PLANO GERAL DE GERÊNCIA DE CONFIGURAÇÃO

Versão 2.6

Responsável: José David Fumo
Gestor de Configuração
josedavidfumo@gmail.com

Erick Paulo Samuel Mahanjane
Gestor de Projectos
erick.mahanjane@gmail.com

Luthermilla Mucula dos Reis Ecole
Gestora de Tecnologia
luthermilla@gmail.com

UEM

ÍNDICE

Tabela de Figuras

1. Introdução

1.1. Propósito

O propósito do Plano de Gerenciamento de Configuração é descrever como a gerência de configuração (GCO) será realizada durante todo o ciclo de vida do projeto. Isto inclui a documentação de como a GCO é gerenciada, os papéis e responsabilidades das pessoas envolvidas, como as mudanças são feitas no item de configuração (IC), e como a comunicação de todos os aspectos da GC são realizadas entre os participantes do projeto.

Sem um plano de gerencia de configuração documentado é provável que um IC seja perdido ou trabalho desnecessário seja feito por falta de versão e controle de documentos. Um plano de gerenciamento de configuração é importante para todos os projetos, especialmente para projetos de tecnologia da informação (TI).

1.2. Definições, Siglas e Abreviações

Nesta seção são descritas as definições, siglas e abreviações utilizadas neste documento e em comunicações diversas nos projetos.

- **IC:** Item de configuração;
- **GC:** Gerência de Configuração;
- **CCC:** Comitê de Controle de Configuração;
- **GP:** Gerente de Projetos;
- **GI:** Gerente de Infraestrutura;
- **Checkout:** Baixar do repositório inteiramente o projeto ou uma pasta dele;
- **Commit**: Enviar para o repositório os arquivos locais modificados;
- **Update:** Atualizar (sincronizar) localmente arquivos específicos ou pastas;

2. Gerência de Configuração de Software

2.1. Organização, Responsabilidades, e Interfaces

2.1.1. Comitê de Controle de Configuração (CCC)

O comitê de controle de configuração (CCC), do inglês *Configuration Control Board*, é composto por patrocinador, gerentes de projetos (GP), gerentes de configuração, gerentes de infraestrutura (GI), e líderes de equipe. Suas responsabilidades são:

- Definir as diretrizes da GC;
- Definir os ICs padrão a serem considerados em todos os projetos;

- Analisar e aprovar / rejeitar solicitações de mudança de configuração;
- Assegurar que todas as mudanças aprovadas estão sendo aplicadas no plano de GC;
- Definir quando são realizadas auditorias de GC nos projetos.

2.1.2. Patrocinador do Projeto

O patrocinador do projeto é quem financia o projeto. Em projetos externos geralmente ele é o cliente e em internos é o diretor da área que solicitou o projeto (Infra, Operação, Administrativo, etc).

- Participar da reunião do CCC;
- Aprovar qualquer assunto que requer escopo, tempo ou custo adicional aos projetos.

2.1.3. Gerente de Projetos

Suas responsabilidades são:

- Responsabilidade geral por todas as atividades de GC do projeto;
- Comunicar quaisquer mudanças aprovadas no plano de GC;
- Participar das reuniões do CCC;
- Fechar nova linha-base caso haja mudanças no escopo ou prazo;
- Solicitar configuração de acesso dos usuários do projeto no repositório.

2.1.4. Analista de Suporte

Suas responsabilidades são:

- Dimensionar os recursos necessários ao bom funcionamento da GC;
- Garantir a disponibilidade e desempenho dos recursos necessários;
- Executar a configuração de permissões solicitada pelo gerente de projeto.

2.1.5. Gerente de Configurações

Suas responsabilidades são:

- Reportar solicitações de mudança do processo de GC ao CCC;
- Identificar novos ICs com apoio dos desenvolvedores;
- Solicitar qualquer necessidade de treinamento em GC;
- Criar as BRANCHES, tags e realizar os merges.

2.1.6. Líder de Equipe

Suas responsabilidades são:

- Reportar ao gerente de configurações qualquer mudança de GC descoberta durante a fase de execução do projeto;
- Identificar ICs candidatos e comunicar ao gerente de configuração;
- Participar das reuniões do CCC.

2.1.7. Analista de Qualidade

- Assegurar que a equipe de desenvolvimento está seguindo os padrões e métodos de GC definidos pelo CCC;
- Orientar os desenvolvedores quanto as boas práticas de GC;
- Fornecer padrões e templates de configuração as equipes de projeto;
- Fornecer qualquer necessidade de treinamento em GC;
- Participar das reuniões do CCC.

2.1.8. Desenvolvedores

Suas responsabilidades são:

- Identificar ICs candidatos e comunicar ao Líder de equipe;
- Reportar ao Líder de equipe qualquer mudança de GC descoberta durante a fase de execução do projeto;
- Seguir todos os padrões e métodos de GC na implementação do projeto.

2.2. Ferramentas, Ambientes e Infraestrutura

2.2.1. Ferramentas Utilizadas

O controle de versões de software será feito através do servidor Subversion (SVN) mais o cliente TortoiseSVN.

O Subversion (http://subversion.apache.org/) é um servidor CVS (Controle de Versão de Software) que centraliza o repositório de arquivos. O histórico de todas as modificações fica armazenado nele e podem ser acessados através de uma ferramenta cliente compatível.

A ferramenta cliente padrão definida para acessar o repositório é o TortoiseSVN (http://tortoisesvn.net/). Ele permite a comunicação da estação de trabalho com o servidor SVN. Seu principal papel é sincronizar o repositório remoto com a máquina local. Dentro outras funções ele da acesso a consultas, atualizações, inclusões e remoções, comparações entre versões, realizações de merge, entre outras.

2.3. Endereço do Repositório

O repositório do Subversion estará disponível internamente e também por acesso externo. O servidor se encontra nas dependências da Matriz. Segue abaixo os endereços:

- << Endereço do Repositório >>

3. Programação de Gerenciamento de Configuração

3.1. Identificação da Configuração

3.1.1. Itens de Configuração

Commented [S1]: Deve ser revisado

Os itens de configuração (IC) são os vários tipos de arquivos que devem ser incluídos no repositório para ter seu histórico de mudanças controlado.
Tanto os documentos como os arquivos-fonte que compõem um produto de software são Itens de Configuração (IC), assim como também o são as ferramentas de software necessárias para o desenvolvimento.
Nem todos os arquivos precisam ser controlados, por exemplo, um executável gerado pelo código fonte não deve estar no repositório, visto que sua fonte geradora já está sendo controlada no repositório.

Abaixo segue a lista de ICs que devem ser considerados em todos os projetos.

Item de Configuração (IC)	Descrição	Responsável	Quando é incluído?

No caso de um projeto ter como produto um novo sistema, ou seja, sem versões anteriores, um plano específico de gerência de configuração deverá ser criado para definir quais itens de configuração daquele sistema serão considerados na gerencia de configuração.

É responsabilidade da equipe do projeto identificar os ICs e incluí-los no repositório.

3.1.2. Padrão de nomeação dos arquivos e pastas

Nesta seção é descrito o formato padrão de nomeação dos arquivos e pastas a ser seguido no projeto.

Item	Padrão de Nome	Local Padrão de Armazenamento no Repositório
Plano Geral de Gerência de Configuração	**Formato:** Plano Geral de Gerencia de Configuração.doc	
Plano de Gerência de Configuração para Banco de Dados	**Formato:** Plano de Gerencia de Configuracao para Banco de Dados.doc	
Plano de Gerência de Configuração do Projeto	**Formato:** <nome do sistema ou projeto>_Plano_Gerencia_Configuração.doc **Exemplo:** Nome_Projeto_Plano_Gerencia_Configuração.doc	/<CLIENTE-X>/<sistema-x>/doc/ra<xxx>/pcg
Prospecção		
Proposta Comercial	**Formato**: PCM_<nome do cliente>_<nome do sistema ou projeto>_RA<contador 3 algarismos>.doc **Exemplo**:	
Análise		
Planejamento		

Desenvolvimento		
Fechamento		

3.1.2.1. Subprojetos

Alguns novos projetos surgem por necessidade de se adequar projetos anteriores. Estes projetos se caracterizam por envolverem modificações nos requisitos iniciais tratados pelo projeto “pai” e estas necessidades são descobertas geralmente quando o projeto pai está em homologação ou logo após ter entrado em produção.

Os motivos desta necessidade são geralmente dois:

1. ***Impactos causados por outros projetos*:** uma modificação no projeto x impacta o requisito do projeto y;
2. ***Novas necessidades descobertas*:** durante a homologação, ou em produção, descobre-se que o requisito x deveria tratar uma situação y que não foi considerada.

Nestes casos, a codificação do projeto deverá refletir o vínculo com o projeto anterior para que os envolvidos percebam de imediato este vinculado ao ler o código do projeto. Para isto um subprojeto deverá ser numerado com um segundo conjunto de numeração iniciado do 1.

3.1.3. Estrutura de pastas padrão do projeto

Esta seção descreve como a estrutura de pastas do projeto deve estar organizada.

A organização da estrutura foi planejada pelo CCC e foi levado em conta necessidades e problemas de GC como merge, fechamento de linhas base, arquivamento de BRANCHES, entre outros. O local de cada pasta tem um porque de estar lá e por isto reforçamos a importância de seguir esta estrutura para obter sucesso na GC.

A estrutura completa de pastas padrão de um projeto foi modelada e é ilustrada na **Error! Reference source not found.**.

<< Definir Estrutura >>

O arquivo fonte da modelagem se encontra no repositório de versões, no endereço <<Endereço >>

3.1.3.1. Principais pastas do projeto

Abaixo é descrito o papel das principais pastas do projeto.

- **TRUNK:** A TRUNK representa a linha de desenvolvimento mais nova em produção. Ela será utilizada para criação de novas BRANCHES de correção de bugs (*hotfixes*) e novas BRANCHES de desenvolvimento (novos projetos);
- **BRANCH:** A BRANCH representa uma nova linha de desenvolvimento, um desenvolvimento de melhoria a ser implantado no futuro, ou uma correção (*bugfix*) em produção. Elas representam um projeto em execução, depois de finalizada ela será fechada;
- **TAG:** A tag representa uma versão estável que foi implantada em produção. Tags são imutáveis, são uma foto no tempo de uma versão que foi homologada ou correção de bug em produção;
- **DOC:** A pasta doc reúne toda a documentação do projeto, como documentos de requisitos, casos de uso, cronogramas, planejamentos, aprovações, etc.

3.1.3.2. Ciclo de Vida das BRANCHES

Como dito anteriormente, uma BRANCH representa um desenvolvimento em andamento, tanto um novo desenvolvimento, quanto uma correção. Após o final do desenvolvimento, a BRANCH deve ser colocada em ambiente de homologação para a validação do cliente (solicitante do desenvolvimento).

Após sua validação, deverá ser realizado o merge da BRANCH para a TRUNK. Neste momento, esta modificação da TRUNK deverá ser replicada em todas as BRANCHES ativas (em desenvolvimento e/ou homologação).

No final do processo esta BRANCH deverá ser desativada (excluída).

3.1.3.3. Desenvolvimento de novas versões

Novas versões de sistemas deverão ser desenvolvidas em novas BRANCHES. Para isto, antes do início do desenvolvimento do projeto uma BRANCH deverá ser criada ou a partir da TRUNK ou de um outra BRANCH que corresponda a necessidade do projeto. As permissões de acesso da equipe na nova BRANCH deverão ser configuradas.

A nomenclatura para a BRANCH de desenvolvimento será:

ra<xxx>

Onde <*xxx*> é o número da versão do RA em desenvolvimento.

3.1.3.4. Correção de bugs em produção

Correções de bugs em produção (*bugfix*) deverão ser realizadas em BRANCHES específicas para este fim.

A BRANCH deverá ser criada dentro da pasta "*branches/manutencao*", pois a mesma já terá as permissões configuradas para a equipe responsável pela manutenção. Com isto,

A nomenclatura para a BRANCH de manutenção será:

mnt<yyyymmdd>_redmine#<identificador_redmine>

Onde:

- <yyyymmdd> é a data da criação da BRANCH. Ela é propositalmente no formato americano, pois permite uma melhor ordenação e visualização no diretório;
- <identificador_redmine> é o nome do projeto, seguido do tipo e o número da BRANCH aberta no Redmine. Exemplos:
  - mnt20131125_redmine#12345
  - mnt20131125_redmine#13579
  - mnt20131125_redmine #11223

3.1.3.5. Uso da TRUNK

Modificações na TRUNK somente serão permitidas quando BRANCHES forem homologadas pelo cliente, assim um merge trazendo tais modificações será permitido. Apenas o gerente de configurações terá permissões de realizar modificações na TRUNK.

A TRUNK é apenas o ponto focal para se criar novas BRANCHES. Caso bugs tenham sido gerados devido ao merge, quando uma cópia dele for criada para um novo desenvolvimento, como ela passará pelas fases de desenvolvimento, testes e homologação, este bug provavelmente será descoberto e sanado. No pior dos casos ele terá sido homologado pelo cliente.

3.1.3.6. Uso da TAG

Para evitar transferência de grande volume de dados entre servidor e estação de trabalho, não será adotado a forma convencional de tratamento de TAGs no SVN. Em vez disso, será usado um arquivo texto onde cada commit representará uma TAG.

O arquivo deverá ser nomeado como TAG.txt dentro da pasta “tags”. Os comentários commitados no arquivo serão o número da revisão correspondente na TRUNK, data do commit e o motivo da modificação (o nome do projeto ou o motivo da correção do bug), ou seja, não precisará escrever dentro do arquivo, apenas no comentário do commit. O acesso será restrito ao gerente de configuração.

O formato do comentário deverá ser:

“REVISAO:<<numero 6 dígitos>>; DATA:<<DD/MM/AAAA>>; MOTIVO:<<descrição do motivo>>”

Exemplo:

“REVISAO:033685; DATA:23/03/2013; MOTIVO:Homologação do projeto SANF RA50”

3.1.3.7. Tratamento da Pasta Documentação

Novas manutenções evolutivas terão sua documentação mantida em nova pasta seguindo o padrão:

`/<CLIENTE-X>/<sistema-x>/doc/<raxxx>/`

Dentro desta pasta teremos as subpastas para documentos de requisitos, cronograma, planos, modelagens, entre outros, conforme ilustrado na **Error! Reference source not found.**.

3.1.4. Permissões de acesso

O acesso às pastas do SVN será restrito, isto é, somente aqueles envolvidos no projeto terão acesso ao mesmo.

Todos os usuários do SVN pertencente ao quadro de colaboradores da NOME EMPRESA terão acesso de leitura a todos os projetos existentes e todas as suas subpastas. Somente não terão acesso às pastas referentes a processos não relacionados a desenvolvimento, como a pasta do administrativo, por exemplo.

Usuários do SVN que não pertencem ao quadro de colaboradores da NOME EMPRESA, cliente e fornecedores, por exemplo, somente terão acesso de leitura ou escrita aos projetos que estão envolvidos. Ou seja, nos demais projetos eles não poderão nem visualizar a pasta.

Abaixo são listados os perfis que

- TAG - somente o Gerente de Configuração terá permissão de escrita, pois será de responsabilidade dele atualizar o arquivo que contém as informações das TAGs;
- BRANCH DE DESENVOLVIMENTO – Os desenvolvedores/testers somente terão permissão de escrita na BRANCH do projeto que está envolvido e durante o período de execução deste. Após concluído o projeto, suas permissões de escrita deverão ser removidas. Isto irá evitar *commits* indevidos em projetos que não àqueles que o desenvolvedor está alocado;
- BRANCH DE MANUTENÇÃO – a pasta "*branch/manutencao*" terá configurada as permissões da equipe de manutenção responsável do Sistema em questão. As BRANCHs criadas dentro desta pasta irão herdar as permissões e com isto não será necessário a definição da mesma em cada criação.
- TRUNK – somente o Gerente de Configuração terá permissão de escrita, para que o mesmo tenha o controle do que irá ou não para o ambiente de produção.

O Gerente de Projetos será responsável por conceder / aprovar permissões de acesso dos usuários nos projetos e o analista de suporte será responsável por executar a configuração no sistema de controle de versão.

3.1.5. Baseline

O trabalho a ser feito no projeto deverá se basear na baseline de escopo, prazo, custo e qualidade, bem como atender somente a mudanças aprovadas pelo comitê de controle de mudanças (CCM). Para cada mudança aprovada, uma nova baseline deve ser fechada.

A tabela a seguir descreve em quais pontos do projeto a baseline será estabelecida, quem as autoriza e o que vai nela.

Observação: *Caso o escopo especificado leve mais de 1 mês para ser aprovada pelo cliente, nova análise de impacto deve ser refeita e assim nova linha base de todos os documentos da fase de prospecção devem* ser fechadas.

Fase	Arquivo	Responsável	Formato e Local
Análise	**Obrigatórios**		
			Formato: <nome do arquivo origem>-baseline-<yyyymmdd>.<extensão> **Local:** Na mesma pasta do documento que originou a baseline
			Formato: <nome do arquivo origem>-baseline-<yyyymmdd>.<extensão> **Local:** Na mesma pasta do documento que originou a baseline
Planejamento			
			Formato: <nome do arquivo origem>-baseline-<yyyymmdd>.<extensão> **Local:** Na mesma pasta do documento que originou a baseline
Desenvolvimento			
			Formato: <nome do arquivo origem>-baseline-<yyyymmdd>.pdf **Local:** Na mesma pasta dos documentos UAT
			Deve ser criado conforme descrito na seção 3.1.3.6.
			Formato: <nome do arquivo origem>-baseline-<yyyymmdd>.pdf **Local:** Na mesma pasta do documento que originou a baseline
			Formato: <nome do arquivo origem>-baseline-<yyyymmdd>.<extensão> **Local:** Na mesma pasta do documento que originou a baseline

3.2. Gerência de Configuração de Banco de Dados

A Gerência de Configuração para Banco de Dados é tratado em um documento separado que pode ser encontrado na mesma pasta deste documento no repositório de versões (//192.168.254.69/repnovo/SPI/processos/Gerencia-Configuracao/Plano de Gerencia de Configuracao para Banco de Dados.doc).

3.3. Controle de Configurações e Mudanças

3.3.1. Solicitação de Mudança e Aprovação

Descreve o processo pelo qual problemas e mudanças são submetidos, revisados e disponibilizados pelo comitê de Controle de Mudanças (CCM).

Solicitações de mudança podem ocorrer a qualquer momento no projeto (iniciação, planejamento, execução ou fechamento). Quanto mais tarde no projeto ocorrer, mais rigido e exigente deverá ser o processo de aprovação. Solicitações de mudança podem ser quanto ao escopo, cronograma, custos ou qualidade.

As solicitações de mudanças podem vir do cliente ou internamente. Independente de onde vem, todas devem ser tratadas pelo (CCM).

A seguir é ilustrado o fluxo de aprovação de solicitações de mudança vindas da equipe interna ou do cliente.

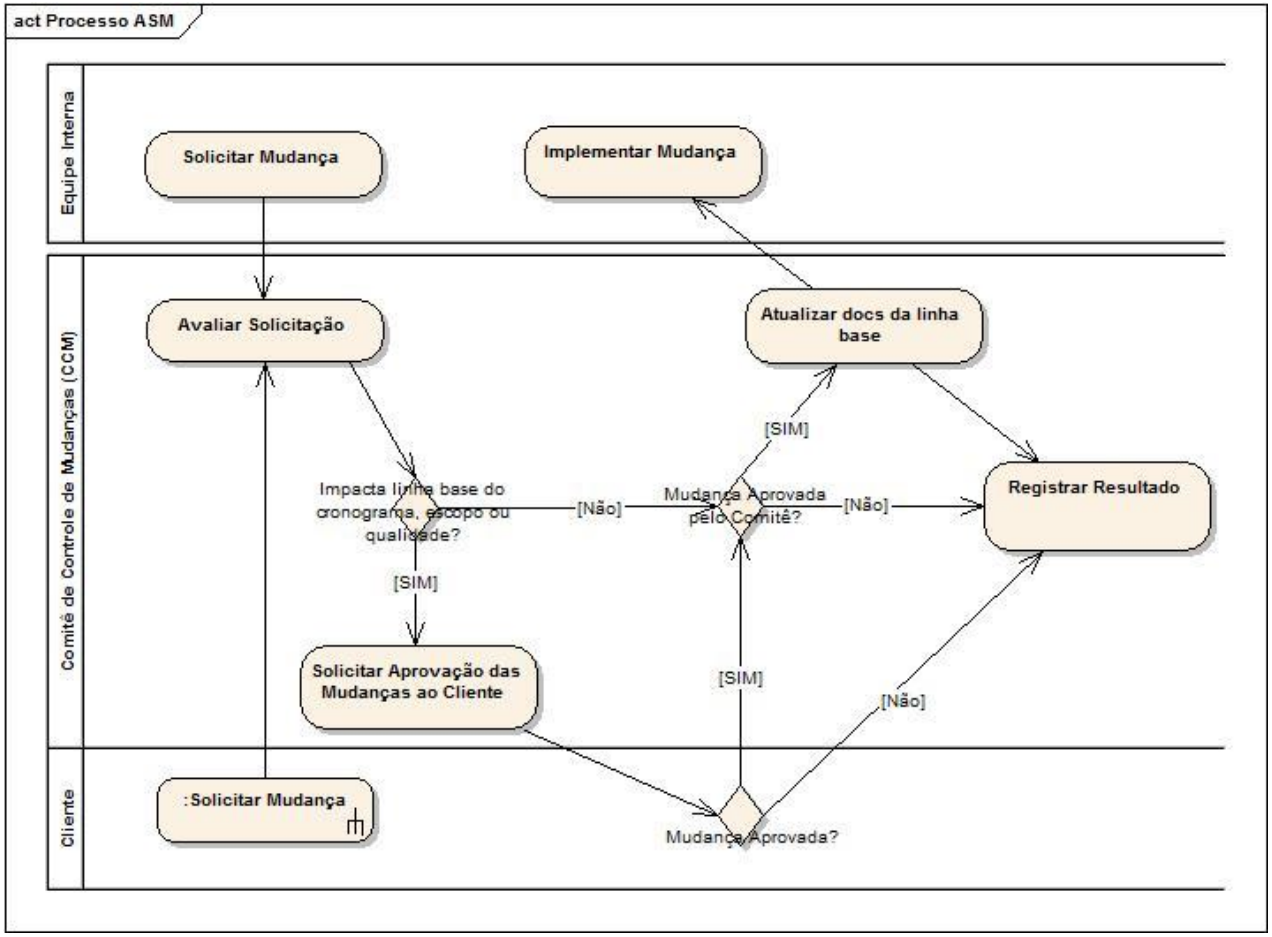

Figura 1 - Fluxo de Aprovação de Solicitação de Mudança

3.4. Registro e Comunicação do Status da Configuração

3.4.1. Processo de Backup

Descreve políticas de retenção, backup, desastre, e plano de recuperação. Também descreve quais mídias devem ser mantidas (online, off-line, tipo de mídia e formato).

3.4.1.1. Backup do repositório SVN

Um backup completo do repositório deverá ser realizado mensalmente e backups incrementais diariamente. Após realizado o backup completo, um expurgo dos backups incrementais anteriores poderá ser executado.

Os backups completos deverão ser gravados em mídia DVD e os backups incrementais em HD Externo.

Os backups incrementais são automatizados por scripts shell e executados através de Jobs (cron Linux). O script dos backups incrementais estão implementados no seguinte endereço do servidor café 192.168.254.109 (mesmo servidor do SVN): `/hd_barracuda/bkp/INFRA-SW/script-backup-svn.bash`.

3.4.2. Expurgo de Projetos Inativos

A cada 3 anos o repositório deverá ser renovado mantendo revisões dos últimos 2 anos e projetos inativos a mais de 2 anos deverão ser removidos.

Caso projetos antigos precisam ser acessados, um chamado para restauração do backup do projeto deverá ser aberto com prazo de 1 semana.

3.4.3. Auditorias

O setor de garantia da qualidade do processo terá a responsabilidade de realizar auditorias esporádicas nos projetos para validar a correta adoção das práticas de gerencia de configuração. Qualquer desvio encontrado deverá ser registrado e reportado ao CCC que tomará as devidas ações.

4. Marcos de projeto

Identifica marcos interno e do cliente relacionados ao esforço de GC do projeto ou produto.

Marco	**Interno ou do Cliente**	**Ações**
Aprovação do documento de requisitos com esforço	Cliente	Fechar uma linha base do escopo e elaborar estimativa de execução do escopo aprovado.
Aprovação do esforço de execução do escopo aprovado	Cliente	Fechar uma linha base do esforço e elaborar cronograma de execução.
Aprovação do cronograma	Cliente	Fechar uma linha base do cronograma e

		divulgar o início da execução do projeto.
Fim da fase de desenvolvimento (implementação e testes)	Interno	Realizar implantação em homologação.
Fim da implantação em homologação	Interno	Realizar a validação do ambiente antes de liberar para o cliente.
Fim da validação do ambiente de homologação	Interno	Liberar para o cliente a homologação do projeto.
Aceite do projeto pelo cliente	Cliente	Pegar o aceite formal do cliente, realizar implantação em produção, repassar mudanças para todas as branches em aberto.
Fim do projeto	Interno	Realizar a reunião de feedback do projeto e registrar as lições aprendidas.

5. Recursos e Treinamento

Descreve as ferramentas de software, pessoal e treinamento requerido para implementar as especificações das atividades de GC.

5.1. Recursos

Segue lista de ferramentas utilizadas nos projetos.

- *Enterprise Architect* (EA): ferramenta de modelagem de banco de dados e software que apoia a GC. Através da exportação dos modelos em formato XML (Padrão XMI) é possível controlar versões dos modelos, bem como realizar merge (com certa limitação) nos arquivos.
- *Microsoft Office Word*: ferramenta de processamento de texto utilizada para elaborar documentos de requisitos, casos de uso, propostas, planos de projeto, de testes, de implantação, entre outros;
- *Microsoft Office Excel:* planilha de processamento de dados utilizada para elaborar estimativas de esforço, cronogramas, controles, etc;
- *Microsoft Office Project:* ferramenta para elaboração e acompanhamento do cronograma de execução do projeto;
- *Tortoise SVN*: ferramenta cliente para acesso e manipulação dos dados do repositório de versão de software;
- *Microsoft Visual Studio e Eclipse*: ferramenta IDE para codificação dos sistemas;
- *Ferramentas SQL Server:* ferramentas para elaboração e consulta de SQL's.

5.2. Treinamento

Para o controle de versão de banco de dados foi feita uma avaliação das ferramentas de modelagem *Power Designer* e *Enterprise Architect* (EA). O EA se saiu melhor na avaliação e será a ferramenta adotada pela empresa para a modelagem de banco de software. Um treinamento de uso da ferramenta será necessário aos analistas responsáveis.

Um treinamento quanto aos padrões, diretrizes e práticas de gerencia de configuração será realizado em todos os níveis da empresa.

5.2.1. Conceitos Básicos e Boas Práticas

Conceitos básicos e boas práticas podem ser vistos em nossa WIKI no endereço a seguir.

- << Endereço >>

Maiores informações sobre controle de versão com SVN podem ser vistas no livro do subversion em:

- << Endereço >>

6. Subcontratação e Controle de Versão do Fornecedor

Descreve como o desenvolvimento de software fora do ambiente do projeto será incorporado.

Fornecedores de desenvolvimento de software contratados deverão seguir nosso plano de gerencia de configuração, salvar os códigos em nosso repositório de versões, reportar o andamento das atividades em nossa ferramenta de gestão dos projetos e se reportarem a nossos gerentes de projetos.