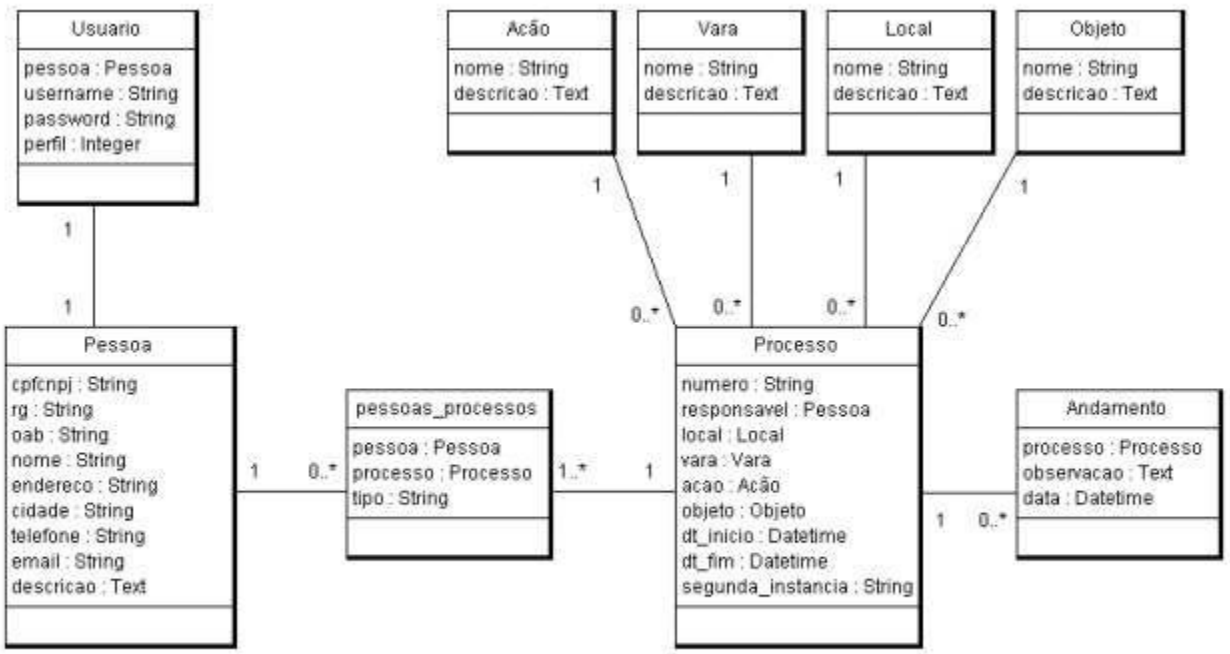

Figura 2
Diagrama de classes do projeto adaptado ao Rails, desenhado no ArgoUML

Seguindo o diagrama de classes, observa-se que a classe Andamento faz parte de um processo. Esse tipo de relacionamento foi efetivamente definido na linha 2 do código. A linha 3 é responsável em validar a presença da data e da descrição do que aconteceu com o processo; ao tentar cadastrar um andamento sem informar uma das informações, o sistema interrompe o processamento e exibe uma mensagem explicando o que deve ser feito para continuar. As linhas 1 e 4 foram geradas pelo Ruby on Rails automaticamente através do *script scaffold* e definem, respectivamente, o ínicio e o fim de uma classe do módulo Active Record, responsável por mapear o banco de dados em objetos Ruby.

O Ruby on Rails utiliza a linguagem Ruby ao extremo para facilitar a vida do programador, tornando o código curto e mais legível. Isso permite realizar tarefas no próprio código, ao invés de utilizar configuração externa (Thomas e Hansson, 2006).

A classe Processo é responsável em definir a integridade dos dados informados pelos usuários e os relacionamentos em relação às demais classes do Sistema, obedecendo ao diagrama de classes. Essa configuração de relacionamentos permite ao Active Record criar automaticamente algumas funcionalidades de consulta, conforme a Tabela 1.

Exemplos de comandos	**Comentários**
Processo.all	Lista todos os processos cadastrados.
p = Processo.find(:first)	Define **p** como o primeiro processo da base de dados.
p.responsavel, p.local, p.objeto, p.vara	Retornam, respectivamente, o responsável pelo processo, o local de abertura do processo, o objeto do processo e a vara onde se encontra.
p.pessoas, p.andamentos	Retornam, respectivamente, todas as pessoas envolvidas no processo e todos os registros de andamento do processo.
p.responsavel.nome	Retorna o nome do responsável do processo.

Tabela 1
Possíveis comandos de consulta de processos

A programação da classe permite utilização imediata de um conjunto de consultas, sem a necessidade de uma linguagem específica para o tratamento dos registros. Além de diminuir linhas de código, diminui a chance de erros de programação, por evitar a repetição de tarefas exaustivas. O mapeamento pelo Active Record permite a manipulação e definição de dados em um formato geral que o próprio Ruby on Rails se encarrega de adaptar ao banco de dados utilizado.

SISTEMA PRODUZIDO

Ao informar corretamente o nome de usuário e a senha, são exibidos os processos pendentes pelos quais o usuário é responsável.

Figura 3
Tela inicial com processos pendentes

O cadastro de novo processo faz-se de maneira bastante simples. Primeiro, são fornecidas informações pertinentes diretamente ao processo e relacionamentos simples, para os quais basta selecionar o nome do campo respectivo. Posteriormente, é possível relacionar as partes envolvidas no processo e incluir os registros de andamento. Essa interface utiliza abas para facilitar a navegação e AJAX para facilitar o preenchimento das informações

Figura 4
Detalhes do processo

Figura 5
Aba para vincular as partes

Figura 6
Aba para registrar o andamento

RESULTADOS E DISCUSSÃO

O sistema foi desenvolvido em Ruby on Rails por trazer ganhos de produtivade pelas suas características, sendo capaz de motivar a equipe desenvolvedora por apresentar um código enxuto. Esse aumento da produtividade, na maior parte dos estudos, tem como fator principal a motivação. No site oficial do produto, é possível assistir à criação de um sistema para weblog em 15 minutos.

Cunha Neto relata o desenvolvimento de duas aplicações com a mesma base de dados: uma na linguagem Java, com framework Struts e Hibernate, e outra em Ruby com framework Rails, com o intuito de comparar quantitativamente o esforço necessário para concluir o trabalho e configurar a aplicação. Foi comprovada a eficiência do Ruby on Rails através dos gráficos apresentados em seu projeto final.

Em seu projeto, a proporção é de 5,66 vezes mais código escrito em linguagem Java para representar as mesmas funcionalidades que foram desenvolvidas em linguagem Ruby. Um dos motivos está na linguagem Ruby, que consegue ser mais concisa e realizar mais funcionalidades com menos código. A outra, está nas funcionalidades oferecidas pelo Rails, como o sistema de validação de modelos utilizando os métodos validades, que possibilitam o desenvolvedor escrever menos código, para preocupar-se mais com os problemas de sua aplicação e deixar com o framework os problemas de implementação. A seguir, um gráfico comparativo.

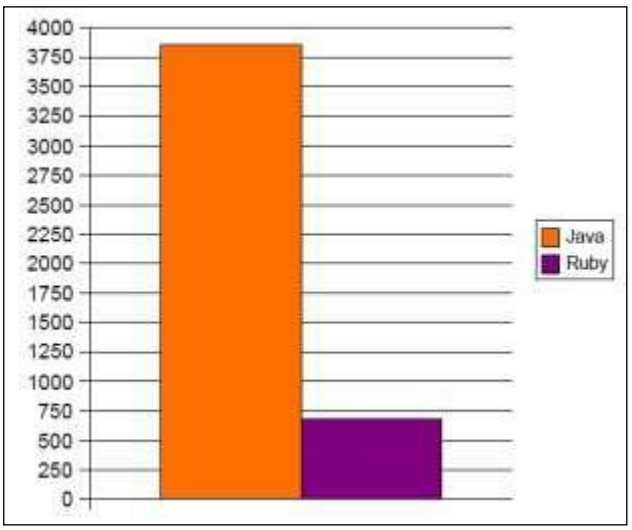

Figura 7
Gráfico comparativo linhas de código
Fonte: Cunha Neto

O segundo gráfico produzido por Cunha Neto apresenta a quantidade de linhas necessárias para configurar os sistemas. O esforço necessário para ter o mesmo resultado no Ruby on Rails foi significativamente superior, o que comprova a eficiência do conceito *convention over configuration*. Enquanto para configurar o Ruby on Rails foi necessário editar 18 linhas de código de um único arquivo YAML, no Java, a quantidade de linhas de código XML foi de 273 linhas.

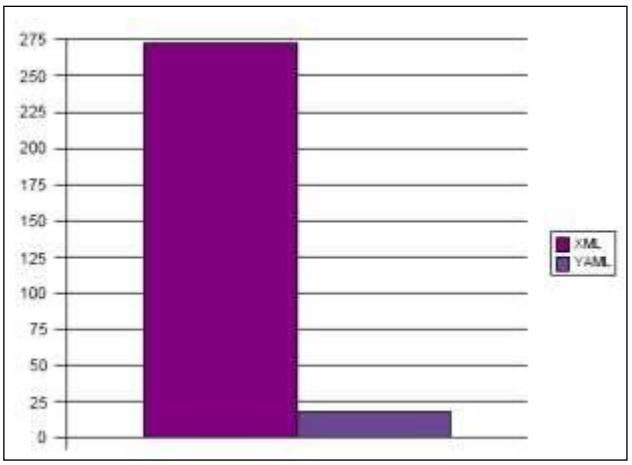

Figura 8
Gráfico comparativo linhas em arquivos de configuração
Fonte: Cunha Neto

As Tecnologias de Informação e Comunicação (TIC) são extremamente importantes no processo de construção e disseminação do conhecimento nas organizações. Os avanços da tecnologia da informação facilitam os processos requeridos pela Gestão do Conhecimento, como a coleta, a seleção, a disponibilização e a disseminação de informações. Porém, existe uma barreira final, que é a tradução pelos homens dessa informação e a sua transformação em ações. A tecnologia da informação, por si só, não assegura um processo eficaz de Gestão do Conhecimento, mas pode propiciar um relevante suporte para a implantação da Gestão do Conhecimento em uma organização.

Sendo assim, o sucesso de um projeto de Gestão do Conhecimento nas organizações não pode estar atrelado à implantação de uma nova tecnologia da informação. A tecnologia da informação, isoladamente, não pode ser a base da criação e gestão de conhecimento organizacional. A tecnologia deve ser encarada como suporte dentro desse processo.

A TI possui um papel importante no suporte para a implementação da gestão do conhecimento nas organizações, atentando que a função mais valiosa da tecnologia, na Gestão do Conhecimento, é aumentar o alcance e a velocidade da transferência do conhecimento. A tecnologia ainda possibilita que o conhecimento de uma pessoa ou de um grupo seja extraído, estruturado e utilizado por outros membros da organização e por seus parceiros de negócios no mundo todo, ajudando na codificação e geração do conhecimento.

CONCLUSÃO

Com a finalização e implantação do Sistema, o resultado foi satisfatório, tendo em vista a conclusão dos principais requisitos levantados na fase inicial do projeto. Assim, obteve-se uma visão ampla sobre o panorama da TI na Instituição em que foi desenvolvido o sistema. Observou-se que, mesmo com dificuldades, as expectativas podem ser respondidas com êxito, ressaltando a TI como fator essencial para a otimização da informação, sendo indispensável para resposta rápida da demanda de processos, como ferramenta relevante para a tomada de decisões estratégicas e administrativas.

Desta forma, pode-se concluir que o sistema criado segue os padrões de desenvolvimento e que os objetivos propostos foram alcançados mediante a implementação desse sistema, fácil e flexível, fornecendo aos usuários (procuradores) informações seguras e instantâneas.

Referências bibliográficas

AKITA, F. *Entendendo o Rails*. Disponível em: <http://www.akitaonrails.com/files/Entendendo_Rails.pdf>. Acesso em: Ago.2009.

ANTONIO, C. S. *Aprendendo Ruby on Rails*. Disponível em: <http: //www.akitaonrails.com/files/Entendendo_ Rails.pdf>. Acesso em: Ago.2009.

LEHMKUHL, G. T; VEIGA, C. R.; RADO, G. J. V. O Papel da Tecnologia da Informação como Auxílio à Gestão do Conhecimento. *Revista Brasileira de Biblioteconomia e Documentação*, v.4, n.1, p.59-67, jan/jun.2008.

CÂNDIDO, G. A.; FILHO, J. F. S. *Aplicação da tecnologia da informação como ferramenta de apoio para a inteligência competitiva e a gestão do conhecimento: um estudo de caso no setor varejista*. São Paulo, 2003. Disponível em: <http: //www.seer.furg.br/ojs/index.php/index/about>. Acesso em: Ago.2009.

CÔRTES, P. L. *Administração de Sistemas de Informação*. São Paulo: Saraiva, 2008.

FREITAS, M. P.; AMARAL, R. M. *Otimização do guia de procedimentos de trabalho para o departamento de processamento técnico da bco/UFSCar*. Disponível em: http://www.sbu.unicamp.br/ snbu2008/anais/site/pdfs/2528.pdf>. Acesso em: Ago.2009.

GARVIN, D. A. Building a learning organization. *Harvard Business Review*, v.71, n.4, p.78-91,1993.

CUNHA NETO, S. M. *Rails Versus Struts: Um Comparativo de Frameworks*. Disponível em: http://mergulhao.info/assets/2007/5/2/monografia.pdf>. Acesso em: Ago.2009.

McGEE, J.; PRUSAK, L. *Gerenciamento estratégico da informação: aumente a competitividade e a eficiência de sua empresa utilizando a informação como ferramenta estratégica*. Rio de Janeiro: Campus, 1994.

O'BRIEN, J. A. *Sistemas de informações e as decisões gerenciais na era da internet*. 9. ed. São Paulo: Saraiva, 2003.

PORTAL DO SOFTWARE PÚBLICO. http://www.softwarepublico.gov.br>.Acesso em: Ago.2009.

RAYMOND, S. *Ajax on Rails: Build Dynamic Web Aplications with Ruby*. United States of America: O'reilly, 2006.

REIS, D. R. *Gestão da Inovação Tecnológica*. São Paulo: Manole, 2004.

REZENDE D. A.; ABREU A. F. *Tecnologia da informação aplicada a sistemas de informação empresariais*. São Paulo: Atlas, 2000.

RUBY ON RAILS. http://www.rubyonrails.org/. Acesso em: Ago.2009.

THOMAS, D.; HANSSON, D. H. *Agile Web Development with Rails*. 2. ed. United States of America: The Pragmatic Bookshelf, 2006.

TERRA, J. C. C. *Gestão do Conhecimento: o grande desafio empresarial*. São Paulo: Negócios Editora, 2001.

ZANETI JUNIOR, L. A. *Sistemas de Informação baseados na tecnologia WEB: um estudo sobre seu desenvolvimento.* Dissertação de Mestrado, FEA/USP, São Paulo, SP, Brasil, 2003. 189p.

WILLIANS, J. *Rails solutions: Ruby on Rails made easy*. United States of America: Springer-Verlag, 2007.

Dados dos autores

Marta Angela de Almeida Sousa Cruz (martasousacruz@gmail.com), assistente em Administração do CEFET/RJ, é graduada em Nutrição pela UERJ, pós-graduada em Obesidade e Emagrecimento pela UGF/RJ, e estudante do curso de graduação em Administração Industrial do CEFET/RJ.

Rômulo Mendes Figueiredo (contato.romulo@yahoo.com.br), técnico de Tecnologia da Informação do CEFET/RJ, é graduado em Tecnologia de Desenvolvimento de Sistemas para Web pelo CEFET/RJ e estudante do curso de graduação em Administração Industrial do CEFET/RJ.

Rosangela Mourat da Rocha Ávila (rmravila@gmail.com), professora do CEFET/RJ, é graduada em Ciências Contábeis pela Faculdade Moraes e Júnior, e mestre em Tecnologia pelo CEFET/RJ.