Universidade de Brasília
Instituto de Ciências Exatas
Departamento de Ciência da Computação

Proposta de Melhoria de Processo de Implementação de Software: Estudo de Caso de um Sistema de Dados de Operações do Mercado Monetário

Lázara Aline de Oliveira Sousa Silveira

Dissertação apresentada como requisito parcial para conclusão do
Mestrado Profissional em Computação Aplicada

Orientador
Prof. Dr. Edgard Costa Oliveira

Brasília
2015

Ficha catalográfica elaborada automaticamente,
com os dados fornecidos pelo(a) autor(a)

S587p

Silveira, Lázara Aline de Oliveira Sousa
Proposta de melhoria de processo de implementação software: estudo de caso de um sistema de dados de operações do mercado monetário / Lázara Aline de Oliveira Sousa Silveira; orientador Edgard Costa Oliveira. -- Brasília, 2015.
179 p.

Dissertação (Mestrado - Mestrado Profissional em Computação Aplicada) -- Universidade de Brasília, 2015.

1. Implementação de Software. 2. ABNT NBR ISO 31000:2009. 3. Mineração de Textos. 4. AHP. 5. BPM. I. Oliveira, Edgard Costa , orient. II. Título.

Universidade de Brasília
Instituto de Ciências Exatas
Departamento de Ciência da Computação

Proposta de Melhoria de Processo de Implementação de Software: Estudo de Caso de um Sistema de Dados de Operações do Mercado Monetário

Lázara Aline de Oliveira Sousa Silveira

Dissertação apresentada como requisito parcial para conclusão do
Mestrado Profissional em Computação Aplicada

Prof. Dr. Edgard Costa Oliveira (Orientador)
UnB - Faculdade do Gama

Prof.ª Dr.ª Simone Borges Simão Monteiro
Departamento de Engenharia de Produção

Prof. Dr. João Carlos Felix Souza
Departamento de Engenharia de Produção

Prof. Dr. Marcelo Ladeira
Coordenador do Programa de Pós-graduação em Computação Aplicada

Brasília, 14 de agosto de 2015

Dedicatória

A meu esposo César Silveira, que foi paciente, companheiro, virtuoso e que muitas vezes, na minha ausência para me dedicar aos estudos, foi pai e mãe do nosso filho Lázaro Felipe.

A minha irmã Lorena, que acompanhou os vários momentos de ansiedade e dificuldade em conciliar a realização das minhas responsabilidades maternas e acadêmicas. E também aos familiares que, mesmo sem saber, foram estímulo e incentivo para que eu não desistisse dos meus sonhos e os tornassem realidade: Lúcia (mãe), Loane (irmã), Mauro (pai), Amália (avó) e Sebastião (avô).

Agradecimentos

Agradecer primeiramente a Deus, que me permitiu realizar dois sonhos no ano de 2013: a benção de engravidar do meu primeiro filho e de ser selecionada para o mestrado na UnB. Ao professor-orientador deste trabalho, Dr. Edgard Costa Oliveira, pela paciência, confiança, serenidade, inteligência e bom humor com que me trouxe ao fim desta jornada.

Ao meu esposo, que me deu todo apoio, incentivo e tranquilidade para continuar nos estudos.

Aos colegas do programa de Mestrado, pela amizade e companheirismo durante os dois anos de convívio.

Resumo

A gestão de riscos constitui-se em uma importante área do conhecimento para a eficiência do processo de desenvolvimento de *software*, pois permite que vulnerabilidades encontradas sejam tratadas adequadamente, possibilitando maior segurança e precisão na tomada de decisão. Um fator relevante para o sucesso da gestão de risco é o estabelecimento do contexto do que se pretende avaliar e melhorar dentro da organização. Portanto, para realização desse estudo, utilizou-se da norma ABNT NBR ISO 31000:2009 de gestão de riscos, para propor melhorias no processo de implementação de software de uma instituição financeira, a partir da visão e análise dos desenvolvedores de um sistema de dados de operações do mercado monetário. Para monitoramento e tratamento dos riscos identificados, foi proposto um modelo de classificação automática de códigos-fonte *Cobol*, e também um processo de especificação de componentes e funcionalidades de software, a fim de proporcionar maior agilidade e segurança ao processo de implementação. A metodologia adotada para a realização desse estudo, foi de natureza exploratória e descritiva, com a utilização do estudo de caso como estratégia de pesquisa e abordagens de pesquisa qualitativa (com a aplicação de questionários e entrevistas para a coletas de dados) e quantitativa (com a aplicação do método multicritério *Analytic Hierarchy Process* - AHP). Ao final é proposto um redesenho do processo de desenvolvimento, por meio do mapeamento de processos *TO BE* e BPM, em que são definidas 11 novas atividades para melhoria do processo de implementação a partir da elaboração de fonte confiável, atualizada e acessível de informação para as áreas técnicas e de negócio da instituição.

Palavras-chave: Implementação de *software*, ABNT NBR ISO 31000:2009, Mineração de Textos, BPM, AHP.

Abstract

Risk management constitutes an important area of knowledge for efficient software development process, as it allows found vulnerabilities is properly treated, enabling greater security and accuracy in decision making. An important factor in the success of risk management is the context of the establishment of what is to evaluate and improve within the organization. Therefore, to conduct this study, we used the standard ISO 31000: 2009 risk management, to propose improvements in the software implementation process of a financial institution, from the vision and analysis of the developers of a data system money market operations. For monitoring and treatment of identified risks, it proposed a automatic classification model of COBOL source code, and also a process of components and software features specification in order to provide greater flexibility and security to the implementation process. The methodology used to conduct this study was exploratory and descriptive, using the case study as a research strategy and qualitative research approaches (with the use of questionnaires and interviews for data collection) and quantitative (with the application of multi-criteria method Analytic Hierarchy Process - AHP). At the end it proposes a redesign of the development process, through process mapping TO BE and BPM, which are set 11 new activities to improve the implementation process from the preparation of reliable source, updated and accessible information for the technical areas and the institution's business.

Keywords: Software implementation, ABNT NBR ISO 31000:2009, Text Mining, BPM, AHP.

Sumário

Lista de Figuras

Lista de Tabelas

Lista de Abreviaturas e Siglas

ABNT	Associação Brasileira de Normas Técnicas
AHP	Analytic Hierarchy Process
ANSI	American National Standards Institute
BPM	Business Process Management
BPMN	Business Process Model and Notation
CASE	Computer Aided *Software* Engineering
CRISP-DM	Cross Industry Standard Process for Data Mining
CVS	Concurrent Versions System
DMAIC	Define-Measure-Analyse-Improve-Control
ERP	Enterprise Resource Planning
IEC	International Electrotechnical Commission
IEEE	Institute of Electrical and Electronic Engineers
ISO	International Organization for Standardization
PMBOK	Project Management Body of Knowledge
MDCA	Multi-Criteria Decision Analysis
MMO	Sistema de dados de Operações Mercado Monetário
MPS-BR	Melhoria de Processo de Software Brasileiro
NBR	Norma Brasileira Regulamentadora
RPO	Sistema de dados de Gestão Negocial de Operações
SIPOC	Supplier Input Process Output Customer
SIMPEP	Simpósio de Engenharia de Produção
SQL	Structured Query Language
SVM	Support Vector Machine
TI	Tecnologia da Informação
USA	United States of America

Introdução

Contextualização

As organizações vêm enfrentando uma variedade de problemas decorrentes da rápida evolução da Tecnologia da Informação (TI) e do descompasso causado pelo atraso na introdução dessa evolução nos modelos de gestão de negócios, em seus sistemas e aplicações. A prova disso é que a área de construção de *software* não consegue desenvolver aplicações na mesma velocidade em que as decisões são tomadas pela área de negócio.

Com o passar dos anos, os sistemas estão se tornando cada vez mais complexos e de difícil manutenibilidade [1]. E isso se deve ao grande volume de informações, ao aumento gradativo das funcionalidades e regras de negócios e à ausência de documentação adequada como fonte de consulta e divulgação do conhecimento na organização.

Os sistemas computacionais cada vez mais participam e colaboram das decisões negociais e estratégicas das organizações. Assim, alinhar os serviços de TI com os objetivos corporativos tem sido um grande desafio, pois a TI tornou-se responsável pela sustentação dos negócios das empresas e das necessidades de seus clientes.

Para propiciar um melhor gerenciamento dos negócios e de seus riscos, os gestores devem conhecer em detalhe os departamentos, as unidades, as gerências, os processos e/ou as funções que desejam realizar melhorias e modificações. Dessa forma torna-se necessário utilizar ferramentas e métodos que permitam conhecer seus processos e atividades, a fim de obter informações rápidas, eficientes e seguras para a tomada de decisão.

Mapear processos é indispensável para entender o funcionamento sistêmico dos processos da organização, de forma que se possa analisá-los por módulos e ter uma visão geral do início, meio e fim deles. Para isso, foi utilizado, neste trabalho, as ferramentas SIPOC e BPM para a representação dos processos, das atividades e das tarefas na implementação de *software*.

As atividades de identificação, análise, avaliação e controle de riscos nesse processo tornam-se cada vez mais necessárias na medida em que a sociedade necessita de novos produtos e serviços de TI. Contudo, essa não é uma tarefa simples, uma vez que a TI e seus processos estão em constante mudança e evolução.

Apesar da existência de vasta literatura sobre metodologias de desenvolvimento de *software*, ainda se verifica a existência de problemas e reincidentes falhas no processo que comprometem a qualidade dos produtos e serviços oferecidos aos clientes.

Nesse contexto, para auxiliar os gestores de TI a identificar quais atividades do processo de implementação de *software* têm maior influência e são mais determinantes na entrega de produto ou serviço de qualidade ao cliente é que este trabalho adotou um dos métodos de Auxílio à Decisão por Múltiplos Critérios (MDCA) mais conhecidos e utilizados no mundo: o *Analytic Hierachy Process* (AHP) [2].

O multicritério AHP foi utilizado com os objetivos de apresentar e aplicar a ferramenta na priorização do processo de implementação de *software*, a fim de obter a consolidação matemática resultante da avaliação de critérios intangíveis para apoio à tomada de decisão. No processo de implementação de uma instituição, o principal artefato de saída é o código-fonte produzido durante a atividade de codificação de unidades. A adoção de padrões na codificação de programas e de princípios, tais como baixo acoplamento e alta coesão, contribuem para o aumento da agilidade na manutenção de aplicações e no reúso de serviços já implementados [3].

Neste trabalho será proposto um modelo de classificação automática, baseado em mineração de textos, que permita identificar quais programas possuem codificação mais complexa e que deveriam ser reescritos, levando-se em consideração a execução de funções mais específicas (coesas), bem como menor dependência de outras aplicações.

As empresas são constantemente desafiadas a produzirem melhores resultados com menor custo, a desenvolverem produtos baseados em um ciclo de vida mais curto e a se relacionarem de forma mais integrada com seus clientes, fornecedores e parceiros.

Dessa forma, elas devem ser capazes de melhorar seus processos de negócio e sua comunicação com a área de TI, de quem dependem para viabilizar suas estratégias e garantir a sustentabilidade de seus negócios.

A necessidade de velocidade e eficiência nas entregas de soluções de *software* traz um grande desafio para a organização: o relacionamento das áreas de negócio da empresa com sua equipe de Tecnologia da Informação.

As empresas dependem de suas parceiras na área de tecnologia para projetarem seus novos produtos, para se relacionar com seus clientes e fornecedores, para conduzir e evoluir seus processos internos. Uma área de TI ágil pode contribuir significativamente para que a área de negócio não perca oportunidades e possa enfrentar os desafios do mercado.

Motivação

A motivação para a realização deste trabalho vieram dos desafios e dificuldades encontrados durante o desenvolvimento de um processo de integração de informações entre dois sistemas de dados de uma instituição financeira do Brasil, simbolicamente chamados de MMO[1] e RPO[2] neste estudo.

No início do desenvolvimento do processo de integração, foi possível verificar que não havia um documento formal de requisitos para subsidiar a construção do *software*. O que existia era um documento justificando a necessidade de se implementar a nova funcionalidade de integração e um detalhamento superficial sobre os campos e tabelas no sistema de destino que seriam responsáveis por armazenar as informações enviadas.

Durante a fase de desenvolvimento, a falta de conhecimento sobre o funcionamento, características, restrições e regras de negócio da aplicação, causaram grande insegurança e desconforto em sua realização, que só não foi superior à descoberta de que ambos os sistemas (MMO e RPO) não possuíam fonte de documentação atualizada sobre o funcionamento e fluxo de informações de seus processos de negócios.

Normalmente, nos Departamentos de TI, os sistemas possuem equipes especializadas, com funcionários experientes e conhecimento detalhado sobre a aplicação, que executam diferentes atividades de definição de requisitos, análise, projeto, implementação e testes de *software*.

Na equipe responsável pela manutenção do sistema MMO (sistema de dados de operações do mercado monetário), foi possível verificar que os funcionários eram experientes tecnicamente, porém nenhum deles possuía domínio suficiente sobre as funcionalidades e restrições da aplicação. Na área de negócio, a situação era semelhante, uma vez que os gestores de negócio[3] não conheciam o sistema suficientemente e, em consequência, demonstraram dificuldades em definir com detalhe os requisitos funcionais e não funcionais das necessidades da área de negócio.

Durante o processo de desenvolvimento, foram surgindo dúvidas em que os gestores demonstraram dificuldades em estabelecer definições e regras. Nessas situações era comum os gestores solicitarem aos desenvolvedores, que consultassem nos códigos-fonte do sistema, informações que pudessem auxiliar no direcionamento de suas definições, sem causar conflito e inconsistências ao que já existia em funcionamento na aplicação.

[1] MMO: sistema de dados responsável por contratar operações no mercado monetário, relacionados a títulos públicos federais.

[2] RPO: sistema de dados responsável por consolidar e divulgar os resultados e rentabilidade das operações contratadas pela instituição financeira com seus clientes.

[3] Gestores de negócio: funcionários responsáveis pela definição dos produtos e serviços das aplicações da instituição financeira.

Pensando na melhoria do trabalho realizado diariamente pelos desenvolvedores, buscou-se identificar a partir do processo de gestão de riscos da norma ABNT NBR ISO 31000:2009, as principais vulnerabilidades do processo de implementação de *software*, a fim de se analisar e avaliar as possíveis alternativas para tratamento e controle dos riscos identificados.

Descrição do Problema

Com o aumento da demanda por novos produtos e serviços, a área de desenvolvimento de *software* tem encontrado dificuldades em acompanhar a velocidade das mudanças e atender às necessidades dos clientes nos prazos acordados.

Os sistemas de processamento de dados de grandes instituições financeiras têm se tornando cada vez mais complexos e difíceis de serem mantidos pelos desenvolvedores. Com o passar dos tempos, novas necessidades foram surgindo e novas funcionalidades foram agregadas as aplicações, sem que isso fosse devidamente documentado.

Nas empresas verifica-se uma grande rotatividade dos funcionários, que, por motivos de aposentadoria, programas de demissão, promoções ou rodízios internos, causa interferência direta na produtividade, eficácia e resultados financeiros da organização.

A falta de documentação e a saída do pessoal técnico que participou inicialmente da modelagem do legado podem resultar em problemas que comprometem a qualidade e o prazo de entrega dos *softwares*, tais como dificuldades de compreensão das regras de negócio, desconhecimento das razões que levaram a determinadas decisões, problemas na estruturação dos programas ou a miscelânea de estilos de programação.

A saída de funcionários experientes, apesar de ser um problema antigo, ainda traz grandes prejuízos às organizações, uma vez que raramente elas possuem um processo de documentação de sistema eficiente para registro do conhecimento tácito dos funcionários.Ainda é comum existir sistemas críticos que são conduzidos apenas por um funcionário, ou até mesmo por funcionários de empresas terceirizadas, e que normalmente possuem peculiaridades que apenas o analista responsável sabe resolver, pois não existe documentação. Essa situação é comum tanto na área técnica de desenvolvimento de *software*, como na área de negócio responsável pelas mudanças e inovações das funcionalidades da aplicação.

Com a perda de conhecimento tácito em razão de afastamentos dos funcionários experientes, perde-se boa parte da qualidade, consistência e completude na elaboração dos artefatos de *software*, que consequentemente tornam-se incompletos, mal definidos e inconsistentes. Nessa situação é comum haver necessidade de realizar novas reuniões de definições de requisitos, com recorrentes consultas aos códigos-fonte, a fim de entender o comportamento e o funcionamento das aplicações.

Apesar de existir uma área responsável e especializada na elaboração de requisitos na instituição, os problemas ainda persistem. É comum, durante os projetos de *software*, ocorrerem interrupções no processo de implementação para consulta e esclarecimentos de situações não previstas inicialmente. Tal situação causa bastante desgaste entre a área de negócio e a área técnica, por ser esse um dos principais motivos de descumprimento nos prazos acordados de entrega dos *softwares*.

Objetivos

Objetivo Geral

Propor melhorias no processo de implementação de *software* de um sistema de dados de operações do mercado monetário, que contemple o processo de gestão de riscos da norma ABNT NBR ISO 31000:2009 e gerenciamento de processos de negócios (BPM).

Objetivos Específicos

1. Fazer o diagnóstico do processo de implementação de *software* de um sistema de dados de operações do mercado monetário de uma instituição financeira.
2. Propor um modelo de classificação automática da complexidade de códigos-fonte *Cobol*, a fim controlar e monitorar a qualidade e os riscos dos produtos produzidos pelo processo de implementação de *software* do sistema de dados analisado.
3. Propor melhorias na especificação de componentes e processo de desenvolvimento de *software*, a fim de aumentar a agilidade, segurança e tratamento dos riscos no processo de implementação do sistema de dados estudado.

Justificativa

A gerência de Gestão Estratégica de TI é a área responsável pela avaliação de riscos dos processos internos de desenvolvimento de *software* na instituição financeira. Desde que foi constituída em 2011, ela tem acompanhado a qualidade dos produtos e serviços de *software*, através do processo de verificação da qualidade dos documentos produzidos ao final da implantação dos projetos e demandas de TI.

No entanto, apesar desse processo ter contribuído para a melhoria da qualidade dos sistemas de modo geral, ele ainda se mostra pouco eficiente quanto ao gerenciamento e controle da qualidade do conteúdo de informações existentes no artefatos, a fim de serem

utilizados como fonte de consulta atualizada do funcionamento e comportamento dos sistemas.

Ao longo desta pesquisa, observou-se que os documentos produzidos pela área de TI durante o processo de desenvolvimento dos *softwares* não eram compartilhados com a área de negócios. Nesse contexto, sempre que existe a necessidade de se alterar as regras de negócio, ou criar novas funcionalidades na aplicação, novos documentos são gerados, sem rastreabilidade e/ou atualização dos documentos existentes. Essa situação onera o processo de desenvolvimento, pois causa muito retrabalho e aumento significativo no prazo de entrega do *software*. A rotatividade de funcionários e o dinamismo das regras de negócios também tornam necessário um processo de documentação de sistemas, a fim de suprir tanto a necessidade da área de TI quanto a área de negócio de documentos e informações atualizadas dos objetivos, características, restrições, necessidades e funções do sistema.

Diante disso, torna-se imprescindível buscar, tanto no mercado quanto no meio acadêmico, soluções que proporcionem melhorias no processo de documentação do negócio e de implementação dos sistemas, a fim de gerenciar riscos como: alta dependência do conhecimento tácito, documentação inadequada do sistema, dificuldade em avaliar custos de alterações e evoluções do sistema, consumo elevado de recursos tecnológicos, redundância de código e componentes com alto acoplamento.

Estrutura do Trabalho

O trabalho está estruturado em cinco capítulos:

Capítulo 1 - inicia a revisão bibliográfica com o objetivo de fundamentar teoricamente o estudo. Este capítulo faz uma introdução sobre as principais áreas da engenharia de *software*, aborda as normas e modelos de qualidade e desenvolvimento de *software*, aborda as normas de gestão de riscos e gerenciamento de processos de negócio e também faz referência aos métodos utilizados para o alcance dos objetivos geral e específicos deste estudo, mediante a utilização de ferramentas e técnicas como: SIPOC, Método de Apoio à Decisão Multicritério AHP, mineração de textos, Matriz de Probabilidade e Impacto e Índice de Riscos, bem como apresentando-se, ainda, as abordagens de estudiosos nacionais e estrangeiros.

Capítulo 2 - neste capítulo é apresentada a metodologia com objetivo, a estratégia e a abordagem de pesquisa, bem como a estrutura de desenvolvimento do presente estudo, detalhando os métodos, técnicas e ferramentas que foram utilizados para o alcance dos objetivos específicos e geral estabelecidos.

Capítulo 3 - realiza um estudo do processo de implementação de *software* utilizado no sistema de dados de operações do mercado monetário, a fim de realizar um diagnóstico do processo de implementação de *software* para identificar, analisar e avaliar as vulnerabilidades que possam comprometer a entrega de um *software* de acordo com o esperado pelo cliente.

Capítulo 4 - apresenta uma proposta de classificação automática de programas *Cobol*, baseado em mineração de textos, para controle e monitoramento de riscos no processo de implementação de *software*, ao identificar programas complexos de difícil manutenção.

Capítulo 5 - apresenta o uso do modelo de gerenciamento de processos de negócio (BPM) e a notação BPMN, como ferramenta para a elaboração documentação de sistema e especificação de componentes de *software* para tratamentos de riscos identificados no processo de implementação. Neste capítulo também é apresentada uma proposta de redesenho do processo de desenvolvimento de *software* por intermédio do mapeamento de processos *TO BE*, para a inclusão de novas atividades referentes aos resultados obtidos nos capítulos 4 e 5 na realização das atividades de tratamento e monitoramento de riscos da norma ABNT NBR ISO 31000:2009 - Gestão de Riscos: Princípios e Diretrizes.

Capítulo 1

Revisão de Literatura

Neste capítulo é apresentada uma revisão bibliográfica sobre engenharia de *software* e suas principais área de conhecimento, gerenciamento de processos de negócio, gestão de riscos e mineração de textos, estabelecendo o contexto em que esse trabalho se insere.

1.1 Engenharia de *Software*

Atualmente, as empresas produtoras de *software* têm perseguido um objetivo em comum: produzir *software* com alto nível de qualidade. A preocupação com qualidade deixou de ser um diferencial competitivo e passou a ser um pré requisito básico para participação ativa no mercado [4]. É neste contexto que a engenharia de *software* tem ganhado espaço dentro das organizações, contribuindo com métodos, ferramentas e metodologias avançadas para a obtenção de nível de qualidade. A engenharia de *software* pode ser definida como a aplicação de uma metodologia sistemática, disciplinada e quantificável para o desenvolvimento, a operação e a manutenção do *software* [5].

A engenharia de *software* é uma tecnologia em camadas, que engloba processos, métodos e ferramentas que possibilitam a construção de sistemas complexos baseados em computador dentro do prazo e com qualidade [6]. O processo define uma metodologia que deve ser estabelecida para a entrega efetiva de tecnologia de engenharia de *software*.

O processo de *software* constitui a base para controle do gerenciamento de projetos de *software* e estabelece o contexto no qual são aplicados métodos técnicos, são produzidos produtos derivados (documentos, relatórios, dados, formulários), são estabelecidos marcos e mudanças são geridas de forma apropriada.

Os métodos da engenharia de *software* fornecem os as informações técnicas para desenvolver o *software*. Os métodos envolvem uma ampla gama de tarefas, que incluem comunicação, análise de requisitos, modelagem de projeto, construção, testes e suporte [6]. Os métodos de engenharia de *software* baseiam-se em um conjunto de princípios

básicos que governam cada área da tecnologia e inclui atividades de modelagem e outra técnicas descritivas.

As ferramentas da engenharia de *software* fornecem suporte automatizado ou semi-automatizado para o processo e métodos. Quando as ferramentas são integradas, de forma que as informações criadas por uma ferramenta possam ser usadas por outras, é estabelecido um sistema para o suporte e desenvolvimento de *software*.

O IEEE Compute Society, através do guia do conjunto de conhecimentos da engenharia de *software* [5], estabelece dez áreas de conhecimento da engenharia de *software*. São elas: requisitos de *software*, design de *software*, construção de *software*, teste de *software*, manutenção de *software*, gerenciamento de configuração, gerenciamento de engenharia de *software*, modelo de engenharia de *software*, ferramentas e métodos da engenharia de *software* e qualidade de *software*.

1.1.1 Requisitos de *software*

A área de requisitos auxilia os engenheiros de *software* a compreender melhor os problemas relacionados ao levantamento e ao gerenciamento das informações fornecidas pelos usuários de *software*. Um requisito de *software* é uma descrição dos principais recursos de um produto de *software*, seu fluxo de informações, comportamentos e atributos, fornecendo uma estrutura básica para o desenvolvimento de um produto de *software* [7].

1.1.2 Design de *software*

O design ou projeto de *software* começa quando a primeira iteração da engenharia de requisitos é concluída. O objetivo do projeto de *software* é aplicar um conjunto de princípios, conceitos e práticas que levam ao desenvolvimento de um sistema ou produto de alta qualidade [8].

1.1.3 Construção de *software*

Esta área de conhecimento da engenharia de *software* trata dos conceitos relacionados à criação de *software* executável, os quais incluem a codificação, verificação, testes de unidade e de integração, e depuração. Possui ligações com todas as demais áreas de conhecimento, mas principalmente com as áreas de design e testes, pois muito do processo de construção de *software* trata de atividades relacionadas a essas áreas [5].

1.1.4 Teste de *software*

A área de conhecimento de testes de *software* compreende as atividades desenvolvidas para a avaliação da qualidade do *software*, através da identificação de defeitos e problemas. Essa verificação compreende a execução de um conjunto finito de casos de teste, adequadamente retirados de um domínio geralmente infinito de possibilidades, e a sua comparação com os resultados esperados [5].

1.1.5 Manutenção de *Software*

A manutenção de *software* é uma atividade importante no ciclo de vida de um *software* pois é o que consome a maior parte dos recursos humanos e financeiros [5]. A falta de habilidade em mudar um *software* rapidamente e de maneira confiável, pode resultar na perda de oportunidade de negócios [9].

1.1.6 Gerenciamento de engenharia de *software*

O gerenciamento da engenharia de *software* pode ser definido como a aplicação do gerenciamento das atividades, do planejamento, da coordenação, da mensuração, do monitoramento, do controle e do nível de reportagem, para assegurar que o desenvolvimento e a manutenção do *software* é sistemática, disciplinada e qualificada [5]. O processo de desenvolvimento de *software* é complexo e abrangente com o objetivo de garantir que ele funcione corretamente, torna-se necessário gerenciá-lo.

1.1.7 Gerenciamento de configuração

É o desenvolvimento e aplicação de padrões e procedimentos para gerenciar um produto de sistema em desenvolvimento [10]. É necessário gerenciar os sistemas em desenvolvimento porque, à medida que eles se desenvolvem, são criadas muitas diferentes versões de *software*. Com o controle de versões, conservam-se versões antigas dos artefatos, que contém material que pode que pode vir a ser aproveitado novamente.

1.1.8 Ferramentas e métodos da engenharia de *Software*

Com o aumento da complexidade dos processos de *software*, passou a ser imprescindível o uso de ferramentas e ambientes de apoio a realização de suas atividades, visando a atingir níveis mais alto de qualidade e produtividades. As ferramentas CASE (Computer Aided *Software* Engineering) podem ser classificadas por função, por seus papéis como instrumentos para os gerentes e para o pessoal técnico, pelo uso que elas têm nas várias etapas do processo de engenharia de *software* [8].

1.1.9 Qualidade de *Software*

Qualidade é, fundamentalmente, a adequação ao uso; ou ainda, a totalidade das características de um produto ou serviço que se relacionam com sua capacidade de atender às necessidades de um cliente [11]. A qualidade deve ser incorporada a um produto ou serviço e requer a conjugação de esforços de todos os membros de uma empresa, a integração de todas as funções e recursos que ela dispõe, desde a alta administração até o ma is simples empregado. A estruturação de recursos, métodos, passos e equipamentos envolvidos com a qualidade constitui o sistema de qualidade da empresa.

Uma definição mais abrangente foi dada pela norma ISO 9001:2000 [12] onde qualidade é definida como o grau em que um conjunto de características inerentes cumprem sua exigências.

1.1.10 Processo de engenharia de *software*

A definição do processo é uma fase importante, onde é verificado qual modelo de processo ou ciclo de vida será adotado. Atualmente vários são utilizados e os mais comuns são: cascata também conhecido por clássico, espiral, incremental, prototipação e espiral para círculo. Um processo de engenharia de *software* é uma representação abstrata de um processo para explicar as diferentes abordagens de desenvolvimento (ciclo de vida) [10].

Na próxima seção é apresentado maiores detalhes sobre o processo e processo de *software*, os conceitos e definições sobre ciclo de vida, bem como a importância e impacto da sua escolha nas fases do processo de desenvolvimento, e também métodos para auxiliar na gestão e acompanhamento desses processos.

1.2 Processo

Processo é qualquer atividade ou conjunto de atividades que toma uma entrada, adiciona valor e fornece uma saída. Os processos utilizam recurso da organização para oferecer resultados objetivos a seus clientes [13], mais formalmente, um processo é um grupo de atividades realizadas numa sequência lógica com o objetivo de produzir um bem ou um serviço que tem valor para um grupo específico de clientes [14].

Os processos vêm sendo aperfeiçoados ao longo do tempo em função do surgimento de novas tecnologias e das novas exigências do mercado. O efetivo conhecimento dos processos proporciona vantagens que vão da gestão eficaz do conhecimento sobre o que e como a empresa realiza ao entendimento mais preciso de requisitos que os usuários definem para os produtos e serviços que são prestados.

Um processo trata de uma série de ações, passos ou procedimentos que conduzem a um resultado [15]. Em alto nível é uma sequência ou fluxo de tarefas realizadas durante uma produção, ou a entrega de um serviço.

Na engenharia de *software*, o processo tem como meta entregar um produto de *software*, de maneira eficiente, previsível, e que corresponda às necessidades do negócio. Geralmente inclui análise de requisitos, programação, testes, entre outras tarefas.

O propósito da orientação por processos é alcançar melhorias em custo, tempo e qualidade, dando a organização, flexibilidade e habilidade de mudança. Organizações hierárquicas tendem a ser estáveis e inflexíveis, enquanto que as orientadas por processos podem agir rapidamente, conforme o ambiente [16].

Na próxima seção será apresentada maiores detalhes sobre o processo de *software*, como ele está organizado e subdivido, a fim de compreender e encontrar as melhores alternativas para seu controle e gerenciamento.

1.2.1 Processo de Software

Um processo de *software* pode ser visto como o conjunto de atividades, métodos, práticas e transformações que guiam pessoas na produção de *software*. Um processo eficaz deve, claramente, considerar as relações entre as atividades, os artefatos produzidos no desenvolvimento, as ferramentas e os procedimentos necessários e a habilidade, o treinamento e a motivação do pessoal envolvido.

Elementos que compõem um processo de *software*: processos, atividades, artefatos (insumos, produtos), recursos (recursos humanos, ferramentas de *software*, hardware), procedimentos (métodos, técnicas e roteiros) .

Há vários aspectos a serem considerados na definição de um processo de *software*. No centro da arquitetura de um processo de desenvolvimento estão as atividades-chave desse processo: análise e especificação de requisitos, projeto, implementação e testes, que são a base sobre a qual o processo de desenvolvimento deve ser construído. Entretanto, a definição de um processo envolve a escolha de um modelo de ciclo de vida, o detalhamento (decomposição) de suas macro-atividades, a escolha de métodos, técnicas e roteiros (procedimentos) para a sua realização e a definição de recursos e artefatos necessários e produzidos [10].

Um processo de *software* não pode ser definido de forma universal. Para ser eficaz e conduzir à construção de produtos de boa qualidade, um processo deve ser adequado ao domínio da aplicação e ao projeto específico. Deste modo, processos devem ser definidos caso a caso, considerando-se as especificidades da aplicação, a tecnologia a ser adotada na sua construção, a organização onde o produto será desenvolvido e o grupo de desenvolvimento [10].

Em suma, o objetivo de se definir um processo de *software* é favorecer a produção de sistemas de alta qualidade, atingindo as necessidades dos usuários finais, dentro de um cronograma e um orçamento previsíveis.

A escolha de um modelo de ciclo de vida (ou modelo de processo) é o ponto de partida para a definição de um processo de desenvolvimento de *software*. Um modelo de ciclo de vida organiza as macro-atividades básicas, estabelecendo precedência e dependência entre as mesmas.

Um modelo de ciclo de vida pode ser entendido como passos ou atividades que devem ser executados durante um projeto. Para a definição completa do processo, a cada atividade, devem ser associados técnicas, ferramentas e critérios de qualidade, entre outros, formando uma base sólida para o desenvolvimento. Adicionalmente, outras atividades tipicamente de cunho gerencial, devem ser definidas, entre elas atividade de gerência e de controle e garantia da qualidade.

De maneira geral, o ciclo de vida de um *software* envolve as seguintes fases [1]:

Planejamento: O objetivo do planejamento é fornecer uma estrutura que possibilite ao gerente fazer estimativas razoáveis de recursos, custos e prazos. Uma vez estabelecido o escopo de *software*, uma proposta de desenvolvimento deve ser elaborada, isto é, um plano de projeto deve ser elaborado configurando o processoa ser utilizado no desenvolvimento de *software*. À medida que o projeto progride, o planejamento deve ser detalhado e atualizado regularmente. Pelo menos ao final de cada uma das fases do desenvolvimento (análise e especificação de requisitos, projeto, implementação e testes), o planejamento como um todo deve ser revisto e o planejamento da etapa seguinte deve ser detalhado. O planejamento e o acompanhamento do progresso fazem parte do processo de gerência de projeto.

Requisitos: Nesta fase, o processo de levantamento de requisitos é intensificado. O escopo deve ser refinado e os requisitos identificados. Para entender a natureza do *software* a ser construído, o engenheiro de *software* tem de compreender o domínio do problema, bem como a funcionalidade e o comportamento esperados. Uma vez identificados os requisitos do sistema a ser desenvolvido, estes devem ser modelados, avaliados e documentados. Uma parte vital desta fase é a construção de um modelo descrevendo o que o *software* tem de fazer.

Projeto e Análise: Esta fase é responsável por incorporar requisitos tecnológicos aos requisitos essenciais do sistema, modelados na fase anterior e, portanto, requer que a plataforma de implementação seja conhecida. Basicamente, envolve duas grandes etapas:

projeto da arquitetura do sistema e projeto detalhado. O objetivo da primeira etapa é definir a arquitetura geral do *software*, tendo por base o modelo construído na fase de análise de requisitos. Esta arquitetura deve descrever a estrutura de nível mais alto da aplicação e identificar seus principais componentes. O propósito do projeto detalhado é detalhar o projeto do *software* para cada componente identificado na etapa anterior. Os componentes de *software* devem ser sucessivamente refinados em níveis de maior detalhamento, até que possam ser codificados e testados.

Implementação: o projeto deve ser traduzido para uma forma passível de execução pela máquina. A fase de implementação realiza esta tarefa, isto é, cada unidade de *software* do projeto detalhado é implementada.

Testes: inclui diversos níveis de testes, a saber, teste de unidade, teste de integração e teste de sistema. Inicialmente, cada unidade de *software* implementada deve ser testada e os resultados documentados. A seguir, os diversos

componentes devem ser integrados sucessivamente até se obter o sistema. Finalmente, o sistema como um todo deve ser testado.

Implantação: uma vez testado, o *software* deve ser colocado em produção. Para tal, contudo, é necessário treinar os usuários, configurar o ambiente de produção e, muitas vezes, converter bases de dados. O propósito desta fase é estabelecer que o *software* satisfaz os requisitos dos usuários. Isto é feito instalando o *software* e conduzindo testes de aceitação (validação). Quando o *software* tiver demonstrado prover as capacidades requeridas, ele pode ser aceito e a operação iniciada.

Operação: nesta fase, o *software* é utilizado pelos usuários no ambiente de produção.

Manutenção: Indubitavelmente, o *software* sofrerá mudanças após ter sido entregue para o usuário. Alterações ocorrerão porque erros foram encontrados, porque o *software* precisa ser adaptado para acomodar mudanças em seu ambiente externo, ou porque o cliente necessita de funcionalidade adicional ou aumento de desempenho. Muitas vezes, dependendo do tipo e porte da manutenção necessária, essa fase pode requerer a definição de um novo processo, onde cada uma das fases precedentes é reaplicada no contexto de um *software* existente ao invés de um novo.

São fatores que influenciam a definição de um processo: tipo de *Software* (sistema de informação, sistema de tempo real), paradigma (estruturado, orientado a objetos), domínio da aplicação, tamanho e complexidade, características da equipe e outros.

Embora diferentes projetos requeiram processos com características específicas para atender às suas particularidades, é possível estabelecer um conjunto de ativos de processo (subprocessos, atividades, sub-atividades, artefatos, recursos e procedimentos) a ser utilizado na definição de processos de *software* de uma organização. Essas coleções de ativos de processo de *software* constituem os chamados processos padrão de desenvolvimento de *software*.

1.2.2 Gestão por Processos

A gestão por processos tem como objetivo maximizar os resultados dos processos, aumentar a satisfação dos clientes, otimizar os recursos e reduzir custos.

Nas organizações tradicionais, cada área entende, trata e gerencia o processo da sua função específica. Os funcionários se restringem apenas à aquilo que estão fazendo, dificilmente conseguem visionar a organização no todo. Com isso acabam restritos às suas próprias funções, dando prioridade á apenas aquilo que são cobradas. As decisões acontecem verticalmente, havendo uma centralização de poder.

Essas empresas priorizam as funções em detrimento dos processos e exageram na divisão de tarefas, com atividades padronizadas, controladas por vários níveis de chefia, cuja função principal é garantir o cumprimento das normas [17].

Surge então uma nova abordagem de gestão, onde a organização é vista de uma forma mais abrangente com o foco nos processos ou pelo menos no híbrido composto funções-processos. A função-processo é predominante na grande maioria das organizações, onde pode se observar de um lado, a estrutura vertical de funções empresariais e de outro lado a estrutura horizontal de processos de negócios, com ambas disputando os mesmos recursos.

1.2.3 Principais Técnicas para Mapeamento de Processos

A literatura apresenta algumas técnicas de mapeamento com diferentes enfoques tornando a correta interpretação destas técnicas fundamental no processo de mapeamento [18].

Dentre as diversas técnicas de mapeamento podemos citar:

- SIPOC: é uma ferramenta usada por um time para identificar todos os elementos pertinentes de um projeto de melhoria de processo antes de o trabalho começar [19].
- Blueprinting: representa um fluxograma de todas as transações integrantes do processo de prestação de serviço [20].
- Fluxograma: técnica que permite o registro de ações de algum tipo e pontos de tomada de decisão que ocorrem no fluxo real.

- Mapofluxograma: é um fluxograma desenhado sobre a planta de um edifício ou layout para visualizar melhor o processo [21].
- Diagrama homem-máquina: tem por objetivo o estudo da inter-relação entre o trabalho [21]. do homem e o da máquina, identificando os tempos ociosos de ambos e balanceando a atividade do posto de trabalho.

Para auxiliar no mapeamento de processos realizado nesse estudo, no próximo item é apresentado com maiores detalhes o SIPOC que é uma ferramenta que proporciona visão de alto nível das partes que se relacionam com o processo.

1.2.4 SIPOC - Supplier, Input, Process, Output e Costumer

O diagrama SIPOC é uma ferramenta utilizada para identificar partes relevantes de um sistema para a melhoria de um processo, através da construção de um mapa de processo. O objetivo é criar uma representação de como o processo funciona, para verificar quais os pontos não estão de acordo com o programado para funcionar [22].

Este mapa define somente as atividades principais do processo, não detalhando pontos de decisão, regras de negócio [23].

A sigla SIPOC resulta das iniciais de Suppliers, Input, Process, Output, Client e tem seus significados da seguinte forma:

Técnica de análise de processo, conseguida através de cinco parâmetros [24]:

- Fornecedores (Suppliers): partes envolvidas que providenciam as entradas no processo (Input);
- Entradas (Input): recursos necessários para que o processo (Process), gere as saídas pretendidas (Ouput);
- Processos (Process): as atividades que transformam as entradas em saídas;
- Saídas (Output): o resultado do processo;
- Clientes (Customers): as partes envolvidas que recebem as saídas do processo.

A definição do processo a ser mapeado deve ser estabelecidos de forma cuidadosa, para que o limite não ultrapasse o escopo definido para análise [22].

As entradas fornecidas aos processos são os materiais, informações ou outros recursos que são trabalhados pelo processo, sendo consumidos ou transformados em saídas para o cliente. Devem-se concentrar nas entradas mais críticas, as quais são essenciais ao trabalho, pois sem as mesmas, o processo não seria realizado da maneira correta [23].

Para o parâmetro *Processo* o autor [22]sugere que sejam utilizados de 5 a 7 passos, enquanto que [23] sugere o número máximo de 10 passos, para evitar muito detalhamento e complexidade na análise do SIPOC.

O diagrama de SIPOC é definida como uma ferramenta usada por um time para identificar todos os elementos pertinentes de um projeto de melhoria de processo antes de o trabalho começar [25]. A ferramenta de SIPOC é particularmente útil quando não estiver claro: quem provê contribuições ao processo, que especificações são colocadas nas contribuições, quem são os verdadeiros clientes do processo, o que são as exigências dos clientes [25].

O SIPOC é uma das ferramentas mais adequadas a serem utilizadas para definir adequadamente o problema, que tem como objetivo a identificação das fronteiras do projeto, isto é, quais os fornecedores e clientes do processo em estudo, e também quais as principais *entradas* a serem processadas e a relação das características mais críticas aos clientes quanto às *saídas* geradas [19].

O SIPOC é também uma das ferramentas recomendadas pelo método DMAIC (Define-Measure-Analyse-Improve-Control) do programa de qualidade Seis Sigma [26], na busca pelo aumento da performance, produtividade e competitividade das organizações.

Nota-se que o uso do SIPOC tem se intensificado em diferentes segmentos e setores das organizações. Cada vez mais as empresas utilizam a ferramenta na identificação de processos críticos, bem como para conhecer o fluxo das atividades e entender de forma clara e rápida os processos.

Nesse contexto podemos ver exemplos de uso do SIPOC no mapeamento de processos de diferentes organizações: Direção Geral de Alfândegas e Imposto Especial do Consumo [27], setor financeiro prestador de seguros [28], serviços oftalmológicos [29] e outros.

1.2.5 Gerenciamento de Processo de Negócio (BPM)

Business Process Management (BPM) é um conjunto de múltiplos elementos, conceitos e metodologias que tem a finalidade de tratar processos de negócio de forma holística [30].

Os processos, em função da importância e do que realizam, podem receber diferentes classificações como elementar e complexo, de apoio e finalístico, estratégico ou de negócio.

O conhecimento em diversas áreas do saber relacionado com as atividades que a empresa desenvolve constitui vantagem competitiva e talvez o ativo mais importante das instituições. Do ponto de vista da gestão o conhecimento dos processos é importante na tomada de decisões em todos os níveis.

O monitoramento constante dos processos possibilita a alimentação de indicadores que orientam as decisões e ações necessárias para assegurar o desempenho de toda a empresa.

Processos de negócio traduzem as iniciativas definidas no planejamento estratégico em valor para os stakeholders.

O gerenciamento de processos é uma metodologia empregada para definir, analisar e gerenciar as melhorias no desempenho dos processos da empresa, com a finalidade de atingir as condições ótimas para o cliente [31]. O gerenciamento de processos é parte de um programa abrangente que tem o objetivo de fortalecer a competitividade da empresa.

Este tipo de abordagem conduz a empresa ao aumento global da qualidade e produtividade no atendimento dos requisitos dos clientes.

O conceito de BPM é suportado por dois conjuntos de conhecimentos [30]. O primeiro engloba teorias, normas, políticas e metodologias pertinentes à análise, desenho, redesenho, modelagem, organização, implantação, gerenciamento e melhoria de processos. O segundo contempla a infraestrutura que possibilita operacionalizar o primeiro.

Como metodologia, a gestão de processos possibilita desenvolver ao máximo as potencialidades internas, e assim, aumentar a eficiência e os resultados gerais. Pode-se dizer que a gestão de processos está relacionada à organização dos processos da empresa, sendo recomendado o suporte da infraestrutura tecnológica.

Etapas do mapeamento de processos

O mapeamento de processos busca entender os processos de negócios existentes e futuros para otimizá-los e maximizar o valor que é agregado ao produto e, assim, proporcionar mais satisfação ao cliente e melhores retornos para a instituição.

O mapeamento de processos compreende as fases apresentadas na Tabela 1.1 [30]:

Tabela 1.1: Fases do BPM

Fase	Descrição
Análise inicial das necessidades (ou do problema)	• Nesta fase busca-se saber as expectativas da organização, quantidade de pessoas envolvidas com o processo, área funcionais envolvidas, etc.
Documentação, desenho e análise do processo atual	• Fundamentalmente nesta fase elabora-se a documentação do processo, o que possibilita que se torne amplamente conhecido, analisado e entendido.

Análise e redesenho	• Modelagem do novo processo.
Implantação do novo processo	• Esta fase contempla o treinamento das pessoas para assegurar a qualidade do processo.
Gerenciamento do processo	• Execução do ciclo Plan–Do–Check–Act (PDCA).

Fonte: CRUZ [30]

Abordagens de Gerenciamento de Processos por Negócios

A modelagem de processos é a fase mais visível do BPM. E ela pode ser do tipo *Top Down* e *Botton Up*.

A abordagem *Top Down* ocorre quando a iniciativa do projeto é a nível da organização, unidade de negócios, ou de uma aplicação. Se a iniciativa está no nível de empresa, a identificação dos processos e serviços deve ser conduzida pela análise da cadeia de valor dos negócios e dos objetivos estratégicos.

Na abordagem *Botton Up*, o mapeamento acontece em nível operacional a partir de uma uma funcionalidade já existente, em que se deseja conhecer o seu funcionamento em nível de dados até se alcançar os negócios e funcionalidades que retratam a visão estratégica da organização.

Como as organizações operam em ambientes muito semelhantes, e em muitos casos utilizam as mesmas tecnologias, o diferencial depende muitas vezes está no quanto são eficientes os seus processos, e na competência de seus colaboradores. Neste contexto, o BPM foi escolhido neste estudo, como instrumento para promover a melhoria no desempenho dos gestores e desenvolvedores na realização de suas atividades, bem como na realização de seus papéis e responsabilidades.

Em um mapeamento de processos envolvendo abordagens *Botton Up* e *Top Down*, o primeiro passo é escolher o processo de negócio que se deseja detalhar. Logo após deve-se identificar as atividades e tarefas que formam este processo. O segundo passo seria identificar os pontos de funcionalidade existentes no sistema. As funcionalidades oferecidas são expostas como subprocessos reutilizáveis, e estes atuam como pontos de interação com outros sistemas e plataformas existentes [32].

Abordagem *Top Down*

É importante que os gestores de negócio e analistas envolvidos tenham completo entendimento da aplicação para fazerem uma análise da cadeia de valor da aplicação. Este entendimento permite decidir o que é crítico e o que não é crítico para que os benefícios do modelo sejam rapidamente alcançados e que isto se converta em vantagem competitiva no mercado.

É também importante que as pessoas envolvidas conheçam as funcionalidades do sistema para identificar os serviços de uma determinada função e torná-las disponíveis aos processos de negócio. A análise da cadeia de valor compreende as seguintes atividades [32]:

- compreender as funcionalidades e como elas interagem um com os outros
- detalhar os processos de negócio e suas interações
- detalhar subprocessos
- identificar em alto nível as tarefas/atividades das funcionalidades possíveis candidatas a serviço.

O processo *AS IS* do BPM permite mapear os sistemas de TI e entender seu contexto e limites. Realizar o mapeamento *AS IS* favorece a identificação das tarefas e atividades realizadas em cada uma das funcionalidades, levando a melhor compreensão do sistema e na definição de um mapa geral da aplicação. Segue as vantagens em documentar o sistema por processos de negócio [32]:

- permite aos analistas entender o contexto de negócios
- preenche a lacuna entre a área de TI e de negócio
- permite elaborar uma lista preliminar de atividades possíveis candidatas a reutilização, para posterior análise.
- identificação de dependência entre atividades em alto nível
- identificação de regras de negócio a partir de caixa de decisão. Isto permitirá aos analistas identificar as regras voláteis da aplicação, ou seja que se alteram com certa frequência.
- iterações entre diferentes pessoas da empresa para as necessidades do fluxo de trabalho

  Após este mapeamento inicial é elaborado uma proposta de redesenho buscando-se aperfeiçoar os processos existentes, o que é chamado de *TO BE*. Este novo desenho

é elaborado buscando-se melhorias nos processos e atividades realizados, em termos de controle, agilidade, eficácia e outros.

Abordagem *Botton Up*

A abordagem *Botton Up* envolve a organização na análise de suas aplicações, avaliação das funcionalidades, atividades e tarefas para reutilização e verificação de redundância. Em geral o objetivo é identificar nos programas redundância de código, entidades de dados de negócios, que resultam em elevados custos operacionais e de manutenção. Identificação de serviços usando essa abordagem pode ou não contribuir para construir um processo de negócio. Mas ainda assim a contribuição desses serviços quanto ao retorno sobre o investimento na linha de reutilização pode ser alto em um determinado cenário.

Esta abordagem seria a mais indicada para a realização de projetos de saneamento, que tem por objetivo promover melhorias nos sistemas, na identificação de programas complexos com muitas dependências a outras aplicações, bem como repetição de código (regras de negócio) em diferentes programas do sistema.

Para apoiar a definição de processos, na próxima seção são apresentadas diversas normas e modelos de qualidade de processo de *software* foram propostas, dentre elas [33]: ISO/IEC 12207, ISO/IEC 15504, MPS-BR. O objetivo dessas normas e modelos de qualidade é apontar características que um bom processo de *software* tem de apresentar, deixando a organização livre para estruturar essas características segundo sua própria cultura.

1.3 Modelos e Normas de Desenvolvimento e Qualidade de *Software*

Com o crescimento do setor de *software*, vários padrões tem sido propostos ao longo dos últimos anos para melhoria do processo de desenvolvimento de *software* [34]. A International Organization for Standardization (ISO) é uma organização não governamental internacional, cuja existência foi motivada pela necessidade de referências para regulamentar obrigações contratuais entre fornecedores e compradores com foco na garantia da qualidade de produtos. Padrões e Normas além de promover a normatização de produtos e serviços *software*, também servem para definir as atividades, a estrutura, organização, artefatos e recursos necessários para realização dos processos. A busca por melhoria da qualidade, faz com que as organizações avaliem e estudem propostas de desenvolvimento, métodos e aplicações que entreguem aos usuários um *software* de qualidade.

As normas ISO há muito tempo são relacionadas à qualidade de *software*, e várias delas são utilizadas como guias e padrões para diversas áreas de atuação dentro do contexto da ciência da computação. Dentre as normas e modelos utilizados no Departamento de TI do sistema MMO como padrão para desenvolvimento de *software*, vale destacar: NBR ISO IEC 12207, 15504 e MPS.BR.

1.3.1 ISO/IEC 12207

A norma internacional ISO/IEC 12207 – Engenharia de Sistemas e *Software* – Processos de Ciclo de Vida de *Software* [35] estabelece um conjunto comum de processos do ciclo de vida de *software* e uma terminologia bem definida para facilitar a comunicação entre os envolvidos com o desenvolvimento de *software*. Ela tem por objetivo auxiliar os envolvidos na produção de *software* a definir seus papéis, por meio de processos bem definidos [33]. Esta norma contém processos que são utilizados durante todo o desenvolvimento de *software*, envolvendo a aquisição, o fornecimento, o desenvolvimento, a operação, a manutenção e a descontinuação do *software*.

Cada processo apresentado nesta norma é descrito em termos de seu propósito e resultados esperados, bem como das atividades e tarefas necessárias para executar o processo e alcançar seus resultados.

Esta norma divide os processos em três grandes classes: Processos fundamentais, Processos de apoio e Processos organizacionais.

- Processos fundamentais são compostos pelos processos de manutenção, aquisição, fornecimento, desenvolvimento e operação, responsáveis pelo início e execução do desenvolvimento, operação ou manutenção do *software* durante o seu ciclo de vida.
- Processos de apoio são compostos pelos processos de documentação, gerência de configuração, garantia da qualidade, verificação, validação, revisão conjunta, auditoria, e resolução dos problemas que têm o papel de auxiliar um outro processo.
- Processos organizacionais são compostos pelos processos de gerência, de infraestrutura, de melhoria e de treinamento que implementam uma estrutura constituída de processos de ciclo de vida e de pessoal associados, melhorando continuamente a estrutura e os processos.

A ISO/IEC 12207 apresenta um detalhamento de cada um dos processos acima. Define como podem ser usados de diferentes maneiras por diferentes organizações (ou parte destas), representando diversos pontos de vista para esta utilização [36]. Estas visões representam a forma como uma organização pode utilizar estes processos, agrupando-os de acordo com suas finalidades e necessidades: Visão de contrato, Visão operacional, Visão de engenharia, Visão de equipe de apoio.

1.3.2 ISO/IEC 15504

A norma internacional ISO/IEC 15504 – Avaliação de Processos (ISO/IEC, 2003) provê uma abordagem estruturada para a avaliação de processos, com o intuito de identificar a capacidade e a maturidade dos processos nas organizações.

A capacidade dos processos, de acordo com esta norma, é definida em seis níveis [37]. Cada nível representa a capacidade do processo em atender seu objetivo. Sendo assim, um processo, ao ser avaliado, pode se enquadrar em um dos seguintes níveis: nível 0 – Incompleto; nível 1 – Executado; nível 2 – Gerenciado; nível 3 – Estabelecido; nível 4 – Previsível; e nível 5 – Otimizado.

Se o objetivo for a melhoria do processos a unidade organizacional pode realizar uma avaliação para estabelecer um perfil dos processos que poderá ser usado para a elaboração de um plano de melhorias. A análise dos resultados identifica os pontos fortes,os pontos fracos e s riscos inerentes aos processos.

A ISO 15504 atualmente é uma norma que representa um padrão internacional que estabelece um framework para construção de processos de avaliação e melhoria do processo de *software*. A ISO 15504 não é um método isolado para avaliação e sua característica genérica permite que possa ser utilizada em conjunto com uma variedade de métodos, técnicas e de ferramentas [38].

1.3.3 Modelo de Melhoria do Processo de *Software* Brasileiro (MPS.BR)

O MPS.BR é um modelo que visa o aumento da qualidade dos processos de desenvolvimento de *software* em empresas brasileiras e está dividido em três componentes: Modelo de Referência (MR-MPS), Método de Avaliação (MA-MPS) e Modelo de Negócio (MN-MPS) [39].

O Modelo de Referência, é onde estão localizados os conceito de maturidade e capacidade dos processos para avaliação e melhoria da qualidade dos processos e produtos/serviços de *software*. O Guia Geral fornece informações gerais e o detalhamento dos componentes e definições necessários para aplicação do MPS.BR em uma empresa. O Guia de Aquisição compreende as recomendações para empresas que querem adquirir um produto/serviço de *software*. Já o Guia de Implementação descreve como implementar os sete níveis de maturidade do MR-MPS, a saber, G (Parcialmente Gerenciado), F (Gerenciado), E (Parcialmente Definido), D (Largamente Definido), C (Definido), B (Gerenciado Quantitativamente) e A (Em Otimização).

Os níveis de maturidade são compostos por processos. Cada processo é definido de forma a atender aos propósitos previamente definidos e aos atributos estabelecidos [40].

O processo é considerado satisfeito se todos os resultados esperados para este forem atendidos. Considera-se que uma empresa atinge um dado nível de maturidade Ni, se esta satisfaz todos os processos dos níveis j, j = G, F, ..., i.

Na seção a seguir descrevemos a norma ABNT NBR ISO 31000:2009 de gestão de riscos que foi utilizada para direcionar o desenvolvimento e alcance dos objetivos específicos pretendidos nesse estudo.

1.4 Gestão de Riscos

Segundo a ABNT NBR ISO 31000:2009 [41] as organizações de todos os tipos e tamanhos enfrentam influências e fatores internos e externos que tornam incerto quando elas atingirão seus objetivos. O efeito que essa incerteza tem sobre os objetivos da organização é chamado de "risco".

A ABNT NBR ISO 31000:2009 [41] apresenta 11 princípios da gestão de riscos e ressalta que para uma gestão eficiente, convém que os princípios sejam atendidos em todos as áreas, níveis e funções da instituição. Os princípios enfatiza que a gestão de riscos:

- cria e protege valor;
- é parte integrante de todos os processos organizacionais;
- é parte da tomada de decisões;
- aborda explicitamente a incerteza;
- é sistemática, estruturada e oportuna;
- baseia-se nas melhores informações disponíveis;
- é feita sob medida;
- considera fatores humanos e culturais;
- é transparente e inclusiva;
- é dinâmica, iterativa e capaz de reagir a mudanças; e
- facilita a melhoria contínua da organização.

A norma ainda enfatiza que o processo de melhoria e maturidade da instituição em gestão de riscos deve ser efetuada juntamente com todos os demais aspectos da sua organização.

Para tornar a gestão de riscos eficaz a ABNT NBR ISO 31000:2009 [41] estabelece um conjunto de princípios e diretrizes genéricas, que podem ser aplicadas a qualquer tipo de risco, processos, operações, projetos, produtos, serviços e ativos.

Respeitadas as peculiaridades de cada organização, a gestão de riscos se aplica com maior ou menor intensidade e cria possibilidades e oportunidades de melhoria para as organizações.

1.4.1 Processo de Gestão de Riscos

Processo de gestão de riscos, segundo a norma ABNT NBR ISO 31000:2009 [41], trata-se da aplicação sistemática de políticas, procedimentos e práticas de gestão para as atividades de comunicação, consulta, estabelecimento do contexto, identificação, análise, avaliação, tratamento e monitoramento dos riscos. Na Figura 1.1 tem-se representados o processo de gestão de riscos e suas atividades.

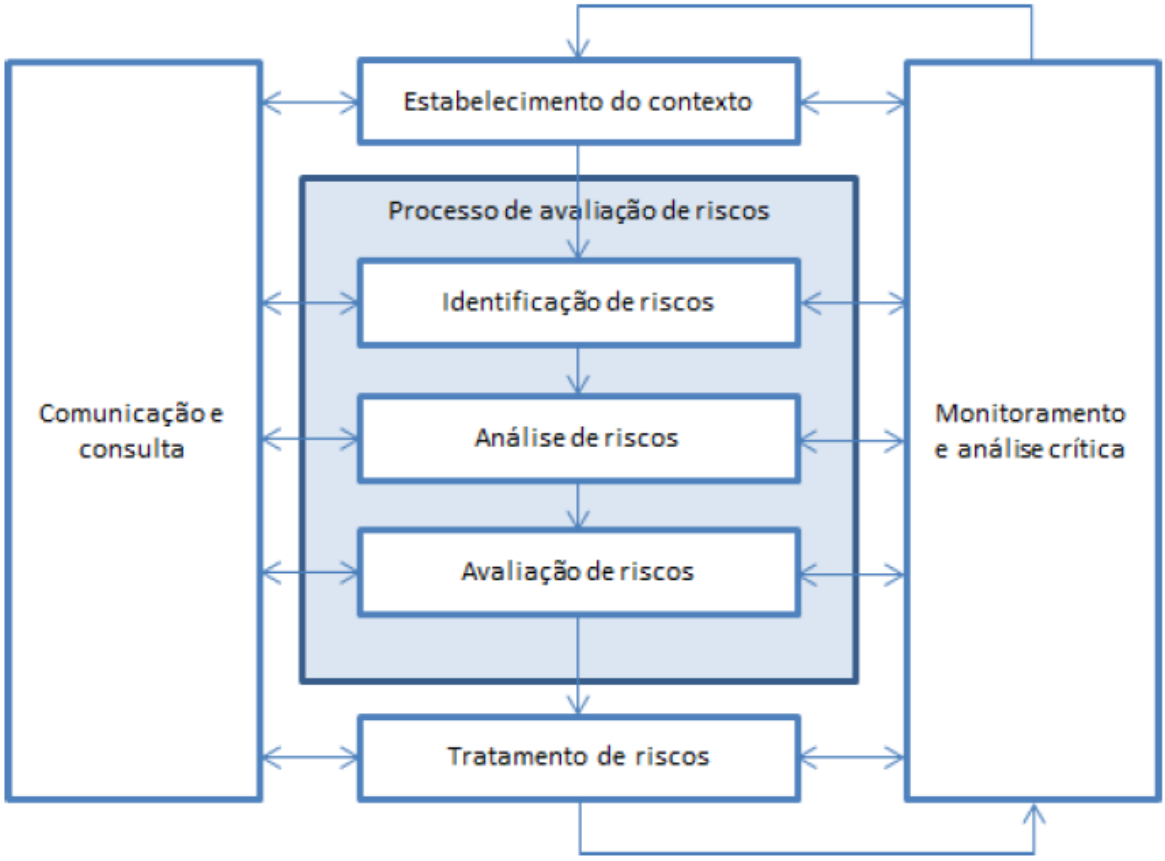

Figura 1.1: Processo de Gestão de Riscos

Fonte: ABNT NBR ISO 31000:2009 - Gestão de Riscos: Princípios e Diretrizes [41]

Para gerenciar riscos é importante que o processo de gestão de riscos esteja incorporado na cultura e nas práticas da organização, bem como adaptado aos seus processos de negócio.

Para gerenciar riscos as organizações devem desenvolver atividades para identificação e análise dos riscos. Os riscos identificados devem ser avaliados e submetidos a tratamento que os eliminem, ou os ajustem a níveis aceitáveis estabelecidos em critérios de risco.

Critérios de risco, conforme a ABNT NBR ISO GUIA 73:2005 [42], são definidos com base nos objetivos e nos contextos externo e interno da organização. Os critérios de risco podem ser derivados de normas, leis, políticas e outros requisitos de interesse da organização.

Um fator crítico para o sucesso da implantação do processo de gestão de riscos é o estabelecimento do contexto que deve contemplar, o ambiente externo e interno, seus stakeholders e os critérios de risco.

O processo de gestão de riscos, quando implantado e mantido em conformidade com a estrutura e os princípios da ABNT NBR ISO 31000:2009 possibilita que a organização alcance os seguintes benefícios:

- aumentar a probabilidade de atingir os objetivos;
- encorajar o desenvolvimento de gestão pró-ativa ao invés da reativa;
- reforçar a necessidade de identificar e tratar riscos em toda a organização;
- aprimorar a identificação de oportunidade e ameaças;
- ficar em conformidade com requerimentos legais e regulatórios e normas internacionais;
- aperfeiçoar os relatórios financeiros e a governança corporativa;
- melhorar a confiança dos stakeholders;
- estabelecer base confiável para o planejamento e a tomada de decisão;
- aprimorar os controles e reduzir perdas;
- tratar os riscos de maneira eficaz;
- melhorar a efetividade e eficiência operacional;
- aprimorar a gestão de incidentes e prevenção; e
- tornar a organização mais resiliente.

Para a realização deste estudo e alcance dos objetivos pretendidos para o processo de implementação de *software*, foram utilizados as atividades do processo de gestão de riscos

ABNT NBR ISO 31000:2009, bem como a norma ABNT NBR ISO 31010:2009 - Gestão de Riscos: Técnicas para o processo de Avaliação de Riscos [43] como direcionadores da estrutura do trabalho.

A atividade de comunicação e consulta, ela deve ocorrer durante todas as fases do processo de gestão de riscos para que as percepções das partes interessadas (stakeholders) sejam identificadas, registradas e levadas em consideração para apoio ao processo de tomada de decisão. Convém que os planos de comunicação abordem questões sobre as causas, as consequências, e as medidas que estão sendo tomadas para tratá-las. A atividade de identificação dos riscos , tem por finalidade gerar uma lista abrangente de riscos baseada em situações que possam criar, aumentar, evitar, reduzir, acelerar ou atrasar a realização dos objetivos. Convém que pessoas com um conhecimento adequado sejam envolvidas na identificação dos riscos.

A análise de riscos pode ser entendida como a atividade que busca a compreensão dos riscos. Ela envolve a apreciação das causas e das fontes de riscos, suas consequências positivas e negativas e a probabilidade de que essas consequências possam ocorrer. A análise pode ser qualitativa, semiquantitativa, quantitativa ou uma combinação destas. As consequências dos riscos podem ser expressas em termos de impactos tangíveis e intangíveis.

A avaliação de riscos, com base nos resultados da análise de riscos, auxilia a tomada de decisões sobre as ameaças e incertezas que necessitam de tratamento e a prioridade para a implantação das medidas mitigadoras. O tratamento dos riscos consiste em desenvolver opções pertinentes que visam modificar a probabilidade de ocorrência do evento, o efeito dos riscos ou ambos. As opções para o tratamento podem incluir ações para: a) evitar o risco; b) remover a fonte de risco; c) alterar a probabilidade de ocorrer o evento; d) alterar as consequências; e) compartilhar o risco; ou f) reter o risco.

O monitoramento e análise crítica envolvem a checagem e vigilância regulares dos processos de gestão de gestão para verificarem se continuam eficientes. Segundo a ABNT NBR ISO GUIA 73:2005 [42], trata-se de um processo de verificação, supervisão e observação crítica, executadas de forma contínua, a fim de identificar mudanças no nível de desempenho requerido ou esperado de algum evento da organização.

1.4.2 Técnicas de Avaliação de Riscos

A ABNT NBR ISO 31010:2009 - Gestão de Riscos: Técnicas para o Processo de Avaliação de Riscos é genérica e destina-se a apoiar a adoção da ABNT NBR ISO 31000:2009. Ela fornece orientações sobre a aplicabilidade, seleção e utilização de técnicas para a avaliação de riscos [43]. Para cada técnica é apresentada a visão geral, a utilização da técnica, as entradas, o processo, as saídas e os pontos fortes e limitações.

O processo de gestão de riscos auxilia a tomada de decisão ao considerar as incertezas e a possibilidade de circunstâncias futuras. A gestão de riscos inclui a aplicação de métodos sistemáticos para:

- a comunicação ao longo do processo;
- o estabelecimento do contexto para identificar, analisar, avaliar e tratar o risco associado a qualquer atividade, processo, função ou produto;
- o monitoramento e análise crítica de riscos[1];
- o reporte e registro dos resultados de forma apropriada.

Para auxiliar no estabelecimento do diagnóstico do processo de implementação de *software* do sistema MMO, bem como na identificação, análise e avaliação dos riscos do processo de implementação, foram utilizadas algumas técnicas e ferramentas, como: método de apoio a tomada de decisão, (aplicação do AHP), Matriz de Probabilidade e Impacto e Índice de Riscos. Logo a seguir será apresentada com mais detalhes as características e uso de cada método.

1.4.2.1 Método de análise a partir de múltiplos critérios (AHP)

Um dos principais desafios das organizações está na sua capacidade de fazer escolhas certas e consistentes, de modo alinhado com seu direcionamento estratégico. Um dos maiores desafios da ciência e tecnologia está em como tomar as decisões certas, dada uma situação [44].

As abordagens tradicionais de decisão surgiram com o desenvolvimento da pesquisa operacional, após a Segunda Guerra Mundial [45]. A modelagem matemática da pesquisa operacional que trabalha com um critério ou com múltiplos critérios, devem representar perfeitamente as preferências do decisor [45]. O uso dos múltiplos critérios não é uma simples generalização das abordagens tradicionais, ele estabelece um novo conceito na análise de contextos decisórios e na tomada de decisão.

A análise de multicritério consiste em um conjunto de métodos e técnicas para auxiliar ou apoiar pessoas e organizações a tomarem decisões, sob a influência de uma multiplicidade de critérios. A aplicação de qualquer método de análise multicritério pressupõe a necessidade de especificação anterior sobre qual objetivo o decisor pretende alcançar, quando se propõe comparar entre si, várias alternativas de decisão, recorrendo ao uso de múltiplos critérios [46].

[1] Análise crítica é a atividade realizada para determinar a adequação, suficiência e eficácia do assunto em questão para atingir os objetivos estabelecidos [41].

O AHP foi desenvolvido na década de 70 pelo professor Thomas L. Saaty sendo um dos mais usados no mundo, e que tem por definição buscar agregar valores a cada alternativa sujeita em cada critério. O método foi um dos primeiros a serem desenvolvidos no ambiente das decisões multicritério discretas [47].

O processo básico de aplicação do AHP consiste em priorizar a importância relativa de n elementos de tomada de decisão em relação a um objetivo, através de avaliações parciais destes elementos, dois a dois, facilitando a análise pelos avaliadores.

Ainda dando sequência as ferramentas que auxiliam nO processo de gestão, para a classificação e priorização dos riscos foi utilizado a Matriz de Probabilidade e Impacto que também é uma das ferramentas recomendadas pela norma ABNT NBR ISO 31010:2009, para combinar classificações qualitativas ou semi quantitativas de consequências e probabilidades a fim de produzir um nível ou classificação de risco.

1.4.2.2 Matriz de Probabilidade e Impacto

Uma matriz de probabilidade e consequência é utilizada para classificar os riscos com base no nível dos riscos. Ela funciona como uma ferramenta de seleção quando muitos riscos são identificados, tanto pra priorizar os que precisam de tratamento mais urgentes, bem como os que precisam ser considerados naquele momento [43].

A mensuração qualitativa de riscos pode ser gerada através de uma matriz, em que o nível de risco é definido pela composição da variáveis probabilidade e impacto, associadas aos eventos de perda (risco) inerentes ao processo avaliado .

A tabulação dos riscos em uma matriz permite a clara e ordenada identificação dos riscos que podem afetar a empresa tanto em termos de frequência quanto de severidade [48].

A matriz de risco pode ser construída pela composição de pesos atribuídos as variáveis de probabilidade e impacto, podendo ser particionada em regiões que caracterizam os níveis de risco associado. A definição dos níveis dos riscos podem variar em função do perfil do gestor, dos processos avaliados e dos produtos e serviços operacionalizados.

As escalas podem ter qualquer número de pontos. As escalas de 3, 4 ou 5 pontos são as mais comuns. As escalas devem ser selecionadas para serem o menos ambíguas possíveis. Se guias numéricos forem utilizados, então as unidades devem ser informadas [43].

Para classificar diferentes riscos associados a uma atividade são recomendados a utilização dos índices de riscos, que de acordo com a norma ABNT NBR ISO 31010:2009 [43], permitem a integração de uma faixa de fatores com impacto sobre o nível de risco, em uma única pontuação numérica para um dado nível de risco. A seguir são apresentadas as principais características e orientações sobre o uso desse método.

1.4.2.3 Índice de Risco

Os índices são utilizados por diferentes tipos de risco, geralmente como um dispositivo de pontuação, para classificar os riscos de acordo com o nível do risco. Isso pode ser utilizado para determinar quais riscos necessitam de avaliação detalhada e profunda [43].

As entradas se dão em razão de uma ampla análise dos sistemaS ou de uma ampla descrição do contexto. Isto requer um entendimento bastante detalhado de todos os caminhos possíveis e o que pode ser afetado.

Uma vez que o sistema tenha sido definido, pontuações são definidas para cada componente, de tal forma que possam ser combinadas para fornecer um índice composto.

As pontuações podem ser adicionadas, subtraídas, multiplicadas e ou divididas. Portanto, como resultado obtém-se uma série de números que se relacionam com um fonte específica e que podem ser comparados com índices desenvolvidos em outras fontes dentro do mesmo sistema.

Na próxima seção, afim de descobrir conhecimento a partir de informações não óbvias de códigos-fonte *Cobol*, será apresentado os principais conceitos e características do processo de Mineração de Texto, que foi utilizado para descobrir padrões e semelhanças em programas e sub-rotinas que pudessem ser monitoradas e controladas a fim de propor melhorias no processos de implementação de *software*.

1.5 Mineração de Textos

Existem várias definições para mineração de textos. O termo se refere ao processo de extração de padrões interessantes e não triviais, ou conhecimento a partir de documentos em textos não-estruturados [49]. Também há autores que descreve a mineração de textos, como sendo uma área de pesquisa tecnológica cujo objetivo é a busca por padrões, tendências e regularidades em textos escritos em linguagem natural [50].

Pode-se afirmar que a Mineração de Textos é uma especialização do processo de mineração de dados [51]. A principal diferença entre os dois processos é que, enquanto a mineração de dados convencional trabalha exclusivamente com dados estruturados, a Mineração de Textos lida com dados inerentemente não-estruturados [52]. Logo, na Mineração de Textos o primeiro desafio é obter alguma estrutura que represente os textos e então, a partir dessa, extrair conhecimento.

Para a extração e organização não supervisionada de conhecimento a partir de dados textuais, o diferencial está na etapa de extração de padrões, na qual são utilizados métodos de agrupamento de textos para organizar coleções de documentos em grupos. Em seguida, são aplicadas algumas técnicas de seleção de descritores para os agrupamentos formados, ou seja, palavras e expressões que auxiliam a interpretação dos grupos. Na etapa de pré-

processamento se encontra a principal diferença entre os processos de Mineração de Textos e processos de mineração de dados: a estruturação dos textos em um formato adequado para a extração de conhecimento. O objetivo do pré-processamento é extrair de textos uma representação estruturada, concisa e manipulável por algoritmos de agrupamento de textos. Para tal, são executadas atividades de tratamento e padronização na coleção de textos, seleção dos termos (palavras) mais significativos e, por fim, representação da coleção textual em uma formato estruturado que preserve as principais características dos dados [53]. Uma forma de realizar a seleção de termos é avaliá-los por medidas estatísticas simples, como a frequência de termo, conhecida como TF (do inglês *term frequency*). A frequência de documentos, por sua vez, contabiliza o número de documentos em que um determinado termo aparece [51]. O modelo mais utilizado para representação de dados textuais é o modelo espaço-vetorial, no qual cada documento é um vetor em um espaço multidimensional, e cada dimensão é um termo da coleção [53]. Para tal, pode-se estruturar os textos em uma bag-of-words (tabela documento-termo), na qual os termos são considerados independentes, formando um conjunto desordenado em que a ordem de ocorrência das palavras não importa.

Por meio da tabela documento-termo, cada documento pode ser representado com um vetor, que permite o emprego de um grande leque de algoritmos de extração de conhecimento.

Com o objetivo de realizar a extração de padrões, após a representação dos textos em um formato estruturado, utiliza-se nesse estudo, métodos supervisionados, como algoritmos de classificação, em que há classes ou rótulos predefinidos para treinamento de um modelo, ou seja, o aprendizado é realizado de forma supervisionada.

A Avaliação do Conhecimento pode ser realizada de forma subjetiva, utilizando um conhecimento de um especialista de domínio, ou de forma objetiva por meio de métricas e índices estatísticos que indicam a qualidade dos resultados, tais como: Acurácia, F-Measure, Kappa e outros.

A Tabela 1.2 mostra uma tabela de contingência que relaciona a classe real de um documento (se é positivo ou negativo) com o julgamento que o classificador realizou (se julgou ser positivo ou se julgou ser negativo).

Tabela 1.2: Matriz de confusão

Classificação real	Positivo	Negativo
Classificado como Positivo	Verdadeiro-positivo	Falso-positivo
Classificado como Negativo	Falso-Negativo	Verdadeiro-negativo

Fonte: SANTOS [54]

A seguir são apresentadas as definições de cada uma das quatro situações possíveis:

- Verdadeiro-positivo (VP): Número de documentos que foram classificados como positivos e realmente são da classe de positivos.
- Falso-positivo (FP): Número de documentos que foram classificados como positivos e são da classe de negativos.
- Falso-negativo (FN): Número de documentos que foram classificados como negativos e são da classe dos positivos.
- Verdadeiro-negativo (VN): Número de documentos que foram classificados como negativos e realmente são da classe de negativos.

Com essas definições pode-se perceber que os Verdadeiro-positivos e os Verdadeiro-negativos são as classificações feitas corretamente, enquanto que os Falso-negativos e os Falso-positivos são as classificações feitas erroneamente. Feitas essas definições, serão explicadas a seguir algumas medidas de desempenho [55]:

1.5.1 Acurácia

Acurácia é uma medida de desempenho dos acertos realizados pelo classificador. Essa não é totalmente confiável na avaliação de desempenho, pois ela falha em alguns casos. Suponha uma base com 1000 comentários. Destes 1000 comentários 995 são negativos e 5 são positivos. Suponha que a classe de interesse é a positiva e que um classificador julgou os 995 negativos corretamente, porém também classificou os outros 5 como negativos.

Essa medida gera uma taxa de acurácia de 99,5% para um classificador que não classificou corretamente nenhum documento da classe positiva, a classe de interesse. Isso ocorreu pois essa medida não levou em consideração os 5 comentários que deveriam ter sido classificados como positivos. E é justamente nesse ponto que as próximas medidas a serem mostradas se diferem da acurácia.

1.5.2 Precisão

A medida de precisão mensura a quantidade de documentos que foram corretamente classificados como positivos dentre todos os documentos que foram julgados como positivos. Para isso, a medida se baseia em uma divisão do número de acertos (VP) pelo número de documentos classificados como positivos (VP + FP).

$$Precisão = \frac{VP}{VP + FP} \tag{1.1}$$

1.5.3 Recall

A medida de recall (ou sensibilidade), ao contrário da precisão, é a relação entre a quantidade de documentos que foram corretamente classificados como positivos dentre todos os documentos que são realmente da classe de positivos.

$$Recall = \frac{VP}{VP + FN} \tag{1.2}$$

A sensibilidade é a proporção de positivos que são corretamente reconhecido como tal. A especificidade é a proporção de negativos que são corretamente reconhecidos. É a porcentagem de itens selecionados que estão corretos.

Exemplificando essas as medidas de precisão e recall, no exemplo dos 1000 comentários, teríamos uma Precisão = 0% e um Recall = 0%, o que é totalmente cabível, uma vez que o classificador realmente não fez nenhuma classificação positiva correta e ainda classificou erroneamente os 5 que eram positivos.

1.5.4 F-Measure

Mesmo com as medidas de precisão e recall sendo de grande utilidade para avaliações de desempenho de classificadores, às vezes se faz necessária uma medida única, para que se possa comparar de forma direta dois ou mais classificadores. E essa é a ideia da medida F que é uma medida baseada na média harmônica dos valores de precisão e recall.

$$F_1 = \frac{2 * Precisao * Recall}{Precisao + Recall} \tag{1.3}$$

Um dos problemas com a análise somente das medidas de especificidade e sensibilidade é que um conjunto de dados com classes desbalanceadas, onde uma classe tem um maior número de elementos do que outros, a especificidade pode ser realmente elevado, e a sensibilidade pode ser realmente baixo. *F-Measure* é a combinação dessas medidas para se evitar este problema [56].

1.5.5 Kappa

Kappa é um coeficiente de estatística que mede a porcentagem de para o que é esperado do acaso [57]. Ele considera em seu cálculo os resultados apresentados pela matriz de confusão, incluindo no cálculo os elementos fora da diagonal principal, que representam as discrepâncias na classificação.

Após as explanações referentes a engenharia de *software*, gerenciamento de processos, gestão de riscos e mineração de textos apresentadas neste capítulo, dar-se-á continuidade com a metodologia da pesquisa no capítulo 2.

Capítulo 2

Metodologia

O estudo proposto encontra-se inserido no campo de pesquisa organizacional, restrito a área de tecnologia da informação, e que encontra-se detalhados nos próximos itens:

2.1 Método de Pesquisa

Quanto ao objetivo da pesquisa o estudo realizado é de natureza exploratória e descritiva. Exploratória porque se buscou identificar, por intermédio de estudo de caso em um sistema de dados de operações do mercado monetário, os riscos relacionados ao processo de implementação e entrega de um *software* de qualidade para o cliente. Apesar da pesquisa exploratória ser bastante flexível, na maioria das vezes, ela assume a forma de pesquisa bibliográfica, entrevistas ou estudo de caso [58].

Por ser descritiva, ela expõe, com a utilização de questionários, qual é a percepção dos analistas de TI sobre as atividades do processo de implementação, mediante os critérios de importância, dificuldade e risco. Pesquisas descritivas [58], são aquelas que têm por objetivo coletar opiniões, crenças e atitudes de uma população, utilizando técnicas padronizadas para a coletas de dados, tais como o questionário e a observação sistemática.

Em relação ao método adotado, este se divide em estratégia e abordagem de pesquisa. Para a estratégia de pesquisa, foi utilizado o estudo de caso, que, é um método de pesquisa que geralmente utiliza dados qualitativos, coletados a partir de eventos reais, que tem por objetivo explicar, explorar ou descrever fenômenos atuais inseridos em seu próprio contexto [59]. Caracteriza-se por ser um estudo detalhado e exaustivo de poucos, ou mesmo de um único objeto, fornecendo conhecimentos profundos.

Para a realização do estudo de caso, foi selecionado uma aplicação da instituição financeira responsável pela contratação e condução das operações contratadas no mercado monetário, composta por um total de 12 funcionários, sendo um gerente de equipe e 11 desenvolvedores.

Quanto à abordagem de pesquisa, desenvolveu-se tanto sob o viés qualitativo quanto o quantitativo. A abordagem qualitativa, caracteriza-se por descrever a complexidade de determinado problema, sendo necessário compreender e classificar os processos dinâmicos ocorridos nos grupos, possibilitando o entendimento das mais variadas particularidades dos indivíduos [60]. Neste estudo, parte do conhecimento e das informações sobre o processo de implementação foram extraídos por observação *in loco* dos documentos utilizados e das atividades realizadas diariamente pelos analistas do sistema MMO, bem como a realização de entrevistas semiestruturadas, com um determinado perfil de desenvolvedores.

A abordagem quantitativa caracteriza-se pelo uso da quantificação que, tanto na coleta quanto no tratamento das informações, utiliza-se técnicas estatísticas, com o objetivo de obter resultados que evitem distorções e possibilitem uma maior margem de segurança [60]. A abordagem quantitativa, nesta pesquisa, foi aplicada na avaliação dos questionários sobre as atividades do processo de implementação, que, por intermédio da avaliação de critérios subjetivos, foi possível atribuir pesos às alternativas mediante análise de um dos critérios, com a aplicação de cálculos estatísticos.

2.1.1 Coleta dos dados

Na etapa de pesquisa documental, as informações foram coletadas por meio de relatórios, demonstrativos, apresentações, normas, padrões, melhores práticas, documentos internos, e comunicados em rede corporativa (intranet). A aplicação dos questionários e a realização das entrevistas com membros da equipe do sistema MMO aconteceram nos meses de janeiro e abril de 2015, respectivamente, a fim de obter as informações fundamentais para o desenvolvimento da pesquisa.

Os questionários e entrevistas foram realizados com perfis diferenciados de um total de doze desenvolvedores do sistema MMO: cinco participaram dos questionários sobre as atividades e oito foram entrevistados sobre os artefatos do processo de implementação.

As entrevistas, questionários e formulários são técnicas bastante úteis para obtenção de informações acerca do que a pessoa "sabe, crê ou espera, sente ou deseja, pretende fazer, faz ou fez, bem como a respeito de suas explicações ou razões para quaisquer das coisas precedentes"[61]. Também foram utilizadas como fonte de dados os códigos-fonte *Cobol* do sistema MMO, localizados no ambiente de produção, para a extração de informações que pudessem auxiliar na melhoria da qualidade dos programas e sub-rotinas do sistema, a fim de evitar falhas e indisponibilidade nas aplicações.

2.1.2 Tratamentos dos dados coletados

Os dados extraídos de documentos, dissertações, revistas, jornais e artigos serviram como fonte de informação sobre as possíveis ferramentas e métodos que poderiam ser utilizados para análise e avaliação das informações levantadas por meio de questionários e entrevistas aos desenvolvedores sobre o processo de implementação de *software*.

Para análise e avaliação dos resultados sobre as atividades do processo de implementação, foi utilizado o método AHP, com a comparação par a par das alternativas e atribuição de pesos diferenciados mediante um critério.

A análise e avaliação das entrevistas sobre os artefatos do processo de implementação foram realizadas por intermédio do manuseio e interpretação dos dados qualitativos, com análise das respostas e agrupamento, padronização das informações em categorias.

2.2 Estrutura de desenvolvimento da pesquisa

Para o alcance do objetivo geral e de cada um dos objetivos específicos pretendidos, foi utilizada como referência para estruturação desta pesquisa a norma ABNT NBR ISO 31000:2009 – Gestão de Riscos: Princípios e Diretrizes, representada na Figura 2.1.

A norma internacional NBR ISO 31000:2009 trata sobre a estruturação e gestão de riscos, com o objetivo de fornecer os fundamentos para a concepção, implementação e monitoramento, análise crítica e melhoria da gestão de riscos através de toda organização. Ela foi escolhida como base para estabelecer os arranjos organizacionais desse estudo, em razão de ser uma norma genérica que pode ser adaptada às características e necessidades de qualquer organização [41], bem como aos diferentes tipos de riscos.

A estrutura de gestão de riscos assegura que a informação de riscos proveniente desse processo seja adequadamente reportada e utilizada como base para tomada de decisão e melhorias dos processos.

De acordo com a norma ABNT NBR ISO 31000:2009, o plano de gestão de riscos pode ser aplicado a um determinado produto, processo e projeto, em parte ou em toda a organização. Portanto, em busca de um plano de gestão organizado e estruturado para promover melhorias no processo de implementação de *software* utilizado por um sistema de dados de operações do mercado monetário, foi estabelecido correlações entre as atividades do processo de gestão de riscos e os objetivos específicos definidos neste trabalho.

Na Figura 2.1, temos a representação da relação entre cada um dos objetivos específicos pretendidos nesse trabalho, com as atividades previstas na norma ABNT NBR ISO 31000:2009 de gestão de riscos. Nele é possível identificar quais atividades foram aplicadas para o alcance no alcance do objetivo geral e específicos desse estudo:

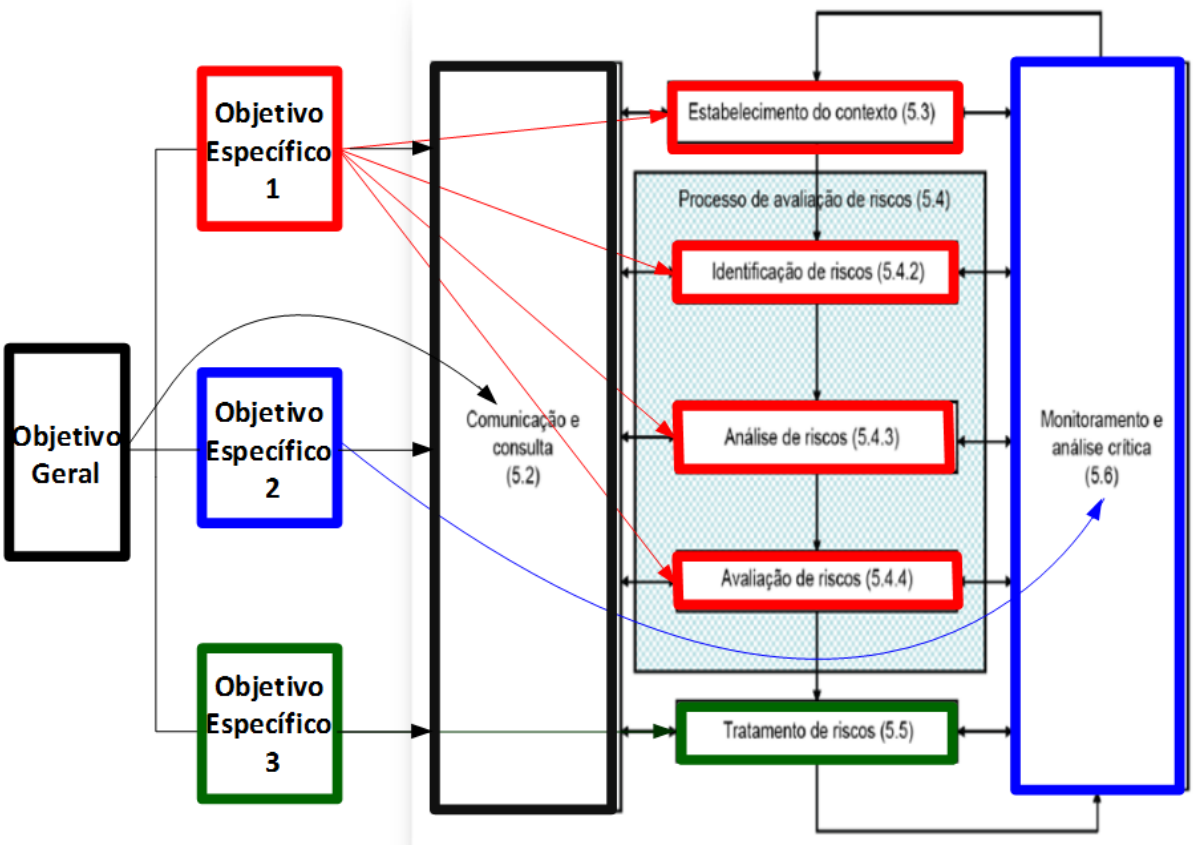

Figura 2.1: Relação entre os objetivos da pesquisa e as atividades do processo de gestão de riscos

Fonte: Elaboração própria

- **Objetivo Específico 1:** aplicação das atividades do processo de gestão de riscos destacadas em vermelho: estabelecimento do contexto, identificação de riscos, análise de riscos e avaliação de riscos.
- **Objetivo Específico 2:** aplicação da atividade do processo de gestão de riscos destacado em azul: monitoramento e análise crítica.
- **Objetivo Específico 3:** aplicação da atividade do processo de gestão de riscos destacado em verde: tratamento de riscos.
- **Objetivo Geral:** trata-se de uma consolidação de todos os resultados obtidos a partir da realização dos Objetivos Específicos, em que após a realização de todas as sete atividades do processo de gestão de riscos, torna-se possível propor melhorias no processo de implementação.

Ainda na Figura 2.1, note que a atividade de comunicação (destacada em preto), por tratar-se de um processo contínuo e iterativo utilizado pela organização para fornecer, compartilhar ou obter informações e estabelecer diálogo com as partes interessadas, a sua

aplicação não se restringiu apenas ao alcance dos Objetivos Específicos, mas também do Objetivo Geral. Neste estudo, a atividade de Comunicação e Consulta ocorreu durante todo o estudo e teve grande relevância na contextualização, na motivação e convencimento das partes envolvidas (gerentes de equipe e desenvolvedores) sobre os benefícios e melhorias a serem alcançados com o processo de gestão de riscos para o processo de implementação de *software*.

Nesta pesquisa, um aspecto muito importante da atividade de comunicação foi a garantia de confidencialidade dos dados apresentados pelas pessoas envolvidas ao longo deste estudo de caso, omitindo seus nomes e utilizando pseudônimos, o que contribuiu para que os desenvolvedores ficassem à vontade para manifestar suas opiniões, sem receio de situações negativas como retaliações ou represálias por parte dos administradores e gerentes da instituição.

O estabelecimento do contexto, conforme apresentado na norma ABNT NBR ISO 31000:2009 [41] , tem por objetivo avaliar e compreender os ambientes externo e interno da organização, a fim estabelecer a gestão de riscos. Dada a importância em definir extensão, abrangência e profundidade das verificações a serem conduzidas nas demais atividades do processo, ela ficou definida como o primeiro passo a ser finalizado na realização do Objetivo Específico 1, referente ao diagnóstico do processo de implementação de *software*.

Apesar de não haver na norma ABNT NBR ISO 31010:2009 – Gestão de Riscos: Técnicas para o processo de avaliação de riscos – nenhuma recomendação de método de apoio para realização da atividade de estabelecimento do contexto, em razão de resultados obtidos em trabalhos correlatos anteriores [62], [63] e [64], utilizou-se, neste estudo, a ferramenta SIPOC e o mapeamento de processos *AS IS* para definição do contexto interno e, de gestão de riscos do processo de implementação de *software*.

Dando sequência às atividades realizadas no Objetivo Específico 1, passamos para a atividade de identificação dos riscos, que, segundo definição descrita na norma ABNT NBR ISO 31010:2009 [43], consiste no processo de encontrar, reconhecer e registrar os riscos. Essa atividade inclui a identificação das causas e fontes do risco, eventos, situações ou circunstâncias que podem ter um impacto material sobre o objetivo desejado.

Várias técnicas de apoio podem ser utilizadas para tornar mais exata e completa a identificação de riscos [43]. Neste trabalho, para esta finalidade, foram utilizadas a entrevista semiestruturada e o Método Multicritério de Apoio a Decisão (MDCA), que segundo a norma ABNT NBR ISO 31010:2009, são consideradas respectivamente como fortemente aplicável e aplicável para a atividade de identificação de riscos.

Com o objetivo de compreender a natureza e a magnitude dos riscos identificados ao longo do processo de implementação, foi realizada, com o uso da ferramenta de Matriz de Probabilidade e Impacto, a análise de riscos. Esta técnica, segundo a norma ABNT NBR

ISO 31010:2009 [43], é considerada um método semiquantitativo fortemente aplicável na determinação e combinação das consequências e probabilidades para a indicação de um nível de risco.

Após concluída a análise de riscos, foi realizada a atividade de avaliação, que consistiu na comparação dos níveis estimados de risco com os critérios de risco, a fim de determinar a significância do nível e do tipo de risco. Para a realização dessa etapa, foi utilizada a técnica de índice de risco, que é considerada pela norma ABNT NBR ISO 31010:2009 [43] fortemente aplicável para a atividade de avaliação de riscos.

O Objetivo Específico 2 fez referência à atividade de monitoramento e análise crítica de riscos, que envolve verificação, supervisão, observação crítica ou identificação da situação, executadas de forma contínua, a fim de identificar mudanças no nível de desempenho requerido ou esperado [43]. Desse modo, a fim de monitorar a qualidade dos artefatos produzidos durante o processo de implementação de *software*, foi utilizado o processo de mineração de textos para a extração de conhecimento em códigos-fonte *Cobol*, com o uso de algoritmos de classificação supervisionado: *Naive Bayes*, *SVM* e *Random Forest*.

Por fim, no Objetivo Específico 3, abordou-se o tratamento de riscos, que envolve diferentes processos para se modificar a ocorrência das vulnerabilidades, seja pela decisão de não iniciar ou descontinuar a atividade que deu origem ao risco, seja pela alteração da probabilidade e ou efeito dos riscos identificados [43].

No objetivo específico 3, foram utilizadas as notações BPMN e os conceitos de gerenciamento de processos de negócio BPM, para a elaboração de documentos de especificação de unidades, para codificação e para a realização do mapeamento de processos *TO BE*, com o redesenho e inclusão de novas atividades para melhoria do processo de implementação.

A seguir é apresentado um resumo dos principais passos realizados em cada um dos objetivos específicos, para o alcance do objetivo geral pretendido neste estudo:

- **Objetivo específico 1)** Fazer o diagnóstico do processo de implementação de *software* de um sistema de dados de operações do mercado monetário de uma instituição financeira.

  Passos: foram realizados pesquisa documental nas normas e padrões de desenvolvimento de *software* adotados pela instituição financeira e questionários para identificar, na visão dos analistas, quais artefatos e atividades do processo de implementação contribuem na entrega de um *software* de qualidade para o cliente. Utilização do SIPOC e do mapeamento de processos *AS IS* do modelo de gestão de processos de negócio BPM no estabelecimento do contexto (limites, escopo) do processo de implementação. Para a análise e avaliação dos questionários sobre as atividades do processo de implementação, foi utilizado o método de apoio à tomada

de decisão multicritério AHP. Para avaliação dos artefatos do processo de implementação foi utilizado a técnica de coleta de dados entrevista, em que os resultados foram padronizados e agrupados em categorias. Para realização da atividade de análise dos riscos foi utilizado a ferramenta de Matriz de Probabilidade e Impacto, e para avaliação das atividades foi utilizado o Índice de Riscos, ambos técnicas recomendadas pela norma ABNT NBR ISO 31010:2009 [43]. Os resultados obtidos com a aplicação da ferramenta SIPOC e multicritério AHP foram publicados respectivamente no Workshop de Pós-Graduação 2014 (Wpos-2014) [65] e no XXI Simpósio de Engenharia de Produção (SIMPEP) [66].

- **Objetivo específico 2)** Propor um modelo de classificação automática da complexidade de códigos-fonte *Cobol*, a fim controlar e monitorar a qualidade e os riscos dos produtos produzidos pelo processo de implementação de *software* do sistema de dados analisado.

  Passos: foram realizadas pesquisa documental nos códigos-fonte *Cobol* do sistema MMO e entrevista com os desenvolvedores para identificar os principais termos da linguagem de programação *Cobol* capazes de categorizar os programas em níveis de complexidade para manutenção. Para atingir este objetivo, foram aplicados a metodologia CRISP–DM na utilização da técnica de mineração de textos com a aplicação dos algoritmos de classificação supervisionados (*SVM*, *Random Forest* e *Naive Bayes*) e o pacote (TM) do *software* RStudio para a limpeza e preparação dos dados.

- **Objetivo específico 3)** Propor melhorias na especificação de componentes e processo de desenvolvimento de *software*, a fim de aumentar a agilidade e segurança no tratamento dos riscos do processo de implementação do sistema de dados estudado.

  Passos: elaborar um processo de melhoria da especificação de componentes de *software* (programas e sub-rotinas) para a implementação, por intermédio da utilização do mapeamento de processos *TO BE* do modelo de gerenciamento de processos de negócio (BPM). Para atingir este objetivo, foi utilizado a notação BPMN para representação, em diagramas, das principais funcionalidades do sistema MMO desde uma visão macro (negócio), até uma visão detalhada (técnica) das sequências de instruções presentes nos códigos-fonte e processos do sistema MMO. Foi realizado, também, pesquisa documental e bibliográfica para utilização da abordagem *Top Down* na especificação de novas funcionalidades do sistema MMO solicitada pelo gestor de negócio à área de desenvolvimento de *software*, bem como o redesenho do processo de desenvolvimento de *software* por meio do mapeamento *TO BE* baseado

nas atividades de tratamento e monitoramento de riscos da norma ABNT NBR ISO 31000:2009 de Gestão de Riscos.

A Figura 2.2 descreve, de forma resumida, a estrutura da pesquisa, bem como os passos utilizados no alcance dos objetivos específicos.

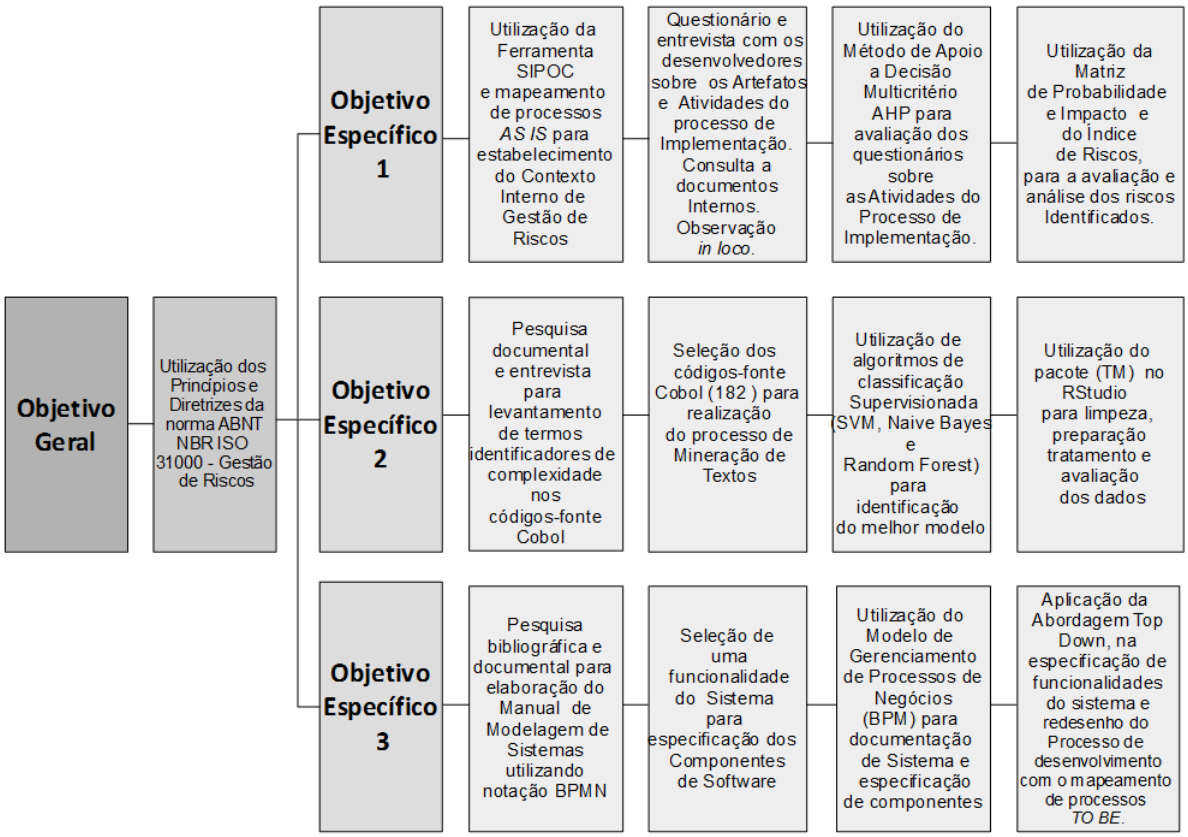

Figura 2.2: Esquema resumido da Estrutura da Pesquisa

Fonte: Elaboração própria

A partir da metodologia e da estrutura de pesquisa proposta, será realizado o estudo de caso, iniciando-se pelo estabelecimento do diagnóstico do processo de implementação de *software* que será apresentado no capítulo 3.

Capítulo 3

Análise e diagnóstico do processo de implementação de *software*

3.1 Estabelecimento do Contexto

Para a realização do diagnóstico do processo de implementação de *software*, foram realizados os seguintes passos:

1. aplicação da ferramenta SIPOC para estabelecimento do contexto do processo de implementação de *software* no Departamento de Tecnologia;
2. mapeamento *AS IS* do processo de desenvolvimento de *software*, com o objetivo de visualizar quais documentos servem de entrada para o processo de implementação de *software* e como é a sua relação com os demais processos;
3. entrevista com os desenvolvedores do sistema de dados MMO, para identificar as atividades de maior impacto na entrega de um *software* estável e de qualidade para o cliente.

O Departamento de Tecnologia da Informação é o órgão responsável por atender aos objetivos corporativos e estratégicos da instituição. O setor financeiro é, sem dúvida, o setor que mais depende da tecnologia da informação e mostra, no Brasil, de forma exemplar, o impacto da revolução tecnológica na competitividade empresarial [67]. Novos serviços surgiram em decorrência de competências essenciais, de modo que, nos últimos anos, os investimentos em Tecnologia da Informação tiveram um grande aumento, pois ela passou a ser vista não apenas como um centro de processamento de dados, mas também como um elemento estratégico que sustenta o crescimento e a participação de toda instituição no concorrido mercado financeiro.

Ao perceber essa nova configuração do setor tecnológico, novos investimentos passaram a ser feitos no Departamento de Tecnologia, pois se percebeu a necessidade de que não basta apenas investir na aquisição de hardware com alto poder de processamento. Os *softwares* e aplicações também devem evoluir e estarem aptos a atender às demandas do mercado e exigências dos órgãos reguladores tão logo solicitadas pelos gestores de negócio.

Esse reconhecimento da Tecnologia da Informação, como parte estratégica da instituição é o que permitiu ao Departamento de TI participar mais ativamente na elaboração das estratégias das áreas de negócio, cooperando com informações dos cenários e tendências nacionais e globais de sua área.

3.1.1 Contexto Interno do Departamento de Tecnologia da Informação

O estudo de caso a que se propõe este trabalho foi realizado no Departamento de TI de uma instituição do setor financeiro, que possui aproximadamente 4.400 colaboradores especializados em processos de desenvolvimento de *software*, governança de TI, suporte e monitoramento de produtos e serviços de *software*. O Departamento de TI atualmente está dividido em quatro unidades, que são: Unidade de Estruturação, Unidade de Operação, Unidade de Construção e Unidade de Governança de TI, sendo responsáveis por todo o processo de gerenciamento e controle dos produtos, serviços e processos de TI.

Esse Departamento possui aproximadamente 934 aplicações que atendem às necessidades de produtos e serviços de 27 departamentos da instituição e apoiam à tomada de decisão por meio de informações, relatórios gerenciais e indicadores de desempenho. A unidade de construção possui aproximadamente 1.300 funcionários responsáveis por manter em funcionamento mais de 740.000 programas ativos escritos em linguagem de baixa e alta plataforma, tais como: *Assembler*, C, *Cobol*, *Easytrieve*, Natural, *JCL, Visual Basic e Java*.

Para o desenvolvimento deste estudo, foi escolhida, para análise e acompanhamento o processo de implementação de uma equipe da área de construção de *software*, responsável pelos processos de desenvolvimento e manutenção de um sistema de dados de operações do mercado monetário. Neste trabalho, por questões de segurança e confidencialidade das informações, o sistema de operações do mercado monetário será simbolicamente representado pela sigla MMO.

A equipe responsável pelo desenvolvimento e manutenção do sistema MMO é composta por um total de 12 funcionários, sendo um gerente de equipe e 11 desenvolvedores.

Esse sistema responsabiliza-se pela compra e venda de operações com o mercado, tem por finalidade disponibilizar serviços e produtos que auxiliem seus usuários a acompanhar

o ciclo de vida de suas operações, bem como gerenciar e controlar a aderência dos contratos com os objetivos estratégicos da instituição, em conformidade com as diretrizes e normas estabelecidas pelos órgãos reguladores do sistema financeiro.

O sistema MMO foi criado inicialmente em 2001 e desenvolvido em plataforma baixa com o uso do programa *Visual Basic* 6.0, da *Microsoft*. Em 2011, por motivos de descontinuidade da linguagem pelo fornecedor, e devido à apresentação de problemas de compatibilidade com os sistemas *Windows* lançados posteriormente, iniciou-se um estudo para migração das suas macro-funcionalidades para a alta plataforma (mainframe) com a utilização da linguagem de programação *Cobol* para a codificação de regras de negócio e acesso à base de dados, e *Java* para o desenvolvimento das telas de interface com o usuário.

Para identificar as partes relevantes e delimitar as fronteiras do processo de implementação, foi utilizado a ferramenta SIPOC, que se comporta como uma tabela de preenchimento sistemático e permite visualizar e compreender, em uma visão ampliada, o processo em estudo, identificando-se quais resultados ele gera e para quem são direcionados [68]. Assim, foram elaborados sucessivos diagramas em que a partir do momento que um processo se encontrava racionalizado, um outro se tornava prioritário e assim sucessivamente, até se alcançar o processo de implementação de *softwares*. O primeiro entendimento de contexto realizado foi o do Departamento de Tecnologia da Informação, que é onde estão inseridos todos os principais processos responsáveis pela entrega de produtos e serviços de *software* para a Instituição. Na Figura 3.1 podemos ver representada a visão macro do Departamento de TI, que normalmente se assemelha à percepção que os clientes e usuários têm sobre o papel da TI na instituição.

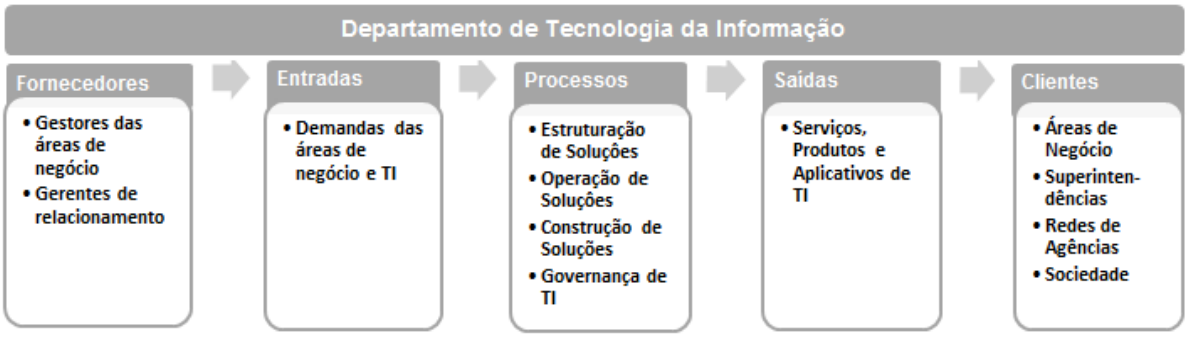

Figura 3.1: SIPOC Nível 1 - Departamento de Tecnologia da Informação

Fonte: Elaboração própria

Nessa Figura 3.1, estão demonstradas os quatro processos do Departamento de Tecnologia da Informação, que são responsáveis por atividades e tarefas essenciais ao desenvolvimento de *software*. São eles:

- **estruturação de solução** - processo responsável por registrar as necessidades do cliente e definir os requisitos de melhoria e inovação de produtos e serviços de uma aplicação;
- **operação de solução** - processo responsável pelo monitoramento e controle das aplicações que já foram entregues ao cliente e encontram-se em funcionamento na instituição;
- **governança de TI** - processo responsável por elaborar e acompanhar o planejamento estratégico e orçamentária do Departamento de Tecnologia da Informação e de suas unidades;
- **construção de soluções** - processo responsável pela análise, projeto e desenvolvimento das aplicações de acordo com os requisitos especificados pelos gestores de negócio da instituição. Aqui são tratadas tanto as novas funcionalidades, quanto a manutenção das que já existem.

O SIPOC do Departamento de TI, por tratar-se do primeiro levantamento realizado foi denominado de SIPOC nível 1. Ele representa a visão mais ampliada de todo o processo de desenvolvimento de *software*, mostrando a origem dos demais processos.

Na Figura 3.1 podemos perceber que os produtos e serviços entregues pelo Departamento de Tecnologia beneficiam a todos os clientes da instituição, sejam eles internos ou externos. A sociedade, de maneira geral, é a principal consumidora e beneficiada dos serviços bancários.

3.1.2 Contexto Interno do Processo de Construção de Soluções

O processo de Construção de Soluções é o segmento da TI responsável pela sequência de atividades que gerenciam o ciclo de vida dos *softwares*, abrangendo desde as atividades de definição de requisitos até a entrega final do *software* aos clientes.

O processo de desenvolvimento de *software* é considerado um dos principais mecanismos para se garantir sua qualidade. Há uma grande diversidade deles no mercado e, não existe um em específico que seja o ideal, segundo [69].

Os processos evoluíram para explorar a capacidade das pessoas em uma organização e as características específicas dos sistemas que estão seno desenvolvidos. No caso de alguns sistemas, como os sistemas críticos, é necessário um processo de desenvolvimento muito estruturado. Nos sistemas de negócio, com requisitos que mudam rapidamente, um processo flexível e ágil é provavelmente mais eficaz [69].

Na Figura 3.2 pode-se observar a visão ampliada do processo de Construção de Soluções, que está dividida em cinco subprocessos: requisitos de *software*, análise e projeto de *software*, implementação, testes e implantação.

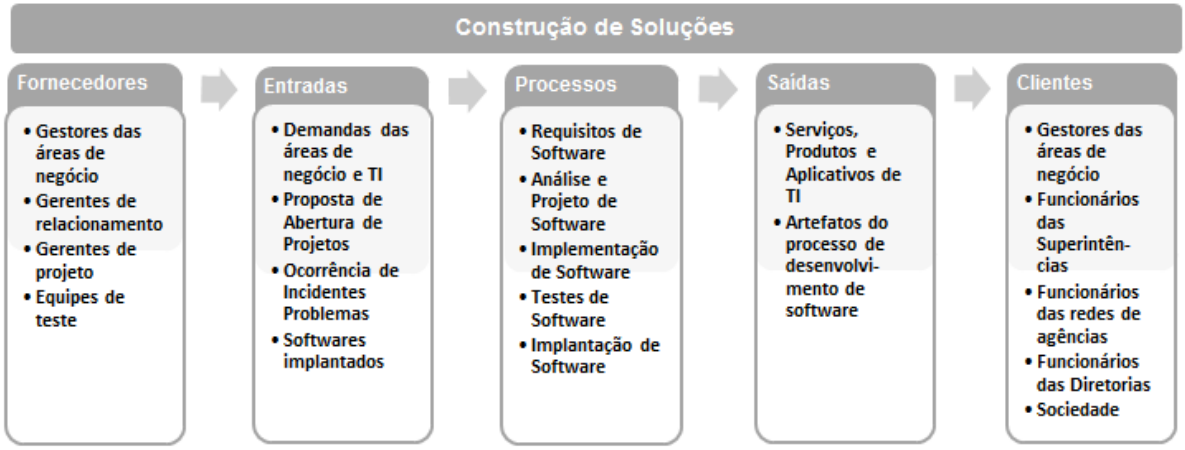

Figura 3.2: SIPOC Nível 2 - Construção de Soluções

Fonte: Elaboração própria

O processo de Construção de Soluções é a responsável por todo o processo de desenvolvimento de *software*, que tem por objetivo atender às solicitações dos clientes, seja com novos projetos ou manutenção dos que já existem.

O SIPOC do processo de Construção de Soluções é definido como SIPOC Nível 2, pois trata-se do segundo diagrama realizado até o momento, que dará subsídio ao entendimento do contexto e estabelecimento do diagnóstico do processo de implementação. Para conhecer em detalhes o seu funcionamento e o fluxo de suas informações, foi realizada a modelagem de processos de negócios (BPM) para descrever como as atividades se relacionam e quais os documentos produzidos.

3.1.3 Mapeamento *AS IS* das atividades do processo de Contrução de Soluções

Para identificar os principais problemas e fragilidades no processo de Construção de Soluções, que também é chamado no sistema MMO como processo de desenvolvimento de *software*, foi necessário levantar e documentar seu real funcionamento. Essa análise é importante para verificar a sequência das atividades, tarefas, documentos e o fluxo das informações em todo o processo, desde o momento em que é definido o escopo da funcionalidade solicitada pelo gestor de negócio até o momento da entrega final do *software*.

Para esse fim, além da consulta aos documentos internos da Instituição (manuais e normas), também foi realizada a observação *in loco* das tarefas realizadas diariamente

pelos analistas. A partir dessas informações, foi possível realizar o mapeamento *AS IS* dos processos adotados pela instituição no desenvolvimento de seus aplicativos, conforme apresentado no *Apêndice A* e obter, como resultado, um diagrama que demonstra o fluxo das informações entre os participantes e suas respectivas atividades.

Pelo mapeamento *AS IS*, consegue-se identificar um total de 23 atividades, distribuídas em cinco raias, que representam os papéis que os atores do processos assumem durante o processo de desenvolvimento de *software*.

Papel é uma definição abstrata de um conjunto de competências, características e habilidades necessárias para a execução de uma ou mais atividade [70]. Durante as observações *in loco* foi possível perceber, no sistema MMO, que o mesmo papel pode ser realizado por uma ou mais pessoas e que um mesmo indivíduo pode desempenhar mais de um papel.

Os atores do processo de desenvolvimento de *software* são:

- **analista de negócio -** é quem aplica as técnicas de requisitos, responsável por fazer entrevista com o cliente, com o objetivo de descrever e registrar a sua necessidade em documentos de requisitos;
- **projetista de *software* -** é quem define a arquitetura de *software*, com os seus padrões, modela os dados, além de projetar o banco de dados. Um arquiteto de *software* é o responsável por criar a arquitetura de um sistema a partir de seus requisitos, possuindo a tarefa de tomar decisões para organizar a estrutura e a inter-operação entre os componentes do sistema [8];
- **implementador de *software* -** é o responsável por confeccionar o *software* com base nos requisitos e também dos padrões arquiteturais e especificações de projetos definidos pelos projetista de *software*;
- **testador de *software* -** é o responsável por projetar os testes, participar da elaboração dos planos de testes e emitir um parecer em cada nível de teste realizado;
- **implantador de *software* -** é quem planeja a implantação e é o responsável por identificar as prioridades, restrições e características especiais do sistema, além de preparar o ambiente-alvo para disponibilizar os *softwares* em funcionamento.

Cada participante deve ter um conjunto de competências para a execução de uma ou mais atividades no processo de desenvolvimento de *software* da instituição. Na Tabela 3.1 temos a relação dos participantes e os respectivos subprocessos em que são responsáveis pelo resultado.

Tabela 3.1: Atores e Subprocessos do processo de desenvolvimento de *software*

Atores	**Subprocesso**
Analista de negócio	Requisitos de *software*
Projetista de *software*	Análise e projeto de *software*
Implementador de *software*	Implementação de *software*
Testador de *software*	Teste de *software*
Implantador de *software*	Implantação de *software*

Fonte: Elaboração própria

Além dos atores do processo de desenvolvimento de *software*, também consegue-se visualizar, no *Apêndice A*, as suas atividades. Para dar sequência às informações obtidas até o momento, com a técnica SIPOC, segue a descrição de todas as atividades do processo de desenvolvimento separadas de acordo com o subprocesso ao qual pertencem.

3.1.3.1 Subprocesso Requisitos de *Software*

Ele é formado por 4 atividades, conforme abaixo:

Consolidar Escopo - nesta atividade os analistas de negócio trabalham junto aos gestores de negócio para definirem os objetivos pretendidos com o *software*.

Aqui busca-se levantar os principais elementos que devem direcionar a construção do *software*, tais como: identificar os principais requisitos funcionais e não funcionais das intervenções, bem como os possíveis intervenientes da solução (clientes, parceiros, peritos de negócio) e verificar quais funcionalidades são novas e quais já existem e, quando for o caso, elaborar o esboço das telas de interface com o usuário.

Planejar as iterações - nesta atividade, avalia-se a possibilidade de dividir o escopo da intervenção em conjuntos coesos de funcionalidades, de tal forma que as iterações propiciem, à área de negócio, uma percepção de entrega de valor.

Caso o escopo seja pequeno, será apenas uma iteração, mas caso houver muitas funcionalidades, deve-se classificá-las considerando o valor agregado ao negócio, conforme critérios abaixo:

- *necessária*: obrigatório para o atendimento das necessidades e expectativas do negócio. Sua ausência inviabiliza a entrega do valor esperado, por esse motivo, não apresenta flexibilidade de negociação;

- *importante*: existe uma necessidade clara a ser atendida pelo requisito, entretanto, sua substituição temporária não inviabiliza a entrega do *software*;
- *desejável*: contribui para o atendimento das necessidades e expectativas do negócio, porém, sua ausência apresenta baixo impacto na percepção de valor. Na maioria dos casos, esse tipo de funcionalidade corresponde. Em outros, pode corresponder a algum tipo de diferencial, quando comparado a outros produtos/soluções existentes no mercado.

Especificar requisitos - é a atividade responsável pela modelagem e detalhamento dos requisitos. Nela são previstas tarefas para completar com maior riqueza de detalhes as informações e regras de negócio documentadas durante a atividade de consolidação do escopo. Nesse momento, também deve-se rever as telas de interface do usuário, para validar se as informações mapeadas são suficientes para a realização das ações pretendidas para a funcionalidade.

Validar os requisitos - atividade que visa garantir que os requisitos definidos de fato esclareçam e limitem o escopo do *software*. Aqui são solicitadas as validações formais do gestor de negócio e demais participantes envolvidos na especificação de requisitos, em relação à viabilidade técnica, adequação, clareza e completude das informações.

3.1.3.2 Subprocesso Análise e Projeto de *Software*

Este subprocesso é formado por quatro atividades, que são:

Analisar a solução - nesta atividade deve-se analisar as informações, regras de negócio, e restrições do documento de requisitos, e descrever o que precisa ser feito pelo sistema, do ponto de vista da engenharia de *software*. Nesta etapa é que deve ser feita uma avaliação conjunta entre os sistemas e canais intervenientes, sobre qual o impacto e a melhor estratégia para viabilizar o atendimento, caso haja alguma concorrência com outras alterações de maior prioridade;

Projetar o banco de dados - tem por finalidade construir ou atualizar o Modelo de Entidade de Relacionamento (MER) e o seu dicionário de dados, além de converter os termos lógicos dos elementos do modelo (Database, Tablespace, Tabelas, Atributos, PK, FK e Índices) em nomes físicos. O modelo de dados é único por sistema e deve ser versionado a cada necessidade de alteração nas entidades, atributos, relacionamentos, chaves ou índices;

Projetar interface com o usuário - esta atividade é a responsável por projetar as interfaces de usuários para atender a solução. Atualmente são utilizados dois tipos de interfaces para o usuário. A mais tradicional e que se encontra em desuso é a codificada em linguagem Natural para emuladores 3270. A segunda opção compõe-se de interfaces gráficas, codificadas em linguagem Java para a utilização via web browser;

Projetar as unidades - atividade responsável por definir a composição física das funcionalidades e a sequência lógica da execução das instruções em uma visão estruturada. Nesta fase são definidos os programas com base na topologia arquitetural definida no documento de requisitos, bem como a sua estrutura hierárquica e o fluxo de dados entre os diferentes programas da intervenção.

3.1.3.3 Subprocesso Implementação de *Software*

Este subprocesso é composto por duas atividades, conforme segue abaixo:

Codificar as unidades - visa à codificação em uma linguagem de programação definida com base em sua topologia arquitetural constante do documento de requisitos, a partir de documentos gerados durantes os subprocessos anteriores;

Integrar as unidades - visa reunir os programas implementados para teste de integração e verifica se as informações entre os diferentes programas estão apresentando os resultados esperados.

3.1.3.4 Subprocesso Teste de *Software*

Este subprocessso possui um total de 9 atividades, que encontram detalhadas logo a seguir:

Planejar os testes - nesta atividade define-se uma proposta de testes que possibilita a verificação dos requisitos funcionais e não funcionais especificados no documento de requisitos e que também demonstre a qualidade da construção na aplicação;

Especificar os testes - esta atividade tem por objetivo refinar a abordagem definida no plano de testes e definir as informações necessárias para que eles sejam executados corretamente;

Preparar os testes - é a etapa responsável por disponibilizar os dados e o ambiente necessários para execução dos testes, fase em que são identificados, dentre outros itens, *software*, hardware, acessos e dados necessários;

Executar testes de unidade - esta atividade executa os testes planejados em cada um dos programas e registra os resultados obtidos em documento que possam ser utilizados para posterior avaliação e validação dos testes unitários.

Executar testes de integração - esta atividade executa os testes planejados integrando-se todos os programas desenvolvidos, bem como os demais sistemas intervenientes. Caso sejam identificados defeitos, deve-se solicitar aos implementadores a devida correção, realizando-se novamente os testes anteriores, mediante a devida documentação do evento ocorrido;

Executar testes de sistema - nesta atividade deve-se executar os testes planejados, evidenciar resultados obtidos, registrar incidentes e solicitar suas correções aos implementadores. Caso necessário, deve-se realizar testes de regressão, para garantir que defeitos não foram introduzidos nas funcionalidades testadas anteriormente;

Avaliar os testes - esta atividade tem por finalidade analisar resultados obtidos e incidentes ocorridos durante esses testes, emitindo parecer sobre sua abrangência e qualidade;

Validar os testes - nessa atividade deve-se analisar os testes realizados, decidindo sobre sua aprovação ou rejeição. Neste momento verifica-se o grau de cobertura dos testes;

Executar os testes de aceitação - esta é a atividade realizada pelo gestor de negócio, na qual ele executa o *software* e registra os resultados obtidos.

3.1.3.5 Subprocesso Implantação do *Software*

Este subprocesso é composto por três atividades, que são:

Planejar a implantação - nessa atividade define-se como os *softwares* devem ser instalados, identificando suas prioridades, condições especiais e restrições. É importante verificar o impacto das transferências em relação a programas já existentes, bem como identificar a necessidade de escalonamento dos processos, a passagem e o recebimento de condições dos programas e sistemas intervenientes;

Disponibilizar o *software* - preparar o ambiente-alvo e a unidade de implantação, que pode ser um conjunto de programas e processos pertencentes a uma mesma iteração;

Implantar o *software* - é feita a instalação do *software* para os usuários no ambiente-alvo. Nessa atividade o implantador deve acompanhar as baixas dos programas, conferindo se todos foram agendados para o ambiente de produção.

No mapeamento *AS IS* é possível observar-se também que quase todas as atividades produzem artefatos, que nada mais são do que os documentos produzidos e consumidos por atividades ao longo do processo e que se encontram presentes em todo o ciclo de vida de desenvolvimento [71]. No total são produzidos 21 tipos de artefatos para plataforma alta, que se dividem em artefatos de implementação ou de documentação. Os artefatos de implementação são os códigos-fonte gerados no processo de implementação. Já os artefatos de documentação dividem-se em:

- **documentos da intervenção**: registram as situações tratadas em cada uma das intervenções a que o sistemas foram submetidos e que portanto não guardam relação com artefatos anteriores. Intervenção é qualquer iniciativa que tenha por finalidade produzir uma aplicação ou realizar modificações em uma aplicação existente. No processo de desenvolvimento de *software*, as intervenções são classificadas como construção de aplicações novas ou manutenção de aplicações já existentes.
- **documentos de sistema**: são únicos para cada sistema e devem ser atualizados a cada intervenção que se faça.

Na Tabela 3.2 temos a relação dos 21 artefatos de desenvolvimento de *software* para plataforma alta (mainframe) listados de acordo com os subprocessos a que pertencem: quatro em Requisitos, onze em Análise e Projeto de *Software*, um em Implementação, quatro em Testes e um em Implantação.

Tabela 3.2: Lista de Artefatos do Processo de Desenvolvimento de *Software*

Subprocesso	**Artefato**
Requisitos	PID - Plano de Iterações
Requisitos	DRI - Documentos de requisitos da intervenção
Requisitos	MDC - Modelo de dados conceitual
Requisitos	RNS - Requisitos de Níveis de Serviço
Análise e Projeto	DES - Documento de especificação de Serviço
Análise e Projeto	MER - Modelo entidade relacionamento
Análise e Projeto	DGA - Diagrama de atividade

Análise e Projeto	DGT - Diagrama de transação
Análise e Projeto	DPB - Diagrama de processamento batch
Análise e Projeto	DGP - Diagrama geral de procedures
Análise e Projeto	DTE - Diagrama de transição de estados
Análise e Projeto	DPS - Diagrama de projeto de *software*
Análise e Projeto	ESC - Especificação de componente
Análise e Projeto	ESI - Especificação de interface entre sistemas
Análise e Projeto	EST - Especificação de tela
Implementação	CDF - Código fonte
Teste de *Software*	PLT - Plano de testes
Teste de *Software*	RET - Roteiro de execução de testes
Teste de *Software*	RRT - Relatório de realização de testes
Teste de *Software*	RFT - Relatário final de testes
Implantação	PLI - Plano de Implantação

Fonte: Elaboração própria

Os artefatos considerados na documentação de sistema são: MDC, MER, CDF e RET. Todos são documentos de uso restrito à área técnica e têm a finalidade de auxiliar os analistas na construção e manutenção do *software*.

Em contrapartida a essa quantidade de documentação da área técnica, observou-se que os gestores de negócio do sistema MMO não possuem uma documentação sobre o funcionamento do sistema. A fonte de informação é os documentos de requisitos que a cada intervenção tornam-se desatualizados, do mesmo modo que as anotações pessoais que cada um faz ao criar ou atualizar uma funcionalidade do sistema, ou seja, grande parte do conhecimento sobre o legado não está registrada e nem disponível em ambiente corporativo, pois trata-se de conhecimento que o gestor adquiriu ao longo da vida, a partir de suas atividades sociais, conceitos, ideias e situações cotidianas. A transformação do conhecimento tácito em explícito para que possa ser compartilhado e difundido tem sido objeto de muito investimento investigativo, resultando em propostas metodológicas que buscam a passagem do individual e pessoal para o coletivo grupal [72].

No entanto, há também as situações em que o gestor não conhece certas particularidades e restrições do sistema. Nessas situações é comum a área de negócio solicitar aos desenvolvedores que consultem os códigos-fonte para verificarem o comportamento do sistema. Esse é um cenário de ameaça à segurança do sistema, visto que os programas, com o passar do tempo, sofrem atualizações e correções que podem indevidamente modificar o que foi definido pelos gestores de negócio inicialmente.

3.1.4 Contexto de gestão de riscos do processo de implementação de *software*

Dependendo do tipo de projeto e do ciclo de vida adotado, o processo de implementação utiliza entre 30 a 80% do tempo total de desenvolvimento, e são responsáveis por 50 a 75% dos erros totais, fatores que demonstram necessidade de estudo de melhorias para essa área [73].

Há estatísticas de mercado que mostram que os profissionais envolvidos com manutenção de *software* gastam, em média, cerca de 50% de esforço somente tentando entender os códigos produzidos no processo de implementação [74].

Essa dificuldade na implementação dos *softwares* reflete-se na entrega do *software* ao cliente, na qual, segundo [75], os usuários ficam insatisfeitos com 25% dos produtos comerciais para microcomputadores, 30% dos produtos para mainframes e 40% dos produtos feitos por encomenda.Nesse contexto, antes de qualquer iniciativa de mudança ou modificação no subprocesso de implementação, deve-se primeiramente investigar a causa de entrega de *softwares* em desacordo com o solicitado pelo cliente. Portanto, para se ter uma visão de alto nível do processo de implementação, a fim de conhecer os seus limites (escopo) e identificar, de forma correta, suas entradas e saídas, em detrimento do detalhe de suas operações, é que foi utilizado a ferramenta SIPOC.

Com ela, foi possível também identificar, em uma visão ampliada, quais são os elementos que interagem para o desenvolvimento e entrega do *software* ao cliente. Esse diagrama é o terceiro na sequência de processos observados e, portanto, denominado por SIPOC Nível 3, conforme apresentado na Figura 3.3.

A hierarquia dos processos em níveis é uma abordagem que permite à organização ter uma visão do todo, além de uma visão operacional de sua cadeia de processos e atividades.

O processo de implementação de *software* é formado por 10 atividades, conforme detalhado na Figura 3.3. São elas:

- Reunir as especificações de requisitos;
- Identificar as unidades de cada funcionalidade;

- Codificar as unidades;
- Verificar o funcionamento das unidades;
- Executar os testes das unidades;
- Esclarecer as dúvidas em relação à implementação;
- Documentar as informações de histórico;
- Verificar com base no Diagrama de Transação, se todas as unidades foram implementadas;
- Identificar os padrões de desenvolvimento adequados;
- Executar os testes de integração;

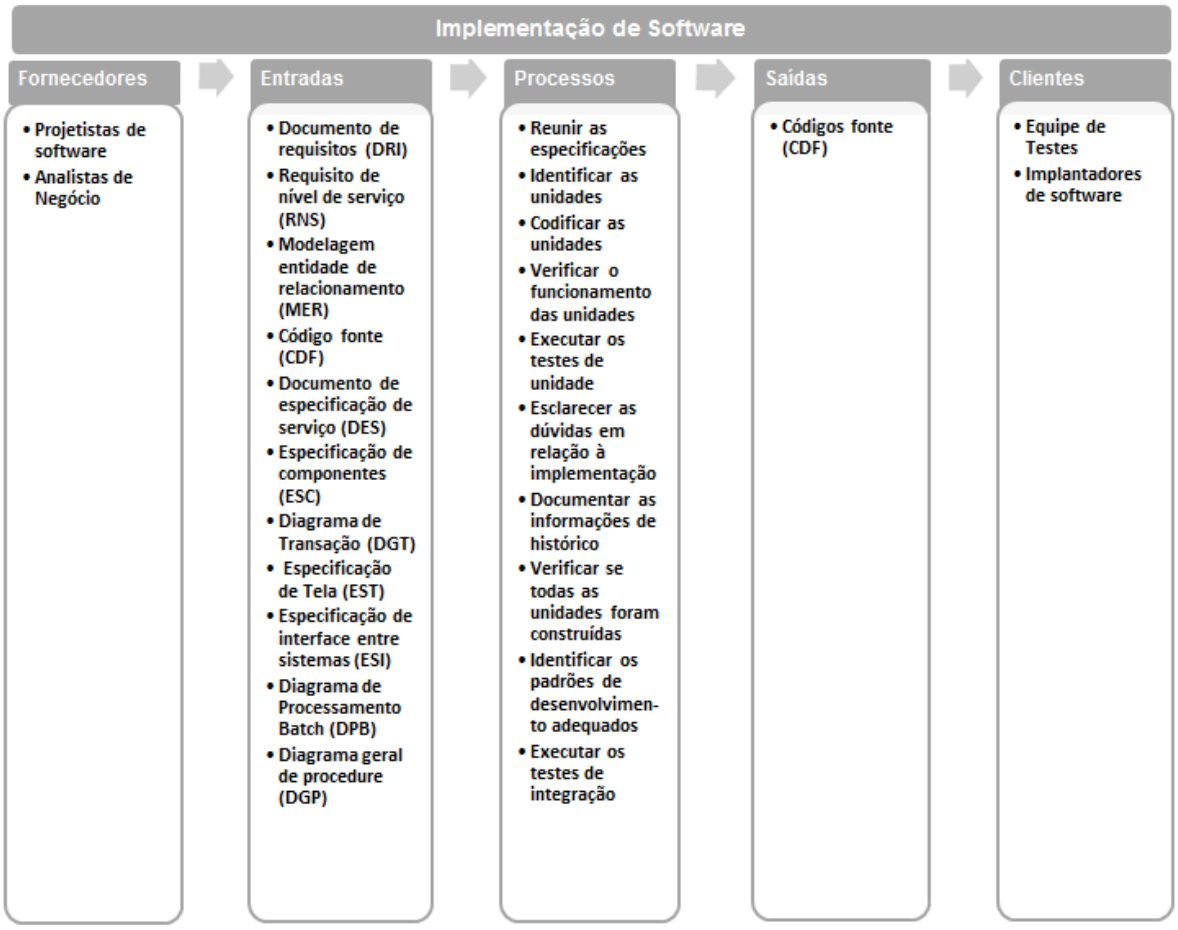

Figura 3.3: SIPOC Nível 3 - Processo de implementação de *software*

Fonte: Elaboração própria

Os fornecedores do processo de Implementação demonstrados na SIPOC Nível 3 são projetistas de *software* e analistas de negócio.

As entradas do processo de implementação são os artefatos previstos no manual de processo de desenvolvimento de *software* da Instituição e utilizados pelos desenvolvedores para iniciarem a codificação do *software*. Os artefatos podem ter sua origem em diferentes subprocessos:

- *Subprocesso de Requisitos*
  - Documento de Requisitos (DRI);
  - Requisito de Nível de Serviço (RNS);
- *Subprocesso de Análise e Projeto de Software*
  - Modelagem Entidade Relacionamento (MER);
  - Código-fonte (CDF);
  - Documento de Especificação de Serviço(DES);
  - Especificação de Componentes (ESC);
  - Diagrama de Transação (DGT);
  - Especificação de tela (EST);
  - Especificação de interface entre sistemas (ESI);
  - Diagrama de Processamento Batch (DPB);
  - Diagrama Geral de Procedure (DGP);

O documento de saída do processo de implementação de *software* é o código-fonte e os clientes receptores desse documentos são:

- *Equipe de testes*: faz parte dela os analistas responsáveis por elaborar e definir uma proposta de teste baseada em requisitos previamente estabelecidos. Também é sua responsabilidade conferir o comportamento do sistema, verificar a existência de falhas e inconsistências que não foram tratadas durante o processo de implementação e registrar os resultados obtidos;
- *Implantador de Software*: é responsável pela elaboração do documento de implantação, com a descrição de todas as atividades que devem ser realizadas para disponibilizar o produto ou funcionalidade no ambiente-alvo. É também o responsável por restabelecer o sistema ao seu estado original, caso ocorra algum problema.

Com o estabelecimento do contexto do processo de implementação por meio do uso da ferramenta SIPOC e do mapeamento de processo *AS IS*, buscou-se identificar, com o

auxílio dos desenvolvedores do sistema MMO, os principais riscos e vulnerabilidades que influenciam na entrega de um código-fonte de qualidade ao cliente.

Para atingir esse objetivo, foram elaborados um questionário *(Apêndice B)* e um formulário para entrevista semiestruturada *(Apêndice C)*, para tratar sobre as Atividades e Artefatos do processo de implementação. Os Artefatos e Atividades correspondem, respectivamente, aos elementos Entrada e Processos descritos no SIPOC Nível 3 (Figura 3.3).

Na próxima seção, será apresentado, em detalhes, as etapas realizadas na avaliação das atividades do Processo de Implementação, a aplicação do método AHP e a análise dos resultados, obtidos na consolidação dos julgamentos realizados individualmente pelos desenvolvedores entrevistados do sistema MMO.

3.2 Identificação dos Riscos

Esta fase do processo de gestão de risco é bastante crítica, pois somente riscos identificados poderão ter resposta adequada [76]. Portanto, os riscos não identificados são uma ameaça à segurança e à estabilidade do sistema, porque não há um plano de mitigação ou eliminação deles. Caso surja algum risco que não foi identificado, o processo de implementação do sistema continuará desprotegido.

3.2.1 Avaliação das atividades do processo de implementação de *software*

Para participar da avaliação das atividades do processo de implementação de *software*, os desenvolvedores do sistema MMO tinham que ter o seguinte perfil:

- formadores de opinião;
- conhecer os papéis e subprocessos relacionados ao processo de desenvolvimento de *software* para alta plataforma (mainframe);
- conhecer os papéis e subprocessos responsáveis pela realização das atividades do processo de desenvolvimento de *software* para alta plataforma (mainframe);
- ter experiência de mais de cinco anos no processo de desenvolvimento de *software*, bem como ter realizado o papel de tíder técnico na condução de projetos;
- formação acadêmica em cursos relacionados a área de informática;

Esse perfil de avaliadores buscou selecionar desenvolvedores influentes que tivessem maior acesso às demais pessoas da equipe em razão de seu papel e importância na construção dos *softwares*, e com conhecimento não apenas dos processos técnicos, mas também

de aspectos estratégicos e formais da organização e, que participassem dos planejamentos e decisões da equipe.

Apesar do sistema MMO contar com um total de 12 desenvolvedores, apenas cinco estavam aptos a responder ao questionário, quatro tinham menos de cinco anos no processo de desenvolvimento de *software* e três não possuíam experiência na condução de projetos para alta plataforma. Os cinco desenvolvedores selecionados eram analistas sênior, todos com mais de sete anos de experiência na implementação de *software*.

3.2.1.1 Critérios de avaliação das atividades do Processo de Implementação

Os critérios de Avaliação das Atividades do processo de implementação de *software* foram selecionados tendo como premissa ser parâmetros de fácil compreensão e julgamento pelos avaliadores.

Para a seleção dos critérios, foi realizada uma reunião em que participaram os cinco desenvolvedores (analistas seniores), que indicaram seis critérios relevantes na avaliação das atividades do processo de implementação para a entrega de um *software* de qualidade. São eles:

- complexidade;
- dificuldade;
- eficiência;
- facilidade;
- importância;
- riscos.

Nessa reunião foi possível obter o consenso confiável de opiniões dos avaliadores, o que permitiu definir com clareza, quais critérios o grupo concordava sobre sua utilização e consequentemente quais não deveriam ser utilizados. Em alguns casos, segundo [77] é importante fazer uma pré-avaliação dos critérios envolvidos no processo de tomada de decisão, pois pode ser necessário reduzir a quantidade de alternativas para julgamento a fim de para diminuir a complexidade das avaliações. Dessa forma foram excluídos:

- complexidade;
- eficiência;
- facilidade.

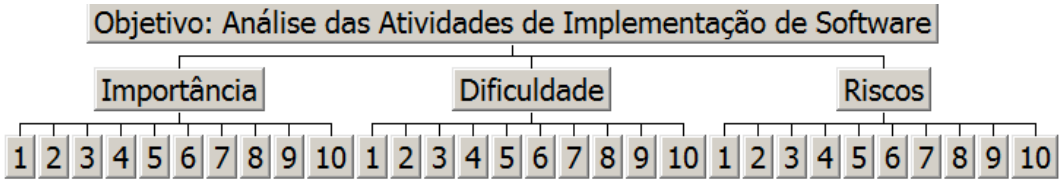

Figura 3.4: Hierarquia do critérios e alternativas do processo de Implementação de *software*

Fonte: Elaboração própria

Os critérios complexidade e eficiência mostraram-se parâmetros subjetivos e de difícil avaliação. Já os critérios facilidade e dificuldade mostraram-se parâmetros redundantes, pois, ao se avaliar um, implicitamente avalia-se o outro na proporção inversa. Dessa forma, foram selecionados os critérios importância, dificuldade e risco, apresentados na Tabela 3.3.

Tabela 3.3: Critérios de avaliação das atividades de Implementação de *software*

Critério	**Descrição**
Importância	Grau de prioridade e relevância de uma atividade
Dificuldade	Grau de esforço e trabalho na realização de uma atividade
Risco	Grau de incerteza, possibilidade de uma atividade falhar

Fonte: Elaboração própria

3.2.1.2 Estabelecimento dos níveis hierárquicos de avaliação das atividades do Processo de Implementação

Sabendo-se que as opiniões são baseadas em critérios tangíveis e intangíveis, arbitrariamente escolhidos por quem toma a decisão, foi empregada a técnica de decisão multicritério Analytic Hierarchy Process (AHP), que é um dos principais modelos matemáticos para apoio à teoria de decisão disponível no mercado [78]. Para apoiar no julgamento das atividades dos processos de implementação foi aplicada essa ferramenta de apoio à decisão, com a definição de uma estrutura hierárquica dos critérios e alternativas, conforme apresentado na Figura 3.4.

Nessa avaliação não houve necessidade de definição de subcritérios, pois o estabelecimento de apenas um nível hierárquico com os critérios foi suficiente para a realização dos julgamentos e alcance dos objetivos pretendidos.

A hierarquização é um processo bastante útil, porque permite observar, de forma clara e objetiva, qual o objetivo pretendido com a aplicação da ferramenta AHP, mediante o julgamento e avaliação dos critérios previamente elencados (importância, dificuldade e risco). As 10 atividades do processo de implementação de *software* que foram utilizadas para análise e julgamento foram:

As 10 atividades do processo de implementação de *software* que foram utilizadas para análise e julgamento são:

1. identificar as unidades;
2. reunir as especificações;
3. identificar os padrões de desenvolvimento adequados para cada unidade;
4. esclarecer as dúvidas em relação à implementação;
5. documentar as informações de histórico da unidade que está sendo implementado;
6. codificar as unidades;
7. verificar se o funcionamento das unidades corresponde ao resultado esperado na especificação;
8. executar a atividade de teste de unidade;
9. verificar com base no Diagrama de Transação se todas as unidades estão codificadas;
10. executar a atividade de teste de integração.

3.2.1.3 Definição da escala de avaliação das atividades do processo de Implementação

Para avaliar os critérios e as atividades do processo de implementação, foi utilizada a escala fundamental de Saaty, conforme Tabela 3.4.

Um dos motivos dessa escala ser de 1 a 9 é a habilidade para fazer distinções qualitativas, representada pelos atributos: igual, importância pequena, importância grande, importância muito grande, importância absoluta; além dos atributos de valores intermediários. Com essa escala, segundo [79], é possível fazer a relação entre os atributos com grande correção e precisão.

Tabela 3.4: Escala Fundamental de Saaty

Intensidade	**Pontuação**	**Forma de Avaliação**

1	Igual importância	As duas atividades contribuem igualmente para o objetivo
3	Importância pequena de uma sobre a outra	A experiência e o juízo favorecem uma atividade em relação à outra
5	Importância grande ou essencial	A experiência ou juízo favorece fortemente em relação à outra
7	Importância muito grande ou demonstrada	Uma atividade é muito fortemente favorecida em relação à outra. Pode ser demonstrada na prática
9	Importância absoluta	A evidência favorece uma atividade em relação à outra, com o mais auto grau de segurança
2, 4, 6, 8	Intermediários	Quando se procura uma condição de compromisso entre duas definições

Fonte: SAATY [79]

3.2.1.4 Avaliação das atividades do processo de implementação de *software*

Como a aplicação envolveu um grande número de alternativas e avaliadores, foi utilizado o Expert Choice para auxiliar os cálculos dos índices e a combinação dos julgamentos individuais de cada participante. Esse *software* possui natureza interativa que simplifica significativamente a construção das hierarquias e o cálculo das prioridades [80]. Ele transforma decisões complexas em comparações par a par e sintetiza os resultados, proporcionando aos gestores a possibilidade da melhor escolha, tendo instrumentos e informações para a sua tomada de decisão. O Expert Choice apresenta uma síntese dos dados e o desenvolvimento de preferências. Sua aplicação não requer um alto grau de conhecimento sobre análises estatísticas, e a vantagem é que proporciona benefícios como economia de tempo, consenso e alinhamento das decisões aos objetivos da organização.

A montagem hierárquica dos critérios e das alternativas no Expert Choice disponibiliza uma tabela matricial que permite determinar o peso de uma atividade em relação a outra mediante um determinado critério.

Na Tabela 3.5 é apresentado um resumo do processo realizado para avaliação das Atividades do processo de implementação:

Tabela 3.5: Resumo do processo de avaliação das Atividades

Avaliação das Atividades	
Objetivo	• avaliar as atividades pelos critérios "importância", "dificuldade"e "risco".
Técnica de Coleta de Dados	• questionário.
Participantes	• 5 desenvolvedores, com o seguinte perfil: – formadores de opinião; – experiência de mais de cinco anos condução de projetos (Líder Técnico); – conhecer os processos e papéis para alta plataforma; – formação acadêmica na área de informática.
Ferramenta de Avaliação	• AHP - Método Multicritério de Apoio a Decisão.

Fonte: Elaboração Própria

No próximo tópico será será apresentado a consolidação dos julgamentos de cada um dos avaliadores realizada pelo Expert Choice, bem como a análise e interpretação dos resultados.

3.2.1.5 Resultados da avaliação dos critérios e atividades do processo de implementação de *software*

A primeira avaliação realizada com os participantes por meio do uso da ferramenta AHP foi sobre a relevância e contribuição dos critérios importância, dificuldade e risco no processo de implementação para a na entrega de um *software* de acordo com o esperado pelo cliente.

Na Figura 3.5 é possível observar que o critério "importância"foi o que obteve o melhor resultado na preferência dos avaliadores, com um percentual de 62,5%, em detrimento dos demais critérios "risco"e "dificuldade"que alcançaram, respectivamente, 24,0% e 13,5%.

Durante o processo de resposta aos questionários, os participantes relataram que todos os critérios avaliados contribuem em proporções diferenciadas na conformidade do *software*. Porém, enfatizaram que alguns critérios, apesar de essenciais para a implementação de um *software* de qualidade, não são tão perceptíveis pelo cliente. O critério "importância" obteve a preferência dos avaliadores por se tratar de um parâmetro capaz de indicar em uma escala de preferência, quais são as atividades essenciais na entrega do *software* ao cliente, não desmerecendo, entretanto, a relevância dos critérios "dificuldade" e "riscos" na avaliação de aspectos como "desempenho", "segurança" e "manutenibilidade das aplicações".

3.2.1.5.1 Avaliação das atividades pelo critério "importância"

Depois de definido o critério de maior relevância na opinião dos avaliadores, passamos para a etapa de consolidação das opiniões e julgamentos realizados individualmente por cada um dos desenvolvedores, na avaliação das atividades do processo de implementação.

Na Figura 3.6, observa-se que a atividade considerada mais importante é "verificar se o funcionamento da unidade corresponde ao resultado esperado na especificação", com um percentual de 15,10%.

Para os avaliadores, desenvolver um *software* que atenda às necessidades de negócio da organização e que funcione corretamente é a garantia da aceitação e satisfação do cliente com o *software* entregue pela área de TI. Porém, esse não é um objetivo tão fácil de ser alcançado, pois existem alguns fatores que influenciam na realização das atividades do processo de implementação, tais como o acúmulo de papéis (responsabilidades) e a ausência de uma fonte única e atualizada de documentação do sistema.

Durante o processo de desenvolvimento de *software*, é comum, aos analistas, desempenharem diferentes papéis (Tabela 3.1) e, consequentemente, não dedicarem tempo suficiente na realização das atividades e de produção dos artefatos necessários ao processo de implementação.

Figura 3.5: Resultado da avaliação dos critérios

Fonte: Elaboração própria

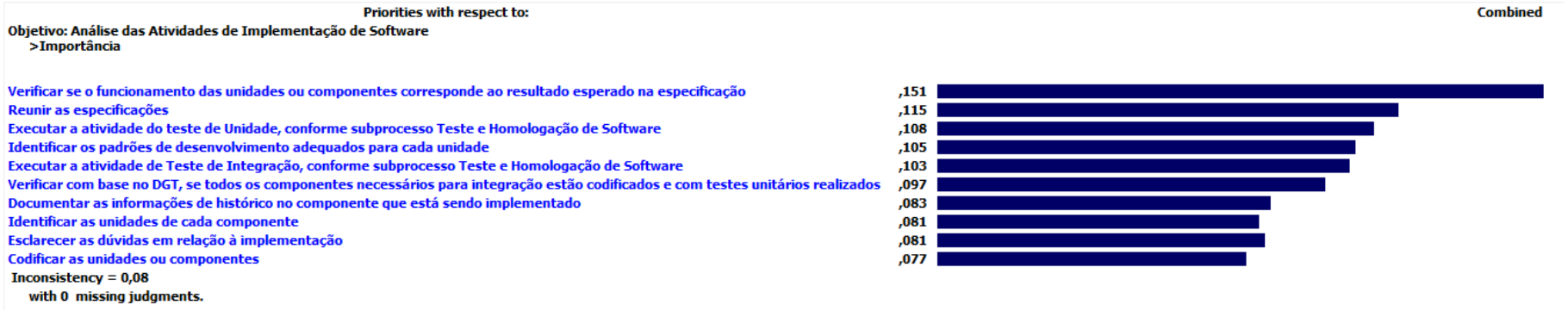

Figura 3.6: Resultado da avaliação das atividades para o critério importância

Fonte: Elaboração própria

O acúmulo de tarefas para a realização em um curto espaço de tempo predispõe os analistas a ignorarem algumas etapas do processo de desenvolvimento, comprometendo o resultado do processo de implementação com a elaboração de códigos-fonte mal escritos, sem padronização e com elevado grau de complexidade para manutenção.

Os avaliadores também mencionaram sobre a dificuldade de se obter informação atualizada sobre o funcionamento do sistema. A ausência de uma fonte única de informações e documentação do legado contribui para situações de impasse e divergências entre os desenvolvedores e gestores de negócio, pois cada área tem uma visão parcial e pessoal sobre o funcionamento, sequência de atividades e processos da aplicação.

Existem situações em que os desenvolvedores se deparam com especificações e necessidades das áreas de negócio que contrariam o objetivo do sistema, que, se implementados, poderiam expor as aplicações, a indisponibilidades, a perda de integridade e as inconsistências nas informações.

Nesse contexto, os desenvolvedores, além de responsáveis pela construção do *software*, acabam desempenhando papel de guardiões e também de consultores sobre o comportamento e funcionamento das aplicações. Isso se deve por causa da inexistência de uma fonte de documentação de sistema confiável e atualizada, para esclarecimento de dúvidas e apoio à tomada de decisão frente às constantes mudanças e necessidades impostas melo mercado.

Ainda no critério “importância”, pode-se observar que a atividade de menor relevância considerada pelos avaliadores é “codificar as unidades ou componentes”, com um percentual de 7,70%. Com esse resultado, compreende-se que, independente da forma como os códigos-fonte estejam escritos ou estruturados, o que realmente importa é que o *software* apresente o comportamento e resultado esperado pelo cliente.

Nesse contexto nota-se que a utilização das normas, padrões e melhores práticas pelos desenvolvedores do sistema MMO estão condicionados ao não comprometimento dos prazos acordados com o cliente e, também, em menor proporção, de aspectos relacionados à praticidade, à agilidade e ao desempenho na realização das atividades do processo de implementação.

3.2.1.5.2 Avaliação das atividades pelo critério "risco"

Na avaliação do critério “risco”, os analistas e gerentes de equipe indicaram, conforme Figura 3.7, que a atividade que oferece maior incerteza no alcance do objetivo é “reunir as especificações”. O armazenamento e recuperação dos requisitos impõem grande fragilidade ao processo de implementação de *software*, dada a dificuldade em reunir todas as necessidades e restrições do produto solicitado pelo cliente.

Ainda no critério “riscos”, verificamos que a segunda e terceira atividades são as que oferecem maior risco ao processo de implementação de *softwares*, conforme podemos observar em “executar a atividade de teste de integração”, com 12% e “executar a atividade de teste de unidade” com 11,6% , respectivamente. Esse resultado ratifica o entendimento de que os avaliadores se preocupam mais com a entrega e satisfação do cliente com o *software*, do que com a qualidade dos processos e meios que viabilizam a entrega do produto final.

Na seção a seguir, será apresentado o resultado da avaliação das atividades pelo critério “dificuldade”, que, por abordar aspectos mais técnicos relacionados à diversidade e excesso de tarefas para a realização do processo de implementação, os desenvolvedores manifestaram maior facilidade na realização dos julgamentos.

3.2.1.5.3 Avaliação das atividades pelo critério "dificuldade"

Depois de realizado o julgamento das atividades do processo de implementação pelo critério "importância", os participantes seguiram para a avaliação das atividades pelo critério "dificuldade". Neste momento, os avaliadores já demonstravam maior familiaridade com o método AHP, bem como maior facilidade na atribuição de pesos, a cada uma das alternativas (atividades), dentro do contexto (critério) selecionado para avaliação.

Na Figura 3.8, é possível observar que mais de um quarto da dificuldade na realização do processo de implementação está relacionado às atividades de teste de *software* unitário[1] e de integração[2] que, juntos, totalizam um percentual de 25,9%.

A atividade “executar a atividade de teste de integração” foi considerada, pelos avaliadores, como o procedimento que demanda maior esforço para a sua realização, apresentando um percentual de 13,4% e, em segundo lugar, a atividade “executar a atividade do teste de unidade”, com um percentual de 12,5%.

Os avaliadores consideraram essas duas atividades como sendo as mais difíceis, em razão, principalmente, da dificuldade na elaboração de documentos de planos de testes que compreendessem todos os possíveis cenários de erros e falhas nas aplicações. Eles relataram que, com o passar do tempo, à medida que os sistemas crescem e tornam-se complexos e difíceis de serem mantidos, a qualidade e a completude dos artefatos são elementos imprescindíveis na entrega de um *software* de qualidade para o cliente.

A execução de um teste normalmente costuma ser rápida e efetiva [82]. No entanto, a execução e repetição de um vasto conjunto deles manualmente é uma tarefa muito dispendiosa e cansativa. Nesse contexto é comum que os desenvolvedores não realizem

[1]Unitário: tem por objetivo explorar a menor parte unidade do sistema, procurando falhas em em decorrência de defeitos em lógica e de implementação, em cada módulo separadamente [81].

[2]Integração: tem por finalidade identificar falhas associadas às interfaces entre os módulos quando esses são integrados para construir a estrutura do *software* [81]

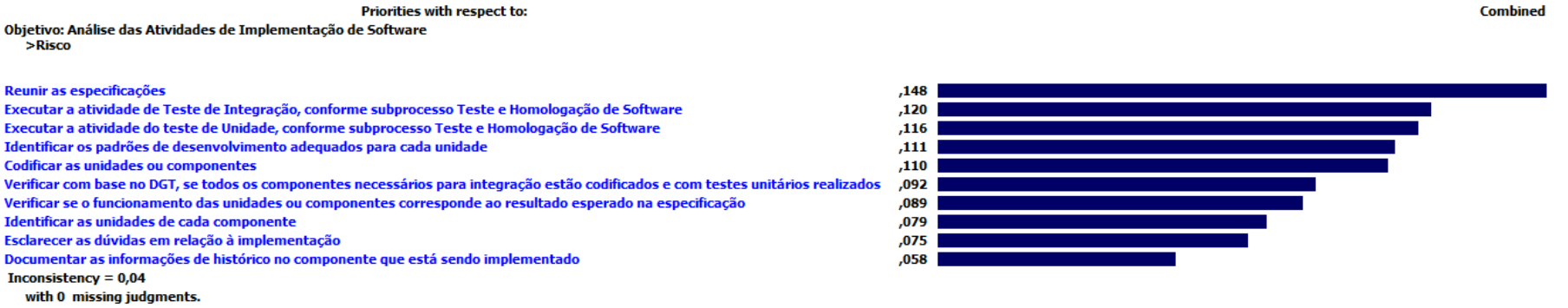

Figura 3.7: Resultado da avaliação das atividades para o critério risco

Fonte: Elaboração própria

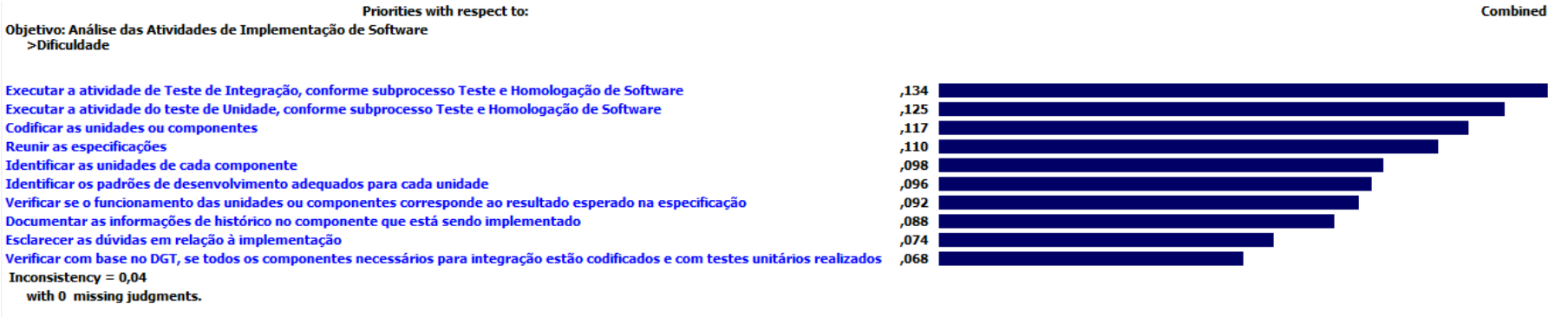

Figura 3.8: Resultado da avaliação das atividades para o critério dificuldade

Fonte: Elaboração própria

todos os casos de teste, a cada nova mudança ou correção no código-fonte. Tal fato expõe a aplicação a riscos, pois, os testes unitários são responsáveis por remover entre 30% e 50% dos defeitos em programas [83].

Atualmente, com a tendência de terceirização do processo de construção do *software*, os problemas com a qualidade dos produtos e serviços de TI têm se tornado cada vez mais evidentes. Uma das principais razões das inconsistências e erros nas aplicações deve-se à baixa qualidade dos documentos elaborados. Para se ter um *software* de qualidade, não basta fornecer as especificações do que está sendo modificado. A aplicação deve também ser fonte de consulta de testes anteriores, para que os desenvolvedores realizem não apenas os testes do que foi modificado, mas também de partes não alteradas dos códigos-fonte, para se certificarem de que o comportamento não foi alterado indevidamente pela intervenção.

Conforme relatado pelos avaliadores, normalmente os documentos produzidos ao fim de um projeto de *software* não acompanham as modificações e evoluções dos sistemas, tonando-se desatualizados e fonte não confiável de informação.

Essa situação causa grande retrabalho e desgaste entre os desenvolvedores, que perdem muito tempo para identificar e corrigir erros, e prejuízo aos clientes, que, entre outros problemas, sofrem com atrasos na entrega de *softwares*, por vezes de qualidade duvidosa, construídos a partir de documentos até mesmo incompletos e mal elaborados.

A terceira atividade considerada de maior "dificuldade" no processo de implementação é "codificar as unidades", com um percentual de 11,7%. Os avaliadores relataram que esta é uma atividade na qual é muito importante a adoção de padrões de desenvolvimento, a fim de facilitar o entendimento entre os desenvolvedores, bem como agilidade em manutenções, em razão de estudos demonstrarem que cerca de 50% do tempo dos desenvolvedores é gasto no processo de compreensão do código que estão mantendo ou evoluindo [84], [85].

A qualidade e simplicidade na codificação tem relação direta com a sua robustez, e como aspectos relacionados à estrutura, design, sintaxe, caminhos lógicos e conjuntos de condições ou laços. Assim, percebe-se que uma codificação mal realizada pode resultar em grande retrabalho aos desenvolvedores, a ponto de tornar a atividade de manutenção tão custosa, que talvez a melhor solução seria reconstruí-lo completamente.

3.2.2 Avaliação dos artefatos do processo de implementação de *software*

Para avaliação dos artefatos do processo de implementação, manteve-se o perfil selecionado para participação dos questionários, com a diferença de que o tempo mínimo de experiência no processo de desenvolvimento de *software* passava de 5 para 2 anos. Os

demais critérios foram todos mantidos. Neste contexto, estavam aptos a participar da entrevista oito desenvolvedores, de um total de doze funcionários do sistema MMO, onde três não conheciam o processo de desenvolvimento para alta plataforma e um tinha menos de dois anos de experiência no processo de desenvolvimento.

Na Tabela 3.6 é apresentado um resumo do processo realizado para avaliação dos Artefatos:

Tabela 3.6: Resumo do processo de avaliação dos Artefatos

Avaliação dos Artefatos	
Objetivo	• analisar na opinião dos desenvolvedores qual a contribuição dos artefatos para o processo de implementação de *software*.
Técnica de Coleta de Dados	• entrevista
Entrevistados	• 8 desenvolvedores, com o seguinte perfil: – formadores de opinião; – experiência de mais de dois anos no processo de desenvolvimento de *software* para plataforma alta; – formação acadêmica na área de informática.
Técnica de Avaliação	• agrupamento e padronização das informações em categorias

Fonte: Elaboração Própria

As perguntas utilizadas na entrevista com os desenvolvedores sobre os Artefatos do

processo de implementação encontram-se listadas ao final deste trabalho, no *Apêndice C*. Logo abaixo, é apresentada a consolidação dos resultados obtidos, de acordo com cada um dos questionamentos realizados.

Pergunta: *Durante o processo de desenvolvimento de software para alta plataforma (mainframe) são previstos a elaboração de alguns artefatos. Quais você deveria utilizar, mas não utiliza? E por quê?*

Para os entrevistados foram apresentados uma lista com os 21 artefatos (Tabela 3.2), sendo que desses, 33% dos artefatos, nunca haviam sido utilizados pelos desenvolvedores no processo de desenvolvimento de *software*. São eles:

1. MDC - Modelo de dados conceitual;
2. RNS - Requisitos de nível de serviço;
3. DES - Documento de especificação de serviço;
4. DGA - Diagrama de atividade;
5. DPS - Documentação de *software*;
6. ESI - Especificação de interface entre sistemas ;
7. RFT - Relatório final de Testes.

Na Figura 3.9, buscou-se apresentar quais subprocessos apresentam maior quantidade de artefatos não utilizados durante o processo de desenvolvimento de *software*. Nele é

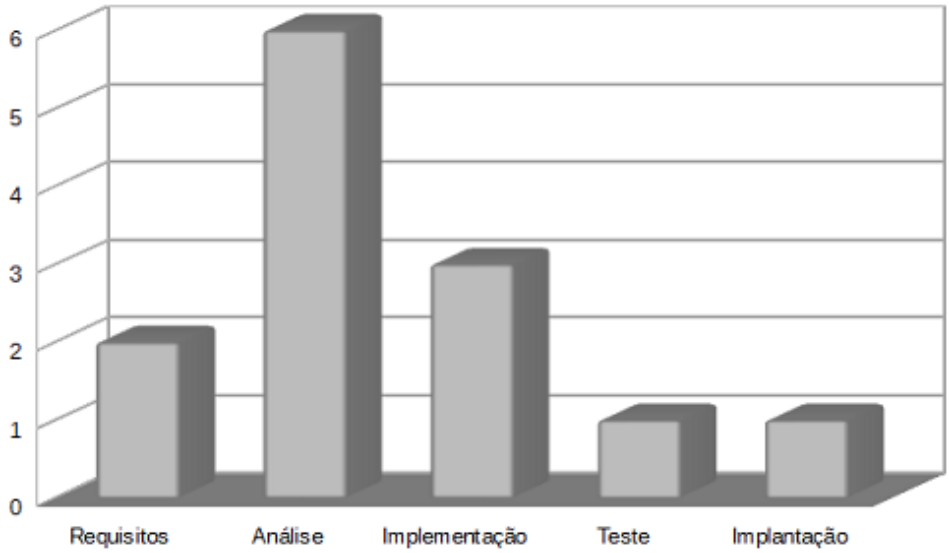

Figura 3.9: Documentos não utilizados por Subprocessos

Fonte: Elaboração própria

possível observar que os subprocessos mais afetados são Análise e Projeto de *Software* (com seis documentos) e o Processo de Implementação (com três), seguida pelos subprocessos, em menor proporção, de Requisitos (com apenas dois documentos), Testes (com um) e Implantação (com um documento).

Na Figura 3.10 está representado o percentual de frequência dos motivos apontados pelos entrevistados como justificativa para não utilização dos artefatos. Nela pode-se observar que o motivo *"não conhece"* ocorre em aproximadamente 62% das respostas, sendo que o restante dos motivos juntos representam o percentual de frequência de 38% das justificativas para não utilização dos artefatos. Os entrevistados relataram que o processo de desenvolvimento de *software* atualmente adotado no sistema MMO, bem como em todas as demais aplicações da instituição, é bastante burocrático e moroso, com um excesso de documentos para serem elaborados, nos quais grande parte do tempo de implementação é dedicado mais aos formalismos do processo, do que realmente com o desenvolvimento de um *software* de qualidade para o cliente.

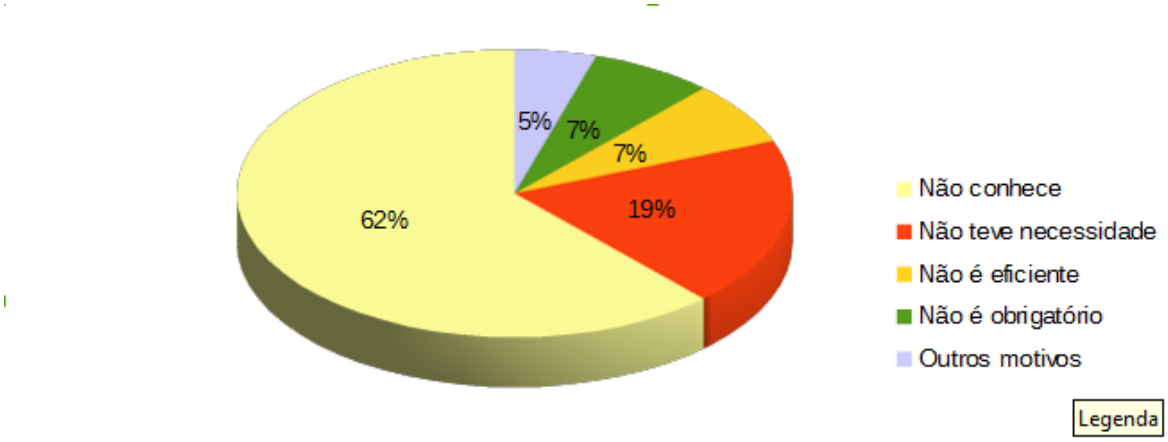

Figura 3.10: Motivos de não utilização dos artefatos do processo de desenvolvimento de *software*

Fonte: Elaboração própria

Pergunta: *Os artefatos são fonte de consulta atualizada e confiável sobre as funcionalidades do sistema?*

Nesta pergunta 85% dos entrevistados consideraram os artefatos fonte de consulta desatualizados e não confiáveis para explicarem o funcionamento dos sistemas. Apenas 15% apontou o CDF e o MER como fonte segura e confiável de pesquisa para a tomada de decisão, já que ambos são de documentos técnicos desenvolvidos durante as fases de Análise e Projeto de *Software* e Implementação, respectivamente. Apesar de terem características de documentos de sistemas, eles são acessíveis aos gestores de negócio e

foram elaborados a partir de análise das necessidades dos clientes, e com o passar do tempo, vão evoluindo aumentando, assim, sua complexidade

Pergunta: *Existe documentação de sistema acessível tanto aos desenvolvedores (área técnica) quanto aos gestores (área de negócio) sobre as funcionalidades e funcionamento do sistema ?*

Todos os entrevistados disseram desconhecer a existência de um documento que atenda tanto à área de negócio quanto à área técnica. Eles manifestaram haver uma lacuna de entendimento entre a os desenvolvedores e gestores de negócio sobre o funcionamento do sistema MMO, em que muitas vezes não é possível chegar a um consenso, sendo necessário, muitas vezes, recorrer ao código-fonte para esclarecimento de dúvidas sobre o comportamento da aplicação.

Nessa situação, fica claro a vulnerabilidade do processo de implementação, em que, para obter informações sobre o comportamento e regras do sistema, tem que se recorrer a documentos que podem estar em desacordo com o originalmente definido pela área de negócio, já que o código-fonte é um artefato em constante mudança, e passível de erros em sua implementação.

Todos os entrevistados foram unânimes em dizer que a existência de um documento de sistema acessível a ambas as áreas, proporcionaria grandes benefícios e melhorias ao processo de implementação, tais como: maior clareza na comunicação, menos retrabalho e maior segurança e agilidade nas especificações.

Pergunta: *Qual o artefato do processo de implementação que você considera mais importante, e o que poderia ser feito para que ele apresentasse melhores resultados?*

Todos os entrevistados, conforme apresentado na na Figura 3.11 apontaram o código-fonte (CDF), seguido pelo modelo de dados (MER), como os artefatos que mais contribuem para o processo de implementação. Eles relataram que alguns documentos apesar de serem muito importantes para o processo de implementação, a exemplo do DRI e do RRT, atualmente a forma como são elaborados, pouco contribuem para a qualidade e segurança do *software*, pois há uma cultura em privilegiar os trâmites burocráticos de iniciar e finalizar os subprocessos de desenvolvimento dentro dos prazos estabelecidos em cronograma, em detrimento da segurança e qualidade do *software* produzido.

Na Figura 3.11, apenas estão representados os artefatos que foram apontados por no mínimo dois entrevistados. Os artefatos que foram indicados por apenas um ou nenhum entrevistado não foram representados na figura. Os desenvolvedores apontaram o código-fonte como o documento mais importante, em razão de sua importância como fonte de

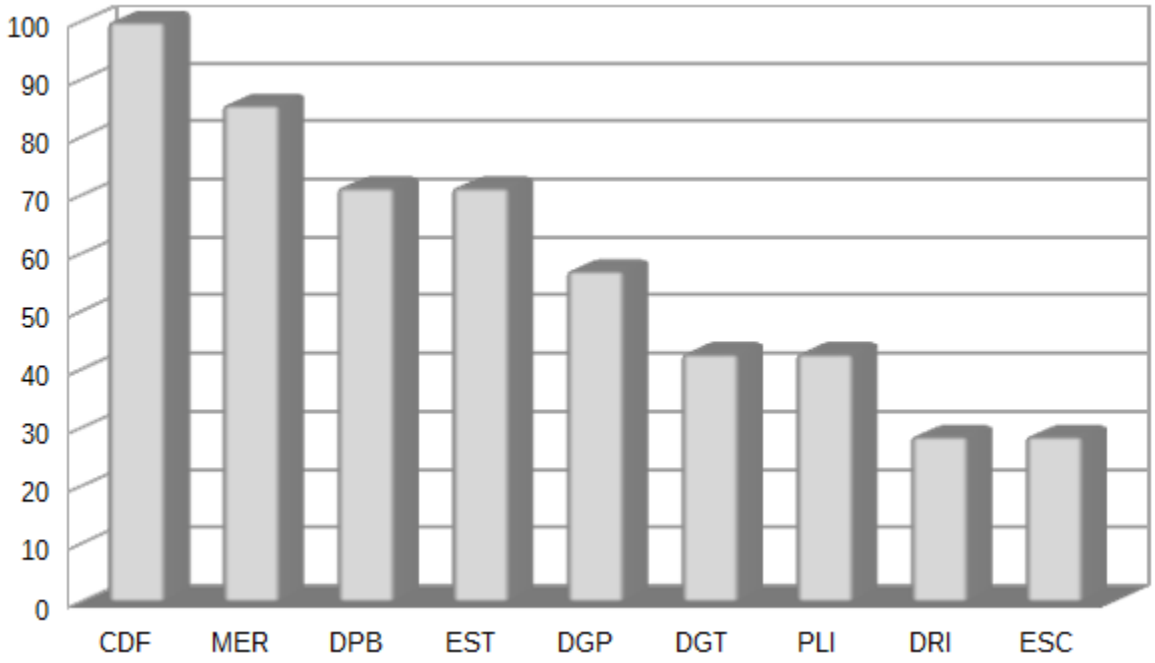

Figura 3.11: Grau de importância dos artefatos para o processo de implementação

Fonte: Elaboração própria

informação atualizada do sistema e também pelo fato de ele ser o artefato final responsável pela satisfação ou não do cliente com o *software* entregue.

Os entrevistados apontaram alguns fatores que prejudicam a qualidade do código-fonte e que consequentemente geram mais custos e retrabalho na manutenção dos sistemas:

- *Padronização*: apesar de existir um manual com as melhores práticas de desenvolvimento para codificação em linguagem de programação estruturada *Cobol*, poucos desenvolvedores a utilizam. Muito tempo se gasta na análise dos códigos-fonte para entender o seu funcionamento. Essa atividade seria bem mais ágil, se todos os códigos estivessem implementados com um mesmo padrão, onde o foco seria apenas nas regras de negócio.

- *Segregação em camadas*: A segregação das responsabilidades executadas pelos programas, separada em camadas de apresentação, negócio e persistência, é uma das orientações contidas no manual de melhores práticas adotado pela Instituição para a implementação do *software*. Essa orientação tem por objetivo evitar problemas, como a dificuldade de compreensão e manutenção do *software*, a dificuldade de reaproveitamento e reúso dos programas e a rigidez do *software* em que a necessidade de alteração em uma determinada função do sistema compromete o funcionamento de outras funções do sistema em virtude de alta dependência entre as funções. A

segregação de funcionalidades em camadas de Apresentação, Negócio e Persistência contribui para que os programas sejam reutilizados e evitam a redundância de código.

Pergunta: *Você conhece ou já ouviu falar sobre gerenciamento de processos de negócio (BPM)? Você sabe que o banco possui ferramenta disponível para mapeamento de processos?*

Cerca de 72% dos entrevistados responderam conhecer sobre gerenciamento de processos apenas na teoria, sem nenhum tipo de experiência prática na sua aplicação. Eles manifestaram conhecimentos sobre seus benefícios para as organizações, a partir de leituras de artigos e revistas sobre o assunto, enfatizando que um dos grandes diferenciais do modelo é permitir a visualização dos processos na organização, facilitando, deste modo, a identificação de processos redundantes ou pouco eficientes para, a partir de então, propor melhorias.

3.2.3 Riscos identificados no processo de implementação

Com base nos resultados obtidos por meio dos questionários e entrevista sobre as Atividades e Artefatos do processo de implementação, foi possível identificar os riscos apresentados abaixo:

1. retrabalho;
2. *software* em desacordo ao desejado pelo cliente;
3. descumprimento dos prazos estabelecidos em cronograma;
4. incidentes e erros nas implantações dos *softwares*;
5. divergências e desentendimentos entre membros da equipe;
6. artefatos incompletos, mal elaborados e desatualizados;
7. *software* complexo de difícil manutenção;

Nessa etapa, ainda não há um nível de hierarquia definido entre os tipos de riscos. Os números são apenas para facilitar a identificação das vulnerabilidades individualmente. A partir deste ponto, faz-se necessário analisá-los sobre diferentes aspectos, a fim de elaborar a melhor estratégia de resposta. Esse é o objetivo a ser alcançado na próxima seção, com a realização da atividade de análise de riscos, recomendada pela norma ABNT NBR ISO 31000:2009 de gestão de riscos.

3.3 Análise dos Riscos

Na realização dessa atividade, buscou-se determinar o nível de impacto (consequências) e probabilidade de ocorrência de cada uma das vulnerabilidades levantadas na atividade de identificação dos riscos. Os métodos utilizados na análise dos riscos, segundo a norma ABNT NBR ISO 31010:2009 [43], podem ser qualitativos, semi-quantitativos e quantitativos.

A análise qualitativa define consequência, probabilidade e nível de risco por níveis de significância, tais como: alto, médio e baixo.

O método quantitativo estima valores práticos para "consequência" e "probabilidade", e produz valores de nível de risco em unidades definidas quando se desenvolveu o contexto.

O método semi-quantitativo, utilizado na realização deste estudo, utilizou escalas de classificação numérica para "consequência" (impacto) e "probabilidade" e a sua combinação para produzir um nível de risco a partir de uma fórmula. As escalas podem ser lineares, ou logarítmicas, ou podem ter alguma outra relação. Além do que, as fórmulas utilizadas também podem variar [41].

Os mesmos analistas que participaram das entrevistas durante a fase de identificação dos riscos, atribuíram pesos para a probabilidade e impacto de ocorrência, utilizando a escala de 1 a 5, conforme apresentado na Tabela 3.7.

Tabela 3.7: Escala de análise dos riscos do processo de implementação

Escala	**Probabilidade**	**Impacto**	**Nível**
5	é quase certa (p > 95%)	Afetará todo o negócio da empresa e os prejuízos são extremamente altos	Muito Alto
4	é muito provável (65% < p < 95%)	Afetará um ou mais negócios da empresa e os prejuízos são muito altos.	Alto
3	é provável (35% < p < 65%)	Afetará uma parte do negócio da empresa e os prejuízos são razoáveis	Médio
2	é pouco provável (5%< p <35%)	Afetará uma parte pequena e localizada do negócio da empresa e os prejuízos são baixos	Baixo

1	improvável (p <5%)	Afetará uma parte muito pequena e localizada do negócio e os prejuízos são desprezáveis	Muito Baixo

Fonte: Adaptado da Tabela explicativa da gestão de riscos [86]

A combinação dos critérios probabilidade e impacto resultaram no cálculo da severidade do risco, que encontra-se representado pela equação:

$$\text{Severidade} = \text{Probabilidade} \times \text{Impacto}$$

Os resultados encontrados para "probabilidade" e "impacto" foram obtidos por meio da média aritmética dos pesos atribuídos individualmente por cada um dos entrevistados, e o índice de severidade foi realizado a partir da combinação dos dois. A Tabela 3.8 detalha cada um dos riscos, bem como os resultados obtidos, a partir desse cálculo. Nela é possível observar que o descumprimento dos prazos é apontado pelos entrevistados como a principal vulnerabilidade a ser mitigada ou eliminada do processo de implementação. É uma das principais causas de ansiedade e aborrecimento no relacionamento entre gestores e desenvolvedores, pois, nos casos de demandas legais estabelecidas por órgãos reguladores e/ou auditorias, o descumprimento dos prazos pode resultar em multas e afastamento temporário dos dirigentes, bem como em suspensão de autorização de funcionamento das organizações. Há, também, os prazos que são definidos em virtude da necessidade estratégica do negócio, em que determinados produtos ou serviços necessitam ser lançados em determinado período, para aproveitar campanhas, programas e ou subsídios disponibilizados pelo Governo.

Tabela 3.8: Matriz de probabilidade e impacto

Risco	**Probabilidade (P)**	**Impacto (I)**	**Severidade (P x I)**
Descumprimento dos prazos estabelecidos em cronograma	4	4	16
Incidentes e erros nas implantações dos *softwares*	3,8	4,2	15,96
Retrabalho	4,6	3,4	15,64

Artefatos incompletos, mal elaborados e desatualizados	4,4	3,4	14,96
Software complexo de difícil manutenção	3,8	3,4	12,92
Software em desacordo com o desejado pelo cliente	3,6	3,2	11,52
Divergências e desentendimentos entre membros da equipe	2,4	2,4	5,76

Fonte: Elaboração própria

Outro risco que merece destaque é "Incidentes e erros no ambiente de produção", em decorrência de erros e falhas no tratamento de dados inconsistentes, além do comportamentos inesperados dos programas, em situações nas quais não foram especificados tratamentos pelos gestores de negócio. Os erros em ambiente de produção normalmente causam indisponibilidades, que são situações problemáticas e pouco toleradas, uma vez que nenhum *software* deveria estar em funcionamento com erros que foram deixados passar em razão de testes e avaliação de qualidade de *software* mal elaborados.

O risco de "retrabalho"também é uma situação que causa bastante incômodo entre os desenvolvedores, que relataram que os prazos normalmente estabelecidos para a implementação, são insuficientes para a construção de um *software* de qualidade. A grande parte dos artefatos elaborados não contemplam todos os detalhes da solução desejada pelos gestores de negócio. Muitas vezes, em decorrência de constantes mudanças dos requisitos durante o processo de implementação, os programas se tornam complexos e difíceis de serem mantidos e, em muitos casos, é mais seguro recomeçar o *software* do zero, do que ficar fazendo novos ajustes que elevam a complexidade do *software*.

Na próxima seção será apresentada a aplicação da fase de avaliação dos riscos, que é uma das atividades previstas na norma ABNT NBR ISO 31000:2009 de gestão de riscos, buscando comparar os índices de severidade com os critérios de risco estabelecidos pelos desenvolvedores a fim de estabelecer uma significância e importância para cada nível de risco.

3.4 Avaliação dos Riscos

Nesta fase do processo de gestão de riscos, a técnica utilizada foi o Índice de Riscos, que, de acordo com a norma ABNT NBR ISO 31010:2009 [43], é considerada como fortemente aplicável para este tipo de atividade.

Esse é um método semi-quantitativo, uma estimativa derivada que utiliza uma abordagem de pontuação com escalas ordinais. Os índices de risco podem ser usados para avaliar uma série de riscos com o uso de critérios similares, de modo que possam ser comparados [43].

Nesta atividade participaram os mesmos 8 desenvolvedores do sistema MMO, que foram entrevistados sobre a contribuição dos artefatos para o processo de implementação.

Na Tabela 3.9, são apresentados os índices de que foram utilizados pelos desenvolvedores para comparar e classificar os riscos identificados no Processo de Implementação de *Software*. Para realização da atividade de avaliação dos riscos, foi estabelecido pelos desenvolvedores, escala de nível de risco baseado nos índices de severidade que foram atribuídos as vulnerabilidades durante a atividade de análise dos riscos apresentada na seção 3.3.

Tabela 3.9: Índices utilizados para comparação entre os riscos

Nível	**Severidade**
Muito Alto	12 a 25
Alto	6,4 a 11
Médio	3,6 a 6,3
Baixo	1,6 a 3,5
Muito Baixo	0 a 1,5

Fonte: Elaboração própria

A Tabela 3.10 apresenta o nível de significância dos riscos identificados. Quanto mais alto esse nível, maior os cuidados que a organização precisa tomar para mitigá-los ou eliminá-los. Observa-se que, do total de riscos identificados, mais da metade (aproximadamente 71%) foram classificados como de nível de risco “Muito Alto”. Logo, percebe-se que a maior parte das vulnerabilidades mapeadas no processo de implementação, são consideradas pelos desenvolvedores, como riscos de grande significância e impacto na na entrega de um *software* de qualidade para o cliente.

Tabela 3.10: Comparação dos níveis dos riscos identificados

Risco	**Nível do Risco**
Descumprimento dos prazos estabelecidos em cronograma	Muito Alto
Incidentes e erros nas implantações dos *softwares*	Muito Alto
Retrabalho	Muito Alto
Artefatos incompletos, mal elaborados e desatualizados	Muito Alto
Software complexo de difícil manutenção	Muito Alto
Software em desacordo com o desejado pelo cliente	Alto
Divergências e desentendimentos entre membros da equipe	Médio

Fonte: Elaboração própria

3.5 Resultados do diagnóstico do processo de implementação de *software*

A utilização do SIPOC mostrou-se bastante útil na definição do contexto do processo de implementação de *softwares*. A ferramenta permite visualizar os processos da organização em uma visão ampliada, permitindo elaborar SIPOC's para os diversos níveis de detalhamento, de acordo com o desejado. É uma ferramenta que auxilia na compreensão da organização em uma visão macro e também na granularidade de seus processos, permitindo uma melhor elaboração na gestão de riscos e, consequentemente, na melhoria dos processos.

A partir da identificação dos principais elementos do subprocesso de implementação, para a realização da atividade de identificação de riscos foram utilizadas as técnicas recomendadas pela ABNT NBR ISO 31010:2009: entrevistas e questionários.

Os questionários foram utilizados para a analisar os resultados das atividades do processo de implementação, cujo resultado foi avaliado pela ferramenta de Análise Hierárquica

de Multicritério (AHP), apontando quais eram as atividades de maior "importância", "dificuldade"e "risco"na entrega de um *software* de qualidade.

A utilização do AHP também se mostrou viável e tem atraído o interesse de muitos pesquisadores, principalmente devido às propriedades matemáticas do método, cuja simplicidade é caracterizada pela comparação, par a par, das alternativas segundo critérios específicos. A aplicação desse multicritério na análise das atividades do processo de desenvolvimento de *software*, permite aos gestores a tomada de decisão, não somente analisando dados qualitativos, mas também quantitativos, já que a ferramenta utiliza pesos e cálculos matemáticos para tornar o resultado mais tangível e com menor subjetividade.

Durante a realização dos questionários sobre as atividades, foi possível observar uma grande insatisfação dos entrevistados sobre a utilização dos artefatos no processo de implementação. Percebe-se que há necessidade de se dedicar tempo na elaboração de documentos de sistema que evidenciem seus comportamentos, funcionalidades, objetivos e características. A elaboração de documentação de sistema não deve ser apenas um procedimento obrigatório para fins de auditoria e certificação. Ele deve ser mantido, preservado e atualizado, para que seja uma fonte de informação segura e confiável aos desenvolvedores e gestores de negócio em suas necessidades e tomadas de decisões.

A análise e avaliação dos resultados obtidos a partir das entrevistas sobre os artefatos, foram realizadas por meio do manuseio e interpretação dos dados qualitativos, com análise das respostas e agrupamento, padronização das informações em categorias.

Na análise e interpretação dos dados obtidos com as entrevistas e questionários, foi possível verificar oportunidades de melhorias a partir da identificação dos principais riscos que influenciavam negativamente a qualidade dos processos e produtos da atividade de implementação.

A fim de estabelecer uma escala de priorização para tratamento dos riscos, foi utilizada a matriz de probabilidade e impacto para cálculo do critério de severidade. Esse cálculo, por sua vez, foi realizado a partir da combinação de pesos fornecidos pelos desenvolvedores para cada um dos riscos em relação a aspectos como frequência e consequência dos riscos para o processo de implementação.

Após finalizada a atividade de análise de riscos, foi utilizada a técnica de Índice de Risco para comparar o nível de criticidade dos riscos identificados entre si. Para realização dessa atividade, a escala de valores calculados para o critério “severidade” foi dividida em cinco faixas de valores menores, em que, de acordo com o resultado do critério de severidade, o risco poderia ser classificado em: muito alto, alto, médio, baixo e muito baixo.

A partir do resultado da atividade de avaliação, observou-se que, na opinião dos desenvolvedores, a maior parte dos riscos identificados eram de criticidade muita alta, com

comprometimento não somente da qualidade do processo de implementação, mas também dos demais processos de desenvolvimento de *software*, bem como na satisfação do cliente.

No próximo capítulo, será apresentada a aplicação de mineração de textos para extração de conhecimento em códigos-fonte *Cobol*, para permitir monitorar e gerenciar a qualidade dos programas e sub-rotinas produzidos no processo de implementação, a fim de mitigar problemas e riscos causados nas aplicações em razão de softwares complexos de difícil manutenção.

Capítulo 4

Classificação automática de códigos-fonte *Cobol* para monitoramento e controle do processo de implementação de *software*

No processo de gestão de riscos convém que os riscos sejam regularmente monitorados e analisados para verificar se as premissas dos risos permanecem as mesmas, bem como verificar se os processo de tratamentos dos riscos estão sendo eficientes e realizados. Neste contexto, buscou-se nesse capítulo, aplicar o processo de mineração de textos para extrair informações de códigos-fonte *Cobol* do sistema MMO, que pudessem auxiliar no monitoramento e controle da qualidade dos *softwares*, a fim de garantir programas e sub-rotinas mais coesos e seguros para as aplicações.

Para a realização deste objetivo utilizou como referência para seu desenvolvimento o modelo CRISP-DM 2.0. Este modelo é constituído de 6 fases que visam orientar a abordagem a ser utilizada em um estudo de mineração de dados ou textos. As fases são: Entender o Negócio, Entender os Dados, Preparação dos Dados, Modelagem, Avaliação e Desenvolvimento. A seguir será descrita cada uma destas fases no contexto deste trabalho.

4.1 CRISP-DM

A metodologia *Cross-Industry Standard Process for Data Mining* - CRISP-DM segue um modelo hierárquico que vai de um conjunto de processos gerais para um conjunto de tarefas

mais específicas, seguindo a hierarquia de fases, tarefas genéricas, tarefas especializadas e, por fim, instâncias do processos.

O principal objetivo do CRISP-DM é fornecer uma metodologia para conduzir projetos de mineração, sendo formada pelas seguintes fases [87]:

Entendimento do negócio - se refere ao entendimento dos objetivos do projeto e dos requisitos necessários para a sua realização na perspectiva do domínio do projeto. Nessa fase o projeto é mapeado como um problema de mineração de dados e é traçado um plano que guiará o projeto durante o seu desenvolvimento. São feitos cronogramas, listas de requisitos, seleção de ferramentas e técnicas de mineração de dados e etc.

Compreensão dos dados - se baseia na familiarização com os dados. Essa fase dá inicio a um processo de extração, verificação e análise dos dados coletados. Os dados devem ser verificados quanto à sua qualidade e se serão realmente úteis ao que é proposto pelo projeto. Nessa fase são feitos relatórios de descrição dos dados, da qualidade, da coleta inicial e etc.

Preparação dos dados - após a análise, uma vez que os dados foram avaliados e considerados satisfatórios pela fase de entendimento, aqui os dados coletados passam por um processo de preparação que transforma os dados brutos iniciais em dados prontos para serem aplicados às ferramentas de mineração de dados. Nessa fase são realizadas tarefas como seleção de atributos, limpeza, construção e formatação dos dados.

Modelagem - são selecionadas técnicas de modelagem para aplicação dos dados. Nessa fase também são escolhidos valores ótimos de parametrização para serem utilizados nessas técnicas de forma a obter os melhores resultados possíveis.

Avaliação - os modelos obtidos na fase de modelagem são avaliados para se ter a certeza de que esse modelo representa de forma suficiente os objetivos do negócio. Nessa fase os modelos são avaliados quanto a sua qualidade utilizando dados de teste diferentes dos dados utilizados para o treinamento e os resultados são verificados de acordo com os critérios adotados no entendimento do negócio.

Aplicação - A última fase é a de colocação em uso (Deployment). Nessa fase os resultados obtidos são utilizados por um analista para auxiliar em tomadas de decisões. Além disso, o modelo desenvolvido pode ser utilizado em outras bases de dados.

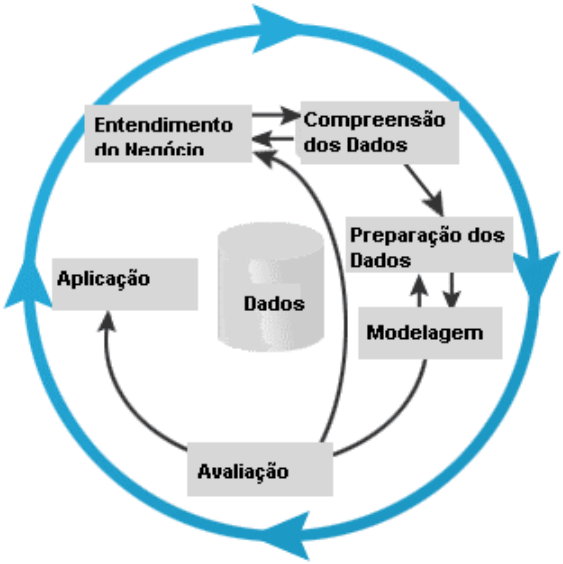

Figura 4.1: Fase da metodologia CRISP-DM

Fonte: VIEIRA [88]

4.2 Entender o negócio do processo de implementação de *software*

Nesta etapa, busca-se definir o problema e mostrar como a Mineração de Textos pode colaborar com a sua solução. A agilidade em nível organizacional, refere-se à eficiência com a qual a organização consegue responder à mudança. Aumentar esta agilidade é altamente atraente para as empresas, especialmente para as do setor privado. Ser apto a adaptar-se mais rapidamente às mudanças do mercado e da competição possui uma importância estratégica significativa. O departamento de TI é percebido, frequentemente como um gargalo, por requerer muito mais tempo e recursos para atender novos requisitos de negócio. Esta é uma das razões pelas quais os métodos de desenvolvimento ágil tem ganho popularidade, por proverem métodos de tratar requerimentos táticos e imediatos mais rapidamente.

Atualmente o Departamento de TI da instituição financeira, possui aproximadamente 300.000 programas escritos na linguagem de programação *Cobol* distribuídos em 934 aplicações que atendem a 27 diretorias (departamentos das áreas de negócio). Com esse volume de código-fonte é de fundamental importância que os programas sejam de fácil manutenção, e adaptáveis a novas regras e funcionalidades solicitadas pela área de negócio e determinadas pelos órgãos reguladores do mercado financeiro. Durante a fase de entendimento do negócio, percebeu-se que para o desenvolvimento e manutenção de programas com maior segurança e rapidez, o departamento de tecnologia da instituição se baseia em

normativos e padrões internacionais, tais como: NBR ISO/IEC 12207 15504, MPS-BR e outros, para o desenvolvimento de suas aplicações.

É comum que os *software*s com o passar dos tempos se tornem complexos em manter, entender, adaptar e difíceis de reusar e evoluir. Isso ocorre em razão do aumento do tamanho e complexidade dos *software*s e pela rápida evolução das tecnologias de processamento de dados. Quando existem nos códigos muitos elementos interligados e dependentes, e várias tarefas específicas sendo realizada em um mesmo código-fonte, existe uma grande possibilidade de se ter problemas com a sua manutenibilidade [89]. Uma das principais estratégias para solucionar esta questão é a aplicação dos princípios de baixo acoplamento e alta coesão.

O Princípio do Desacoplamento se baseia na redução do acoplamento. O acoplamento é uma medida da dependência que um programa ou sub-rotina possui de outros, ou ainda, o quanto outros componentes de *software* dependem e sabem detalhes da implementação de um outro programa ou sub-rotina.

O Princípio da Coesão se baseia no aumento da coesão dos componentes de *software*. A coesão é uma medida das responsabilidades que um programa ou sub-rotina possui e se baseia no conceito de 'Responsabilidade Única'. Levando este conceito para a alta plataforma (Mainframe), o programa ou sub-rotina tem coesão fraca quando muitas funções que, eventualmente tratam de assuntos diferentes, são executadas dentro dele.

Neste estudo será analisado os códigos-fonte *Cobol* de um sistema de dados de operações do mercado monetário (MMO) responsável pelo funcionamento e gestão de produtos e serviços relacionados a área de finanças e investimentos. O sistema MMO possui atualmente um inventário de 182 programas escritos na linguagem de programação cobol. Atualmente em busca de uma maior flexibilidade na escolha de tecnologias para implementar as aplicações, foi adotado pelo Departamento de Tecnologia da instituição, um padrão de desenvolvimento baseado em 3 camadas: apresentação, negócio e persistência.

A camada de apresentação é a responsável pela entrada e saída dos dados no sistema. A integração do usuário nesta camada é feita por meio de um web browser e por emuladores 3270 que são programas de computador que simulam as funções de um terminal de grande porte (mainframe) em um microcomputador. A camada de negócio é a responsável por implementar as regras de negócio do sistema, todas as entradas vindas da camada de apresentação passam pela camada de negócio que utiliza a camada de persistência quando necessário. A camada de persistência tem por finalidade o acesso e manipulação dos dados diretamente na base de dados.

Entre as linguagens de programação adotadas pelo sistema MMO, encontram-se o Natural e *Java* que devem ser utilizados para implementar os programas da camada de apresentação, enquanto que a linguagem de programação *Cobol* deve ser utilizada para

implementar os programas da camada de negócio e de acesso a Base de Dados, que é denominado como programas de persistência.

Os programas para serem implantados e liberados para o ambiente de produção, conforme padrão adotado pelo organização, passam pelos subprocessos de requisitos, análise e projeto, implementação, teste e implantação. Na etapa de implementação, o desenvolvedor realiza a codificação, testes unitários e de integração no *software* para identificar erros de lógica e inconsistências no processamento dos dados.

Após realizados os testes de integração os programas são disponibilizados para homologação e é nesta fase que são realizados os testes de sistema e de aceitação pelo Gestor de Negócio. Nesta etapa de testes é possível verificar se os requisitos funcionais e não funcionais foram realmente atendidos e se a aplicação está funcionando conforme o esperado.

Todos os programas utilizados neste estudo, se referem a programas em suas versões finais e que encontravam-se disponíveis para manutenção e consulta no ambiente de produção. Note que o mesmo programa, pode possuir mais de uma versão de código-fonte, em razão de que exitem bibliotecas para armazenamento exclusivos para cada ambiente: desenvolvimento, homologação e produção.

4.3 Entendendo sobre os códigos-fonte *Cobol* do processo de implementação de *software*

A linguagem *Cobol* foi criada em 1959 numa iniciativa do Pentágono (USA) que tinha por objetivo estabelecer uma linguagem comum de programação para ser utilizada na solução de problemas comerciais. *Cobol* é uma linguagem de 3ª geração, e é padronizada pelo *American National Standards Institute* (ANSI) desde 1968. O *American National Standards Institute* é a instituição responsável pela analise das necessidades de novas implementações para a linguagem *Cobol*. Periodicamente se reúnem com grupos de usuário.s Todo programa *cobol* consiste obrigatoriamente, em 4 (quatro) divisões, onde cada uma tem um papel específico dentro da linguagem. Estas divisões auxiliam na simplicidade, eficiência e leitura do programa. Elas são escritas em inglês, para diminuir o esforço e facilitar a compreensão do programa por pessoas alheias ao processamento de dados. As quatro divisões são: IDENTIFICATION DIVISION, ENVIRONMENT DIVISION, DATA DIVISION E PROCEDURE DIVISION.

- IDENTIFICATION DIVISION - Divisão de identificação, nela são identificados, o autor, programa-fonte e módulo-objeto.
- ENVIRONMENT DIVISION - Especifica o equipamento a ser utilizado bem como os periféricos envolvidos no funcionamento do programa.

- DATA DIVISION - Descreve os dados que o programa aceitará como entrada e os que serão produzidos como saída, bem como todos os dados intermediários utilizados no programa.
- PROCEDURE DIVISION - Descreve todas as ações a serem tomadas pelo programa, é nesta divisão que se desenvolve a lógica do programa.

A PROCEDURE DIVISION possui uma estrutura hierárquica e consiste de seções, parágrafos, sentenças e comandos. A seção é opcional, porém deve haver pelo menos um parágrafo, sentença ou comando na PROCEDURE DIVISION. Os nomes de parágrafo e seção é escolhido pelo desenvolvedor.

As divisões devem sempre aparecer nesta ordem, dentro de um programa, e podem ser divididas em seções e estas em parágrafos. Todas as outras instruções do programa são consideradas declarações *Cobol*.

Embora existam 4 divisões dentro do programa *Cobol*, apenas o conteúdo existente na PROCEDURE DIVISION foram utilizados no desenvolvimento desse estudo, uma vez que o restante das divisões não possuem termos significativos para o tipo de classificação pretendida e também pelo fato de que a complexidade do módulo está em seu algoritmo e não na definição de variáveis, definição de periféricos nome e objetivo do programa.

Os programas escritos na linguagem de programação *Cobol*, não possuem limitação de tamanho, ou seja, podemos ter programas extensos e curtos, cuja a complexidade não possui relação direta com a quantidade de linhas do programas, mas sim com o nível de conhecimento e experiência do do desenvolvedor. Não há nenhuma orientação ou norma na instituição que estabeleça um limite de quantidades de linhas para um programa. Outro fator observado, foi em relação a declaração e definição de variáveis, onde apesar de existir um glossário de termos apropriados para o uso, a falta de monitoramento e controle permite que os desenvolvedores realizem abreviações de termos de forma livre e arbitrária.

Diante desse contexto, em busca de excluir termos que causam ruído na identificação de padrões comuns entre os programas e reduzir o volume de texto para classificação, foram selecionados para análise nos códigos-fonte *Cobol*, somente o conteúdo localizado na parte da PROCEDURE DIVISION.

No período de 12/2014 a 05/2015, acompanhou-se as falhas ocorridas no ambiente de produção para os programas e sub-rotinas do sistema MMO. Durante esse período foram contabilizadas um total de 112 falhas, sendo que 17 eram em programas do tipo base de dados, 30 em programas do tipo negócio, e 65 em programas mistos que tratavam tanto de regras de negócio, quanto a acesso a banco de dados e que realizavam muitas tarefas específicas em um mesmo código-fonte. Na Figura 4.2, temos representado o percentual de

falhas ocorridos em relação a cada tipo de programa: 58,04% em programas misto, 26,79% em programas de negócio e 15,18% em programas de base de dados. Nos programas mistos analisados foi possível verificar que além das características de acesso a banco de dados e regras de negócio, também observou-se várias tarefas específicas sendo realizada em um mesmo código-fonte.

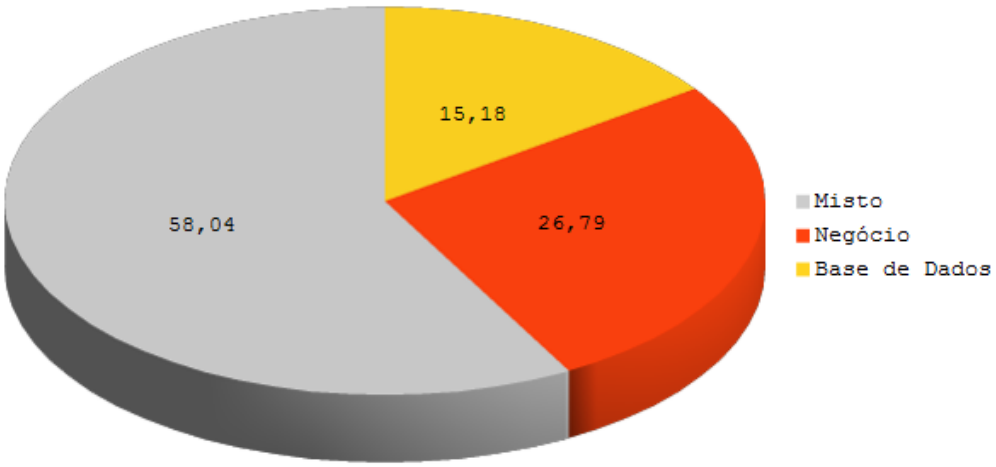

Figura 4.2: Ocorrências de falhas por tipos de programas do sistema MMO

Fonte: Elaboração Própria.

Com base neste resultado, percebe-se que os programas mistos eram responsáveis por mais de 50% das falhas e indisponibilidades do sistema MMO. A fim de reduzir este percentual, buscou-se por métodos que pudessem avaliar os códigos-fonte, e a partir da análise de seus termos, classificá-los para auxiliar na identificação de programas complexos. Portanto para classificação dos códigos-fonte quanto a complexidade foram estabelecidas as seguintes classes: Negócio, Base de Dados e Misto.

Neste contexto, para classificação dos programas nestas três classes, adotou-se que os códigos-fonte com parágrafos e seções sem acesso direto a base de dados (tabelas), seriam classificadas como Negócio, os códigos-fonte com acesso a tabelas, sem regras de negócio sem leitura e gravação de arquivos seriam classificados como Base de Dados.

Programas com características de Negócio e Base de Dados cumulativamente seriam classificados como Misto. Portanto, neste trabalho, o Misto é a classe a que se pretende identificar, pois entende-se que é o tipo de programa com maior complexidade, que exigem maior tempo de análise e custo de manutenção.

4.4 Preparação dos códigos-fonte para mineração de textos

O processo de preparação dos dados se inicia quando o negócio ja está bem definido. Para a realização deste estudo foram utilizados para o processo de aprendizado de máquina todos os programas cobol localizados na biblioteca do ambiente de produção, um total de total de 182 códigos fonte. Todos os códigos-fonte localizados na biblioteca de produção, foram salvos no formato texto, em pasta local, para posterior limpeza e organização dos dados.

A ferramenta escolhida para mineração de textos e aprendizado de máquina, foi o RStudio e para a limpeza e tratamento dos códigos fonte foi utilizado o pacote TM. Este pacote torna possível processar, organizar, transformar, e analisar os dados textuais com maior eficiência. Previamente ao processo de aprendizado de máquina, foi realizado um processo manual para juntar as PROCEDURES DIVISIONS dos 182 programas em um mesmo documento texto, separados por ";", para indicar o início e fim de um programa e outro.

A fim de aumentar o desempenho dos modelos de classificação utilizados no processo de aprendizado de máquina, foram eliminados os termos de baixa frequência, transformação das palavras maiúsculas em minúsculas, remoção de pontuação, acentuação, números e espaços em branco. Estas técnicas são comuns,e bastante utilizadas no processo de mineração de texto.

Na primeira transformação dos códigos-fonte em Matriz Documento Termo (DTM), foi gerado um resultado com 6589 termos. Diante dessa quantidade de termos, e considerando o consumo de memória para automatização do processo de categorização dos textos, foi realizada entrevista com os desenvolvedores do sistema MMO para verificar quais os termos e padrões existentes em um código-fonte *Cobol*, poderiam ser relevantes significativos na categorização dos códigos-fonte em negócio, misto e base de dados.

O perfil de desenvolvedores selecionados para a participar da entrevista, foi o mesmo adotado para participação dos questionários sobre as atividades do processo de implementação, portanto estavam aptos a serem entrevistados cinco desenvolvedores, de um total de doze existente no sistema MMO. O roteiro utilizado para a entrevista encontra-se ao fim desse trabalho no *Apêndice D*.

Como resultado da entrevista, foram apontados os seguintes termos:

- read - coloca disponível para processamento um registro lógico de um arquivo. A presença desse comando nos códigos-fonte indica a leitura de um arquivo com dados para processamento, aplicações de regras e sumarizações para a obtenção de algum resultado.

- write - libera um registro lógico para arquivo de saída. Este comando trata-se da gravação de arquivo com dados processados para utilização em processos seguintes da própria aplicação, ou para envio a outros sistemas e entidades e externas.
- sql - trata-se de comando utilizado para indicar a execução de alguma instrução para acesso direto a tabelas de banco de dados, seja para realizar consulta, exclusão, alteração ou inclusão de dados.
- display - é utilizado para exibir uma mensagem e também conteúdos de veriáveis durante um processamento. Este tipo de comando possui restrições para o uso, já que consome bastante recurso, e portanto seu uso não seria indicado para programas e sub-rotinas de execução on-line.
- stop - interrompe definitivamente a execução do programa, retornando o controle para o sistema operacional. Serve para indicar que o programa já fez tudo o que deveria fazer e que portanto já pode ser finalizado.
- goback - termina a execução do programa retornando o controle para o programa que o chamou. Serve para indicar que a sub-rotina já finalizou seu processamento e que o programa principal pode continuar com o processamento.

Pelas características e padrões de codificação adotados no sistema MMO, os desenvolvedores apontaram read, write, display, stop run, sql e goback como os termos para categorização dos programas e sub-rotinas, onde o que determinaria a classe dos programas e sub-rotinas, seriam as possíveis combinações entre esses termos, bem como a ausência deles. Ex: programas com sql e goback pertencem a classe base dados, programas com sql e write ou read pertencem a classe misto e assim por diante.

4.5 Modelagem dos termos do código-fonte *Cobol* para classificação

Para a modelagem dos termos foi construída uma Matrix Document-Term (DTM) utilizando Term Frequency - Inverse Document Frequency (TF-IDF) como parâmetro. DTM é uma matriz onde as linhas representam os documentos, que são cada um dos códigos-fonte e as colunas representam as palavras da linguagem de programação cobol (Termos). Além dos termos e documentos, foi necessário incluir uma nova coluna com a informação da classe a que pertence cada um dos códigos-fonte.

Os dados da matriz documento-termo foram divididos em três conjuntos: treinamento, validação e teste. O treinamento recebeu um total de 109 documentos representado um

total de 60% por cento da base, a validação e o teste ficaram com 36 e 37 documentos respectivamente, representando cada um 20%.

O conjunto de treinamento é utilizado para que o modelo de classificação aprenda os padrões e características dos documentos. O conjunto de validação é utilizado para ajustar os parâmetros do aprendizado de máquina e para a escolha do melhor modelo. O conjunto de teste é utilizado para avaliar o desempenho do modelo de classificação que obteve melhor resultado com a base de validação, a fim de verificar se ele pode ser generalizado para os demais códigos-fonte *Cobol* desconhecidos.

Os conjuntos de códigos-fontes foram treinados e validados para gerar os modelos de classificação a partir da utilização de três algoritmos: *Naive Bayes*, *Random Forest* e *Suporte Vetor Machine* (SVM).

A Floresta Randômica é um algoritmo de classificação que utiliza um conjunto de árvores de decisão construídas simultaneamente [90], no RStudio foi utilizado o pacote denominado randomForest.

O *Naive Bayes* é um algoritmo de aprendizado de máquina de classificação probabilística em que os grafos representam a probabilidade de um conjunto de variáveis independentes. O pacote de *Naive Bayes* utilizado no RStudio foi o KlaR.

SVM [91] é um algoritmo utilizado para classificação e análise de regressão.Este algoritmo foi escolhido para este estudo, em virtude dele ter sido utilizado em trabalhos correlatos de mineração de textos e ter obtido bons resultados.O pacote usado para o RStudio gerar o modelo com *SVM* foi E1017.

4.6 Avaliação e desenvolvimento do processo de classificação dos códigos-fonte

Esta seção é responsável por apresentar a análise e avaliação dos algoritmos e atributos de classificação selecionados. O processo de experimentação de um algoritmo de classificação, geralmente mede a sua eficácia, que consiste em avaliar a sua capacidade de tomar decisões de categorização corretamente.

Quando se trabalha com classificadores, é necessário se trabalhar também com medidas que possam avaliar se a classificação realizada foi eficaz, ou seja, é necessário saber qual foi a taxa de acerto e erro do classificador. Entre as medidas de desempenho existentes, serão citadas aqui algumas comumente utilizadas na literatura e também utilizadas nessa pesquisa.

Neste estudo os modelos de classificação, foram avaliados pelos seguintes critérios de avaliação: Sensibilidade, Especificidade, *Kappa*, *F-Measure*, Matriz de Confusão e Acurácia. Matriz de Confusão é geralmente usado para avaliar o desempenho de um algoritmo.

Esta matriz mostra o número de situações em que as classes foram corretamente e incorretamente classificadas.

Combinando algumas técnicas é possível alcançar um resultado mais consistente que valide os resultados apresentados pelos modelos de classificação. Portanto foram avaliados os algoritmos de classificação *SVM*, *Random Forest* e *Naive Bayes* usando a validação em conjunto das métricas descritas.

4.7 Resultados da classificação automática dos códigos-fonte *Cobol*

A Tabela 4.1 mostra os Resultados de *Kappa*, *F-Measure* e Acurácia para cada classificador.

O melhor resultado foi obtido com o classificador *Naive Bayes* com um *Kappa* de 0,91 e *F-Measure* de 0,97. Os outros dois modelos *Random Forest* e *SVM* tiveram desempenho bastantes semelhante com *Kappa* de 0,52 e 0,5286 respectivamente. Entretanto, no cálculo do *F-Measure* para estes dois classificadores a diferença se tornou mais significante, onde o *Random Forest* apresentou um desempenho melhor, com um valor de 0,92 e *SVM* com um valor de 0,88.

Tabela 4.1: Métricas utilizadas para avaliação de desempenho dos algoritmos

Algoritmo	*Kappa*	*F-Measure*	*Acurácia*
Random Forest	0,52	0,92086	0,6944
SVM	0,5286	0,88631	0,6944
Naive Bayes	0,9161	0,97619	0,9444

Fonte: Elaboração própria

As Tabelas 4.2 a 4.4 mostram a Matriz de Confusão de cada algoritmo usando o arquivo de validação. Pode-se observar que o *Naive Bayes* apresentou o melhor resultado e o *SVM* e o *Random Forest* tiveram um pior desempenho. Veja que na Matriz Confusão do *SVM* e do *Random Forest* a célula da diagonal principal para a classe Negócio encontra-se com zero, ou seja, ambos os algoritmos não tiveram desempenho nenhum para esta classe, ou seja, não conseguiram identificar nenhum padrão para identificação dessa classe. No entanto o *Naive Bayes* para essa mesma classe teve um acerto de quase 100%.

Tabela 4.2: Matriz de Confusão para *SVM*

SVM	Base de Dados	Misto	Negócio
Base de Dados	11	0	5
Misto	0	14	6
Negócio	0	0	0

Fonte: Elaboração própria

Tabela 4.3: Matriz de Confusão para *Random Forest*

Random Forest	Base de Dados	Misto	Negócio
Base de Dados	11	0	0
Misto	0	14	11
Negócio	0	0	0

Fonte: Elaboração própria

Tabela 4.4: Matriz de Confusão para *Naive Bayes*

Naive Bayes	Base de Dados	Misto	Negócio
Base de Dados	11	0	0
Misto	0	13	01
Negócio	0	1	10

Fonte: Elaboração própria

Portanto, com a realização desse estudo, foi alcançado o que foi definido no objetivo específico 2, que era propor um modelo de classificação automática para a identificação de complexidade em código-fonte. É importante observar que, de acordo com o levantamento realizado na quantidade de falhas ocorridas no sistema MMO no período de 12/2014 a 05/2015, aproximadamente 58,04% das falhas ocorreram em programas do tipo misto que possuem regras de negócio e acesso à base de dados em um mesmo código-fonte. Da mesma

forma, é preciso notar que os programas do tipo Misto são um tipo de implementação em que a instituição já percebeu seu impacto e custos no desenvolvimento de software, haja vista existirem normas que orientem os desenvolvedores a implementarem programas em camadas, seguindo os princípios de coesão e baixo acoplamento.

Não raro os analistas têm grandes dificuldades em realizar manutenção em aplicações críticas, dada a sua complexidade, uma vez que os programas, em sua grande maioria, não realizam apenas uma tarefa, e também pelo fato de que as intervenções nos programas podem resultar em mudanças de regras e comportamentos não desejados nas aplicações.

Nesse cenário, programas mais complexos têm maior probabilidade de deixar as aplicações de *software* inoperantes, bem como de processarem dados inconsistentes.

A fim de auxiliar a área de TI a controlar e monitorar a qualidade de seus programas e sub-rotinas e de identificar programas complexos que realizem várias funções e acesso a banco de dados em um mesmo código fonte, é que se utilizou a mineração de textos com a aplicação de algoritmos supervisionados para identificação do melhor modelo de classificação.

Esse modelo permitirá mapear os módulos mais críticos existentes dentro das aplicações e, assim que os gerentes de TI acharem necessário e decidirem pela melhoria do sistema, esses programas poderão ser reescritos, buscando torná-los mais específicos, mais claros e seguros para o uso nas aplicações.

Para os programas novos, o classificador poderá ser utilizado como um critério de aceite dos códigos-fonte. Os desenvolvedores, após concluírem a codificação dos programas, irão submetê-los ao classificador automático, na qual, caso ele seja classificado como misto, caberá ao gerente decidir em aceitá-lo ou solicitar ao desenvolvedor que o reescreva seguindo os princípios de alta coesão e baixo acoplamento.

O uso do classificador automático de códigos-fonte *Cobol*, também poderá ser usado como ferramenta de controle e monitoramento capaz de apoiar as equipes de desenvolvimento a se organizarem para reescreverem esses códigos seguindo os princípios de alta coesão e baixo acoplamento em futuras manutenções de saneamento e melhoria da qualidade dos componentes de *software* da aplicação.

Além de auxiliar na melhoria da qualidade do processo de implementação de *software*, o classificador automático representará uma economia aos processos de TI, uma vez que, com o tempo, a quantidade de programas complexos diminuirá gradativamente e consequentemente serão necessárias menos horas para realizar as atividades de análise e manutenção corretivas desses tipos de programas e sub-rotinas nas aplicações.

Capítulo 5

Proposta de melhorias para tratamento de riscos no processo de implementação de *software*

Neste capítulo, para dar continuidade ao processo de gestão de riscos iniciado nos capítulos 3 e 4, foram selecionados para tratamento de riscos, as vulnerabilidades do processo de implementação classificadas como de risco "Muito Alto", cuja as causas estivessem relacionadas ao principal e único artefato do processo de implementação: o código-fonte.

Também influenciou no critério de seleção, o tempo de resposta ao tratamento do risco. Inicialmente, para garantir o apoio dos desenvolvedores na adoção de novas práticas para o tratamento de riscos, mostrou-se de essencial importância que os benefícios e vantagens fossem logo percebidos pelos desenvolvedores, a fim de que se sentissem motivados a contribuir com a evolução e melhoria contínua do processo de implementação.

Nesse contexto, foram selecionados os riscos "*software* complexo de difícil manutenção"e "artefatos incompletos, mal elaborados e desatualizados", em razão de que estes riscos teriam relação direta com detalhamento e especificação de *software* ineficiente durante a fase de Análise e Projeto de *Software*, em que as regras e instruções para atendimento aos requisitos não foram totalmente detalhados.

Tais riscos comprometem diretamente o processo de implementação de *software*, em razão de que códigos-fonte complexos demandam maior tempo de manutenção e consequentemente implicam em maior custo para organizações. Artefatos desatualizados e incompletos são ameaça para a estabilidade das aplicações, haja vista que não são fonte segura de consulta e tomada de decisão, podendo expor as instituições a riscos de imagens e prejuízos financeiros ao tratar informações e dados inconsistentes.

Portanto, neste capítulo pretende-se propor um processo de especificação de programas e sub-rotinas, que seja facilmente compreendido pelos desenvolvedores e que proporcione

maior agilidade, segurança, qualidade e satisfação na entrega dos *softwares* aos clientes.

5.1 Qualidade de artefatos do processo de implementação de *software*

Durante o processo de desenvolvimento de *software*, são produzidos diversos artefatos, conforme apresentado no capítulo 3. Eles são essenciais para o gerenciamento e controle de qualidade, tanto do produto quanto do processo usado para o seu desenvolvimento. Uma documentação de qualidade propicia uma maior organização e agilidade no atendimento de novas demandas e nas manutenções, contribuindo para que sistemas se tornem mais estáveis e o cliente mais satisfeito. Além disso, reduz o impacto da saída ou substituição de membros da equipe e o tempo de desenvolvimento de novos projetos, bem como auxilia na prevenção de falhas e eliminação de erros.

No processo de implementação de *software*, documentos de entrada (insumos), como por exemplo a especificação de componentes, normalmente são pouco utilizados, pois os desenvolvedores não enxergam grandes benefícios em sua utilização. Além disso, culturalmente se está acostumado a produzir apenas documentos obrigatórios, por acreditar ser perda de tempo especificar, sendo que escrevê-lo direto na linguagem de programação desejada reduziria parte do tempo dedicado ao subprocesso de análise e projeto, podendo significar antecipação no prazo de entrega final do *software*. Há, portanto, necessidade não apenas de apresentar uma nova ferramenta que substitua a especificação dos programas e sub-rotinas em documento texto, mas que realmente agregue valor, motive e incentive os desenvolvedores a documentarem o comportamento dos sistemas desde o seu nível operacional, até o seu nível de negócio.

Existe, portanto, a necessidade de se definir um processo para controlar a documentação de uma organização, incluindo atividades de planejamento, análise, aprovação ou reprovação, identificação de alterações, situação da revisão atual, disponibilidade das versões pertinentes de documentos aplicáveis, dentre outras. Para a garantia de que o processo esteja sendo executado corretamente, é necessário o acompanhamento e controle de garantia da qualidade do *software*, o que deve ser feito por uma equipe responsável pela garantia da qualidade dos artefatos.

Estudos demonstram que o esforço gasto por organizações de *software* com retrabalho pode variar em média entre 40% e 50% do esforço total de desenvolvimento de um projeto ou funcionalidade [92], o que tende a aumentar na medida em que o projeto progride. Uma das razões para que isso aconteça é que os esforços para corrigir os defeitos são intensificados mais nas atividades finais do projeto. Nessas situações muitas vezes torna-

se necessário voltar à fase de requisitos para análise e esclarecimentos, o que resulta em grandes custos e insatisfação por parte do cliente.

A qualidade do *software* é a conformidade de requisitos funcionais e não funcionais explicitamente declarados, aliados a padrões de desenvolvimento que tenham sido claramente documentados [8]. Entre as atividades que podem ser utilizadas para se verificar essa qualidade encontram-se revisões, testes, padrões e procedimentos formais (artefatos), controle de mudanças, métricas de *software* e procedimentos para coleção e disseminação de informações (documentação de sistema) [92].

Documentar sistemas é diferente de documentar um projeto, pois este último está limitado a gerar documentos que evidenciem a realização dos processos de requisitos, análise, implementação, testes e implantação de um escopo bem definido e delimitado; enquanto o primeiro é um processo bem mais complexo, que deve acompanhar as evoluções do sistema, a fim de se manter uma fonte atualizada e confiável de informação, para o atendimento de forma ágil e segura dos objetivos estratégicos da organização.

Dentre as atividades de controle e garantia de qualidade do *software*, estão as de verificação, validação e testes. O objetivo é assegurar-se que ele esteja sendo construído de forma correta e, para isso, deve-se verificar se os artefatos produzidos atendem aos requisitos, e se os padrões organizacionais de desenvolvimento dos processos e produtos foram consistentemente aplicados.

5.1.1 Padrões Organizacionais

Para verificação da qualidade dos processos e produtos desenvolvidos, deve-se estabelecer padrões organizacionais, como a ISO/IEC 12207, ISO/IEC 15504, MPS-BR e outros. Os padrões de produtos aplicam-se a artefatos produzidos ao longo do processo de *software*. Podem ser modelos, roteiros, guias e normas, dependendo do artefato a que se aplicam. Um modelo de documento define a estrutura (seções, subseções, informações de cabeçalho e outros), estilo (tamanho, tipos de fonte) e o conteúdo esperado para documentos de um tipo específico, sendo necessário que sejam padronizados, o que facilita a leitura e compreensão, dada a familiaridade dos profissionais envolvidos com o seu formato.

Padrões organizacionais, sejam de processo ou de produtos, são muito importantes, pois fornecem um meio de capturar as melhores práticas de uma organização e divulgá-las entre os demais profissionais e setores da empresa em busca de processos mais ágeis e seguros.

Além disso, em se tratando de padrões organizacionais, todos os desenvolvedores tendem a estar familiarizados com eles, facilitando a manutenção dos artefatos e a substituição de pessoas responsáveis pela condução dos sistemas sem grandes impactos. Para aqueles artefatos considerados mais importantes no desenvolvimento dos sistemas, é re-

comendado que haja um padrão de documentação associado, baseado em padrões gerais propostos por instituições nacionais ou internacionais voltadas para a área de qualidade de *software*, como por exemplo a ISO [1].

5.2 Gerenciamento de Processos de Negócio (BPM) aplicado ao processo de implementação de *software*

Nesse estudo, para realização da atividade de tratamento dos riscos do processo de gestão de riscos, foi utilizado o BPM, um modelo bastante conhecido pela sua eficiência na identificação e proposta de melhoria de processos deficitários por meio do mapeamento de processos *AS IS* e *TO BE*. Recentemente, vários estudos na linha de gestão de riscos têm utilizado o BPM para mapear os processos [93], [64], assim como identificar fragilidades e desvios que comprometem a eficiência das organizações e o alcance de seus objetivos.

A aplicação do BPM quando utilizada na implementação de *software*, deve por objetivo mostrar os seus benefícios e vantagens, no mapeamento das funcionalidades do sistema MMO, com representação gráfica dos processos, atividades e tarefas, por meio de diagramas da notação BPMN, abrangendo informações desde um nível macro até um nível detalhado de verificação do comportamento de programas e sub-rotinas utilizados na aplicação. Além do mapeamento, outro benefício pode ser observado na utilização do BPM quando aliado a outros *softwares* de armazenamento de documentos. A combinação de gerenciamento de processos aliado a outros *softwares* de armazenamento de artefatos mostrou-se intuitivo, ágil e prático na disponibilização centralizada das versões mais recentes de todos os artefatos produzidos durante o desenvolvimento, versionamento ou manutenção do *software*.

A localização dos documentos em somente um endereço de consulta facilita e estimula sua utilização por um maior número de desenvolvedores, em razão da facilidade de localização, bem como da segurança de não estarem sendo utilizadas informações de uma versão desatualizada do documento.

Atualmente no sistema MMO, a grande maioria dos artefatos possuem locais de armazenamento distintos, o que acaba provocando desinteresse entre os desenvolvedores em utilizá-los em razão do tempo gasto para encontrá-los e acessá-los, por encontrar-se em *softwares* disponibilizados onde demandam maior tempo para localização e com outros tipos de restrições e dificuldades, como necessidade de autorizações de acesso e limites de usuários. Conforme apresentado na Tabela 5.1, os documentos podem estar armazenados em seis locais diferentes, conforme o tipo e finalidade da informação.

Tabela 5.1: Localização dos artefatos do processo de desenvolvimento de *software*

Local de Armazenamento	**Artefatos**
Site de Arquitetura de Dados	• MER • MDC
Clarity	• RNS
Rational Quality Manager (RQM)	• PLT • RET • RRT • RFT
Particionado Mainframe	• CDF
Concurrent Version System (CVS)	• PLI • ESI • EST • ESC • DTE • DGT • DPS • DGA • DES • PID • DPB • DGP
Rational Requirements Composer (RRC)	• DRI

Fonte: Elaboração própria

Diante do exposto, percebe-se que os desenvolvedores identificam poucas vantagens e benefícios na utilização dos artefatos, já que, com tantos locais de armazenamento, aliado à não obrigatoriedade de confecção de alguns deles, não há garantia de se encontrar o que precisam e desejam, já que é tão difícil localizá-los e consultá-los.

Neste contexto o BPM é apresentado como um instrumento capaz de representar graficamente os artefatos existentes para uma determinada funcionalidade, bem como referenciar um atalho para o local onde os artefatos estão de fato armazenados. Na Figura 5.1 pode-se ver representados, em um mesmo local, os artefatos que foram elaborados para a implementação de um programa do sistema MMO. Eles continuam armazenados em locais diferentes, porém, com o uso do *software* de gerenciamento de processos é possível encontrá-los e consultá-los mais rapidamente a partir de uma consulta no processo, que pode ficar disponível em um site corporativo para uso interno aos funcionários da instituição.

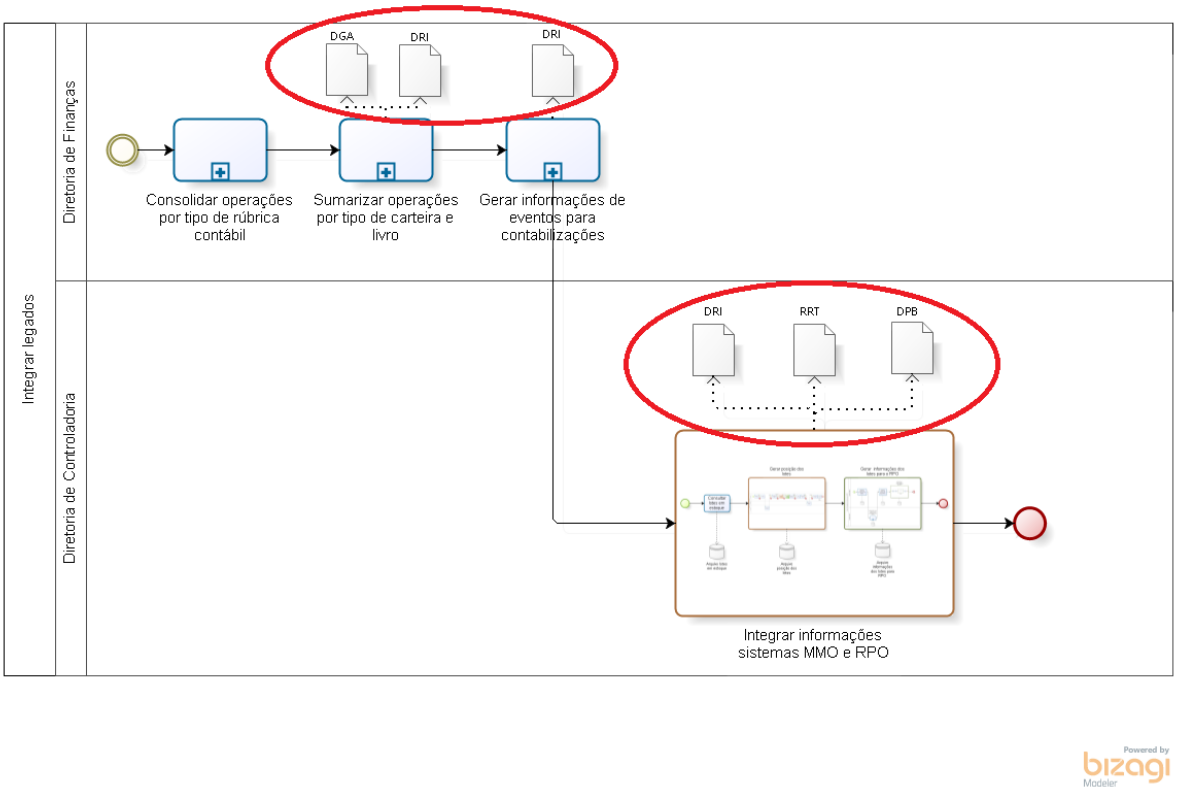

Figura 5.1: Consulta aos artefatos por meio da ferramenta de gerenciamento dos processos

Fonte: Elaboração própria

Diante disso, a modelagem de processos de negócio mostra-se bastante importante para a gestão de riscos de *software*, pois permite levantar, modelar, documentar e monitorar as funcionalidades dos sistemas, com o objetivo de agir proativamente a situações de ameaças e garantir a continuidade dos negócios, sem expor a organização a riscos de imagem e prejuízos financeiros.

Conforme apresentado no capítulo 3, o artefato de especificação de componente (ESC) é pouco utilizado pelos desenvolvedores do sistema MMO. Aproximadamente 43% dos entrevistados não o utilizam, por tratar-se de documento não obrigatório e pelo fato de

não ser garantida a sua atualização pelos demais desenvolvedores a cada modificação do sistema, que, ao longo do tempo, passam por sucessivas fases de crescimento, evolução e declínio [94], tornando-se, muitas vezes, complexos. Nessa situação, torna-se de grande importância a existência e disponibilidade de uma documentação de sistemas que torne as modificações mais precisas e seguras, sem grandes impactos ao restante das funcionalidades da aplicação.

Portanto, neste capítulo, buscou-se apresentar uma proposta de documentação de sistemas, por meio do uso do modelo de gerenciamento de processos de negócios (BPM), para mapeamento e documentação das funcionalidades por meio de diagramas, e a disponibilização dessas informações como fonte de consulta confiável e atualizada de funcionamento do sistema.

5.3 Contexto dos processos de negócios do sistema MMO

O sistema utilizado para o estudo de caso, conforme mencionado anteriormente, trata-se de um sistema de dados de uma instituição financeira, responsável pelo controle e gerenciamento de operações do mercado monetário, desenvolvido em plataforma baixa, com linguagem de programação *Visual Basic*, da Microsoft.

Atualmente, no sistema MMO, existem demandas e projetos sendo conduzidos em paralelo, envolvendo plataformas e linguagens de programação diferentes. Esse é um contexto de elevado risco, que deve ser gerenciado pelos analistas responsáveis pela aplicação, porque modificações em determinadas funções podem impactar processos que já estão funcionamento, tornando-as indisponíveis.

A partir dos levantamentos realizados com os desenvolvedores, foram identificadas, para o sistema MMO, um total de nove macro funcionalidades, que devem:

1. estabelecer metas para os operadores;
2. cadastrar boletas de compra e venda de títulos;
3. particionar os lotes;
4. cadastrar garantia;
5. realizar os eventos de carteira;
6. reavaliar a carteira de lotes;
7. marcar o valor a mercado dos lotes;

8. realizar contabilidade;

9. integrar legados.

Não foi encontrada documentação sobre seus objetivos, responsabilidades e restrições para nenhuma delas. Todo o conhecimento sobre as funcionalidades e características do sistema foi obtido com os desenvolvedores mais antigos da aplicação e quase nada houve de contribuição dos artefatos existentes, pois a grande maioria encontrava-se desatualizada.

O desenvolvimento da nova funcionalidade do sistema MMO foi determinado pelo Comitê Estratégico da organização, que foi conduzido pelo Departamento de Controladoria, com o objetivo claro de dar maior transparência, legitimidade e confiabilidade para as operações contratadas pelo sistema MMO. Tratava-se de uma importante ferramenta no gerenciamento das operações do mercado monetário, ao permitir consultar, de forma sumarizada, os resultados financeiros das operações e avaliar a performance dos operadores na contratação de operações no mercado financeiro.

A nova funcionalidade de "Integrar informações MMO e RPO"deverá ser implementada em alta plataforma, com acesso remoto às bases de dados da baixa plataforma, e deverá ser codificado utilizando a linguagem *Cobol*, para as camadas de negócio e persistência, e *Java* para a camada de apresentação.

Para realizar o mapeamento dos programas e sub-rotinas da nova funcionalidade denominada "Integrar informações sistemas MMO e RPO", foi utilizada a abordagem de mapeamento *Top Down* (BPM) e também o Manual de Modelagem de Sistemas, que se encontra anexo ao final do trabalho, no *Apêndice E*. Esse manual é um guia elaborado para ser usado pelos desenvolvedores, com a descrição e orientações de uso dos principais elementos da notação BPMN, assim como a indicação de termos padronizados para nomeação e identificação dos processos, programas e sub-rotinas quanto ao seus objetivos e finalidades.

No processo de implementação de *software* do sistema MMO, existem algumas regras de implementação que devem ser seguidas, principalmente quanto ao uso de programas e sub-rotinas, que é uma parcela de código computacional que executa uma tarefa bem definida, que pode ser executa diversas vezes em um mesmo programa. Elas devem ser utilizadas sempre que for necessário utilizar um mesmo código em várias partes de um programa, ou mesmo quando se deseja que vários programas utilizem um mesmo código.

As sub-rotinas normalmente auxiliam a abstrair a complexidade e facilitam o entendimento dos programas. Não funcionam sozinhas, pois dependem de chamadas de programas ou de outras sub-rotinas para funcionarem.

Nesse sentido, no processo de desenvolvimento para alta plataforma utilizado pelo sistema MMO, os programas podem fazer chamadas a várias sub-rotinas para a realização

de seus objetivos, mas não podem chamar outros programas, enquanto as sub-rotinas caracterizam-se por realizarem tarefas bem definidas e podem fazer chamadas a outras sub-rotinas.

Desse modo, para estabelecer um padrão para especificação de programas e sub-rotinas por meio do uso do BPM, conforme definido no Manual de Modelagem de Sistema localizado no *Apêndice E*, instituiu-se representar as sub-rotinas pelo elemento tarefa de sistema, por tratarem-se da menor parte executável de uma função do sistema, e os programas por subprocessos, uma vez que podem fazer chamadas a outras sub-rotinas.

A utilização da abordagem *Top Down* do BPM, em detrimento da abordagem *Botton Up*, se deu em virtude de que para integrar a nova funcionalidade dentro da cadeia de valor já existente no sistema MMO, seria necessário partir do negócio, tal como definido pelo gestor, para a partir de então detalhar os demais subprocessos, tarefas e fluxo de informações da aplicação. A abordagem ao longo do seu desenvolvimento tem a participação do gestor de negócio, que define quais os conjuntos de informações que deseja que sejam tratados pela funcionalidade. Com a definição dos requisitos, torna-se possível ao desenvolvedor verificar se já existem processos mapeados que ofereçam as informações desejadas (subprocessos reutilizáveis), ou se realmente tratam-se de novos processos (programas e sub-rotinas) que devem ser mapeados (especificados) para codificação.

Durante a fase de definição dos requisitos, alguns eventos causaram grandes dificuldades para a elaboração de um documento de requisitos de qualidade:

- o gestor de negócio do sistema MMO e o da nova funcionalidade pertenciam a departamentos diferentes (Finanças e Controladoria), o que foi um grande dificultador de compartilhamento e colaboração entre as partes;
- dificuldade em marcar as salas e horários de reuniões. As diretorias estavam localizadas em cidades diferentes, e os stakeholders envolvidos não possuíam o mesmo horário de trabalho;
- falta de conhecimento sobre o negócio e também sobre as principais características da aplicação pelo gestor de negócio. O Departamento de Controladoria, responsável pela definição das regras de negócio e restrições da nova funcionalidade, não possuía conhecimento suficientes e necessários para a elaboração do documento de requisitos;
- ausência de documentação de sistema. Não foi localizada a documentação sobre as características, regras de negócio e funcionalidades do sistema MMO. Durante as reuniões de requisitos, em diversos momento houve a necessidade de o analista de TI ausentar-se das reuniões para consultar nos códigos-fonte as informações necessárias para esclarecimentos de dúvidas dos gestores de negócio;

- grande parte das reuniões foram feitas sem a participação do principal stakeholder (gestor de negócio do Departamento de Finanças). Nessas condições, algumas regras de negócio ficaram pendentes de definição, ficando para serem definidos posteriormente durante o processo de implementação.

A falta de documentação de sistema e a não participação do Departamento de Finanças na grande maioria das reuniões foram decisivos para que os encontros de requisitos não fossem suficientes, tornando-se necessário marcar novas reuniões, assim como replanejar as datas definidas em cronograma para os demais processos: análise e projeto de *software*, implementação, testes e implantação.

Apesar de o Departamento de Finanças ser o principal gestor de negócios do sistema MMO, não foi designado como responsável pela nova funcionalidade. Quem a assumiu foi o Departamento de Controladoria, por se tratar de uma necessidade da área estratégica da Instituição, que tinha por objetivo gerenciar o resultado financeiro das operações contratadas e acompanhar o desempenho dos operadores do Departamento de Finanças.

Portanto, neste estudo, para especificação das funcionalidades, programas e sub-rotinas do sistema MMO, foram definidos níveis de detalhamento para organizar os mapeamentos dos processos e auxiliar o usuário na localização do tipo de informação desejado. Portanto decidiu-se por usar os seguintes níveis de detalhamento para o mapeamento dos processos do sistema MMO: nível Macro, nível Intermediário e nível Detalhado, respectivamente.

5.4 Níveis de detalhamento para mapeamento de processos de sistema

Para realizar os mapeamentos dos processos de sistema foram definidos três níveis de detalhamento, que são:

5.4.1 Nível Macro

Trata-se da representação de macro funcionalidades e/ou funcionalidades do sistema. Nesse nível ainda não se tem a visão de programas e sub-rotinas. Para representar-se os processos de nível Macro são utilizados os elementos processos e subprocessos.

Características: normalmente tratam-se de processos não reutilizáveis, pois atendem às necessidades de *software* muito específicas, em relação aos objetivos estratégicos da organização. Nesse nível serão representados, caso existam, os documentos que serviram de direcionamento para o desenvolvimento da solução de *software* e os artefatos que evidenciam que os testes para as funcionalidades foram realizados.

Público Alvo: gestores da área de negócio e desenvolvedores de TI.

No modelo de nível Macro, o gestor de negócio poderá observar a localização da funcionalidade dentro do sistema e identificar com quais outros processos, sistemas e intervenientes ele se integra, seja fornecendo ou recebendo informações.

5.4.2 Nível Intermediário

Trata do detalhamento de partes menores que compõem uma funcionalidade. Neste nível os elementos utilizados para mapeamento das partes que compõem uma funcionalidade são os processos e subprocessos.

Características: no nível intermediário, os processos representam os programas, bem como o fluxo de informações e o relacionamento entre eles.

Público Alvo: gestores da área de negócio e desenvolvedores de TI.

No nível Intermediário é possível identificar os processos que compõem a funcionalidade, visualizar os fluxos dos dados (quem recebe e quem envia as informações) e também verificar quais subprocessos são próprios, ou utilizados de outras aplicações (sistemas).

5.4.3 Nível Detalhado

Trata-se do detalhamento de níveis operacionais do sistema, que representa o passo a passo de diferentes tarefas realizadas internamente por um programa ou sub-rotina. Neste nível, os elementos utilizados para o mapeamento das instruções provenientes das unidades de *software* são respectivamente, as tarefas de sistema, de regra de negócio, subprocessos reutilizáveis, subprocessos de repetição.

Características: este mapeamento substitui o artefato de especificação de componentes, e caracteriza-se por ter descrições mais técnicas, como nomes de variáveis, instruções SQL, tratamentos de arrays. Portanto, é mais direcionado aos desenvolvedores, não se restringe à área de negócio, caso esta venha a interessar-se pelas particularidades de instruções realizadas pelos programas.

Público Alvo: desenvolvedores de TI.

No nível Detalhado, consegue-se identificar o fluxo operacional de um programa ou sub-rotina e verificar o fluxo de informações entre os elementos mais utilizados, que são as tarefas de sistema, regras de negócio e quantidades de gateways (pontos de decisão), realizados internamente por cada unidade de *software*.

Na Figura 5.2 encontram-se representados os níveis de detalhamento dos processos de sistemas, bem como os principais elementos da notação BPMN, utilizados na especificação de cada um dos níveis.

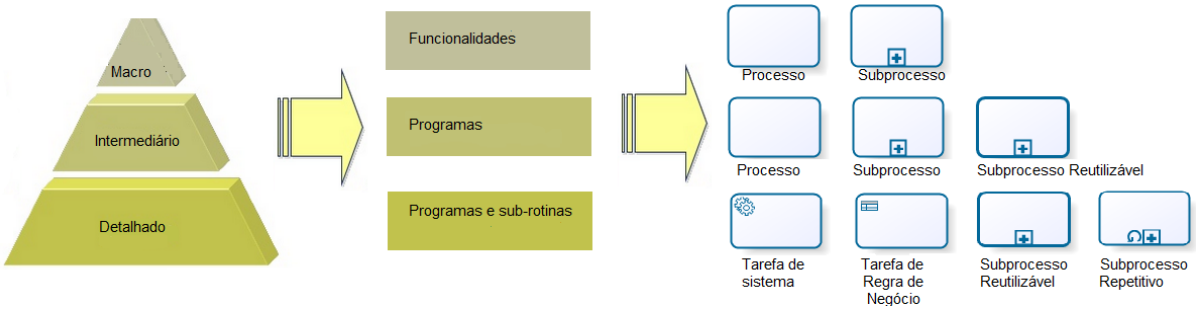

Figura 5.2: Níveis de detalhamento para modelagem de processos de sistema

Fonte: Elaboração própria

No detalhamento das descrições dos processos mapeados nos níveis Macro, Intermediário e Detalhado, vale o princípio de hereditariedade, no qual processos e subprocessos localizados nos níveis Intermediários e Detalhado (filhos) herdam as mesmas informações, descrições e restrições localizadas no nível Macro (pai). Esse é um processo importante que visa garantir a consistência e integridade das informações entre os diferentes níveis de mapeamento de processos e tornar mais simples as modificações e atualizações, que devem ser pontuais e específicas, atualizando apenas os processos de fatos afetados.

As informações apresentadas na descrição dos processos, subprocessos e tarefas, independente do nível de detalhamento (Macro, Intermediário ou Detalhado) selecionado, devem ser concisas, claras e objetivas, a fim de que tanto a área técnica ou negocial, ao analisarem os documentos não tenham dúvidas quanto aos objetivos, comportamento, características e restrições das funcionalidades, programas e sub-rotinas da aplicação.

Nos níveis Macro e Intermediário as informações devem ser mais genéricas e explicativas, com o máximo de abstração de complexidade, apenas descrevendo o que as funcionalidades “devem fazer”, e não o “como fazer”. Por tratarem-se de mapeamentos que representam em diagramas o negócio na visão do gestor, os termos utilizados devem estar mais próximos da linguagem, já que os campos de descrição devem ser utilizados para detalhar as características importantes do sistema, os requisitos não funcionais, como por exemplo, o impacto e a criticidade das funcionalidades para a organização, em situação de falhas e indisponibilidades do sistema.

No modelo de nível Detalhado, as informações devem ser mais técnicas, por tratarem-se de documentos de especificação dos programas e sub-rotinas que serão utilizados pelos implementadores para codificação. Por exemplo, a informação de que todas os programas e sub-rotinas da funcionalidade devem ser implementadas em linguagem de programação *Cobol* devem constar apenas na descrição das funcionalidades de mapeamento de nível Macro, pois dessa forma já se subentende que tal restrição vale para os demais processos e subprocessos nos níveis de mapeamento Intermediários e Detalhado.

No nível Detalhado, as informações devem ser mais precisas e específicas, a fim de auxiliarem os implementadores nas atividades de codificação dos programas e sub-rotinas, como a Instruções SQL para execução de querys, a parametrização de linkages (áreas de dados) na chamada de sub-rotinas, os tratamentos de sqlcodes e os códigos de retornos de sub-rotinas.

Nesse contexto, os usuários das áreas técnica e de negócios, ao consultarem a base de processos de seus sistemas, poderão selecionar as funcionalidades e os níveis de mapeamento de seu interesse que melhor atendam a suas necessidades por informações sobre o funcionamento e comportamento de suas aplicações.

Para facilitar a identificação dos níveis e padronizar a nomeação dos elementos, se o processo ou subprocesso tratar-se de um programa ou sub-rotina, deverá conter o nome do componente seguido de seu objetivo, de modo que fiquem separados por hífen (–). O nome do componente é formado por oito caracteres, que são: sigla do sistema com três letras + um caractere para indicar se o componente é um programa (P) ou sub-rotina (S) + quatro números (aleatórios). Os nomes dos processos, subprocessos, atividades e tarefas, independentes do nível (Macro, Intermediário ou Detalhado), não devem ultrapassar a 64 bytes, conforme definido no Manual de Modelagem de Sistemas, anexado ao final desse trabalho, no *Apêndice E*.

Na próxima seção, serão apresentados os mapeamentos elaborados para a especificação de uma nova funcionalidade do sistema MMO. Nela será demonstrado o passo a passo do desenvolvimento da nova funcionalidade solicitada pelo Departamento de Controladoria utilizando, para isso, o gerenciamento de processos de negócio (BPM) na identificação de processos reutilizáveis.

5.5 Mapeamento dos processos de sistema de uma nova funcionalidade do sistema MMO

Nesta seção buscou-se mapear por meio da abordagem *Top Down* buscou-se mapear a nova funcionalidade do sistema MMO, abrangendo processos desde uma visão de granularidade mais grossa, até uma visão de granularidade mais fina, com o mapeamento e especificação dos programas e sub-rotinas a serem codificados no processo de implementação.

Durante o levantamento das informações, as áreas técnica e negocial demonstraram ter visões particulares e muitas vezes divergentes sobre o funcionamento do sistema, o que tornou necessário a elaboração de vários mapeamentos até que ambas concordassem com o modelo apresentado na Figura 5.3. Observe que as macro-funcionalidades não afetadas pela nova funcionalidade permanecem na cor azul, que é a cor padrão de mapeamento dos processos no *software* Bizagi.

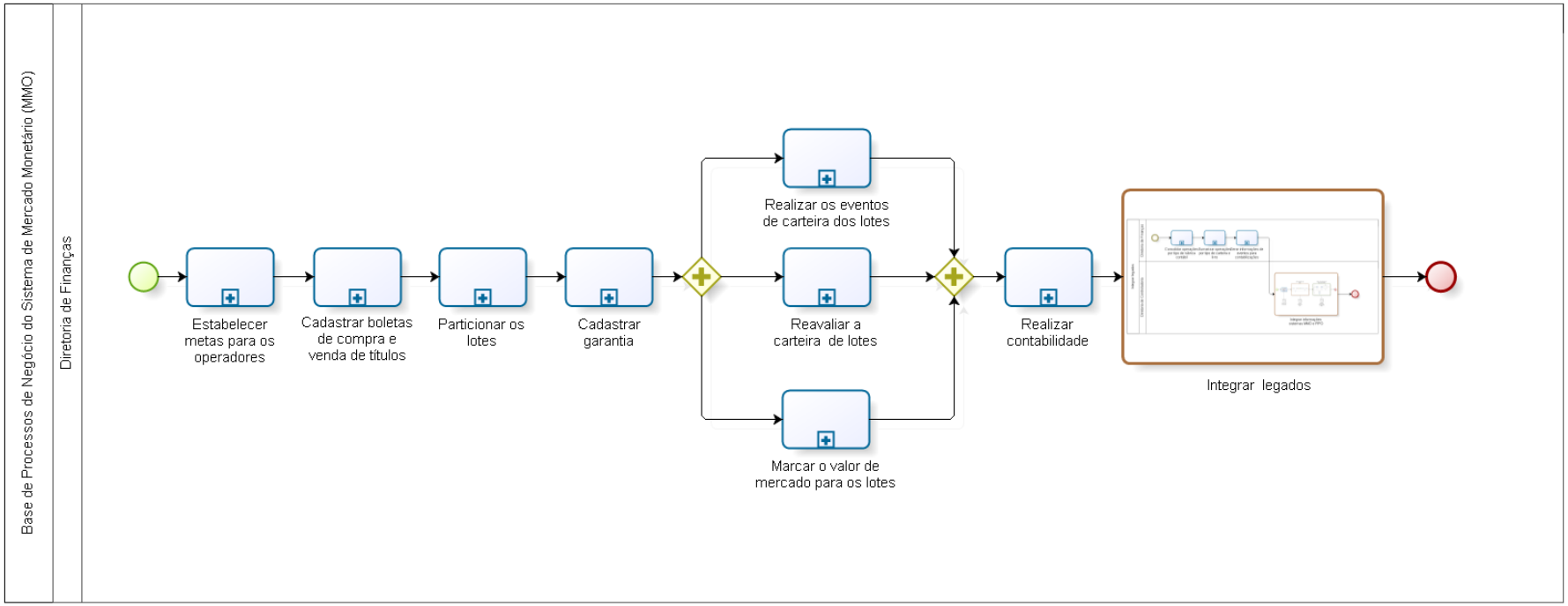

Figura 5.3: Modelo de nível Macro das funcionalidades do sistema MMO

Fonte: Elaboração própria

Na Figura 5.3 é possível ver representado em diagramas as nove macro-funcionalidades do sistema e identificar que o interveniente responsável pelas regras e definições de funcionamento da aplicação é o Departamento de Finanças. Note que a macro-funcionalidade "Integrar legados"encontra-se expandida, a fim de indicar onde será integrada a nova funcionalidade do sistema denominada "Integrar informações MMO e RPO".

Na Figura 5.4, observa-se o nível macro da funcionalidade "Integrar legados", que, por meio da modelagem de processos, é possível não só mapear as funcionalidades do sistema, mas também identificar seus intervenientes, que, neste caso, são dois: Departamento de Controladoria, que é responsável pela definição do processo (funcionalidade) destacado em laranja; e Departamento de Finanças, responsável pelos demais processos já existentes e em funcionamento no sistema MMO.

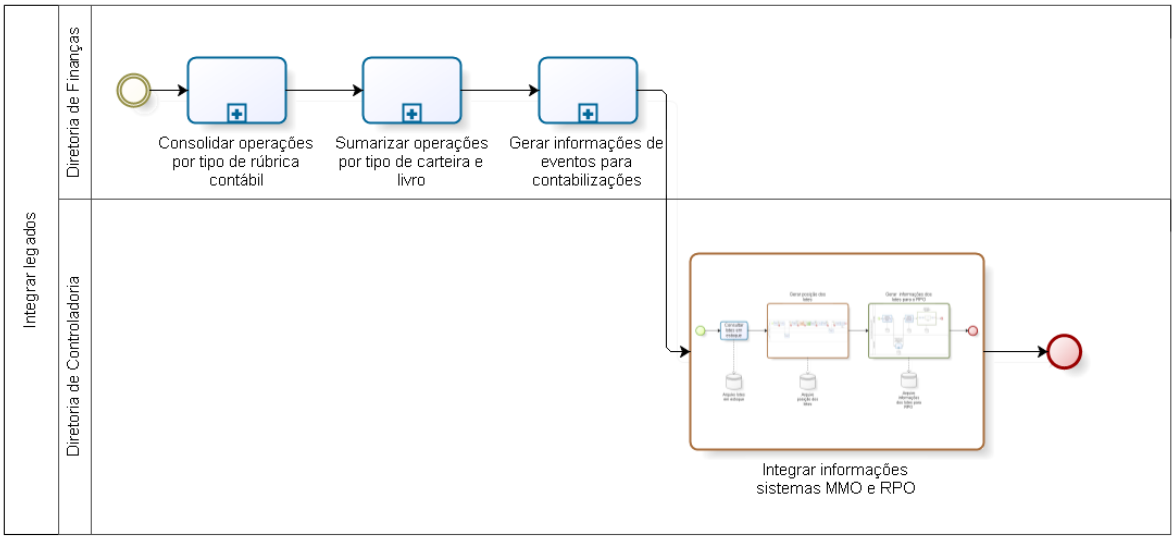

Figura 5.4: Modelo de nível Macro da Funcionalidade Integrar Legado

Fonte: Elaboração própria

Os processos na Figura 5.4 que permanecem com as cores azul são funcionalidades existentes no sistema MMO que deverão ser migrados gradativamente da baixa para a alta plataforma e substituídos na linguagem de programação *Visual Basic*, por *Cobol* e *Java*

Na Figura 5.5 observa-se a representação da funcionalidade "Integrar informações MMO e RPO"no nível Intermediário, onde se consegue identificar o mapeamento de três programas principais: "MMOP0620 – Consultar lotes em estoque", "MMOP0622 – Gerar posição dos lotes"e "MMOP0624 – Gerar informações dos lotes para o sistema RPO".

Nesse mapeamento, quando os processos são muito grandes, eles devem ser transformados em subprocessos, a fim de organizar o mapeamento e facilitar o entendimento dos usuários (gestores de negócio e desenvolvedores) por meio de abstração da complexidade.

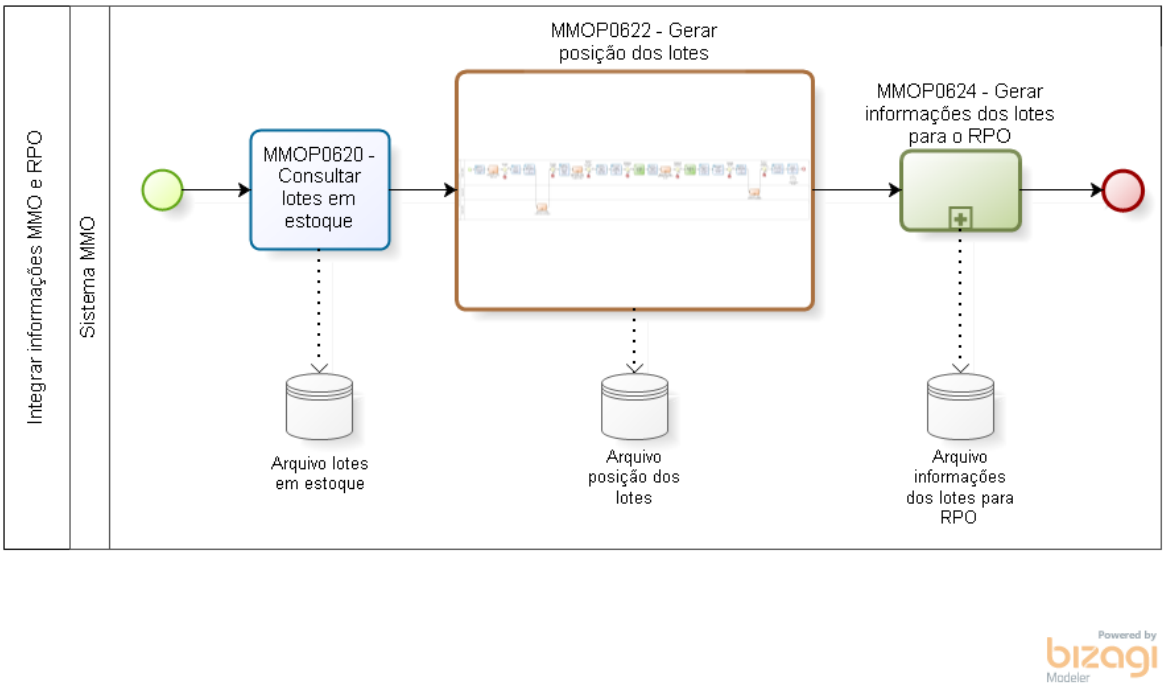

Figura 5.5: Modelo de nível Intermediário de especificação do Processo de Integração sistemas MMO e RPO

Fonte: Elaboração própria

Ainda na Figura 5.5 é possível observar que o programa "MMOP0620 – Consultar lotes em estoque"encontra-se representado pelo elemento “processo”, por tratar-se de um programa simples, para a execução direta de uma única instrução SQL, cujas instruções e orientações necessárias foram colocadas diretamente no campo descrição. Os outros dois programas, "MMOP0622 – Gerar posição dos lotes"e "MMOP0624 – Gerar informações dos lotes para o sistema RPO", foram representados pelos elementos subprocessos por tratarem-se de programas com um maior número de instruções, em que é recomendado a utilização de elementos como tarefas de sistema, tarefas de regra de negócio e elementos de decisão, para especificação do comportamento esperado dos programas e sub-rotinas da aplicação.

Após finalizada a modelagem dos níveis Macro e Intermediário, foi iniciada a fase de Projeto e Análise *Software*, que envolve os seguintes subprocessos: Analisar a solução, Projetar banco de dados, Especificar interface com usuário e Especificar componentes.

Os modelos apresentados (Macro, Intermediário e Detalhado), tiveram sua maior contribuição na atividade de especificar componentes, pois foram eficientes na substituição do artefato "Especificação de Componentes", haja vista que, além de servirem como documentos para os implementadores codificarem os programas e sub-rotinas da funcionalidade, também exerceram o papel de documentação do sistema.

Inicialmente, buscou-se verificar se existiam programas ou sub-rotinas que já disponibilizavam as informações necessárias para geração dos arquivos no sistema RPO. Nessa análise, foi possível identificar que uma parte dos programas já se encontrava pronta, podendo ser reutilizada, enquanto que, outra parte precisaria ser especificada e implementada. Como resultado do processo de projetar *software*, foi especificado um total de 10 componentes entre programas e sub-rotinas, sendo que três tratavam-se de programas compostos por diferentes sub-rotinas, em que parte delas eram rotinas reutilizáveis de outras aplicações, e outras tratavam-se de sub-rotinas com finalidades para atendimento de necessidades particulares da funcionalidade.

Os casos em que as sub-rotinas pudessem ser reutilizadas por outras aplicações, ou tivessem muitas regras de negócio para ser implementadas, foram representados pelo elemento subprocessos da notação BPMN, por sua vez as sub-rotinas coesas, simples ou que não fossem ser reutilizadas, foram implementadas como tarefas de sistema, conforme apresentado no Manual de Modelagem de Processos, localizado ao afinal deste trabalho, no *Apêndice E*.

No nível Intermediário (Figura 5.5), ao se detalhar o processo destacado em laranja "Consultar informações complementares de lotes em estoque", obtém-se acesso à modelagem e à especificação do programa MMOP0622, composto cinco subrotinas do sistema MMO e um do sistema CAD, conforme pode ser observado na *Apêndice F*.

No programa MMOP0622, para consultar as especificações de cada um dos serviços utilizados, deve-se selecionar a propriedade de descrição do processo. No *Apêndice F*, tem-se a especificação de um dos programas da nova funcionalidade, note que esse é um nível detalhado de mapeamento de processos, em que é possível identificar a utilização de diferentes tipos de elementos da notação BPMN para representar todas as regras de negócio, pontos de decisão e movimentação de variáveis necessárias para o alcance do resultado pretendido com o processo de "Gerar a posição dos lotes em estoque".

Para especificação do programa MMOP0622, foram utilizados os seguintes elementos da notação BPMN, são eles:

- **elemento de início** - utilizado para indicar onde o programa se inicia;
- **subprocessos reutilizáveis** - foram utilizados um total de sete subprocessos reutilizáveis para representar as sub-rotinas utilizadas no programa MMOP0622, sendo

cinco do próprio sistema MMO e um do sistema ICM (sistema de cadastro de clientes) e um do sistema ESC (sistema de escrituração de operações). Note que no *Apêndice F* os processos reutilizáveis encontram-se destacados em laranja;

- **tarefas de sistema** - foram utilizados, para representar as atribuições de variáveis para acesso a sub-rotinas, bem como a movimentação de dados para arquivos, e também acesso a sub-rotinas (que não apresentam características de reutilização). Foram definidas um total de 12 tarefas de sistemas para o programa MMOP0622 que se encontram representados na cor azul;
- **tarefas de regras de negócio** - foram utilizadas para identificar mecanismos pré-definidos para tratamento de determinadas informações, a fim de se obter um resultado desejado. No programa MMOP0622 foram definidas duas regras de negócio, que se encontram destacadas em verde;
- **elementos de decisão (Gateway)** - os gateways foram utilizados para momentos de tomada de decisão. No programa MMOP0622, foram especificados um total de sete gateways, que foram colocados imediatamente após as chamadas às sub-rotinas para verificar se as informações solicitadas existiam ou não. De acordo com a resposta, tomava-se a decisão de qual caminho seguir no gateway;
- **elemento de término (Erro)** - os elementos de indicação de término com erro, foram utilizados como um dos caminhos do gateways exclusivos, que para as situações em que as sub-rotinas não encontrassem as informações solicitadas, o programa deveria ser interrompido com a indicação de erro, a fim de que o desenvolvedor pudesse tomar as providências necessárias. Para especificação do programa, foram utilizados um total de 7 gateways exclusivos;
- **elemento de término** - indica fim normal do programa.

Na Figura 5.6, encontra-se representado o procedimento utilizado no *software* Bizagi, para adicionar informações complementares a processos, subprocessos e tarefas utilizados na especificação dos programas e sub-rotinas. Deve-se clicar com o botão direito no elemento desejado e no item Propriedade, selecionar o campo Descrição da aba Básico. As informações contidas no campo descrição dos elementos não apenas devem auxiliar aos desenvolvedores na realização de suas tarefas, mas também servir como fonte de informação atualizada e confiável dos principais processos em funcionamento na aplicação.

Ainda na descrição dos elementos utilizados para a especificação dos programas, merece destaque os subprocessos de repetição. Eles devem ser utilizados para organizar a sequência de eventos que devem acontecer repetidamente para um conjunto de dados, até que uma condição seja atendida.

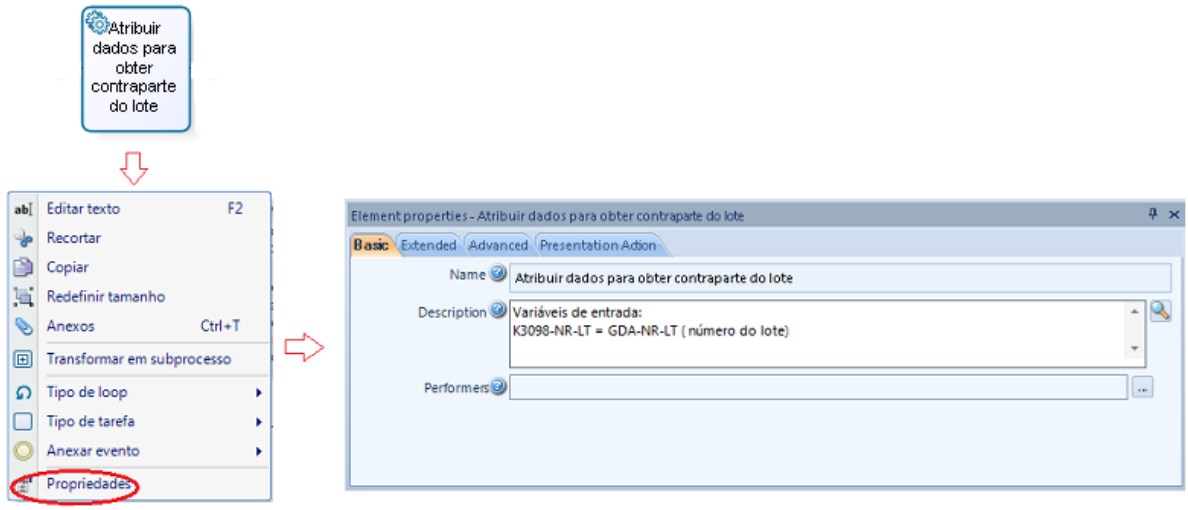

Figura 5.6: Utilização do campo descrição, para atribuição de valores a variáveis

Fonte: Elaboração própria

A Figura 5.7 mostra o subprocesso de repetição na sua forma expandida para demonstrar o seu potencial em relação a abstração de complexidade, a fim de auxilar na compreensão da funcionalidade, bem como deixar o processo simples e acessível a quem quer que o consulte, seja da área técnica ou área negocial.

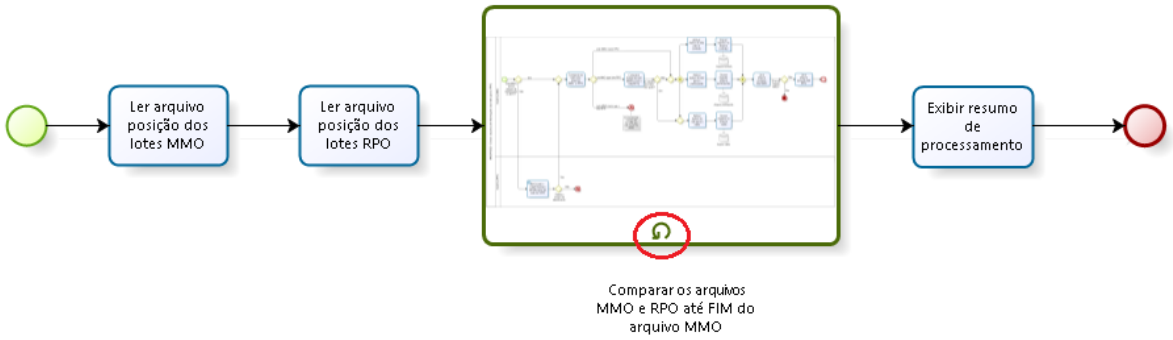

Figura 5.7: Exemplo de utilização do subprocesso de repetição

Fonte: Elaboração própria

5.6 Benefícios do BPM para melhoria dos processos de negócios de um sistema de dados

A aplicação do BPM para modelagem de sistemas pemitiu identificar alguns benefícios como:

- **identificação de intervenientes** - na modelagem e especificação dos processos de negócios de sistema, consegue-se identificar nas rais, quais são os sistemas ou departamento envolvidos na funcionalidade ou processo. Em um nível macro de modelagem, podemos identificar os departamentos responsáveis pelas definições e regras de negócio das funcionalidades do sistema. Na Figura 5.4 consegue-se ver que existem processos de responsabilidade do departamento de finanças e do departamento de controladoria. No nível detalhado,consegue-se observar quais são os sistemas que fornecem informações, e que interagem com a funcionalidade, fornecendo ou recebendo informações. No *Apêndice F* é possível identificar dois sistemas intervenientes o ESC e o ICM.

- **falhas e indisponibilidade** - em caso de falha ou erro na execução de algum processo (programas, sub-rotinas), consegue-se identificar mais facilmente no modelo de negócios quais são os processos diretamente impactados pela indisponibilidade, podendo neste caso ter a dimensão e alcance dos prejuízos. O mapeamento de fluxo de informações entre os processos é essencial no controle e gerenciamento dos riscos, pois uma vez identificado o problema, consegue-se administrá-lo com maior precisão: identificar quais são os processos e intervenientes impactados, saber a criticidade e severidade do risco para a imagem da instituição, e tão logo aconteça o problema, comunicar as partes envolvidas para que utilizem seus planos de contingência para a continuidade dos negócios.

- **comunicação** - o mapeamento no nível Macro e Intermediário é capaz de promover e facilitar a comunicação entre a área técnica e área de negócio, pois trata-se de uma representação gráfica dos processos, capaz de abstrair a complexidade, fazendo com que o foco da análise do processo esteja no "que fazer"ao invés do "como fazer". Os mapeamentos de processos apresentado nas Figuras 5.4 a 5.5 são um tipo de documentação de sistema que preenche uma lacuna de comunicação existente entre as áreas técnica e negocial. Ambas as áreas ao analisarem o mapeamento de processos do sistema MMO, conseguirão identificar em um mesmo documento, diferentes níveis de detalhamento (Macro, Intermediário e Detalhada), podendo consultar, analisar as informações dos processos de acordo com os níveis de detalhamento que forem de seu interesse.

- **localização de artefatos** - a combinação de gerenciamento de processos aliado a outros *softwares* de armazenamento de artefatos mostrou-se intuitivo, ágil e prático na disponibilização centralizada das versões mais recentes de todos os artefatos produzidos durante o desenvolvimento, versionamento ou manutenção do *software*.

- **especificações dos componentes de *software*** - a utilização da modelagem de processos nas especificações de programas e sub-rotinas são mais facilmente compreendidos pelos implementadores, pois trata-se de uma representação gráfica em que é possível "visualizar"com clareza a sequência de tarefas e subprocessos que devem ser realizados para que os resultados sejam alcançados conforme definido em requisito pelo gestor de negócio.

Na próxima seção, dando continuidade ao processo de tratamento dos riscos, para melhoria do processo de implementação, será apresentada proposta de redesenho do processo de desenvolvimento de *software* utilizado no sistema MMO, a partir da inclusão de novas atividades relacionadas ao resultados obtidos nos capítulos 4 e 5 relacionados a: 1) monitoramento e controle da complexidade de códigos-fonte *Cobol*; e 2) relacionadas as atividades de tratamento de riscos para melhoria da qualidade dos artefatos de especificação de programas e documentação de sistema.

5.7 Proposta de melhoria do processo de implementação de *software*

Tendo em vista a importância do processo de implementação na entrega de *software* de qualidade para a área de negócio, nesta seção será apresentado um redesenho do processo de desenvolvimento de software, a fim de promover melhoria no processo de implementação por intermédio da inclusão de atividades relacionadas a monitoramento, controle e tratamento dos riscos que foram abordados nos capítulos 4 e 5 respectivamente.

A fim de promover a melhoria no processo de implementação e a elaboração de artefatos mais completos, foi realizado o redesenho do processo de desenvolvimento, com a inclusão de novas atividades para controle e monitoramento dos riscos relacionados ao principal do processo de implementação: código-fonte. Também foram adicionadas atividades relacionadas ao tratamento de riscos por meio da utilização de modelo de gerenciamento de processos de negócio (BPM) para a especificação de componentes de *software* (programas e sub-rotinas) e documentação do sistema.

As novas atividades foram elaboradas com o objetivo de produzir artefatos que realmente espelhassem o comportamento do sistema e também códigos-fonte mais simples, coesos e estruturados, a fim de proporcionar maior agilidade nas manutenções, bem como no atendimento de novos negócios.

Após o mapeamento *AS IS* do processo de desenvolvimento, foi realizada a análise das atividades e artefatos, para identificar em quais pontos do processo de desenvolvimento deveriam ser incluídas as novas atividades relacionadas à elaboração, à validação e à atu-

alização de documentação de sistema por intermédio de mapeamento dos processos e em quais deveriam ser incluídas as atividades relacionadas ao gerenciamento da complexidade e da qualidade dos códigos-fonte produzidos.

As atividades na cor azul, contempladas no redesenho *TO BE* (*Apêndice G*) são atividades que já existem no modelo atual e que não sofreram qualquer tipo de alteração. A descrição de cada uma dessas atividades já foi feita quando foi elaborado o modelo *AS IS* no capítulo 3.

As atividades em laranja são as atividades que tem por objetivo propor melhorias no processo de implementação ao fornecer, como insumo para a atividade de codificação, os processos e os programas mapeados pelo uso de notação BPMN de mapeamento de processos. Além do que, a documentação elaborada não será descartada quando finalizado o *software*, tal como acontece atualmente.

No *Apêndice G*, encontra-se destacadas em laranja as 11 novas atividades que foram incluídas no processo de desenvolvimento para melhoria do subprocesso de implementação. Nos próximos subitens, serão apresentadas com maiores detalhes as novas atividades incluídas no mapeamento *TO BE*, para melhoria do subprocesso de implementação de software. Elas foram definidas a partir de resultados obtidos na combinação do processo de gestão de riscos a outros modelos e métodos, como a mineração de textos e o gerenciamento de processos de negócios (BPM).

5.7.1 Consultar Ideia

O processo de desenvolvimento de *software* se inicia partir do cadastramento de uma nova Ideia[1] no aplicativo de Solicitação de Serviços de TI.

É responsabilidade do analista de negócio verificar periodicamente se existe “Ideia” pendente de análise e avaliação. Para isso, deve-se executar a atividade "Consultar Ideia", conforme descrito na Tabela 5.2.

Tabela 5.2: Descrição da atividade Consultar Ideia

Nome da Atividade	**Consultar Ideia**
Descrição	• O analista de negócio realiza consulta no aplicativo de Solicitação de Serviços de TI, para verificar se existem “Ideias” cadastradas pela área de negócio pendentes de análise e avaliação.

[1]Ideia: necessidade de solução de *software*, cadastrada pelo gestor de negócio, para aperfeiçoamento, adaptação ou manutenção corretiva do sistema.

Papéis	• Analista de Negócio.
Entrada	• Argumento de pesquisa para localização da Ideia de um determinado departamento de negócio. Ex: Prefixo do departamento, Nome do departamento, Estado da ideia.
Sequência de tarefas	• Acessar o aplicativo de Solicitação de Serviços de TI; • efetuar a consulta a partir da informação do argumento desejado; • avaliar os resultados da pesquisa e selecionar a “Ideia” na situação desejada: Aberta, Avaliada, Em Análise e outros.
Saída	• Número da Ideia, Nome da Ideia.

Fonte: Elaboração própria

5.7.2 Identificar Macro Funcionalidades

Para iniciar a análise das Ideias cadastradas pela área de negócio no aplicativo de Solicitação de Serviços, deve-se levantar todas as informações referentes ao objetivo tratado na nova ideia. Para isso, deve-se realizar a atividade de "Consultar Base de Processos de Negócio do Sistema"conforme apresentado na Tabela 5.5, para localizar os processos e funcionalidades existentes, caso ainda não existam mapeamentos para o sistema deve-se iniciar levantamento das funcionalidades para posterior cadastramento seguindo as orientações da Tabela 5.4.

Nos casos em que ainda não exista uma base de processos de negócio cadastrada para o sistema, deve-se consultar outros documentos sobre o assunto para expandir o entendimento das funcionalidades a serem desenvolvidas. Os documentos que podem ser utilizados para consulta são: Ideia, proposta de abertura de projeto, instrução normativa, procedimento operacional padrão e outros. Também se deve buscar informações complementares junto aos gestores de negócio, bem como com os desenvolvedores do sis-

tema. Para realizar a atividade de "Identificar Macro Funcionalidades"deve-se seguir as orientações contidas na Tabela 5.3.

Tabela 5.3: Descrição da atividade Identificar Macro Funcionalidades

Nome da Atividade	**Identificar Macro Funcionalidades do Sistema**
Descrição	• Levantar informações sobre o funcionamento do sistema a partir de informações levantadas com os gestores de negócio e desenvolvedores, bem como em documentos que possam complementar e esclarecer características importantes da aplicação.
Papéis	• Analista de Negócio.
Entrada	• Ideia, demandas de solicitações de serviços, proposta de abertura de projetos (PAP's) e anexos, instruções normativas, procedimento operacional padrão da área de negócio e outros.
Sequência de tarefas	• Analisar nos documentos os conceitos sobre o negócio; • esclarecer as dúvidas com o gestor de negócio; • identificar os intervenientes das funcionalidades: parceiros, clientes, usuários finais e outros sistemas.
Saída	• Relatório das funcionalidades com a descrição dos seus objetivos, características.

Fonte: Elaboração própria

5.7.3 Cadastrar Base de Processos de Negócios do Sistema

Para analisar a ideia, deve-se entender bem o negócio do sistema e, caso não exista uma Base de Processos do Sistema, deve-se solicitar autorização ao administrador de acesso para o cadastramento de um repositório, a fim de organizar logicamente os mapeamentos de processos do sistema, seguindo o passo a passo das orientações constantes da Tabela 5.4.

Tabela 5.4: Descrição da atividade Cadastrar Base de Processos de Negócios do Sistema

Nome da Atividade	**Cadastrar Base de Processos de Negócios do Sistema**
Descrição	• Autorizar a aplicação a incluir no software de gerenciamento de processos de negócios as funcionalidades mapeadas.
Papéis	• Analista de Negócio.
Entrada	• Cadastrar no *software* de modelagem de processos de negócios, um repositório de organização lógica das funcionalidades e processos de um sistema (aplicação) desejado. Exemplos de *software* de gerenciamento de processos: Aris, Bizagi, Bonita e outros.
Sequência de tarefas	• solicitar ao administrador acesso a ferramenta de modelagem de processos de negócio; • cadastrar a sigla do sistema e demais informações importantes para a sua identificação: Unidade Gestora, Área de Interesse, Núcleo Responsável pelo Suporte e outros; • representar graficamente as funcionalidades que se encontram em funcionamento no sistema, bem como suas características, requisitos funcionais e não funcionais e intervenientes, usuários.
Saída	• Mensagem de operação realizada com sucesso ou mensagem indicando o erro.

Fonte: Elaboração própria

5.7.4 Consultar Base de Processos de Negócio do Sistema

O analista de negócio deve consultar os mapeamentos de funcionalidades do sistema, para verificar se já existem processos mapeados que possam ser reutilizados, bem como consultar os mapeamentos para esclarecimento de dúvidas e para ampliar o conhecimento sobre o comportamento do sistema e seus processos de negócio. O desenvolvedor, para

realizar a consulta à Base de processos do sistema, deve seguir as orientações contidas na Tabela 5.5.

Tabela 5.5: Descrição da atividade Consultar Base de Processos de Negócios do Sistema

Nome da Atividade	**Consultar Base de Processos de Negócio do Sistema**
Descrição	• Consultar no *software* de modelagem de processos de negócios, o detalhamento da cadeia de valor de uma funcionalidade, bem como seus processos e intervenientes.
Papéis	• Analista de Negócio.
Entrada	• Argumento de pesquisa para localização e detalhamento da funcionalidade, processo ou subprocesso desejado. Ex.: pesquisar pelo processo "Cadastrar Boleta".
Sequência de tarefas	• Acessar o aplicativo de modelagem de processos de negócio; • efetuar consulta a partir do argumento desejado; • avaliar os resultados da pesquisa para verificar se os processos, ou subprocessos encontrados referem-se ao detalhamento da funcionalidade desejada.
Saída	• Diagramas, mapas ou modelos de representação gráfica das funcionalidades, processos e subprocessos.

Fonte: Elaboração própria

5.7.5 Classificar Forma de Atendimento da Ideia

O analista de negócio, após análise dos documentos e modelos de mapeamento de negócios relacionados ao assunto e objetivo da nova “Ideia”, deve direcionar a sua forma de atendimento como demanda ou projeto. Para determinar a forma de atendimento da “Ideia” deve-se avaliar os critérios constantes da Tabela 5.6.

A classificação da “Ideia” em demanda ou projeto, leva em consideração a quantidade de critérios satisfeitos e a quantidade e complexidade de itens a serem atendidos. Após, então, inicia-se as atividades de consolidação do escopo, planejamento das iterações, especificação e validação dos requisitos.

Tabela 5.6: Descrição da atividade Classificar Forma de Atendimento da Ideia

Nome da Atividade	**Classificar Forma de Atendimento da Ideia**
Descrição	• O Analista de Negócio, com base nas informações e critérios analisados, direciona o atendimento da Ideia como demanda ou projeto.
Papéis	• Analista de Negócio.
Entrada	• Diagramas armazenados na base de modelagem de processos de negócio; • artefatos de intervenções anteriores armazenados em repositório de controle de versão (CVS).
Sequência de tarefas	• Avaliar os diagramas e informações da ideia quanto: – quantidade de intervenientes; – clareza de requisitos; – índice de qualidade do processo produtivo (percentual de falhas no ambiente de produção); – estimativa de esforço; – quantidades de sistemas envolvidos; – complexidade de construção.
Saída	• Definição da forma de atendimento da Ideia como projeto ou demanda.

Fonte: Elaboração própria

5.7.6 Projetar Processos de Negócios do Sistema - nível Macro

Após finalizada a etapa de validação dos requisitos, segue-se para a etapa de elaboração do modelo de nível Macro, fase em que a necessidade e a visão dos gestores sobre o negócio são representados graficamente por diagramas, que oferecem alguns benefícios e vantagens como:

- • facilita o entendimento do negócio ao representar graficamente e decompor em processos menores os casos de uso definidos em requisitos;
- promove a comunicação entre a área técnica e de negócio ao utilizar uma notação simples e intuitiva que expressa os processos em um único diagrama compreensível por todos os participantes.

Tabela 5.7: Descrição da atividade Projetar Processos de Negócios do Sistema - nível Macro

Nome da Atividade	**Projetar Processos de Negócios do Sistema - nível Macro**
Descrição	• O analista de negócio deve definir uma visão de processos para a funcionalidade, observando o que foi solicitado e validado em requisitos.
Papéis	• Analista de Negócio.
Entrada	• Documento de Requisitos (DRI), Requisitos de Níveis de Serviço (RNS).
Sequência de tarefas	• Acessar o *software* de modelagem de processos; • se a especificação tratar-se de uma funcionalidade nova, selecionar a opção de inclusão de novo modelo; • definir uma visão de processos para a funcionalidade, de acordo com o que foi definido em requisitos;

Sequência de tarefas	• Utilizar para a modelagem as orientações do Manual de Modelagem de Sistema localizados no *Apêndice E*; • registrar no campo de descrições os intervenientes, sistemas, canais, requisitos não funcionais e outras informações importantes que auxiliem no entendimento do funcionamento do novo processos de negócio; • salvar o modelo e disponibilizá-lo para iniciar o subprocesso de análise e projeto de *software*.
Saída	• Mapeamento dos sistemas das funcionalidades em nível Macro.

Fonte: Elaboração própria

5.7.7 Consultar Catálogo de Componentes

Após finalizada a atividade de "Atualizar Base de Processos de Negócios do Sistema – nível Macro", deve-se iniciar as atividades do subprocesso de Análise e Projeto de *Software*: Analisar a solução, Projetar Banco de dados e Especificar interface com o usuário.

Depois de reunir o máximo de informações sobre a funcionalidade do sistema, o projetista de *software* deve verificar se já existem outros processos no sistema que realizem as mesmas funções (tarefas) que se deseja implementar.

Essa consulta tem por finalidade agilizar o processo de desenvolvimento, ao identificar programas ou sub-rotinas que já ofereçam o serviço solicitado e que possam ser reutilizados. Para a realização dessa atividade deve-se seguir as orientações contidas na Tabela 5.8

Tabela 5.8: Descrição da atividade Consultar Catálogo de Componentes

Nome da Atividade	**Consultar Catálogo de Componentes**
Descrição	• O projetista de *software* deve verificar se já existem programas ou sub-rotinas que ofereçam os serviços necessários e possam ser reutilizados.

Papéis	• Projetista de *software*.
Entrada	• • Argumento de pesquisa sobre componentes (programas e sub-rotinas) a serem consumidos na intervenção: Nome do programa, objetivo.
Sequência de tarefas	• Avaliar os resultados da pesquisa, para verificar se os programas e sub-rotinas atendem às necessidades de implementação do processo de negócio da funcionalidade.
Saída	• Nome do programa, Versão do programa.

Fonte: Elaboração própria

Após finalizada a consulta, o projetista de *software* deve elaborar o Documento de Componentes de Publicados (DCP), indicando quais os programas e sub-rotinas já se encontram prontos para utilização e quais devem ser especificados para codificação.

5.7.8 Projetar Componentes de *Software* - nível Intermediário e Detalhado

Após finalizada a atividade de "Consultar Catálogo de Componentes"deve-se iniciar a especificação dos programas e sub-rotinas novos e atualizar os que já existem utilizando o *software* de mapeamento de processos de negócios.

Essa atividade deverá ser realizada a partir do modelo de mapeamento da funcionalidade em nível Macro, realizada anteriormente na atividade "Atualizar Base de Processos de Negócios – nível Macro" pelo analista de negócio.

Tabela 5.9: Descrição da atividade Projetar Componentes de *Software* - nível Intermediário e Detalhado

Nome da Atividade	**Projetar Componentes de *Software* - nível Intermediário e Detalhado**

Descrição	• Especificar os componentes de *software* para que possam ser implementados pelos desenvolvedores de *software*.
Papéis	• Projetista de *software*.
Entrada	• Documento de Requisitos (DRI), Documento de Componentes Publicados (DCP), Especificação Tela (EST) e Modelo de Dados (MER).
Sequência de tarefas	• Acessar o *software* de modelagem de processos; • se a especificação tratar-se de um componente novo, selecionar a opção de inclusão de novo modelo; • se a especificação tratar-se de alteração de um componente já existente, seguir as orientações da atividade "Consultar Base de Processos de Negócio do Sistema"para localizar o mapeamento existente para a funcionalidade. Caso não seja localizado, realizar a inclusão de um novo modelo; • elaborar a especificação dos componentes (novos e alterações), seguindo as orientações do Manual de Modelagem de Sistemas *Apêndice E*; • representar graficamente a sequência do fluxo de informações entre os componentes (cadeia de valor); • Salvar o modelo e disponibilizá-lo para iniciar o subprocesso de implementação.
Saída	• Mapeamento dos processos de sistema em nível Detalhado e Intermediário de especificação para codificação de programas e subrotinas.

Fonte: Elaboração própria

5.7.9 Validar Processos de Negócio do Sistema

Finalizada a etapa de especificação dos componentes de nível Intermediário e Detalhado, deve-se submeter o modelo para avaliação de qualidade ao "Comitê de Validação dos Processos de Negócios do Sistema", que irá verificar se todas as funções e processos descritos como necessárias no documento de requisitos encontram-se mapeados nos modelos de processos de negócios, bem como se todos os intervenientes foram listados.

O comitê de validação é composto por um analista de negócio, um gestor de negócio e um analista de TI, que tem por finalidade garantir que as informações e descrições dos processos estejam claros, concisos e completos tanto para a área de negócio, quanto para a área técnica.

Para realizar esta atividade, seguir as orientações contidas na Tabela 5.10.

Tabela 5.10: Descrição da atividade Validar Processos de Negócio do Sistema

Nome da Atividade	**Validar Processos de Negócio do Sistema**
Descrição	• Atividade que visa garantir consonância entre o que foi solicitado em requisitos e o que foi projetado para implementação.
Papéis	• Comitê de Validação dos Processos de Negócios do Sistema.
Entrada	• Modelos de Processos de negócios nível Macro, Intermediário e Detalhado.
Sequência de tarefas	• Acessar o *software* de modelagem de processos; • validar os relacionamentos entre os processos, subprocessos e tarefas nos diferentes níveis de mapeamento; • verificar se os elementos, descrições e nomes encontram-se de acordo com o Manual de Modelagem de Sistemas, *Apêndice E*; • Validação formal do gestor de negócio, analista de negócio e desenvolvedor quanto a conformidade dos mapeamentos com o que foi solicitado no documento de requisitos.

Saída	• Mapeamento dos processos de negócios do sistema em níveis Macro, Intermediário e Detalhado.

Fonte: Elaboração própria

5.7.10 Avaliar Códigos-fonte

Após finalizada a atividade de "Validar modelos de processos de negócios do sistema", deve-se iniciar o subprocesso de implementação de *software* para codificação dos componentes especificados nos modelos de processos de negócios - nível Macro, Intermediário e Detalhado.

A atividade de codificação deverá ser realizada de acordo com o que foi especificado pelo projetista de *software*. Após finalizada a atividade de codificação, a fim de garantir a qualidade do *software*, os programas e sub-rotinas deverão ser submetidos ao classificador automático de códigos-fonte *Cobol*, para verificação da classe a que o programa ou sub-rotina pertence: Misto, Negócio ou Base de Dados.

Para os programas classificados como Misto, o desenvolvedor deverá fazer ajustes no programa até que ele se enquadre na classe de Negócio ou de Base de Dados, pois, conforme levantamento realizado no capítulo 4, essas duas classes de programas foram as que apresentaram o menor percentual de falhas no sistema MMO durante o período de 12/2014 a 05/2015, 26,79% e 15,18% respectivamente.

Para as situações em que realmente se mostre necessário a utilização de programas mistos, como por exemplo manutenção emergencial de programas antigos, o desenvolvedor deverá cadastrar, no aplicativo de Transferência Integrada de Módulos, uma justificativa para solicitar a autorização de seu uso no sistema.

Após finalizada a atividade de codificação, os programas e sub-rotinas classificados como Negócio ou Base de Dados já podem iniciar as atividades de testes unitários e de integração para validação dos resultados.

Os programas classificados como Misto somente seguem para as próximas atividades de implementação, após serem avaliadas pelos administradores dos processos de implementação.

Para realizar essa atividade, deve-se seguir as orientações contidas na Tabela 5.11.

Tabela 5.11: Descrição da atividade Avaliar Códigos-fonte

Nome da Atividade	**Avaliar Códigos-fonte**

Descrição	• Atividade que visa avaliar a estrutura dos códigos-fonte para classificá-los quanto a sua finalidade e complexidade.
Papéis	• Implementador de *software*.
Entrada	• Código-fonte (CDF).
Sequência de tarefas	• Acessar o *software* Rstudio; • verificar se o pacote TM (próprio para mineração de textos) encontra-se instalado; • executar o script que prevê a preparação dos dados para a atividade de mineração (limpeza dos dados, remoção de stopwords, transformação dos termos em Matriz Documento Termo); • verificar o resultado de classificação do *software*: Misto, Base de Dados ou Negócio.
Saída	• Códigos-fonte classificado em Misto, Base de Dados ou Negócio.

Fonte: Elaboração própria

5.7.11 Analisar Justificativa de Complexidade

É responsabilidade do administrador do processo de implementação verificar periodicamente se existem pedidos de avaliação de complexidade de códigos-fonte *Cobol* cadastrado no aplicativo de Transferência Integrada de Módulos, conforme descrito na Tabela 5.12.

Tabela 5.12: Descrição da atividade Analisar Justificativa de Complexidade

Nome da Atividade	**Analisar Justificativa de Complexidade**

Descrição	• Esta atividade tem por objetivo avaliar manutenção e continuidade do uso de programas e sub-rotinas consideradas complexas pelo classificador de códigos-fonte *Cobol*.
Papéis	• Administrador do processo de implementação.
Entrada	• Código-fonte, histórico de falhas e alterações no código-fonte, informações do negócio do sistema, quantidade de processos dependentes.
Sequência de tarefas	• Acessar o aplicativo de Transferência Integrada dos Módulos; • verificar a existência de programas e sub-rotinas pendentes de avaliação; • analisar a informação preenchida no campo de descrição da justificativa; • avaliar critérios como: qual o negócio do sistema, quantos processos dependem da rotina, quais são os intervenientes afetados, qual o histórico de falhas e alterações do código-fonte em um determinado período e outros; • registrar a decisão, informando um parecer, justificativa.
Saída	• Recusa ou aceite do módulo, manifestando um parecer justificando a decisão.

Fonte: Elaboração própria

Com a inclusão dessas 11 novas atividades no processo de implementação, espera-se ter como resultados a redução de falhas nos sistemas, em razão da realização das atividades de tratamento e monitoramento de riscos que se encontram detalhados. Nas atividades encontram-se definidas as responsabilidades, os objetivos e as saídas de cada uma, a fim de facilitar o processo de gerenciamento e controle realizados pelos administradores da instituição.

Como resultado desse processo de melhoria, é possível verificar que houve necessidade de criar o "Comitê de Validação de Base de Processos de Negócios", com pessoas responsáveis pela verificação e avaliação da qualidade de especificações e documentações das funcionalidades do sistema, a fim de verificar se as atendem ao que foi solicitado pelo gestor de negócios na especificação de requisitos.

Também se observa benefícios com relação à reutilização de programas e sub-rotinas dos sistemas, o que representa redução de custos, já que não será necessário reinventar a roda a todo instante, buscando especificar serviços que já se encontram prontos, fazendo com que clientes e gestores fiquem satisfeitos com a segurança e a agilidade no atendimento das novas necessidades de *software* e manutenção das que já existem.

Conclusão

Dentre as tensões presentes no mercado altamente dinâmico e competitivo, a conquista e manutenção de posições e clientes é um desafio perene. Aperfeiçoar a gestão e os processos e mitigar riscos asseguram melhores condições para as instituições atingirem seus objetivos.

A fim de promover melhorias no processo de implementação de *software* de um sistema de dados de operações do mercado monetário, desenvolveu-se um estudo baseado na norma ABNT NBR ISO 31000:2009 de gestão de riscos, para direcionamento e estruturação da pesquisa.

O processo de gestão de riscos foi selecionado por tratar e explorar, de maneira abrangente e ampla, as ameaças e incertezas que envolvem as organizações, como por exemplo, os aspectos financeiros, operacionais, de segurança da informação, dentre outros. Outra característica que coloca o processo de gestão de riscos em destaque é o fato de ser um padrão internacional genérico e abrangente que pode ser aplicado a diferentes organizações, assim como também pode ser adaptado para processos e funções específicos.

Em relação ao objetivo específico 1, "Fazer o diagnóstico do processo de implementação de *software* de sistema de dados de operações do mercado monetário de uma instituição financeira", ele foi atendido no capítulo 3. Para o diagnóstico do processo de implementação, foram utilizadas algumas atividades do processo de gestão de riscos para sua realização: estabelecimento do contexto, identificação dos riscos, avaliação dos riscos e análise dos riscos.

No estabelecimento do contexto interno e de gestão de risco do processo de implementação, foram utilizados a ferramenta SIPOC e o mapeamento de processos *AS IS*, que foram essenciais para o levantamento dos elementos que tinham influência nos resultados das atividades e tarefas do processo de implementação, bem como para se ter uma visão macro e ampliada do processo.

Finalizada a atividade de estabelecimento do contexto, seguiu-se para a de identificação de riscos, na qual foram utilizados questionários e entrevistas para a coleta de dados junto aos desenvolvedores do sistema MMO.

Para a verificação dos questionários sobre a avaliação das atividades do processo de implementação de *software* foi utilizado o método de análise multicritério AHP, que é um método que se caracteriza por comparar, par a par, as alternativas segundo critérios específicos. A aplicação do multicritério na análise das atividades do processo de desenvolvimento de *software* não somente auxiliou na identificação das atividades mais relevantes do processo sobre diferentes aspectos, como também permitiu analisar, de forma quantitativa, dados qualitativos, já que a ferramenta utiliza pesos e cálculos matemáticos para tornar o resultado mais tangível e com menos subjetividade aos tomadores de decisão. O uso do AHP permitiu avaliar as atividades e identificar, na opinião dos desenvolvedores, qual era a atividades de maior importância, dificuldade e risco.

Com a aplicação do processo de gestão de riscos, os desenvolvedores do sistema MMO puderam conhecer e controlar os principais riscos e vulnerabilidades identificados no processo de implementação. Além do que puderam definir critérios para classificar e comparar os riscos entre si, a fim de aplicar tratamento apropriado de acordo com o nível ou grau de risco associado.

O objetivo específico 2, "Propor um modelo de classificação automática da complexidade de códigos fonte *Cobol*, a fim de controlar e monitorar a qualidade dos produtos e serviços oferecidos pelo processo de implementação de *software* de um sistema de dados de operações do mercado monetário de uma instituição financeira"foi atendido no capítulo 4, no qual se recorreu ao uso do processo de mineração de textos com a aplicação de algoritmos supervisionados de classificação para identificar complexidade em códigos-fonte, a partir da identificação de termos que pudessem identificar padrões repetitivos entre esses códigos-fonte analisados. Pelos resultados, observou-se que o Naive Bayes foi o que apresentou um melhor resultado, com um F-Measure de 0,97.

O objetivo específico 3, "Propor melhorias na especificação de componentes e processo de desenvolvimento de *software*, a fim de aumentar a agilidade, segurança e tratamento dos riscos no processo de implementação do sistema de dados estudado"foi atendido no capítulo 5, no qual foi aplicado o modelo de gerenciamento de processos de negócio para a especificação de componentes de*software* (programas e sub-rotinas), bem como foi elaborado um manual com orientações para Modelagem de Processos de Sistema localizado no *Apêndice E*.

Portanto, neste estudo, para a especificação das funcionalidades, programas e sub-rotinas do sistema MMO, foram definidos níveis de detalhamento para organizar os mapeamentos dos processos de negócios do sistema e auxiliar o usuário na localização do tipo de informação desejada. Para isso, foram definidos e utilizados os seguintes níveis de detalhamento para o mapeamento dos processos de negócios do sistema MMO: nível Macro, nível Intermediário e nível Detalhado.

O BPM se mostrou viável para especificação de componentes de *software* por tratar-se de um modelo simples, capaz de abstrair a complexidade dos processos, apenas exibindo os detalhes dos processos e funcionalidades, somente quando solicitados pelos usuários. Com o uso do BPM na especificação de componentes de *software*, busca-se manter a documentação do sistema atualizada, a fim de que seja uma fonte segura e confiável de informações, para apoio à tomada de decisão dos gestores de negócio, bem como que auxilie os analistas a identificarem situações de risco em suas aplicações.

O objetivo geral deste trabalho, "Propor melhorias no processo de implementação de *software* de um sistema de dados de operações do mercado monetário, que contemple o processo de gestão de riscos da norma ABNT NBR ISO 31000:2009 e gerenciamento de processos de negócios (BPM)"foi alcançado no capítulo 5 a partir do redesenho elaborado com a aplicação do mapeamento de processo *TO BE*, no qual foram propostas a inclusão de 11 novas atividades no processo de desenvolvimento baseados nos resultados e nas técnicas utilizadas nos objetivos específicos 2 e 3 relacionados ao tratamento, monitoramento e análise crítica dos riscos identificados.

Dessa forma, sugere-se que a organização passe a adotar esse conjunto de modelos, a fim de registrar e gerar informações atualizadas sobre o funcionamento do sistema e facilitar o processo de trabalho dos desenvolvedores no processo de implementação de *software*, por meio do uso de uma documentação de processos capaz de absrair a complexidade das informações desejadas.

Por fim, pode-se afirmar que o processo de gestão de riscos contribui para a eficiência de processos de desenvolvimento de *software*, em razão de que ela permitiu estruturar, organizar e coordenar a realização dos objetivos referentes a riscos para o alcance do objetivo geral. Nesse contexto, percebe-se que a qualidade de um *software*, não está apenas na redução de custos, na otimização dos processos e inclusão de novas ferramentas do mercado, ela também está na atuação sobre as suas incertezas e ameaças.

Trabalhos Futuros

Espera-se que o trabalho realizado induza ao desenvolvimento de novos trabalhos para ampliação da pesquisa e desenvolvimentos de novos ajustes mais aderentes às necessidades dos subprocessos de desenvolvimento de *software*.

A pesquisa sobre mineração de textos poderá ser ampliada a partir da utilização de todos os termos do código-fonte para a identificação de padrões entre programas com estruturas semelhantes. A mineração de textos também poderá ser aplicada a outras linguagens de programação, assim como podem ser utilizados outros algoritmos para detecção de padrões, tendências e características não triviais e óbvias nos códigos-fonte.

Também poderá ser feito um estudo comparativo entre a qualidade e performance dos códigos-fonte produzidos de forma tradicional e utilizando os novos modelos de especificação de componentes (Macro, Intermediário e Detalhado).

Outra aplicação possível é propor tratamento aos demais riscos identificados no processo de implementação de software, assim como realizar o mesmo processo de gestão de riscos para os demais subprocessos de desenvolvimento de *software*: requisitos, análise e projeto, testes e implantação.

Referências Bibliográficas

[1] FALBO, R. de A. Engenharia de software. Universidade Federal do Espirito Santo, UFES, 2014. 1, 13, 98

[2] GOMES, L. F. A. M. *Teoria da decisão*. [S.l.]: Thomson Learning, 2007. 2

[3] SILVA, M. T. Mecanismos de tratamento de exceções e controle de fluxo excepcional em linhas de produto. Pontifícia Universidade Católica do Rio de Janeiro, 2008. 2

[4] CÔRTES, M. L.; CHIOSSI, T. C. *Modelos de qualidade de software*. [S.l.]: Ed. da Unicamp, 2001. 8

[5] ABRAN, A. et al. Guide to the software engineering body of knowledge. Los Alamitos. *CA: IEEE Computer Society*, 2004. 8, 9, 10

[6] PRESSMAN, R. S. *Engenharia de software*. [S.l.]: McGraw Hill Brasil, 2011. 8

[7] LOPES, P. S. N. D. *Uma taxonomia da pesquisa na área de Engenharia de Requisitos*. Dissertação (Mestrado) — Universidade de São Paulo, 2002. 9

[8] PRESSMAN, R. S. *Software engineering: a practitioner's approach*. [S.l.]: Palgrave Macmillan, 2005. 9, 10, 47, 97

[9] BENNETT, K. H.; RAJLICH, V. T. Software maintenance and evolution: a roadmap. In: ACM. *Proceedings of the Conference on the Future of Software Engineering*. [S.l.], 2000. p. 73–87. 10

[10] SOMMERVILLE, I. et al. *Engenharia de software*. [S.l.]: Addison Wesley São Paulo, 2003. v. 6. 10, 11, 12

[11] CAMPOS, V. F. TQC: Controle da qualidade total. *Belo Horizonte: Fundação Christiano Ottoni*, 1992. 11

[12] MELLO, C. H. P. et al. *ISO 9001: 2000: sistema de gestão da qualidade para operações de produção e serviços*. [S.l.]: Atlas, 2002. 11

[13] HARRINGTON, H. J.; ESSELING, E. K.; VAN, H. Business process improvement. McGraw, 1991. 11

[14] HAMMER, M. Reengineering work: don't automate, obliterate. *Harvard business review*, v. 68, n. 4, p. 104–112, 1990. 11

[15] GONÇALVES, J. E. L. As empresas são grandes coleções de processos. *RAE*, Scielo Brasil, v. 40, n. 1, p. 7, 2000. 12

[16] AGUILAR, E. R. et al. Evaluation measures for business process models. In: ACM. *Proceedings of the 2006 ACM symposium on Applied computing*. [S.l.], 2006. p. 1567–1568. 12

[17] GONÇALVES, J. E. L. et al. Reengenharia das empresas: passando a limpo. In: . [S.l.]: Atlas, 1995. 15

[18] MELLO, A. E. N. S. de. *Aplicação do mapeamento de processos e da simulação no desenvolvimento de projetos de processos produtivos*. Dissertação (Mestrado) — Universidade Federal de Itajubá, 2008. 15

[19] FERNANDES, M. M. *Análise do processo de seleção de projetos Seis Sigma em empresas de manufatura no Brasil*. Dissertação (Mestrado) — Universidade Federal de Itajubá, 2006. 15, 17

[20] MORRIS, B. Service management: Operations, strategy and information technology. *International Journal of Service Industry Management*, Emerald Group Publishing Limited, v. 10, n. 2, p. 3–4, 1999. 15

[21] BARNES, R. M. *Estudo de movimentos e de tempos: projeto e medida do trabalho*. [S.l.]: Editora Edgard Blu cher, 1977. 16

[22] ECKES, G. *A revolução Seis Sigma: o método que levou a GE e outras empresas a transformar processos em lucro*. [S.l.]: Elsevier, 2004. 16, 17

[23] PANDE, P.; NEUMAN, R.; CAVANAGH, R. Estratégia Seis Sigma: como a GE, a Motorola e outras grandes empresas estão a melhorar o seu desempenho. *Rio de Janeiro: Qualitymark*, 2001. 16, 17

[24] KELLER, P. *Six Sigma Demystified*. [S.l.]: McGraw Hill Professional, 2005. 16

[25] SIMON, K. SIPOC diagram. *Retrieved January*, v. 15, p. 2008, 2007. 17

[26] HARRY, M. J. Six sigma: a breakthrough strategy for profitability. *Quality progress*, American Society for Quality, v. 31, n. 5, p. 60, 1998. 17

[27] VIANA, A. d. F. d. S. *Melhoria contínua no controlo de materiais críticos*. Dissertação (Mestrado), 2011. 17

[28] MIYASHITA, P. T.; SALOMON, V. A. P. Mapeamento de processos em empresa prestadora de serviços de seguros. *Encontro Nacional de Engenharia de Produção, XXXI*, 2011. 17

[29] DIAS, S. M. *Implementação da metodologia Lean Seis Sigma O caso do serviço de oftalmologia dos hospitais da Universidade de Coimbra*. Dissertação (Mestrado) — Faculdade de Ciências e Tecnologia da Universidade de Coimbra, 2011. 17

[30] CRUZ, T. *BPM & BPMS-Business Process Management & Business Process Management Systems*. [S.l.]: Brasport, 2008. 17, 18, 19

[31] VARVAKIS, G. J. et al. Gerenciamento de processos. apostila da disciplina gerenciamento de processos. *Grupo de Análise de Valor. EPS/UFSC. Florianópolis*, 1999. 18

[32] INAGANTI, S.; BEHARA, G. K. Service identification: BPM and SOA handshake. *BPTrends*, v. 3, p. 1–12, 2007. 19, 20

[33] ROCHA, A. R. C. d.; MALDONADO, J. C.; WEBER, K. C. *Qualidade de software*. [S.l.]: São Paulo: Prentice Hall, 2001. 21, 22

[34] MOREIRA, R. T. Gestão do Conhecimento em Qualidade de Software: Construção de um portal da qualidade de software para o Brasil. 2004. 21

[35] _____. NBR ISO 12207: Tecnologia de informação Processos de ciclo de vida de software. *Rio de Janeiro*, 1998. 22

[36] GUSMÃO, C. M. G. de; MOURA, H. P. de. Gerência de risco em processos de qualidade de software: uma análise comparativa. *Simpósio Brasileiro de Qualidade de Software, III*, 2004. 22

[37] _____. NBR ISO 15504: Tecnologia da informação Avaliação. *Rio de Janeiro*, 2008. 23

[38] VASCONCELOS, A. M. L. de et al. Introdução à engenharia de software e à qualidade de software. 2006. 23

[39] _____. MPS.BR Melhoria de Processo do Software Brasileiro. 23

[40] SILVA, F.; HOENTSCH, S. C.; SILVA, L. Uma análise das metodologias ágeis fdd e scrum sob a perspectiva do modelo de qualidade mps.br. *Scientia Plena*, v. 5, n. 12, 2009. 23

[41] _____. ABNT NBR ISO 31000: Gestão de Riscos: Princípios e Diretrizes. *Rio de Janeiro*, 2009. 24, 25, 28, 36, 38, 75

[42] _____. ABNT NBR ISO Guia 73: Gestão de Riscos: Vocabulário- Recomendações para uso em normas. *Rio de Janeiro*, 2005. 26, 27

[43] _____. ABNT NBR ISO 31010: Gestão de Riscos: Técnicas para o processo de avaliação de riscos. 2009. 27, 29, 30, 38, 39, 40, 75, 78

[44] TRIANTAPHYLLOU, E. *Multi-criteria decision making methods a comparative study*. [S.l.]: Springer, 2000. 28

[45] MIRANDA, L. M. de. *Contribuição a um modelo de análise multicritério para apoio à decisão da escolha do corredor de transporte para escoamento da produção de granéis agrícolas de Mato Grosso*. Tese (Doutorado) — Universidade Federal do Rio de Janeiro, 2008. 28

[46] GOMES, L. F. A. M.; GOMES, C. F. S.; ALMEIDA, A. T. de. *Tomada de decisão gerencial: enfoque multicritério*. [S.l.]: Atlas, 2009. 28

[47] SANTOS, R. F. D.; VIAGI, A. F. Uso do método AHP (Analytic Hierarchy Process) para otimizar a cadeia de suprimentos durante o desenvolvimento integrado de produtos. *Simpósio de Administração da Produção, Logística e Operações Internacionais, XII. FGV/EAESP*, 2009. 29

[48] JÚNIOR, S. B. Controles internos como instrumento de governança corporativa. *Revista do BNDS, Rio de Janeiro*, v. 12, n. 24, p. 149–188, 2005. 29

[49] LOPES, M. C. S. *Mineração de dados textuais utilizando técnicas de clustering para o idioma português*. Tese (Doutorado) — Universidade Federal do Rio de Janeiro, 2004. 30

[50] MOURA, M. F. Proposta de utilização de mineração de textos para seleção, classificação e qualificação de documentos. Campinas: Embrapa Informática Agropecuária., 2004. 30

[51] REZENDE, S. O.; MARCACINI, R. M.; MOURA, M. F. O uso da mineração de textos para extração e organização não supervisionada de conhecimento. *Revista de Sistemas de Informação da FSMA*, n, v. 7, p. 7–21, 2011. 30, 31

[52] WEISS, S. M. et al. *Text mining: predictive methods for analyzing unstructured information*. [S.l.]: Springer Science & Business Media, 2010. 30

[53] FELDMAN, R.; SANGER, J. *The text mining handbook: advanced approaches in analyzing unstructured data*. [S.l.]: Cambridge University Press, 2007. 31

[54] SANTOS, F. L. d. *Mineração de opinião em textos opinativos utilizando algoritmos de classificação*. Dissertação (Mestrado) — Universidade de Brasília, 2014. 31

[55] MANNING, C. D. et al. *Introduction to information retrieval*. [S.l.]: Cambridge university press Cambridge, 2008. v. 1. 32

[56] LAZAREVIC-MCMANUS, N.; RENNO, J.; JONES, G. Performance evaluation in visual surveillance using the F-measure. In: ACM. *Proceedings of the 4th ACM international workshop on Video surveillance and sensor networks*. [S.l.], 2006. p. 45–52. 33

[57] CARLETTA, J. Assessing agreement on classification tasks: the kappa statistic. *Computational linguistics*, MIT Press, v. 22, n. 2, p. 249–254, 1996. 33

[58] GIL, A. C. Como elaborar projetos de pesquisa. *São Paulo*, v. 5, p. 61, 2002. 34

[59] YIN, R. K. *Estudo de Caso: Planejamento e Métodos*. [S.l.]: Bookman editora, 2015. 34

[60] DIEHL, A. A.; TATIM, D. C. *Pesquisa em ciências sociais aplicadas: métodos e técnicas*. [S.l.]: Pearson Brasil, 2004. 35

[61] SELLTIZ, C. *Métodos de pesquisa nas relações sociais*. [S.l.]: EPU, 1974. 35

[62] KRAKHECHE, I. A.; TRABASSO, L. G. *Proposta de um Método para a Implantação da Gestão de Riscos em Conformidade com o Procedimento Técnico PAS 55:2008 - Gestão de Ativos Físicos*. Dissertação (Mestrado) — Instituto Tecnológico de Aeronáutica, 2014. 38

[63] MACHADO, M. J. C. *A Gestão de Riscos como Ferramenta de Apoio a Execução do Planejamento Estratégico da SEFAZ-PI*. Dissertação (Mestrado) — Universidade de Brasília, 2014. 38

[64] CRUZ, M. M. *Proposta de Gerenciamento de Risco Aplicado ao Processo de Registro e Cadastro dos Profissionais de Enfermagem no Sistema Cofen/Coren segundo a Norma ABNT NBR ISO 31000*. Dissertação (Mestrado) — Universidade de Brasília, 2014. 38, 98

[65] SILVEIRA, L. A. O. S.; OLIVEIRA, E. C.; MONTEIRO, S. B. S. Melhoria do processo de implementação de software através da aplicação da ferramenta SIPOC e dos métodos de gestão: BPM e Balanced Scorecard. Universidade de Brasília, WPOS-CIC, 2014. 40

[66] SILVEIRA, L. A. O. S.; OLIVEIRA, E. C. Aplicação do método multicritério AHP na seleção de atividades do processo de implementação de software. *Simpósio de Engenharia de Produção, XXI*, 2014. 40

[67] MARCOVITCH, J. *Tecnologia da informação e estratégia empresarial*. [S.l.]: Ed. da USP, 1996. 42

[68] LEITE, V. de C. Implantação de um projeto de mapeamento de processos para melhoria da qualidade em uma instituição de ensino superior pública. *Tekhne e Logos*, v. 3, n. 1, 2012. 44

[69] SOMMERVILLE, I. *Engenharia de Software.(2007)*. [S.l.]: Addison-Wesley, Reading, 8ª Edição, MA. 45

[70] OLIVEIRA, C. C. d. Implantação sistema de workflow. Universidade Federal do Rio Grande do Sul, 2009. 47

[71] NUNES, V. B.; SOARES, A. O.; FALBO, R. A. Apoio à documentação em um ambiente de desenvolvimento de software. In: *VII Workshop Iberoamericano de Ingeniería de Requisitos y Desarrollo de Ambientes de Software, IDEAS*. [S.l.: s.n.], 2004. p. 50–55. 52

[72] GROPP, B. M. C.; TAVARES, M. d. G. P. Dimensões intangíveis: A relevância do conhecimento tácito em processos de inovação e sustentabilidade. *Conferência Internacional sobre Inovação e Gestão*, n. 6, 2009. 53

[73] MCCONNELL, S. *Code complete: a practical handbook of software construction (Redmond, WA*. [S.l.]: Microsoft Press, 1993. 54

[74] JUNIOR, H. E. Engenharia de software na prática. *São Paulo: Novatec Editora*, 2010. 54

[75] FILHO, W. de P. P. *Engenharia de software*. [S.l.]: LTC, 2003. v. 2. 54

[76] CARVALHO, M. d.; JR, R. R. Fundamentos em gestão de projetos: construindo competências para gerenciar projetos. *São Paulo: Editora Atlas,(3ª edição)*, 2011. 57

[77] GOMES, L. F. A. M. et al. *Tomada de decisões em cenários complexos: introdução aos métodos discretos do apoio multicritério à decisão*. [S.l.]: Thomson, 2004. 58

[78] VARGAS, R. V.; IPMA-B, P. *Utilizando a programação multicritério Analytic Hierarchy Process-AHP para selecionar e priorizar projetos na gestão de portfólio*. [S.l.: s.n.], 2010. 1–22 p. 59

[79] SAATY, T. L. Método de análise hierárquica. Makron Books do Brasil São Paulo, 1991. 60, 61

[80] SCHMOLDT, D. L. et al. The analytic hierarchy process and participatory decision making. In: CITESEER. *Proceedings of the fourth international symposium on advanced technology in natural resources management*. [S.l.], 1995. p. 129–143. 61

[81] MOLINARI, L. Testes funcionais de software. *Ed. Visual Books. Florianópolis*, 2008. 66

[82] BERNARDO, P. C.; KON, F. A importância dos testes automatizados. *Engenharia de Software Magazine*, v. 1, n. 3, p. 54–57, 2008. 66

[83] MYERS, G. J.; SANDLER, C.; BADGETT, T. *The art of software testing*. [S.l.]: John Wiley & Sons, 2011. 68

[84] FJELDSTAD, R. K.; HAMLEN, W. T. Application program maintenance study: Report to our respondents. *Proceedings Guide*, v. 48, 1983. 68

[85] STANDISH, T. A. An essay on software reuse. *Software Engineering, IEEE Transactions on*, IEEE, n. 5, p. 494–497, 1984. 68

[86] DATASUS. Metodologia de gestão de risco do MS/DATASUS. *Technical Report 4.0, Ministério da Saúde - MS*, 2013. 76

[87] DIAS, M. M.; PACHECO, R. C. dos S. Uma metodologia para o desenvolvimento de sistemas de descoberta de conhecimento. *Acta Science Technology Maringá*, v. 27, n. 1, p. 61–72, 2005. 83

[88] VIEIRA, M. H. P. *Aplicação de técnicas de mineração em um programa de concessão de benefícios ao consumidor: o caso do Programa Nota Legal do Distrito Federal*. Dissertação (Mestrado) — Universidade de Brasília, 2014. 84

[89] PONTES, D. P. N.; ARAKAKI, R. Evolução de software baseada em avaliação de arquitetura de software. In: *XVII Congreso Argentino de Ciencias de la Computación*. [S.l.: s.n.], 2011. 85

[90] BHATTACHARYYA, S. Confidence in predictions from random tree ensembles. *Knowledge and information systems*, Springer, v. 35, n. 2, p. 391–410, 2013. 91

[91] TONG, S.; KOLLER, D. Support vector machine active learning with applications to text classification. *The Journal of Machine Learning Research*, JMLR. org, v. 2, p. 45–66, 2002. 91

[92] KALINOWSKI, M. *Introdução à Inspeção de Software*. 2008. 96, 97

[93] SANTOS, R. F. dos. *Gestão de Riscos Aplicada a Processo de Concessão de Financiamento Imobiliário*. Dissertação (Mestrado) — Universidade de Brasília, 2014. 98

[94] JÚNIOR, O. R. da R.; MORAES, R. M. de. Implantação de sistemas ERP em pequenas e médias empresas. *Nucleus*, v. 6, n. 2, 2009. 101

Apêndice A

Mapeamento *AS IS* do processo de desenvolvimento de *software*

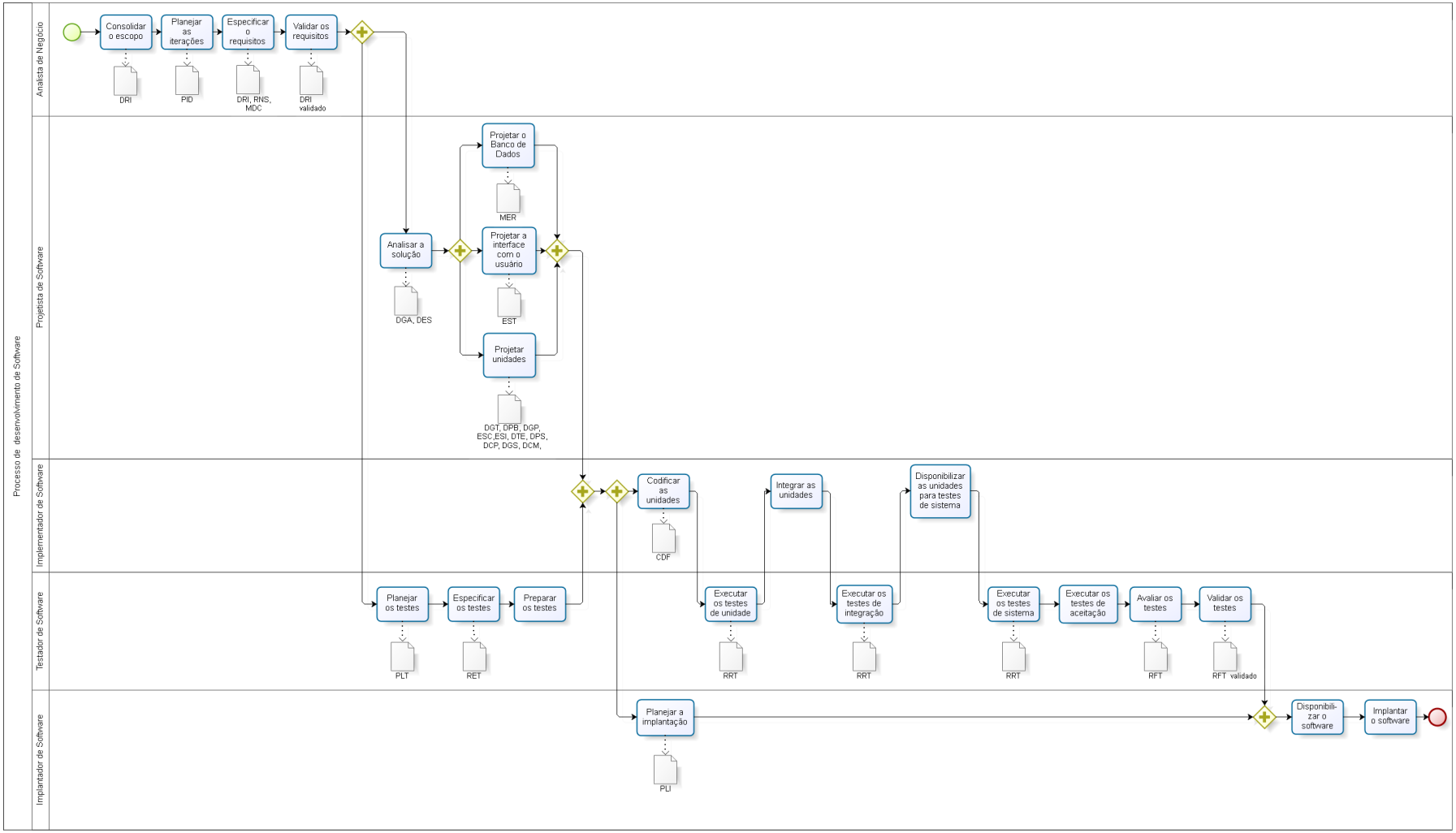

Figura A.1: Mapeamento *AS IS* do processo de desenvolvimento de *software* de uma instituição financeira

Figura A.2: Fonte: Elaboração própria

Apêndice B

Questionário para coleta de dados sobre as atividades do processo de implementação de *software*

O questionário abaixo foi utilizado para a coleta de dados durante entrevista realizada com desenvolvedores do sistema MMO sobre a importância, dificuldade e risco das atividades para o processo de implementação de *software*.

Os dados coletados foram posteriormente transferidos para a ferramenta Expert Choice, que foi utilizada na aplicação da técnica AHP para obtenção dos índices de cada uma das atividades mediante os critérios.

UNIVERSIDADE DE BRASÍLIA – UNB
MESTRADO EM COMPUTAÇÃO APLICADA - PPCA
DEPARTAMENTO DE CIÊNCIA DA COMPUTAÇÃO – CIC

Estudo baseado na norma ABNT NBR ISO 31000:2009 de gestão de riscos, para identificação de riscos no processo de implementação de *software*.

Aluna: Lázara Aline de Oliveira Sousa Silveira
Orientador: Dr. Edgard Costa Oliveira

Formulário para coleta de dados sobre as atividades do processo de implemen-

tação de *software*, aplicados aos desenvolvedores de um sistema de dados de operações do mercado monetário.

Data...: _____
Nome..: _____
Função: _____

Orientações para preenchimento do formulário sobre as atividades do processo de implementação:

a) Seguem anexo três planilhas para julgamento das atividades do processo de implementação mediante os critérios: IMPORTÂNCIA, DIFICULDADE e RISCO.

b) Preencher apenas as células com fundo em azul-claro.

c) A nota deve ser atribuída sempre comparando a atividade da linha em relação a atividade da coluna.

d) Se na avaliação dos critérios, a atividade da linha tem maior relevância em relação a atividade da coluna, a nota deve ser atribuída precedida pelo sinal de – (menos). Em caso contrário, a nota não deve assumir nenhum tipo de sinal.

e) O valor absoluto das notas variam de 1 a 9, sendo que 9 é a maior nota, conforme tabela de Saaty.

Avaliação das atividades pelo critério *Importância*										
Nome da Atividade	**Identificar as unidades**	**Reunir as especificações**	**Identificar os padrões de desenvolvimento**	**Esclarecer as dúvidas em relação à implementação**	**Documentar as informações de histórico na unidade**	**Codificar as unidades**	**Verificar se as unidades corresponde ao resultado esperado na especificação**	**Executar a atividade de Teste de Unidade**	Verificar com base no Diagrama de Transação, se todas as unidades estão codificadas	**Executar a atividade de Teste de Integração**
Identificar as unidades	1									
Reunir as especificações		1								
Identificar os padrões de desenvolvimento			1							
Esclarecer as dúvidas em relação à implementação				1						
Documentar as informações de histórico na unidade					1					
Codificar as unidades						1				
Verificar se as unidades corresponde ao resultado esperado na especificação							1			
Executar a atividade de Teste de Unidade								1		
Verificar com base no Diagrama de Transação, se todas as unidades estão codificadas									1	
Executar a atividade de Teste de Integração										1

Avaliação das atividades pelo critério *Dificuldade*										
Nome da Atividade	**Identificar as unidades**	**Reunir as especificações**	**Identificar os padrões de desenvolvimento**	**Esclarecer as dúvidas em relação à implementação**	**Documentar as informações de histórico na unidade**	**Codificar as unidades**	**Verificar se as unidades corresponde ao resultado esperado na especificação**	**Executar a atividade de Teste de Unidade**	Verificar com base no Diagrama de Transação, se todas as unidades estão codificadas	**Executar a atividade de Teste de Integração**
Identificar as unidades	1									
Reunir as especificações		1								
Identificar os padrões de desenvolvimento			1							
Esclarecer as dúvidas em relação à implementação				1						
Documentar as informações de histórico na unidade					1					
Codificar as unidades						1				
Verificar se as unidades corresponde ao resultado esperado na especificação							1			
Executar a atividade de Teste de Unidade								1		
Verificar com base no Diagrama de Transação, se todas as unidades estão codificadas									1	
Executar a atividade de Teste de Integração										1

Avaliação das atividades pelo critério *Risco*										
Nome da Atividade	**Identificar as unidades**	**Reunir as especificações**	**Identificar os padrões de desenvolvimento**	**Esclarecer as dúvidas em relação à implementação**	**Documentar as informações de histórico na unidade**	**Codificar as unidades**	**Verificar se as unidades corresponde ao resultado esperado na especificação**	**Executar a atividade de Teste de Unidade**	Verificar com base no Diagrama de Transação, se todas as unidades estão codificadas	**Executar a atividade de Teste de Integração**
Identificar as unidades	1									
Reunir as especificações		1								
Identificar os padrões de desenvolvimento			1							
Esclarecer as dúvidas em relação à implementação				1						
Documentar as informações de histórico na unidade					1					
Codificar as unidades						1				
Verificar se as unidades corresponde ao resultado esperado na especificação							1			
Executar a atividade de Teste de Unidade								1		
Verificar com base no Diagrama de Transação, se todas as unidades estão codificadas									1	
Executar a atividade de Teste de Integração										1

Apêndice C

Formulário sobre os artefatos do processo de implementação *software*

Roteiro utilizado para a coleta de dados sobre os artefatos do processo de implementação de *software* durante entrevista realizada com desenvolvedores do sistema MMO.

UNIVERSIDADE DE BRASÍLIA – UNB
MESTRADO EM COMPUTAÇÃO APLICADA - PPCA
DEPARTAMENTO DE CIÊNCIA DA COMPUTAÇÃO – CIC

Estudo baseado na norma ABNT NBR ISO 31000:2009 de gestão de riscos, para identificação de riscos no processo de implementação de *software*.

Aluna: Lázara Aline de Oliveira Sousa Silveira
Orientador: Dr. Edgard Costa Oliveira

Entrevista realizada para coleta de dados sobre os artefatos do processo de implementação de *software*, aplicados aos desenvolvedores de um sistema de dados de operações do mercado monetário.

Contexto: a entrevista teve por objetivo identificar os riscos envolvidos na utilização ou não dos artefatos produzidos durante o processo de desenvolvimento de *software*, para o

processo de implementação. Identificar a importância e impacto do uso dos artefatos para a entrega de um *software* de qualidade para o cliente.

Roteiro para entrevista sobre os artefatos do processo de implementação na identificação de riscos.

Pergunta: *Durante o processo de desenvolvimento de software para alta plataforma (mainframe) são previstos a elaboração de alguns artefatos. Quais você deveria utilizar mas não utiliza? E porquê?*

Comentários: _____

_____

_____

Pergunta: *Os artefatos são fonte de consulta atualizada e confiável sobre as funcionalidades do sistema?*

Comentários: _____

_____

_____

Pergunta: *Existe documentação de sistema acessível tanto aos desenvolvedores (área técnica) quanto aos gestores (a área de negócio) sobre as funcionalidades e funcionamento do sistema ?*

Comentários: _____

_____

_____

Pergunta: *Qual o artefato do processo de implementação que você considera mais importante, e o que poderia ser feito para que ele apresentasse melhores resultados?*

Comentários: _____

_____

_____

Pergunta: *Você conhece ou já ouviu falar sobre gerenciamento de processos de negócio (BPM)? Você sabe que o banco possui ferramenta disponível para mapeamento de processos?*

Comentários: _____

_____

_____

Apêndice D

Formulário para coleta de dados sobre os termos da linguagem de programação Cobol

Roteiro utilizado para a coleta de dados com desenvolvedores do sistema MMO, sobre os termos que poderiam ser utilizados para identificação de complexidade nos códigos-fonte escritos com a linguagem de programação *Cobol*.

UNIVERSIDADE DE BRASÍLIA – UNB
MESTRADO EM COMPUTAÇÃO APLICADA - PPCA
DEPARTAMENTO DE CIÊNCIA DA COMPUTAÇÃO – CIC

Estudo baseado na norma ABNT NBR ISO 31000:2009 de gestão de riscos, para tratamento de riscos identificados no processo de implementação de *software*.

Aluna: Lázara Aline de Oliveira Sousa Silveira
Orientador: Dr. Edgard Costa Oliveira

Contexto: A entrevista teve por objetivo coletar informações a partir da experiência dos agilidade e segurança nas atividades de manutenção.

Roteiro para entrevista sobre os artefatos do processo de implementação na identificação de riscos.

Pergunta: *Quais termos da linguagem de programação Cobol seriam capazes de identificar se um programa é ou não responsável por tratar de regras de negócio do sistema?*

Comentários: _____

_____

Pergunta: *Quais termos da linguagem de programação Cobol seriam capazes de identificar os programas que tratam das atualizações das bases de dados (CRUD - Create, Read, Update e Delete)?*

Comentários: _____

_____

Apêndice E

Manual de Modelagem de Sistemas

Objetivo

Estabelecer padrões para a modelagem de desenvolvimento de aplicativos a serem utilizados para elaboração de artefatos de especificação de componentes por meio de mapeamento de processos, bem como a confecção e manutenção de *documentação de sistema*.

Conceito de modelagem de processos

Modelar um processo significa representar graficamente a realização das atividades do processo, registrando por meio de símbolos predefinidos, todos os elementos que fazem parte do processo tais como: entradas/insumos, executores, tarefas, saídas/produtos, dentre outros.

Ambiente

O modelo será desenvolvido utilizando-se o Bizagi Modeler 2.9.0.4 que é uma ferramenta gratuita de Business Process Model and Notation (BPMN) que permite desenhar, documentar processos, criar fluxogramas e mapas mentais.

E.1 Elementos básicos de modelagem de processos

Os elementos de modelagem são os principais objetos gráficos que representam o processo. Existem três tipos de elementos de fluxo: atividades, gateways e eventos. Os elementos devem ser dispostos em linha horizontal, da esquerda para a direita, na sequência lógica em que ocorrem durante o fluxo do processo.

As atividades representam um trabalho realizado dentro do processo. Elas podem ser do tipo processos, subprocessos ou tarefas. Para que o fluxo do processo seja objetivo, de

fácil entendimento e conciso, o nome das atividades devem começar com letra maiúscula e conter apenas um verbo no infinitivo acompanhado de complemento. Exemplo: Impostar dados; Conferir dados.

O nome da atividade deve ser o mais curto possível sem perda do significado.

E.1.1 Processo

O processo representa um conjunto de subprocessos e/ou tarefas que estão organizados para desempenhar uma ação específica num ponto do processo a ser desenhado.

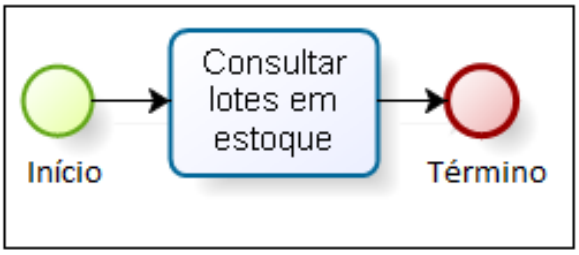

Figura E.1: Elementos de início e fim de um processo.

Os processos devem, obrigatoriamente, conter elemento de início e fim, contendo os seus respectivos rótulos, conforme apresentado a seguir na Figura E.1. Não é necessário descrever no rótulo o evento, gatilho ou trigger que inicia o processo a ser mapeado.

E.1.2 Subprocesso

São processos que são utilizados como partes menores de outro processo. Deve ser utilizado como uma forma de organizar a execução das tarefas dentro de um processo, não podendo ser reaproveitado e/ou rastreado.

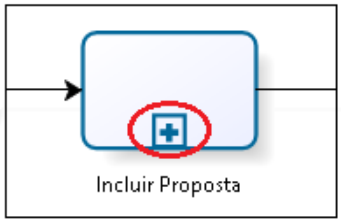

Figura E.2: Subprocesso.

Graficamente, devem ser representados na cor de apresentação padrão, conforme Figura E.2 abaixo:

E.1.3 Subprocesso Reutilizável

Um processo, quando utilizado dentro de outro processo, passa a ser considerado um *Processo reutilizável*, deve ser representado no Bizagi Modeler pelo elemento subprocesso reutilizável conforme apresentado na Figura E.3.

Observe que a borda deste elemento é ressaltada quando comparada com o elemento subprocesso. Para definir o elemento de processo reutilizável, selecione e arraste o elemento de tarefa para a área do processo. Clique com o botão direito e selecione o item transformar em subprocesso, e posteriormente selecione o item subprocesso reutilizável.

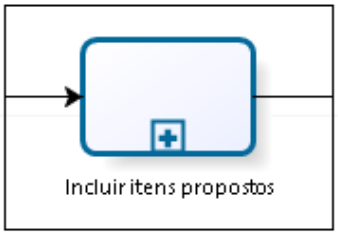

Figura E.3: Subprocesso reutilizável.

E.1.4 Subprocessos de repetição

Os subprocessos de repetição (loop) são muito utilizados nas definições de programas e sub-rotinas, pois ele permite aplicar um mesmo tratamento repetidas vezes, a vários dados selecionados nas aplicações.

Graficamente, são identificados por um subprocesso adicionado de um ícone, conforme abaixo:

Figura E.4: Elemento subprocesso de repetição.

Internamente ao subprocesso de repetição, devem ser descritos todas as tarefas que deverão ser realizadas repetidamente até que uma condição de fim seja atendida.

E.1.5 Tarefa

As tarefas estão presentes em processos e subprocessos representando as atividades que não podem ser decompostas. Para sua nomenclatura, não devem ser utilizadas formas negativas, como por exemplo: Não assinar documento.

Tarefa de Usuário

As tarefas de usuário devem ser utilizadas para anotar os momentos de interação entre o usuário e o sistema. Graficamente, são identificadas pelo ícone destacado, conforme Figura E.5 abaixo:

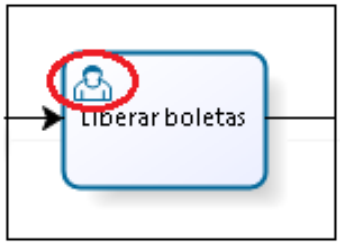

Figura E.5: Tarefa de usuário.

Tarefa de sistema

As tarefas de sistema deverão ser utilizadas para documentar acessos a sub-rotinas do próprio sistema que possuem caracterítica de reutilização ou procedimento importantes feitos pelo módulos.

Graficamente, são identificadas pelo ícone destacado, conforme Figura E.6 abaixo:

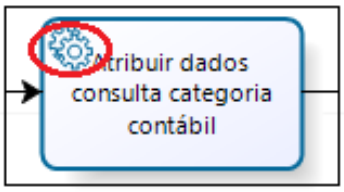

Figura E.6: Tarefa de sistema.

Quando há a necessidade de modelar o comportamento de uma sub-rotina, significa que ela não está com o nível de atomicidade apropriado. Nesse caso, essa tarefa de sistema deverá ser documentada como um processo reutilizável, e descritos passo a passo os eventos importantes.

Tarefa de sistema para descrição de variáveis de entrada de uma sub-rotina

Deve-se utilizar a tarefa de sistema para descrever orientações quanto a leitura de arquivos sequenciais, os campos/parâmetros de entrada utilizados no acionamento de sub-rotinas, bem como as variáveis de saída (retorno), com as mensagems de erro ou tratamento de exceção. Esta descrição deve estar localizada no campo descrição (aba "Básico"no item Propriedades da Tarefa de sistema). A documentação deve seguir o exemplo abaixo:

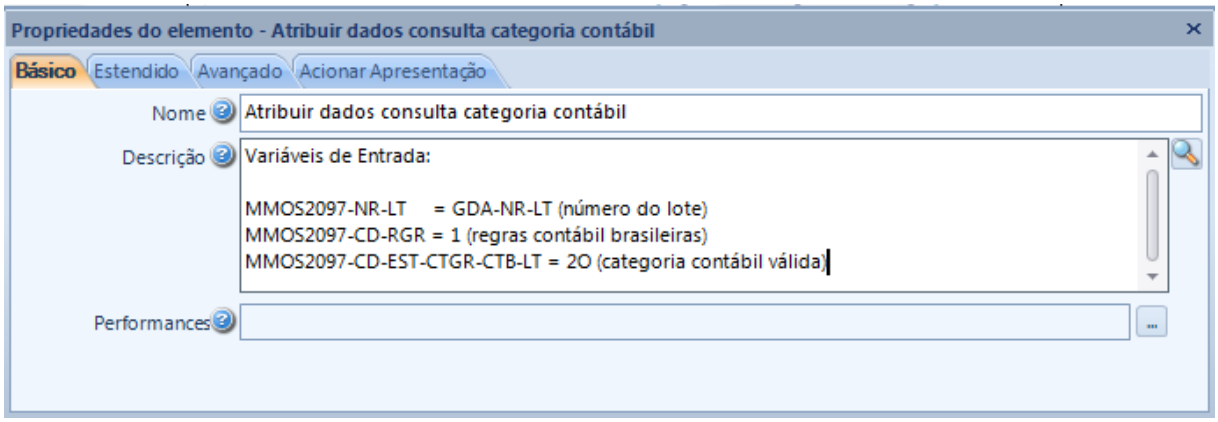

Figura E.7: Descrição de variáveis de sub-rotinas no elemento Tarefa de sistema.

Todas as variáveis de entrada, saída e retorno devem estar documentadas e descritas no elemento Tarefa de sistema. Este é um procedimento importante, pois na situação de reuso da funcionalidade por outras aplicações, a informação sobre o acionamento da rotina estará disponível, bem como a descrição de cada uma das variáveis, seja as de entrada, saída, ou as de retorno (para tratamento de erros, ou situações de exceção)

Portanto deve-se documentar todas as entradas necessárias, uma vez que o comportamento da tarefa de sistema ou do processo reutilizável pode depender das entradas fornecidas.

Tarefa de sistema para descrição de instrução SQL

Uma tarefa de sistema também pode ser utilizada para representar outros procedimentos, tais como as querys executadas pela sub-rotina e também o tratamento de arrays para os casos em que a consulta retornar mais de um registro.

Deve-se documentar esses procedimentos da mesma forma que a utilizada para descrição das variáveis (aba "Básico"no item Propriedades do elemento Tarefa de sistema), conforme abaixo:

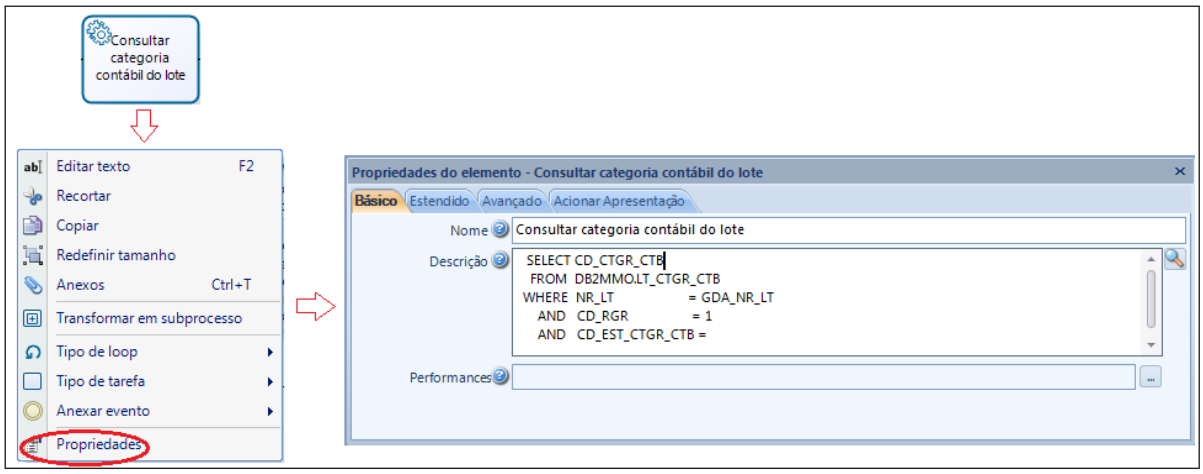

Figura E.8: Instrução SQL de sub-rotinas no elemento Tarefa de sistema.

E.1.6 Decisão (Gateway)

Os gateways são utilizados para representar as tomadas de decisão dentro de um processo. Utilizaremos dois tipos: exclusivo e paralelo.

Figura E.9: Tipos de Gateway.

Gateway Exclusivo

O gateway exclusivo indica que o processo deve seguir um só caminho baseado na resposta da respectiva condição. Seguem abaixo algumas características:

- É utilizado para perguntas do tipo "sim/não" ou para indicar dois ou mais fluxos alternativos. Deverá ser explicitado o critério de decisão em cada caminho a ser seguido.
- Deve ser utilizado como ponto de retorno de dois ou mais fluxos em um único caminho.

Graficamente, deve ser utilizado conforme exemplos abaixo:

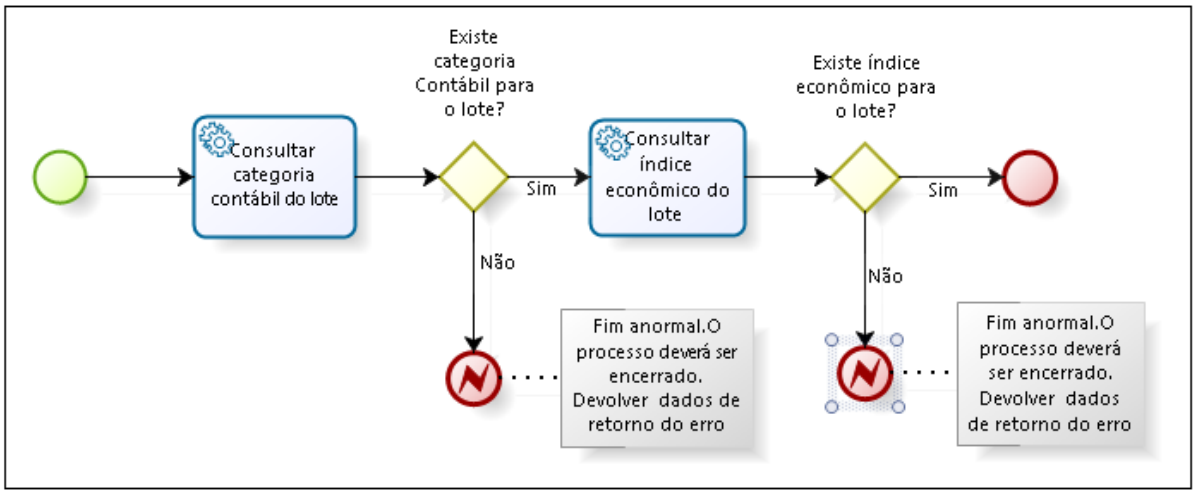

Figura E.10: Elemento de decisão do tipo exclusivo.

Gateway Paralelo

O gateway paralelo indica que duas ou mais tarefas ocorrem em paralelo, obrigatoriamente, ou para indicar o ponto de retorno para sincronizar dois ou mais fluxos de caminhos criados em paralelo.

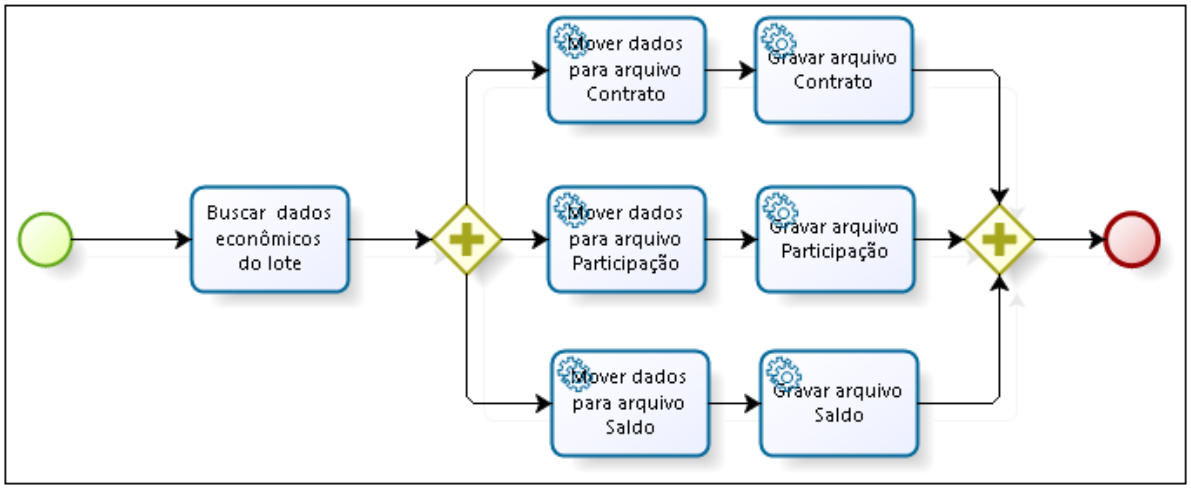

Figura E.11: Elemento de decisão do tipo paralelo.

E.1.7 Conector

Os conectores são responsáveis por fazer a ligação entre os outros elementos do processo.

- As atividades devem originar uma única conexão;

- Os gateways podem receber ou originar “N” conexões.

Deve-se manter o processo simples e limpo, dispondo elementos de forma a evitar o cruzamento de conexões.

E.1.8 Anotação

As anotações representam notas ou comentários que podem ser inseridos em qualquer ponto do processo. Elas têm o objetivo de clarear ou explicitar alguma informação relacionado ao elemento ao qual a anotação está vinculada.

Ele pode, por exemplo, ser utilizado para explicar um determinado comportamento dos programas ou sub-rotinas, nas situações de exceção ou de erro nas aplicações.

Figura E.12: Comentário.

E.1.9 Raia

Devem ser utilizadas para organizar e categorizar as atividades dentro de um processo, com o objetivo de dar uma maior clareza na sua visualização. Servem para separar ações de intervenientes, que podem ser usuários ou sistemas.

E.2 Padronização

Visando à padronização e governança dos componentes, algumas regras deverão ser seguidas dependendo do elemento utilizado. Seguem abaixo algumas regras a serem seguidas ao nomear um novo processo ou tarefa:

- Deverá ter no máximo 64 caracteres;
- Se o processo especificado tratar-se de um programa ou sub-rotina, deverá conter o nome do módulo seguido do seu objetivo, separados por hífen “ - ”;

E.2.1 Padronização de termos para modelagem de sistemas

Seguem abaixo alguns verbos a serem utilizados na modelagem de sistemas:

- Alterar: na modificação de um dado via sub-rotina;
- Atribuir: movimentações de informações das variáveis para sub-rotina;
- Atualizar: modificação de um dado via query direta;
- Consultar: obtenção de um dado via query direta;
- Deletar: apagar um dado via query direta;
- Excluir: apagar um dado via sub-rotina;
- Gravar: persistir dados em um arquivo;
- Incluir: adicionar um dado a base de dados via sub-rotina;
- Inserir: adicionar um dado a base de dados via query direta;
- Listar: obtenção de mais de um dado via sub-rotina;
- Mover: movimentações de informações de variáveis para arquivos;
- Obter: obtenção de um dado via sub-rotina;
- Tratar: utilizar em subprocessos de repetição que simulam a situação *do-while*;
- Gerar: produzir, fornecer informação para a própria aplicação, ou para outras;
- Ler: processo de leitura de informações contidas em arquivos sequenciais;
- Definir: definir valores para um conjunto de variáveis a partir da análise de outras informações;
- Exibir - mostrar valores contidos em variáveis, mostrar resumos de processamentos de programas (quantidade de registros processados);
- Comparar - utilizado principalmente para identificar processos de balance line onde as informações dos arquivos depois de ordenados ordenados por campos chave, são comparados para uma tomada de decisão;

Apêndice F

Mapeamento detalhado de especificação do programa MMOP0622

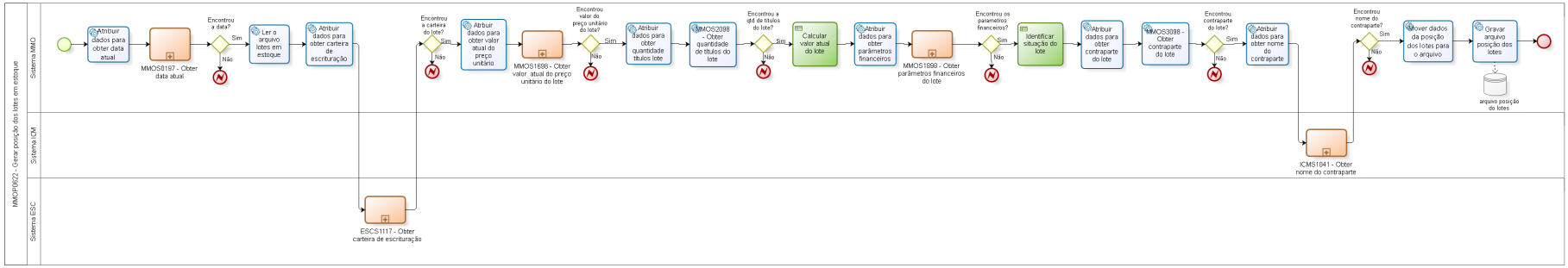

Figura F.1: Modelo de Nível Detalhado do programa: MMOP0622 - Gerar informações de lotes em estoque do sistema MMO

Fonte: Elaboração própria

Apêndice G

Mapeamento *TO BE* do processo de desenvolvimento de *software*

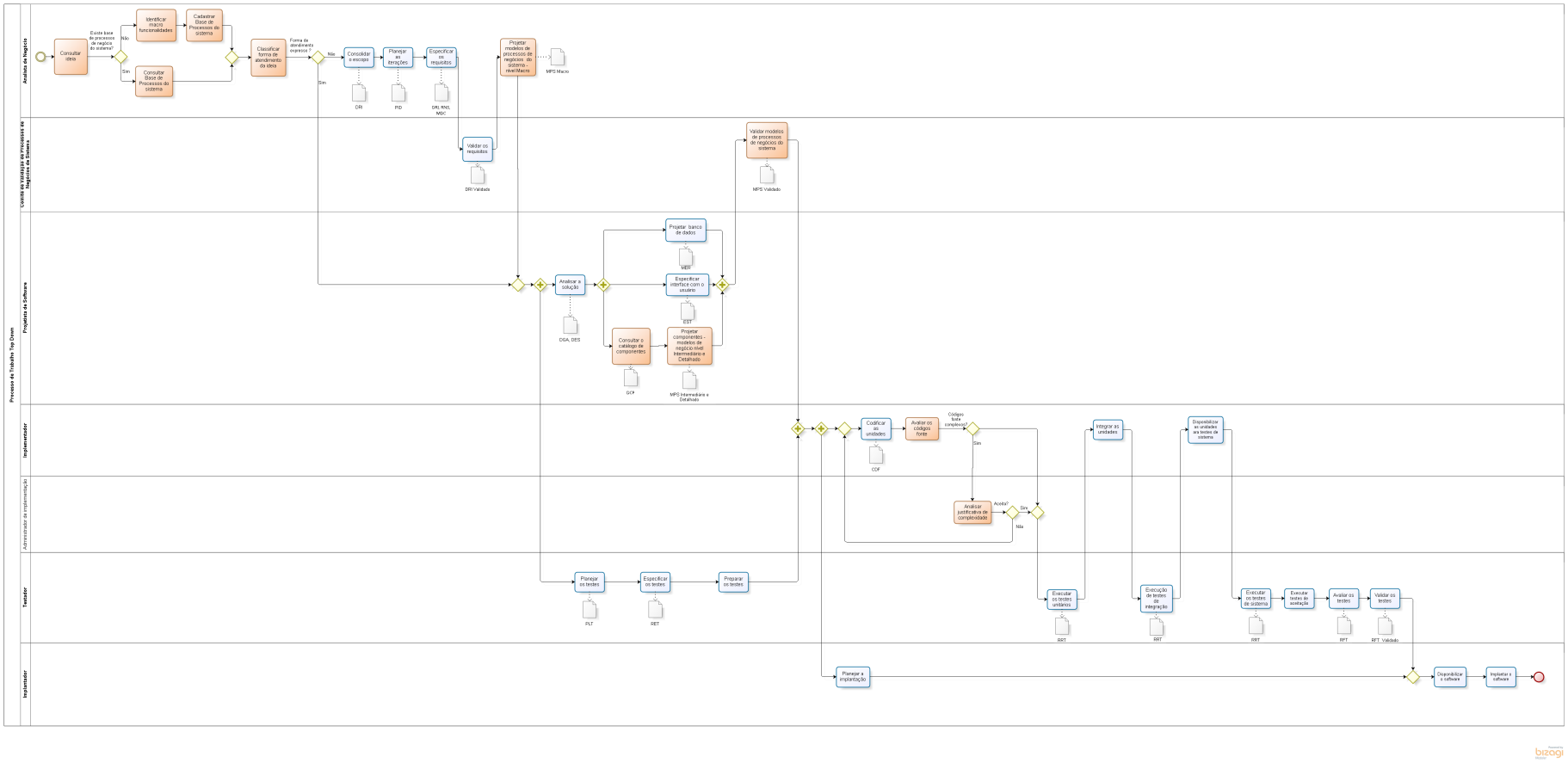

Figura G.1: Mapeamento *TO BE* do processo de desenvolvimento de *software* de uma instituição financeira

Fonte: Elaboração própria