8
Análise de Dados

Neste capítulo está descrito todo o trabalho realizado durante este estudo, desde o planejamento das tarefas a serem realizadas, passando pelas dificuldades surgidas, pelos pontos positivos e negativos encontrados, pelos recursos utilizados, chegando aos dados coletados e às conclusões desenvolvidas.

Este capítulo apresenta o conjunto de métricas que foi utilizado com a finalidade de avaliar o trabalho, no sentido de verificar o impacto das mudanças realizadas. A partir dos objetivos estabelecidos para este estudo, criaram-se algumas métricas (que serão detalhadas na Seção 8.2) segundo o modelo GQM (referenciado na Seção 3.4) para auxiliar a medição dos dados coletados. Sem a utilização de métricas, torna-se muito difícil a atividade de avaliar as melhoras obtidas a partir de mudanças propostas, portanto, esta seção é de extrema importância para a avaliação do resultado deste trabalho.

Na Seção 8.4, as dificuldades encontradas durante a realização deste trabalho são relatadas. Esta seção também é bastante importante para este estudo, uma vez que pode auxiliar trabalhos semelhantes, funcionando como uma base de lições aprendidas.

Na Seção 8.6, são descritos os dados coletados (a partir das métricas estabelecidas) e as conclusões relacionadas aos dados, isto é, a análise desenvolvida como resultado. Nas seções seguintes, são mencionados os pontos positivos e negativos da utilização de *Scrum* entre os integrantes da unidade e da reestruturação do ambiente de trabalho, incluindo a utilização de um conjunto de ferramentas padrão.

8.1.Planejamento

As mudanças principais realizadas neste trabalho foram a implantação do *Scrum* e o desenvolvimento de uma infraestrutura de trabalho que facilitasse o cotidiano dos integrantes da equipe e que diminuísse a possibilidade de erros

durante o processo de alteração de uma página ou do desenvolvimento de uma nova funcionalidade.

O ponto de partida para a realização deste estudo foi fazer a equipe entender a necessidade de utilização de um método de gerenciamento dos projetos e a importância de se utilizar um método de gerenciamento de configuração, mais especificamente, mostrar os benefícios atingidos com o uso de um sistema de controle de versão.

Grande parte da equipe – a maior parte dela – não é composta por desenvolvedores ou pessoas da área técnica, portanto, é natural que os integrantes da unidade não tenham conhecimento destas tecnologias. Dois workshops foram criados para apresentar os conceitos relacionados ao *Scrum* e ao controle de versão, com o objetivo de obter uma aderência entre as pessoas que participariam do trabalho.

Sabe-se que mudanças são acompanhadas de muita resistência e de dificuldades ao longo do caminho, porém, era de interesse dos superiores que estas técnicas passassem a ser adotadas pela equipe. Portanto, estes workshops foram muito importantes para esclarecer o que poderia ser melhorado a partir das mudanças propostas.

Uma vez que toda a equipe sentiria os impactos das mudanças, era necessário que os integrantes da mesma se sentissem confortáveis com isto e que estivessem dispostos a contribuir para uma melhoria na realização de seu trabalho diário e também na melhoria do resultado final – o produto desenvolvido.

A primeira parte das mudanças realizadas está relacionada à infraestrutura de trabalho, que será detalhada na Seção 8.3. Aspectos como alteração de hardware, conjunto de softwares que passaram a ser utilizados, padrão de codificação criado para os desenvolvedores e com a ajuda destes, utilização de um framework para Coldfusion e a criação de uma estrutura de gerenciamento de configurações serão citados na Seção 8.3.

Decidiu-se alterar a infraestrutura antes de iniciar a utilização de *Scrum* por acreditar que era necessário organizar o ambiente de trabalho antes de implantar um método de gerenciamento de projetos. *Scrum* utilizado sozinho não faz milagres, é importante que ele esteja acompanhado de métodos e processos da Engenharia de Software durante o desenvolvimento de um projeto [18].

O segundo passo foi a utilização de *Scrum* em alguns projetos que foram desenvolvidos, os quais eram formados praticamente por desenvolvedores, aqueles que conhecem mais as práticas da Engenharia de Software e que estariam mais propensos a colaborar em um primeiro momento. Decidiu-se por *Sprints* curtos (de uma semana, uma vez a unidade não utilizava qualquer método de gerenciamento de projetos anteriormente e suas prioridades mudavam constantemente). *Sprints* mais curtos são mais vantajosos quando ainda não há maturidade para planejamentos longos de trabalho.

Por último, pequenas alterações em ferramentas existentes foram realizadas para atender a demandas da unidade, como por exemplo, a preocupação com o SEO; também se pensou na construção de um mecanismo de atualização dos servidores quando novas versões do site fossem disponibilizadas.

8.2.Métricas Utilizadas

Como citado na Seção 3.4, alguns objetivos foram estabelecidos neste trabalho e, com a utilização do Modelo GQM, geraram-se métricas para avaliar os impactos das mudanças realizadas. Desta maneira, para cada objetivo (*Goal*) listado a seguir, têm-se os dados referentes à pergunta (*Question*) e às métricas (*Metric*) utilizadas.

8.2.1.Diminuir o número de erros reportados por mês

A cada vez que ocorre um erro nas páginas do site, um email é enviado aos desenvolvedores, reportando o erro que ocorreu. Muitas vezes, estes emails acompanham uma grande mudança de versão do site: uma seção nova foi criada ou grande parte de uma seção foi atualizada.

Assim, este problema está mais relacionado ao trabalho dos desenvolvedores do que ao dos editores. Há duas causas principais para a ocorrência destes problemas: falta de cuidados na atribuição de parâmetros padrão nas páginas e erros cometidos no momento de atualizar o site (esquece-se de alguns arquivos). Desta maneira, estabeleceu-se o seguinte modelo:

Goal: diminuir em 10% a quantidade de erros reportados mensalmente.

Question: Qual a média de erros reportados em um mês após as mudanças? Qual a média de erros reportados em um mês antes das mudanças?

Metric: Número de erros reportados em um mês. Relação com o número de erros e a implementação das mudanças.

Esta melhoria está relacionada ao produto de trabalho desenvolvido na unidade. Para atingir os objetivos aqui estabelecidos, seria necessário melhorar o processo de desenvolvimento dos softwares(utilzando boas práticas da Engenharia de Software) e melhorar o processo de atualização de versões, que ocasionava muitos erros posteriores.

8.2.2.Melhorar o processo de atualização do site

A cada vez que o site precisa ser atualizado, seja para adição de uma nova funcionalidade, seja para pequenas alterações, é necessário copiar os arquivos manualmente, um por um, para o servidor desejado (muitas vezes para mais de um servidor).

Deste modo, este processo, além de ser muito trabalhoso, está sujeito a falhas, uma vez que a pessoa que estiver fazendo a cópia pode se esquecer de algum arquivo ou copiar algo para o lugar errado, ambos ocorrem com freqüência. Portanto, um dos objetivos deste trabalho é melhorar este processo, para facilitar o trabalho da equipe e diminuir a quantidade de falhas inseridas. Para este objetivo, estabeleceu-se o seguinte modelo:

Goal: Diminuir o tempo de atualização do site em 20%.

Question: Quanto tempo é necessário para atualizar os arquivos de um servidor após a mudança? Quanto tempo, em média, era gasto para atualizar os arquivos de um servidor antes da mudança?

Metric: Minutos gastos para a atualização de um servidor. Número de atualizações feitas por mês. Quantidade de atualizações corretas e completas por mês.

Esta melhoria está relacionada ao processo de trabalho seguido pela equipe, uma vez que representa mudanças na maneira em que se deveria atualizar os arquivos alterados nos servidores. Porém, esta melhoria se reflete no produto em si, pois este passaria a ter menos erros introduzidos na sua atualização.

8.2.3.Diminuir os problemas de desperdício de trabalho

Uma vez que os projetos desenvolvidos na unidade estudada não seguiam um método de gerenciamento de projetos, ficava difícil saber o que estava sendo desenvolvido pelos seus integrantes e qual o status dos mesmos. As prioridades mudam constantemente e muitas vezes se desenvolve algo que não é mais de interesse da gerência no momento em que está pronto. Por último, por não haver um sistema de controle de versão de código anteriormente, era comum acontecer de duas pessoas alterarem um mesmo arquivo simultaneamente, ocasionando perda de trabalho.

Deste modo, foi desenvolvido o seguinte modelo para evitar estes problemas (aqui as métricas são subjetivas, uma vez que é mais complicado encontrar características para medir estes fatores quantitativamente):

Goal: Diminuir os problemas de desperdício de trabalho.

Question: Você desenvolveu alguma funcionalidade neste mês que não foi utilizada? Você teve que refazer alguma funcionalidade por conta de perda de trabalho (suas alterações sumiram)? Você consegue ter uma visão melhor do status de um projeto – quanto falta para terminá-lo, consegue verificar se está atrasado ou se vai terminar antes da hora?

Metric: As métricas são as respostas do questionário, respondidos por todos os desenvolvedores da unidade (aqueles que utilizaram *Scrum* em uma primeira etapa).

Esta melhoria está intimamente ligada ao método de gerenciamento de projetos, que estabalece as prioridades e decide o que deve ser desenvolvido. Na unidade estudada não havia qualquer tipo de gerenciamento de projetos e se decidiu por utilizar o *Scrum* em alguns projetos para verificar os benefícios de sua adoção.

8.2.4.Aumentar a popularidade do site

A popularidade do site é medida pela ferramenta Web Trends e Google Analytics. O objetivo é aumentar o valor medido por estas ferramentas. Assim, criou-se o modelo a seguir:

Goal: Aumentar em 5% a popularidade do site.

Question: Qual a média mensal de visitantes do site após a mudança? Qual a média mensal de visitantes do site um ano atrás?

Metric: Número de visitantes por mês.

Esta melhoria está ligada ao produto de software desenvolvido. A partir de algumas funcionalidades novas adicionadas ao gerenciador de conteúdo utilizado, pretendia-se aumentar a popularidade do site. Essas funcionalidades foram citadas na Seção 7.3.1 e foram desenvolvidas tendo o *Scrum* como método ágil de gerenciamento.

8.2.5.Aumentar a popularidade das mídias sociais e seus conteúdos relacionados

Neste trabalho, também se teve a preocupação com o assunto relacionado às mídias sociais, muito utilizadas pela unidade. Desenvolveu-se um módulo no gerenciador de conteúdo que facilita a atualização dos conteúdos relativos às ferramentas como Facebook, Youtube e Twitter nas páginas Web da Organização. Este módulo permite uma rápida atualização dos conteúdos relacionados às mídias sociais, promovendo-os.

Assim, as métricas criadas para medir este objetivo estão descritas a seguir:

Goal: Aumentar em 5% a popularidade da Organização nas ferramentas de mídias sociais que utiliza.

Question: Qual a quantidade de amigos da organização no Facebook atualmente? E cincos meses atrás? Qual a quantidade de usuários inscritos no canal Youtube da organização atualmente? E cinco meses atrás? Qual a quantidade de seguidores da organização no Twitter atualmente? E cinco meses atrás?

Metric: Quantidade de pessoas que seguem a instituição no Facebook hoje. Quantidade de pessoas que seguiam a instituição no Facebook cinco meses atrás. Quantidade de pessoas que seguem a instituição no e Youtube. Quantidade de pessoas que seguiam a instituição no Youtube cinco meses atrás. Quantidade de pessoas que seguem a instituição no Twitter. Quantidade de pessoas que seguiam a instituição no Twitter cinco meses atrás.

Esta melhoria também é classificada como uma melhoria do produto de software desenvolvido, conforme citado na Seção 7.3.2. Pretendia-se aumentar a

popularidade das mídias sociais com a melhora de seus conteúdos. E essa melhora seria alcançada após incluir no gerenciador de conteúdo utilizado uma funcionalidade que permitisse uma rápida atualização dos dados relativos às mídias sociais no site da organização. O incremento desta funcionalidade também foi desenvolvido tendo o *Scrum* como método ágil de gerenciamento. Assim, os desenvolvedores poderiam adquirir um conhecimento maior de suas práticas, antes de utlizá-lo em projetos maiores.

8.3.Trabalho Realizado

O primeiro passo realizado após a apresentação dos workshops sobre *Scrum* e controle de versão foi a criação de um ambiente propício à utilização do Subversion – ferramenta de controle de versão escolhida. O ambiente integrado de desenvolvimento utilizado era o Dreamweaver 3, que, como mencionado anteriormente neste trabalho, não dá suporte ao Subversion (alguns *plugins* foram testados para este fim, porém, sem sucesso).

Decidiu-se então que outra ferramenta deveria ser utilizada – alguma que desse suporte ao Subversion. Porém era necessário atentar-se ao fato de que grande parte da equipe é formada por pessoas não-técnicas e que poderiam não se adaptar a outras ferramentas.

Deste modo, foi realizado um teste por duas semanas com um conjunto de ferramentas que se assemelham ao Coldfusion Builder (Eclipse + *plugin* CFEclipse + *plugin* Aptana Studio) com alguns editores para verificar se eles estariam dispostos a começar a utilizar esta ferramenta. Porém, eles reportaram que tinham muita dificuldade em manipular esta ferramenta e que preferiam o Dreamweaver 3.

Portanto, decidiu-se por utilizar o Dreamweaver 4, uma vez que este suporta as funcionalidades do Subversion e se assemelha bastante à ferramenta que era altamente utilizada pelos integrantes da equipe. Uma vez disponíveis as novas versões do Dreamweaver, houve um treinamento direcionado ao uso das funcionalidades do Subversion.

Em seguida, foi montado o repositório de dados da parte mais importante do site – aquela que é alterada com mais freqüência. Uma vez que havia uma quantidade muito grande de arquivos no servidor (algo em torno de 4 GB), optou-

se por fazer o controle de versão apenas de um conjunto de arquivos – aqueles mais relevantes. Como mencionado na Seção 6.1, é muito caro e algumas vezes indesejável realizar controle de versão de todos os arquivos. Em vez disto, deve-se escolher o conjunto relevante para realizar o controle de versão dos arquivos.

Desta maneira, o problema de controle de versão foi resolvido: a unidade passou a manter histórico de seus arquivos mais importantes e os integrantes da equipe poderiam utilizar o controle de versão dentro do próprio ambiente integrado de desenvolvimento que estavam acostumados a utilizar. Esta implantação foi realizada no final do mês de novembro. Porém, verificou-se que apenas a utilização de um controle de versão não seria capaz de resolver os problemas encontrados durante as atualizações do site.

Era necessário criar um processo de gerenciamento de configurações para melhorar os processos de mudanças de arquivos. Portanto, além de configurar o repositório de dados e todas as máquinas dos integrantes da equipe, também se criou um documento relacionado à Gerência de Configuração; isto é, como as novas versões deveriam ser lançadas, quando o resultado de um trabalho deveria ser considerado uma nova versão de uma seção do site e não uma simples atualização, entre outras informações.

Este documento possui um conteúdo simples, direto e de fácil acesso (tanto para leitura, quanto para alteração). Ele está dentro da ferramenta Wiki (Google Wiki) criada para a unidade. A partir da utilização das práticas contidas neste documento, organizou-se o processo de atualização do site. Entretanto, este processo ainda era muito demorado e sujeito a falhas, uma vez que era feito manual e individualmente, em cada servidor que se desejava atualizar.

Era necessário então, tentar automatizar este processo. Uma ferramenta Java foi criada para auxiliar o procedimento de alteração do site. Esta ferramenta permite a escolha dos servidores que serão atualizados (a partir dos arquivos presentes no repositório). Nela, é possível escolher um número de revisão específico para copiar os arquivos alterados na revisão respectiva ou então escolher uma versão do site para qual se deseja atualizar.

A ferramenta somente permite atualização caso a versão corrente do site seja menor do que a versão para a qual se deseja atualizar. Assim, foi implantada uma gerência de configuração que, além de possuir um controle de versão,

também cobria o controle de mudanças. Esta ferramenta foi desenvolvida e colocada em produção no início do mês de janeiro.

Uma reclamação recorrente passou a ser a lentidão das máquinas, que possuíam apenas 1GB de memória RAM. Como havia memória disponível na organização, as máquinas foram atualizadas para 2GB, o que melhorou bastante o desempenho das mesmas. Esta alteração de hardware foi feita no mês de janeiro.

O passo seguinte foi o de tentar melhorar o processo de desenvolvimento das novas funcionalidades – isto é, aquele que impacta somente os desenvolvedores. A primeira atividade a ser realizada foi a criação de um padrão de codificação, feito com a ajuda de todos os desenvolvedores. Este documento também é bastante direto, simples e de fácil acesso (disponível no Google Wiki).

Antes deste trabalho, não era utilizado qualquer padrão de codificação, o que tornava a manutenção do código mais complicada, pois diferentes desenvolvedores criavam código de uma maneira bastante diferente. A partir deste documento, o novo código desenvolvido passou a ser mais uniforme – porém, o código legado não seria alterado para se adequar aos padrões instaurados, uma vez que este seria um trabalho com esforço e custo muito altos.

Somente a utilização de um padrão de codificação não é suficiente para garantir uma boa arquitetura e organização do código desenvolvido. Verificou-se que era de extrema importância a utilização de um framework para organizar as várias camadas de código que deveriam existir, separando acesso a banco de dados de regras de negócio e de código relacionado à apresentação da página. Decidiu-se utilizar o Coldbox (http://www.coldbox.org/), que implementa o modelo MVC, pela sua boa aceitação no mercado e também pelo fato de outros times da mesma organização já o utilizarem. Mais uma vez, a mudança seria realizada apenas nos códigos a serem desenvolvidos, ou seja, o código legado continuaria como antes. Estas adoções de boas práticas foram feitas no final do mês de janeiro.

Em seguida, como era uma demanda muito grande da unidade, começou-se a se preocupar com as técnicas de SEO com o objetivo de tentar melhorar a popularidade do site entre as ferramentas de busca da *Web*. Uma vez que se utiliza um gerenciador de conteúdo para atualização do conteúdo do site,
tornou-se necessário adicionar algumas funcionalidades ao mesmo para que as páginas criadas pudessem seguir as boas práticas de SEO. Desta maneira, campos

importantes como título, cabeçalhos e descrição foram adicionados à criação de cada página.

Também foi implementada uma maneira de reescrever as *urls* do site, para que elas apresentassem palavras ao invés de códigos, tornando-as mais amigáveis e confiáveis tanto pelos usuários, quanto pelas ferramentas de busca da *Web*. Utilizando-se a ferramenta Isapi Rewrite (http://www. isapirewrite.com/), dentro do servidor IIS, foi possível realizar esta tarefa.

Outra modificação realizada no gerenciador de conteúdo do site foi a adição de uma funcionalidade de alteração de conteúdo específico das redes sociais. Por meio desta funcionalidade, tornou-se mais fácil e rápido a alteração de partes do site que referenciam as páginas do Facebook, Twitter e Youtube da organização. Estas alterações de incrementos de funcionalidades ao gerenciador de conteúdo foram feitas no mês de fevereiro, utilizando o *Scrum*, como citado na Seção 7.3.

Por último, quando todas as alterações anteriores foram implementadas e se encontravam maduras, começou-se o processo de amadurecimento do *Scrum*, no mês de março. Para isto, foram escolhidos alguns projetos(com duração de dois meses aproximadamente cada) para serem os pilotos deste estudo. Estes projetos só continham desenvolvedores em seu time e os *Sprints* eram de uma semana. O primeiro projeto foi terminado com sucesso, isto é, foi terminado dentro do prazo previsto. As estimativas tiveram sucesso porque se preocupou em deixá-las grandes, pois ainda não se conhecia a velocidade do time e como o mesmo ia desenvolver seu trabalho utilizando este método.

O segundo projeto foi mais organizado, uma vez que o time já havia incorporado os conceitos do *Scrum* e seus integrantes tinham alguma experiência em implementá-los. As estimativas foram mais apertadas, para tentar chegar mais próximo do tempo necessário para realizar o trabalho sem deixar uma brecha muito grande de tempo. O projeto terminou fora do prazo, isto é, após o tempo previsto e um *Sprint* a mais teve que ser criado para terminá-lo. Isto ocorreu devido ao fato de, no período de duração do *Sprint*, ocorreram vários imprevistos que atrapalharam o andamento do trabalho.

O último projeto a ser realizado seguindo *o Scrum* terminou com sucesso, isto é, dentro do prazo esperado e se verificou uma melhora na estimativa de tempo e esforço do time. Neste último projeto, a ferramenta Banana Scrum foi

utilizada para acompanhar o andamento do projeto, permitindo que toda a unidade tivesse acesso aos dados do *Sprint*, *online*.

O uso desta ferramenta agradou bastante o gerente da unidade, uma vez que ele poderia ter acesso aos projetos online, imprimir relatórios, visualizar gráficos, ou seja, fazer muitas atividades que precisava fazer anteriormente, porém não tinha o recurso disponível.

8.4.Dificuldades Encontradas

A primeira dificuldade encontrada foi a multidisciplinaridade da equipe. Apesar de o *Scrum* defender que as equipes devem ser pequenas e multidisciplinares, neste contexto, este foi um fator indesejável, uma vez que grande parte dos integrantes da equipe é composta por pessoas que não tiveram formação em computação ou áreas afins. Isto é, estas pessoas nunca haviam se deparado com conceitos relacionados à Engenharia de Software, controle de versão e métodos ágeis.

Somado a este fato, também se pode citar a característica dos integrantes da equipe, de serem pessoas adversas a mudanças, talvez pela própria tradição da organização, uma grande instituição que não aceita mudanças de maneira rápida. Então, o trabalho de convencimento se tornou ainda mais difícil e desgastante.

Para contornar este problema, foi necessário fazê-los entender a importância de um controle de versão de arquivos, deixando claro que problemas de perda de trabalho causado por alterações simultâneas em um arquivo seriam evitados, entre outras vantagens. Em relação ao *Scrum*, a equipe também não tinha conhecimento sobre este método ágil. Por este motivo, escolheu-se implantar o *Scrum* em projetos que envolveriam somente desenvolvedores em um momento inicial. Após a verificação de sucesso deste método, ele poderia ser expandido para outros times.

Durante o estudo de utilização de um controle de versão, verificou-se que o Dreamweaver 3 não poderia mais ser usado, uma vez que não permitia o uso de Subversion de uma maneira satisfatória (os *plugins* que existem não funcionam da maneira desejada). Assim, a busca de outro ambiente de desenvolvimento que possuísse as funcionalidades do Subversion foi iniciada. Tanto o Dreamweaver 4 quanto o Coldfusion Builder pareciam atender a este requisito.

A próxima dificuldade encontrada foi a burocracia da organização estudada. Foi necessário comprar novas ferramentas (como o Dreamweaver 4 e o Coldfusion Builder), porém, este processo foi bastante demorado, o que limitou um pouco o espaço de tempo em que se pôde observar a evolução do trabalho causado pelas mudanças realizadas. A mesma dificuldade também é observada quando é necessário adquirir hardware.

Como a compra do Coldfusion Builder seria demorada, foi montado um ambiente semelhante a esta IDE (com ferramentas open source) para verificar se os editores conseguiriam se adaptar a esta ferramenta. Uma vez que os editores não demonstraram aptidão para utilizar esta ferramenta, o Coldfusion Builder não seria mais indispensável.

Já o software Banana Scrum, além de possuir uma versão *free*, ele também permite que seja utilizado para fins educacionais sem precisar de uma licença paga – com todas as funcionalidades das versões pagas. Portanto, inicialmente, esta foi a utilizada, enquanto este estudo estava sendo realizado.

A partir da implementação do *Scrum*, novas dificuldades foram encontradas. As pessoas não estavam muito acostumadas a trabalhar em equipe, uma vez que a própria estrutura física da unidade dificulta a interação entre as pessoas: cada integrante da unidade fica em uma sala isolada. Porém, os desenvolvedores normalmente estavam mais próximos e como eram eles os participantes do *Scrum*, ficava mais fácil de reuni-los.

Os primeiros projetos que utilizaram *Scrum* serviram como aprendizado, uma vez que os outros desenvolvedores nunca tinham trabalhado utilizando este método ágil. Apesar de os primeiros projetos terem terminado antes do prazo previsto, foram visíveis os inúmeros erros relacionados às estimativas de tempo e esforço.

Além disto, como os usuários e o PO também não tinham experiência com *Scrum*, algumas vezes o *Product Backlog* não estava pronto nas reuniões de planejamento do *Sprint* e algumas funcionalidades a serem desenvolvidas eram adicionadas durante o *Sprint*. A partir dos próximos projetos, este problema foi reduzido.

Por último, apesar de este trabalho ter utilizado métricas seguindo o Modelo GQM, é difícil avaliar a real influência de algumas alterações aqui propostas, como por exemplo, do aumento da popularidade do site. O site pode ter

aumentado de popularidade por outros fatores e coincidentemente aparece aqui como uma meta atingida. Mas, verificar que este trabalho de fato foi o responsável por aumentar a popularidade do site é muito difícil. Em seguida, muitas métricas subjetivas foram utilizadas, uma vez que era complicado medir quantitativamente alguns pontos.

Assim, é difícil dizer o quanto este trabalho influenciou melhorias em algumas métricas e se fatores externos também contribuíram para esta mudança. Métricas nunca haviam sido a preocupação desta unidade, portanto, apenas alguns meses de trabalho e medições (aproximadamente 6 meses) é uma quantidade pequena para poder avaliar com mais precisão as melhoras atingidas.

Além disto, muitas mudanças foram realizadas, nas mais diversas áreas (criação de uma ferramenta, criação de padrões, mudança na maneira de criar código, utilização do *Scrum*, utilização de controle de versão), então se torna difícil saber o quanto somente o *Scrum* conseguiu melhorar o trabalho da unidade

e quanto que a criação de uma infraestrutura de trabalho baseado em ferramentas, padrões e processos influenciou no atingimento das metas estabelecidas.

8.5.Recursos Utilizados

Os recursos utilizados para a realização deste trabalho consistem em um conjunto de ferramentas, algumas proprietárias e outras *open source*, além de requisitos de hardware. Há algumas ferramentas que foram citadas no Capítulo 5 e que não aparecem aqui, pois a relevância das mesmas não é tão grande quando comparadas à relevância das ferramentas aqui citadas. Entre as ferramentas proprietárias, foram utilizadas:

a) Dreamweaver 3: ferramenta utilizada inicialmente pela organização, que, por não dar o suporte desejado e necessário ao Subversion, não pôde mais ser usada.

b) Dreamweaver 4: ferramenta utilizada para substituir o Dreamweaver 3, uma vez que aquela possui o Subversion integrado e que não apresentaria uma curva de aprendizado muito grande, pois a equipe já utilizava Dreamweaver 3.

c) Coldfusion Builder: ferramenta utilizada pelos desenvolvedores, que possui todos os recursos para atualização de arquivos em diferentes servidores,

integração com o Subversion e é compatível com as diferentes linguagens e tecnologias utilizadas para o desenvolvimento Web: CDML, HTML, Javascript e CSS.

d) Banana Scrum: ferramenta utilizada para guardar as informações dos projetos que utilizaram o Scrum como método de gerenciamento. Esta ferramenta permitia reproduzir no computador todas as alterações realizadas no Sprint Backlog, Product Backlog e Burndown Chart do Scrum, com a vantagem de acesso *online* e rápida impressão de relatórios.

e) Web Trends: ferramenta utilizada para realizar a medição de acesso de usuários ao site da organização.

Além destas ferramentas proprietárias, também foi necessário aumentar a memória RAM das máquinas, uma vez que estes programas são muito pesados, exigindo muito espaço de memória (esta passou a ser de 2GB).

Outro recurso utilizado neste trabalho foi uma ferramenta desenvolvida para a atualização dos servidores, já mencionada anteriormente como parte da implantação de um gerenciamento de configuração.

Por último, utilizaram-se também algumas ferramentas open source, listadas a seguir:

a) Subversion: ferramenta utilizada para a criação e manipulação do repositório de dados que realiza o controle de versão dos arquivos gerados pela unidade estudada.

b) Coldbox: framework utilizado para organizar a arquitetura de desenvolvimento do site, separando apresentação de funcionalidade e visualização.

c) Eclipse + CFEclipse + Aptana Studio: conjunto de ferramentas utilizado para simular um ambiente do Coldfusion Builder, para verificar se os editores teriam uma boa adaptação a este tipo de IDE.

8.6.Análise de Dados Coletados

Nesta seção, encontram-se os resultados obtidos a partir deste trabalho. Utilizando-se as métricas estabelecidas na Seção 8.2, vários pontos puderam ser mensurados e as respectivas análises puderam ser montadas com base nos mesmos.

A seguir, estão apresentados os dados coletados e as análises relacionadas a cada conjunto de métricas GQM criado.

8.6.1.Diminuir o número de erros reportados em um mês

Um dos objetivos deste trabalho era o de diminuir a quantidade de erros reportados por meio de emails automáticos. Decidiu-se que seria interessante medir a quantidade média mensal de emails recebidos em conseqüência destes erros, a fim de verificar se as mudanças propostas impactariam esta média. A meta seria de diminuir em pelo menos 10% a quantidade de erros reportada mensalmente.

A partir das medições feitas entre novembro de 2009 e maio de 2010, foi possível montar a Figura 16, disponível a seguir.

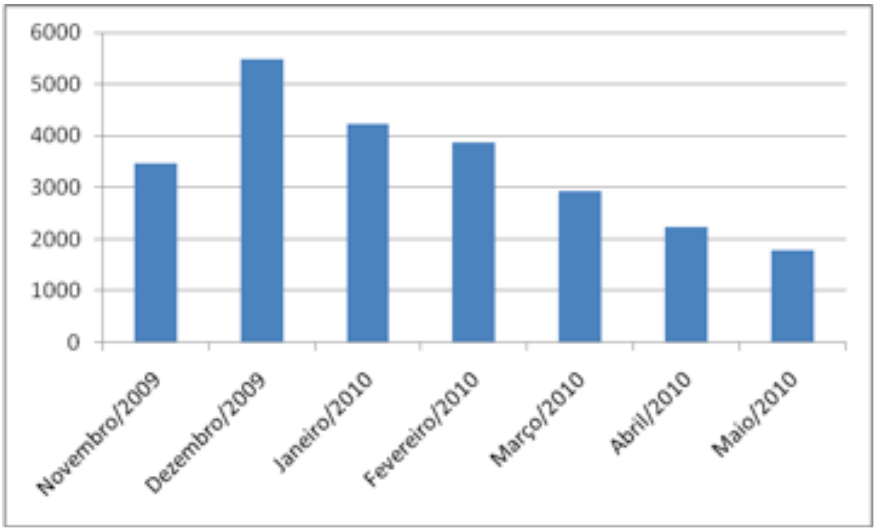

Figura 16 – Quantidade média de erros ocorridos por mês.

Visualizando a figura, é possível verificar que a média de erros caiu de um número entre quatro e três mil para um número menor do que dois mil. Ou seja, a diminuição foi muito maior do que 10% - meta estabelecida. Assim, este objetivo foi atingido com sucesso.

É importante citar aqui quais alterações foram responsáveis por esta mudança na quantidade de erros encontrados. O site da organização não é um sistema de informações, com regras altamente complexas. Trata-se de um conjunto de páginas, algumas delas com conteúdo estático (tratado pelo

gerenciador de conteúdo) e outras com conteúdo dinâmico, criadas pelos desenvolvedores, que utilizam Coldfusion e jQuery principalmente.

A partir de verificações e correções dos erros recebidos, constatou-se que a maioria deles estava ligada a problemas de atualização do site – isto é, quando alguém atualizava o site, esquecia de adicionar todos os arquivos necessários ou cometia alguma outra falha neste processo – ou a erros nas páginas dinâmicas, causadas por pequenos deslizes dos desenvolvedores e de fácil correção.

Deste modo, com a definição de um gerenciamento de configuração, que inclui um plano de mudanças do site, o controle de versão de código e uma ferramenta para auxiliar as atualizações dos servidores, acredita-se que muitos erros foram evitados. O processo de atualização do site se tornou mais fácil e rápido, além de menos sujeito a falhas, devido à automatização de tarefas repetitivas.

A implantação do controle de versão foi feita em novembro e, como se pode observar na Figura 16, acarretou um aumento de erros encontrados no mês seguinte. Este fato se explica pela curva de aprendizado da nova ferramenta.

A equipe não tinha experiência com o Subversion e acabou apresentando alguns problemas na sua adoção durante um período inicial.

Porém, no mês de janeiro, foi implantada uma melhor estrutura de atualização dos arquivos, inclusive foi quando entrou em produção a ferramenta de atualização automática. Neste momento, a equipe já dominava também o Subversion e é possível verificar na mesma figura um decrescimento na quantidade de erros encontrados a partir de então.

Outro ponto de mudança foi a utilização de um padrão de codificação, que contém boas práticas e a utilização de um framework, que organiza a estrutura do código, facilitando sua manutenção. O principal erro que ocorria nas páginas dinâmicas era a falta de inicialização dos parâmetros que a página utiliza. Assim, a inicialização de parâmetros foi adicionada ao padrão de codificação como requisito fundamental e passou a ser implementada pelos desenvolvedores. Com isto, diminui-se bastante a quantidade de erros nas páginas dinâmicas. Vale citar também que o code review passou a ser uma prática obrigatória antes de um commit.

A utilização do framework melhora a manutenibilidade do site, o que também ajudou a diminuir a quantidade de erros gerados em uma atualização,

uma vez que o código se tornou mais organizado e mais fácil de modificar. Estas mudanças foram feitas em janeiro, contribuindo para a diminuição de erros encontrados na figura.

8.6.2.Melhorar o processo de atualização do site

Como mencionado anteriormente, a alteração do site era um processo muito desgastante, uma vez que era necessário copiar manualmente os arquivos desejados, para todos os servidores que receberiam a atualização.

Foi estabelecido, então, o objetivo de diminuir o tempo gasto para a atualização dos arquivos dos servidores. Com o desenvolvimento de um plano de configuração e de uma ferramenta para auxiliar a cópia dos arquivos, foi possível melhorar bastante o processo de atualização dos servidores, tornando-o menos susceptível a falhas e também mais rápido.

Antes de realizar as alterações, estudou-se por um mês a quantidade de tempo gasto para a atualização dos servidores. Cada pessoa que fosse alterar o servidor deveria contabilizar o tempo gasto e enquadrá-lo nas faixas estabelecidas (menos de 15 minutos, entre 15 e 30 minutos, entre 30 minutos e uma hora, entre 1 hora e 2 horas e mais de 2 horas). A partir destes dados, foi possível calcular a média semanal destes dados, que pode ser vista na Figura 17.

A partir desta figura, é possível perceber que as alterações de menos de 30 minutos são maioria. Estas alterações representam atualizações pequenas, isto é, de poucas páginas, realizadas constantemente por todas as pessoas da unidade (tanto desenvolvedores, quanto editores). Já as alterações mais demoradas – aquelas que gastam mais do que 30 minutos – normalmente são realizadas pelos desenvolvedores, uma vez que aqui se trata de novas funcionalidades, que demandam a cópia de vários arquivos e alterações de diversos parâmetros. Estas ocorrem em uma periodicidade bem menor do que as pequenas alterações.

Após as alterações implantadas, independente do tamanho da alteração (total de arquivos que serão copiados) e da quantidade de servidores que serão atualizados, estas alterações são realizadas em um tempo igual e pequeno – não maior do que 10 minutos (uma vez que é preciso uma navegação para conferência das alterações).

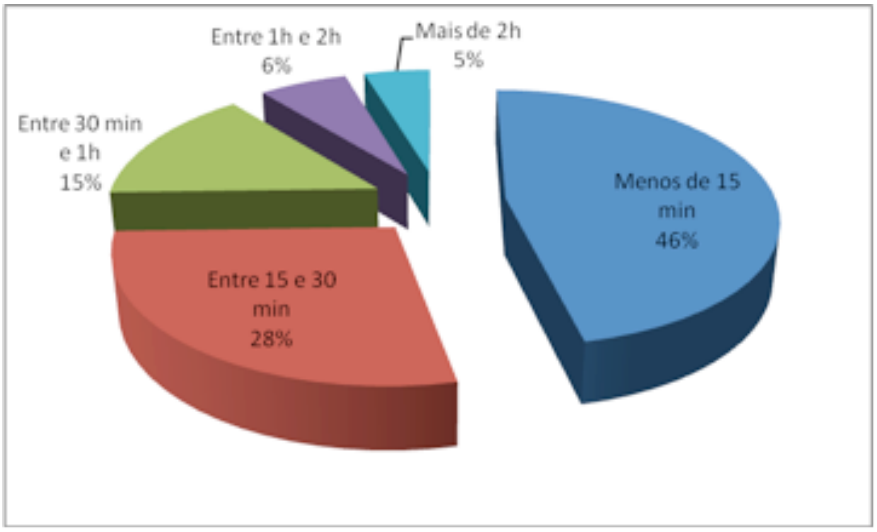

Figura 17 – Percentagem de tempo gasto em atualização dos servidores, medidos em uma semana de trabalho.

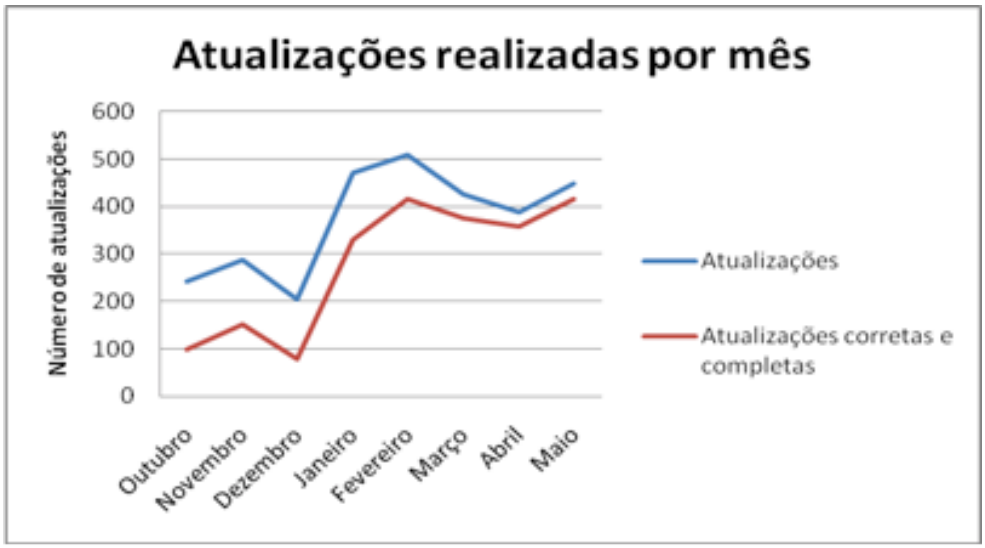

Figura 18 – Atualizações realizadas por mês pela equipe, comparando com o número de atualizações corretas e completas.

Ou seja, grande parte do tempo que era gasto nos processos de atualização do site pôde ser economizada, deixando as pessoas mais satisfeitas, uma vez que não precisam fazer um trabalho tão mecânico e desinteressante. Além disto, há o fato de que muitos erros humanos são evitados, uma vez que as cópias são automatizadas.

A quantidade de atualizações realizadas por mês, bem como a quantidade de atualizações corretas e completas está representada na Figura 18. É importante observar que houve um aumento significativo do número de atualizações, devido

ao volume de trabalho nos respectivos meses, ao aumento da produtividade e também devido ao aumento da equipe.

Em relação ao número de atualizações corretas e completas, a ferramenta de auxílio começou a ser usada em janeiro, passando por melhoras sucessivas. Assim, a partir de março, é possível verificar que as duas linhas estão mais próximas, o que é interessante para a organização. As linhas não se encontram totalmente, possuindo o mesmo número, devido ao fato que, mesmo com a utilização da infraestrutura adotada, alguns problemas de atualização ainda ocorrem. Algumas atualizações ainda são feitas manualmente e ainda é possível melhorar este processo. Porém, é importante notar a grande melhora em relação ao processo.

8.6.3.Diminuir os problemas de desperdício de trabalho

Um problema que acontecia constantemente na organização era o desperdício de trabalho, ocasionado tanto pelo desenvolvimento de trabalho que depois de pronto nunca seria usado, quanto pela perda de trabalho conseqüente da ausência de controle de versão.

Figura 19 – Respostas do questionário relativo à ajuda do Scrum.

Deste modo, foi criado o objetivo de diminuir os problemas relacionados ao desperdício. Para medir o avanço conseguido pelas mudanças propostas, estabeleceu-se uma série de perguntas que deveriam ser respondidas pelos desen-

volvedores apenas, uma vez que estes eram os mais afetados por estes problemas e também seriam os primeiros a utilizar o *Scrum*.

Com a utilização do Subversion, o problema de perda de trabalho, quando duas pessoas alteravam simultaneamente um mesmo arquivo, poderia ser evitado. Além disto, com a utilização do *Scrum*, este por sua vez com *Sprints* semanais, teria como um de seus objetivos evitar o problema de trabalho desenvolvido que não seria usado e também permitir a todos os integrantes da equipe um melhor acompanhamento do status dos projetos.

A partir da Figura 19, é possível verificar que as pessoas conseguem ter uma visão melhor do status do projeto, permitido pelo uso do *Scrum* e de suas ferramentas (*Sprint Burndown* e *Burndown Chart*). Antes, não havia qualquer maneira de saber quanto do projeto já estaria pronto, nem quanto tempo seria necessário para terminá-lo.

Também é fácil notar a partir da figura que o problema de perda de trabalho foi completamente evitado a partir do uso do Subversion. É importante entender que isto representa um grande avanço, uma vez que este problema acontecia com grande freqüência, causando desconforto entre os integrantes da equipe.

Por último, a utilização de *Scrum* com *Sprints* curtos (de uma semana) não foi suficiente para evitar que algo fosse desenvolvido sem que nunca fosse usado depois de pronto. Isto é, mesmo que as funcionalidades e tarefas fossem estabelecidas semanalmente, o problema de trabalho desperdiçado ainda ocorre, talvez porque ele dependa de muitos outros fatores (como maturidade da organização, por exemplo) e não somente do método de gerência utilizado.

8.6.4.Aumentar a popularidade do site

Nesta seção, estabeleceu-se a meta de aumentar em 5% a popularidade do site. Para realizar as medições, utilizaram-se a ferramenta Web Trends e a Google Analytics. O resultado obtido foi que a quantidade média de acessos (em torno de 600 mil acessos por mês) não teve um aumento significativo, ou seja, o aumento foi menor do que a meta estabelecida. Isto é, este objetivo não foi alcançado, uma vez que, em algumas comparações, o número de acessos foi até menor em um mês, quando comparado ao mesmo mês em um ano anterior.

Porém, em relação à encontrabilidade do site, foi possível verificar uma melhora. Isto é, quando eram realizadas buscas pelas ferramentas de busca da Web, era muito difícil encontrar o site da organização na primeira página de resultados encontrados. Contudo, após as atividades relacionadas ao SEO (todas as alterações que foram introduzidas seguindo as boas práticas recomendadas pela Google), é comum encontrar o site da organização na primeira página de resultados de uma busca pertinente, que não esteja buscando especificamente a organização como parâmetro de busca (principalmente quando esta busca contém algum conteúdo relacionado à América Latina).

Com a ajuda das ferramentas aqui citadas, verificou-se que 57% dos acessos ao site da organização são feitos de países da América Latina, 28% dos Estados Unidos e 11% de países europeus.

8.6.5.Aumentar a popularidade das mídias sociais e seus conteúdos relacionados

Um dos objetivos deste trabalho era o de aumentar a popularidade das mídias sociais utilizadas pela organização, estabelecendo-se a meta de aumentar em 5%.

Em relação ao Youtube e ao Twitter, a meta não foi atingida, uma vez que não houve mudança significativa na quantidade de pessoas inscritas nas respectivas páginas. Normalmente, quando um usuário assiste a um vídeo no Youtube, não necessariamente passará a assistir a todos os vídeos do canal. Talvez isto explique o fato de o número de pessoas inscritas no canal do Youtube não ter aumentado significativamente.

Em relação ao Twitter, a mudança também foi bastante pequena, talvez pelo fato de que a maior propaganda e grande parte do conteúdo estejam relacionadas à página da organização no Facebook. Desta maneira, não foi possível medir quantitativamente os aspectos relacionados ao acesso do Twitter e do Youtube – em ambos os casos, verificou-se que a média de seguidores se manteve constante.

Por último, verificou-se um grande crescimento dos usuários que se interessam pela organização no Facebook. A página foi criada em fevereiro de 2009, continha 4000 fãs em março de 2010 e passou a ter 6825 fãs em maio do mesmo ano. Há cinco meses, o número de usuários inscritos era de 1906, ou seja

um au- mento de mais de 200%. Porém, é muito difícil avaliar quanto que esse número cresceu somente pelo fato de o Facebook ter crescido ou se este crescimento de- pendeu de alguma maneira das melhoras realizadas (de criação de um módulo no gerenciador de conteúdo da página que permite a rápida atualização do conteúdo relacionado às mídias sociais).

8.7.Pontos Positivos

Apesar de não ter atingido todos os objetivos propostos, este trabalho apresenta uma série de benefícios para a unidade estudada. Houve uma melhoria significativa no processo de atualização do site, uma vez que esta se tornou uma atividade rápida e confiável, pois deixou de ser uma atividade completamente manual, sujeita a falhas.

O processo de atualização do site, que poderia durar até mesmo mais do que duas horas algumas vezes (quando uma funcionalidade muito grande fosse colocada em produção), agora não leva mais de dez minutos. Este processo também não adiciona tantas falhas como acontecia em um momento anterior, quando um conjunto de arquivos não era copiado ou sua cópia se destinava a um local errado. Com a diminuição drástica do número de erros que aparecia no site, ficou evidente a melhora conseguida.

Outro ponto importante foi a instauração de um sistema de controle de versão. Confiar somente em operações de backup é muito complicado quando se trata de um *website* de uma grande organização. Além do perigo de se perder arquivos importantes que se encontram em produção, tem-se o problema de perda de trabalho que acontecia constantemente: duas pessoas alteravam ao mesmo tempo um mesmo arquivo, causando problemas de sincronização: um trabalho acabava substituindo o anterior.

Portanto, a utilização de um sistema de controle de versão foi um passo importante para a estruturação de trabalho da unidade estudada. A criação de um gerenciamento de configurações e padrões de codificação também foram fatores que melhoraram bastante a qualidade do trabalho desenvolvido.

Por último, a utilização do *Scrum* permitiu um controle muito maior do projeto a ser desenvolvido, uma vez que com ele é possível acompanhar todo o status do projeto.

A utilização da ferramenta Banana Scrum foi de fundamental ajuda, uma vez que possibilitou a extração de relatórios e gráficos online, isto é, os gerentes da unidade conseguem acompanhar também todos os projetos utilizando *Scrum*, sem precisar se deslocar aos locais onde os *Sprints Burndowns* se encontravam fisicamente.

Além de os projetos ficarem mais organizados, o *Scrum* possibilitou um rápido desenvolvimento dos projetos em andamento, uma vez que, por ter *Sprints* semanais, as prioridades eram sempre reorganizadas e discutidas, fazendo com que o desperdício de trabalho não utilizado fosse menor. Isto é, quando se desenvolvia algo, aquela funcionalidade já estava bastante entendida por parte dos desenvolvedores e dificilmente não seria utilizada depois de pronta.

8.8.Pontos Negativos

Apesar de este problema de desperdício de trabalho ter diminuído com a utilização do *Scrum*, ele não foi completamente erradicado. Isto pode ser explicado pelo fato de que um método de gerenciamento de projetos por si só não é capaz de evitar este tipo de problema. Quando um requisito é desenvolvido e depois de pronto, não é mais desejado pelo usuário, muitas vezes, isto depende da vontade do cliente e do seu conhecimento do problema, ou seja, o cliente tem que estar muito seguro em relação ao que deseja.

Se ele não souber o que deseja, pode ocorrer de mudar de ideia constantemente, o que pode atrapalhar o andamento do projeto. O que ocorre muitas vezes é que o cliente não é capaz de imaginar os resultado e benefícios do software, por desconhecer como a tecnologia pode auxiliar o seu negócio. Deste modo, ao longo do desenvolvimento do projeto, quando o cliente começa a ganhar mais conhecimento em relação à tecnologia utilizada e ao negócio em si, ele passa a ter novas ideias, requerendo novas funcionalidades.

Assim, ao longo do desenvolvimento de um projeto em que o cliente tenha acesso aos incrementos de funcionalidade produzidos (utilizando o *Scrum*, por exemplo), ele consegue ter uma visão mais concreta de como a tecnologia pode auxiliá-lo, fazendo a requisição de novas funcionalidades nos estágios iniciais do desenvolvimento. Este comportamento pode ser mais facilmente tratado como de

baixo impacto, comparado com o impacto que estas mudanças teriam se fossem requeridas ao final do projeto.

Outro ponto negativo foi o fato de alguns objetivos não terem sido atingidos. Em relação ao aumento da popularidade do site e das ferramentas de mídia social, foi complicado medir as melhoras atingidas – porque não havia dados históricos suficientes e porque é difícil precisar o quanto as melhoras atingidas dependem deste trabalho ou apenas refletem fatores externos a ele.

Por último, pode-se perceber a aversão das pessoas a mudanças, o que torna as propostas de melhora ainda mais difíceis. Apesar de esta ser uma organização tradicional, as pessoas integrantes da unidade eram em sua maioria composta por jovens, que normalmente são mais adeptos a mudanças. Porém, verificou-se que é muito complicado convencer as pessoas a tentar algo novo, principalmente quando elas não possuem o embasamento teórico necessário, uma vez que a insegurança trazida pela mudança se torna maior neste caso.

8.9.Resumo

Neste capítulo foram apresentadas todas as mudanças realizadas a fim de atingir os objetivos estabelecidos. Viu-se que é muito importante o estabelecimento de métricas para avaliar o impacto das mudanças, uma vez que é necessário fazer medições para comprovar as melhoras obtidas. Porém, dependendo de alguns fatores (como a métrica utilizada e a disposição de dados históricos suficientes) pode ser complicado chegar a uma conclusão acurada. Normalmente, este é o caso das métricas subjetivas, aquelas que dependem da opinião de um grupo de pessoas. Muitas vezes se torna difícil chegar a alguma conclusão a partir destas.

As métricas objetivas são em geral mais interessantes, uma vez que conseguem gerar resultados quantitativos e muitas vezes podem ser automatizadas. Isto é, elas podem ser extraídas a partir de um processo automático, que não dependa da interferência manual ou da opinião de um desenvolvedor. Estas métricas são as que geram resultados mais acurados e devem ser utilizadas sempre que possível.

A partir deste trabalho, foi possível verificar o quanto é importante estruturar o ambiente de trabalho, criando algumas regras e utilizando um

conjunto de ferramentas que interagem entre si. É importante também utilizar algumas práticas da Engenharia de Software, aquelas que podem auxiliar o trabalho realizado (como por exemplo, a estruturação de uma gerência de configurações).

A realização de algumas mudanças (implantação de um sistema de controle de versão e de um controle de mudanças para a atualização dos arquivos nos servidores, além da utilização de um framework e de um padrão de codificação) foi suficiente para alcançar grandes melhoras no produto desenvolvido. Isto diminuiu o número de erros encontrados e facilitou o trabalho diário das pessoas envolvidas.

A utilização do *Scrum* também pode ser muito benéfica para o gerenciamento dos projetos, mesmo em organizações mais tradicionais. Com o seu uso, é mais fácil acompanhar o andamento do projeto, resolver os problemas encontrados durante o processo com mais rapidez e responder às mudanças surgidas, que muitas vezes podem ser constantes.

A utilização do software Banana Scrum também se mostrou muito eficiente no sentido que permite a geração de diferentes gráficos e relatórios, importantes para o acompanhamento do projeto, especialmente quando a equipe não se encontra muito próxima fisicamente.

Por último, é importante frisar que este conjunto de mudanças realizadas não representa um investimento financeiro alto, uma vez que as principais atividades de estruturação do ambiente foram feitas por uma só pessoa, utilizando ferramentas gratuitas, em sua maioria. Isto é, este trabalho representa uma implementação bem-sucedida e de baixo custo de *Scrum* e de boas práticas de Engenharia de Software, em um ambiente heterogêneo.