PLANO GERAL DE GERÊNCIA DE CONFIGURAÇÃO

Grupo 06

Sistema de venda de peças de Viaturas

# Introdução

## Propósito

O propósito deste documento 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 gerência 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).

## Definições, Siglas e Abreviações

Nesta seção descrevemos as 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;

# Gerência de Configuração de Software

## Organização, Responsabilidades, e Interfaces

Nesta secção descrevemos o comitê de controle de configuração (CCC), desde a sua constituição, os seus papéis e as respectivas entidades responsáveis.

|  |  |  |
| --- | --- | --- |
| **Cargo** | **Nome** | **Principais papéis** |
| Patrocinador do Projecto | dr. Vali Issufo e eng Michaque | * Financiar o projecto; * Aprovar qualquer assunto que requer escopo, tempo ou custo adicional ao projecto. |
| Gerente do Projecto | Nelson Niero | * Comunicar quaisquer mudanças aprovadas no plano de GC; * Fechar nova linha-base caso haja mudanças no escopo ou prazo. |
| Analista de Suporte | Ramos Domingos Sambo | * Dimensionar os recursos necessários ao bom funcionamento da GC; * Garantir a disponibilidade e desempenho dos recursos necoperaessários; * Executar a configuração de permissões solicitada pelo gerente do projecto. |
| Gerente de Configurações | Marcio Estevão de Jesus | * Repor as solicitações de mudança do processo de GC ao CCC; * Identificar novos IC com apoio dos desenvolvedores; * Solicitar qualquer necessidade de treinamento em GC. |
| Líder de Equipe | Rubens Vilanculos | * Reportar ao gerente de configurações qualquer mudança de GC descoberta durante a fase de execução do projecto; * Identificar ICs candidatos e comunicar ao gerente de configuração; |
| Analista de qualidade | Arnaldo Manguene | * Assegurar que a equipe de desenvolvimento está seguindo os métodos e padroes de GC definidos pelo CCC. * Orientar os desenvolvedores quanto as boas práticas de GC; * Fornecer padroes e templates de configuração à equipe de projecto. |
| Desenvolvedores | Nelson Niero , Marcio Estevao de Jesus e Messias Macamo | * Identificar ICs candidatos e comunicar ao líder da equipe; * Reportar o líder da equipe qulaque mudança de GC descoberta durante a fase de execução do projecto; * Seguir todos os padroes e métodos de GC na implementação do projecto. |

## Ferramentas, Ambientes e Infraestrutura

### 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.

## 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 o endereço:

# https://github.com/Niero/SVPV

# Programação de Gerenciamento de Configuração

## Identificação da Configuração

### Itens de Configuração

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.

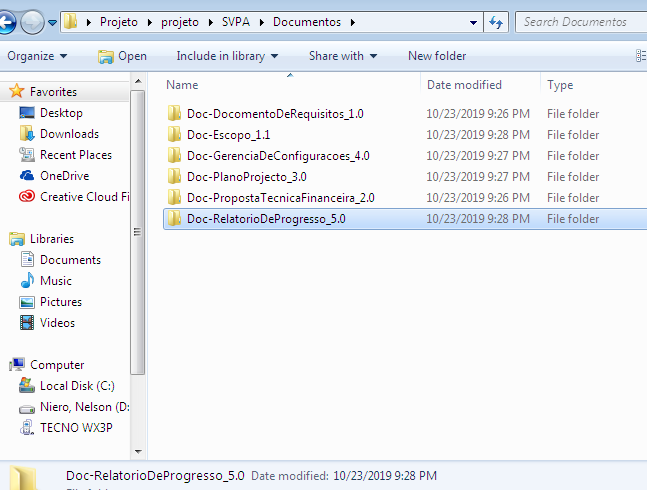
Nem todos os arquivos precisam ser controlados, por exemplo, um executável gerado pelo código fonte não deve estar no repositório, visto que sua fonte geradora já está sendo controlada no repositório.

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

|  |  |  |  |
| --- | --- | --- | --- |
| **Item de Configuração (IC)** | **Descrição** | **Responsável** | **Conclusão** |
| Artister | Ferramenta usada para a prototipagem do sistema em causa (versão 2.3.31) | Nelson Afonso Niero | 10/11/19 |
| Laravel | Ferramenta usada para melhorar o design do sistema. É um front-end framework (versão 3.2) | Ramos Sambo e Nelson Niero | 25/10/19 |
| Adobe Photoshop | Ferramenta usada para a edição de fotos a inserir no sistema. Versão 12.3.2 | Rubens Vilanculos | 01/10/19 |
| Git | Ferramenta usada para a gestão de versões | Vicente Magaia | 12/10/19 |
| Sublime Text | Ferramenta para a edição do código | Arnaldo Maguene | 10/10/19 |
| PHP 7.0 | Versão do Software | Armando Cardoso | 25/10/19 |
| Redmine | Gestão de tarefas | Messias Macamo | 10/10/19 |
| MS Project | Gestão do projecto | Nelson Niero | 10/10/19 |
| MySQL | Plano de Gerência de Configuração para Banco de Dados | Nelson NIERO | 10/10/19 |

### 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.



### Padrões de nomeação de arquivos

*Serão utilizados para nomear os arquivos, a sigla do projeto(SVPA) acrescido de espaço “-”, espaço novamente, nome que identifique o arquivo, underline “\_”, e da versão do mesmo. Exemplo:*

*“SIGLA – SVPA – Plano De Projeto\_3.0”*

### Padrões de nomeação dos objetos de banco de dados

Para nomear os objectos do banco de dados será usada a seguinte estrutura: sigla do projecto (*SVPA*), espaço “-”, nome do objecto

### Padrões de nomeação de código fonte

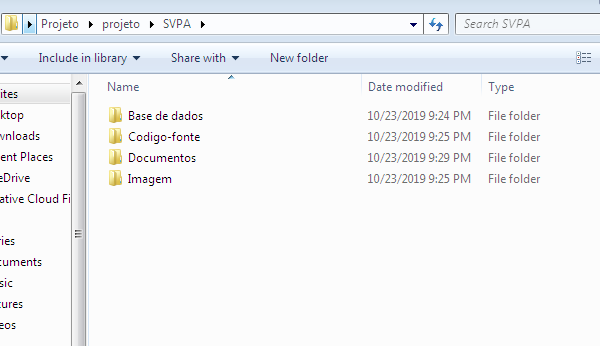
Para nomear os ficheiros do código fonte será usada a seguinte estrutura: sigla do projeto (*SVPA*), espaço “-”, nome do ficheiro*.*

### 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 Figura 1.



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

<< C:\Users\Niero Nelson\Desktop\Projeto\projeto\SVPA\Imagem >>

#### 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.

#### 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).

#### 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<2.0>*

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

#### 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<20191029>\_redmine#<SVPV>**

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

#### 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.

#### 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:<<123455>>; DATA:<<29/10/2019>>; MOTIVO:<<Homologacao do projeto>>”**

#### Tratamento da Pasta Documentação

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

### 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.

### 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

## Controle de Configurações e Mudanças

### Solicitação de Mudança e Aprovação

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

Solicitações de mudança podem ocorrer a qualquer momento no Projecto (iniciação, planeamento, execução ou fechamento). Quanto mais tarde no Projecto ocorrer, mais rígido 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.

## Registo e Comunicação do Status da Configuração

### Processo de Cópia de segurança

Descreve políticas de retenção, cópia de segurança, desastre, e plano de recuperação. Também descreve quais médias devem ser mantidas (online, off-line, tipo de média e formato).

#### Cópia de segurança do repositório SVN

Um cópia de segurança 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 média DVD e os backups incrementais em HD Externo.

Os backups incrementais são automatizados por scripts shell e executados através de Jobs (cron Linux).

# Marcos de Projecto

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

|  |  |  |
| --- | --- | --- |
| **Marco** | **Interno ou do Cliente** | **Acções** |
| 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 Projecto. |
| 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 Projecto. |
| Aceite do Projecto pelo cliente | Cliente | Pegar o aceite formal do cliente, realizar implantação em produção, repassar mudanças para todas as branches em aberto. |
| Fim do Projecto | Interno | Realizar a reunião de feedback do Projecto e registar as lições aprendidas. |

# Recursos e Treinamento

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

## 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.

## 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.

### Conceitos Básicos e Boas Práticas

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

* www.EngSoftIISVPV.co.MZ

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

* WWW.EngSoftIIVersao.co.MZ

# 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.