UNIVERSIDADE FEDERAL DO RIO GRANDE DO SUL
INSTITUTO DE INFORMÁTICA
CURSO DE CIÊNCIA DA COMPUTAÇÃO

FILIPE JONER

Modelos de Processos de Negócio: Um Estudo de Caso para Extração de Casos de Teste

Monografia apresentada como requisito parcial para a obtenção do grau de Bacharel em Ciência da Computação

Orientador: Profa. Dra. Lucineia Heloisa Thom

Porto Alegre
2018

“If no one is in charge,
then I guess you are.”
— PENNY WELLER’S MOTHER

AGRADECIMENTOS

Agradeço a minha mãe e a meu pai pela paciência, carinho e amor que tiveram comigo. Com todos os sacrifícios necessários ou momentos difíceis pelos quais passaram, sempre me proveram o meio necessário para que eu continuasse meus estudos.

A meu irmão mais velho Guilherme Joner que sempre esteve ao meu lado e cujos conselhos me fizeram um estudante, um profissional e um irmão melhor.

À minha irmã mais nova Laura Joner, cujo sorriso e palavras sempre me fizeram sorrir não importando a situação em que eu estivesse.

À minha namorada Bruna, que esteve comigo durante minha trajetória acadêmica e sempre me apoiou de forma única, não só com seu amor, mas também com palavras não tão amigáveis quando necessitei.

A meu padrasto Hélio (*in memoriam*), um dos meus melhores amigos e em muitas vezes um pai, que por seus conselhos tomei o caminho para ser hoje um cientista da computação.

Às amizades que construí durante minha vida, em especial a colegas como Guilherme Ribeiro e Matheus Bica, que me motivaram e ajudaram em momentos difíceis da Universidade.

À minha professora e orientadora Profa. Dra. Lucineia Heloisa Thom que, com sua experiência, paciência e atenção, conduziu este trabalho e disciplinas importantes que me auxliam nos dias de hoje no exercício de minha profissão.

Aos meus gerentes, em especial Frank Vianna e Guilherme Longoni, e meus colegas de trabalho que me proveram o ambiente e condições necessárias, para que eu pudesse completar meus estudos e realizar este trabalho.

À minha mentora Catherine Heldt, cujas duras palavras me auxiliaram a escrever este trabalho.

RESUMO

A transparência entre gerenciamento de processos de negócio e equipe de desenvolvimento de software se faz cada vez mais necessária no ambiente corporativo, para que se entenda exatamente o propósito e funcionalidades que precisam ser desenvolvidas nos sistemas aos usuários finais. Para isso, o uso da modelagem de processos de negócio é importante, tanto para grandes empresas para avaliação do uso de seus recursos e governanças gerenciais, quanto para pequenas empresas que pretendem implementar um melhor modelo de negócio. A partir disso, uma modelagem de processos de negócio bem detalhada se torna um ótimo insumo para análise de requisitos de software e, por consequência, para extração de casos de teste. Este trabalho propõe, portanto, um estudo de caso acerca das regras de mapeamento para extração de casos de teste a partir de um modelo de processo de negócio propostas por Sousa et al. (2014) ainda nas fases iniciais do desenvolvimento de uma aplicação.

Palavras-chave: Gerenciamento de processos de negócio. BPM. BPMN. modelagem de processos de negócio. casos de teste.

ABSTRACT

Transparency between business process management and software development teams becomes increasingly necessary in the corporate environment to understand exactly the purpose and end users functionalities for which a system needs to be developed. For this goal the utilization of business process modeling is important both for large companies to evaluate the use of their resources and management governance and for small companies that want to impprove their business models. An well detailed business process modeling becomes a great input for software analsys and requirements consequently for test cases extraction. Therefore this paper proposes a case study for the mapping rules for test cases extraction from a business process model proposed by Sousa et al. (2014) that can be used in the early stages of development of an application.

Keywords: Business process, business process management, business process modeling, BPMN, test case.

LISTA DE FIGURAS

LISTA DE TABELAS

LISTA DE ABREVIATURAS E SIGLAS

BDD	*Behavior-Driven Development*
BPEL	*Business Process Execution Language*
BPM	*Business Process Management*
BPM	*Business Process Model*
BPMS	*Business Process Management System*
BPMN	*Business Process Modeling Notation*
BPSS	*Business Performance System Solution*
ebXML	*e-business XML*
EPC	*Event-Process Chains*
IDEF	*Icon DEFinition for Function Modeling*
LOVeM	*Line of Visibility Enterprise Modeling*
KPI	*Key Performance Indicators*
OASIS	*Organization for the Advancement of Structured Information Standards*
OMG	*Object Management Group*
RCPN	*Rede de Cenário de Processos de Negócio*
RCT	*Rede de Casos de Teste*
RPC	*Rede de Petri Colorida*
UI	*User Interface*
UML	*Unified Modeling Language*

SUMÁRIO

1 INTRODUÇÃO

Processos de negócio vêm sendo cada vez mais importantes às organizações, devido ao seu potencial de aumentar significativamente a produtividade e reduzir custos (AALST; HOFSTEDE; WESKE, 2003). Um processo de negócio é uma série de atividades relacionadas entre si executadas por seus respectivos atores, para atingir um determinado objetivo de negócio. A disciplina encarregada pelo mapeamento desses processos é conhecida como gerenciamento de processos de negócio.

Um modelo de processo de negócio é um documento, seja textual, seja gráfico, que objetiva descrever essas atividades, atores e relacionamentos para um determinado público de uma organização. Graficamente, um modelo de processo pode ser representado através da Notação e Modelo de Processos de Negócio (*Business Process Modeling Notation - BPMN*), um padrão da Organização Internacional para Padronização (*International Organization for Standadirzation - ISO*) amplamente utilizado por ferramentas de mercado (DUMAS et al., 2013). Além de ser uma notação de fácil compreensão, por possuir uma tradução do modelo a código na Linguagem Extensiva de Marcação (*Extensible Markup Language - XML*), modelos de processos de negócio descritos em BPMN são comumentemente utilizados para integração de processos a aplicações e desenvolvimento de Sistemas de Gerenciamento de Processos de Negócio *(Business Process Management Systems - BPMS)*.

Considerando-se o desenvolvimento de sistemas, teste de software é um importante processo no ciclo de vida de desenvolvimento de software para verificar sua corretude referente às necessidades do usuário, a eficácia e confiabilidade, e encontrar potenciais defeitos antes da entrega da aplicação aos usuários finais (YOTYAWILAI; SUWANNASART, 2014). Assim sendo, conforme Cai (2011), requerimentos de teste mais detalhados são cada vez mais solicitados pelas organizações .

Um modelo de processo de negócio em BPMN pode indicar quais atividades de uma organização são executadas por sistemas e quais são manuais. Ademais, a notação possui um vasto conjunto de elementos que possibilita a visualização de requisitos, tanto funcionais, quanto técnicos, de um processo.

Um modelo de processo de negócio (também denominado no presente texto modelo de processo) na notação BPMN torna-se, portanto, um candidato natural a material de insumo no processo de desenvolvimento de software.

Diversos autores já pesquisam o uso prático desses modelos de processos na área

de desenvolvimento de sistemas. Em (LOUMOS et al., 2010) é proposto um desenvolvimento de software orientado a modelos de processos, onde o código escrito de um sistema é baseado nas atividades executadas em um processo. Em (LüBKE; LESSEN, 2016), casos de teste são escritos na notação BPMN, uma vez que os autores expõem que cada vez mais recursos funcionais de organizações são responsáveis pela verificação do correto funcionamento do sistema, de modo que se faz necessário um caso de teste escrito em um formato que essas pessoas consigam facilmente compreendê-lo.

Na área de teste de software, diversos trabalhos ((MOURA et al., 2017), (YOTYAWILAI; SUWANNASART, 2014), (NONCHOT; SUWANNASART, 2016)) focam na obtenção de testes automatizados a partir de modelos em BPMN. Em ((SOUSA et al., 2014)), porém, são apresentadas regras de mapeamento de um modelo de processo em BPMN para casos de testes manuais.

1.1 Objetivo

O objetivo deste trabalho é apresentar um estudo de caso a partir do conjunto de regras propostas por (SOUSA et al., 2014) para mapeamento de modelos de processos em BPMN a casos de testes. As regras de (SOUSA et al., 2014) foram escolhidas nesse estudo por apresentarem uma proposta de um passo-a-passo desse mapeamento e por possibilitarem a extração de casos de teste funcionais do modelo de processo.

O estudo de caso foi realizado com 16 funcionários, 8 de áreas técnicas e 8 de áreas não-técnicas, da empresa alemã de desenvolvimento de *software* SAP, localizada no município de São Leopoldo no Estado do Rio Grande do Sul.

O presente trabalho visa a verificação da hipótese de que com a utilização das regras de (SOUSA et al., 2014) um número maior de atividades do modelo de processo em BPMN é coberto por casos de teste. Ademais, o estudo de caso tem o propósito de averiguar se as regras são compreendidas por pessoas, tanto de áreas funcionais, quanto de áreas relacionadas à TI.

1.2 Principais contribuições

O estudo de caso investiga a identificação de casos de teste às atividades dos modelos de processos de uma organização. As regras de mapeamento devem ser de fácil

compreensão, não só por pessoas de áreas de TI, mas também por estudantes e conhecedores de áreas funcionais, visto que eles estão cada vez mais presentes na área de teste de software (LüBKE; LESSEN, 2016). Essas pessoas poderão, então, auxiliar não só na execução, mas também na descrição de casos de teste, apoiando as equipes de desenvolvimento de sistemas no mapeamento de requisitos funcionais das aplicações.

No presente trabalho, um experimento controlado foi conduzido, para investigar se com o uso das regras propostas por (SOUSA et al., 2014) haveria uma maior cobertura de casos de teste às atividades de um modelo de processo em BPMN. Foi possível perceber a partir desse experimento que o número de atividades cobertas por casos testes foi superior com o uso de regras de mapeamento propostas em (SOUSA et al., 2014). Para todos os oito participantes funcionais, o número de atividades de modelo de processo cobertas por casos de teste aumentou.

1.3 Estrutura do trabalho

Este trabalho está estruturado da seguinte maneira: o capítulo 2 trata de fundamentos de processo de negócio, seu gerenciamento, modelagem e introdução à notação BPMN. O capítulo 3 aborda fundamentos de teste de software e um padrão de modelo de documento de caso de teste a ser seguido na especificação (STANDARD, 2008). O capítulo 4 apresenta um estudo das propostas de regras de mapeamento de modelos em BPMN a documentos de casos de teste por (SOUSA et al., 2014), e por sua vez o capítulo 5 aborda a metodologia e formas de coleta de resultado, para validação das regras propostas. O capítulo 6 trata os trabalhos relacionados. Por fim, o capítulo 7 aborda as conclusões do trabalho e sugestões para estudos futuros.

2 FUNDAMENTOS DE PROCESSOS DE NEGÓCIO

Este capítulo apresenta os principais fundamentos de gerenciamento de processos *(Business Process Management - BPM)*, os quais são utilizados ao longo do texto. Além da apresentação dos fundamentos de BPM, esse capítulo abordará a importância da área de BPM para as organizações e como esta área proveniente da administração se torna cada vez mais importante para a computação.

2.1 Conceitos e Gerenciamento de Processos de Negócio

Em 1776 que o filósofo e economista britânico Adam Smith em sua obra *Riqueza das Nações* descreveu a divisão de trabalho a partir de atividades consecutivas executadas por pessoas específicas para atingir um determinado objetivo, nesse caso, a criação de um alfinete. Ainda que Adam Smith não tenha feito referência direta a processos de negócio, intui-se que o objetivo de melhorar o processo, seja de fabricação, seja de processo de negócio não é algo absolutamente novo.

Conforme Dumas et al. (2013), diferentemente de produtos, serviços, marca, ou bens físicos e monetários, a importância de processos de negócio não foi apreciada por muito tempo segundo, porém vem recebendo considerável atenção nos anos recentes devido ao seu potencial para aumentar significantemente a produtividade e reduzir custos (AALST, 2013).

Processos de negócio são atualmente foco de organizações, pois podem ser considerados um dos meios pelo qual as organizações atingem seus objetivos, seja em prestação de serviços, seja na manufatura de um produto. Segundo Weske (2007), um processo de negócio consiste em um conjunto de quaisquer atividades realizadas coordenadamente para atingir um objetivo de negócio.

Dessa forma processo de negócio pode ser definido como atividades que podem ser, então, divididas em sub-atividades. Para cada atividade, podem haver dados de entrada que são transformados em dados de saída. A relação entre dados de entrada e saída, consiste, portanto, na sequência em que as atividades são executadas (AAGESEN; KROGSTIE, 2015).

São esses processos que determinam tarefas e responsabilidades de cada funcionário, além de impactar diretamente a qualidade de produtos e serviços percebidos pelo mercado (DUMAS et al., 2013). Por isso, esses processos de negócio recebem cada vez

mais atenção tanto pela comunidade de administração e negócios quanto pela comunidade da computação (WESKE, 2007).

Voltando a Adam Smith, o processo de negócio da fábrica de alfinetes poderia ser descrito da seguinte maneira. Seguindo Weske (2007), o conjunto de atividades seria cortar fio, afiar o fio, selar a cabeça do alfinete e por fim soldar a cabeça do alfinete ao fio. Os operários seriam especializados nessas atividades, logo os especialistas de cada etapa do processo realizariam suas respectivas atividades, de tal forma que cada ator teria seu papel no processo, conforme o conceito apresentado por Dumas et al. (2013). Essas atividades realizadas pelos seus devidos atores resultariam, portanto, no objetivo final do processo, o alfinete.

Do ponto de vista da administração, processos de negócio interessam para a melhoria das operações das empresas, aumento da satisfação dos clientes, redução dos gastos do negócio e estabelecimento de novos serviços a baixos custos (WESKE, 2007). Para as comunidades da ciência da computação o tópico interessa a duas em específico, segundo Weske (2007). A comunidade de pesquisadores com experiência em métodos formais investigam a propriedades estruturais dos processos, e para a comunidade de desenvolvedores o interesse está em prover robustez e escalabilidade a sistemas de *software* (WESKE, 2007), uma vez que esses processos usualmente envolvem interação entre trabalho, pessoas e tecnologia (PAGNACOS, 2012).

Segundo Dumas et al. (2013), são esses processos que determinam o potencial das organizações em se adaptarem a novas circuntâncias e a cumprirem cada dia o maior número de requerimentos legislativos. Logo, para reagir a essas alterações frequentes em uma forma mais eficiente, as empresas necessitam primeiramente adaptar seus processos de negócio (YUAN et al., 2008), pois enquanto organizações possuírem um framework simples que identifique pessoas, processos, governanças e tecnologia, então sempre terão a habilidade para se adaptarem e melhorar a eficiência, apesar de quaisquer condições externas que enfrentem (PAGNACOS, 2012).

Atualmente, organizações determinam esforços para padronizar seus processos de negócio, de modo a ganhar em melhorias de desempenho de acordo com Schafermeyer, Grgecic and Rosenkranz (2010). Dados integrados e processos padronizados acarreta na redução de custos e tomada de decisões em níveis gerenciais (SCHAFERMEYER; GRGECIC; ROSENKRANZ, 2010).

Conforme Schafermeyer, Grgecic and Rosenkranz (2010), maior nível de qualidade nos processos de empresas pode ser alcançado a partir de padronizações e alterações

bem sucedidas desses. Devido à grande variedade de atributos e às diferentes complexidades que caracterizam um processo de negócio, é necessário uma disciplina de estudo específica que facilite essas modificações nesse processo.

Por exemplo, alguns processos tendem a ser altamente criativos e singulares a empresas, enquanto outros são completamente padronizados e automatizados (SCHAFERMEYER; GRGECIC; ROSENKRANZ, 2010).

Neste contexto, BPM auxilia na padronização e alteração de processos de negócio. Conforme Dumas et al. (2013), BPM é a ciência de orientar como o trabalho é realizado em uma organização, para assegurar resultados consistentes e se beneficiar as constantes oportunidades de desenvolvimento.

O BPM consiste, portanto, no desenho e na identificação dos processos de uma organização, na análise da eficiência desses por meio da mensuração de resultados obtidos em um determinado período de tempo e em suas otimizações se possíveis (PAGNACOS, 2012).

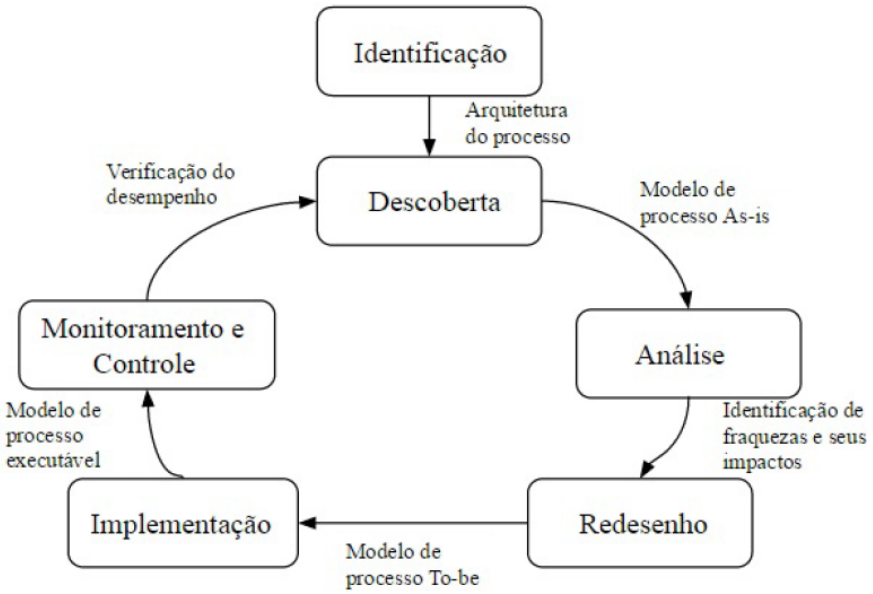
BPM cria, dessa forma, uma inteligência de negócio em tempo real, para auxiliar organizações a responderem rapidamente a mudanças de mercado (PAGNACOS, 2012), e busca, segundo Lindsay, Downs and Lunn (2003), a melhor compreensão em mecanismos chave do negócio, para desenvolver, e, se necessário modificar radicalmente, a forma como o processo é desempenhado a partir da identificação de oportunidades de mercado. Por fim, BPM busca uma maior eficiência especialmente em áreas de negócios em que a tecnologia pode suportar os processos de negócio de uma organização (LINDSAY; DOWNS; LUNN, 2003).

2.1.1 Ciclo de vida de BPM

Em geral, a primeira dúvida que uma equipe possui ao iniciar seu gerenciamento de processos de negócio é "quais processos temos a intenção de melhorar?"(DUMAS et al., 2013). Comumentemente, BPM não é aplicado desde o zero, ou seja, sem que haja algum processo já instaurado na empresa.

Assim, o ciclo de vida do BPM costuma ser descrito em seis fases conforme apresentado na figura 2.1 (DUMAS et al., 2013): identificação, descoberta (ou modelagem), análise, redesenho, implementação e monitoramento. Outros autores como WESKE propõe fases com objetivos semelhantes, porém nomenclaturas diferentes (2007). Neste trabalho, entretanto, é utilizado o ciclo de vida proposto por Dumas et al. (2013).

Figura 2.1: Ciclo de vida de BPM conforme Dumas et al. (2013)



Fonte: (DUMAS et al., 2013)

Identificação do processo: O resultado desta fase *arquitetura de processo*, uma visão geral dos processos de uma organização. É na fase de identificação que um problema de negócio é identificado, e processos relevantes a esse são delimitados e relaciondaos.

Descoberta do processo (modelagem do processo as-is): Após serem identificados quais processos deverão ser modelados,tem-se a fase de descoberta de processos. Essa fase resulta na documentação dos processos relevantes em um ou mais modelos na forma *as-is*, ou seja, como são atualmente exeutados na organização. É importante que nessa fase os modelos representem exatamente como os funcionários entendem que os processos são desempenhados na organização.

Análise do processo: Tendo o conhecimento dos processos *as-is* da organização, o próximo passo é a análise e identificação dos problemas associados a eles. O resultado dessa fase é a documentação dos problemas encontrados nos processos com suas prioridades relacionadas a impacto e esforço para solucioná-los.

Redesenho do processo: Essa fase é diretamente dependente da fase de análise, pois é nela que é gerada a documentação do processo *to-be*, ou seja, o processo é redesenhado com seus erros corrigidos e melhorias aplicadas, resultando no processo como ele deverá ser.

Implementação do processo: Realizado o redesenho do processo, as alterações necessárias aos sistemas devem ser executadas, de modo que o processo *to-be* possa ser posto em execução. Nessa fase, as mudanças necessárias para transformar os processos

são preparadas e desempenhadas. A fase de implementação cobre, assim, dois aspectos: organizacional e automação de processos. O aspecto organizacional está ligado ao conjunto de atividades necessárias para modificar a maneira que os participantes do processo executam as tarefas. Já a automação dos processos, em contrapartida, se refere ao desenvolvimento e implantação dos sistemas de TI.

Monitoramento e controle de processo: Uma vez redesenhado o processo e posto em execução, a fase de monitoramento e controle se inicia, para que se analise o desempenho do novo processo. O resultado dessa fase é a coleta de métricas, para verificar se o novo processo atinge os objetivos esperados na fase de redesenho.

2.2 Modelagem de Processo de Negócio

Um processo objetiva melhorar o entendimento de uma situação tal que possa ser comunicada para e entre os diferentes stakeholders de uma organização, a fim de utilizar aquele como ferramenta para atingir os propósitos de negócio da empresa. Não obstante, para processos de negócio serem aptos a atenderem esses objetivos, eles passam por constantes modificações, que surgem das diversas melhorias, adatações e inovações que as empresas buscam e necessitam em seus processos (ROLÓN et al., 2008).

Em BPM os *modelos de processos de negócios* são fundamentais, pois podem ser utilizados tanto para implementação e configuração de sistemas de TI, quanto para análise, compreensão e melhoria dos processos que descrevem (AALST, 2013). Segundo Giaglis (2001), esses modelos facilitam a tomada de decisões e possibilitam filtrar complexidades irrelevantes aos processos, de modo que esforços possam ser direcionados às partes mais importantes do negócio e de sistemas de TI.

O aumento de interesse em uma abordagem mais disciplinada para BPM tem motivado cada vez mais as organizações a realizarem investimentos significativos na área de modelagem de processos (RECKER et al., 2006). Dessa forma, a modelagem de processos de negócio vem obtendo maior importância nos últimos anos, uma vez que ambas as áreas de administração e TI perceberam que tanto sistemas quanto empresas bem sucedidas começam com o completo e correto entendendimento do processo de negócio (AGUILAR-SAVéN, 2004).

O termo modelagem de processo de negócio é utilizado para incorporar todas as atividades relacionadas ao negócio das organizações para modelos que descrevem como os processos são realizados por essas (GIAGLIS, 2001). Um modelo de processo de

negócio consiste em uma série de atividades e limitações de execução entre elas (WESKE, 2007).

É, portanto, um modo estruturado, coerente e consistente de entender, documentar, analizar, simular, executar e melhorar continuamente um processo de negócio e a participação de todos os recursos envolvidos mediante suas contribuições para o desempenho do processo (RECKER et al., 2006). Assim, um modelo de processo equivale à descrição e visualização de processos por meio de um modelo que os representa em modos formais ou informais, seja no formato de um gráfico, seja no formato de um diagrama (ROLÓN et al., 2008).

Antes de iniciar a modelagem de um processo, porém, é crucial entender o porquê dessa modelagem (DUMAS et al., 2013), já que para descrever um processo de negócio diversas formas de informação devem ser integradas em um modelo de processo. Informações que as pessoas querem extrair a partir dos modelos são: o que será feito, quem fará, quando e onde será feito, como e por que será feito, e quais as dependências entre essas atividades (LIST; KORHERR, 2006).

É de suma importância identificar os usos ou propósitos dos modelos quando houver uma modelagem de qualquer tipo (AGUILAR-SAVéN, 2004). Em Krogstie (2012) cinco motivos para modelagem de processos são citados:

1. Compreensão humana: o modelo descritivo de um estado corrente pode ser muito útil para pessoas compreenderem e aprenderem como a situação atual de um processo é percebida e como deve ser realizada.
2. Comunicação entre pessoas de uma organização: modelos podem possuir um importante papel na comunicação humana. Então, além de auxiliar a compreensão individual, um modelo pode atuar como um framework comum para auxiliar a comunicação entre pessoas, tanto na relação de suas atividades, quanto na descrição dessas.
3. Análise assistida por computação: deve ser utilizado para adquirir conhecimento a respeito dos processos da organização, por meio de simulações e deduções, normalmente por meio de comparações entre modelos de um estado *as-is* com um modelo de estados *to-be*.
4. Controle de qualidade: garantir que uma organização atua de acordo com processos certificados desenvolvidos a partir de processos de certificações ISO.
5. Ativação de modelos: integrar o modelo *to-be* em um sistema. Essa integração pode

ser realizada de três maneiras:

1. por meio de pessoas, quando nenhum sistema oferece qualquer tipo de suporte;
2. automaticamente, quando sistemas participam em um papel ativo, como na maioria das automatizações de sistemas workflow;
3. interativamente, quando tanto sistemas de TI, quanto usuários cooperam para a implementação do novo modelo;

Macintosh (1993) e Sousa et al. (2014) reforçam as diferentes modelagens e ferramentas abordando cinco níveis de maturidade e de processos:

1. Inicial: definição dos processos.
2. Repetição: repetição dos processos.
3. Documentação: documentação e padronização dos processos na empresa.
4. Gerenciamento: avaliação de resultados e controle dos processos.
5. Otimização: melhoria contínua, reparos e readaptações de processos.

Sousa et al. (2014) expõem três níveis de abstração de processos:

1. Primeiro nível: é formado pelos macroprocessos (nível mais abstrato, respectivo à cadeia de valor organizacional).
2. Segundo nível: detalha o fluxo de atividades do processo, composto por um ou mais papéis executores, atividades, fluxos, operadores lógicos e eventos.
3. Terceiro nível: detalha o nível operacional demonstrando informações sobre a execução de cada atividade e os elementos envolvidos.

A partir das diferenças de cada modelo, seus objetivos, níveis de maturidade e abstração de um processo, parece claro que, a fim de escolher a técnica correta, o modelador deve saber o propósito do modelo que será construído (AGUILAR-SAVéN, 2004). Devido à popularidade que BPM vem adquirindo, entretanto, o número de representantes de diversas áreas de negócio e departamentos técnicos que não são necessariamente peritos na área de modelagem e que estão cada vez mais envolvidos nas fases de desenho e redesenho do processo vem aumentando significativamente (BECKER; ROSEMANN; UTHMANN, 2000).

Da mesma forma, não só o número de responsáveis pela modelagem dos processos, como também o número de ferramentas e técnicas aumentam cada vez mais. Isso faz com que a seleção dessas ferramentas nas etapas de desenho e redesenho se torne também

cada vez mais complexas (HOMMES; REIJSWOUD, 2000).

As fases de desenho e redesenho são imprescindíveis ao desenvolvimento de um processo de negócio. Essas fases se referem às etapas de modelagem de processos, portanto a escolha correta das ferramentas e pessoas responsáveis têm um grande valor. A etapa de redesenho especialmente, pois costuma ser a mais despendiosa das duas. Essa fase envolve tanto pessoal técnico, quanto analistas de negócio para definir qual a melhor abordagem para melhoria de processos correntes na empresa (ROLÓN et al., 2008). As diferentes técnicas de modelagem de processos de negócio não serão cobertas neste trabalho, visto que o modelo de process já deve estar desenhado corretamente para a aplicação das regras propostas.

As ferramentas utilizadas para o mapeamento dos processos de uma empresa vão desde aplicações pagas (Microsoft Visio[1] e SAP Solution Manager[2]), a gratuítas (Aris [3]) para gerar o model, até linguagens de modelagem de processos, *ou Business Process Modelling Languages BPML*, específicas para cada área. BPMLs diferem na extensão das informações que seus construtores provêm aos leitores do modelo (LIST; KORHERR, 2006). Neste trabalho será apresentada, portanto, a notação que vem ganhando maior popularidade e utilização nos últimos anos, BPMN.

2.3 Notação para Modelagem de Processo de Negócio

O principal objetivo da BPMN é fornecer uma notação compreensível a todos os usuários de negócio. Assim a BPMN auxilia, não só analistas de negócio que criam os rascunhos iniciais dos processos, mas também recursos técnicos responsáveis pela implementação da técnologia que suportará esses até aqueles responsáveis pelo monitoramento e melhorias dos processos (SPECIFICATION, 2014).

Os membros do *Object Management Group* (OMG) com experiência em diversas notações já existentes conseguiram consolidar as melhores ideias dessas em uma notação padrão, BPMN (SPECIFICATION, 2014). Exemplos de diferentes notações ou metodologias revisadas são: *diagramas de atividades UML, UML EDOC Business Process, IDEF, ebXML BPSS, Activity-Decision Flow (ADF) Diagram, RosettaNet, LOVeM e Event-Process Chains (EPCs).*

[1] https://products.office.com/pt-br/visio/flowchart-software
[2] https://support.sap.com/en/alm/solution-manager.html
[3] https://www.ariscommunity.com/aris-express

Segundo Recker (2010), BPMN é utilizada mundialmente por um número cada vez maior de organizações como ferramenta de documentação de processos por possuir um enorme poder de expressão. É considerada também dentro da comunidade de BPM como o padrão preferido por fornecer uma representação gráfica dos processos (PAGNACOS, 2012).

A BPMN possui cinco categorias básicas de elementos na notação (SPECIFICATION, 2014):

1. Objetos de fluxo
2. Dados
3. Objetos de conexão
4. Partições
5. Artefatos

Este trabalho aborda somente os objetos de fluxo e partições. Demais categorias de elementos serão abordados em trabalhos futuros.

2.3.1 Objetos de fluxo

Objetos de fluxo são os principais elementos gráficos utilizados para definir o comportamento de um processo de negócio. Existem três objetos de fluxo: eventos, atividades e desvios.

Evento: é algo que "ocorre"durante o curso de um processo. Esses eventos afetam o fluxo do modelo e usualmente tem uma causa (gatilho) ou um impacto (resultado). Eventos são representados em BPMN por círculos com centros abertos, para possibilitar marcadores internos para diferenciar diferentes gatilhos de resultados. Há três diferentes eventos: início, intermediário e final.

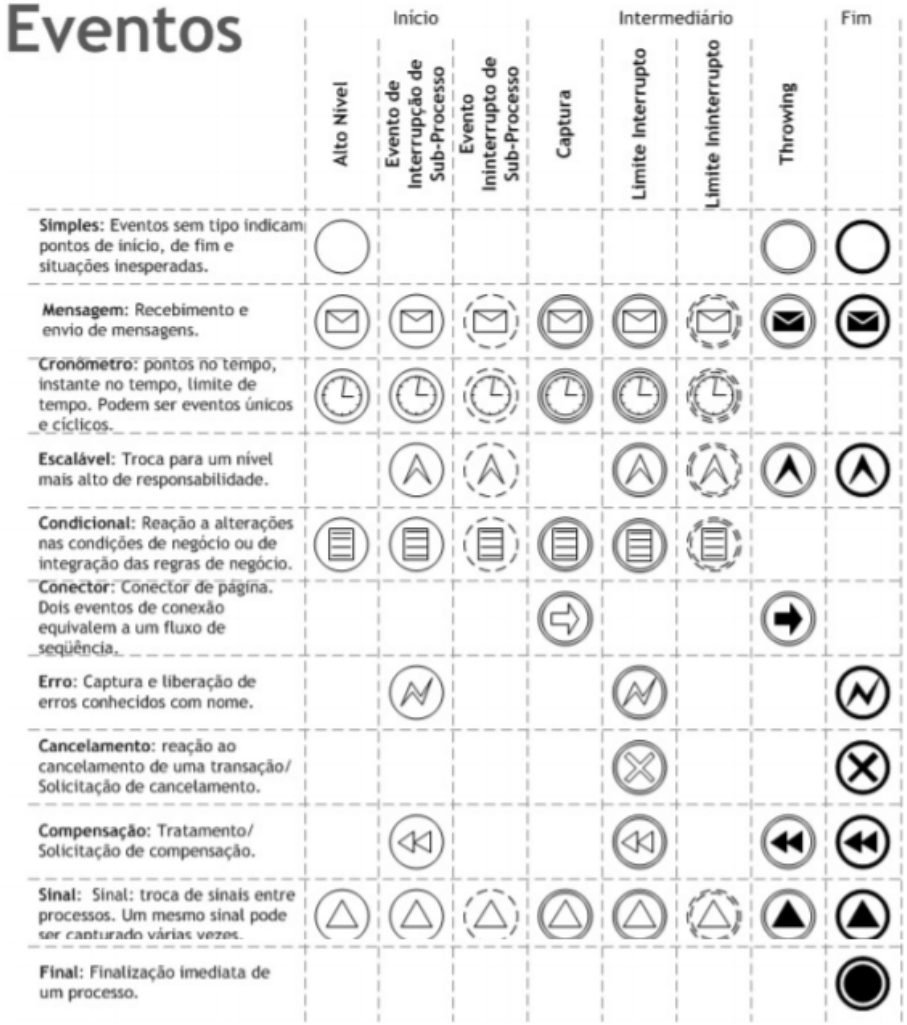
Evento de início como o nome intui, indica "quando"o processo iniciará.

Evento intermediário ocorre entre o evento de início e fim. Eles afetam o fluxo do processo, porém não iniciam ou (diretamente) finalizam um processo.

Evento de fim sinaliza onde e quando o processo chega ao seu término.

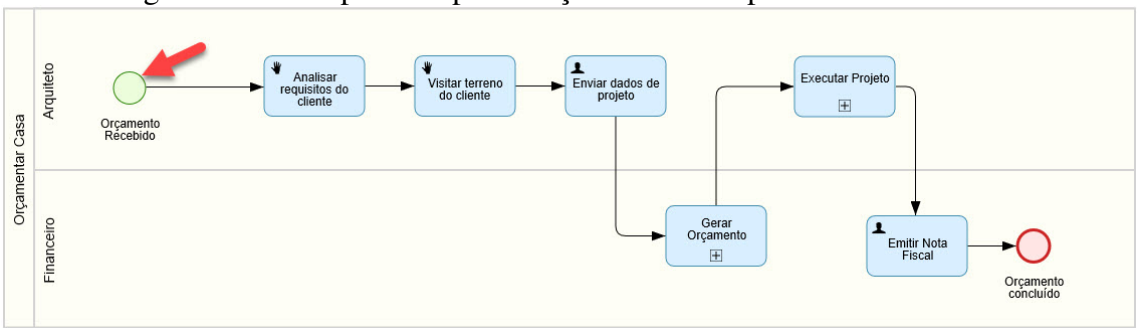
De acordo com a figura 2.2, é possível verificar que existem diferentes tipos de eventos de início, intermédio e fim. Este trabalho abordará somente os eventos qualificados para testes funcionais, visto que as regras são feitas para obtenção de casos de teste manuais funcionais. Assim sendo, desvio múltiplo paralelo não será coberto.

Figura 2.2: Representação de eventos em BPMN



Fonte: (SPECIFICATION, 2014)

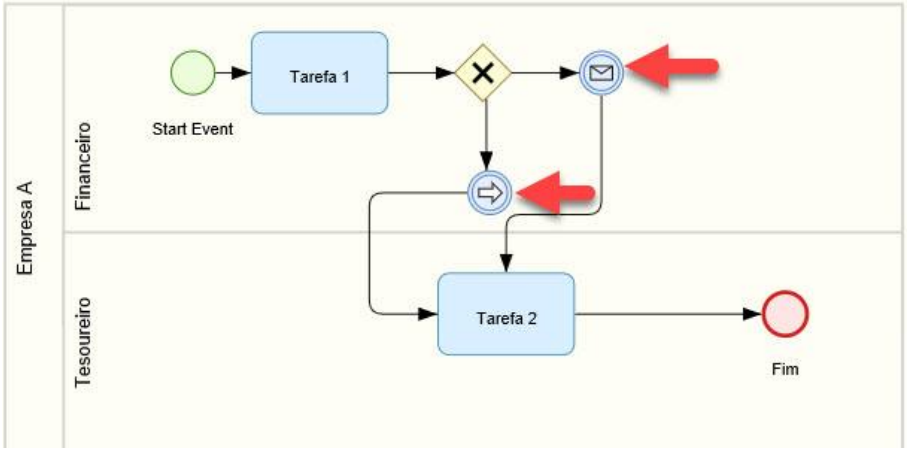
Figura 2.3: Exemplo de representação de BPMN para evento de início



Fonte: O Autor, 2018

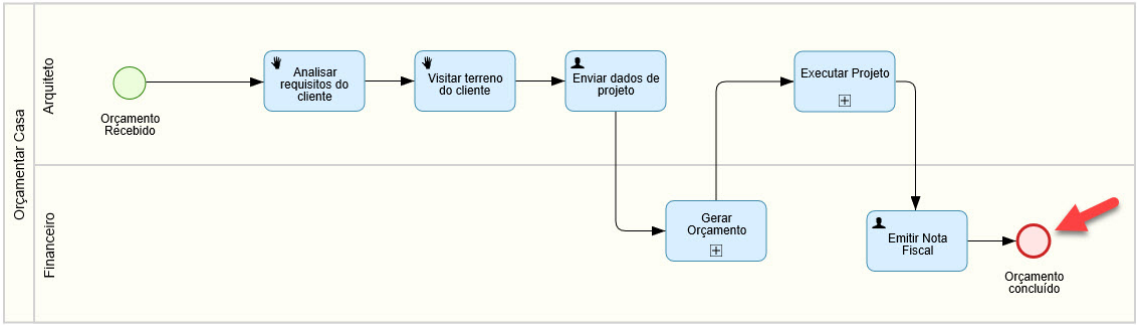
Atividades: é o termo genérico utilizado para representar a tarefa realizada em um processo. Em BPMN há as atividades atômicas, isto é, tarefas executadas diretamente, e não-atômicas, ou seja, subprocessos. As atividades do tipo tarefa podem ser manuais, automáticas ou de usuário.

Figura 2.4: Exemplo de representação de BPMN para eventos intermediários



Fonte: O Autor, 2018

Figura 2.5: Exemplo de representação de BPMN para evento de fim



Fonte: O Autor, 2018

Tarefa manual é uma tarefa que se espera ser executada sem qualquer auxílio de sistemas ou aplicações. As atividades "Analisar requisitos do cliente"e "Visitar terreno do cliente"na figura 2.13 são exemplos de tarefas manuais.

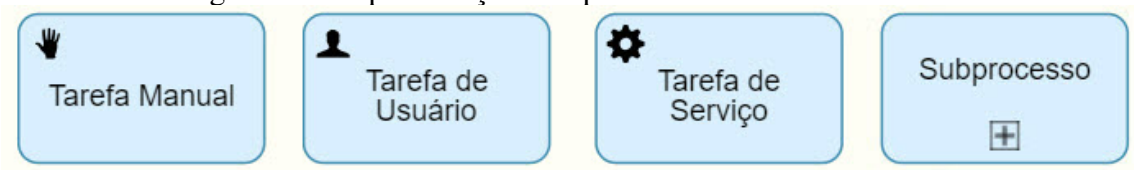
Tarefa automática é uma tarefa que utiliza algum sistema ou aplicação para sua execução sem qualquer intervenção externa. Na notação BPMN há diferentes objetos para representar tarefas automáticas, como tarefa de serviço (por exemplo, utilização de um serviço web), tarefas de envio (utilizada para descrever tarefas que enviam dados a participantes externos), tarefas de recebimento (desenvolvida para indicar o aguardo de dados de participantes externos).

Tarefa de usuário é a típica tarefa de "workflow", onde um humano executa uma tarefa com a assistência de uma aplicação ou sistema de TI. A atividade "Envia dados de projeto"na figura 2.13 ilustra uma tarefa de usuário.

Subprocesso , ou atividade não-atômica, é uma atividade composta que pode ser analisada em um maior nível de detalhes por meio de um conjunto de sub-atividades. Uma atividade de subprocesso é identificado por um sinal de "soma"dentro de um processo, e

suas atividades não são visíveis dentro do processo que o comporta. As atividades "Gera Orçamento"e "Executa Projeto"na figura 2.13 são exemplos de subprocessos. As figuras 2.14 e 2.15 ilustram os detalhes dos respectivos subprocessos.

Figura 2.6: Representação de tipos de atividades em BPMN



Fonte: O Autor, 2018

Desvios são utilizados na modelagem de processo, para permitir a descrição de não somente uma única forma de ocorrência desse, mas também as diversas atividades que devem ocorrer, caso aconteça alguma exceção em nível de negócio durante o processo ou a possibilidade de paralelização de atividades. Dessa forma, BPMN possui diversos objetos para representar os mais variados desvios que podem vir a ocorrer em um fluxo de atividades de um processo. Os desvios em BPMN podem ser exclusivos, inclusivos, paralelos, condicionados por eventos, complexos e condicionados por eventos paralelos.

Desvio exclusivo representa um caminho exclusivo, em que somente um dos fluxos criados a partir do gateway será seguido, de acordo com uma informação a ser testada. Semanticamente, este gateway funciona como um “ou”, já que ou um ou outro caminho será seguido – nunca mais de um (SPECIFICATION, 2014). Esse desvio é representado pelo símbolo de um losango "vazio"ou com um "X"em seu interior.

Desvio inclusivo representa uma condição de fluxo inclusiva, onde pode ocorrer uma junção de caminhos a partir do desvio, de acordo com uma informação a ser verificada. Semanticamente, este *gateway* funciona como um “e/ou”, já que o caminho a ser seguido pode ser um e/ou outro, de acordo com as informações e a lógica do negócio (IPROCESS, 2012). O objeto é representado em BPMN por um losango com um círculo em seu interior.

Desvio paralelo representa a possibilidade de duas ou mais atividades ocorrerem em paralelo em um processo de negócio. Semanticamente, este gateway funciona como um “e”, já que um e outro caminho serão executados (IPROCESS, 2012). Em BPMN utiliza-se um losango com um sinal de "soma"em seu interior, para representar esse objeto.

Desvio condicionado por evento, semelhante ao desvio exclusivo, representa um caminho a ser tomado por um processo mediante o recebimento de um evento ou mensa-

Figura 2.7: Representação de tipos de desvios em BPMN

Fonte: (SPECIFICATION, 2014)

gem externas ao processo.

Desvio condicionado por eventos paralelos unifica os desvios condicionados por evento e desvios paralelos, para representar a decisão de fluxo a ser tomada se eventos ou mensagens externas ocorrerem paralelamente no processo.

Desvio complexo é utilizado quando nenhum dos desvios anteriores pôde ser usado. A especificação BPMN não indica o uso deste recurso sem a tentativa de representar as condições por meio dos outros desvios apresentados anteriormente.

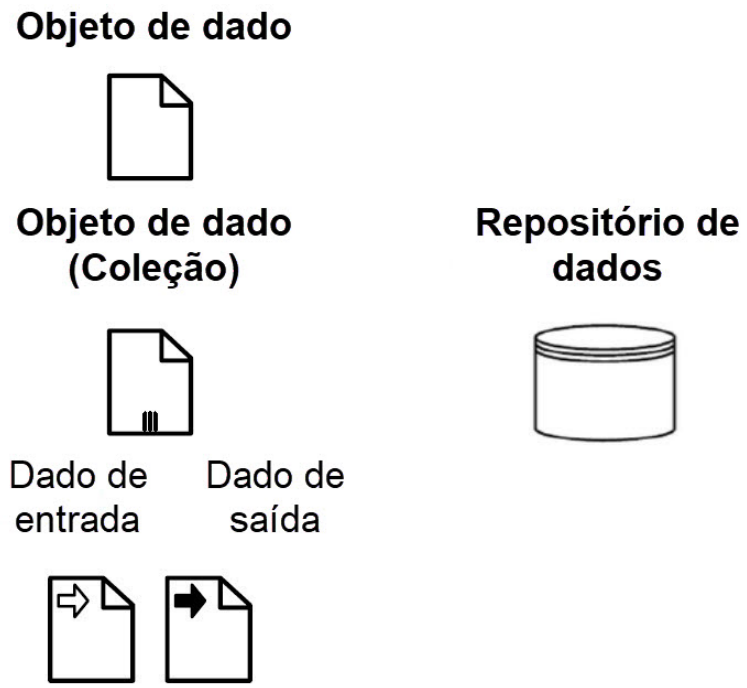
2.3.2 Dados

Durante a modelagem de processos de negócio, pode ser necessário que dados de entrada ou saída de uma atividade estejam explícitos aos leitores do modelo. Desta forma, na BPMN os objetos de dados são utilizados para fornecer informação sobre o quê as atividades requerem para ser realizadas e/ou o quê elas produzem (SPECIFICATION, 2014).

Na BPMN esses objetos podem ser representados por objetos de dados, dados de entrada, dados de saída ou repositório de dados. O objeto repositório de dados é utilizado,

a fim de representar a consulta, recuperação e atualização de dados em um processo.

Figura 2.8: Representação de tipos de desvios em BPMN



Fonte: (SPECIFICATION, 2014)

2.3.3 Objetos de conexão

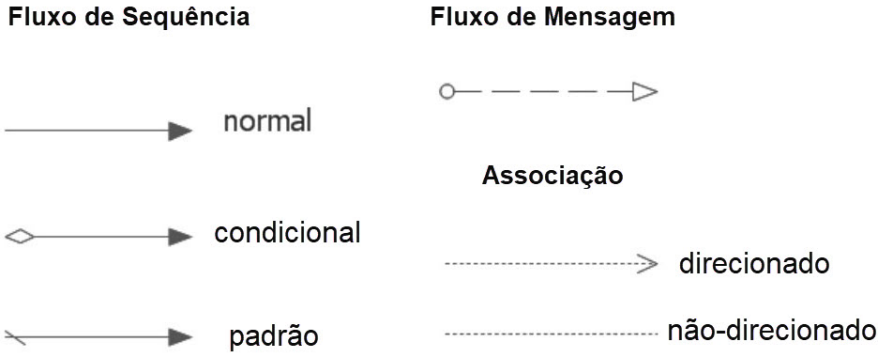
Os objetos de conexão, como o nome indica, servem para conectar diferentes objetos de fluxo na BPMN. Os objetos de conexão são fluxo de sequência, fluxo de mensagem, associações e associações de dados (SPECIFICATION, 2014).

Fluxo de sequência é usado para indicar a ordem de execução das atividades do processo.

Fluxo de mensagem representa a troca de mensagens entre dois participantes/atores em um processo. Para que esse objeto possa estar contido em um modelo, são necessárias pelo menos duas piscinas identificando os participantes do processo.

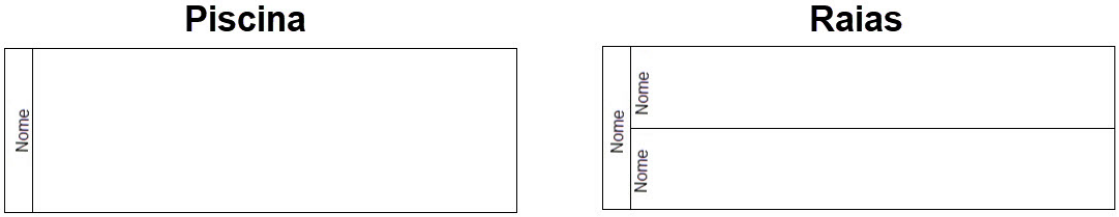
Associações e associações de dados são elementos utilizados na BPMN para conectar informações, artefatos e dados a objetos de fluxo no modelo de processo. A representação desse elemento pode conter uma seta apontando a direção da conexão, caso seja necessário.

Figura 2.9: Representação de tipos de objetos de conexão em BPMN



Fonte: (SPECIFICATION, 2014)

Figura 2.10: Representação de Piscinas e Raias em BPMN



Fonte: (SPECIFICATION, 2014)

2.3.4 Partições

Partições reprensentam os encarregados pelas atividades de um processo. Na BPMN, as partições podem ser piscina ou raia.

Piscina é a representação gráfica de um participante do processo. Esse elemento atua como um container para particionar um conjunto de atividades de outras piscinas.

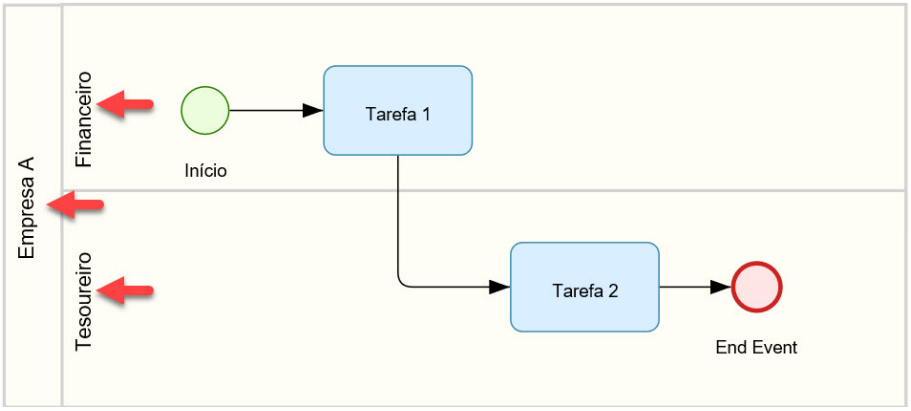
Uma piscina pode conter um processo que será executado, ou simplesmente ser uma caixa preta.

Raia é a subpartição dentro de uma piscina. Podem representar papéis específicos dentro de uma piscina (por exemplo, departamentos de uma organização). Logo, não é possível existir uma raia sem que haja uma piscina no modelo

2.3.5 Artefatos

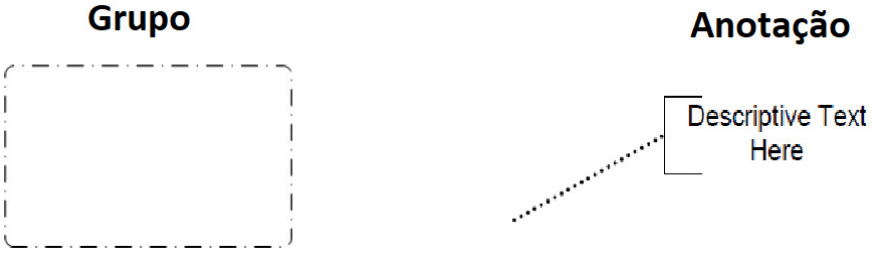
Artefatos provêm informações adicionais sobre o processo ao leitor do modelo. Existem dois artefatos padrões, porém os modeladores ou as ferramentas de modelagem

Figura 2.11: Exemplo de piscina Empresa A e suas respectivas raias: Tesoureiro e Financeiro



Fonte: O Autor, 2018

Figura 2.12: Representação de Grupo e Anotação em BPMN



Fonte: (SPECIFICATION, 2014)

são livres para adicionar quantos sejam necessários. Na BPMN, o conjunto atual de artefatos incluem grupos e anotações (SPECIFICATION, 2014).

Grupo é utilizado para associar objetos de um processo meramente para visualização. A especificação de BPMN não atribui nenhuma semântica a esse elemento.

Anotações são usadas para fins de comentários no modelo.

2.3.6 Modelo textual e gráfico de processo de negócio

Um processo de negócio pode ser representado textualmente como uma lista de atividades a serem cumpridas, ou no formato gráfico de um diagrama. Um exemplo de modelo de processo textual de orçamentos de projetos de casa é apresentado a seguir. O processo foi obtido pelo autor a partir de entrevistas com funcionários de uma empresa de arquitetura da cidade de Cotiporã no Estado do Rio Grande do Sul.

O processo inicia com o recebimento de um pedido de um cliente para orçamento de projeto de uma casa pelos arquitetos da empresa. Os arquitetos verificam qual o escopo

da obra, podendo gerar um orçamento inicial relativo a m^2 ou por tempo estimado de realização dos projetos, visto que são entregues sempre três projetos arquietônicos ao cliente, anteprojeto (rascunho de projeto mostrado ao cliente), projeto legal (protocolado na prefeitura da cidade, para iniciar as obras) e projeto executivo (projeto da obra em si).

O valor estimado pelos arquitetos de cada projeto é, então, enviado ao departamento financeiro, onde é gerado o orçamento ao cliente junto com seu cadastro, a fim de manter dados pessoais para geração de contratos entre cliente e empresa.

O departamento financeiro envia o orçamento ao cliente, e o cliente possui a opção de aceitar ou não.

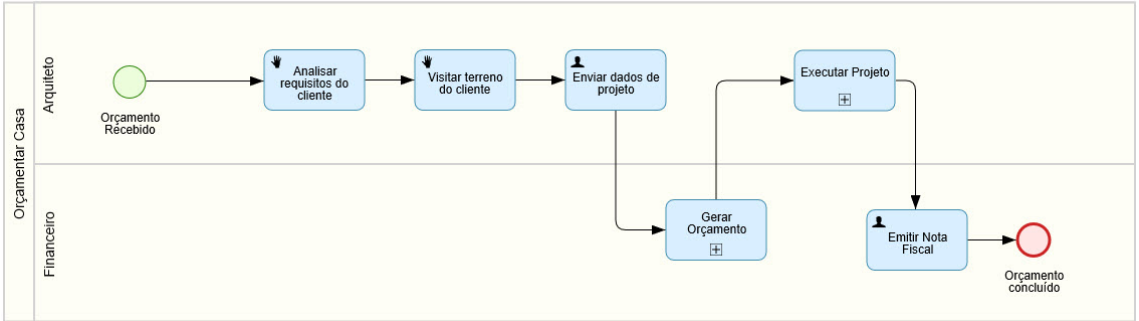
1. Caso aceite, necessita fazer um depósito de 30% do valor total dos projetos, para que se inicie o desenho dos projetos. Após a protocolação na prefeitura do projeto legal, mais 40% devem ser debitados na conta da empresa, e depois de entregues os três projetos ao cliente, a empresa recebe os 30% restantes.
2. Caso recuse, o cadastro do cliente é mantido, porém nenhuma execução de projeto é realizada.

Na notação BPMN o processo de orçamentar projeto de casa descrito textualmente anteriormente pode ser representado através do modelo de processo apresentado na 2.13.

Os subprocessos "Gerar Orçamento"e "Executa Projeto"são ilustrados nas imagens 2.14 e 2.15.

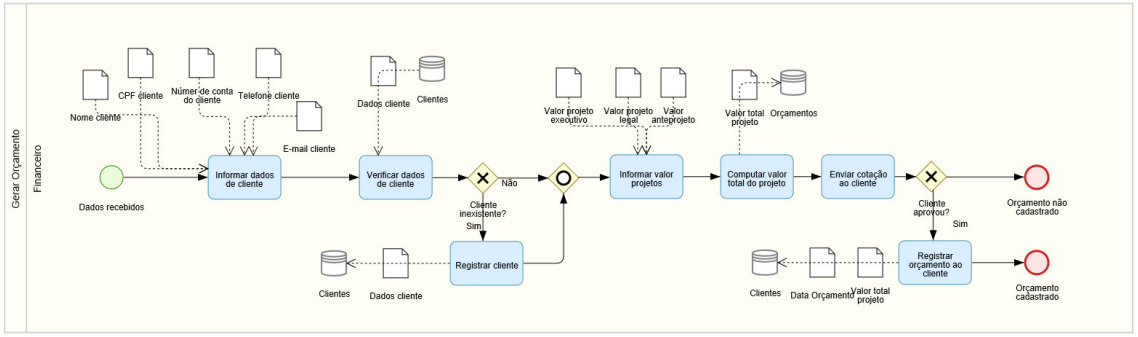
É necessário enfatizar que um dos objetivos do desenvolvimento de BPMN foi gerar um mecanismo simples e compreensível para criar modelos de processos, mas que ao mesmo tempo possuísse a capacidade de lidar com as inerentes complexidades desses (SPECIFICATION, 2014). Assim, a notação possui um grande número de símbolos possíveis de se utilizar para representar qualquer tipo de processo.

Figura 2.13: Modelo de processo em BPMN para orçamentar casa



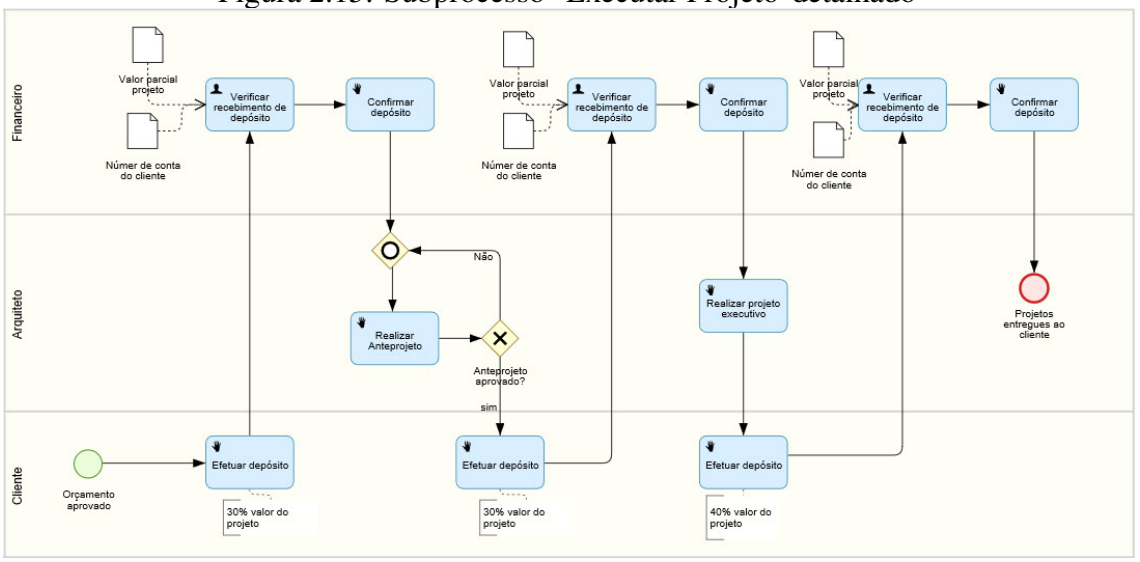
Fonte: O Autor, 2018

Figura 2.14: Subprocesso "Gerar Orçamento"detalhado



Fonte: O Autor, 2018

Figura 2.15: Subprocesso "Executar Projeto"detalhado



Fonte: O Autor, 2018

3 FUNDAMENTOS DE TESTE DE SOFTWARE

Este capítulo apresenta conceitos e fundamentos, os quais serão usados neste trabalho, de teste de *software*, suas metodologias e artefatos. Serão contemplados também a relevância de testes no ciclo de vida do desenvolvimento de uma aplicação e os benefícios que essa práticas traz, não só em termos econômicos, mas também em critérios de qualidade de *software*.

3.1 Conceitos de teste

Teste de *software* é o processo que envolve a execução de um *software* (programa/aplicação), a fim de encontrar os erros nesse resultando em um *software* livre de defeitos (SNEHA; MALLE, 2017). É uma das principais atividades no processo de desenvolvimento para encontrar defeitos de *software*, que nos mostram como e onde a aplicação carece de melhorias e correções (RATHI; MEHRA, 2015).

Segundo Chauhan and Singh (2014), começar testes em fases iniciais de desenvolvimento auxilia em encontrar e corrigir erros de programação e especificação, portanto reduzindo o custo de manutenção em estágios finais do ciclo de desenvolvimento de *software*. Estima-se que cerca de 67% dos custos de um *software* se encontram em estágios de manutenção e operação, dessa forma, um *software* desenvolvido com boa qualidade traz retornos às organizações a longo prazo (KANER; FALK; NGUYEN, 2000). Chauhan and Singh (2014) sugerem que as empresas já entenderam a importância do processo de teste, e essas gastam em média 40% de seus recursos nesse.

Conforme Sharma and Vishawjyoti (2013), o objetivo final de desenvolvimento de *software* é produzir *software* de alta qualidade. Alta qualidade significa custos baixos, alta confiabilidade e satisfação de usuário. Teste de *software* se faz, portanto, o principal recurso de garantia de qualidade de uma aplicação, sendo um elemento crítico para revisão das especificações e verificação da corretude de *software* (RATHI; MEHRA, 2015).

Chauhan and Singh (2014) afirmam que existem quatro propósitos para testar uma aplicação:

- *Detectar* erros, defeitos e deficiências, a fim de medir limitações e qualidade dos componentes do sistema.
- *Prevenir* e reduzir número de erros provendo as informações necessárias para es-

clarecer as especificações e avaliar o desempenho do sistema.

- *Demonstrar* como o sistema pode ser utilizado com riscos aceitáveis, funções com condições especiais de execução e o quão preparado o sistema está para integração com outros sistemas.
- *Melhorar a qualidade* minimizando o número de erros do sistema ao fazer testes efetivos.

Chauhan and Singh (2014) complementam que esses propósitos servem para:

- *Verificar* se o sistema se comporta conforme especificação. É a verificação e teste da conformidade e consistencia de *software* por avaliação de resultados sobre requerimentos pré-definidos. Na verificação é feita a pergunta: "estamos construindo o produto correto?".
- *Validar* a corretude do sistema, isto é, o proceso de checar que o que foi desenvolvido pela equipe técnica está de acordo com o especificado pelo usuário. Aqui é realizada a questão: "estamos construindo o sistema correto?"
- *Detectar* erros por meio de um número de testes, que devem ser realizados, para verificar o que acontece com o sistema quando algo dá errado, seja em nível funcional, seja em nível técnico.

Avaliar a qualidade de teste de *software* costuma ser uma tarefa complexa. Moura et al. (2017) afirmam que uma das principais métricas para teste de *software* é a cobertura de testes, a qual mede a quantidade de código testada e pode ser expressa pela cobertura de casos de teste ou cobertura de código executado.

Com o surigmento de novas tecnologias, o número de técnicas de verificação e métodos de testes de *software* antes de chegarem à produção aumentaram (SNEHA; MALLE, 2017). Testes podem ser executados de forma automatizada ou manual, e cada método possui vantagens e desvantagens, que devem ser analisadas pela equipe de desenvolvimento considerando custos e maturidade do projeto.

Rathi and Mehra (2015) expõem que a criação de casos de testes e sua execução sem o suporte de qualquer ferramenta é conhecido como teste manual. Essa tarefa normalmente é realizada com um humano sentado na frente de um computador lendo atentamente uma especificação com uma série de combinações de dados de entrada, executando as tarefas com esses dados e, então, os comparando com resultados esperados. Testes manuais requerem que um usuário, testador, realize operações de testes manuais sem o auxílio de uma ferramenta automatizada. É o processo onde um testador geral-

mente segue um plano de teste escrito que o guia através de um conjunto de casos de testes (SHARMA; VISHAWJYOTI, 2013).

Logo, a quantidade necessária de recursos humanos de uma empresa que utiliza testes manuais será maior do que em empresas que utilizam testes automatizados, e o número de erros que podem passar despercebidos pelos testadores é igualmente maior (RATHI; MEHRA, 2015).

Automatização de testes é a execução de scripts de testes, onde a intervenção manual não é necessária para tal. Essa estratégia utiliza ferramentas especiais, tanto para escrever os scripts, quanto para executá-los (SHARMA; VISHAWJYOTI, 2013). O objetivo da automatização é reduzir o número de casos de testes que devem ser rodados manualmente e não eliminar os testes manuais (RATHI; MEHRA, 2015). Segundo Rafi et al. (2012), os benefícios da automatização de testes são a reusabilidade de scripts, facilidade em repetição de execução e economia de esforço na execução dos casos de testes.

Pesquisas mostraram que as limitações são o alto investimento inicial em compra de licenças e configuração de ferramentas, além de treinamento para utilização. Ademais, em (RAFI et al., 2012), 45% dos entrevistados concordaram que as ferramentas existentes no mercado não se enquadram nas suas necessidades, e 80% discordam que a automatização substituirá completamente os testes manuais.

Outras limitações de testes automatizados são a dificuldade de manutenção de scripts, tempo para consolidação do processo de automatizado, falsas expectativas de economia ao gerar scripts de testes automatizados inúteis, estratégia de teste inapropriada e falta de habilidade nas ferramentas por parte dos usuários (RAFI et al., 2012).

Assim, os objetivos de testes manuais e automatizados devem ser considerados diferentemente. Testes manuais são melhores para ambientes onde requerimentos mudam continuamente e auxiliam principalmente na descoberta de defeitos relacionados à usabilidade e interface de usuário (SHARMA; VISHAWJYOTI, 2013). Testes automatizados devem ser utilizados para tarefas com maior repetição de trabalho, pois é a melhor maneira de aumentar a eficiência e efetividade no processo de teste, segundo Sharma and Vishawjyoti (2013).

Em (KANER; FALK; NGUYEN, 2000), os testes são ainda divididos em funcionais e estruturais. Testes funcionais são aqueles em que funções específicas da aplicação são testadas por meio da inserção de dados e análise da saída, sem qualquer verificação do código fonte. O método de teste de caixa preta apresentado na próxima seção é um exemplo de teste funcional. Testes estruturais, por outro lado, são aqueles em que se ana-

lisa o código fonte, a fim de gerar casos de testes específicos para uma determinada parte do programa. O método da caixa branca, citado posteriormente, é um exemplo de teste estrutural.

3.2 Estratégias e metodologias para teste de *software*

Uma estratégia de teste pode ser definida como um esboço que retrata a abordagem do teste dentro do ciclo de vida do desenvolvimento de *software* e é constituída por uma grande variedade de metodologias de casos de teste (SNEHA; MALLE, 2017). Para cada etapa do ciclo de vida do desenvolvimento de uma aplicação há formas mais adequadas de se criar casos de teste para essa.

Segundo Chauhan and Singh (2014), uma estratégia de teste integra vários métodos para casos de testes em um plano cosistente que resultará em um *software* livre de defeitos. As estratégias de testes propostas pelos autores são as seguintes: unitário, integrado, sistema e aceitação.

- **Teste Unitário**
  É realizado no menor nível testando a unidade básica de um *software*, que pode ser um módulo ou componente. Unidade é considerado o menor módulo de um componente da aplicação, isto é, um conjunto de linhas de código que pode ser testado.
- **Teste Integrado**
  É realizado quando duas ou mais unidades são combinadas em uma estrutura maior. Essa estratégia costuma ser empregada entre as interfaces que conectam componentes e a estrutura maior que está sendo desenvolvida.
- **Teste de Sistema**
  Executa-se um teste de sistema para verificar a qualidade ponta-a-ponta de uma aplicação como um todo. É baseado nas especificações funcionais do *software*, porém critérios não funcionais como segurança, confiabilidade e manutenibilidade também são validados.
- **Teste de Aceitação**
  É realizado quando o sistema está pronto para ser entregue ao cliente. O objetivo dessa estratégia é assegurar aos usuários finais que o sistema está funcionando corretamente, ao invés de encontrar erros.

As metodologias mais comuns em teste de *software* são conhecidas como caixa branca e caixa preta (SNEHA; MALLE, 2017). Essas diferem primariamente na forma que o código do programa é considerado (KANER; FALK; NGUYEN, 2000).

Segundo Kaner, Falk and Nguyen (2000), na metodologia caixa preta o testador não vê o código que está sendo executado. O responsável pelo teste simplesmente insere dados de entrada no programa e compara as saídas de execução com resultados esperados, conforme indicado no plano de teste. Dessa forma, os casos de testes são baseados em condições e classes de dados que a equipe de engenharia de testes acredita que podem ocasionar em erros de execução.

As técnicas para geração de casos de teste à metodologia caixa preta costumam ser, segundo Sneha and Malle (2017): tabela de decisão, advinhação de erros e transição de estados. Chauhan and Singh (2014) complementa essas técnicas citando: partição de equivalência, "se você espera o mesmo resultado de dois casos de teste, eles são equivalentes"(KANER; FALK; NGUYEN, 2000), e análise de valores de fronteiras, "maiores, menores, mais feios valores são o que chamamos de valores de fronteiras; programas que falham com valores que não são de fronteira também costumam falhar com valores de fronteira"(KANER; FALK; NGUYEN, 2000).

O principal benefício dessa metodologia é que os testadores não necessitam despender tempo em aprender o código do *software*. Utilzando o método de caixa preta, comumentemente as equipes de testes são integradas por empregados funcionais da organização sem qualquer fundamentação em áreas da tecnologia de informação (KANER; FALK; NGUYEN, 2000).

A metodologia caixa branca, também conhecida como caixa de vidro, em contrapartida faz uso do conhecimento das linhas de código para criação dos casos de testes. Assim, esse método costuma ser utilizado por programadores, especialmente em testes unitários, pois esses sabem como o programa funciona, seus desvios e condições (KANER; FALK; NGUYEN, 2000).

Diversas técnicas para criação de casos de teste ao método da caixa branca podem ser utilizadas. Sneha and Malle (2017) citam as seguintes: cobertura de decisão, teste de fluxo de dados, de caminho principal e de fluxo de controle.

3.3 Artefatos para engenharia de teste

Os processos de teste determinam se os produtos de desenvolvimento de uma determinada atividade estão em conformidade com os requisitos dessa e se o sistema e / ou *software* satisfaz seus uso pretendido e necessidades do usuário (STANDARD, 2008). As tarefas do processo de teste são especificadas em diferentes níveis de integridade, com o intuito de verificar a necessidade de maior ou menor detalhamento de informações aos documentos de cada uma daquelas.

Esses documentos relacionados ao processo de teste de *software* são chamados de artefatos de teste de *software*. Esta seção objetiva apresentar alguns artefatos relevantes ao presente trabalho, como planos de testes, desenho de teste de níveis e casos de teste baseados no padrão (STANDARD, 2008). Demais artefatos poderão ser considerados em trabalhos futuros.

3.3.1 Plano de teste mestre

O plano de teste é o documento resultante da etapa de planejamento de teste, atividade que compreende a seleção das partes constituintes de ciclos de testes. Segundo Standard (2008), ciclo de teste equivale a plano de teste de nível, que será apresentado posteriormente.

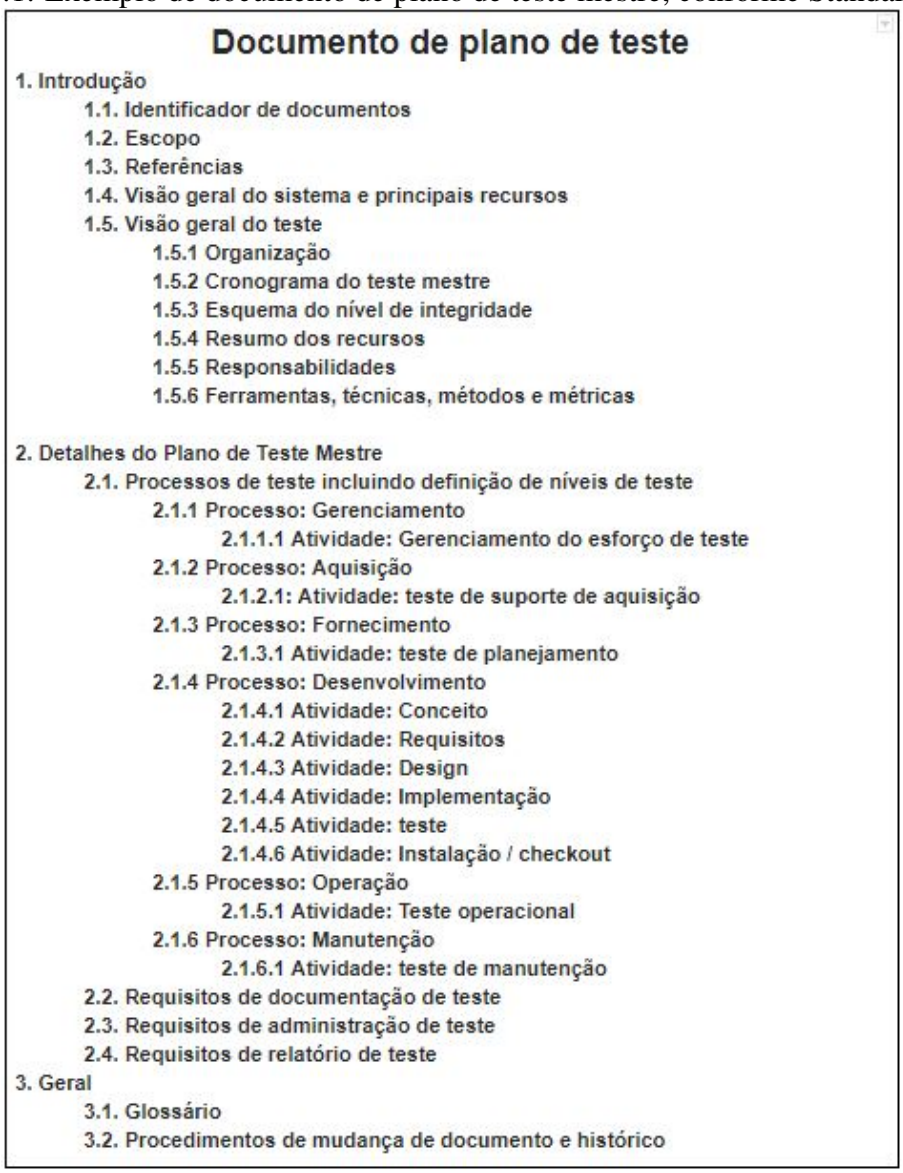
O plano de teste mestre estabelece, então, objetivos, divisão de trabalho e tempo dos recursos assinalados a cada ciclo de provas e a interrelação entre cada uma das partes. Ademais, o planejamento de teste mestre visa identificar riscos definindo o nível de integridade de cada ciclo. Um exemplo de documento de plano de teste mestre é ilustrado na figura 3.1.

3.3.2 Plano de teste de nível

Para cada plano de teste de nível é especificado o escopo, abordagem, recursos e cronograma das atividades de teste para cada ciclo de provas. O documento de plano de teste de nível identificará os itens que serão testados, os recursos que testarão, as tarefas de teste a serem executadas, o pessoal responsável por cada tarefa e o(s) risco(s) associado(s).

Um exemplo de documento de plano de teste mestre é ilustrado na figura 3.2.

Figura 3.1: Exemplo de documento de plano de teste mestre, conforme Standard (2008)



Documento de plano de teste

1. Introdução
    1.1. Identificador de documentos
    1.2. Escopo
    1.3. Referências
    1.4. Visão geral do sistema e principais recursos
    1.5. Visão geral do teste
        1.5.1 Organização
        1.5.2 Cronograma do teste mestre
        1.5.3 Esquema do nível de integridade
        1.5.4 Resumo dos recursos
        1.5.5 Responsabilidades
        1.5.6 Ferramentas, técnicas, métodos e métricas

2. Detalhes do Plano de Teste Mestre
    2.1. Processos de teste incluindo definição de níveis de teste
        2.1.1 Processo: Gerenciamento
            2.1.1.1 Atividade: Gerenciamento do esforço de teste
        2.1.2 Processo: Aquisição
            2.1.2.1: Atividade: teste de suporte de aquisição
        2.1.3 Processo: Fornecimento
            2.1.3.1 Atividade: teste de planejamento
        2.1.4 Processo: Desenvolvimento
            2.1.4.1 Atividade: Conceito
            2.1.4.2 Atividade: Requisitos
            2.1.4.3 Atividade: Design
            2.1.4.4 Atividade: Implementação
            2.1.4.5 Atividade: teste
            2.1.4.6 Atividade: Instalação / checkout
        2.1.5 Processo: Operação
            2.1.5.1 Atividade: Teste operacional
        2.1.6 Processo: Manutenção
            2.1.6.1 Atividade: teste de manutenção
    2.2. Requisitos de documentação de teste
    2.3. Requisitos de administração de teste
    2.4. Requisitos de relatório de teste
3. Geral
    3.1. Glossário
    3.2. Procedimentos de mudança de documento e histórico

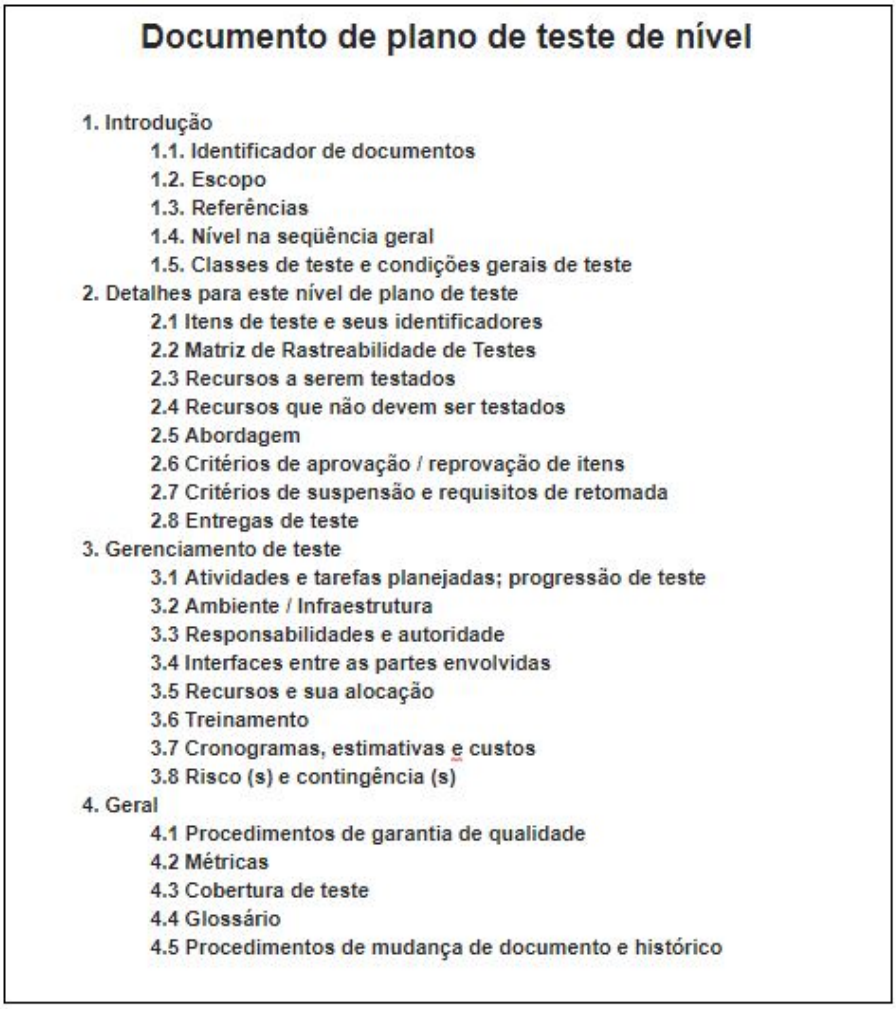
Fonte: (STANDARD, 2008)

3.3.3 Desenho de teste de nível

O desenho de teste de nível é o detalhamento das atividades a serem testadas descritas no documento de plano de teste de nível. Dessa forma, o desenho de teste de nível identifica as funcionalidades a serem testadas e os casos de testes associados a essas funcionalidades.

Um exemplo de documento de desenho de teste de nível é ilustrado na figura 3.3.

Figura 3.2: Exemplo de documento de plano de teste de nível, conforme Standard (2008)



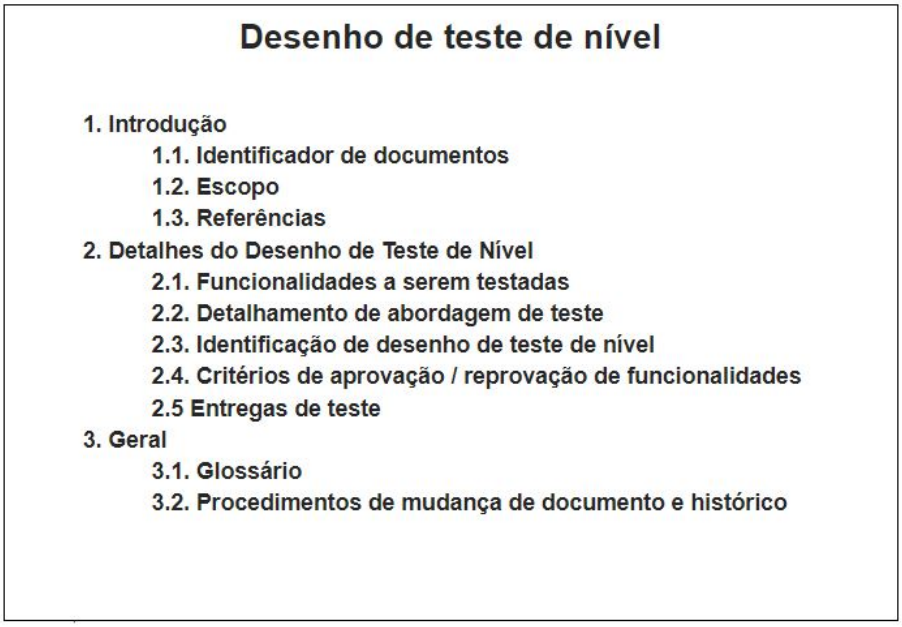
Fonte: (STANDARD, 2008)

3.3.4 Caso de teste

Caso de teste é um documento com o propósito de definir a informação necessária, para que alguém execute uma determinada função de um *software*. Dados de entrada, saída, objetivo da função, são alguns dos campos que (STANDARD, 2008) recomenda conter em um caso de teste.

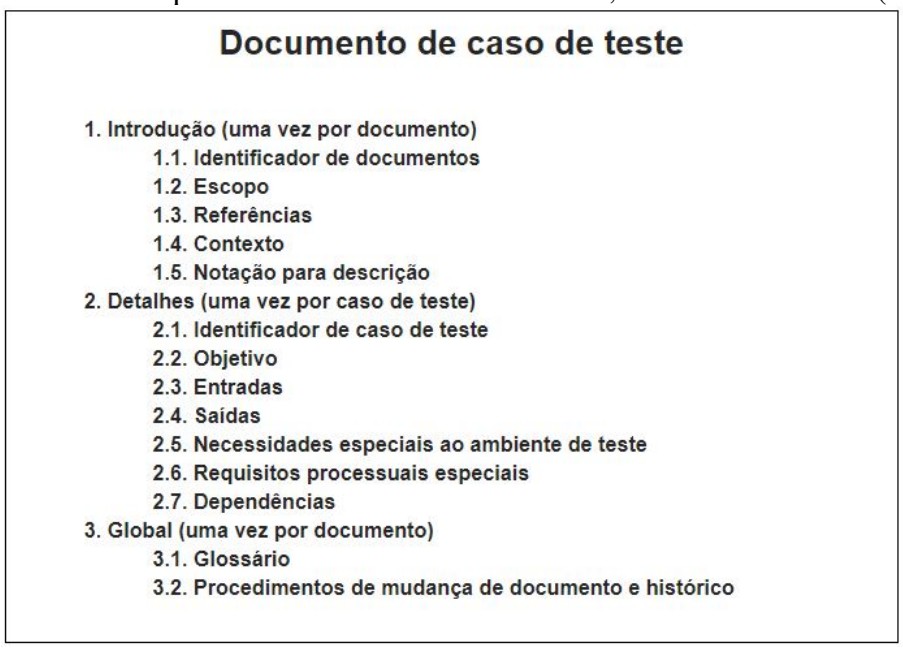
Um exemplo de documento de caso de teste é ilustrado na figura 3.4.

Figura 3.3: Exemplo de documento para desenho de teste de nível, conforme Standard (2008)



Fonte: (STANDARD, 2008)

Figura 3.4: Exemplo de documento de caso de teste, conforme Standard (2008)



Fonte: (STANDARD, 2008)

4 MAPEAMENTO DE MODELOS DE PROCESSO DE NEGÓCIO PARA CASOS DE TESTE

Este capítulo objetiva apresentar um estudo de caso a partir do conjunto de regras propostas por (SOUSA et al., 2014), que possibilite o mapeamento de um modelo de processo para casos de testes durante a fase de implementação no ciclo de vida de BPM. Esse estudo de caso tem o intuito de auxiliar pessoas, tanto de áreas relacionadas à tecnologia da informação, quanto funcionais que necessitam fazer uso desses modelos como material de insumo para auxílio ao desenvolvimento de sistemas de TI.

As regras de (SOUSA et al., 2014) têm como base modelos na notação BPMN (SPECIFICATION, 2014) em um terceiro nível de detalhamento. Os elementos notacionais analisados, a fim de identificar um possível mapeamento estão descritos na tabela 4.1.

Para o presente trabalho foram considerados somente os objetos cobertos em (SOUSA et al., 2014): objetos de fluxo (atividades de serviço, usuário e subprocessos, e desvios com exceção de temporais e paralelos), objetos de dados (dados de entrada, saída, e repositório de dados) foram mapeados. Objetos de conexão, como fluxo de sequência servirão para medir o critério de cobertura do modelo de processo. Outros tipos de atividades, eventos e desvios não foram utilizados nesse estudo, pois não foram tratados em (SOUSA et al., 2014).

Tabela 4.1: Elementos notacionais da notação BPMN relevantes a este trabalho

Elemento BPMN	*Descrição*
Atividade	Utilizada para descrever uma tarefa dentro de um processo neste trabalho (somente atividades do tipo tarefa de usuário, serviço e subprocessos serão consideradas).
Dados	Utilizados para identificar dados de entrada, saída e persistência no modelo.
Desvio	Usado para controle de fluxo de sequência de um processo por meio de divergências e convergências de tarefas (neste trabalho, somente será considerado o desvio exclusivo do tipo XOR).
Eventos	Representam ocorrências de eventos no início, fim, ou durante um processo (neste trabalho, apenas eventos de mensagem, temporais, erros e cancelamentos serão abordados).

Fonte: O Autor, 2018

A fim de obter melhores resultados na extração dos casos de teste, as regras de negócio e atividades do processo devem ser mapeadas corretamente ao modelo sem ambiguidades (SOUSA et al., 2014). Quanto maior o detalhamento dos dados de entrada

e saída, assim como atividades de persistência de dados corretamente assinaladas, mais detalhado será também o caso de teste obtido para a atividade. "Identificamos que quanto maior a abstração do modelo de processo, maior é o impacto nos casos de teste"(SOUSA et al., 2014).

As atividades do tipo "sub-processo"gerarão casos de testes independentes, assim cada modelo definirá um conjunto específico de casos de testes, conforme proposto em (SOUSA et al., 2014). Para processos ponta-a-ponta, compostos por diversas tarefas de "sub-processos", os casos de testes deverão ser integrados pela equipe responsável, pois "a escolha de realizar um caso de teste mais específico ou mais abrangente fica a cargo dos engenheiros de software"(SOUSA et al., 2014).

O modelo textual não possui heurísticas de mapeamento em (SOUSA et al., 2014). Segundo os autores, um dos benefícios de suas heurísticas é "aplicação direta nos modelos gráficos, sem a necessidade de transformações ou construção de modelos intermediários que auxiliem no processo de definição de casos de teste". O modelo textual, porém, se existente, pode auxiliar, não só na obtenção de requisitos não funcionais, que podem não estar presentes no modelo gráfico, mas também na compreensão de regras de negócio não explicitadas no diagrama (SOUSA et al., 2014).

As regras propostas em (SOUSA et al., 2014) objetivam gerar uma suíte compostas de testes unitários e integrados. Esses testes deverão ser analisados por usuários funcionais de negócio da empresa e por engenheiros de testes, com intenção de validar se os testes estão coerentes e se as regras de negócio, de fato, estão sendo respeitadas (SOUSA et al., 2014).

O resultado final será uma base de documentos de casos de teste e desenhos de testes de nível que irão conter as atividades que estão sendo analisada, seus dados de entrada e métodos para testá-los, saída esperada e caminho percorrido no processo de negócio. Esses documentos poderão, então, serem utilizados para testes manuais funcionais na metodologia caixa preta, além de teste de integração e de sistema.

A forma proposta em (SOUSA et al., 2014) como documento de caso de teste é demasiadamente complexa. Os autores dispõem todos os testes obtidos do modelo em uma única tabela com dados de linhas muitas vezes repetidos. Para o presente trabalho, serão utilizados os formatos de documentos padrão reconhecido pela IEEE, (STANDARD, 2008), apresentados no capítulo 3, pois possibilitam a visualização clara e simplificada de um caso de teste. Os campos considerados dos documentos serão somente os mapeados em (SOUSA et al., 2014), de modo que apenas a disposição dos dados obtidos do modelo

diferirá entre o presente estudo de caso e o trabalho de (SOUSA et al., 2014).

Para documentos de casos de teste serão considerados, portanto, os campos: identificador de caso de teste, entradas, saídas, nome da atividade, condições e consequências. Os mesmos campos são considerados em (SOUSA et al., 2014).

O documento de desenho de teste de nível da proposta de (STANDARD, 2008) será utilizado, pois no trabalho de (SOUSA et al., 2014), os autores mantém as informações de fluxo do modelo em campos da tabela. Dessa forma, os autores não fazem distinção entre casos de testes unitários e integrados. Para o presente estudo de caso, a adaptação do documento de (STANDARD, 2008) objetiva facilitar às pessoas que não têm conhecimento na área de teste a distinguir um teste unitário de um teste integrado.

No documento de desenho de teste de nível da proposta de (STANDARD, 2008), foram utilizados somente os campos: funcionalidades a serem testadas, identificador do desenho de teste de nível, critérios de aprovação / reprovação de funcionalidade. Os campos são análogos à proposta de (SOUSA et al., 2014).

4.1 Verificação de um modelo de processo

A corretude de um processo de negócio é uma das principais áreas de pesquisa na disciplina de modelagem de processos de negócio (MENDLING; ROSA; HOFSTEDE, 2008). Essa área visa a remoção de ambiguidades e o correto mapeamento das regras de negócio a um modelo gráfico.

A própria especificação BPMN (SPECIFICATION, 2014) não cita as melhores práticas para nomear atividades, ou mapear regras de negócio no modelo. Para este trabalho, a nomenclatura das atividades do tipo tarefa seguirá o padrão estabelecido por (MENDLING; REIJERS; RECKER, 2010), que também é utilizado por grandes empresas de software como SAP. As atividades serão nomeadas sempre por "Verbo + objeto", com o intuito de diminuir ambiguidades.

Conforme Sousa et al. (2014), para a geração de casos de teste, os seguintes elementos são necessários no modelo de processo: comportamento do sistema, informações que são tratadas e geradas no requisito, momento de inicio de execução do requisito, produtos gerados na execução do requisito (se houver) e momento de finalização do requisito.

Comportamento do sistema é responsável por definir como ele deve responder aos comandos do usuário e aos estados alcançados. Informações tratadas e geradas no requisito são os dados (em seus devidos formatos) necessários para oferecer o insumo

correto na execução dos testes e a verificação do produto final. Momento de início é caracterizado pelo estado que dispara a execução do requisito. Momento de finalização é caracterizado pelo estado que deve ser alcançado após a execução do teste (SOUSA et al., 2014).

4.1.1 Mapeamento de atividades

Seguindo a proposta de (SOUSA et al., 2014), as atvididades candidatas a um caso de teste são as atividades do tipo de usuário ou de sistema. Neste estudo de caso para cada uma dessas atividades no modelo de processo será obtido um documento caso de teste unitário, que poderá ser utilizado em um desenho de nível de teste posteriormente.

O resultado final desta etapa será um esboço inicial do documento de caso de teste contendo os campos: "identificador da atividade"(identificador único para todos para referenciar a atividade nos diferentes desenhos de testes de nível do modelo) e "nome da atividade".

4.1.2 Mapeamento de dados

Os objetos de dados serão integrados ao caso de teste adicionando-se os campos "Dados de entrada"e "Dados de saída", como em (STANDARD, 2008), análogos aos campos "entradas"e "saídas"da tabela proposta em (SOUSA et al., 2014). Para atividades de usuário, os dados de entrada devem ser corretamente mapeados para o nome dos campos da interface gráfica da aplicação. Dessa forma, o caso de teste servirá também como material de insumo, para validar se os requisitos de *UI* estão sendo cumpridos, conforme o esperado pelos usuários finais.

Assim como em (SOUSA et al., 2014), quando houver uma atividade relacionada a um objeto "repositório de dados", deverá existir um caso de teste para verificação do CRUD da atividade. Objetos do tipo "repositório de dados"indicam a persistência de dados de saída da atividade ou leitura de uma base de dados, logo é possível gerar casos de testes unitários individuais para essas atividades que sejam conectadas com esse objeto.

No presente estudo de caso, tipos básicos, como inteiros e *string* deverão ser testados por equivalência de partições e valores de fronteira, conforme Yotyawilai and Suwannasart (2014). Por exemplo, para um dado do tipo string validar se um texto com tamanho

igual ou maior do que o permitido causa exceção na aplicação.

Tipos de dados específicos à aplicação deverão ser analisados em conjunto com a equipe de engenharia de testes para gerar entradas e definir o método de prova.

Assim, esse estudo de caso propõe que cada atividade deverá conter no mínimo dois conjuntos de dados de entrada, o chamado "caminho feliz", cujos dados são os ideais e corretos no processo, e "caminho tortuoso"em que algum tipo de exceção pode ocorrer. Os casos de testes poderão ser utilizados posteriormente para testes de sistema, com o propósito de verificar o correto funcionamento da aplicação.

Para aplicações em que, devido o tamanho do conjunto de dados de entrada, possam ocorrer um número muito grande de combinações de valores ficará a encargo da equipe de testes e negócios definir o escopo dos campos a serem testados. Assim, espera-se que a equipe limite o escopo a um número plausível de dados de entrada ao caso de teste.

4.1.3 Mapeamento de eventos

Os elementos do tipo "eventos"podem ser de início, intermediário ou fim. Os tipos de eventos intermediários considerados neste trabalho são apenas os que podem gerar alguma mensagem ao usuário final.

Esses serão mapeados para um campo chamado "condições", análogo ao campo "dependências"de (STANDARD, 2008), caso o evento dispare uma atividade, ou para o campo "consequência", caso a atividade dispare o evento. Isso possibilitará que, principalmente eventos de interface gráfica como notificações ou *pop-ups* estejam explícitos e contidos em casos de teste (SOUSA et al., 2014). O campo foi nomeado como "condições", para seguir a proposta de (SOUSA et al., 2014).

O campo "condições"possibilitará a visualização de forma explícita que, por exemplo, um sub-processo é disparado em um processo ponta-a-ponta, acarretando a utilização obrigatória de eventos de início em um modelo. O evento de fim é obrigatório, para mapear corretamente o campo de "critério de aprovação / reprovação"de um desenho de teste de nível. O mesmo é feito na tabela proposta por Sousa et al. (2014).

O objetivo é facilitar, não só para o executor, como também ao organizador dos planos de testes a seleção de casos de teste que melhor se adequa ao escopo das provas.

4.1.4 Mapeamento de desvios

Os desvios serão mapeados conforme as heurísticas de padrão de *workflow* de (SOUSA et al., 2014). Esses indicam as regras de negócio que a aplicação deverá suportar aos usuários finais. Eles serão as principais fontes para definir um desenho de teste de nível para testes integrados aos modelos de processos de negócio, análogo ao campo "fluxo envolvido"da tabela proposta por (SOUSA et al., 2014).

Baseado nesses elementos, o documento de desenho de teste de nível poderá ser preenchido. Seguindo a especificação proposta em (STANDARD, 2008) e (SOUSA et al., 2014), o campo "funcionalidade a ser testada"conterá os nomes dos casos de testes obtidos nas regras anteriores (nome das atividades do processo).

Um campo adicional chamado "caminho"será utilizado, para auxiliar na validação de que todas as possibilidades de desvio estarão sendo contempladas, conforme critério de cobertura proposto em (SOUSA et al., 2014). Esse será preenchido pelos identificadores dos casos de teste. Caso não existam desvios no modelo, um único caminho será considerado, sendo possível alterar os casos de testes envolvidos gerando, portanto diferentes desenhos de testes de nível. Para desvios com função de *join*, não foram propostas heurísticas especiais em (SOUSA et al., 2014).

O documento resultante desta etapa, o desenho de teste de nível, representará apenas um dos caminhos possíveis descritos no modelo de processo, de modo que o campo "identificação do desenho de teste de nível"deverá ser preenchido por um identificador único. O mesmo acontece em (SOUSA et al., 2014), onde para cada caminho possível os autores descrevem no campo "fluxo envolvido"quais as atividades serão executadas.

Será possível fazer uso dos identificadores dos documentos resultantes dessa etapa, para documentos de planos de testes de nível, visando a validação de um processo ponta-a-ponta posteriormente.

Um resumo contendo as heurísticas propostas por (SOUSA et al., 2014) utilizadas no mapeamento do modelo de processo para casos de teste neste estudo de caso é apresentado na tabela 4.2.

4.1.5 Exemplo ilustrativo de mapeamento completo das regras propostas

Com o propósito de ilustrar as regras propostas em (SOUSA et al., 2014) de mapeamento ao modelo de documento de caso de teste proposto ao estudo de caso, são

Tabela 4.2: Processo de mapeamento de modelo de processo em BPMN a casos de teste

Regra	*Descrição*
1	Selecionar as atividades candidatas para casos de teste unitários: 1.1 removendo as atividades manuais do escopo de teste. 1.2 mapeando as atividades de usuário e de serviço a seus casos de testes unitários
2	Definir os dados utilizados nos casos de testes unitários: 2.1 particionando por equivalência os dados do tipo inteiros ou *strings* 2.2 verificando a necessidade da geração de dados de entrada e saída às estruturas de dados específicas da aplicação 2.3 Mapear os dados de entrada aos campos corretos da interface gráfica da aplicação no caso de teste.
3	Se houver, mapear os eventos para os devidos campos de condições e elementos alvo dos casos de testes unitários.
4	Se houver, mapear os desvios compostos pelos casos de testes unitários que serão utilizados nos caminhos possíveis ao processo de negócio.

Fonte: Adaptado de (SOUSA et al., 2014)

apresentados o modelo BPMN e descrição textual de um processo extraído de (KHADER; YOUSEF, 2016). A descrição textual do processo, por não conter detalhamento suficiente em (KHADER; YOUSEF, 2016) foi adaptada, com o intuito de prover as informações necessárias para os testes unitários das atividades.

O registro de um procedimento é um processo real extraído da reitoria de pesquisas acadêmicas da Universidade da Jordânia por Khader and Yousef (2016) e é retratado textualmente da seguinte maneira: um usuário acessa o sistema X; verifica os procedimentos existentes em um período de tempo definido pelos campos "Data: a partir de"e "Data: Até", que podem ser preenchidos por números válidos correspondentes aos meses existentes, de 1 a 12; uma lista de procedimentos existentes no sistema cadastrado ao período de busca é disponibilizada ao usuário; o usuário insere um novo procedimento indicando seu nome com limite máximo de 8 caracteres e mínimo de 1 caracter;

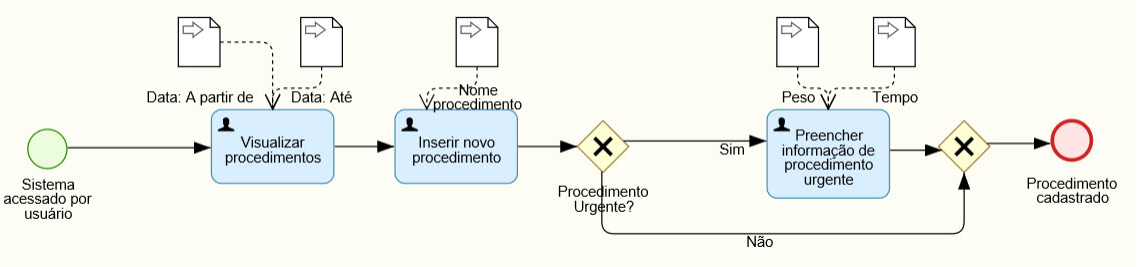
1. Se o procedimento é urgente, preenche peso do procedimento e o tempo estimado do procedimento com números inteiros maiores que zero; aparece um *pop-up* com mensagem de que cadastro de procedimento foi realizado com sucesso.

2. Caso contrário cadastra automaticamente o procedimento, aparecendo um *pop-up* com confirmação de cadastro de procedimento com sucesso.

O modelo BPMN do processo de registrar procedimento está ilustrado na figura 4.1.

Para o modelo acima, inicia-se aplicando a regra de mapeamento das atividades: selecionam-se as atividades candidatas (do tipo de usuário ou de serviço) e criam-se os

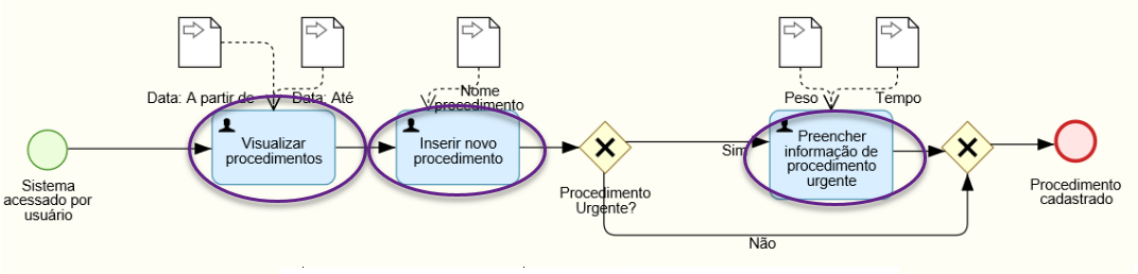
Figura 4.1: Modelo BPMN de processo para registrar procedimentos



Fonte: Adaptado de Khader and Yousef (2016)

casos de teste relacionados a essas. A figura 4.2 ilustra o resultado dessa atividade.

Figura 4.2: Exemplo de mapeamento de atividades para casos de teste



Identificador do caso de teste: 1
Nome atividade: Visualizar Procedimentos

Identificador do caso de teste: 2
Nome atividade: Inserir novo procedimento

Identificador do caso de teste: 3
Nome atividade: Preencher Informação de procedimento urgente

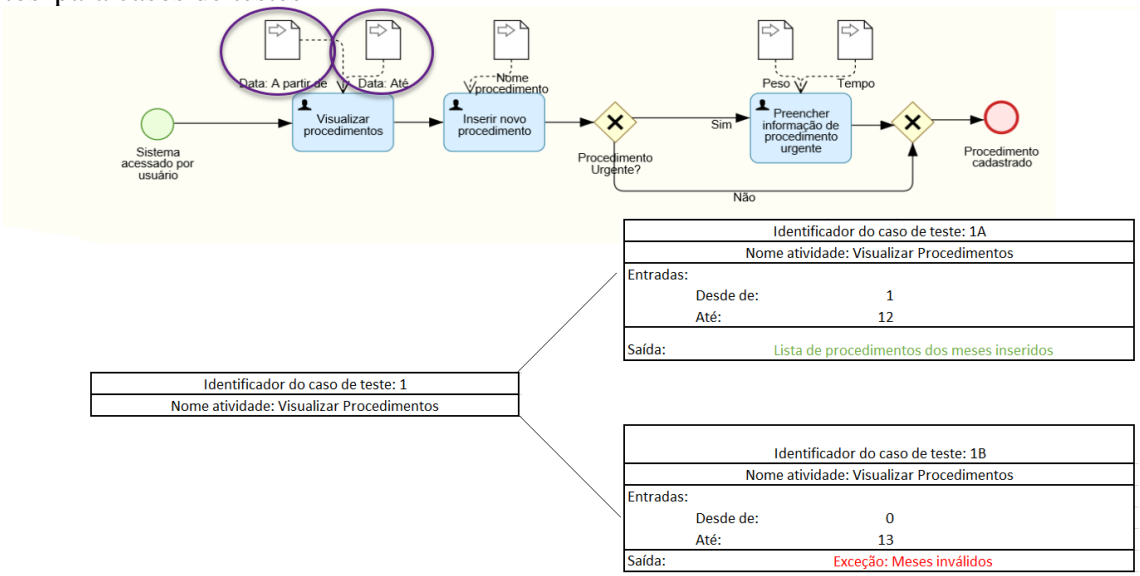
Fonte: Adaptado de Khader and Yousef (2016)

Após o mapeamento das atividades, realiza-se o mapeamento dos dados. O uso de anotações neste passo pode ser de extrema ajuda, para facilitar a definir quais serão os dados de entrada para os testes unitários. As partições utilizadas para a geração dos casos de teste foram propostas pelo autor, por meio da adaptação do modelo de processo textual.

A figura 4.3 ilustra o mapeamento de dados relacionados à atividade "visualizar procedimentos"para casos de teste. Para esse passo, foi necessário a utilização do modelo textual, para saber quais seriam os números válidos. Os números 1 e 12 são válidos, pois correspondem a meses existentes. Números 0 e 13 devem gerar uma exceção ao sistema, por tratar de meses inexistentes.

O mapeamento de dados da atividade "inserir novo procedimento"cria três casos

Figura 4.3: Exemplo de mapeamento de dados da atividade "visualizar procedimentos"para casos de testes



Fonte: Adaptado de Khader and Yousef (2016)

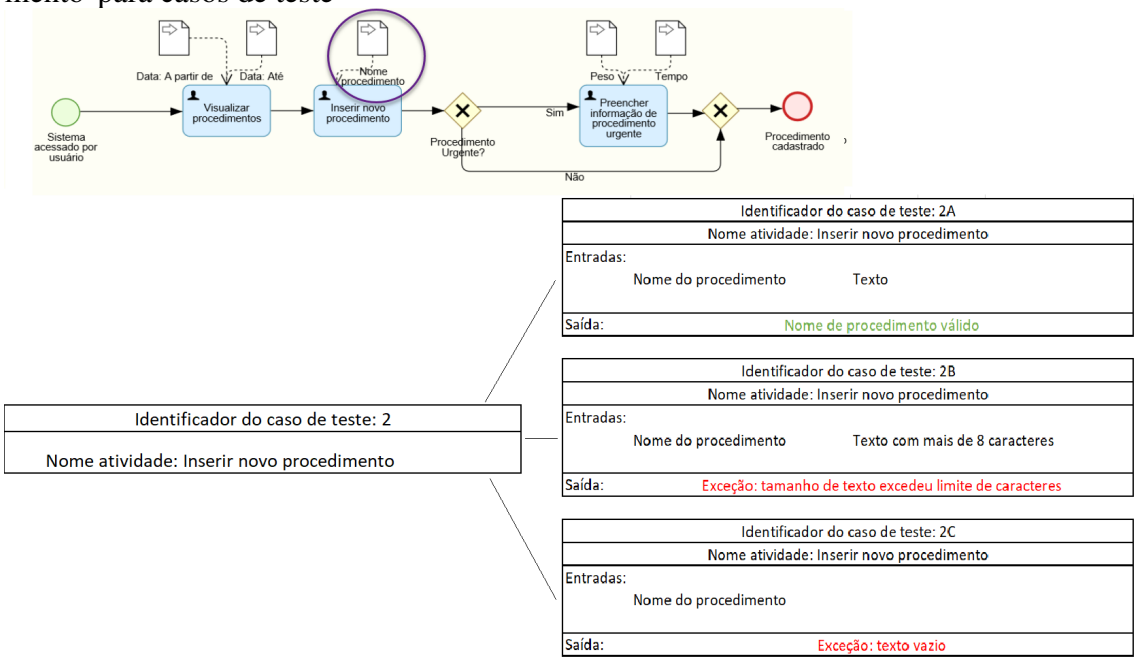
de teste, para verificar as exceções que podem ocorrer seguindo o modelo textual do processo. O nome do procedimento, de acordo com a descrição, deve ter no mínimo 1 e no máximo 8 caracteres. Assim, um caso de teste é gerado para testar o funcionamento correto da atividade com campo preenchido pelo texto "Texto"(*string* não vazia, com menos de 8 caracteres), e dois para acionar as necessárias exceções (um com *string* com número de caracteres maior que 8, e outro com o campo nome com *string* vazio). Os casos de teste são ilustrados na figura 4.4.

Para a atividade "preencher informações de procedimento urgente"são utilizados os números 0 e 10 para teste de peso e tempo, pois, conforme descrição, o sistema deve aceitar apenas números inteiros maiores que 0. Dessa forma, as partições utilizadas foram maior que 0 (para testes com saída esperada com sucesso) e menor ou igual a 0 (para testes de exceções), levando à geração de três casos de teste. A figura 4.5 ilustra os casos de teste. O número 10 poderia ser substituído por qualquer número pertencente ao conjunto dos Naturais Não-Nulos.

Os dados escolhidos ao mapeamento foram escolhidos pelo autor do presente trabalho para mera ilustração. Foram escolhidos valores de fronteiras das partições de equivalência numérica, por se tratarem de valores que costumam apontar mais erros, segundo Chauhan and Singh (2014).

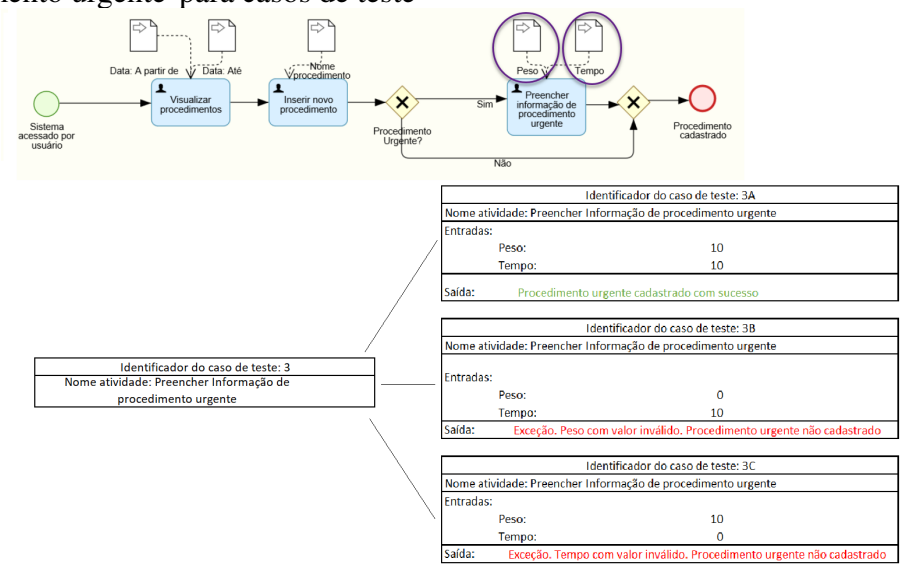
No modelo de processo de (KHADER; YOUSEF, 2016), não há eventos intermediários a serem mapeados. De acordo com as regras propostas em (SOUSA et al., 2014),

Figura 4.4: Exemplo de mapeamento dos dados da atividade "inserir novo procedimento"para casos de teste



Fonte: Adaptado de Khader and Yousef (2016)

Figura 4.5: Exemplo de mapeamento dos dados da atividade "preencher informação de procedimento urgente"para casos de teste



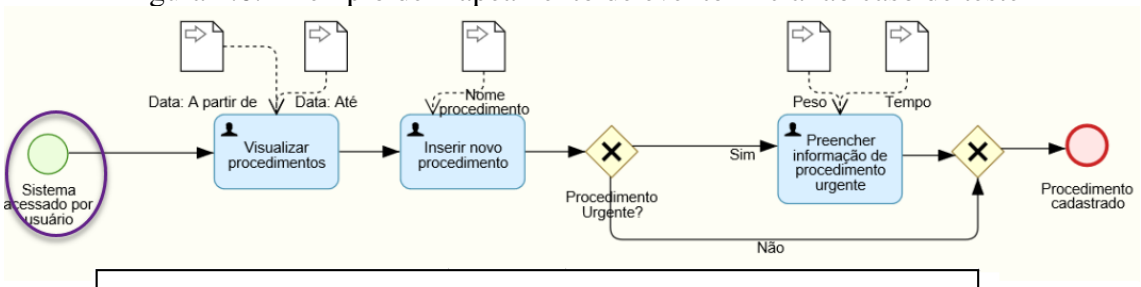
Fonte: Adaptado de Khader and Yousef (2016)

o evento inicial é mapeado ao campo "condições"do caso de teste da atividade "visualizar procedimentos". O evento final é mapeado ao campo de "consequência"do caso de teste da atividade "preencher informação de procedimento urgente", e ao campo "critério de aceitação"nos documentos de níveis de teste, que contêm o caminho feliz do processo.

Da mesma forma o critério de rejeição do documento de nível de teste poderá ser preenchido com a negativa do evento de cadastro com sucesso.

A figura 4.6 exemplifica o mapeamento do evento inicial ao campo "condições"do documento de caso de teste da atividade "visualizar procedimentos". A figura 4.7 ilustra o mapeamento do evento final ao campo "consequências"da atividade preencher "informação de procedimento urgente". Enfim, a figura 4.8 ilustra o mapeamento do evento final como critério de aceitação de um documento de desenho de teste de nível.

Figura 4.6: Exemplo de mapeamento de evento inicial ao caso de teste



Identificador do caso de teste: 1A	
Nome atividade: Visualizar Procedimentos	
Entradas:	
Desde de:	1
Até:	12
Saída:	Lista de procedimentos dos meses inseridos

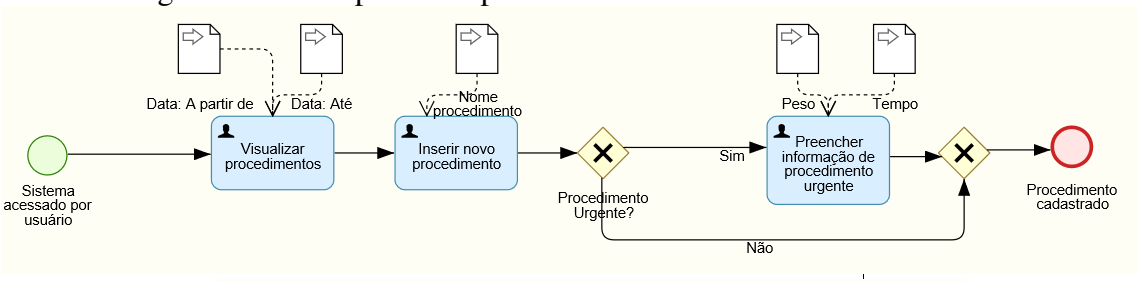
Identificador do caso de teste: 1A	
Nome atividade: Visualizar Procedimentos	
Entradas:	
Desde de:	1
Até:	12
Saída:	Lista de procedimentos dos meses inseridos
Condições	Usuário acessa sistema

Fonte: Adaptado de Khader and Yousef (2016)

Após esse passo, temos mapeado o modelo de processo completo para documentos de casos de testes unitários e um esboço inicial dos documentos de níveis de teste, que poderão integrar planos de níveis de testes posteriormente.

Os desvios são mapeados diretamente aos documentos de desenhos de níveis de teste. O campo "caminho"do documento de desenho de nível de teste conterá os identificadores dos casos de testes obtidos nas etapas anteriores. Os nomes das atividades serão mapeadas ao campo "funcionalidades testadas". Dessa forma, para modelos com desvio exclusivo, haverá sempre no mínimo dois documentos de desenho de nível de teste, a fim

Figura 4.7: Exemplo de mapeamento de evento final ao caso de teste

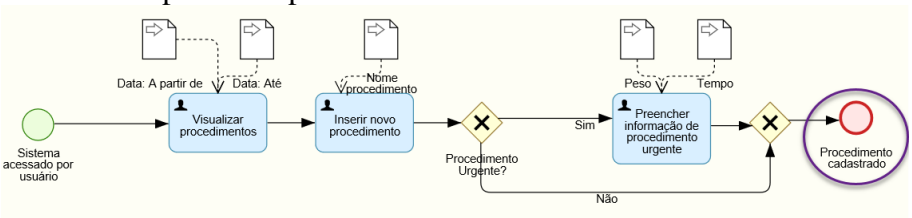


Identificador do caso de teste: 3A	
Nome atividade: Preencher Informação de procedimento	
Entradas:	
Peso:	10
Tempo:	10
Saída:	Procedimento urgente cadastrado com sucesso

Identificador do caso de teste: 3A	
Nome atividade: Preencher Informação	
Entradas:	
Peso:	10
Tempo:	10
Saída:	Procedimento urgente cadastrado com sucesso
Consequência:	Pop-up com mensagem de cadastro com sucesso

Fonte: Adaptado de Khader and Yousef (2016)

Figura 4.8: Exemplo de mapeamento de evento final ao desenho de nível de teste



Identificador de desenho de nível de teste: 1A
Critério de aceitação: Procedimento cadastrado com sucesso

Fonte: Adaptado de Khader and Yousef (2016)

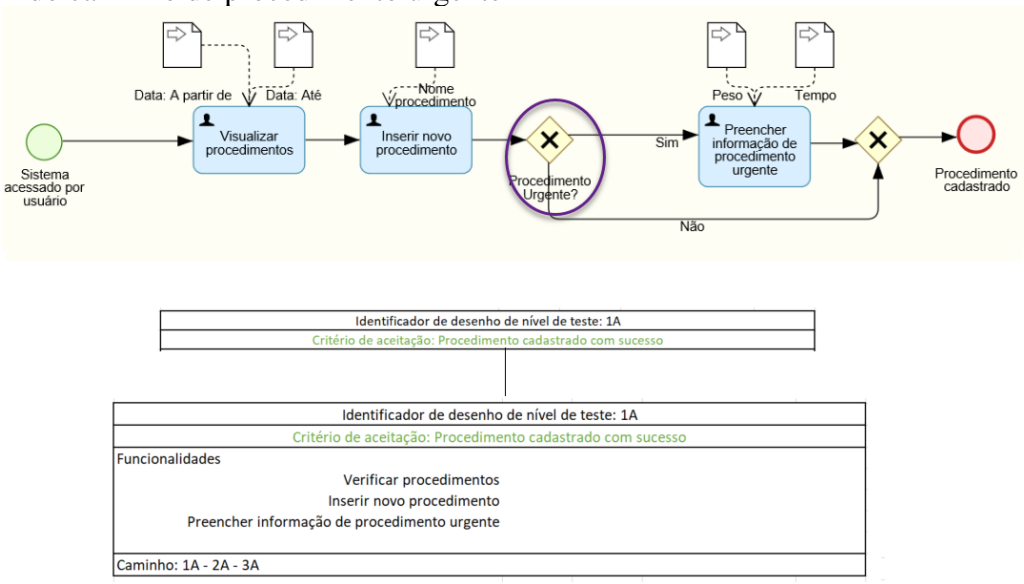
de cobrir todos os caminhos possíveis e aumentar a cobertura de teste, consequentemente a qualidade de software.

As figuras 4.9 e 4.10 ilustram o mapeamento do desvio a documentos de desenho de nível de teste, juntamente com os campos acima citados preenchidos. Nas figura, ambos os caminhos possíveis do modelo estão contemplados, seguindo o chamado "caminho feliz", onde todos as entradas providas por usuários estão de acordo com o esperado.

Os casos de testes poderão ser utilizados para testar as diferentes entradas possíveis dos usuários, a fim de melhorar, tanto o processo, quanto a aplicação. Um exemplo de melhoria ao processo proposto por (KHADER; YOUSEF, 2016) seria a utilização de

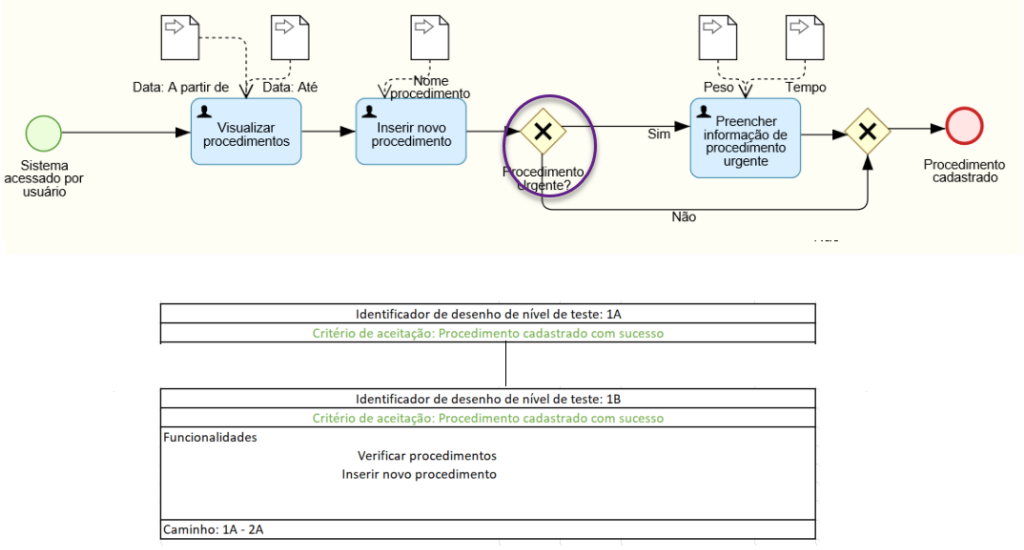
um evento informando que o nome de procedimento vazio ou com limite de caracteres excedido é inválido.

Figura 4.9: Exemplo de mapeamento de desvio a documento de desenho de teste de nível seguindo caminho de procedimento urgente



Fonte: Adaptado de Khader and Yousef (2016)

Figura 4.10: Exemplo de mapeamento de desvio a documento de desenho de teste de nível seguindo caminho de procedimento comum



Fonte: Adaptado de Khader and Yousef (2016)

5 EXPERIMENTAÇÃO E VALIDAÇÃO DO MÉTODO PROPOSTO

Este capítulo apresenta os experimentos realizados com funcionários da empresa SAP de São Leopoldo graduandos e graduados, não só em cursos relacionados à TI, mas também voltados às áreas de Administração, Economia e Egenharia de Produção, para realizar as atividades de extração de casos de teste de software a partir de um modelo de processo na notação BPMN.

A seção 5.1 descreve as metodologias aplicadas nos experimentos, enquanto a seção 5.2 aponta os resultados encontrados por meio daqueles.

5.1 Metodologia adotada

Um experimento controlado foi realizado para validar a hipótese de que com as regras propostas por (SOUSA et al., 2014) e com o formato de documento sugerido no estudo de caso haveria uma maior cobertura de testes às atividades do modelo de processo, tanto por pessoas de áreas funcionais, quanto de áreas técnicas, do que sem a utilização das regras. Um tutorial[1] foi disponibilizado aos participantes do experimento apresentando definições sobre elementos notacionais da BPMN e sobre documentação para teste de software.

O método utilizado para verificação da hipótese foi o teste T-pareado, já que esse método propicia comparar o desempenho obtido por um só grupo executando o mesmo experimento, porém com ferramentas diferentes. Corroborado por Wohlin et al. (2012), "se duas ferramentas são comparadas em grupos independentes, é possível utilizar apenas o teste T em cada grupo, porém se quiser medir a diferença de desempenho de um mesmo grupo utilizando ferramentas distintas, se utiliza, então, o teste T-pareado."

No total foram 16 pessoas que realizaram o experimento. Da população que realizou o experimento, 8 participantes eram de áreas relacionadas à TI, e o restante relacionados a áreas funcionais. Dentre os participantes, existiam pessoas já graduadas assim como graduandos. Um formulário via Google forms foi respondido, a fim de verificar o conhecimento dos participantes, tanto em BPMN, quanto em teste de software.

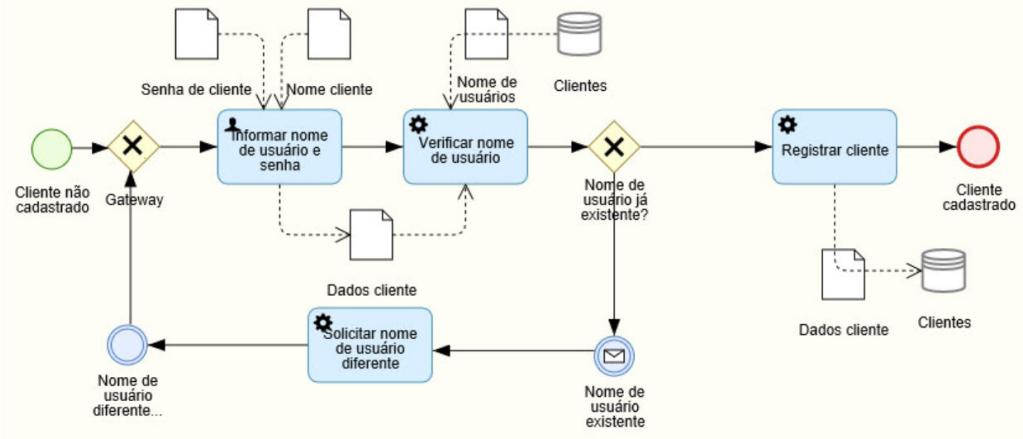
Para a extração de casos de teste, foi utilizado um processo adaptado de (SOUSA et al., 2014) de registro de usuários em um sistema ilustrado na figura 5.1. Nenhum dos participantes conhecia o processo ou havia tido contato com o texto de (SOUSA et al.,

[1] https://docs.google.com/document/d/1F6XVy0qMZzZfwnJp4LjV-U5XulU8TOdtDrS1x8ES7iM/edit?usp=sharing

2014) anteriormente.

Foi escolhido esse processo simples, uma vez, que em (SOUSA et al., 2014), os autores apresentam o número doze como número de casos de teste que acreditam cobrir todas as atividades do modelo. O número mínimo de casos de testes, porém, de acordo com os autores seria quatro, para cobrir unitariamente as atividades do modelo. O modelo de processo de cadastro de usuários é ilustrado na figura 5.1.

Figura 5.1: Modelo de processo cadastrar usuários



Fonte: (SOUSA et al., 2014)

5.1.1 Método estatístico do teste T-Pareado

A fórmula e notação do teste T-pareado são apresentadas nas imagens 5.2 e 5.3.

Figura 5.2: Fórmula e descrição do teste T-pareado

Fórmula

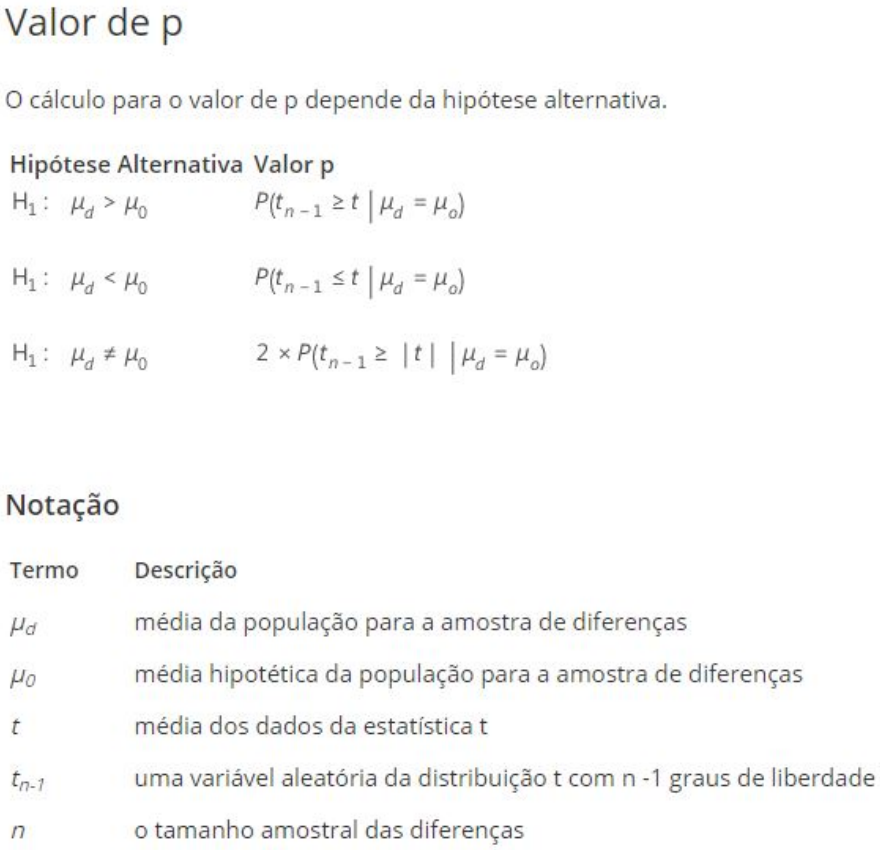
$$t = \frac{\bar{d} - \mu_0}{s_d / \sqrt{n}}$$

Notação

Termo	Descrição
μ_0	média hipotética da população das diferenças
$\bar{d}$	média das diferenças da amostra pareada
s_d	desvio padrão da amostra das diferenças da amostra pareada
n	tamanho amostral

Fonte: (MINITAB, 2018)

Figura 5.3: Fórmula e descrição hipóteses e valor de P



Valor de p

O cálculo para o valor de p depende da hipótese alternativa.

Hipótese Alternativa	Valor p
$H_1: \mu_d > \mu_0$	$P(t_{n-1} \geq t \mid \mu_d = \mu_o)$
$H_1: \mu_d < \mu_0$	$P(t_{n-1} \leq t \mid \mu_d = \mu_o)$
$H_1: \mu_d \neq \mu_0$	$2 \times P(t_{n-1} \geq \lvert t \rvert \mid \mu_d = \mu_o)$

Notação

Termo	Descrição
μ_d	média da população para a amostra de diferenças
μ_0	média hipotética da população para a amostra de diferenças
t	média dos dados da estatística t
t_{n-1}	uma variável aleatória da distribuição t com n -1 graus de liberdade
n	o tamanho amostral das diferenças

Fonte: (MINITAB, 2018)

A fórmula depende das hipóteses nula e alternativa. A hipótese nula utilizada neste trabalho é a de que o "número de atividades do modelo do experimento cobertas por casos de testes é maior sem o uso das regras de mapeamento propostas em (SOUSA et al., 2014)". Consequentemente, a hipótese alternativa é a de que "com as regras propostas pelos autores e com a utilização do modelo de documento de casos de teste proposto no presente trabalho haveria uma maior cobertura de testes às atividades do modelo de processo de negócio".

Assim, é possível comparar o valor t obtido com a taxa crítica obtida na tabela T-student utilizando grau de liberdade[2] igual a 15. A taxa crítica representa a significância estatística, ou seja, a probabilidade de rejeitar a hipótese nula, mesmo que essa seja verdadeira.

A significância (alfa) utilizada no trabalho foi obtida atrvés das considerações de Moore David S. and Craig (2009). Segundo os autores, o tamanho da amostra influencia diretamente o valor de alfa. Para amostras pequenas, utiliza-se uma significância maior,

[2]Grau de liberdade é calculado com número de participantes (n) subtraído de 1. Assim, com 16 participantes, n-1 é igual a 15.

a fim de evitar falsos positivos e falsos negativos. Os autores consideram como valores padrões de significância para o método 1% e 5% .

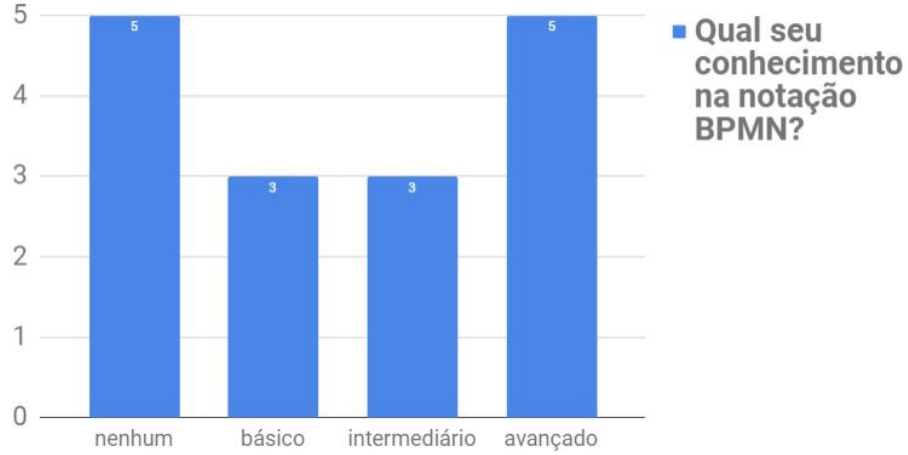
Dessa forma, utilizou-se o valor de 5% para siginificância, visto que a amostra era de apenas 16 pessoas. Esse valor foi também utilizado em (WOHLIN et al., 2012), para seus exemplos que possuíam em média 10 indivíduos de amostra.

5.2 Resultados obtidos

O roteiro aplicado no experimento foi elaborado, a fim de avaliar a capacidade de os participantes obterem casos de teste a partir de um modelo de processo na notação BPMN. Inicialmente eles tentaram cobrir as atividades do processo gerando casos de teste sem o uso das regras de mapeamento, e depois com.

Um questionário para medir os conhecimentos em BPMN e teste de software foi respondido pelos participantes. Esse questionário objetivava saber se os casos de testes obtidos inicialmente sem as regras de mapeamento derivavam da experiência dos participantes. As figuras 5.4 e 5.5 ilustram os resultados do questionário.

Figura 5.4: Conhecimento dos participantes em BPMN

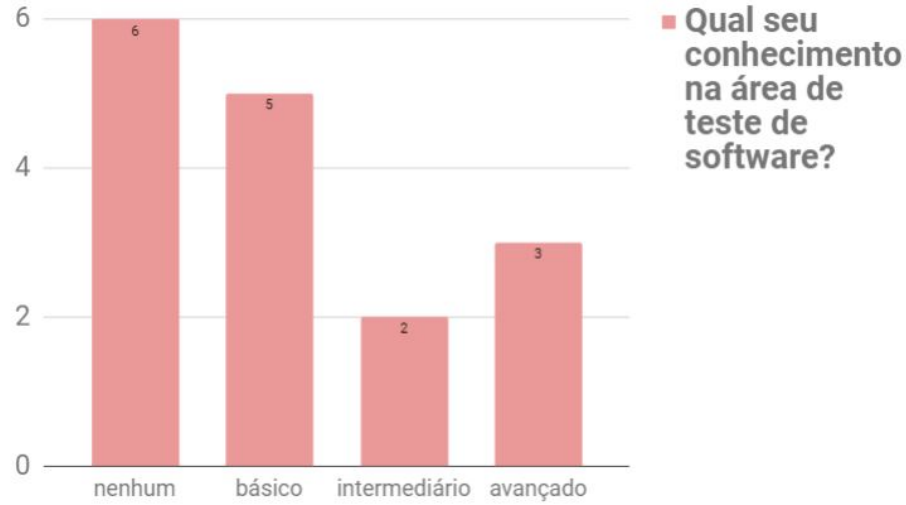


Fonte: O Autor, 2018

Os resultados numéricos de casos de testes obtidos a partir das regras de mapeamento se encontram na figura 5.6. É possível verificar um aumento no número da extração de casos de teste na maioria dos participantes. Os participantes de 1 a 8 eram as pessoas relacionadas a áreas funcionais, enquanto os de 9 a 16 de áreas técnicas.

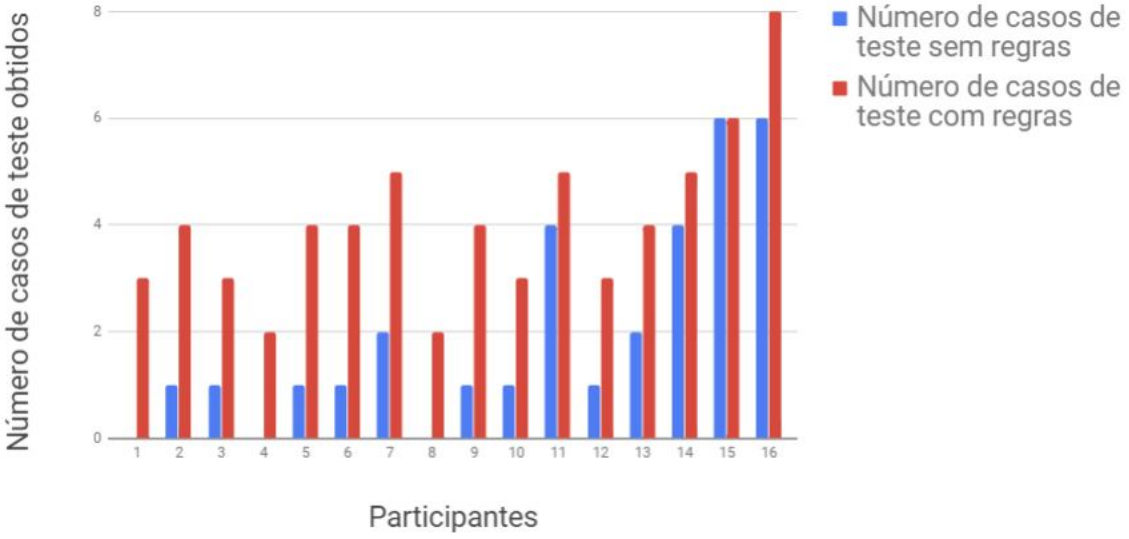
Na figura 5.7 são apontados os números de atividades cobertas pelos casos de testes obtidos pelos participantes. É possível ver um aumento no número de atividades

Figura 5.5: Conhecimento dos participantes em teste de software



Fonte: O Autor, 2018

Figura 5.6: Número de casos de teste obtidos sem e com o uso das regras pelos participantes



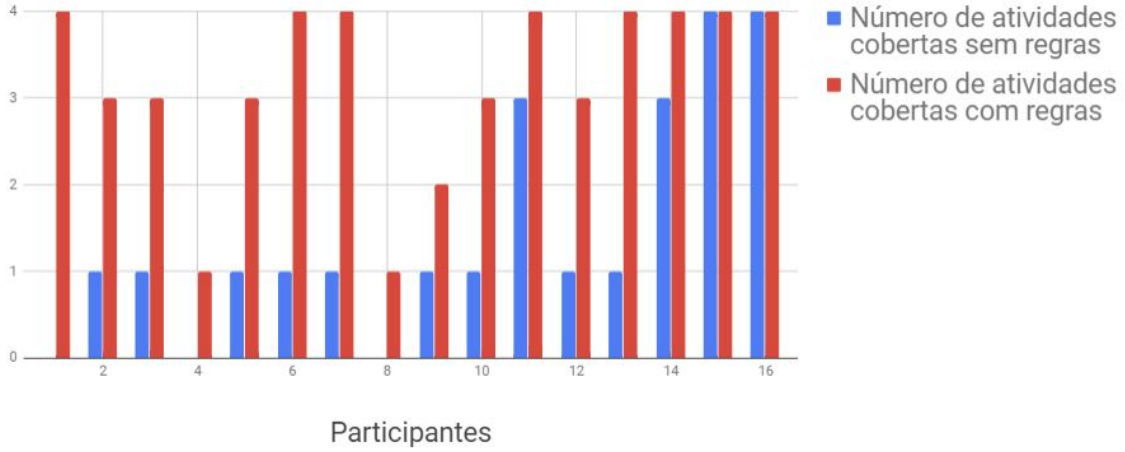
Fonte: O Autor, 2018

cobertas, não só pelos participantes pertencentes a áreas funcionais, mas também daqueles pertencentes a àreas técnicas.

O resultado (t-obs) obtido para função do teste T-pareado, a fim de medir se de fato houve um acréscimo no número de atividades cobertas obtido foi de -6.219662. T-obs deveria ser menor que -1,7531, para rejeitar a hiótese nula e aceitar a hipótese alternativa. Dessa forma, a hipótese alternativa de que um número maior de atividades seria coberta com o uso das regras de mapeamento foi validada.

A realização de um experimento controlado presencial foi um fator de extrema importância que possibilitou a análise de diversos pontos no estudo. O número de atividades

Figura 5.7: Número de atividades cobertas por casos de teste obtidos sem e com o uso das regras pelos participantes



Fonte: O Autor, 2018

que os participantes de áreas não relacionadas à TI mapearam sem o uso das regras nunca foi maior que 1. Isso devido ao fato de que a atividade que mais eram familiarizados era a "Informar nome de usuário e senha", visto que trabalham rotineiramente com aplicações que necessitam a entrada de dados.

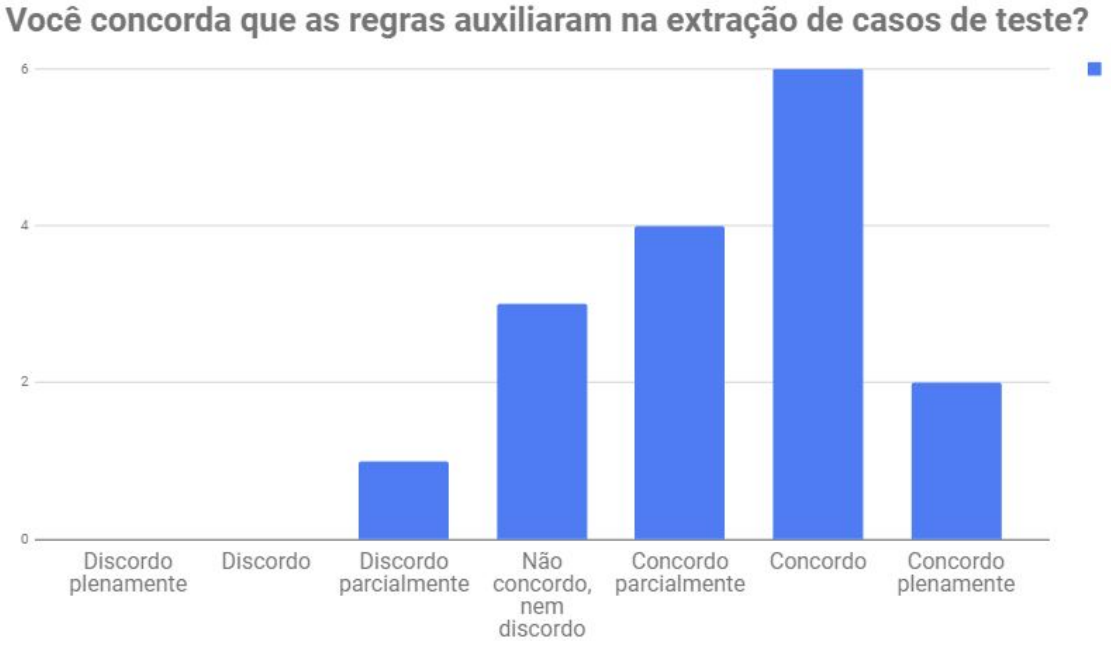
Assim sendo, muitos cobriram inicialmente essa atividade, mesmo sem terem qualquer conhecimento em teste de software, evidenciando que para atividades funcionais esses participantes podem agregar em muito na elaboração de casos de teste durante o desenho do processo, e, portanto, nos estágios iniciais do desenvolvimento de uma aplicação.

Um número bastante considerável de 3 atividades cobertas foi obtido. Pôde ser verificado que a atividade de sistema "solicitar nome de usuário diferente"não foi diversas vezes considerada. Perguntado aos participantes o porquê, muitos consideraram que a falta da conexão de um objeto de dados à atividade dificultou a compreensão de que a atividade deveria ser considerada na cobertura de casos de teste. É perceptível, portanto, que diversos participantes tiveram dificuldade em mapear regras do tipo desvio e evento.

Por outro lado, o público relacionado a áreas de TI obteve em sua maioria o número total das atividades mapeadas a casos de teste. Foi posível verificar que, especialmente no mapeamento de eventos, os participantes de TI tiveram maior facilidade do que os das áreas funcionais.

Após o experimento, foi aplicado um questionário, a fim de medir a satisfação dos participantes em relação às regras. A figura 5.8 aponta os resultados númericos dessa pesquisa.

Figura 5.8: Satisfação em relação às regras de mapeamento



Fonte: O Autor, 2018

Dentre os 8 participantes de áreas funcionais nenhum discordou do auxílio das regras propostas em (SOUSA et al., 2014). Dois participantes desses não concordaram nem discordaram do auxílio das regras de (SOUSA et al., 2014) e do modelo de documento do estudo de caso, pois acreditam que não deverão no futuro utilizá-los em suas futuras atividades.

Dos participantes relacionados a áreas de TI, o participante com maior conhecimento em BPMN e teste de software foi quem respondeu discordar parcialmente das regras propostas em (SOUSA et al., 2014) e do modelo de documento do estudo de caso. Por já trabalhar na área de teste de software e gerenciamento de processos há 4 anos, o participante acredita que seguir as regras o levou a despender mais tempo para encontrar os mesmos casos de teste que havia encontrado anteriormente.

6 TRABALHOS RELACIONADOS

Neste capítulo, alguns trabalhos relacionados à extração de casos de teste a partir de um modelo de processo serão apresentados. O estudo desses trabalhos ocorreu do início de março de 2018 até o fim de setembro de 2018 pelo autor. Bibliotecas eletrônicas como IEEExplore, ACM e ResearchGate, além de editoras como Springer foram as principais fontes de pesquisa.

Depois da análise de diversos artigos, foi possível perceber que a extração de casos de testes a partir de modelos é algo já consolidado e pesquisado por diversos autores. O uso de notação BPMN para extração de casos de testes, entretanto, não possui um modelo padrão a ser seguido, e cada autor propõe uma maneira diferente para desenvolver um cenário de teste.

Em (YUAN et al., 2008), os autores propõem um modelo para extrair casos de testes executáveis a partir de um modelo de processo desenhado na notação UML e exportado no padrão BPEL *(Business Process Execution Language)*, uma linguagem padrão OASIS executável para especificar ações de processos de negócio com web services. Nesse trabalho, os autores utilizam um diagrama de sequência que é, então, transformado em código executável. A descrição de heurísticas utilizadas para extração dos casos de testes, contudo, não estava clara no artigo. Esse serviu, portanto, para investigar referências capazes de evidenciar maneiras de extrair casos de testes para modelos de processos.

O artigo de (LOUMOS et al., 2010) foi de suma importância no estudo deste trabalho, por propôr um desenvolvimento de *software* através de uma metodologia orientada a modelo de processos de negócio, onde o código fonte de um sistema é gerado baseando-se nas atividades executadas em um processo. De acordo com Loumos et al. (2010), pesquisas mostram que aproximadamente 80% da implementação de projetos que são relacionados a diversos setores de Informação da Comunicação e Tecnologia (ICT) falham em atingir seus objetivos iniciais, devido à análise inadequada de requerimentos, estimativas errôneas ou desenvolvimento inadequado de um sistema.

Assim, os autores propõem desenvolver um sistema por meio de casos de testes extraídos de um processo de negócio e métricas específicas *(Key Performance Indicators - KPIs)* das atividades, para medir a eficiência do sistema e a usabilidade dos usuários. (LOUMOS et al., 2010) acreditam que, por meio da coleta dessas métricas, é possível adquirir informações de uma devida atividade do processo a qual está demasiada lenta ou ineficiente, e identificar se o problema está em nível de usabilidade da ferramenta por

parte dos usuários finais ou de performance da aplicação.

Em (CAI, 2011), um cenário de testes de processo de negócio baseado em composição de casos de teste é proposto. O método se concentra em encontrar os erros lógicos de implementação de um processo de negócio, de acordo com a apresentação formal desse. Os casos de testes obitdos são provas manuais que podem ser reutilizados em outros cenários. O modelo utilizado para exposição do caso de teste é no formato de redes de Petri. O trabalho de Cai (2011) será descrito mais detalhadamente em uma subseção, por prover heurísticas de como extrair casos de testes manuais a partir de um modelo BPMN.

Em (LüBKE; LESSEN, 2016), é descrito como implementaram de fato um sistema utilizando a metodologia *Behavior-Driven Development (BDD)* gerando casos de testes automatizados descritos em notação BPMN. BDD mira estabelecer comunicação baseada em exemplos entre stakeholders de negócio e desenvolvedores aplicando uma linguagem específica de domínio *(Domain Specific Language - DSL)* compreensível ao negócio. Essa DSL tipicamente emprega o padrão "contexto-ação-resultado", para descrever o cenário. Contexto descreve a pré-condição. Ação a atividade que será realizada, caso a pré-condição seja satisfetia. Resultado define asserções para validar se as ações resultaram no estado desejado. Maioria das DSL de BDD são textuais e utilizam palavras como: "dado"para pré-condição, "quando"para ações, "então"para asserções. Por exemplo, "Dado que um usuário acessa o sistema, quando o usuário selecionar a parcela para buscar e pressionar ok, então um arquivo no formato pdf deverá ser carregado"

Para modelar os processos de negócio executáveis, foi utilizada a linguagem BPEL, e para os modelos analíticos de negócio (LüBKE; LESSEN, 2016) utilizaram a notação BPMN. Os autores destacam que, a fim de extrair os casos de testes corretamente, os processos de negócios deveriam estar o mais detalhados e inteiramente corretos, isto é, nenhuma atividade poderia estar faltando, e atividades desnecessárias não deveriam estar contidas no modelo.

Em (LüBKE; LESSEN, 2016) não foi realizada a extração de casos de teste via modelo em BPMN, mas sim a representação de casos de teste utilizando um sub-conjunto de elementos dessa notação, uma vez que, segundo Lübke and Lessen (2016), isso possibilitava a fácil compreensão de todos os envolvidos do objetivo e o que estava sendo testado no caso modelado. Isso demonstra que modelos em BPMN podem ser utilizados, tanto como insumo para elaboração de casos de teste em fases iniciais de desenvolvimento, quanto para modelar os casos de teste, para que todos os stakeholders do sistema saibam o comportamento esperado daqueles.

Os trabalhos de ((MOURA et al., 2017) e (NONCHOT; SUWANNASART, 2016)) fazem uso do trabalho de (YOTYAWILAI; SUWANNASART, 2014), a fim de definir heurísticas e elaborar propostas de ferramentas capazes de gerar casos de teste automatizados a partir de um modelo em BPMN. O trabalho de (YOTYAWILAI; SUWANNASART, 2014), portanto, será descrito em uma subseção com maior detalhes, para demonstrar a forma de extração de testes automatizados funcionais obtidos pelos autores.

Em (SOUSA et al., 2014), é proposto um conjunto de heurísticas capazes de extrair casos de testes manuais a partir de um modelo de process em BPMN. Ainda que as heurísticas não tenham sido validadas por alguma forma de experimento, o trabalho de Sousa et al. (2014) será detalhado abaixo por fornecer regras explícitas e um passo a passo para a obtenção dos casos de teste.

Assim como em (CAI, 2011), (KHADER; YOUSEF, 2016) apresenta um framework que possibilita a obtenção de casos de teste a partir de um modelo em BPMN. A pesquisa de (KHADER; YOUSEF, 2016) objetiva utilizar modelos de processos de negócio no ciclo de vida de desenvolvimento de *software* em seus estágios iniciais, não apenas para prover um superior entendimento do negócio e verificar possíveis melhorias nesse, mas também para gerar casos de teste a partir desses modelos, a fim de aperfeiçoar o desenvolvimento e a testabilidade de *software*. O trabalho, porém, não apresenta claramente a forma de obtenção de casos de teste, especificando de forma genérica o algoritmo utilizado.

Serão descritos em maiores detalhes os trabalhos de (CAI, 2011), (YOTYAWILAI; SUWANNASART, 2014) e (SOUSA et al., 2014). Em (CAI, 2011) há uma proposta de mapeamento de modelos de processos genéricos, isto é, em qualquer notação, a casos de teste. (YOTYAWILAI; SUWANNASART, 2014) propõe regras para extração automatizada de casos de teste a partir de um modelo na BPMN, que podem ser também utilizadas para tarefas manuais. Enfim, em (SOUSA et al., 2014) são propostas heurísticas que serviram como fundamentação para o presente trabalho.

6.1 *A Business Process Testing sequence generation approach based on test cases composition* (CAI, 2011)

Este trabalho propõe um cenário de teste de processos de negócio baseado em composição de casos de teste. O autor sugere que qualquer notação de modelo de processo de negócio pode ser utilizada como insumo para obtenção de casos de teste, e descreve

em Redes de Petri (PETERSON, 1981) a representação desses casos e sua execução para cobertura de testes.

De acordo com o autor, o teste de um processo de negócio pode ser dividido em três níveis: processo de teste para processo de negócio, teste combinatorial para múltiplos parâmtetros, casos de teste para um parâmetro.

O autor sugere que cada cenário de processo de negócio descreve uma sequência específica de ações e interações entre sistemas e usuários. Cada uma dessas interações deve ser pensada como um nodo de um processo, em que uma série de parâmetros pode, ou não, ser necessária para execução. Esses parâmetros devem ser testados por particionamento de equivalência de classes, análise de valor de fronteira ou outras técnias de teste já conhecidas.

Dessa forma, a abordagem do trabalho é a reutilização de casos de testes desenhados para testar cada um dos parâmetros de entrada de um nodo (atividade) de um modelo de processo. De acordo com Cai (2011), essa abordagem provém um grande número de benefícios para aperfeiçoar o processo de desenvolvimento de *software* e a produtividade nesse, auxliando no desenho e execução de testes.

O autor utiliza uma rede de Petri para representação do processo de negócio do qual será extraído os casos de teste. Cai (2011), porém, não descarta diferentes notações para o modelo de processo e cita como exemplos as linguagens UML e BPEL como possíveis de representar o modelo.

No trabalho, um caso de teste é definido como uma operação executável específica que examina todos os aspectos de uma atividade, incluindo entradas e saídas, além de uma descrição detalhada dos passos necessários para sua ocorrência e resultados esperados. O caso de teste é representado por uma Rede de Petri Colorida (RPC) (PETERSON, 1981).

O autor propõe diferentes modelagens em rede de Petri para casos de testes, de acordo com a forma com que a atividade trata os dados e a sequência de estados após sua execução.

Por fim, depois de identificadas as formas que as atividades lidam com dados e seus estágios consecutivos, o autor propõe a composição dos casos de testes através de quatro métodos: composição sequencial, alternativa, paralela e de laço. Essas composições são utilizadas para definir as possibilidades de desvio que um processo pode possuir devido às regras de negócio.

Cada um dos casos de teste é chamado de RCT (Rede de Casos de Teste). A Rede de Cenário de Processos de Negócio (RCPN) é o resultado final obtido pela composição

das RCT. Por fim, o autor sugere um algoritmo de busca em profundidade para cobrir todas as possibilidades do fluxo do processo, para aumetar a cobertura de teste.

6.2 *Design of a tool for generating test cases from BPMN* (YOTYAWILAI; SUWANNASART, 2014)

Este trabalho apresenta uma abordagem para elaborar uma ferramenta capaz de gerar automaticamente casos de teste a partir de um modelo de processo em BPMN. A heurística da cobertura de todos os caminhos é aplicada nos casos de teste para certificar-se da corretude desses.

A ferramenta, de acordo com Yotyawilai and Suwannasart (2014), executaria os seguintes passos, a fim de extrair os casos de teste:

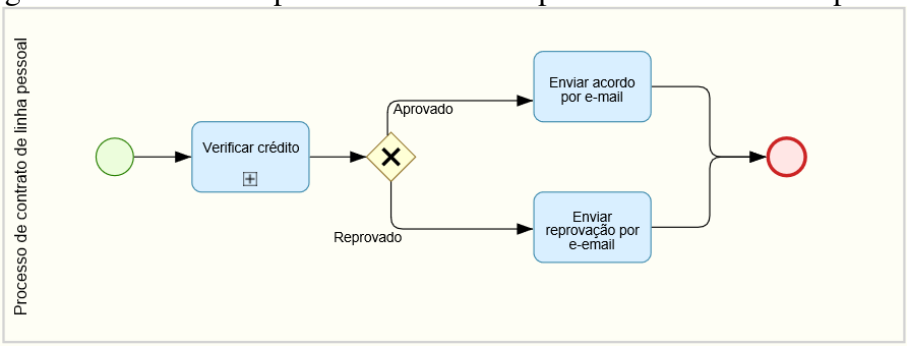
1. Análise das propriedades de variáveis de entrada
2. Adição das propriedades de variáveis de entrada
3. Extração dos dados dos elementos do modelo
4. Criação do grafo de fluxo
5. Criação do caso de teste

Análise das propriedades de variáveis de entrada é o primeiro passo quando o modelo em BPMN é exportado em um arquivo no formato XML. Nessa etapa cada elemento que requer um dado de entrada do usuário deve ser analisado, para verificar a completudo da definição dos dados de entrada. Esses dados de entrada serão mais tarde utilizados para geração dos casos de teste. Para completar as variáveis de um elemento, as seguintes propriedades devem ser definidas: nome de variáveis, tipos das variáveis, tamanho do valor da variável, maior valor e menor valor.

Nome da variável é usado, para referenciar a variável. Os dados de maior e menor valor são utilizados para teste de valores de fronteiras de dados numéricos. Tipos de dados suportados nativamente pela ferramenta seriam *NUMBERS e STRING*. Caso o usuário tenha algum dado específico, deverá criar esse tipo no arquivo XML manualmente.

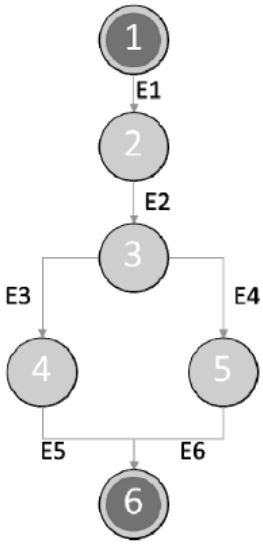
Adição das propriedades de variáveis de entrada é a etapa quando, encontrado que alguma variável de entrada possui definições incompletas, o sistema avisaria o usuário de modo a fazer com que esse especifique as propridades faltantes do elemento. Esse passo é de suma importância, uma vez que a corretude e completude dos elementos do processo de negócio são pré-requisitos para a correta execução da ferramenta.

Figura 6.1: Modelo de processo em BPMN para contratar telefone pessoal



Fonte: (YOTYAWILAI; SUWANNASART, 2014)

Figura 6.2: Grafo obtido pelo sistema a partir do processo contratar telefone pessoal



Fonte: (YOTYAWILAI; SUWANNASART, 2014)

Extração dos dados dos elementos do modelo é a etapa em que o arquivo no formato XML constituído pelas propriedades dos valores de entrada da etapa anterior são extraídos com o uso de um *parser* da aplicação.

Criação do grafo de fluxo é a etapa quando, depois de armazenado o modelo BPMN em um arquivo com os dados necessários exportado em XML, se criam os fluxos possíveis de execução do processo de acordo com suas regras de negócio. Cada elemento é representado como um nodo no grafo, e as arestas direcionadas do grafo indicam os caminhos possíveis de se percorrer.

O grafo é construído a partir dos campos ID do Nodo, Condição e tarRef obtidos da etapa anterior. Para o processo descrito na figura 6.1 seria gerado o grafo correspondente na figura 6.2.

Criação dos casos de teste é a última etapa e inicia utilizando o método de busca

em profundidade, para percorrer o grafo. Se inicia a partir do elemento que representa o evento de início no modelo de processo até encontrar o elemento que representa o evento de fim.

O critério para verificação de sucesso do cenário de teste é verificar se todos os nodos do grafo foram percorridos pelo menos uma vez, a partir dos dados de entrada do usuário nas etapas anteriores.

6.3 *Extraction of test cases from business process models* (SOUSA et al., 2014)

Este trabalho introduz um conjunto de regras para obtenção de casos de testes manuais a partir de um modelo de processo na notação BPMN. Esses casos deverão ser analisados por uma pessoa com expertise na área de teste para julgar se estão corretos ou não.

Segundo Sousa et al. (2014), com uma suíte de testes projetada em conjunto com os requisitos, se espera que essa oriente a etapa de desenvolvimento do *software* fornecendo informações prévias sobre o fluxo dos casos de testes que deverão ser realizados. Dessa forma, produzindo um efeito colateral que pode ser benéfico à qualidade dos artefatos a serem produzidos.

Para a aplicação das regras de extração dos testes, os modelos de processos devem estar detalhados em nível dois ou superior. Assim, é possível destacar elementos operacionais de cada atividade do processo.

As regras apresentadas por Sousa et al. (2014) se iniciam com a obtenção de requisitos canditados inspiradas em (AZEVEDO et al., 2010). O método para identificação dos requisitos candidatos se baseia em três passos:

1. Seleção de atividades
2. Identificação de requisitos candidatos
3. Consolidação de requisitos candidatos

Seleção de atividades é a seleção das atividades automatizadas (executadas por sistema), apoiadas por sistema (executadas por pessoas, mas utilizando um sistema) e automatizáveis (atividades manuais que se pretende automatizar). Atividades manuais e que não estão sendo consideradas para automatização são descartadas, já que não há requisitos envolvidos.

Identificação dos requisitos candidatos é a etapa onde os modelos de processos de

negócio são analisados de acordo com um conjunto de heurísticas definidas por Azevedo et al. (2010). As heurísticas analisam a estrutura e semântica dos modelos de processos.

Para análise da estrutura baseiam-se nos padrões de workflow e cobrem todas as possibilidades de fluxos de atividades que podem ser representadas por um modelo de processo.

Para análise semântica, são considerados os elementos dos modelos, tais como os requisitos de negócio, informações de entrada e saída e regras de negócio. A semântica destes elementos indica as funcionalidades que podem ser implementadas em serviços de apoio ao processo. O resultado desta etapa é a identificação de requisitos candidatos de dados (executam operações *CRUD (Create, Read, Update, Delete)* sobre os dados) e/ou serviços candidatos de lógica (executam regras de negócio e eventualmente podem incluir operações CRUD).

Consolidação de requisitos candidatos é a terceira etapa, quando os requisitos candidatos identificados na etapa anterior são analisados e são geradas informações referentes à granularidade, grau de reuso, dependência entre serviços, associações dos serviços com as atividades de onde foram identificados, papéis que os executam e sistemas que poderão invocá-los.

Após a identificação dos requisitos candidatos, é possível determinar a extração dos casos de teste com os seguintes passos:

Primeiro passo consiste na modelagem dos processos de negócio.

Segundo passo consiste na aplicação do método de identificação de requisitos, apresentado ateriormente.

Terceiro passo consiste em delimitar os casos de teste. Neste passo, foram utilizados como base alguns métodos de geração de casos de testes aplicados em outros modelos, por exemplo, um diagrama de estados ou caso de uso. Um caso de uso, por exemplo, pode possuir diversos “caminhos” a serem testados, já que define além do fluxo principal, os “n” possíveis fluxos alternativos. Isso se assemelha a um processo e suas regras de negócio, que através dos operadores lógicos e eventos, criam diversos caminhos possíveis para a execução do processo.

Por fim, no quarto passo os engenheiros de *software* devem criticar a lista final de casos de teste de forma a identificar quais se aplicam às suas necessidades de teste, que podem, por exemplo, ser reduzida pelo escopo de atuação dos testes que desejam no dado momento.

Assim, é no terceiro passo que as heurísticas para obtenção dos casos de teste são

utilizadas. As regras utilizadas são propostas da seguinte maneira:

1. Heurísticas de requisitos básicos (HRB)
2. Heurística de padrão de workflow (HPW)
3. Heurísticas de cobertura de cenários (HCC)

Heurísticas de requisitos básicos (HRB):

HRB1 (regra de negócio): Um requisito candidato deve ser identificado a partir de uma regra de negócio.

HRB2 (informação de entrada e saída): Um requisito candidato deve ser identificado para cada informação de entrada de uma atividade (caracterizado como requisito de leitura), assim como um requisito candidato de dado deve ser identificado para cada informação de saída de uma atividade (caracterizado como requisito de escrita) desde que as informações estejam associados a portadores de informação (por exemplo, banco de dados, e-mail e documentos eletrônicos).

HRB3 (atividades automatizadas): Um requisito candidato deve ser identificado para cada atividade automatizada.

Heurísticas de padrão de workflow (HPW):

HPW1 (atividades sequenciais): Um requisito candidato deve ser identificado a partir de um conjunto de atividades sequenciais automatizadas ou apoiadas por sistema (uma atividade sequencial é definida por uma sequência de no mínimo duas atividades executadas seguidamente, sem a interferência de operadores lógicos (OR, XOR ou AND)).

HPW2 (ciclo de atividades - Laço): Um requisito candidato deve ser identificado a partir de uma estrutura do workflow onde uma ou mais atividades podem ser executadas repetidamente. Ainda existem as heurísticas de padrão de workflow OR, AND, XOR, Interface de processo e atividades de múltiplas instâncias que podem ser acessadas.

Heurísticas de cobertura de cenários (HCC):

HCC1 (eventos inicial e final): um fluxo de cenário é identificado quando houver uma sequência de atividades presentes entre um evento inicial e um evento final. Se houver mais de um fluxo a ser percorrido, todos deverão ser considerados.

HCC2 (eventos intermediários): um fluxo de cenário é identificado quando houver um fluxo de atividades presentes entre dois eventos intermediários, evento inicial e intermediário ou evento intermediário e evento final.

Para medição de sucesso e efetividade dos casos de testes, (SOUSA et al., 2014)

propões Critérios de cobertura (CC). São esses:

CriCob1 (cobertura de requisitos): todos os requisitos identificados pelas heurísticas devem ser considerados nos casos de testes pelo menos uma vez, ou seja, os requisitos devem estar presentes em algum fluxo extraído do processo de negócio.

CriCob2 (cobertura de fluxos): todos os fluxos identificados pelas heurísticas devem ser considerados nos casos de testes pelo menos uma vez.

6.4 Considerações sobre os trabalhos relacionados

Em (CAI, 2011) é realizada uma proposta para obtenção de casos de testes reutilizáveis a partir de qualquer notação de BPM. As heurísticas utilizadas para isso, entretanto, não estão claras no artigo. Além disso, eventos temporais que são cada vez mais utilizados por sistemas de BPM para execução e escalonamentos de serviços regulares não estão contemplados na composição de testes do autor.

Em (YOTYAWILAI; SUWANNASART, 2014) é proposta uma ferramenta para extração de casos de testes a partir de um modelo em notação BPMN, porém muitas tarefas devem ser realizadas por usuários. É necessário, contudo, estar consciente do quão despendioso pode ser para uma empresa adotar tal ferramenta, à medida que analistas de negócio são os principais recursos de uma organização para a aplicação de testes funcionais e são eles quem sabem os dados necessários para entrada e saída de uma atividade.

O trabalho de (LOUMOS et al., 2010) não sugere uma maneira de extrair casos de teste, porém é muito importante para demonstrar que os modelos de processos de negócio são cada vez mais utilizados como insumo para desenvolvimento de código. Segundo Yotyawilai and Suwannasart (2014), os diagramas e modelos adquirem cada vez mais importância no ciclo de vida de desenvolvimento de aplicações, pois esses são desenvolvidos ainda em estágios iniciais de um negócio. Consequentemente, os casos de teste podem ser gerados juntos com a aplicação, ao invés de criados após o desenvolvimento do código fonte de um sistema.

A proposta de Loumos et al. (2010) é reforçada pelo trabalho de Lübke and Lessen (2016), que utilizaram BPM em BPMN para o desenvolvimento de um sistema.

Após análise da literatura, não foi possível identificar um mapeamento padrão com heurísticas definidas de um modelo de processo em BPMN para casos de teste. Muitos trabalhos como ((YOTYAWILAI; SUWANNASART, 2014), (CAI, 2011), (MOURA et al., 2017)) focam na tentativa de propor uma automatização da extração dos casos de

testes a partir desses modelos, porém sem conseguir resultados concretos.

Em (SOUSA et al., 2014), são propostas heurísticas para obtenção de casos de testes manuais já na fase inicial de desenvolvimento de *software*. As heurísticas, porém, não são de fácil compreensão aos usuários, além de não deixar claro a maneira de testar os dados de entrada e saída de cada uma das atividades, e não possuir um experimento de validação para as regras propostas.

7 CONCLUSÃO

Neste trabalho foram apresentados conceitos em modelagem de processos, BPM e teste de software. Foi possível perceber que a discplina de gerenciamento de processos vem adquirindo cada vez mais importância nas organizações, e que um modelo de processo pode ser um importante material de insumo para o desenvolvimento de sistema de TI.

Abordou-se, portanto, um método para auxiliar a obtenção de casos de teste a partir de um modelo de processos na notação BPMN. O modelo de processo foi mapeado a um modelo de caso de teste baseado em um documento na especificação proposta por (STANDARD, 2008). O modelo de caso de teste apresentado poderá ser melhorado em trabalhos futuros, a fim de deixá-lo mais completo.

Pelos resultados obtidos, foi possível concluir que as regras, de fato, podem auxiliar, tanto pessoas da TI, quanto pessoas de áreas funcionais, atendendo os objetivos do trabalho.

O tempo a ser despendido em um detalhamento de terceiro nível nos modelos, porém, é algo a ser pensado em conjunto com as equipes de análise de negócio com equipes de TI, uma vez que um grande nível de detalhamento do modelo de processo tende a levar mais tempo na modelagem do processo.

Outros elementos da BPMN como piscinas e raias poderão ser objetos de estudos futuros. Esses elementos possibilitam atender critérios de segurança das organizações mirando as corretas e necessárias permissões às execuções dos casos de teste. Esses elementos poderiam, portanto, pertencer ao preenchimento de documentos de planos de teste de nível futuramente.

REFERÊNCIAS

AAGESEN, G.; KROGSTIE, J. Bpmn 2.0 for modeling business processes. p. 219–250, 04 2015.

AALST, W. M. V. D. Business process management: a comprehensive survey. **ISRN Software Engineering**, Hindawi Publishing Corporation, v. 2013, 2013.

AALST, W. van der; HOFSTEDE, A. H. M. T.; WESKE, M. Business process management: A survey. In: **Proceedings of the 1st International Conference on Business Process Management, volume 2678 of LNCS**. [S.l.]: Springer-Verlag, 2003. p. 1–12.

AGUILAR-SAVéN, R. S. Business process modelling: Review and framework. **International Journal of Production Economics**, v. 90, n. 2, p. 129 – 149, 2004. ISSN 0925-5273. Production Planning and Control. Available from Internet: <http://www.sciencedirect.com/science/article/pii/S0925527303001026>.

AZEVEDO, L. et al. **Processos para Governança SOA no ponto de vista da Arquitetura de Tecnologia de Informação**. [S.l.], 2010.

BECKER, J.; ROSEMANN, M.; UTHMANN, C. von. Guidelines of business process modeling. In: ____. **Business Process Management: Models, Techniques, and Empirical Studies**. Berlin, Heidelberg: Springer Berlin Heidelberg, 2000. p. 30–49. ISBN 978-3-540-45594-3. Available from Internet: <https://doi.org/10.1007/3-540-45594-9_3>.

CAI, L. A business process testing sequence generation approach based on test cases composition. In: **2011 First ACIS/JNU International Conference on Computers, Networks, Systems and Industrial Engineering**. [S.l.: s.n.], 2011. p. 178–185.

CHAUHAN, R. K.; SINGH, I. Latest research and development on software testing techniques and tools. **International Journal of Current Engineering and Technology**, v. 4, n. 4, 2014.

DUMAS, M. et al. **Fundamentals of Business Process Management**. [S.l.]: Springer, 2013.

GIAGLIS, G. M. A taxonomy of business process modeling and information systems modeling techniques. **International Journal of Flexible Manufacturing Systems**, v. 13, n. 2, p. 209–228, Apr 2001. ISSN 1572-9370. Available from Internet: <https://doi.org/10.1023/A:1011139719773>.

HOMMES, B.-J.; REIJSWOUD, V. V. Assessing the quality of business process modelling techniques. In: IEEE. **System Sciences, 2000. Proceedings of the 33rd Annual Hawaii International Conference on**. [S.l.], 2000. p. 10–pp.

IPROCESS. **Um guia para iniciar estudos em BPMN (II): Gateways**. 2012. Available from Internet: <http://blog.iprocess.com.br/2012/11/um-guia-para-iniciar-estudos-em-bpmn-ii-gateways/>. Accessed in: 26 sep. 2018.

KANER, C.; FALK, J.; NGUYEN, H. Q. **Testing Computer Software Second Edition**. [S.l.]: Dreamtech Press, 2000.

KHADER, S.; YOUSEF, R. Utilizing business process models to generate software test cases. **International Journal of Computer Science Issues (IJCSI)**, International Journal of Computer Science Issues (IJCSI), v. 13, n. 2, p. 1, 2016.

KROGSTIE, J. **Model-based development and evolution of information systems: A Quality Approach**. [S.l.]: Springer Science & Business Media, 2012.

LINDSAY, A.; DOWNS, D.; LUNN, K. Business processes—attempts to find a definition. **Information and Software Technology**, v. 45, n. 15, p. 1015 – 1019, 2003. ISSN 0950-5849. Special Issue on Modelling Organisational Processes. Available from Internet: <http://www.sciencedirect.com/science/article/pii/S0950584903001290>.

LIST, B.; KORHERR, B. An evaluation of conceptual business process modelling languages. In: **Proceedings of the 2006 ACM Symposium on Applied Computing**. New York, NY, USA: ACM, 2006. (SAC '06), p. 1532–1539. ISBN 1-59593-108-2. Available from Internet: <http://doi.acm.org/10.1145/1141277.1141633>.

LOUMOS, V. et al. Change management and quality of service through business process modeling: The n-vis, a public sector project. **2010 Seventh International Conference on Information Technology: New Generations**, p. 1300–1303, 2010.

LüBKE, D.; LESSEN, T. van. Modeling test cases in bpmn for behavior-driven development. **IEEE Software**, v. 33, n. 5, p. 15–21, Sept 2016. ISSN 0740-7459.

MACINTOSH, A. L. The need for enriched knowledge representation for enterprise modelling. In: **IEE Colloquium on AI (Artificial Intelligence) in Enterprise Modelling**. [S.l.: s.n.], 1993. p. 3/1–3/3.

MENDLING, J.; REIJERS, H.; RECKER, J. Activity labeling in process modeling: Empirical insights and recommendations. **Information Systems**, v. 35, n. 4, p. 467 – 482, 2010. ISSN 0306-4379. Vocabularies, Ontologies and Rules for Enterprise and Business Process Modeling and Management. Available from Internet: <http://www.sciencedirect.com/science/article/pii/S0306437909000313>.

MENDLING, J.; ROSA, M. L.; HOFSTEDE, A. H. ter. Correctness of business process models with roles and objects. 2008.

MINITAB. **Métodos e fórmulas para Teste t pareado**. 2018. Available from Internet: <https://support.minitab.com/pt-br/minitab/18/help-and-how-to/statistics/basic-statistics/how-to/paired-t/methods-and-formulas/methods-and-formulas/>. Accessed in: 26 sep. 2018.

MOORE DAVID S., G. P. M.; CRAIG, B. A. **Introduction to the Practice of Statistics**. [S.l.: s.n.], 2009.

MOURA, J. L. de et al. Test case generation from bpmn models for automated testing of web-based bpm applications. In: **2017 17th International Conference on Computational Science and Its Applications (ICCSA)**. [S.l.: s.n.], 2017. p. 1–7.

NONCHOT, C.; SUWANNASART, T. A tool for generating test case from bpmn diagram with a bpel diagram. In: **Proceedings of the International MultiConference of Engineers and Computer Scientists**. [S.l.: s.n.], 2016. v. 1.

PAGNACOS, T. **The Ultimate Guide to Business Process Management: Everything You Need to Know and How to Apply It to Your Organization**. [S.l.]: CreateSpace Independent Publishing Platform, 2012.

PETERSON, J. L. Petri net theory and the modeling of systems. Prentice Hall PTR, 1981.

RAFI, D. M. et al. Benefits and limitations of automated software testing: Systematic literature review and practitioner survey. In: **2012 7th International Workshop on Automation of Software Test (AST)**. [S.l.: s.n.], 2012. p. 36–42.

RATHI, P.; MEHRA, V. Analysis of automation and manual testing using software testing tool. **International Journal of Innovations & Advancement in Computer Science, ISSN**, p. 2347–8616, 2015.

RECKER, J. Opportunities and constraints: the current struggle with bpmn. **Business Process Management Journal**, Emerald Group Publishing Limited, v. 16, n. 1, p. 181–201, 2010.

RECKER, J. C. et al. How good is bpmn really? insights from theory and practice. 2006.

ROLÓN, E. et al. Evaluation of bpmn models quality - a family of experiments. In: . [S.l.: s.n.], 2008. p. 56–63.

SCHAFERMEYER, M.; GRGECIC, D.; ROSENKRANZ, C. Factors influencing business process standardization: A multiple case study. In: **2010 43rd Hawaii International Conference on System Sciences**. [S.l.: s.n.], 2010. p. 1–10. ISSN 1530-1605.

SHARMA, S.; VISHAWJYOTI, M. Study and analysis of automation testing techniques. **Journal of Global Research in Computer Science**, v. 3, n. 12, p. 36–43, 2013.

SNEHA, K.; MALLE, G. M. Research on software testing techniques and software automation testing tools. In: **2017 International Conference on Energy, Communication, Data Analytics and Soft Computing (ICECDS)**. [S.l.: s.n.], 2017. p. 77–81.

SOUSA, H. et al. Extraction of test cases from business processes models. p. 547–560, 01 2014.

SPECIFICATION, B. P. M. N. **Business Process Model and Notation (BPMN), Version 2.0.2**. 2014. Omg final adopted specification. Available from Internet: <https://www.omg.org/spec/BPMN/2.0.2/PDF>. Accessed in: 18 sep. 2018.

STANDARD, T. Ieee standard for software and system test documentation. **IEEE Std 829-2008**, p. 1–150, July 2008.

WESKE, M. **Business Process Management: Concepts, Languages, Architectures**. 14482 Potsdam,Germany: Springer, 2007.

WOHLIN, C. et al. **Experimentation In Software Engineering**. [S.l.: s.n.], 2012. ISBN 978-0792386827.

YOTYAWILAI, P.; SUWANNASART, T. Design of a tool for generating test cases from bpmn. In: **2014 International Conference on Data and Software Engineering (ICODSE)**. [S.l.: s.n.], 2014. p. 1–6.

YUAN, Q. et al. A model driven approach toward business process test case generation. In: **2008 10th International Symposium on Web Site Evolution**. [S.l.: s.n.], 2008. p. 41–44. ISSN 1550-4441.

APÊNDICE A — QUESTIONÁRIO APLICADO

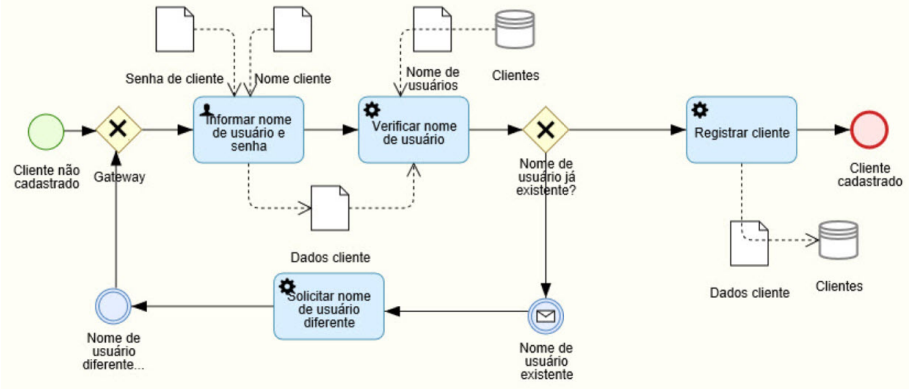
Roteiro para Extração de Casos de Teste a partir de um Modelo de Processo de Negócio

Para o experimento a seguir, você deverá extrair casos de teste a partir de um modelo de processo de negócio gerado na notação BPMN 2.0 e sua descrição textual. Inicialmente você deverá extrair casos de teste a partir de seu próprio conhecimento e depois utilizando as regras propostas pelo autor: Será disponibilizado um tutorial com alguns elementos da BPMN e conceitos de documentos de casos de teste, para auxiliar inicialmente.

A descrição textual do processo é: o processo inicia quando o cliente não está cadastrado. O usuário informa nome de usuário e senha (ambos com no mínimo 8 caracteres), o sistema verifica se o nome de usuário já se encontra registrado na base de dados. Caso o nome de usuário já esteja presente na base de dados, o usuário deverá informar novamente o nome de usuário e senha, porém alterando sua proposta de nome de usuário. Caso o nome do usuário não esteja presente na base de dados, o sistema registra o cliente. Ao final do processo, o cliente estará cadastrado.

O modelo de processo de negócio utilizado neste experimento será o seguinte:

Figura A.1: Modelo de processo cadastrar usuários



Fonte: (SOUSA et al., 2014)

1. Você estuda/estou em algum curso relacionado à area de TI? (campo múltipla escolha. Opções: ”Sim“ e “Não”)

2. Qual seu conhecimento na notação BPMN? (campo múltipla escolha. Opções: ”nennhum“, “básico”, “intermediário” e “avançado”)

3. Qual seu conhecimento em teste de software? (campo múltipla escolha. Opções: ”nennhum“, “básico”, “intermediário” e “avançado”)

4. Com base no tutorial que você leu sobre BPMN e casos de teste, para o modelo de processo de negócio acima, quais casos de teste você utilizaria?

5. Para quais as atividades você gerou casos de teste?

6. Quantos casos de teste você obteve?

Agora você deverá ler o documento que contém as regras do autor.

1. A partir das regras de mapeamento do autor, para o modelo de processo de negócio acima, quais casos de teste você utilizaria?

2. Para quais as atividades você gerou casos de teste?

3. Quantos casos de teste você obteve?

4. Com base no tutorial e regras que você leu, você concorda que essas lhe auxiliaram na extração de casos de teste a partir do modelo em BPMN? (campo múltipla escolha. Opções: “Discordo plenamente”, “discordo”, “discordo parcialmente”, “não concordo nem discordo”, “concordo parcialmente”, “concordo” e “concordo plenamente”)