

6

Infraestrutura de Trabalho

Este capítulo tem como objetivo fornecer uma visão geral do ambiente de trabalho encontrado na organização estudada, bem como confrontá-lo com a organização ideal tal como vem sendo proposta na literatura relativa à adoção de *Scrum* e de boas práticas da Engenharia de Software. Deste modo, detalhes como formação das pessoas envolvidas na equipe, estrutura física e infraestrutura técnica são aqui citados.

Para se atingir a qualidade esperada de um software, é muito importante que haja disciplina ao longo do desenvolvimento do mesmo. A utilização de um método de gerência de projetos como o *Scrum* pode ajudar muito o processo de desenvolvimento. Porém, também é imprescindível que, juntamente com a adoção deste método ágil, tenha-se a preocupação de utilizar boas práticas da Engenharia de Software.

6.1. Gerência de Configuração

A Gerência de Configuração é uma área da engenharia de software muito importante para o desenvolvimento de um projeto. Ela é responsável pelo controle de versão, pelo controle de mudanças e pela auditoria das configurações. Desenvolvida pelo Departamento de Defesa dos Estados Unidos nos anos 50, seus conceitos e práticas tem sido largamente adotados por diferentes modelos de gerência, como o *Capability Maturity Model Integration* (CMMI), ISO9000 e COBIT. Para o ITIL (*Information Technology Infrastructure Library*), a gerência de configurações é o componente de serviço responsável por unir os outros processos do ITIL (<http://www.itil-officialsite.com/home/home.asp>).

Segundo Pressman [22], Gerência de Configuração é um “conjunto de atividades projetadas para controlar as mudanças pela identificação dos produtos do trabalho que serão alterados, estabelecendo um relacionamento entre eles, definindo o mecanismo para o gerenciamento de diferentes versões destes

produtos, controlando as mudanças impostas, e auditando e relatando as mudanças realizadas.”

Assim, a Gerência de Configuração tem como objetivo identificar, guardar, avaliar, acompanhar, coordenar, analisar e controlar os itens de configuração (CI) por meio de atividades que mantêm a integridade dos itens durante o ciclo de vida do projeto. Isto é, ela é responsável por manter as versões destes itens, seus componentes e as relações entre eles.

Os itens de configuração são, portanto, qualquer componente da estrutura de desenvolvimento que é controlado e monitorado pela Gerência de Configuração. Entre eles, pode-se incluir partes do código, documentos interessantes para a manutenção do software, scripts de banco de dados, entre outros. É muito importante que se faça uma escolha correta dos itens a serem gerenciados, uma vez que simplesmente transformar todo e qualquer artefato em um CI pode gerar despesas e trabalho desnecessários.

No ITIL, há um conceito muito importante relacionado à Gerência de Configuração: a base de dados da Gerência de Configuração (CMDB). Este nada mais é do que um repositório de informações relacionadas aos componentes que fazem parte do ciclo de vida de um projeto, devendo possuir detalhes importantes dos atributos dos CIs, bem como as relações entre eles.

Segundo [25], há três regras que devem ser seguidas na implantação de uma Gerência de Configuração:

1. controle contínuo,
2. usar tecnologia e
3. integrar os processos.

Controle contínuo no sentido de que os itens de configuração devem ser continuamente inspecionados, bem como as suas modificações. Se em algum ponto o controle for perdido, o processo se torna inútil.

Gerência de Configuração manual pode ser perda de tempo, uma vez que se torna extremamente difícil. É necessário utilizar a tecnologia para ajudar a descobrir, guardar e manter as informações relativas aos CIs. Isto é, deve-se utilizar ferramentas que auxiliem este trabalho, senão o processo não funciona.

Por último, é importante integrar a Gerência de Configuração com outros processos, como atividades de mudanças, gerenciamento de incidentes e problemas e atividades de lançamento de versão. A Gerência de Configuração é

um processo que requer gasto de recursos, tanto de tempo quanto financeiros, que precisam desta integração para obter os retornos necessários.

Portanto, apesar de requerer uma quantidade de recursos, a Gerência de Configuração é fundamental para o desenvolvimento de um projeto – até mesmo aqueles projetos pequenos, que suportam algumas poucas aplicações de complexidade pequena. Em um estudo, notou-se que 60% dos serviços de impacto durante o ciclo de vida de um projeto são relacionados a problemas de configuração [25].

Portanto, além da frustração pessoal (sentido pelos membros de uma equipe) de ter que resolver incidentes causados por falta de uma gerência de configuração adequada, há também sérios impactos de negócio que podem resultar de uma gerência de configuração insuficiente, causando perdas significativas.

Na organização estudada não é utilizado qualquer método de gerência de configuração. As mudanças são feitas nos arquivos e replicadas para os servidores manualmente. É difícil saber qual a versão disponível em cada servidor e muitas vezes os diferentes servidores apresentam versões distintas do *site*, quando deveriam estar sincronizados.

Portanto, é necessário o estabelecimento de uma gerência de configuração que responda a perguntas como “O que mudou, por que, por quem e quando?” e que defina procedimentos de alteração dos arquivos nos servidores, guardando histórico e organizando as diferentes versões do site produzidas.

6.1.1. Controle de Versão

Uma atividade fundamental para uma correta gerência de configuração é o controle de versão. Na organização estudada não há qualquer ferramenta de controle de versão, portanto, é necessário confiar nas atividades de *back-up* e na organização da equipe. Assim, quando um arquivo é alterado ou uma série de arquivos é modificada, é necessário atualizar os quatro ambientes de trabalho. Isto é importante para que os servidores estejam sempre sincronizados e para testar as alterações em diferentes servidores antes de fazer qualquer modificação na página externa.

Contudo, não há uma ferramenta para automatizar este processo, tornando-o muito trabalhoso e sujeito a falhas, pois muitas vezes as pessoas responsáveis pelas alterações não se lembram quais foram os arquivos alterados.

Como não há um controle de versão, que exiba as últimas alterações a partir de um evento específico (como a última atualização do site), é difícil saber quais os arquivos precisam ser copiados para os demais servidores. Outro problema é a falta de histórico e difícil restauração de versões antigas do *site*.

6.1.2. Boas Práticas

A Gerência de Configuração (CM) estabelece algumas boas práticas que devem ser seguidas durante o desenvolvimento de software. Segundo Laura Wingered [31], há seis principais áreas de CM, dentre elas: ambiente de trabalho, código principal, variantes de código (*branches*), propagação de mudanças, *builds* e processos.

Em relação ao ambiente de trabalho, este não pode ser compartilhado. Isto é, deve ser usado um controle de código, porém cada desenvolvedor deve possuir sua própria cópia de arquivos, onde fará suas alterações. Também é imprescindível que ele esteja sempre sincronizando o seu trabalho com o repositório, para que não possua uma versão muito desatualizada.

O código principal deve possuir padrões de programação e também regras para criação de variantes – os *branches*. Em relação aos *branches*, estes devem ser evitados ao máximo. Porém, quando forem extremamente necessários, devem ser feitos o mais tarde possível dentro do ciclo de desenvolvimento, para minimizar os *merges* posteriores.

A propagação de mudanças deve ser feita constantemente. Isto é, os desenvolvedores devem ter o cuidado de sempre possuírem uma versão em suas máquinas bastante próxima daquelas encontradas no repositório, para minimizar a quantidade de *merges*. Também é importante escolher a pessoa certa na hora de fazer *merge* – normalmente aquela que realizou as mudanças nos arquivos.

Os *builds* devem ser feitos com o auxílio de ferramentas, para automatizar o processo. É muito importante a utilização de scripts para a realização de *builds* automáticos. Assim, eles podem ser feitos constantemente, por ferramentas de integração contínua, que guardam os *logs* e os produtos resultantes do *build*.

O processo em si deve acompanhar os pacotes de mudança e também as propagações dos mesmos. É importante também diferenciar o que foi pedido do que foi realmente realizado.

Por exemplo, na implementação de uma nova funcionalidade pode surgir uma falha no sistema. Assim, o que foi pedido é representado pela nova funcionalidade, porém a falha representa o que foi realmente feito. Deste modo, pode-se relacionar estes dois grupos na tentativa de minimizar a ocorrência deste mesmo erro novamente ou no momento de descoberta de novos problemas e busca por soluções.

Também em relação ao processo, é muito importante que cada item de configuração possua um dono, responsável pela sua manutenção. Assim, caso algum CI não esteja no padrão adequado, esta pessoa será responsável por buscar a informação correta. Por último, o processo deve utilizar documentos atualizados e de fácil acesso. Documentos de difícil acesso e desatualizados são inúteis, uma vez que não serão consultados pela equipe [31].

6.2. Gerência de Recursos

Recursos no contexto deste trabalho são representados por todo o conjunto de ativos que a organização possui, tais como equipamentos, estrutura física das salas onde a equipe trabalha e os próprios membros da mesma. Em relação aos integrantes da equipe, é importante fornecer detalhes como a formação dos mesmos e seus papéis desempenhados na equipe.

6.2.1. Estrutura da Equipe

A equipe estudada é formada por um coordenador, um gerente de projetos, *web designers*, editores e desenvolvedores. Além de assumir diversos papéis, a equipe é muito diversificada, uma vez que a formação acadêmica e experiência profissional de seus integrantes são altamente variadas.

Os editores muitas vezes possuem formação relacionada à comunicação e jornalismo. Entretanto, eles não têm conhecimento de desenvolvimento *web* e somente alteram o conteúdo das páginas.

Os desenvolvedores têm formação em computação e são responsáveis pela parte dinâmica do site e também pelas ferramentas que auxiliam a manutenção do

conteúdo do mesmo. O coordenador e o gerente são os responsáveis por liderar a equipe e representar o interesse dos clientes, que neste caso, são os usuários externos do site (podendo ser pessoas que trabalham na organização ou pessoas de fora que acessam o site).

O *web designer* é responsável pela aparência da página, criando as figuras e toda a combinação de cores e estilo das mesmas. Há um padrão de estilos a ser seguido que não pode ser negligenciado, portanto o *web designer* deve levar isto em consideração. Porém, a cada novo conjunto de páginas a ser criado, mesmo que o estilo em geral do site esteja definido, é necessário criar um estilo único para este novo conjunto, composto por cores e figuras diferentes.

Por último, vale também ressaltar que a equipe é formada por pessoas de diferentes países – portanto diferentes culturas, possuindo como idioma nativo inglês, português, espanhol ou francês. Este fator pode gerar algum tipo de falha na comunicação.

Apesar de a equipe ser grande, dificilmente todos os membros trabalham em um mesmo projeto. No desenvolvimento de um projeto, são escolhidos apenas alguns componentes desta equipe, geralmente pessoas que representam papéis diferentes; por exemplo, em um projeto X tem-se um desenvolvedor, um *web designer* e três editores. Ou seja, os times formados são multidisciplinares, o que é bastante interessante quando se utiliza *Scrum* [18]. Também são times pequenos, o que facilita o gerenciamento e a comunicação entre os seus integrantes – outro fator positivo em relação à adoção de *Scrum*.

Um fator negativo em relação à equipe é que em sua grande maioria, ela é formada por consultores, que por questões contratuais não podem trabalhar longos períodos seguidos para a organização. Eles precisam fazer pausas constantes e muitas vezes mudam de emprego, causando uma alta rotatividade.

A área de desenvolvimento de software já é conhecida por apresentar uma alta rotatividade de pessoas trabalhando, que trocam de empresa em intervalos pequenos de tempo (menos de um ano, por exemplo). Porém, com esta limitação no contrato, talvez esta rotatividade seja ainda maior do que a já encontrada na área de desenvolvimento de software, o que é desfavorável para a gestão do conhecimento.

6.2.2.Organização Técnica

A equipe de desenvolvedores cresceu recentemente e hoje é composta por quatro pessoas. Um ano atrás, apenas uma pessoa era responsável pela adição das funcionalidades ao site. Este cenário talvez explique a ausência de padrões de programação e a falta de utilização de um *framework* de desenvolvimento.

Entre os artefatos desenvolvidos, é muito comum encontrar código sem padrões de programação e que misture acesso a banco e apresentação visual em um mesmo módulo. Há apenas padrões de apresentação do site, isto é, regras de CSS que devem ser seguidas para assegurar uma aparência uniforme para o mesmo.

Assim, no código legado, pode se encontrar em um mesmo arquivo acesso a banco, controle visual, fragmentos de CSS e JavaScript. Este tipo de organização é altamente desencorajado, uma vez que dificulta a manutenção do site, além de ser contra os padrões da W3C – que defendem a separação entre estrutura, apresentação e funcionalidade.

Deste modo, na infraestrutura de desenvolvimento, é desejável que se utilize um modelo *Model-View-Contoler* (MVC), que separe o acesso a banco da camada de apresentação, por exemplo. Também é importante que o código HTML seja separado dos arquivos de script (.js) e de estilo (.css). Para alcançar este objetivo, um padrão de codificação foi desenvolvido, além da adição de um *framework* para ColdFusion, com a finalidade de facilitar a organização do código – o Coldbox (<http://www.coldbox.org/>).

6.2.3.Estrutura Física

A equipe estudada dispõe de quatro ambientes de trabalho, configurados em servidores distintos. Dois destes ambientes só funcionam internamente na organização e os outros dois são acessíveis via *web*. Toda a infraestrutura de tecnologia (como banco de dados, servidores, entre outros) é controlada pelo departamento de Tecnologia da Informação da organização, distinto do estudado. Porém, a equipe tem autonomia para requerer mudanças nestes servidores, o que é fundamental para a realização deste estudo.

Em relação à estrutura física do local de trabalho, toda a equipe se encontra no mesmo andar do prédio, porém cada pessoa tem a sua própria sala (isolada por

paredes e portas), o que acaba criando um ambiente muito disperso, desfavorável à troca de opiniões e experiências. Um dos pontos importantes que o *Scrum* ressalta é a fácil comunicação e troca de idéias, defendendo que as pessoas de uma equipe devem sentar-se próximas uma das outras para que possam conversar face-a-face nos momentos que surgem dúvidas. Ou seja, esta boa prática não está sendo coberta pela organização, uma vez que as pessoas estão fisicamente isoladas.

Apesar deste isolamento físico, normalmente todos os integrantes da unidade tem uma idéia dos projetos que estão sendo desenvolvidos pela mesma e quem é responsável por cada um deles. Isto é possível porque uma vez por semana, no mesmo horário e mesmo local, todos os integrantes da unidade se reúnem para acompanhar o andamento dos projetos e estabelecer as prioridades. Esta reunião é importante para que as várias equipes tenham uma idéia geral dos projetos sendo executados. Porém, estas não se confundem com as reuniões do *Scrum* (como a reunião de planejamento e o *Daily Scrum*, realizadas apenas entre os membros de cada time, sendo aquelas uma vez por semana e estas, diariamente).

A reunião semanal com todos os integrantes da unidade é bastante produtiva, uma vez que é importante para a equipe tomar conhecimento dos projetos desenvolvidos e das responsabilidades de cada membro da mesma. Porém, ao longo da semana, muitas outras reuniões ocorrem e algumas duram por longos períodos de tempo, muitas vezes sem agregar muito ao trabalho da equipe.

Por último, em relação aos equipamentos, os computadores utilizados são Pentium 4 com Windows XP, que se tornam um pouco lentos ao executarem programas pesados, como o Dreamweaver e Eclipse, uma vez que a maioria das máquinas possui somente 1 GB de memória RAM. Esta definitivamente é uma limitação que precisava ser trabalhada.

6.2.4.Ferramentas Utilizadas

Um dos principais objetivos deste estudo é melhorar a infraestrutura de trabalho da equipe estudada, para que sua produtividade seja aumentada. De um lado, deseja-se estabelecer o uso de uma ferramenta para gerenciamento dos projetos, uma vez que a ferramenta utilizada para gerenciar os projetos em

andamento era um arquivo Excel, que muitas vezes se encontrava desatualizado com o corre-corre diário.

Somado a isto, desejava-se melhorar a infraestrutura das ferramentas que a equipe utilizava para realizar o seu trabalho diário, automatizando ao máximo tarefas repetitivas e sujeitas a falhas. A principal ferramenta utilizada para atualizar os arquivos do site era o Dreamweaver CS3. Esta ferramenta, como citado no capítulo anterior, não dá suporte ao controle de versão. Os arquivos alterados eram diretamente atualizados em um servidor interno, utilizado por toda a equipe como principal servidor de desenvolvimento.

Assim, era muito comum duas pessoas alterarem o mesmo arquivo simultaneamente, ocasionando algum tipo de problema, como perda de trabalho. Outra ocorrência indesejável era a alteração de vários arquivos, que acabava levando o desenvolvedor ao esquecimento. Isto é, a pessoa não se lembrava da lista completa de arquivos que alterou para desenvolver uma nova funcionalidade.

Portanto, algumas vezes se tornava difícil a migração de uma nova funcionalidade para os outros servidores. Com a utilização de um controle de versão, este problema poderia ser minimizado, uma vez que a pessoa teria um histórico de alteração dos arquivos, podendo verificar quais arquivos foram alterados e precisavam ser migrados.

Outras ferramentas também utilizadas pela equipe em sua rotina diária de trabalho eram aquelas para edição gráfica, usadas pelo *web designer*, ferramentas de acesso e manipulação do banco de dados e ferramentas para transferência FTP, utilizadas pelos desenvolvedores. Não era utilizado qualquer tipo de teste automático pelos mesmos.

6.3. Resumo

Este capítulo descreveu a situação de trabalho da equipe estudada antes das modificações propostas, em relação à estrutura física da unidade, ao ambiente de trabalho diário (ferramentas e processos utilizados) e à estrutura da equipe de trabalho. Dois pontos são fundamentais neste capítulo: melhorar a gerência de projetos em andamento e melhorar o ambiente de trabalho para que a equipe possa trabalhar com uma infraestrutura mais produtiva.

Em relação à gerência dos projetos, não havia qualquer metodologia de gerência e acompanhamento dos projetos em execução, o que dificultava o

trabalho dos gerentes e coordenadores. O capítulo seguinte lista as mudanças realizadas ao longo deste estudo com o objetivo de melhorar a qualidade de trabalho desenvolvido pela equipe.