Universidade Federal do Pampa
Giovane D'Avila Mendonça

# PROPOSIÇÃO DE MELHORIAS EM UM PROCESSO DE GERENCIAMENTO DE CONFIGURAÇÃO DE SOFTWARE

Alegrete
2016

Giovane D'Avila Mendonça

# PROPOSIÇÃO DE MELHORIAS EM UM PROCESSO DE GERENCIAMENTO DE CONFIGURAÇÃO DE SOFTWARE

Trabalho de Conclusão de Curso apresentado ao Curso de Graduação em Engenharia de Software da Universidade Federal do Pampa como requisito parcial para a obtenção do título de Bacharel em Engenharia de Software.

Orientador: Prof. Dr. Cristiano Tolfo

Alegrete

2016

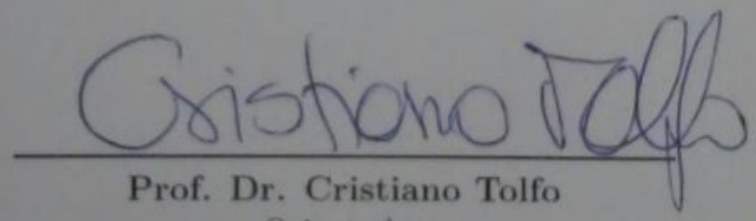
Giovane D'Avila Mendonça

# PROPOSIÇÃO DE MELHORIAS EM UM PROCESSO DE GERENCIAMENTO DE CONFIGURAÇÃO DE SOFTWARE

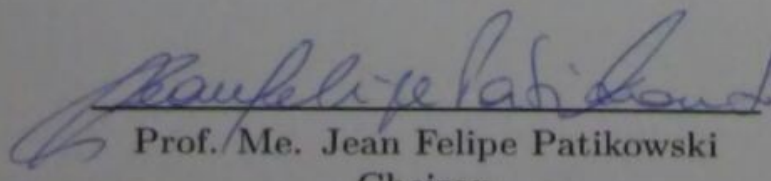
Trabalho de Conclusão de Curso apresentado ao Curso de Graduação em Engenharia de Software da Universidade Federal do Pampa como requisito parcial para a obtenção do título de Bacharel em Engenharia de Software.

Trabalho de Conclusão de Curso II defendido e aprovado em 23 de junho de 2016.

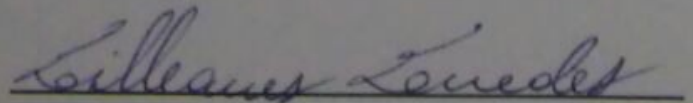
Banca examinadora:



**Prof. Dr. Cristiano Tolfo**
Orientador
UNIPAMPA



**Prof. Me. Jean Felipe Patikowski Cheiran**
UNIPAMPA



**Prof. Dr. Gilleanes Thorwald Araujo Guedes**
UNIPAMPA

*Este trabalho é dedicado à minha família por estarem presente nos momentos bons e apoiarem os momentos difíceis.*

# Agradecimentos

Agradeço primeiramente à Deus por me manter equilibrado em todos os momentos.

À minha família, amada e companheira de longa data Denise, filhas, Hanna e a pequena Valentina, que proporcionaram momentos de alegria para seguir em frente e superar as dificuldades. Pela paciência que tiveram nessa longa jornada, superando os momentos de minha ausência. Pai e mãe por incentivarem e acreditarem no meu potencial.

Professor Cristiano Tolfo, muito obrigado pela dedicação e orientação para que este trabalho fosse possível. Aos demais professores que transmitiram suas experiências e conhecimentos.

Aos colegas de trabalho da Diretoria de Tecnologia da Informação e Comunicação da Unipampa pelo apoio e palavras de incentivo. Em especial aos servidores Daniel Biasoli e Sérgio Bortolin Júnior pelo auxílio e prestação de informações importantes para elaboração deste trabalho.

# Resumo

Toda equipe de desenvolvimento de software deve possuir um processo de software definido, e para isso pode ser escolhido o modelo que melhor atenda as necessidades da equipe. Independente do processo escolhido, deve-se adotar como atividade de apoio o gerenciamento de configuração, que permite identificação dos itens de configuração que devem ser controlados e definição de papéis dos envolvidos. A proposta deste trabalho consiste em modelar o processo de gerenciamento de configuração da Diretoria de Tecnologia da Informação e Comunicação, que atualmente é realizado de forma empírica e propor melhorias com base na literatura atual. Com isso, espera-se contribuir com a formalização do processo de gerenciamento de configuração de software, a identificação de melhorias e a proposta de um processo de gerenciamento de configuração de software adequado à realidade da Diretoria de Tecnologia da Informação e Comunicação. Como metodologia foi utilizado um estudo de caso, para entender o processo atual de gerenciamento de configuração de software foram realizadas entrevistas informais e observações diretas no local de estudo de caso. Para modelagem e representação do processo formalizado foi utilizada a notação Business Processing Model Notation. Como resultado, foi possível verificar o fluxo geral do processo de gerenciamento de configuração de software da Diretoria de Tecnologia da Informação e Comunicação, identificação dos envolvidos no processo. Após o entendimento do processo atual foram identificados pontos possíveis de melhorias que foram adicionados ao modelo de processo melhorado. Este modelo foi discutido e refinado com a equipe da Coordenação de Desenvolvimento de Sistemas para que o modelo de processo futuro atendesse a realidade do setor. Ao final do trabalho proposto, teve-se como resultado um modelo de processo de gerenciamento de configuração de software que agregou qualidade e permitiu a melhor visualização dos processos de gerenciamento de configuração de software.

**Palavras-chave**: Gerenciamento de Configuração de Software. Gerenciamento de Mudanças. Modelagem de Processos. BPMN.

# Abstract

Every software development team must have a defined software process, and for this it must be chosen the model that is best suitable to the team necessities. Independently of the chosen process, it must be adopted as support activity the configuration management, which allows the identification of the configuration items that must be controlled and the definition of the role of the involved ones. The proposal of this work is based in to model the configuration management process of the Communication and Information Technology Directory, which is currently realized in an empirical way, and to propose improvements based on the actual literature. Then, it is waited to contribute with the formalization of the software configuration management process, the identification of improvements and the propose of a software configuration management process adequate to the reality of the Communication and Information Technology Directory. As methodology, it was utilized a case study, to understand the current software configuration management process it were realized informal interviews and direct observations in the case study local. For modeling and representation of the formalized process, it was utilized the Business Processing Model Notation. As result, it was possible to verify the general flow of the software configuration management process of the Communication and Information Technology Directory, identifying the involved on the process. After the comprehension of the current process, it were identified possible improvement points that were added to the improve process model. This model was discussed and refined with the System Development Coordination team in order to allow the model of the future process to attend the sector reality. At the end of the proposed work, it was obtained as result a model of software configuration management process that attended the reality of the sector, aggregating quality and allowing better visualization of the software configuration management process.

**Key-words**: Software Configuration Management. Changes Management. Process Modeling. BPMN.

# Lista de ilustrações

# Lista de tabelas

# Lista de siglas

**APF** Administração Pública Federal

**ATI** Analista de Tecnologia da Informação

**BPMN** Business Process Modeling Notation

**CAU** Coordenação de Apoio ao Usuário

**CMMI** Capability Maturity Model Integration

**CODEV** Coordenação de Desenvolvimento de Sistemas

**CSI** Coordenadoria de Segurança da Informação

**DRF** Documento de Requisitos Funcionais

**DRS** Documento de Requisitos de Software

**DTIC** Diretoria de Tecnologia da Informação e Comunicação

**ER** Entidade Relacionamento

**EUA** Estados Unidos da América

**GC** Gerenciamento de Configuração ou Gestão de Configuração

**GCM** Gerenciamento de Configuração de Mudanças

**GCS** Gerenciamento de Configuração de Software

**IC** Item de Configuração

**ICs** Itens de Configuração

**NTIC** Núcleo de Tecnologia da Informação e Comunicação

**TI** Tecnologia da Informação

**UC** Casos de Uso

**UML** *Unified Modeling Language* ou Linguagem de Modelagem Unificada

**UNIPAMPA** Fundação Universidade Federal do Pampa

# Sumário

# 1 Introdução

A alta rotatividade de pessoal em equipes de desenvolvimento de software tem como consequência atrasos em entregas, uma vez que novos integrantes precisam conhecer o ambiente de trabalho, fluxos de atividades e papéis de cada envolvido no processo de *software*.

Existem várias mudanças durante o ciclo de vida de um projeto de *software*. Além das pessoas envolvidas, requisitos de software também são alterados ou novos requisitos são incluídos. Estas mudanças são constantes durante as fases de desenvolvimento, e este fato torna essencial um controle rigoroso do impacto das mudanças sofridas por estas alterações.

Segundo Engholm (2010), para termos um gerenciamento de configuração eficaz, é preciso definir um processo apropriado ao desenvolvimento de projetos, definindo atividades e procedimentos administrativos, técnicos e de inspeções relacionadas aos itens de controle de configuração. Com isso, é preciso definir um modelo de processo de configuração de *software* adequado à realidade da Coordenação de Desenvolvimento de Sistemas (CODEV).

A CODEV está subordinada à Diretoria de Tecnologia da Informação e Comunicação[1] (DTIC) da Fundação Universidade Federal do Pampa (UNIPAMPA). Este setor é responsável pelo desenvolvimento de softwares para a instituição. Tendo como política a melhoria contínua de seus processos, para isso, a Diretoria apropria-se de trabalhos científicos de alunos de graduação, baseando-se nestes estudos, para analisar e melhorar seus processos.

Um dos trabalhos utilizados é a pesquisa elaborada por SANTOS (2014), que teve como objetivo a modelagem de processo de testes de software para CODEV. Este processo e suas respectivas melhorias ainda são utilizadas atualmente.

Com isso, este trabalho tem como propósito completar os processos de suporte ao ciclo de projeto de software proposto por Engholm (2010), sendo eles, a Garantia da Qualidade de Software e Gerenciamento de Configuração ou Gestão de Configuração (GC).

Com a formalização do processo de gerenciamento de configuração, será possível a melhor visualização das atividades envolvidas no processo atual, o que permitirá a verificação, análise e proposição de melhorias no processo de Gerenciamento de Configuração

[1] Conforme Boletim de Serviço UNIPAMPA Nº 244-2016, o Núcleo de Tecnologia da Informação e Comunicação (NTIC) passou a se chamar Diretoria de Tecnologia da Informação e Comunicação (DTIC).

de Software da Diretoria de Tecnologia da Informação e Comunicação.

## 1.1 Objetivo Geral

Propor melhorias no processo de gerenciamento de configuração de *software* para a Diretoria de Tecnologia da Informação e Comunicação (DTIC).

## 1.2 Objetivos Específicos

- Modelar o processo de gerenciamento de configuração da Coordenação de Desenvolvimento de Sistemas (CODEV);
- Formalizar o processo empírico pela modelagem do processo de gerenciamento de configuração de software;
- Identificar pontos que podem ser melhorados a partir do estudo proposto por este trabalho;
- Facilitar o planejamento e a comunicação entre os membros da Diretoria de Tecnologia da Informação e Comunicação DTIC com a formalização do processo;
- Diminuir o tempo de adaptação de novos integrantes da equipe da Coordenação de Desenvolvimento de Sistemas (CODEV);

## 1.3 Justificativa

Atualmente, a CODEV possui um processo de controle de versão e mudanças utilizado de forma implícita, no qual as tarefas realizadas não estão formalizadas. Artefatos utilizados foram escolhidos de forma empírica, por isso, é importante que se desenvolva o mapeamento da situação atual do processo de Gerenciamento de Configuração e identificação de pontos de melhorias neste processo.

Ao encontro disso, este trabalho contribui para modelar o processo de gerenciamento de configuração de software da CODEV, que, atualmente, é empírico, permitindo melhorar a gerência de projetos, minimizar os riscos na fase de desenvolvimento, aumentar a produtividade, melhorar o controle de incidências, manter a integridade dos artefatos, manter a rastreabilidade das alterações realizadas entre versões de software e facilitar a auditoria de mudanças.

Nesse sentido, SANTOS (2013) menciona que o mapeamento de processos traz vantagens competitivas, tendo em vista que melhora o entendimento dos processos por

parte dos administradores e colaboradores. Além disso, a comunicação é facilitada pela padronização dos processos que, antes, estavam implícitos.

## 1.4 Estrutura da Monografia

Este trabalho está organizado da seguinte forma: no Capítulo 2 realiza-se a revisão da literatura com os conceitos relacionados a este trabalho; Já o Capítulo 3 apresenta trabalhos relacionados à esta pesquisa que seviram de base teórica para desenvolvimento deste trabalho; O Capítulo 4 aborda a metodologia utilizada para realização do trabalho; Na sequência, o Capítulo 5 apresenta o desenvolvimento do trabalho, sumarizando-se os resultados encontratos; No Capítulo 6 estão descritas as conclusões do trabalho elaborado e possibilidades de trabalhos futuros. E, por fim, são apresentadas as referências utilizadas neste trabalho.

# 2 Revisão da Literatura

Este capítulo consiste na revisão de literatura que serviu de apoio ao desenvolvimento deste trabalho. A seção 2.1, descreve a definição de processo de software e suas atividades. Na seção 2.2, é descrito conceitos relacionados ao gerenciamento de configuração. E por fim, na seção 2.3, são abordados conceitos de BPMN utilizados como apoio para desenvolvimento da proposta deste trabalho.

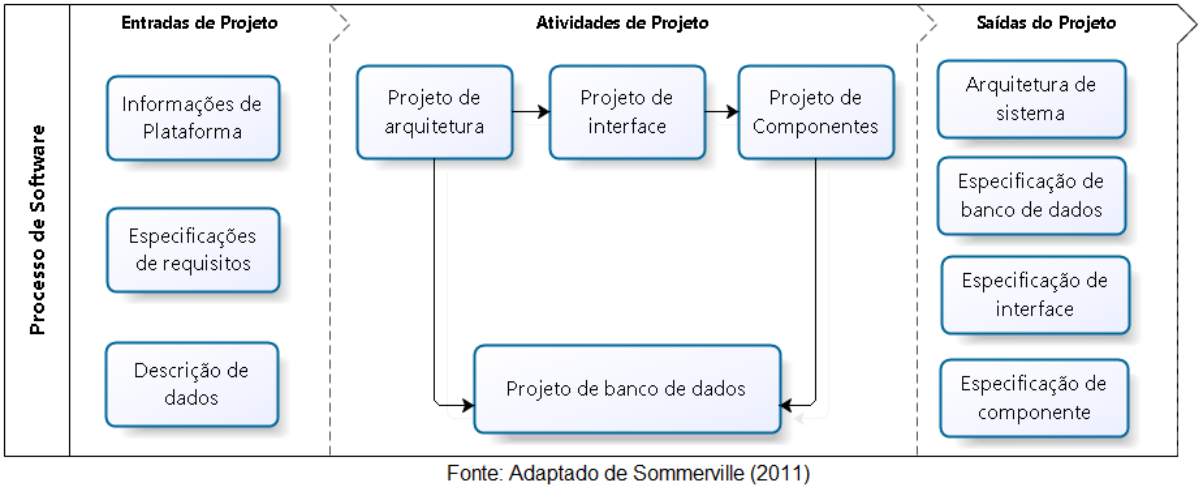
## 2.1 Processo de Software

Segundo Sommerville (2011), o processo de software consiste em um conjunto de atividades relacionadas que levam à produção de um produto de software. Nesse sentido, Engholm (2010) complementa que um processo de software é usado para criar, inventar, projetar, transformar, produzir, controlar, manter e usar produtos ou sistemas.

Wazlawick (2013) complementa que processos usualmente são definidos como conjuntos estruturados de atividades para as quais são determinados artefatos de entrada e saída, papéis de responsáveis e participantes, além de recursos necessários.

Para melhor entendimento das atividades envolvidas em processo de software, Sommerville (2011) apresenta um fluxo geral que pode ser visualizado na Figura 1.

Figura 1 – Processo de software




Fonte: Adaptado de Sommerville (2011)

O estabelecimento de um processo de software definido e formalizado traz benefícios, como melhorar a visibilidade das atividades, destacar as partes envolvidas em cada atividade e identificar os artefatos gerados nestas atividades. Nesse sentido, Sommerville

(2011) reforça que a padronização dos processos melhora a comunicação, além de reduzir o tempo de treinamento dos novos integrantes da equipe de desenvolvimento.

Para formalização de um processo de software, não podem ser adotados os mesmos métodos, ferramentas e processos em empresas ou organizações que desenvolvem produtos semelhantes, pois sempre existem fatores organizacionais locais (SOMMERVILLE, 2011).

Com isso, na proposta de melhoria do processo de gerenciamento de configuração da CODEV, serão considerados fatores organizacionais e a estrutura da organização, tendo como resultado um modelo de processo que atenda à realidade do setor.

### 2.1.1 Atividades básicas de um processo

Existem vários processos de software que podem ser adotados, podendo ser seguidas as atividades de um processo específico ou a combinação das melhores práticas de vários processos. A rigor, segundo Sommerville (2011), existem quatro atividades básicas do processo, sendo elas: especificação de software, projeto e implementação de software, validação de software e evolução de software. No entanto, a organização dessas atividades terá formas diferentes conforme o processo de desenvolvimento adotado.

A primeira atividade básica é a especificação de software ou engenharia de requisitos, que, segundo Sommerville (2011), é o processo de compreensão e definição dos serviços requisitados do sistema e identificação de restrições relativas à operação e ao desenvolvimento do sistema. Esta atividade é uma etapa crítica do processo de software, pois erros, nesta fase, geram problemas no projeto e na implementação do sistema.

Pressman (2011) acrescenta que a engenharia de requisitos é uma importante ação que se inicia na atividade de comunicação e continua na de modelagem. Ainda complementa que ela deve ser adaptada às necessidades do processo, do projeto, do produto e das pessoas que estarão realizando o trabalho. Pfleeger (2004) complementa que a análise de requisitos fornece um meio para que os clientes e desenvolvedores entrem em um acordo sobre o que o sistema fará.

Na engenharia de requisitos, é produzido um documento de requisitos acordados, que vai especificar o sistema que irá satisfazer os requisitos dos *stakeholders*, sendo estes todos os envolvidos com o projeto de software. Os requisitos são apresentados, geralmente, em dois níveis de detalhe, sendo que os usuários finais e clientes necessitam de uma declaração de requisitos em alto nível, já os desenvolvedores precisam de uma especificação mais detalhada do sistema (SOMMERVILLE, 2011).

A segunda atividade básica do processo de software é a de projeto e implementação de software, que é o processo de conversão de uma especificação do sistema definido no processo de engenharia de requisitos em um sistema executável. Essa atividade envolve processos de projeto e programação de software e, se usada uma abordagem incremental

para o desenvolvimento, também pode envolver o refinamento da especificação do software. Em consequência, o documento de requisitos sofrerá mudanças (SOMMERVILLE, 2011).

A terceira atividade básica consiste na validação de software, ou mais conhecida, como verificação e validação (V&V), tendo como propósito mostrar que o software se adequa as suas especificações ao mesmo tempo que satisfaz as especificações do cliente do sistema (SOMMERVILLE, 2011).

Para Pfleeger (2004), a validação dos requisitos é o processo no qual se determina se a especificação é consistente em relação à definição dos requisitos. Ou seja, a validação assegura que os requisitos atenderão às necessidades dos clientes.

As mudanças são inevitáveis em todos os grandes projetos de software. Os requisitos do sistema mudam e estas mudanças podem ser feitas a qualquer momento durante ou após o desenvolvimento do sistema (SOMMERVILLE, 2011). Este fato é denominado como evolução de software ou manutenção de software, sendo esta a quarta atividade básica que todo processo de software deve possuir.

Segundo Pressman (2011), a manutenção começa quase imediatamente após o software ser liberado. Novas necessidades dos clientes que não foram solicitadas durante o desenvolvimento ou até mesmo adaptações ao sistema solicitados por novos clientes podem ser levantadas, isto tudo gera demandas para manutenção do sistema.

As atividades envolvidas no processo de software citadas anteriormente são iniciadas com artefatos de entrada e produzem, como saídas, novos artefatos ou modificação destes (WAZLAWICK, 2013). Estes artefatos sofrem mudanças constantes, devendo-se incluí-los no processo de gerenciamento de configuração.

Cada atividade do processo de software pode ser realizada por uma pessoa ou grupo de pessoas, que são identificadas como papéis num processo de software. Conforme destaca Wazlawick (2013), estas pessoas devem realizar as atividades e responder pela sua conclusão.

### 2.1.2 Papéis em processo de software

Papéis ou responsáveis definidos facilitam a comunicação e identificação de responsabilidades durante o ciclo de desenvolvimento de software. Segundo Wazlawick (2013), "[..] na descrição de um processo, as atividades devem ser atribuídas a perfis ou cargos, e não a pessoas".

Esta definição, entre outros benefícios, facilita o entendimento e visualização por novos integrantes da equipe de desenvolvimento, tendo em vista que pessoas mudam, mas os papéis permanecem os mesmos.

Após o mapeamento das atividades envolvidas no projeto de desenvolvimento e definição de papéis para cada atividade, deve-se identificar os artefatos produzidos, alterados ou excluídos durante o ciclo de desenvolvimento de software. Isso permite a visualização do processo como um todo, em quais atividades os artefatos serão utilizados, monitorados ou alterados.

### 2.1.3 Artefatos em processo de software

Wazlawick (2013) define artefato como qualquer documento que puder ser produzido durante um projeto de desenvolvimento de software. Podemos citar alguns exemplos de artefatos, como diagramas, documentos de texto, protótipos, partes de código, cronogramas, entre outros.

Esses artefatos devem ser incluídos no gerenciamento de configuração de software, para que sejam monitoradas as alterações e identificados os responsáveis pelas mudanças, bem como para manter a rastreabilidade do impacto destas mudanças.

No contexto de Gerenciamento de Configuração, um artefato é denominado como um Item de Configuração, portanto, nesse contexto, todos os artefatos gerados e que devem ser monitorados e controlados são chamados de Itens de Configuração.

### 2.1.4 Gerenciamento de configuração no processo de software

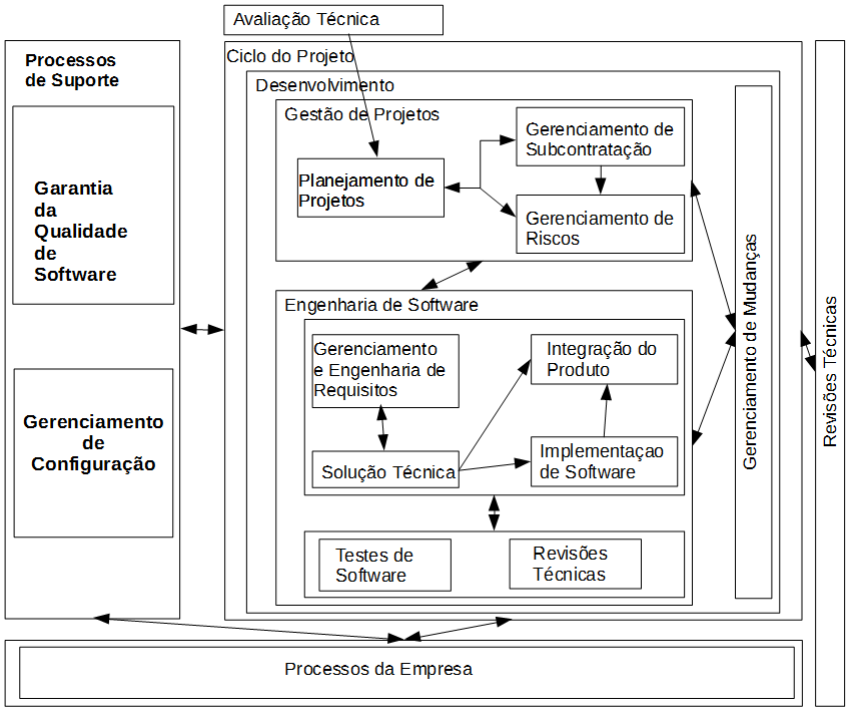
Além das atividades fundamentais, existem atividades de apoio que dão suporte ao processo de software. Sommerville (2011) descreve duas dessas atividades: documentação e gerenciamento de configuração de software.

Engholm (2010) apresenta uma proposta de processo para o ciclo de projeto de software, conforme pode ser visualizado na Figura 2.

Como pode ser observado na Figura 2, define o Gerenciamento de Configuração como um processo de suporte ao ciclo do projeto. Assim como o processo de Garantia da Qualidade do Software, o processo de Gerenciamento de Configuração tem o propósito de garantir a qualidade do software desenvolvido.

A CODEV já possui o processo de V&V formalizado, o que permitiu ganhos em qualidade. Com isso, este trabalho tem como propósito completar o Processo de Suporte ao Ciclo de Projeto proposto por Engholm (2010). Com a formalização do processo, a CODEV poderá analisar e melhorar seus processos de desenvolvimento de software.

Figura 2 – Proposta de processo de software.




Fonte: Engholm (2010)

## 2.2 Gerenciamento de Configuração

Para Wazlawick (2013), o Gerenciamento de Configuração de Software (GCS), ou Gerenciamento de Configuração de Mudanças (GCM), é considerado uma disciplina à parte dentro do gerenciamento de projetos. Nesse sentido, Poppendieck (2011) reforça que a gerência de configuração é uma disciplina central em qualquer ambiente de desenvolvimento de software.

O GCS está fortemente ligado à qualidade de software, tendo como finalidade facilitar o desenvolvimento de software. Assim, mantém a integridade dos itens de configuração produzidos no ciclo de desenvolvimento, além de controlar as mudanças sofridas pelos produtos de software (ENGHOLM, 2010).

Cada versão de sistema possui mudanças incorporadas, sendo que estas alterações devem ser gerenciadas para que não se perca o controle de quais mudanças são inseridas em cada versão lançada.

A adoção de um controle de mudanças justifica-se pelo fato de que, no processo de software, podem existir vários envolvidos, em consequência, diversas pessoas da equipe de desenvolvimento alteram ou criam linhas de código, gerando novos itens de configuração ou criando novas versões dos já existentes. E, sem um controle rigoroso destas alterações, certamente surgirão problemas na integração do projeto.

Pressmann (2005) indica que o gerenciamento de configuração e mudança dever ter como objetivo responder às seguintes perguntas:

a) O que mudou e quando mudou?

b) Por que mudou?

c) Quem fez a mudança?

d) Pode-se reproduzir esta mudança?

O GCS está presente nos principais guias de referência em boas práticas de desenvolvimento de software, com abrangência internacional, entre eles o *Capability Maturity Model Integration* (CMMI, 2010) e *SWEBOK* (2004).

O modelo CMMI consiste em um modelo de referência que contém práticas a serem adotadas para melhorar a qualidade dos processos e produto de software. Este modelo está de acordo com a norma *ISO/IEC* 12207:2008 (ISO/IEC, 2015), que consiste em uma norma internacional que possui detalhes dos processos, atividades e tarefas que envolvem o desenvolvimento, fornecimento, operação e manutenção de software.

A norma *ISO/IEC* 12207:2008 estabelece, entre outros processos, a GC como processo de apoio ao ciclo de vida de software, tendo como objetivo a identificação dos itens de software, incluindo-se atividades de controle de armazenameto, liberação, manutenção, distribuição e modificação de cada um dos itens de software que são mais comumente chamados de Itens de Configuração (ICs).

Os objetivos do Capability Maturity Model Integration (CMMI) é eliminar inconsistências e reduzir duplicações, estabelecer regras de construção uniformes e manter componentes comuns.

Este modelo está organizado de duas formas. Em uma das formas, as áreas de processo são representadas de maneira contínua e, na outra, as áreas de processos são divididas em níveis de maturidade. Na primeira forma, o GC está disposto no grupo 4, que possui processos de apoio; já, no *cmmi* por níveis, o GC é exigido no nível 2 de maturidade, conhecido como Nível Gerenciado (CMMI, 2010).

Segundo o CMMI (2010), o objetivo do GC é estabelecer e manter a integridade dos produtos de trabalho, usando a identificação de configuração, controle de configuração, status da contabilidade de configuração e de configuração de auditorias.

No cenário nacional, o GCS está presente no Guia de Melhoria de Processos do Software Brasileiro (MPS-BR, 2012), que se caracteriza em um modelo de referência, contendo guias de processos a serem seguidos com propósito de melhorar a produção de software brasileiro (MPS-BR, 2012). O guia orienta a adoção de processo de GC durante o ciclo de vida do projeto de software. Embora exista esta recomendação, o guia não

determina a forma como deve ser realizado o GCS.

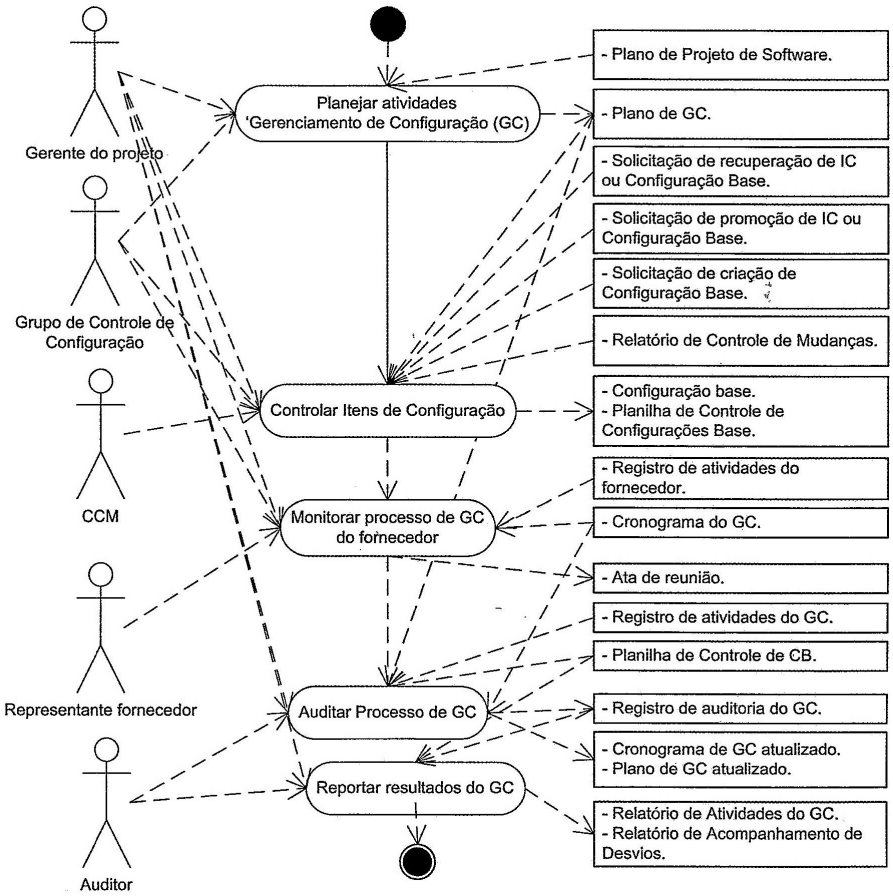
### 2.2.1 Atividades do gerenciamento de configuração

As normas, padrões e modelos existentes não possuem o mesmo fluxo de atividades que devem ser seguidas para garantir uma efetiva aplicação e benefícios do Gerenciamento de Configuração de Software.

Cada etapa do GCS possui procedimentos que devem ser adotados para garantir o seu objetivo principal, que é manter os itens de configuração íntegros, completos e consistentes, para que possam ser recuperados em qualquer momento do ciclo de vida do projeto.

Para facilitar a visualização da sequência de atividades necessárias para implantação de GCS, o fluxo geral do processo de gerenciamento de configuração apresentado por Engholm (2010) pode ser visualizado na Figura 3.

Figura 3 – Fluxo geral do processo de gerenciamento de configuração.




Fonte: Engholm (2010)

Como pode ser observado na Figura 3, o fluxo geral do processo de gerenciamento de configuração proposto por Engholm (2010) consiste em cinco processos, sendo que

cada processo possui atividades que devem ser realizadas. Algumas atividades produzem artefatos de saída que são utilizadas como artefatos de entrada em outras atividades.

O primeiro processo do fluxo geral de gerenciamento de configuração consiste em planejar as atividades do GC, tendo como objetivos definir um plano de GC a ser utilizado em todo o processo de software. Entre estas atividades, está a criação e disponibilização de repositório de Gerenciamento de Configuração. Neste repositório, estão depositados todos os itens de configuração colocados no GC, no qual podem ser acessados por todos os envolvidos no projeto.

No plano de GC, devem ser detalhadas as configurações de base do projeto de software que devem ser adicionados no gerenciamento e controle. As configurações de base são definidas como todos os itens de configuração que fazem parte do projeto, ou seja, todos os artefatos criados, alterados ou excluídos durante o ciclo de vida do software que serão monitorados.

Após a definição das configurações de base, a etapa seguinte consiste em controlar itens de configuração, tendo como objetivos criar e liberar configurações de base e recuperá-las para implementação de mudanças. Nesse caso, serve como artefato de saída a planilha de configuração de base que, no escopo de GC, é denominada de Baseline. Segundo Engholm (2010), esta atividade é executada pelo Gerente de Projeto e pela Comissão de Controle de Mudanças e (CCM). Esta comissão é composta por um grupo de pessoas envolvidas no processo de GC e tem como atribuição apenas auxiliar no controle de Itens de Configuração (ICs) e definição de configurações de base.

A terceira etapa consiste em monitorar o processo de GC do fornecedor, tendo como objetivo monitorar as tarefas alocadas ao fornecedor. Para controlar o andamento desta etapa, devem ser realizadas reuniões regulares para cumprimento do cronograma estabelecido junto ao fornecedor.

Já a quarta etapa é a Auditoria de Processo de GC, que consiste em realizar auditorias constantes no projeto, tendo como objetivo a verificação e validação dos processos de GC, garantindo que estejam de acordo com o Plano de GC.

E, por fim, a etapa de reportar resultados tem como objetivo disponibilizar o status do gerenciamento de configuração. Esta atividade permite o acompanhamento em alto nível do processo de GC, o que facilita a visualização de possíveis desvios e tomada de descisões para regularizá-los.

### 2.2.2 Papéis em gerenciamento de configuração

A identificação dos envolvidos nas atividades é de grande importância para atender os objetivos gerais do processo de GC. Como o GC está relacionado com todos os processos do ciclo de vida de um software, devem existir papéis bem definidos.

Segundo Molinari (2007), dentro do GC, um papel, comumente chamado de role, refere-se a quem pode fazer o quê, e esta definição permite que os demais envolvidos no processo de software visualizem, de forma clara, as suas responsabilidades e a quem se reportar em caso de necessidade.

O autor ainda complementa que, na definição dos papéis dos envolvidos com o processo de GC, deve-se levar em consideração a estrutura organizacional da empresa. Segundo Engholm (2010), nos projetos de desenvolvimento de software, deve-se eleger um grupo de pessoas que serão os responsáveis por controlar, versionar, armazenar e recuper os itens de configuração pelos projetos. Como esta tarefa depende de um controle rigoroso, recomenda-se o uso de ferramentas que auxiliam e automatizam as atividades de Gerenciamento de Configuração.

Dentre os principais tipos de roles no GC, Molinari (2007) define um como "Relacionados a Projetos". Esses roles tratam de papéis que estão geralmente conectados à implementação de um projeto. A seguir, são descritos cada papel e sua atribuição no GCS.

O Analista lida, diariamente, com atividades relacionadas ao GCS, e sua contribuição para o processo é na identificação dos itens de configuração relevantes, colocando estes itens em armazenamento após aprovados. O Designer/Projetista tem como atribuição projetar toda a arquitetura e mantém a documentação ao longo da vida do produto, produzindo novas versões ou alterações. Este role contribui no GCS através da identificação de itens de configuração relevantes relacionados ao design e colocação destes itens em armazenamento após aprovação.

Já o Programador executa todas as atividades de programação, produzindo códigos-fonte. Sua contribuição no GCS é com identificação de Itens de Configuração relevantes gerados em suas atividades, como, por exemplo, os códigos-fonte, colocando-os em armazenamento após aprovação.

O Integrador tem função de integrar códigos-fonte em maiores subsistemas de acordo com o objetivo do sistema. Também tem a função de reintegrar e recriar subsistemas para posterior reteste. A contribuição do integrador no GCS é com a identificação dos itens de configuração relevantes, como scripts de construção, subsistemas, etc; além da colocação destes itens de configuração em armazenamento depois de aprovados.

O role de Testador tem por função testar o sistema de acordo com o plano de teste. Utilizam versões apropriadas para teste e reteste,contribuindo para o GCS através da identificação de itens de configuração relevantes para os testes, tais como: planos de testes, descrições de testes, scripts de testes, dados de testes, etc.

Já o Gerente de projeto é responsável pelo projeto ou produto criado, tendo a responsabilidade de realizar o planejamento de GCS de acordo com os requerimentos

do projeto. As principais tarefas envolvidas no GCS são: produzir e atualizar o plano de GC de acordo com plano geral de projeto, identificação de papéis e distribuição de responsabilidades das atividades GCS, alocar recursos de GCS que sejam necessários e acompanhar se as atividades de GC estão sendo executadas conforme planejamento.

O role de Responsável pela Qualidade é atribuído à pessoa que tem como atribuição assegurar que as atividades da garantia da qualidade estejam de acordo com os requerimentos do produto ou do projeto. As principais tarefas envolvidas no GC são: estabelecer requerimentos de acordo com as atividades da GC e de seus resultados; acompanhar a qualidade das atividades definidas no Plano de GC e resultados dessas atividades; e criar relatórios e sumários de resultados armazenados relacionados à qualidade.

Já o Responsável pelo Contato junto ao Cliente tem por função garantir que a cooperação com o cliente seja satisfatória de acordo com o contrato.Ele deve entender as necessidades do cliente que tenham impacto na acgc. As principais tarefas são: produzir documentação relativa de GC para atender ao cliente. Receber e acompanhar eventos dos clientes que geram demandas de GC.

E, por fim, o Responsável pelo Contato junto ao Subcontrato tem como atribuição garantir a cooperação com os possíveis subcontratados. Além disso, deve garantir que as atividades de GC da subcontratada estejam de acordo com os requerimentos. As principais atividades deste papel são: definir requerimentos para a GC da subcontratada, manter contato entre o subcontratado e o projeto, acompanhar o subcontratado para garantir que os requisitos de GC estejam corretos, receber e executar a garantia de qualidade em subentregas e encaminhar registros de eventos e possíveis registros de mudanças para itens de configuração que estejam sob responsabilidade do subcontratado.

A definição de papéis em um processo de Gerenciamento de Configuração de Software formalizado pode variar de acordo com a estrutura da equipe de desenvolvimento de software. Alguns papéis definidos anteriormente não serão necessários dependendo do tipo de empresa ou instituição.

### 2.2.3 Ferramentas de apoio ao gerencimanto de configuração

O uso de ferramentas de GC tem como propósito principal automatizar as tarefas que envolvem grande esforço e ações repetitivas. Como menciona Molinari (2007), todas as ferramentas de GC realizam a automação até certo nível de atividades, poucas possuem a capacidade de suprir todas as necessidades da organização, pois existem muitas pessoas envolvidas em GC.

Para controle de versão, existem várias ferramentas disponibilizadas. Para exemplificar, podem-se citar duas mais comumente utilizadas, que são o *Subversion* ou SVN (SUBVERSION, 2015) e o Git (GIT, 2015). A ferramenta *Subversion* é uma ferramenta

de controle de versão livre/*open-source* sob licença GNU GPL (GPL, 2015). Esta ferramenta distribui seu repositório na forma de árvore, compreendendo, basicamente, um repositório centralizado, contendo um fluxo principal de onde são criados fluxo alternativos, chamados de *branches* (galhos).

Já o Git é um software livre, liberado sob licença GNU GPL (GPL, 2015), para controle de versão distribuído. Este software trouxe uma nova abordagem para sistemas de controle de versão, em que cada diretório de trabalho git é um repositório com todos os históricos, permitindo o uso do conceito de ramos para desenvolvimento.

Estes ramos são criados apartir de um projeto inicial, podendo ser desenvolvidas soluções independentes e, após conclusão, revisões são necessárias para unir novamente os ramos criados independentes. Esta ferramenta, ao contrário do SVN, não exige um respositório central, vários ramos podem assumir o papel de repositório central a partir do qual são criados novos ramos.

Tanto o SVN quanto o Git são ferramentas que automatizam tarefas que exigem grande esforço manual da equipe de desenvolvimento. Com isso, a equipe pode gastar mais tempo com planejamento do GC. Diante disso, a ferramenta escolhida deve ser a que atenda melhor às necessidades da equipe de desenvolvimento, utilizando-a de forma que se tire melhor proveito de suas funcionalidades, podendo, em alguns casos, utilizar uma combinação de ferramentas.

Para controlar mudanças de forma automatizada, existes várias ferramentas que podem ser utilizadas, entre elas, podem-se citar duas, que são o Trac (TRAC, 2016) e Mantis (MANTIS, 2016). O Trac é uma ferramenta livre/open-source sob licença GNU GPL (GPL, 2015) que tem como objetivo gerenciar bugs. A ferramenta possui funcionalidade que permite verificar o andamento de projetos, facilitando o seu acompanhamento e a sua gerência. Já o Mantis, que também é uma ferramenta livre/open-source sob licença GNU GPL (GPL, 2015), tem como principal função gerenciar defeitos de outros softwares. Assim como o Trac, o Mantis também possui funcionalidades que facilitam a gerência de projetos.

Atualmente, a CODEV utiliza a ferramenta Mantis na versão 1.2.18 para controlar as mudanças nos produtos de software e, para controle de versões, utiliza a ferramenta Subversion na versão 1.6.17.

### 2.2.4 Itens de configuração

Todo elemento criado no desenvolvimento de software ou necessário para esse desenvolvimento é considerado Item de Configuração (IC) (ENGHOLM, 2010).

O IC é o menor item de controle num processo de GC, podendo ser qualquer coisa (MOLINARI, 2007). A Tabela 1 apresenta alguns exemplos de ICs definidos por Molinari

(2007) e Engholm (2010).

Tabela 1 – Itens de configuração.

| **Fonte** | **Exemplos de itens de configuração** |
|---|---|
| Molinari (2007) | Aplicação corporativa;<br>Documento;<br>Parte de um documento;<br>Um executável;<br>Código fonte;<br>Características de Hardware. |
| Engholm (2010) | Código fonte e compilado do software desenvolvido;<br>Documentação do projeto;<br>Documentação de Requisitos;<br>Documentação de Especificação;<br>Sistema Operacional requerido;<br>Frameworks utilizados;<br>Banco de dados;<br>Scripts de Banco de dados;<br>Características de Hardware utilizada;<br>Qualquer item que possa sofrer mudanças e<br>que possa ser rastreado. |

Fonte: Elaboração própria.

Com isso, para que seja possível verificar o que sofrerá mudança, é necessário, primeiramente, identificar as atividades do processo de GC, identificar os papéis dos envolvidos e, posteriormente, apontar os artefatos que serão controlados e rastreados. Para isso, será modelado em Business Process Modeling Notation (BPMN) o processo de gerenciamento de configuração de software identificando as atividades, os envolvidos em cada atividade e ICs gerados e controlados pela CODEV.

### 2.2.5 Linhas de Base

Linhas de base ou *baseline* podem ser vistas como uma fotografia de todos os ICs utilizados em uma aplicação em determinado momento.

O benefício do uso de linhas de base é facilitar a possibilidade de criação e recuperação de diferentes versões de um produto de software (ENGHOLM, 2010). Com isso, pode-se retornar a um momento específico na linha de vida de um software e executá-lo exatamente como era naquele momento, com suas características de hardware e software.

## 2.3 BPMN

*Business Process Modeling Notation* (BPMN) é uma notação padrão que oferece a capacidade de compreender os procedimentos internos de negócios utilizando elementos básicos suficientes que permitem às organizações comunicar procedimentos internos (BPMN, 2015).

Segundo Braconi e Oliveira (2010), o propósito básico do *bpmn* é oferecer uma notação padrão para a modelagem de processos de negócio, de modo a superar as deficiências das outras técnicas de modelagem.

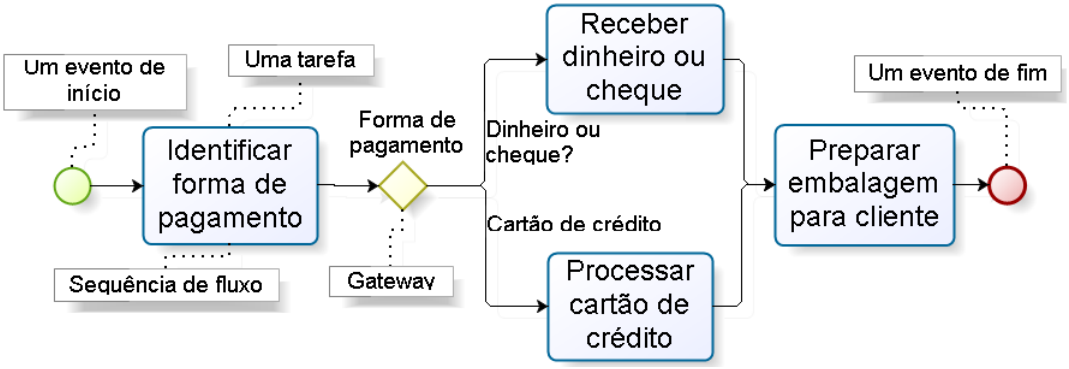
Com apenas alguns elementos básicos disponibilizados pela notação *bpmn*, é possível modelar os mais variados tipos de processos de negócio. Na Figura 4 apresentada por Braconi e Oliveira (2010), estão alguns destes elementos.

Como pode ser observado na Figura 4 o processo é composto por uma sequência de atividades ou tarefas com BPMN e permite a compreensão do fluxo do processo de pagamento de um produto qualquer, sem necessidade de um conhecimento prévio sobre modelagem de processo.

Para marcar o início da sequência, utilizam-se eventos de início, que são representados com círculo de borda simples. Conforme menciona Campos (2010), eventos provocam a ocorrência ou encerramento de uma tarefa ou atividade, evento de encerramento é representado com uma borda escura.

O elemento tarefa é um tipo de atividade que tem como propósito representar as ações que são executadas no processo de negócio.

Figura 4 – Exemplo de processo modelado utilizando BPMN.




Fonte: Braconi e Oliveira (2010)

Já o elemento *gateway*, representado por um losango, é utilizado para representar a exceção de um processo, identificando a ocorrência de um fluxo diferente do caminho natural do processo.

Além desses elementos, a notação BPMN permite a representação de objetos de dados que são basicamente artefatos gerados pelas tarefas. Segundo Campos (2010), objetos de dados podem se referir a documentos, a banco de dados, às tabelas nesses bancos de dados, ou mesmo, a campos de uma tabela.

Para este trabalho, a notação BPMN auxiliou na identificação de itens de configuração como objetos de dados gerados durante o fluxo de atividades do processo modelado. Sua utilização deu-se pelo fato de ser bastante utilizada e aceita para modelagem de processos de negócio, ser de fácil aprendizado e já ser utilizada na DTIC para modelagem de processos.

# 3 Trabalhos Relacionados

Esta seção apresenta trabalhos relacionados com a temática e o propósito deste trabalho. A pesquisa foi realizada em base de dados (principalmente na ACM *Digital Library*, e na *SCOPUS*) entre outros periódicos científicos. Foram utilizadas as palavras chave processo de gerenciamento de configuração, juntas e separadamente para consulta nas bases de dados. O retorno da consulta obteve-se poucos artigos relacionados ao tema do trabalho e não foram encontrados trabalhos publicados que utilizaram abordagem de elaboração do estado atual, do estado melhorado e do estado futuro do processo de GCS.

## 3.1 Gestão de configuração de software por Pilatti, Audy e Prikladnicki (2006)

No trabalho intitulado *Software Configuration Management over a Global Software Development Environment: Lessons Learned from a Case Study*, os autores apresentam os principais desafios de Gerenciamento de Configuração de Software (GCS) em um ambiente de desenvolvimento com equipes distribuídas globalmente.

O estudo de caso é de uma empresa multinacional de desenvolvimento de software com equipes distribuídas entre o Brasil, Índia e Rússia. Por motivos de incentivos fiscais ou disponibilidade de recursos no desenvolvimento de software, a organização decidiu distribuir suas equipes de desenvolvimento. Esta distribuição trouxe vários desafios, pois devem ser avaliados os impactos negativos, como por exemplo, diferenças culturais, de fuso horário e de comunicação.

Através das lições aprendidas em cada projeto, os autores identificaram alguns dos problemas e propuseram soluções, compartilhando-as dentro da organização.

Segundo os autores, em um dos projetos analisados, a equipe brasileira utilizou processos definidos no CMMI no qual possuía certificação no nível 2. Este projeto foi desenvolvido em conjunto com equipes dos Estados Unidos da América (EUA) e Índia, sendo que a falta de maturidade destas duas equipes com desenvolvimento distribuído foi o maior desafio. Para solucionar este problema, foi atribuído a um membro da equipe o papel de coordenador de gerenciamento de configuração, cujo papel ganhou cada vez mais espaço no projeto, tendo, entre outras atribuições, as atividades de executar a integração e validação do código.

Nos projetos distribuídos em que foram utilizados dois ambientes diferentes de GCS, foi requerido um esforço maior do coordenador de gerenciamento de configuração,

sendo exigido entre 3 a 4 horas de trabalho semanal para executar a integração e validação do código. Do contrário, a decisão de utilizar apenas um ambiente de GCS para outro projeto foi crucial para o seu sucesso . Alguns erros durante a integração foram encontrados, mas representavam apenas 3% do número total de erros encontrados no projeto. Apesar da utilização de apenas uma instância de GCS, foi atribuído a um membro da equipe o papel de controlador do sistema de GCS.

No terceiro projeto analisado pelo estudo de caso, a equipe de desenvolvimento decidiu utilizar dois controles de GCS. Embora os membros das equipes fossem desenvolvedores experientes, a falta de definição de responsabilidades e comunicação do projeto às partes interessadas foi a principal identificação do alto índice de falhas durante o ciclo de vida do projeto, ocasionando atrasos nas entregas.

Já para o último projeto analisado, o maior desafio foi lidar com as dependências entre os módulos distribuídos entre às equipes brasileiras e norte-americanas. Em alguns casos, o atraso na entrega dos módulos ocasiou ociosidade para alguns desenvolvedores, que dependiam das entregas para seguir o desenvolvimento de outros módulos.

Após análise dos quatro projetos, os autores destacam que administrar a configuração do contexto de desenvolvimento de software distribuído pode ser uma tarefa árdua, se os processo não estão bem definidos e as equipes não estiverem preparadas para trabalhar neste cenário.

Percebeu-se que todo o trabalho que envolve a CMMI nível 2 pode minimizar os problemas encontrados no cenário distribuído. Os processo de GCS baseados no CMMI trouxe excelentes resultados, contribuindo na padronização do trabalho entre as equipes, auxiliando na definição de processos do GCS. Além disso, as lições aprendidas em projetos anteriores são aplicadas em projetos recentes, permitindo um ganho de qualidade dos novos projetos.

Os autores concluem que o GCS tem um papel fundamental no processo de desenvolvimento de software. Surgem várias dificuldades quando se tenta sincronizar as atividades de GCS entre as equipes, e estas dificuldades aumentam quando as equipes estão distribuídas. Considerando o aumento na adoção de desenvolvimento de software global, existem poucos estudos sobre o impacto das tarefas no processo de GCS.

O estudo permitiu uma melhor compreensão da área de desenvolvimento de software global, a relação entre a equipe de projeto e os usuários com relação ao gerenciamento de configuração de software.

## 3.2 Controle de versão por Junqueira, Bittar e Fortes (2008)

No trabalho realizado pelos autores, denominado de *A fine-grained and flexible version control for software*, é apresentado um modelo e sistema para controle de versão flexível e refinado, dando suporte ao Gerenciamento de Configuração de Software (GCS). Os autores destacam que as ferramentas utilizadas para controle de versão, até o momento da publicação do artigo, tinham sérias limitações em garantir um controle sob a rastreabilidade dos arquivos adicionados ao controle de versão.

Os principais problemas na utilização do sistema de controle de versões é durante a fusão e envio das alterações realizadas pelos usuários. Estes devem possuir conhecimentos dos principais comandos e quando utilizá-los na ferramenta de controle de versão. Cada usuário deve resolver os conflitos em sua cópia local antes de submeter as alterações ao sistema.

Os autores apresentam uma breve evolução das ferramentas utilizadas para controle de versão, passando um breve histórico desde a primeira ferramenta chamada de SCCS criada em 1972 por Marc Rochkind. No qual inspirou o lançamento do projeto RCS, que incorporou algumas melhorias, sendo lançado dez anos após o SCCS.

Quatro anos mais tarde, ocorreram mudanças nos ambientes de desenvolvimento. Com isso, foi lançado o CVS, que trouxe o conceito de repositórios para o sistema de controle de versão. Esta ferramenta estabeleceu-se como padrão por duas décadas. Depois disso, somente em 2002, surgiram inovações nos sistemas de controle de versões, com o lançamento de sistemas de controle de versão distribuído.

Em 2002, foi lançado o Bitkeeper, que foi utilizado para desenvolvimento do Kernel do Linux e serviu de inspiração para o lançamento do Mercurial.

Com o trabalho elaborado, os autores fizeram uma breve revisão das ferramentas utilizadas para controle de versão e apresentaram a ferramenta Phoca, que tem como objetivo fornecer controle de versão refinado e flexível. O artigo não faz referência de qual abordagem é melhor entre as apresentadas, apenas mostra o funcionamento da ferramenta Phoca para uma aplicação qualquer, permitindo ao leitor fazer a escolha da abordagem que melhor se adeque à equipe de desenvolvimento.

## 3.3 Gestão de configuração de software apresentado por Premraj et al. (2011)

O estudo realizado pelos autores, denominado de *To Branch or Not to Branch?*, apresenta, na prática, como é aplicada a Software configuration management (SCM), ou gestão de configuração de software (GCS) em uma indústria fabricante de impressoras.

Para os autores, os estudos mais recentes centravam-se apenas na utilização de ferramentas, modelos de GCS, estratégias, padrões ou melhores práticas, sendo que havia poucos estudos na indústria sobre como uma organização realmente utiliza a ramificação e fusão.

Em termos de ferramentas, os últimos estudos evidenciam que vários modelos estão em ampla utilização, sendo que as mais usadas são o CVS, SVN, Git, Bazaar, Mercurial e Synergy. Estas ferramentas são indispensáveis em ambientes de desenvolvimento, em que várias pessoas acessam e modificam o mesmo arquivo, em algumas situações, alterando simultaneamente a mesma linha.

Estas ferramentas permitem realizar ações repetitivas que exigiriam grande esforço manual, tendo em vista que, conforme o estudo de caso apresentado pelos autores, uma indústria tem cerca de 1.790 ramos criados em um período de 18 meses. A ferramenta para GCS adotada pela empresa é a *IBM Rational Synergy*.

O trabalho faz uma investigação do motivo pelo qual são criados os ramos e qual o custo da fusão destes ramos. Além disso, em uma equipe de desenvolvimento ágil, é comum a re-edição de um arquivo e, inclusive, a edição simultânea de um arquivo. Com isso, o trabalho se propõe a pesquisar uma boa prática de fusão para equilibrar o custo com a necessidade da ramificação nestas equipes.

A empresa adota um método de trabalho que permite aos desenvolvedores planejarem suas próprias tarefas. Este ambiente é propício à utilização de ramificação para desenvolvimento dos projetos de software.

No estudo, os autores Premraj et al. (2011) tentaram compreender as implicações do uso de ramificação com o custo de fundir as alterações entre os ramos. Com o uso da ferramenta, a ação de fusão levou pouco tempo para ser executada, sendo que o custo desta atividade teve resultados dentro do esperado. Para atividades desenvolvidas pelos arquitetos de software, cujas alterações são realizadas em vários arquivos pertencentes a diversos ramos, as melhores práticas de GCS podem auxiliar este grupo de pessoas durante as fusões dos ramos.

Conforme estudo, o uso combinado de uma ferramenta de GCS e as melhores práticas em GCS não são suficientes para compartilhar arquivos em um ambiente de desenvolvimento ágil. O contéudo de arquivos compartilhados deve ser alinhado com a responsabilidade dos proprietários principais dos arquivos. Dessa forma, podem ser minimizados os conflitos na fusão que emerge entre as ramificações, atividade que demanda mais tempo para ser executada.

## 3.4 Gestão de configuração por Bendix e Pendleton (2011)

No trabalho intitulado de *The Role of Configuration Management in Outsourcing and Distributed Development*, os autores reforçam que o desenvolvimento de software distribuído tem se tornado cada vez mais comum. As vantagens desta nova abordagem é a possibilidade de terceirizar os recursos necessários para conclusão dos projetos, mas isso traz desafios ainda maiores.

Entre os riscos apontados pelos autores, está a probabilidade de perder o controle sobre os grupos remotos de pessoas. E, para minimizar os impactos dos riscos neste tipo de desenvolvimento, o Gerenciamento de Configuração de Software (GCS) permite coordenar estas equipes e atividades durante o ciclo de vida do projeto.

Outro fator determinante para sucesso neste tipo de equipe, segundo os autores é simplesmente aplicando conceitos bem conhecidos e princípios de gestão de configuração em conjunto com outras atividades.

## 3.5 Uso de CMMI por Staples e Niazi (2010)

O estudo realizado pelos autores, denominado de *Two Case Studies on Small Enterprise Motivation and Readiness for CMMI*, apresenta dois estudos de casos sobre a motivação do uso de CMMI em pequenas empresas.

O CMMI pode ser utilizado para melhoria do processo de software e algumas grandes organizações têm alcançado elevados benefícios com a utilização do CMMI. O gerenciamento de configuração está presente e exigido no nível 2 de maturidade. No entando, o CMMI é pensado como uma tarefa difícil de ser realizada em pequenas e médias empresas. Devido ao seu grau elevado de complexidade, exige grande esforço e dedicação por parte dos envolvidos, podendo ser um processo demorado que tem como consequência o aumento de custos para este tipo de empresa.

As principais razões para implantação do CMMI é para melhorar a qualidade de software, redução de custos e redução de prazos na entrega de produtos de software. E foram estas as principais razões para as empresas objeto do estudo de caso decidirem adotar o CMMI nos processos internos. Apesar do custo elevado para sua implantação, o CMMI trará benefícios a médio e longo prazo.

Embora houvesse razões favoráveis à implantação, ambas as empresas estavam preocupadas com os recursos necessários para efetivar o CMMI. Além disso, um dos questionamentos é que o CMMI poderia exigir que alguns funcionários tivessem que desempenhar papéis diferentes dentro da empresa devido ao número reduzido de colaboradores neste tipo de empresa.

## 3.6 Melhoria de Processo por Santos (2013)

O trabalho elaborado pelo autor, com título de Proposta de Melhoria do Processo de Contratação de Serviços de TI e da Gestão dos contratos na Administração Pública Federal baseou-se numa análise exploratória com o objetivo de analisar e mapear os processos envolvidos na contratação de serviços de Tecnologia da Informação (TI) na Administração Pública Federal (APF), utilizando a notação BPMN para modelagem destes processos. O autor destaca a importância da análise e mapeamento dos processos para as organizações. Esta oportunidade permite uma visão do fluxo e sequência das atividades, duração do ciclo, pessoas envolvidas, relações e dependências existentes no processo.

Para o autor, os maiores benefícios do mapeamento de processos são: redução de custos, redução de tempo de execução, melhoria da qualidade, aumento da produtividade, maior foco na satisfação dos clientes internos e externos, agilidade no gerenciamento de mudanças e maior compreensão da organização.

A proposta de mapeamento e modelagem de processos, segundo o autor, não é uma atividade voltada apenas para um perfil específico de organização, deve ser implantada por todo tipo de organização, independente do ramo de atividade, tamanho e participação no mercado.

O trabalho desenvolvido no local de estudo de caso consistiu em mapear os processos atuais envolvidos para contratação de serviços de TI e da gestão de contratos na APF. Na sequência, o autor realizou a análise do modelo AS IS (atual), o qual resultou na modelagem do processo atual. Com isso, observou que alguns processos poderiam ser melhorados. O autor destaca que a formalização do AS IS contribuiu para identificação de melhorias.

## 3.7 Gestão de Processos de Negócio por Flora e Tolfo (2016)

No trabalho com título de A Gestão de Processos de Negócio como Ferramenta de Apoio na Gestão da Segurança da Informação, os autores apresentam um relato de experiência com o uso de gestão de processos de negócio com foco nos processos de segurança da informação.

O trabalho consiste em um estudo de caso na Coordenadoria de Segurança da Informação (CSI) subordinada ao Núcleo de Tecnologia da Informação e Comunicação (NTIC) da UNIPAMPA, onde foram mapeados e melhorados processos. Utilizando a notação BPMN, um dos processos modelados foi o processo de liberação de IP público

Segundo os autores, a modelagem do estado AS IS permitiu observar a forma como ele estava ocorrendo e auxiliou na análise de melhorias. O AS IS refere-se ao estado atual do processo, exatamente como ele se encontra no momento utilizado para modelagem.

Estas melhorias foram acrescentadas no estado TO BE (estado futuro) do processo, além disso, para modelagem do estado TO BE, os autores levaram em consideração fatores técnicos e organizacionais. O TO BE refere-se ao estado futuro do processo.

Para melhor visualização e auxiliar na modelagem do estado TO BE do processo objeto do estudo de caso, os autores elaboraram um fluxo textual contendo, de forma sequencial, as atividades envolvidas neste processo, incluindo as melhorias identificadas, que serviu de subsídio para modelagem do estado TO BE.

Após a conclusão da modelagem da versão TO BE do processo de liberação de IP público e finalizada as etapas de discussões e melhorias, o processo foi incorporado aos processos de trabalho da CSI.

Como conclusão, os autores afirmam que a perspectiva de processos de negócio auxilia na gestão de segurança da informação. A formalização do AS IS permite aos gestores e equipes revisarem as atividades envolvidas e os recursos que estão sendo utilizados. Com a elaboração do TO BE destacam que foi possível identificar melhorias, planejar o gerenciamento da segurança da informação, assim como formalizar e comunicar este planejamento por meio da modelagem de processos.

A utilização da notação BPMN mostrou-se adequada para modelagem de processos relacionados à gestão da segurança da informação. Os autores ainda relatam que esta notação permite que processos sejam planejados pelos responsáveis por implantar a gestão de segurança da informação e comunicar aos usuários dos serviços que necessitem de segurança da informação.

## 3.8 Melhoria de Processo por Fiorenza, Della Flora e Tolfo (2016)

O trabalho elaborado pelos autores, com título de Melhoria no Processo de Testes de Restauração, aborda um relato de experiência baseado na melhoria do processo de testes de restauração na CSI do NTIC. Para isso, os autores realizaram o mapeamento do AS IS, o mesmo que estado atual e modelagem do processo utilizando a notação BPMN.

O processo analisado é o de *backup* de dados, que, segundo os autores, é uma das etapas mais importantes quando se fala em desastre e recuperação de dados. Este processo envolve, entre outras atividades, a cópia segura dos dados, recuperação destes dados armazenados em local seguro, restauração de serviços em caso de perda de dados nos ambientes de produção.

Com base na modelagem do AS IS, os autores observaram que as atividades envolvidas eram insuficientes para atender às necessidades do NTIC, tendo em vista que alguns incidentes recentes mostraram que os backups não continham todos os dados necessários para reestabelecer os serviços.

A partir da modelagem do AS IS, os autores relizaram o refinamento do modelo e identificaram novas atividades que deveriam ser incluídas no TO BE do processo de backup da CSI. O TO BE refere-se ao estado futuro do processo.

Com esta nova configuração, a CSI passou a ter maior envolvimento em todas as etapas do processo, esta melhoria foi fundamental para garantir maior segurança para a equipe da CSI e para o administrador do serviço que necessitar de restauração de dados.

A modelagem do processo de testes de restauração, segundo os autores, proporcionou mitigar os problemas enfrentados até então com relação ao testes de restauração, garantindo que os dados armazenados em *backup* sejam fiéis aos dados em produção nos servidores do NTIC.

Aliado a isso, o uso da notação BPMN facilitou a compreensão e visualização de todos os envolvidos na melhoria, além de auxiliar a identificação de falhas e otimizar o processo.

# 4 Metodologia

Nesta seção, será descrita a metodologia utilizada para elaboração da pesquisa, incluindo quais foram os meios utilizados para construção do estado atual, análise e discussões, identificação de melhorias e modelagem de processo.

Este trabalho consiste em uma pesquisa exploratória que tem por finalidade verificar o maior número de informações sobre o problema a fim de torná-lo mais explícito (GIL, 2010). Nesta pesquisa, foram utilizados livros, artigos e teses que contêm textos processados pelos autores.

Além da consulta em livros, a pesquisa foi realizada em base de dados (principalmente na ACM *Digital Library*, e na *SCOPUS*) entre outros periódicos científicos. Para a consulta na base de dados, foram utilizadas as palavras-chave processo de gerenciamento de configuração, juntas e separadamente. Salienta-se que o retorno da consulta obteve poucos artigos relacionados ao tema do trabalho. Não foram encontrados trabalhos publicados que utilizaram abordagem de elaboração do estado atual, do estado melhorado e do estado futuro do processo de GCS.

Quanto à técnica empregada na elaboração da pesquisa, foi utilizado o estudo de caso que, segundo Bertucci (2011), caracteriza-se na identificação de um problema específico de uma organização e análise deste problema em profundidade.

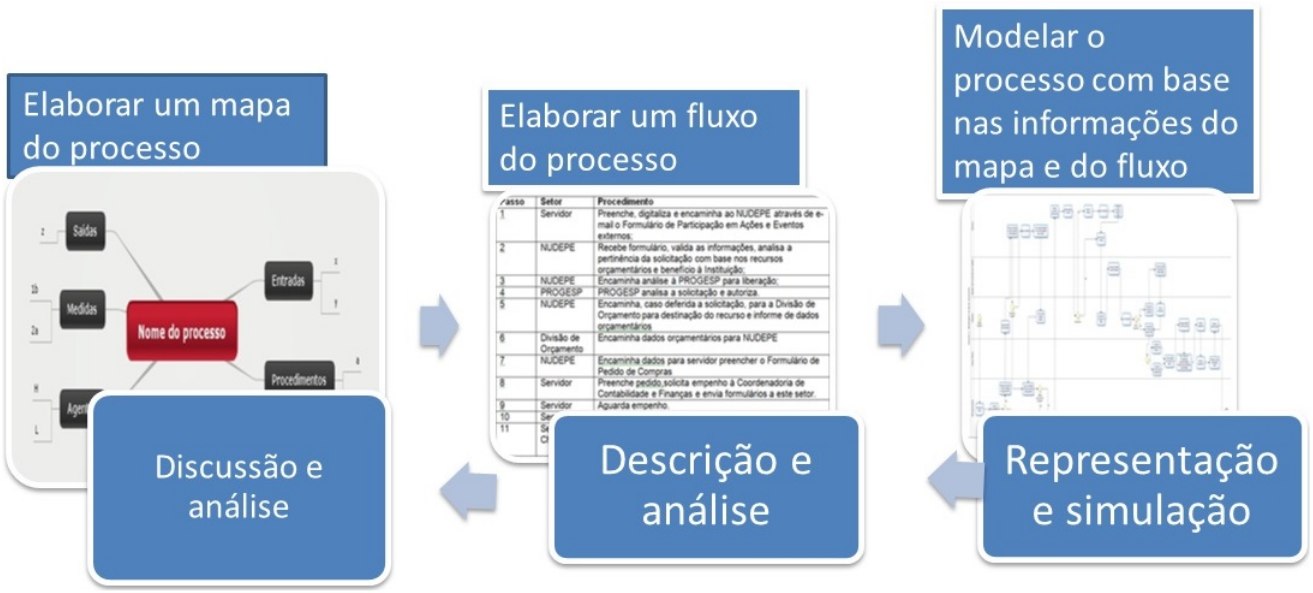
A organização escolhida como objeto de estudo para esta pesquisa foi a DTIC. A coleta de dados foi realizada por meio de duas técnicas: entrevistas informais e observações diretas. As entrevistas tiveram por finalidade verificar a perspectiva da equipe da CODEV sobre as atividades que envolvem o processo de GC. Por outro lado, as observações no local de estudo de caso tiveram como objetivo observar minuciosamente como o processo empírico ocorria, identificando atividades, artefados gerados e definindo o papel de cada envolvido.

Para melhor visualização e entendimento do problema, foi adotada uma abordagem para mapeamento e modelagem do processo proposta por Tolfo (2014), a Figura 5 apresenta as etapas da abordagem utilizada.

Como pode ser observado na Figura 5, esta abordagem possui três etapas: a primeira é a elaboração do mapa mental (BUZAN, 2005) do processo; na sequência, é elaborado o fluxo textual do processo; e a terceira consiste na modelagem do estado atual do processo. Após o término das três etapas desta abordagem, os artefatos passam por análise e discussões para refinamento.

Após o processo de investigação, para analisar e identificar melhorias, primeira-

Figura 5 – Abordagem para modelagem do processo.




Fonte: Tolfo (2014)

mente, foi modelado o AS IS do processo de GCS, utilizando a notação *Business Process Model and Notation* BPMN (BPMN, 2015). O AS IS refere-se ao estado atual do processo, exatamente como ele se encontra. Após a etapa de análise e discussões do AS IS, foi modelado o estado SHOULD BE do processo de GCS. O SHOULD BE refere-se ao estado desejado, como é idealizado o GCS.

O SHOULD BE foi analisado e discutido com a equipe da CODEV para obter como resultado o TO BE do processo de GCS. Já o TO BE refere-se ao estado futuro do processo de GCS.

A notação BPMN, utilizada para modelagem, segundo Braconi e Oliveira (2010), tem como propósito básico oferecer uma notação padrão para modelagem de processo de negócio. A utilização da notação BPMN, deu-se pelo fato de ser bastante utilizada e aceita para modelagem de processos de negócio, além de ser adotada pela DTIC para modelagens de processos.

Com a utilização da abordagem proposta e apresentada na Figura 5 para modelagem de processo, foi possível visualizar, de forma clara, o processo atual de GCS da DTIC, permitindo identificar melhorias e modelar o estado TO BE do processo de GCS adequado à estrutura do setor.

# 5 Desenvolvimento

Este capítulo apresenta as etapas realizadas nesta pesquisa a fim de atingir os objetivos do trabalho. A seção 5.1 apresenta a construção do estado atual do processo de GC da CODEV. Na seção 5.2 são descritas as melhorias identificadas no estado atual e modelagem do estado melhorado. Já a seção 5.3 apresenta os resultados das discussões do estado melhorado. E, por fim, a seção 5.4 apresenta os resultados do estado futuro do processo.

## 5.1 Mapeamento e construção do AS IS

Antes de iniciar a modelagem do processo de GC da CODEV, foi preciso entender o processo de software utilizado pelo setor.

Seguindo o fluxo de etapas da abordagem definida por Tolfo (2014), apresentada na Seção Metodologia, foram realizadas entrevistas informais e observações no local do estudo de caso, com a finalidade de coletar o maior número possível de informações.

Após entrevistas informais e observações, foi verificado que o setor possui um modelo de tarefas contendo a descrição de atividades e identificação de responsáveis, o que pode ser consultado no ANEXO A - Modelo de Tarefas. As atividades e responsáveis estão identificadas e disponíveis na forma de planilha no site da DTIC. Estas informações foram importantes para realizar o primeiro mapeamento e modelagem do processo de software utilizando a abordagem adotada.

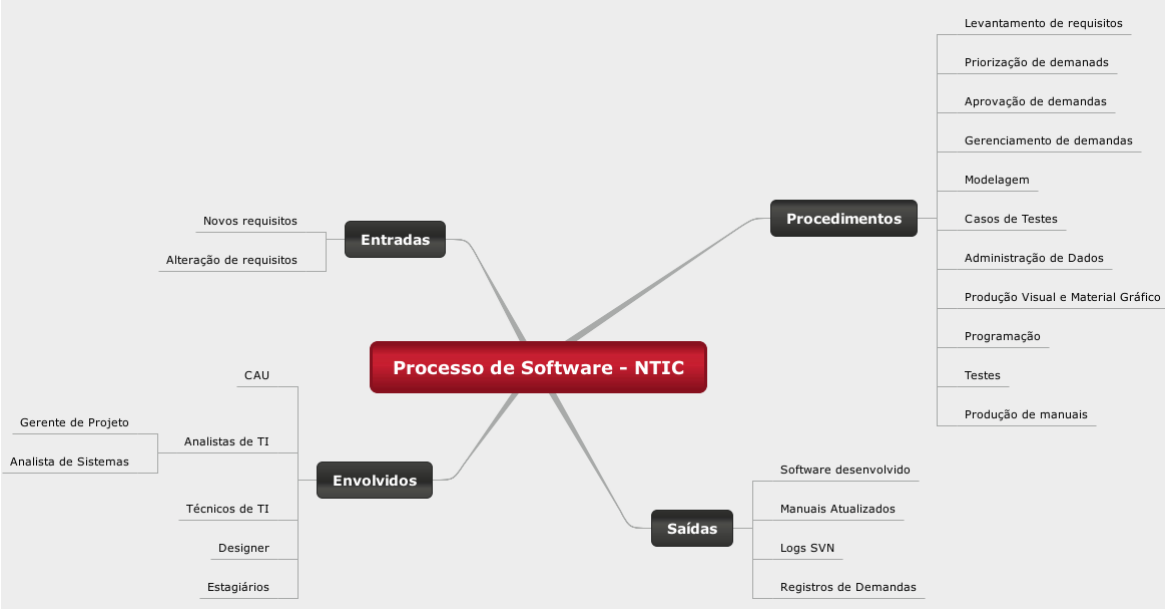
- **Primeira etapa: elaboração do mapa mental.**

Nesta etapa, foi elaborado o mapa mental, no qual podem-se observar as principais atividades, os envolvidos, as entradas e saídas presentes no processo de software adotado pela CODEV, além dos procedimentos adotados para realizar as atividades. A Figura 6 apresenta o mapa mental do processo de GC da CODEV.

Como pode ser observado na Figura 6, o mapa mental representa o processo de software da CODEV, contendo elementos de entrada, identificação dos envolvidos no processo, os procedimentos envolvidos e as saídas do processo. A elaboração do mapa mental auxiliou no entendimento do processo atual de software e identificação inicial de artefatos gerados durante o processo.

Para modelagem do mapa mental, foi utilizada a ferramenta web (MINDOMO, 2015), que permite a modelagem de mapa mental sem a necessidade de instalar o aplicativo

Figura 6 – Mapa mental do processo de software da DTIC.




Fonte: Elaboração própria

no computador, contendo vários recursos e de fácil utilização.

- **Segunda etapa: elaboração do fluxo do processo.**

Após gerado o mapa mental na etapa anterior, iniciou-se a etapa de elaboração do fluxo textual do processo atual. Nesta etapa é descrito cada passo, setor e atividade envolvida no fluxo do processo.

O fluxo textual permitiu organizar, de forma sequencial, as atividades realizadas no processo de software da CODEV e respectivos envolvidos em cada atividade. Além disso, também foi possível identificar os artefatos gerados em cada atividade. Esta forma de representação facilitou o entendimento do estado atual do processo de software da CODEV servindo de fonte de informação inicial para identificação dos artefatos adicionados ao GCS do setor.

A Tabela 2 apresenta o fluxo textual elaborado.

Tabela 2 – Fluxo atual do processo de software

| Passo | Responsável | Procedimento |
|---|---|---|
| **1** | Coordenação de Apoio ao Usuário (CAU) | Registrar requisitos dos novos projetos ou manutenção dos já existentes. |
| **2** | Gerente de Projeto | Elaborar Casos de Uso |
| **3** | Designer | Refinar protótipo de telas. |
| **4** | CAU | Valida casos de uso e protótipo de telas. |
| **5** | Gerente de Projeto | Cria cronograma. |
| **6** | Banco de Dados | Elaborar diagrama Entidade Relacionamento (ER). |
| **7** | Banco de Dados | Criar Tabelas do Banco de Dados |
| **8** | Analista de Tecnologia da Informação (ATI) | Elaborar diagramas de sequência |
| **9** | ATI | Elaborar diagramas de Classes |
| **10** | ATI ou Testador | Descrever casos de testes. |
| **11** | Programador | Iniciar programação. |
| **12** | ATI ou Testador | Inspecionar código. |
| **13** | ATI ou Testador | Realizar testes de aceitação. |
| **14** | Testador ou Programador | Produzir manual. |
| **15** | Gerente de Projeto | Revisar projeto. |
| **16** | CAU | Validar desenvolvimento. |
| **17** | CAU | Autorizar liberação para ambiente produção. |
| **18** | Banco de Dados | Executar script em ambiente produção. |
| **19** | ATI | Exportar pacote de ambiente produção. |

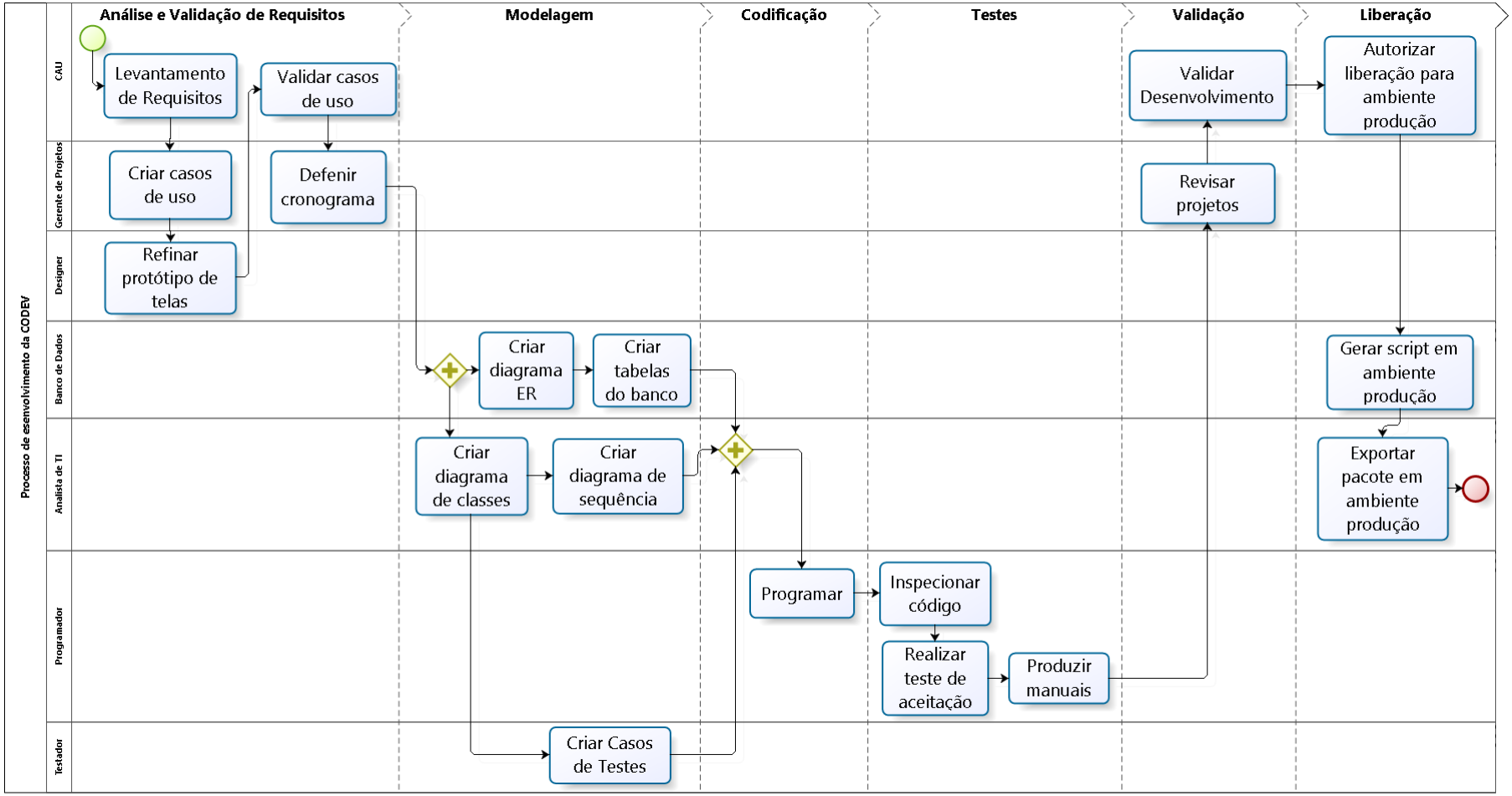
Fonte: Elaboração própria.

- **Terceira etapa: elaboração da modelagem do estado atual.**

Nesta etapa, utilizando o mapa mental e fluxo textual de processo elaborados nas etapas anteriores, passou-se para a elaboração da modelagem AS IS, o mesmo que estado atual, fazendo uso da notação BPMN para representar as atividades, os envolvidos e o

fluxo da sequência de atividades. O resultado da modelagem do estado atual do processo de software pode ser visualizado na Figura 7.

Figura 7 – AS IS do processo de desenvolvimento de software da CODEV.




Fonte: Elaboração própria

Para construção do modelo do estado atual de processo apresentada na Figura 7, foi utilizada a ferramenta Bizagi (BIZAGI, 2015), no qual a escolha deu-se pelo fato de ser uma ferramenta de fácil aprendizado e possuindo vasta documentação e tutoriais.

Após encerramento do primeiro ciclo da abordagem utilizada, que consiste na elaboração do mapa mental, fluxo do processo e modelagem do estado atual, iniciou-se o ciclo de discussões e análise, descrição e análise e representação e simulação.

Na fase de análise e refinamento, não foi necessária a utilização do mapa mental, apenas foi refinado o fluxo do processo de software da CODEV. Com a utilização da modelagem em BPMN, foram produzidos dois fluxos do processo atual de gerenciamento de configuração da CODEV, um fluxo para as atividades de controle de mudanças e outro para as atividades de controle de versão.

As atividades de gerenciamento de configuração da CODEV, embora empíricas, foram adotadas a partir de outubro de 2014. O fluxo textual do processo de controle de mudança pode ser visualizado na Tabela 3.

Tabela 3 – Fluxo AS IS do processo de controle de mudanças da CODEV

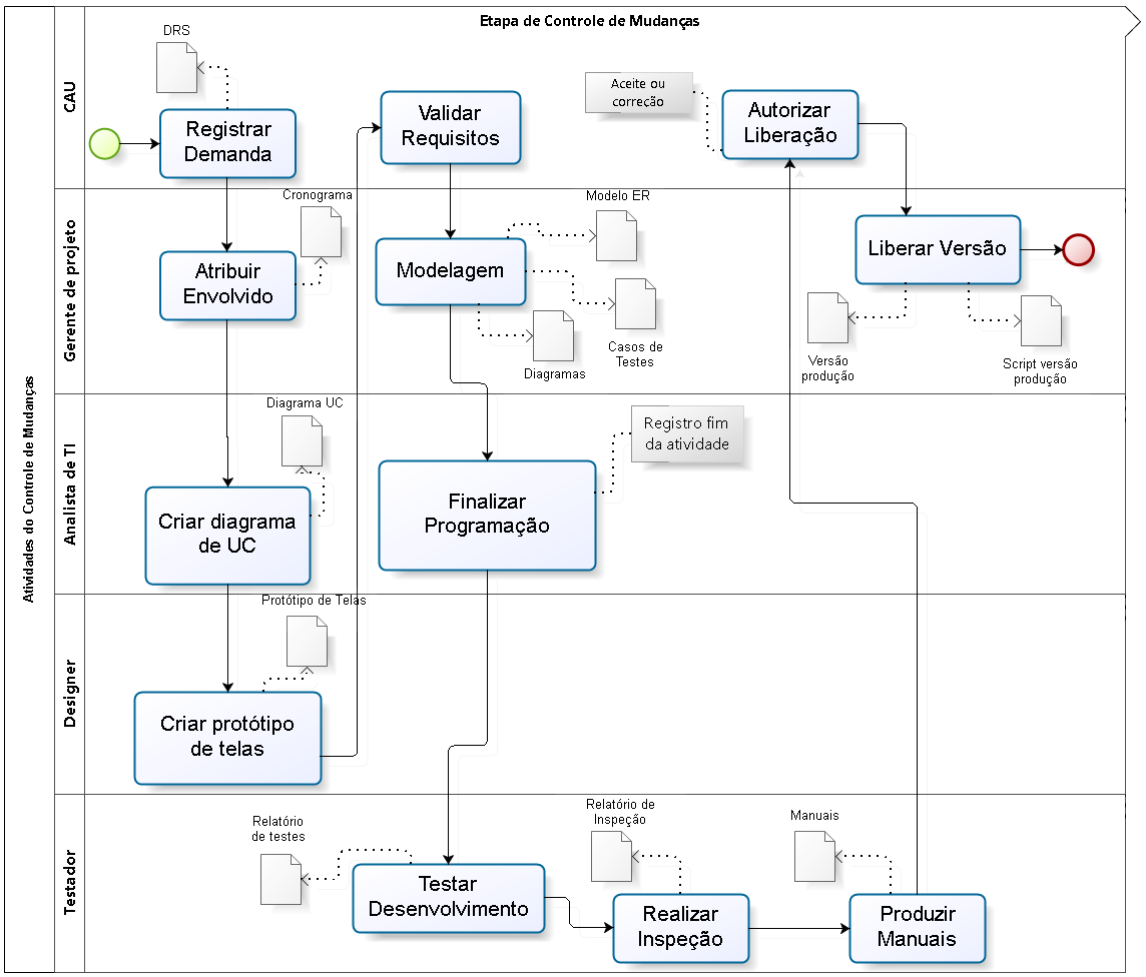
| Passo | Atividade | Responsável | Artefato | Ferramenta |
|---|---|---|---|---|
| **1** | Registrar demanda | CAU | **Entrada:**Não possui<br>**Saída:**Documento de Requisitos de Software (DRS) | Sistema de Controle de Mudança |
| **2** | Atribuir envolvido | Gerente de Projeto | **Entrada:**DRS<br>**Saída:**Cronograma. | Sistema de Controle de Mudança |
| **3** | Refinar requisitos | ATI e Designer | **Entrada:**Cronograma.<br>**Saída:**Diagrama de Casos de Uso (UC) e Protótipos consolidados. | Sistema de Controle de Mudança |
| **4** | Validar com Usuário | CAU e Usuário | **Entrada:**Diagrama de UC e Protótipos consolidados.<br>**Saída:** Aceitação | Sistema de Controle de Mudança |
| **5** | Modelagem | Gerente de projeto e ATI | **Entrada:**Diagrama de UC e Protótipos consolidados.<br>**Saída:** Modelo de dados, diagramas de classes, diagramas de sequências e casos de testes. | Sistema de Controle de Mudança |
| **6** | Finalizar programação | ATI | **Entrada:** Commits<br>**Saída:** Registro de fim da fase de programação, código compilado e scripts de banco da versão teste. | Sistema de Controle de Mudança |
| **7** | Testar programação | Testador ou ATI | **Entrada:** Código compilado e scripts de banco da versão teste, modelo de dados, diagramas de classes, diagramas de sequências e casos de testes.<br>**Saída:** Relatório de testes e resultado de teste. | Sistema de Controle de Mudança |
| **8** | Inspeção de código e modelos | Testador ou ATI | **Entrada:**Template do check-list, código compilado e scripts de banco versão teste.<br>**Saída:** Resultado de inspeção. | Sistema de Controle de Mudança |
| **9** | Produzir manuais | Testador e ATI | **Entrada:** Diagrama de UC e Protótipos consolidados.<br>**Saída:** Manual | Sistema de Controle de Mudança |

Fonte: Elaboração própria.

Para melhor visualização das atividades envolvidas no processo atual de controle de mudanças, as atividades foram identificadas baseando-se nas mesmas atividades do processo de software. Esta forma de identificação facilitou a validação com a equipe da CODEV.

Após a elaboração do fluxo textual do processo de controle de mudanças apresentado na Tabela 3, seguindo as etapas da abordagem adotada, iniciou-se a modelagem do estado atual do processo de controle de mudanças da CODEV, conforme pode ser visualizado na Figura 8.

Figura 8 – Modelagem AS IS do processo de controle de mudanças da CODEV.




Fonte: Elaboração própria

Além da atividade de controle de mudança, a CODEV também adotou, a partir de outubro de 2014, o processo de controle de versão. Para modelagem do processo de controle de versão, primeiramente, foi refinado o fluxo do processo de controle de mudança apresentado na Tabela 3. Para isso foi levado em consideração apenas os itens controlados e gerenciados pelo sistema de controle de versão. As atividades envolvidas no processo de controle de versão também foram identificadas baseando-se no processo de software da

CODEV, o qual facilitou a validação com a equipe da CODEV.

O fluxo textual do processo de controle de versão pode ver visualizado na Tabela 4.

Tabela 4 – Fluxo AS IS do processo de controle de versão da CODEV

| Passo | Atividade | Responsável | Artefato | Ferramenta |
|---|---|---|---|---|
| **1** | Atribuir envolvido | Gerente de Projeto | **Entrada:**DRS **Saída:**Não Possui | Sistema de Controle de Versão |
| **2** | Refinar requisitos | Analista de TI e Designer | **Entrada:**Não possui **Saída:**Diagramas de UC casos de uso e Protótipos consolidados. | Sistema de Controle de Versão |
| **3** | Modelagem | Gerente de Projeto e Analista de TI | **Saída:**Diagramas de UC Protótipos consolidados. **Saída:**Modelo de dados, diagramas de classes e de sequências. | Sistema de Controle de Versão |
| **4** | Iniciar Programação | Programador | **Entrada:**Base de desenvolvimento **Saída:** Commits | Sistema de Controle de Versão |
| **5** | Finalizar programação | Analista de TI ou Analista de TI | **Entrada:**Commits **Saída:** Código compilado e scripts de banco versão teste. | Sistema de Controle de Versão |
| **6** | Realizar inspeção | Testador ou Analista de TI | **Entrada:** Pacote da versão teste, modelo de dados, diagramas de classes e diagramas de sequências. **Saída:** Relatório de Inspeção | Sistema de Controle de Versão |
| **7** | Produzir manuais | Testador ou Analista de TI | **Entrada:** Diagrama de UC e Protótipos consolidados **Saída:** Manual | Sistema de Controle de Versão |
| **8** | Liberar Versão | Gerente de Projeto | **Entrada:**Versão de teste **Saída:** Versão de produção e scripts. | Sistema de Controle de Versão |

Fonte: Elaboração própria.

Como pode ser observado na Tabela 4, cada atividade é atribuída a um responsável. Além disso, foram apresentados artefatos de entrada e saída para cada atividade, o artefato gerado ou modificado em cada atividade. A partir disto, foi possível visualizar, de forma clara, os artefatos gerados ou modificados em cada atividade.

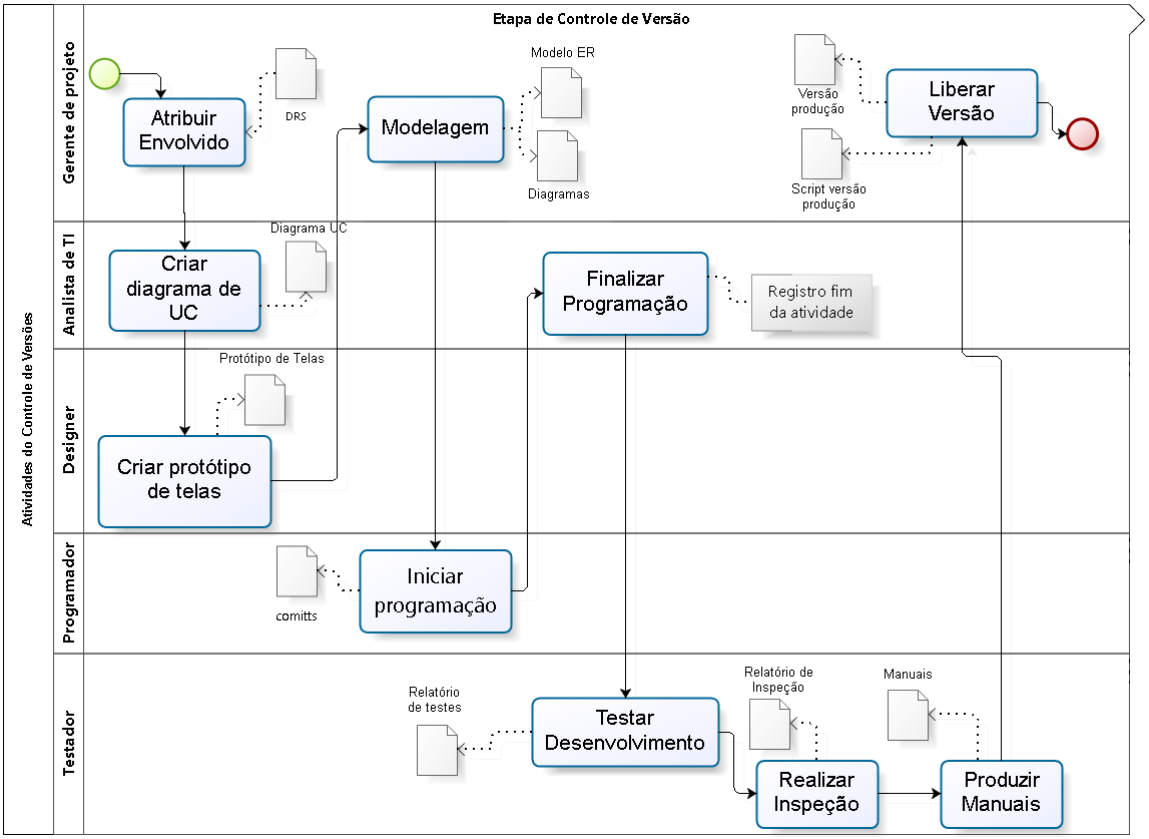
Após elaboração do fluxo textual do processo de controle de versão, inicou a etapa de representação do processo com a modelagem do estado atual deste processo. Para mo-

delagem do processo, foram utilizados os itens de configuração colocados sob controle de versão, as atividades do processo de software, itens de entrada e saída, além da ferramenta utilizada para tal controle.

Para representação das atividades envolvidas no processo de controle de versão, foram utilizadas as mesmas atividades do processo de software, o qual permitiu a melhor visualização das atividades do processo de software que interagiam com o processo de controle de versões. Esta representação auxiliou a validação com a equipe da CODEV para identificação dos itens de configuração controlados pelo sistema de controle de versões e os responsáveis por cada atividade.

A Figura 9 representa o modelo AS IS do processo de controle de versão modelado apartir da informações identificadas na Tabela 4.

Figura 9 – Modelagem AS IS do processo de controle de versão da CODEV.




Fonte: Elaboração própria

### 5.1.1 Revisões e refinamento do AS IS

Como parte da reestruturação da CODEV, em outubro de 2015, houve uma re-modelagem do fluxo das atividades envolvidas no processo de software do setor. O novo fluxo do processo de software pode ser consultado no ANEXO B - Processo de Software

Novos Requisitos e ANEXO C - Processo de Software Requisitos Não Previstos ou *Bugs*. As mudanças realizadas no processo de software da CODEV referem-se a reoganização de papéis no processo, sendo adicionados os papéis de Analista de Sistemas e Analista de Testes. Outra mudança foi a retirada do papel de Analista de TI.

Após as mudanças sofridas no processo de software da CODEV, foi necessário revisar e refinar as modelagens dos processos de controle de mudanças e controle de versão apresentados na seção 5.1.

Para identificar o novo fluxo de atividades dos processos de controle de mudanças e controle de versões, foi realizado o refinamento do fluxo textual destes dois processos. O fluxo textual do processo de controle de mudanças apresentado na Tabela 3 foi refinado, dando origem ao novo fluxo do processo de controle de mudanças, conforme pode ser observado na Tabela 5.

A Tabela 5 apresenta o fluxo textual do processo de controle de mudanças após refinamento.

Tabela 5 – Fluxo do processo de controle de mudanças da CODEV refinado

| **Passo** | **Atividade** | **Responsável** | **Artefato** | **Ferramenta** |
|---|---|---|---|---|
| **1** | Registrar solicitação de mudança | CAU | **Entrada:**Não possui **Saída:**Documento de Requisitos Funcionais (DRF) | Sistema de Controle de Mudança |
| **2** | Analisar solicitação de mudança | Gerente de Projeto | **Entrada:**DRF **Saída:** Nenhum | Sistema de Controle de Mudança |
| **3** | Registrar mudança no UC | Gerente de Projeto | **Entrada:**Nenhum **Saída:**Diagrama de UC | Sistema de Controle de Mudança |
| **4** | Registrar mudança na modelagem de telas | Designer | **Entrada:**DRF **Saída:**Protótipos refinados. | Sistema de Controle de Mudança |
| **5** | Aprovar mudança | CAU | **Entrada:**Nenhum **Saída:**Nenhum | Sistema de Controle de Mudança |
| **6** | Registrar mudança no modelo de dados | Analista de Dados | **Entrada:**Protótipos refinados e Diagrama UC **Saída:**Modelo ER | Sistema de Controle de Mudança |
| **7** | Registrar mudanças nos diagramas UML | Analista de Sistemas | **Entrada:**Protótipos refinados e Diagrama UC **Saída:**Diagramas de classes, de sequência e de estados | Sistema de Controle de Mudança |
| **8** | Registrar casos de testes | Analista de Testes | **Entrada:**Diagrama UC e Modelo ER **Saída:**Casos de testes | Sistema de Controle de Mudança |
| **9** | Registrar mudanças no código-fonte | Programador | **Entrada:**Diagramas de classes, de sequência e de estados **Saída:**Nenhum | Sistema de Controle de Mudança |
| **10** | Registrar verificação de mudanças | Analista de Sistemas | **Entrada:**Código-fonte versão teste **Saída:**Check-list validado | Sistema de Controle de Mudança |
| **11** | Registrar teste de usabilidade | Designer | **Entrada:**Código-fonte versão teste **Saída:**Check-list de usabilidade validado | Sistema de Controle de Mudança |
| **12** | Registrar teste de sistema | Analista de Testes | **Entrada:**Casos de testes **Saída:**Nenhum | Sistema de Controle de Mudança |
| **13** | Registrar mudanças no manual do usuário | Analista de Testes | **Entrada:**Teste validado, diagrama UC e protótipos de telas refinados **Saída:**Nenhum | Sistema de Controle de Mudança |
| **14** | Registrar validação dos requisitos | Analista de Testes | **Entrada:**Manual do usuário e Sistema versão teste **Saída:**Nenhum | Sistema de Controle de Mudança |
| **15** | Registrar liberação | CAU | **Entrada:**Nenhum **Saída:**Nenhum | Sistema de Controle de Mudança |

Fonte: Elaboração própria.

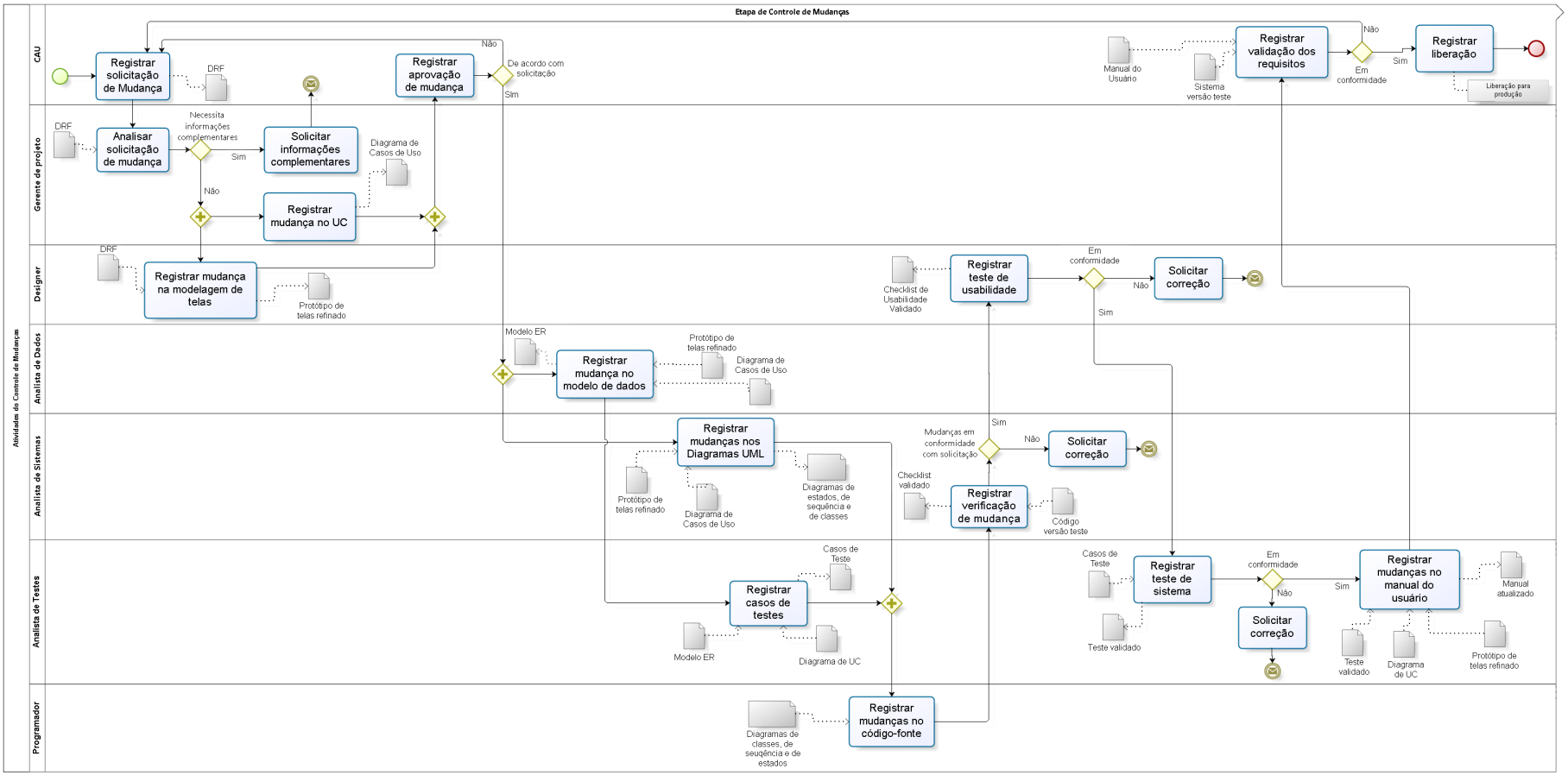
Nesta nova representação do fluxo textual do processo de controle de mudanças, as atividades identificadas são apenas as que têm interação com o Mantis, que é a ferramenta de controle de mudanças utilizada pela CODEV.

A seguir, são descritas as alterações no fluxo do processo de controle de mudanças após o refinamento. A atividade de Criar Diagrama de UC foi atribuída ao Gerente de Projeto que, no fluxo anterior, estava atribuído ao ATI. Já a atividade de Criar Protótipo de Telas foi substuída pela Atividade de Modelagem de Telas, permanecendo o mesmo envolvido. Foi adicionada a atividade de Atribuir Envolvido na posição cinco do fluxo do processo.

Dando proseguimento nas mudanças, foi constatado que atividades de Criar diagrama de dados e Criar diagramas UML foram retiradas da responsabilidade do gerente de projeto e atribuídas ao Analista de Dados e ao Analista de Sistemas, respectivamente. Ao papel de Analista de Testes criado no novo fluxo do processo foram atibuídas as atividades de inspeção de modelos e criar casos de testes. Além das atividades de Realizar Testes e criar ou atualizar manual. Além destas mudanças, foi adicionada nova atividade de Realizar Teste de Usabilidade, tendo como envolvido o Designer.

Com as mudanças sofridas no fluxo do processo de controle de mudança, seguindo a abordagem utilizada, o AS IS do novo fluxo de atividades foi representado novamente. A Figura 10 apresenta a modelagem AS IS refinada do controle de mudanças.

Figura 10 – Modelagem AS IS do subprocesso Controlar Mudanças refinado




Fonte: Elaboração própria.

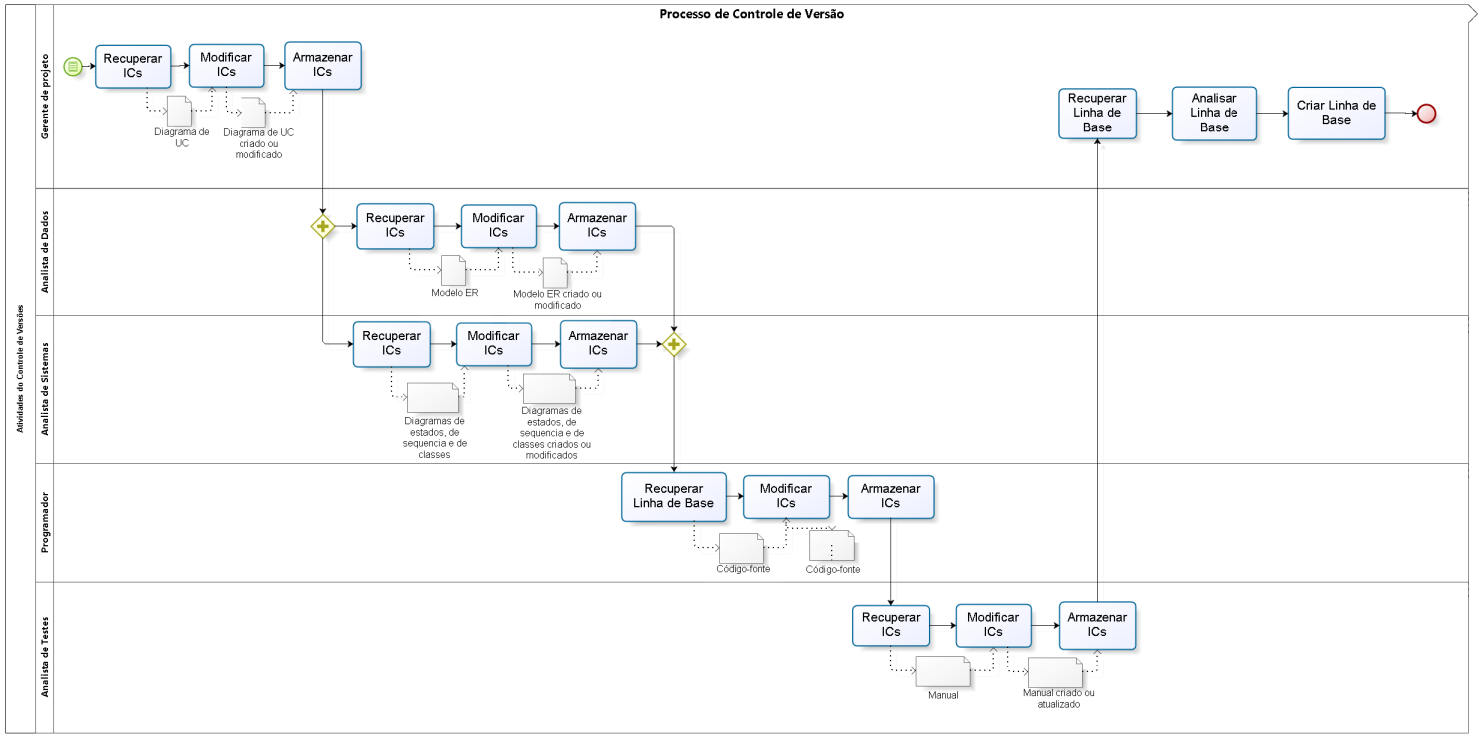
No refinamento do fluxo textual do processo de controle de versão, foram identificadas apenas as atividades que interagem com o Subversion, que é o sistema de controle de versões utilizado pela CODEV. Além do Subversion, que é utilizado para automatizar o controle de versões dos ICs, a CODEV utiliza o repositório do *Enterprise Archtect* para versionar, manualmente, os artefados produzidos. O *Enterprise Archtect* é uma ferramenta para modelagem e design visual baseado na *Unified Modeling Language* ou Linguagem de Modelagem Unificada (UML) (UML, 2016).

Tabela 6 – Processo de controle de versão da CODEV refinado

| Passo | Atividade | Responsável | Artefato | Ferramenta |
|---|---|---|---|---|
| **1** | Recuperar Itens de Configuração ICs | Gerente de Projeto | **Entrada:**Nenhum **Saída:**Diagrama de UC | Sistema de Controle de Versão |
| **2** | Modificar Itens de Configuração ICs | Gerente de Projeto | **Entrada:**Diagrama de UC **Saída:**Diagrama de UC modificado | Sistema de Controle de Versão |
| **3** | Armazenar Itens de Configuração ICs | Gerente de Projeto | **Entrada:**Diagrama de UC modificado **Saída:**Nenhum | Sistema de Controle de Versão |
| **4** | Recuperar Itens de Configuração ICs | Analista de Dados | **Entrada:**Nenhum **Saída:**Modelo ER | Sistema de Controle de Versão |
| **5** | Modificar Itens de Configuração ICs | Analista de Dados | **Entrada:**Modelo ER **Saída:**Modelo ER modificado | Sistema de Controle de Versão |
| **6** | Armazenar Itens de Configuração ICs | Analista de Dados | **Entrada:**Modelo ER modificado **Saída:**Nenhum | Sistema de Controle de Versão |
| **7** | Recuperar Itens de Configuração ICs | Analista de Sistemas | **Entrada:**Nenhum **Saída:**Diagramas de estados, | Enterprise Archtect |
| **8** | Modificar Itens de Configuração ICs | Analista de Sistemas | **Entrada:**Diagramas de estados, de sequências e de classes **Saída:**Diagramas de estados, de sequências e de classes modificados | Enterprise Archtect |
| **9** | Armazenar Itens de Configuração ICs | Analista de Sistemas | **Entrada:**Diagramas de estados, de sequências e de classes **Saída:**Nenhum | Enterprise Archtect |
| **10** | Recuperar Linha de Base | Programador | **Entrada:**Nenhum **Saída:**Código-Fonte | Sistema de Controle de Versão |
| **11** | Modificar Itens de Configuração ICs | Programador | **Entrada:**Código-Fonte **Saída:**Código-Fonte modificado | Sistema de Controle de Versão |
| **12** | Armazenar Itens de Configuração ICs | Programador | **Entrada:**Nenhum **Saída:**Código-Fonte | Sistema de Controle de Versão |
| **13** | Recuperar Itens de Configuração ICs | Analista de Testes | **Entrada:**Nenhum **Saída:**Manual | Sistema de Controle de Versão |
| **14** | Modificar Itens de Configuração ICs | Analista de Testes | **Entrada:**Manual **Saída:**Manual modificado | Sistema de Controle de Versão |
| **15** | Armazenar Itens de Configuração ICs | Analista de Testes | **Entrada:**Manual modificado **Saída:**Nenhum | Sistema de Controle de Versão |
| **16** | Recuperar Itens de Configuração ICs | Gerente de Projeto | **Entrada:**Nenhum **Saída:**Itens de Configuração ICs | Sistema de Controle de Versão |
| **17** | Analisar Itens de Configuração ICs | Gerente de Projeto | **Entrada:**Itens de Configuração ICs **Saída:**Nenhum | Sistema de Controle de Versão |
| **18** | Criar linha de Base | Gerente de Projeto | **Entrada:**Itens de Configuração ICs **Saída:**Linha de base | Sistema de Controle de Versão |

Fonte: Elaboração própria.

Figura 11 – Modelagem AS IS do subprocesso de Controlar Versões refinado.




Fonte: Elaboração própria.

Após o refinamento do fluxo textual do processo de controle de mudanças e controle de versão, representados através das modelagens em BPMN apresentadas nas Figuras 10 e 11, iniciou-se a etapa de análise e discussões com a finalidade de identificar possibilidades de melhorias no processo atual.

## 5.2 Construção do SHOULD BE

O mapeamento e modelagem do AS IS, conforme apresentado na seção 5.1, permitiu visualizar quais ICs são adicionados no GC da CODEV. Após a modelagem do AS IS do controle de mudanças e controle de versões apresentados, respectivamente, nas Figuras 10 e 11, foi possível identificar pontos de melhorias para modelagem do estado SHOULD BE.

Para construção do estado melhorado do processo de GC, foram utilizadas como referência as áreas do CMMI e MPS.BR, além do fluxo de atividades proposto por Engholm (2010).

A necessidade de melhoria identificada no item 1 da Tabela 7 faz-se necessária para garantir o controle rigoroso de acesso aos sistemas de controle de mudança e versões. O acesso por perfil deve evitar que ICs sejam alterados por pessoas não autorizadas.

Já o item 2 tem o propósito de facilitar a auditoria no processo de GC, permitindo que, durante as revisões das mudanças, as alterações sejam rastreadas rapidamente.

A necessidade de melhoria identificada no item 3 tem como propósito definir papel e responsabilidade pelo processo de GC. Atribuindo a um membro da equipe da CODEV a responsabilidade pelo Processo geral de GC.

Já o item 4 tem como propósito verificar, a cada solicitação de mudanças, a necessidade de alterar o ambiente de GC, descrevendo as alterações no Plano de GC.

O item 5 tem como propósito estabeler uma atividade para planejamento de GC, em que serão executadas atividades para melhoria do processo e garantia da sua integridade. O subprocesso tem como objetivo atender ao GCO 2 e GCO 3 - MPS.BR.

O item 6 tem como propósito garantir que, durante o ciclo de vida dos projetos, sejam adotados os procedimentos definidos no Plano de GC, atendendo ao GCO7/MPS-BR e SP 3.2-CMMI. Também atendendo ao fluxo de atividades proposto por Engholm (2010).

E, por fim, o item 7 tem como objetivo estabelecer o marco do processo de GC para reportar resultados do gerenciamento de configuração, atendo ao fluxo de atividade proposto por Engholm (2010).

Tabela 7 – Tabela AS IS e SHOULD BE

| | **AS IS** | **SHOULD BE** |
|---|---|---|
| **Item** | **Ponto de Melhoria** | **Melhoria Sugerida** |
| **1** | Mesmo nível de usuário no sistema de controle de versão. | Criar níveis de usuário para acesso ao sistema de controle de versão. |
| **2** | Necessidade de manter rastreamento automatizado das versões criadas no sistema de controle de versões com o sistema de controle de mudanças | Vincular as mudanças no controle de versão ao sistema de controle de mudanças. |
| **3** | Necessidade de definir papel para gerente de configuração | Definir papel de gerente de GC no processo. |
| **4** | Necessidade de atividade para definição do ambiente de GC | Definir atividade para definição de ambiente de GC.<br>Definir atividade para verificação da necessidade de mudança no ambiente. |
| **5** | Necessidade de atividade para planejar o GC | Definir atividade de planejar GC.<br>Adicionar subatividades de planejar GC. |
| **6** | Necessidade de atividade de auditoria do processo de GC | Adicionar atividade no processo de GC |
| **7** | Necessidade de atividade de reportar resultados de GC | Definir atividade no processo de GC para reportar resultados |

Fonte: Elaboração própria.

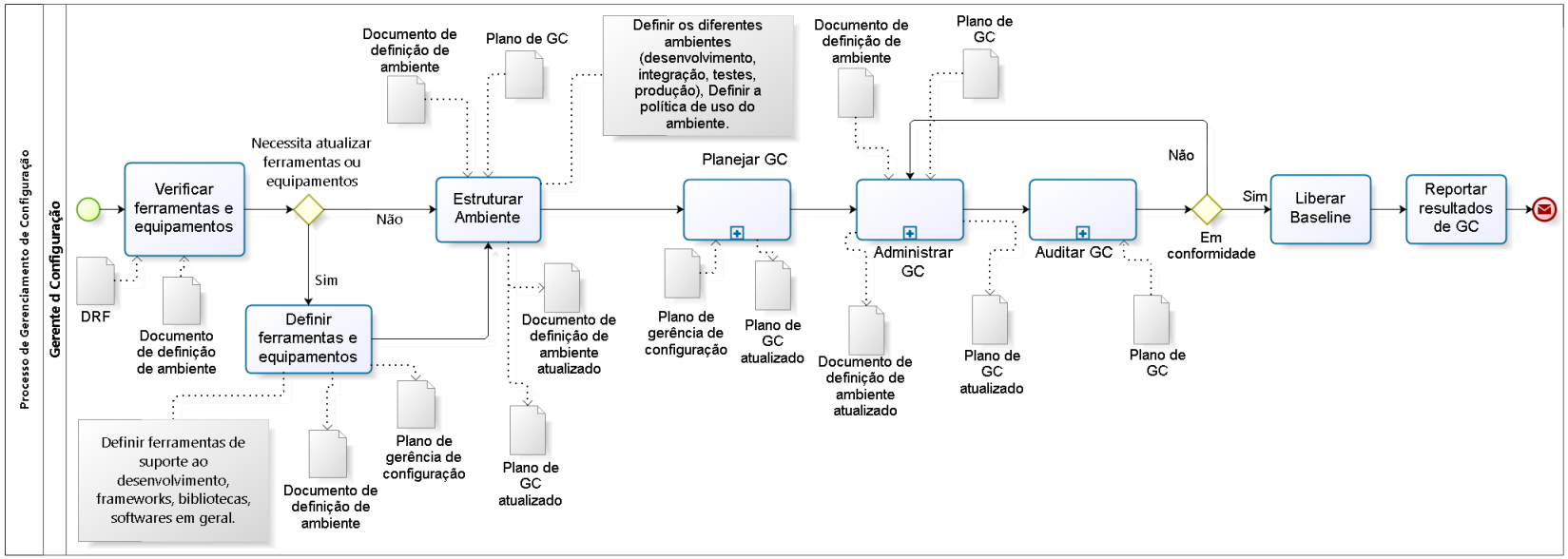
### 5.2.1 Modelagem do SHOULD BE

Finalizada a etapa de identificação de melhorias no processo atual de GC da CODEV, inciou-se a etapa de modelagem do SHOULD BE, considerando-se as melhorias identificadas e apresentadas na Tabela 7.

Para modelagem do SHOULD BE não foi necessário elaborar o fluxo textual da sequência das atividades do processo atual com as melhorias sugeridas, tendo em vista que já existe o entendimento do fluxo de atividades do processo atual de GC. Sendo assim, inciou-se pela modelagem do SHOULD BE.

Nesta modelagem, foram adicionadas as melhorias identificadas e descritas na Tabela 7. A Figura 12 apresenta a modelagem do processo geral de GC da CODEV.

Figura 12 – SHOULD BE do processo geral de GC.




Fonte: Elaboração própria.

Com a elaboração da modelagem SHOULD BE do processo, verifica-se, conforme a Figura 12, o surgimento de novas atividades para o processo de GC além da definição de novo papel e atribuição de responsabilidade pelo processo. A seguir, serão descritas cada atividade do processo melhorado e uma breve explicação dos objetivos de cada atividade.

Atividades realizadas durante o fluxo do processo:

1. Verificar ferramentas e equipamentos

   Esta atividade tem como objetivo permitir ao gerente de configuração verificar se o ambiente atual de GC é satisfatório para desenvolvimento da solução contida no DRF. O artefado de entrada para esta atividade é o documento de definição de ambiente que possui a descrição das ferramentas de suporte ao desenvolvimento, definição de bibliotecas, *frameworks* e softwares em geral necessários para o desenvolvimento.

2. Definir ferramentas e equipamentos

   Caso seja necessário adicionar ou modificar as ferramentas utilizadas de suporte ao desenvolvimento da solução, o gerente de configuração executará esta atividade, tendo como artefato de saída o Documento de definição de ambiente e o Plano de gerenciamento de configuração criados ou modificados.

3. Estruturar ambiente

   O objetivo desta atividade é definir os diferentes ambientes utilizados para desenvolvimento, tais como ambiente de desenvolvimento, testes, integração e ambiente de produção. Esta atividade possui como artefatos de entrada o Documento de definição de ambiente e o Plano de gerência de configuração, sendo que, após executada esta atividade, os dois documentos de entrada são atualizados.

4. Planejar GC

   Esta atividade consiste em um conjunto de atividades que são executadas para planejamento do GC. Esta atividade tem como artefato de entrada o Plano de GC que, após conclusão da atividade, este documento é atualizado. A seguir, serão descritas cada atividade do subprocesso de Planejar GC.

   a) Definir organização, papéis e responsabilidades

      A primeira atividade tem como propósito definir os papéis dos envolvidos no processo de GC e suas respectivas responsabilidades no processo.

   b) Definir, políticas e procedimentos para registro do status da configuração

      Na sequência, é realizada a atividade para definir políticas de registro de GC, tendo como resultado a definição da forma de registro do status da configuração

dos produtos de software. Nesta atividade, é definido como será este registro, por exemplo através de tags no sistema de controle de versão ou outra forma de registro.

c) Identificar e registrar itens de configuração

Nesta atividade, são identificados os ICs os quais serão colocados e mantidos na GC, bem como os critérios para promoção de artefatos como ICs.

d) Definir padrão para nomeação de itens de configuração

Esta atividade tem como objetivo definir a forma de nomeação dos itens de configuração, descrevendo o padrão para nomear ICs.

e) Planejar auditorias

Tem como propósito estabelecer critérios para verificação do andamento do processo de GC. Nesta etapa, é definida a forma de registro das auditorias de GC e descritos os momentos da execução da atividade de Auditar GC.

f) Planejar baselines

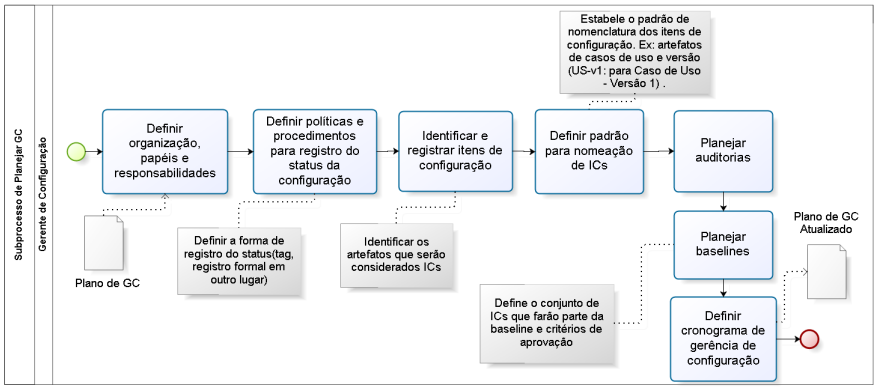
Nesta atividade, deve ser definido o conjunto de ICs que fará parte de uma baseline. O momento da entrada de cada IC é definido na atividade de identificar e registrar itens de configuração.

g) Definir cronograma de GC

Esta atividade tem como objetivo elaborar um conjunto de marcos no processo de GC para acompanhamento do processo e verificar possíveis desvios. Apartir do cronograma definido, o gerente de configuração poderá reportar os resultados do processo de GC e avaliar o andamento do processo.

A Figura 13 apresenta a modelagem do subprocesso de Planejar GC com as atividades descritas anteriormente.

Figura 13 – SHOULD BE do processo planejar GC.




Fonte: Elaboração própria.

5. Administrar GC

   Nesta atividade, é que existe a interação de toda a equipe da CODEV, ela possui subatividades que são executadas durante o ciclo de vida de um projeto de software. Possui como artefatos de entrada o Documento de definição de ambiente e o Plano de GC. A seguir, são descritas cada atividade do subprocesso de Administrar GC e respectivos objetivos.

   a) Controlar mudanças

      Esta atividade está diretamente ligada ao processo de software da CODEV, apresentado nos Anexo B e Anexo C. Nesta atividade, é que ocorre os registros das mudanças sofridas pelos ICs durante o ciclo de desenvolvimento. Estas mudanças são registradas na ferramenta Mantis, podendo ser acompanhadas pelo gerente de projeto e gerente de configuração, conforme apresentado na Figura 8.

   b) Controlar versões

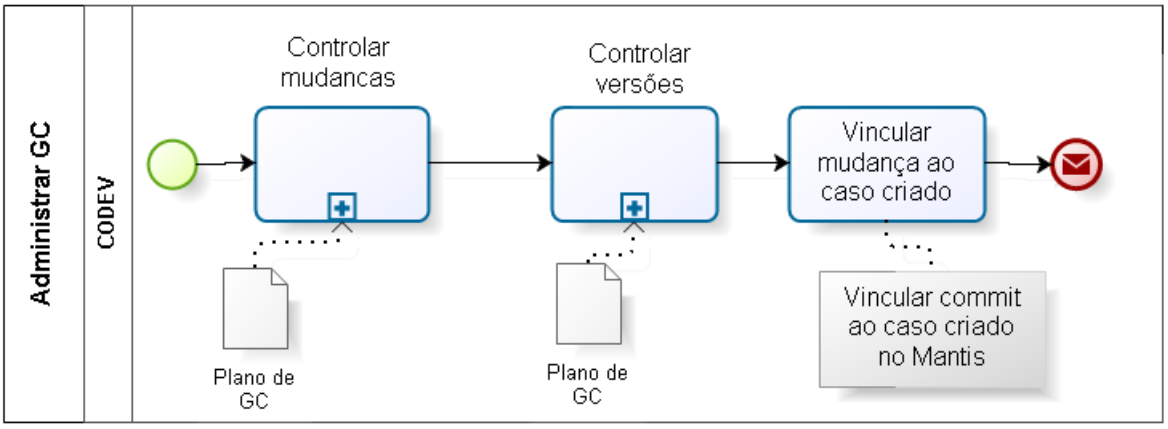
      Já esta atividade é realizada durante o desenvolvimento da solução proposta e contida no DRF. Nesta atividade, ocorre o armazenamento dos ICs para controle de versões, sendo uma tarefa distribuída entre a equipe da CODEV, conforme apresentado na Figura 9.

   c) Vincular mudança ao caso criado

      Esta ação tem como finalidade rastrear as alterações registradas no sistema de controle de versão com o sistema de controle de mudanças. Nesta atividade, o responsável pela mudança de um IC deverá vincular esta alteração ao caso criado no sistema de controle de mudanças.

A Figura 14 apresenta a modelagem do subprocesso de Planejar GC contendo as atividades descritas anteriormente.

Figura 14 – SHOULD BE da atividade de Administrar GC.




Fonte: Elaboração própria.

6. Auditar GC

   Em períodos descritos no Plano de GC, o gerente de configuração realizará auditorias no processo de GC, com objetivo de verificar se o processo está sendo executado conforme planejado e definido pela atividade de Planejar GC. Se encontrada alguma irregularidade, o gerente de GC deve reportar a ocorrência ao responsável pelo item de configuração.

   Após correção da irregularidade, o gerente de GC executa novamente a atividade de auditoria e, se o processo estiver de acordo com o Plano de GC, segue-se o fluxo do processo.

   A seguir, estão descritas as atividades do subprocesso de Auditar GC e seus objetivos.

   a) Analisar integridade da baseline

      O gerente de configuração deve analisar a integridade da baseline e verificar se contém todos os ICs definidos no Plano de GC. Se encontrada alguma irregularidade, o gerente de configuração comunicará ao responsável pelo IC a fim de solucionar o problema.

   b) Analisar padrões de GC

      Nesta atividade, o gerente de configuração verifica se os padrões estabelecidos no Plano de GC estão sendo seguidos. Caso encontre alguma irregularidade, deverá comunicar o responsável pelo IC a fim de sanar o desvio.

   c) Analisar integridade de ICs

      Esta atividade tem como propósito verificar se os ICs estão de acordo com o estabelecido na atividade de Planejar GC e registrados no Plano de GC. Verificar se a nomenclatura segue o padrão estabelecido.

   d) Registrar auditoria

      Finalizando o subprocesso de Auditar GC, o gerente de configuração deverá reportar os resultados da auditoria. A forma de registro poderá ser através de Check List ou registro no sistema de controle de mudanças.

   As atividades do subprocesso de Auditar GC podem ser visualizadas na Figura 15.

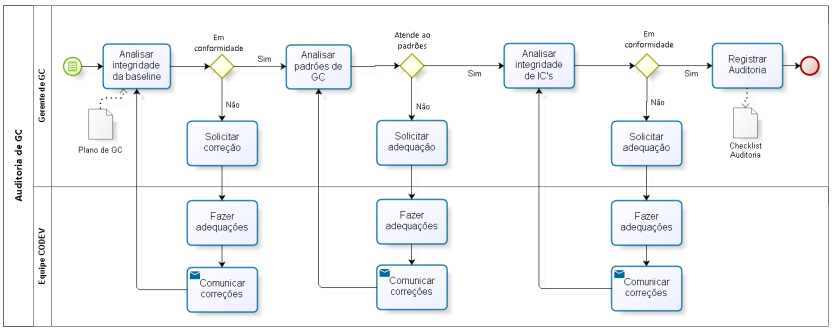
7. Liberar baseline

   Após realizada a auditoria de GC, o gerente de GC executa a atividade de Liberar Baseline, que tem como propósito autorizar a liberação da baseline para ambiente de produção. Esta atividade deverá ser executada após concluída a atividade de auditoria, a qual garantirá a integridade da Baseline.

8. Reportar resultados de GC

   A última atividade do processo geral de GC é a de Reportar resultados de GC,

Figura 15 – SHOULD BE da atividade de auditoria.




Fonte: Elaboração própria.

que tem como objetivo disponibilizar os resultados da gestão de configuração e andamento do processo de software. Estes resultados são fundamentais para que a equipe da CODEV realize discussões a fim de melhorar ou solucionar problemas encontrados durante o processo de GC.

## 5.3 Discussões do SHOULD BE

Esta etapa consistiu na apresentação da modelagem SHOULD BE na CODEV, a qual foi avaliada pelo coordenador da unidade com a finalidade de verificar a viabilidade de adicionar as melhorias sugeridas ao processo atual de GC da CODEV.

Para seleção dos critérios de viabilidade da implantação das melhorias, foi atribuído um conjunto de critérios identificados como: Prioritário (P), Recomendável (R) e Necessita Análise do Contexto (NAC).

O coordenador da CODEV analisou a viabilidade de cada item identificado como melhoria sugerida e selecionou uma opção para indicar o grau de viabilidade com a respectiva justificativa.

A Tabela 8 apresenta as alternativas selecionadas e as justificativas para cada item de melhoria sugerida.

Tabela 8 – Tabela Discussões SHOULD BE

| | **SHOULD BE** | Selecione uma opção | | | Justifique |
|---|---|---|---|---|---|
| **Item** | **Melhoria Sugerida** | P | R | NAC | |
| **1** | Criar níveis de usuário para acesso ao sistema de controle de versão. | | | X | Como não há projetos sigilosos ou dados confidenciais mantidos no SVN, no momento não é necessário implantarmos níveis de usuário neste sistema. O desenvolvedor, uma vez autenticado no SVN, pode acessar qualquer projeto. |
| **2** | Vincular as mudanças no controle versão ao sistema de controle de mudanças. | X | | | É interessante implantarmos de uma forma efetiva, tendo em vista facilitar o rastreamento da mudança, identificar de uma forma mais rápida qual solicitação de mudança ocasionou uma determinada alteração no código. Atualmente utilizamos um plugin do Netbeans para comunicação com SVN e Mantis o qual provê esta funcionalidade(commit no SVN linkando com a task do Mantis), no entanto, nem todos os membros do time tem o costume de utilizá-lo. |
| **3** | Definir papel de gerente de GC no processo. | X | | | Acredito que o papel de gerente de configuração possa ser desempenhado pelo coordenador de desenvolvimento (no caso, eu mesmo), pelo menos num primeiro momento |
| **4** | Definir atividade para definição de ambiente de GC. Definir atividade para verificação da necessidade de mudança no ambiente. | | X | | Como o ambiente de GC (SVN, Mantis) já está implantado, não seria necessário defini-lo de uma forma diferente ao iniciar um novo projeto. |
| **5** | Definir atividade de planejar GC. Definir subatividades de planejar GC. | | X | | Idem ao anterior. |
| **6** | Adicionar atividade no processo de GC | X | | | Seria importante realizarmos uma auditoria numa determinada entrega, inspecionando se os itens de configuração estão corretos, como também as atividades do processo de software como um todo. Esta função pode ser desempenhada pelo coordenador de desenvolvimento ou pelo analista responsável pelo projeto. |
| **7** | Definir atividade no processo de GC para reportar resultados | X | | | Os resultados devem ser expostos e analisados, tendo em vista justificar as atividades (ligadas a GC) que por ventura venham a ser implantadas dentro do procsso de software. |
| **Legenda: P - Prioritário; R - Recomendável ; NAC - Necessita de análise do contexto** | | | | | |

Fonte: Elaboração própria.

Após discussões com o coordenador da CODEV e levando em consideração os itens assinalados na Tabela 8, foi iniciada a etapa de construção do estado futuro do processo de GC.

## 5.4 Construção do TO BE

Para elaboração da modelagem TO BE do processo de GC, foram utilizados apenas os itens considerados Prioritários (P) pelo coordenador da CODEV, conforme apresentado na Tabela 8. Os itens identificados como Recomendável (R) e Necessita de análise do contexto (NAC) não foram utilizados, por tratarem-se de ações para médio prazo, longo prazo ou não se aplicarem à realidade da CODEV.

A Tabela 9 apresenta apenas os itens que serão utilizados para modelagem TO BE do processo de GC.

Tabela 9 – Tabela AS IS e TO BE

| | **AS IS** | **TO BE** |
|---|---|---|
| **Item** | **Ponto de Melhoria** | **Melhoria Sugerida** |
| **1** | Necessidade de manter rastreamento automatizado das versões criadas no sistema de controle de versões com o sistema de controle de projetos | Vincular as mudanças no controle versão ao sistema de controle de mudanças. |
| **2** | Necessidade de definir papel para gerente de configuração | Definir papel de gerente de GC no processo. |
| **3** | Necessidade de atividade de auditoria do processo de GC | Adicionar atividade no processo de GC |
| **4** | Necessidade de atividade de reportar resultados de GC | Definir atividade no processo de GC para reportar resultados |

Fonte: Elaboração própria.

Apartir das informações descritas na Tabela 9, foi elaborada modelagem TO BE do processo de GC da CODEV, com as melhorias sugeridas atendendo à realidade do setor.

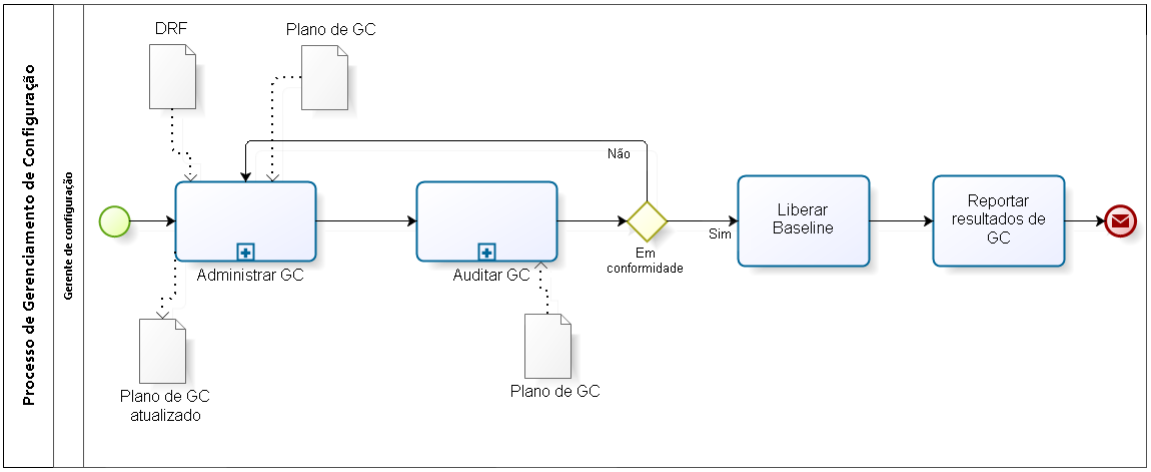
Conforme definido pelo coordenador da CODEV, para construção do TO BE foram retiradas da modelagem SHOULD BE as atividades de Verificar ferramentas e equipamentos, Definir ferramentas e equipamentos, Estruturar ambiente e atividade de Planejar GC.

Como o processo de GC do setor já está implantado e definido, o coordenador do setor optou por retirar as atividades que antecedem a implantação e administração de GC.

A configuração atual dos ambientes atende à realidade da CODEV e não sofrerá mudanças de imediato. Por esta razão, o coordenador optou por retirá-la da modelagem TO BE do processo de GC.

A Figura 16 apresenta a modelagem do processo geral de GC da CODEV.

Figura 16 – Modelagem TO BE do processo geral de GC.




Fonte: Elaboração própria.

A seguir, estão descritas a sequência de atividades do TO BE do processo de GC da CODEV e identificação das mudanças.

1. Administrar GC
   Como foram retiradas as atividades de planejamento da GC, a primeira atividade do processo de GC é a de Administrar GC, o que possui subatividades, conforme descritas a seguir.

   a) Controlar mudanças

   - Esta atividade não sofreu alterações entre a modelagem SHOULD BE e TO BE.
   - Permaneceu a mesma configuração conforme pode ser observado na Figura 10.

   b) Controlar versões

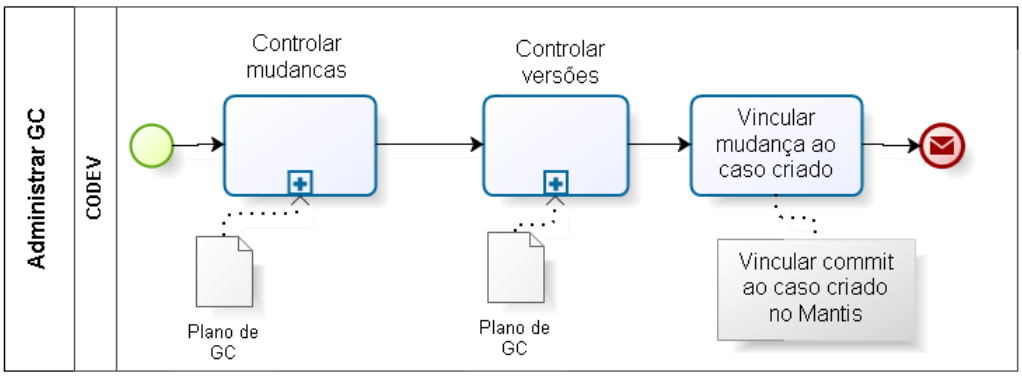
   - Não sofreu alterações entre a modelagem SHOULD BE e TO BE.
   - Permaneceu a mesma configuração, conforme pode ser observado na Figura 11.

c) Vincular mudança ao caso criado

- E a última atividade do subprocesso de Adminitrar GC também foi mantida na modelagem TO BE, conforme opção do Coordenador da CODEV.

A Figura 17 apresenta as atividades da versão TO BE da atividade de Administrar GC.

Figura 17 – Modelagem TO BE da atividade Administrar GC.




Fonte: Elaboração própria.

2. Auditar GC
   A modelagem TO BE da atividade de Auditar GC permaneceu a mesma da modelagem SHOULD BE. A seguir, são descritas as atividades do subprocesso de Auditar GC para versão TO BE.

   a) Analisar integridade da baseline

   b) Analisar padrões de GC

   c) Analisar integridade de ICs

   d) Registrar auditoria

   A Figura 18 apresenta a modelagem TO BE do subprocesso de Auditar GC contendo as quatro atividades citadas anteriormente.

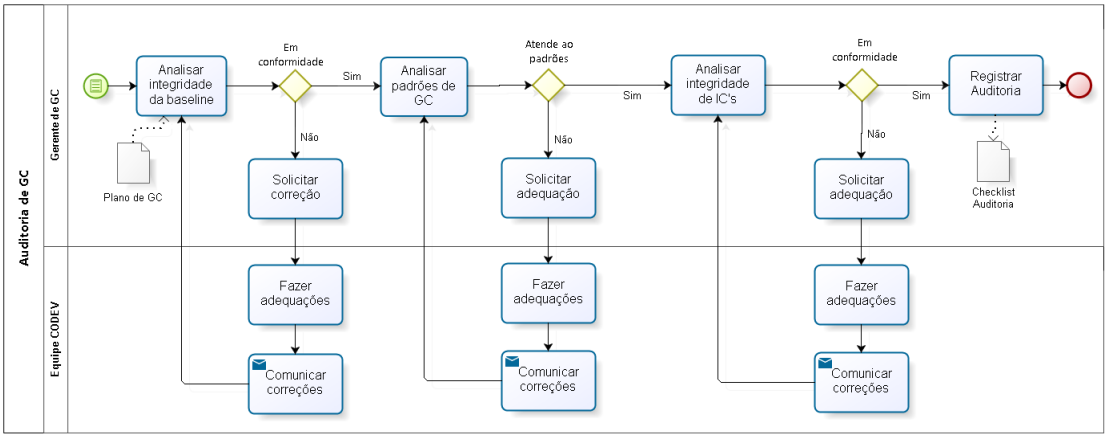
3. Liberar baseline

   - Por opção do coordenador da CODEV, manteve-se esta atividade na modelagem TO BE.

4. Reportar resultados de GC

   - Manteve-se esta atividade na modelagem TO BE conforme opção do coordenador da CODEV.

Figura 18 – Modelagem TO BE da atividade de Auditar GC




Fonte: Elaboração própria.

Com a conclusão da etapa da modelagem TO BE do processo de gerenciamento de configuração, encerra-se o ciclo de etapas da metodologia proposta por Tolfo (2014) e utilizada como suporte para elaboração deste trabalho.

O ciclo de representações, simulações e discussões com o coordenador da CODEV permitiram a elaboração de um processo de gerenciamento de configuração de software que atendeu à realidade do setor e mantendo a estrutura e organização das atividades envolvidas nos subprocessos modelados.

## 5.5 Síntese dos resultados obtidos

O processo atual de GC da CODEV, que estava de forma empírica, foi formalizado através da modelagem do AS IS, conforme apresentado na seção 5.1. Com a formalização do processo, foi possível identificar pontos de melhorias que estão descritos na Tabela 7 da seção 5.2. As discussões e refinamento da modelagem SHOULD BE, apresentada na seção 5.3, permitiu a comunicação e planejamento na CODEV para definição do TO BE.

Finalizada a etapa de discussões e refinamento com a CODEV, teve-se, como resultado final, o TO BE do processo de GC atendendo à realidade do setor.

# 6 Conclusão

Para atingir o objetivo desta pesquisa que consistiu em propor melhorias no processo atual de gerenciamento de configuração da DTIC, primeiramente, foi necessário entender o estado atual, identificando atividades realizadas de forma empírica. Após identificação destas atividades, o AS IS de processo de gerenciamento de configuração foi modelado, tendo como resultado a formalização do processo atual.

A partir da representação do AS IS, foi possível analisar o estado atual do processo e identificar pontos de melhorias. Estes pontos foram utilizados para construção da modelagem SHOULD BE do processo. Esta modelagem foi apresentada ao coordenador da CODEV, o que resultou no planejamento e comunicação entre a equipe da CODEV para construção do TO BE.

Ao final das discussões e refinamento, obteve-se a modelagem TO BE do processo de gerenciamento de configuração adequado ao setor. Esta representação do estado futuro conforme metodologia adotada poderá estar sempre em constante melhoramento através de discussões e análises. Com a formalização do processo de gerenciamento de configuração de software, a equipe da DTIC poderá organizar a implantação do processo modelado ou adequação deste processo de GCS ao processo de software utilizado pelo setor.

Com isso, conclui-se que a abordagem para mapeamento e modelagem de processo utilizada para elaboração da pesquisa é adequada para o estudo e representação de processos no contexto da Engenharia de Software.

Além disso, o ciclo de etapas desta abordagem permitiu o entendimento do processo realizado de forma implícita e identificação de melhorias neste processo.

Já a notação BPMN mostra-se eficiente para representação de processos de desenvolvimento de software, o que facilita a comunicação e melhora a visualização de processos.

## 6.1 Trabalhos futuros

A seguir são descritos possíveis trabalhos futuros.

A partir do estado TO BE do processo de gerenciamento de configuração de software da CODEV, pode-se elaborar um estudo para adequação do TO BE do processo de GCS ao processo de software da CODEV, planejando as etapas de implantação, acompanhamento e análise de resultados.

Também se pode elaborar um estudo das ferramentas para integração entre o sistema de controle de mudanças e o sistema de controle de versões. Através deste estudo,

poderá ser sugerida a ferramenta que seja compatível com as ferramentas de gerenciamento de configuração utilizadas pela CODEV.

Pode-se elaborar um estudo aprofundado para Auditoria do processo de GC, identificando o perfil para conduzir esta atividade.

Além disso, pode-se elaborar um estudo para adequação da modelagem SHOULD BE do processo em empresas que possuam nível 2 CMMI e nível F do MPS.BR.

# Referências

CMMI, Product Team. CMMI for Development, Version 1.3.Software Engineering Institute, 2010. Disponível em:<http://resources.sei.cmu.ed/> Acesso em: 20/04/2015.

CAMPOS, André L.N. Modelagem de Processos com BPMN. 2.ed. Rio de Janeiro: Brasport, 2014.

BERTUCCI, J.L.O. Metodologia Básica para Elaboração de Trabalhos de Conclusão de Curso. 3.ed. São Paulo: Atlas S.A., 2011.

BPMN. 2015. Disponível em:https://www.bpmn.org. Acessado em: 03/05/2015.

BRACONI, J. OLIVEIRA, S. B. et. al. Análise e Modelagem de Processos de Negócio: Foco na notação BPMN. São Paulo: Atlas, 2009.

BUZAN, T. Mapas Mentais. 1.ed. São Paulo: Cultrix, 2005.

Engholm Júnior, Hélio. Engenharia de Software na Prática. São Paulo: Novatec Editora, 2010.

GIL, A. Como elaborar projetos de pesquisa. Atlas S.A., 2010.

GIT, 2015. Disponível em: https://git-scm.com. Acessado em: 06/09/2015.

GPL. General Public Licence, 2015. Disponível em: https://opensource.org/licenses/MIT. Acessado em: 26/09/2015.

ISO/IEC. Systems and software engineering - Software life cycle processes, 2013. Disponível em: http://www.iso.org/iso/. Acessado em: 21/11/2015.

MANTIS. 2015. Disponível em: http://www.mantisbt.org/. Acesso em: 20/06/2016.

MPS.BR. Softex. Melhoria de Processo do Software Brasileiro. Guia Geral MPS de Software: 2012. Disponível em:<http://www.softex.br/mpsbr/guias/>. Acessado em: 05/03/2015.

MOLINARI, L. Gerência de Configuração: Técnicas e Práticas no Desenvolvimento do Software. Florianópolis: Visual Books, 2007.

POPPENDIECK, Mary e Tom. Implementando o Desenvolvimento Lean de Software: Do Conceito ao Dinheiro. Porto Alegrete: Bookman, 2011.

SANTOS, C.W. *PROCESSO DE V&V APLICADO AO DESENVOLVIMENTO DE SOFTWARE DO NTIC*. Alegrete, 2014. 145 p. Monografia (Graduação em Engenharia de Software), Universidade Federal do Pampa, 2014.

SOMMERVILLE, Ian. *Engenharia de Software*. 8.ed. São Paulo: Perason Addison-Wesleyr, 2011.

SOMMERVILLE, Ian. *Engenharia de Software*. 9.ed. São Paulo: Perason Addison-Wesleyr, 2011.

SUBVERSION. 2015. Disponível em:https://subversion.apache.org/. Acessado em: 03/09/2015.

SWEBOK, Guia do Conjunto de Conhecimentos em Engenharia de Software, Versão 3.0, IEEE Computer Society, 2014. Disponível em:<http://www.swebok.org/> Acesso em: 20/04/2015.

TOLFO, Cristiano. Disciplina de sistemas de informação. Alegrete: Unipampa, 2014. Notas de aula.

TRAC. *Integrated SCM & project management*, 2016. Disponível em:<https://trac.edgewall.org>.

UML. 2016. Disponível em:https://www.uml.org/. Acessado em: 15/05/2016.

WAZLAWICK, R.S. *Engenharia de Software: conceitos e práticas*. 1.ed. Rio de Janeiro: Elsevier, 2013.

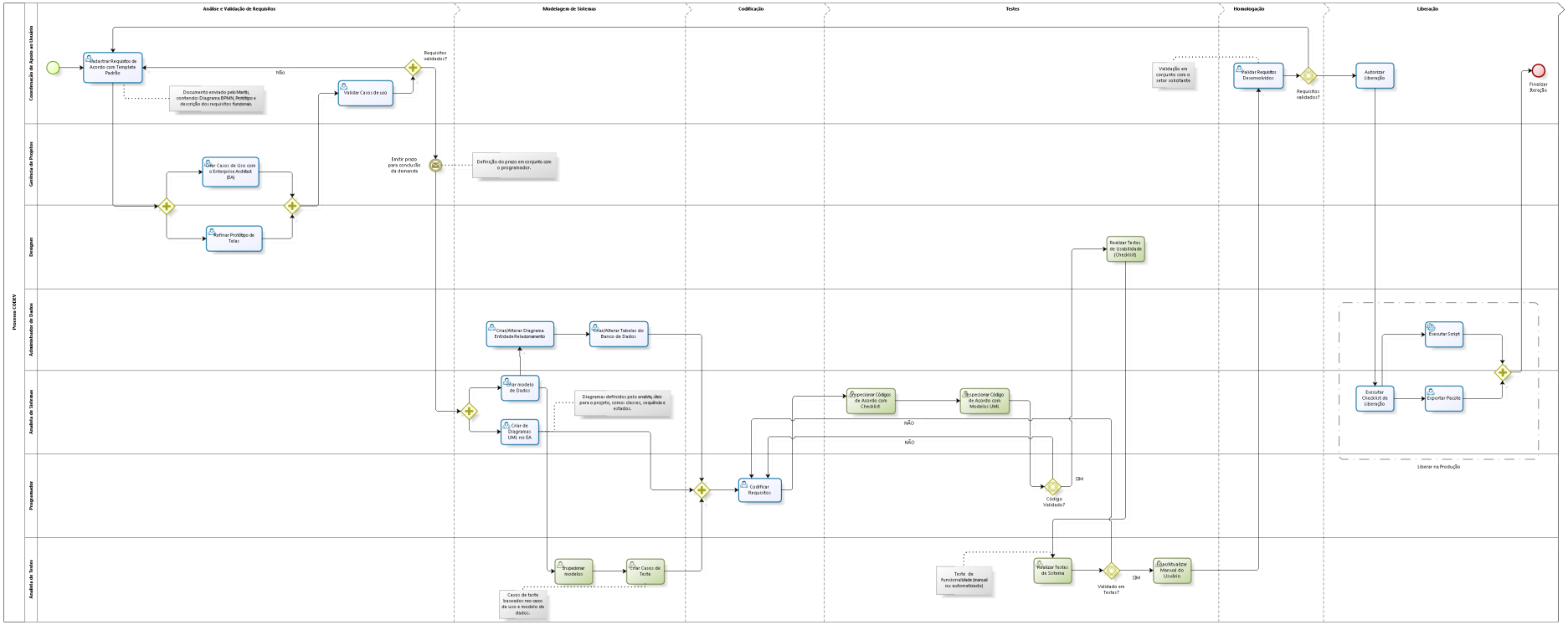
# Anexos

# ANEXO A – Modelo de Tarefas

| *nr* | *Tipo de Tarefa* | *Título da Tarefa* | *Texto/Descrição* | *Filha de* | *Depende de* | *Responsável/Grupo* |
|---|---|---|---|---|---|---|
| 1 | CDU | Versao vK | Desenvolvimento da Versao vK | - | - | Gerente |
| 2 | MODELAGEM | [CODIGOCDU]-[DESCRICAO_CDU] | Modelos do [CODIGOCDU]-[DESCRICAO_CDU] | 1 | - | Gerente |
| 3 | UML | MD - [CODIGOCDU]-[DESCRICAO_CDU] | Criar Modelo de Dados [CODIGOCDU]-[DESCRICAO_CDU] | 2 | - | ATI |
| 4 | SGBD | DER - [CODIGOCDU]-[DESCRICAO_CDU] | Criar Diagrama Entidade Relacionamento do [CODIGOCDU]-[DESCRICAO_CDU] | 2 | 3 | DBA |
| 5 | SGBD | SQL - [CODIGOCDU]-[DESCRICAO_CDU] | Criar SQL [CODIGOCDU]-[DESCRICAO_CDU] | 2 | 4 | DBA |
| 6 | UML | DCL - [CODIGOCDU]-[DESCRICAO_CDU] | Criar Diagrama de Classe do [CODIGOCDU]-[DESCRICAO_CDU] | 2 | - | ATI |
| 7 | UML | DER - [CODIGOCDU]-[DESCRICAO_CDU] | Criar Diagrama de Sequencia do [CODIGOCDU]-[DESCRICAO_CDU] | 2 | 6 | ATI |
| 8 | INSPEÇÃO | Versao vK | Inspecionar modelagem da Versao vK | 1 | 2 | ATI/TTI |
| 9 | TESTES | Versao vK | Casos de Teste da Versao vK | 1 | - | Gerente |
| 10 | TESTES | [CODIGOCDU]-[DESCRICAO_CDU] | Criar Casos de Teste do [CODIGOCDU]-[DESCRICAO_CDU] | 9 | - | ATI/TTI |
| 11 | IMPLEMENTAÇÃO | Versao vK | Implementação da Versao vK | 1 | 8 | Gerente |
| 12 | CODIFICAÇÃO | [CODIGOCDU]-[DESCRICAO_CDU] | Codificar [CODIGOCDU]-[DESCRICAO_CDU] | 11 | 8 | ATI/TTI |
| 13 | INSPEÇÃO | Versao vK | Inspecionar codificação da Versao vK | 1 | 11 | ATI/TTI |
| 14 | TESTES | Versao vK | Testar Versao vK | 1 | 13 | Gerente |
| 15 | TESTES | TUNIT [CODIGOCDU]-[DESCRICAO_CDU] | Realizar teste unitário do [CODIGOCDU]-[DESCRICAO_CDU] | 14 | 13 | ATI/TTI |
| 16 | ACEITAÇÃO | [CODIGOCDU]-[DESCRICAO_CDU] | Realizar teste de aceitação do [CODIGOCDU]-[DESCRICAO_CDU] | 14 | 13 | ATI/TTI |
| 17 | MANUAL | Versao vK | Criar/Atualizar Manual da Versao vK | 1 | 16 | ATI/TTI |

Legenda: ATI - Analista de Tecnologia da Informação; DBA - Administrador de Banco de Dados; TTI - Técnico em Tecnologia da Informação.

# ANEXO B – Processo de Software Novos Requisitos



# ANEXO C – Processo de Software Requisitos Não Previstos ou Bugs