**SIGEAS**

**Plano de Gerência de Configuração**

Monteiro, 2015

# Revisões

|  |  |  |
| --- | --- | --- |
| **Data** | **Descrição** | **Autor** |
| 05.12.2015 | Alteração em tabela de identificadores e nomes dos artefatos. Versão: 1.1 | Luiz Antonio Pereira |
| 08.12.2015 | Alteração em tabela de identificadores e nomes dos artefatos. Versão: 1.1 | Luiz Antonio Pereira |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |

**CONTEÚDO**

[1 Introdução 4](#__RefHeading___Toc413326730)

[1.1 Objetivos 4](#__RefHeading___Toc413326731)

[1.1.1 Organização do Documento 4](#__RefHeading___Toc413326732)

[2 Papeis e Responsabilidades 5](#__RefHeading___Toc413326733)

[3 Plano de Configuração 5](#__RefHeading___Toc413326734)

[3.1 Controle de Configuração 5](#__RefHeading___Toc413326735)

[3.1.1 Estrutura do Repositório de Gerência de Configuração 6](#__RefHeading___Toc413326736)

[4 Métodos de Identificação 7](#__RefHeading___Toc413326737)

[4.1 Documentos 7](#__RefHeading___Toc413326738)

[4.2 Versionamento 8](#__RefHeading___Toc413326739)

[4.3 Baselines e Releases 9](#__RefHeading___Toc413326740)

[5 Ambiente, Ferramentas e Infra-estrutura 10](#__RefHeading___Toc413326741)

[5.1 Plano de Software 10](#__RefHeading___Toc413326742)

[6 Estrutura de Branches do Projeto 11](#__RefHeading___Toc413326743)

# Introdução

Este documento descreve o Plano de Gerência de Configuração para o projeto de desenvolvimento do sistema SIGEAS.

## Objetivos

O presente documento tem por objetivo apresentar a organização, nomenclatura e regras de versionamento para a gerencia de configuração do projeto de desenvolvimento do sistema SIGEAS.

Este plano é destinado a todos os integrantes da equipe responsável pelo o desenvolvimento do sistema.

### Organização do Documento

As seções subsequentes deste documento estão assim organizadas:

* Seção 2, onde são descritos os papeis e responsabilidades da gerência de configuração;
* Seção 3 onde é apresentado o plano de configuração onde é definido a estrutura do armazenamento, as configurações bases do projeto, o controle de configuração e as políticas de segurança e acesso aos itens de configuração;
* Seção 4, onde são descritos os métodos de identificação de documentos, versionamentos, baselines e releases.
* Seção 5, onde são descritos o ambiente, as ferramentas e a infraestrutura utilizadas no desenvolvimento do projeto.
* Seção 6, onde contém a estratégia de branches para execução do projeto.
* Seção 7, onde estão presentes as referências.

# Papeis e Responsabilidades

|  |  |
| --- | --- |
| **Papel** | **Responsabilidade** |
| Gerente de Desenvolvimento (GD) | Juntamente com a CCM receber, analisar e aprovar os PFM. |
| Líder de Projeto (LP) | Planejar as atividades de GC juntamente com o Responsável pela Configuração, designar executante, finalizar SM, autorizar a criação das configurações bases conforme descrito na seção Plano de Configuração. |
| Responsável pela Configuração (RC) | Criar e manter infraestrutura corporativa (servidores) de GC; Implementar as políticas de Controle de Acesso ao ambiente de GC, Realizar os backups dos repositório de configuração dos projetos |

# Plano de Configuração

## Controle de Configuração

O servidor que será usado para controle de configuração é o Git. Nossa escolha se justifica não somente pelo fato deste servidor estar presente em um grande número de projetos de softwares de alto-padrão (como Eclipse, GNOME e jQuery) [1], mas também por conter características necessárias e importantes para o gerenciamento de configuração, como: suporte consistente para desenvolvimentos não-lineares; desenvolvimento distribuído; autenticação criptográfica do histórico; estratégias de *merge* conectáveis. [2]

### Estrutura do Repositório de Gerência de Configuração

A seguir, será apresenta a estrutura definida para armazenamento dos artefatos do projeto no repositório.

Repositório

+-01-SISTEMA

| +-01-REQUISITOS

| +-02-ANALISE E PROJETO

| +-03-IMPLEMENTACAO

| +-04-TESTE

| +-05-GERENCIA DE CONFIGURACAO

+-02-PROJETO-2015.2

| +-01-GERENCIA DE PROJETO

| +-02-ACOMPANHAMENTO

| +-CONSVV-ITYY

| +-03-ATAS DE REUNIAO

| +-04-TESTE

| +-CONSVV-ITYY

O repositório é dividido em Sistema, para artefatos que pertencem ao sistema independente de quantos projetos irão desenvolvê-lo e Projeto-XXX, onde XXX é o identificador do projeto. Isso é necessário pois vários projetos podem ser executados para o desenvolvimento do mesmo sistema. Os subdiretórios estão organizados pelas disciplinas do processo. A tabela abaixo identifica o que deve ser armazenado em cada subdiretório do projeto.

|  |  |  |
| --- | --- | --- |
| Diretório Principal | Subdiretório | Descrição |
| 01-SISTEMA | 01-REQUISITOS | Armazenamento de Documentos de requisitos, documentos de casos de uso, Product backlog, User stories, etc. |
| 02-ANALISE E PROJETO | Documentos referentes à análise e projeto do sistema como documentos de arquitetura, modelos UML, modelo de dados, dicionário de dados, etc. |
| 03-IMPLEMENTACAO | Nesse diretório deve estar todo o código relativo ao sistema. |
| 04-TESTE | Documento relativo aos cenários de teste para as User Stories. |
| 05-GERENCIA DE CONFIGURACAO | Armazenamento do plano de gerência de configuração. |
| 02-PROJETO-2015.2 | 01-GERENCIA DE PROJETO | Armazenamento do plano de projeto. |
| 02-ACOMPANHAMENTO | Deve possuir um subdiretório para cada iteração do projeto contendo o plano da iteração, reuniões de acompanhamento da iteração, Reunião de finalização da iteração, Burndown da iteração e timeline da iteração. |
| 03-ATAS DE REUNIAO | Armazenamento das Atas de reunião do projeto. |
| 04-TESTE | Armazenamento dos relatórios de execução de testes. Deve possuir um subdiretório para cada iteração uma vez que os testes são executados por iteração. |

# Métodos de Identificação

## Documentos

Todos os documentos disponibilizados no repositório devem ser identificados baseados na seguinte nomenclatura:

<ID\_ARTEFATO>-<NOME\_ARTEFATO>

Onde:

* <ID\_ARTEFATO> é a sigla de identificação do artefato conforme Tabela 1.
* <NOME\_ARTEFATO> é nome de identificação do artefato conforme Tabela 1.

|  |  |
| --- | --- |
| ID\_ARTEFATO | NOME\_ARTEFATO |
| PPJ | Plano de Projeto |
| ARQ | Documento de Arquitetura |
| REQ | Documento de Requisitos |
| UCS | Documento de Casos de Uso |
| PEC | Documentação de Projeto Estrutural e Comportamental |
| PBL | Product Backlog |
| TEA | Testes de Aceitação |
| PGC | Plano de Gerência de Configuração |
| PIT | Plano de Iteração |
| DRI | Documento de Riscos |
| ATA | Ata de Reunião |
| RAP | Relatório de Acompanhamento do Projeto |
| BRD | Burndown |

Tabela 1 - Identificadores e Nomes dos Artefatos

## Versionamento

A abordagem de versionamento de software se baseará no seguinte modelo:

<MAJOR>.<MINOR>.<PATH>

* <MAJOR> é a versão da release disponibilizado ao cliente. Esta versão deve ser incrementada se uma nova release for liberada. Nesse caso, as outras versões (<MINOR> e <PATH>) devem ser redefinidas para zero (0).
* <MINOR> é a versão que faz referência a novas funcionalidades implementadas na release disponibilizada ao cliente. Esta versão deve ser incrementada a cada nova funcionalidade promovida a release liberada. Nesse caso, a versão referente a correção de erros (<PATH>) deve ser redefinidas para zero (0).
* <PATH> é a versão que faz referência a correção de erros encontrados em uma versão de release disponibilizada ao cliente. Esta versão deve ser incrementada a cada correção de erro encontrado na release liberada.

## Baselines e Releases

Sempre que tivermos uma versão estável do software para testes ou aprovada da documentação deve ser gerada uma baseline com o objetivo de recuarmos os estado em que ficou o código e a documentação.

Abaixo, segue o modelo para nomenclatura de baselines e releases. Durante uma iteração, várias baselines são geradas, com o objetivo de ser utilizada para testes. Caso uma baseline possua erro uma nova baseline deverá ser gerada para uma nova rodada de testes. Assim, as baselines devem seguir a seguinte nomenclatura:

B-CONS<ID\_VERSAO>-IT<NUMERO\_ITERACAO>-<NUMERO\_DA\_BASELINE>

Onde:

* ID\_VERSAO: O identificador da versão que está sendo construída.
* NUMERO\_ITERACAO: O número da iteração em que foi gerada a baseline.
* NUMERO\_BASELINE: O número sequencial da baseline dentro da iteração.

Para releases a nomenclatura é parecida. Todo o fim de iteração deve ser gerada uma release com a versão a ser validada pelo cliente. Assim, as releases de iteração devem seguir a seguinte nomenclatura.

R- CONS<ID\_VERSAO>-IT<NUMERO\_ITERACAO>

Onde:

* ID\_VERSAO: O identificador da versão que está sendo construída.
* NUMERO\_ITERACAO: O número da iteração em que foi gerada a release.

# Ambiente, Ferramentas e Infra-estrutura

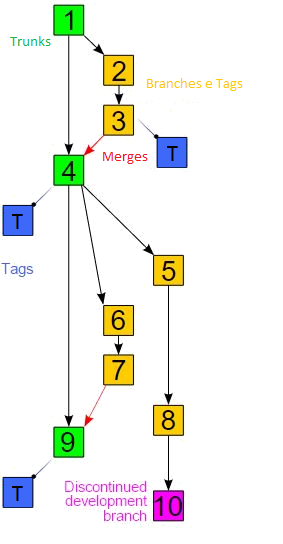
## Plano de Software

|  |  |  |  |
| --- | --- | --- | --- |
| **Software** | **Propósito** | **Ambiente** | **Release/Versão** |
| Git | Controle de Repositório | Desenvolvimento | 2.6.3 |
| MS-Office | Documentos do Word | Todos | 2010 |
| Libbre Office | Documentos do Word | Todos | 5.x |
| GitHub | Acesso ao repositório | Desenvolvimento |  |
| IceScrum | Edição do cronograma | Desenvolvimento | 2007 |
| Java JDK | Kit básico para o desenvolvimento em Java | Desenvolvimento | 1.8.x |
| JSF | Framework MVC para o desenvolvimento de aplicações Web.. | Desenvolvimento | 2.2 |
| Hibernate | Framework de persistência. | Desenvolvimento | 5.0.x |
| iReport | IDE para design de relatórios. | Desenvolvimento | 5.5.x |
| JasperReports | Biblioteca responsável em gerar relatórios em PDF. | Desenvolvimento | 6.1.x |
| Eclipse | IDE de desenvolvimento. | Desenvolvimento | 4.4 |
| Astah Community | Modelagem de diagramas de dados. | Desenvolvimento | 7.0 |
| PGAdmin | Gerenciador de banco de dados PostgreSQL. | Desenvolvimento | 1.2 |
| PostgreSQL | Gerenciador de banco de dados. | Desenvolvimento | 9.4.x |
| TomCat | Servidor de Aplicações | Desenvolvimento | 9.0 |

Tabela 2 - Plano de Software

# Estrutura de Branches do Projeto

A estratégia de branches para execução do projeto segue o modelo descrito abaixo:



O ‘*Trunk*’ será responsável por conter todas as *releases* programadas para o fim de cada iteração.

Os ‘*branches’* serão responsáveis por abrigar a evolução do desenvolvimento de cada iteração. Após a finalização do desenvolvimento então, um *merge* é feito para integrar o desenvolvimento do *branch* atual ao *trunk.*

Ao fim do desenvolvimento de cada *branch* serão adicionadas *tags* que farão referências as *baselines.* E, após o *merge* ser realizado, *tags* então serão inseridas para referenciar as *releases*.