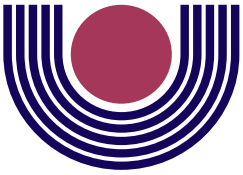


**Unioeste - Universidade Estadual do Oeste do Paraná**
CENTRO DE CIÊNCIAS EXATAS E TECNOLÓGICAS
Colegiado de Ciência da Computação
***Curso de Bacharelado em Ciência da Computação***

# Comparação do processo de derivação de Casos de Uso a partir de modelos BPMN e i* utilizando a ferramenta de apoio JGOOSE

*Rafael Gomes da Silva*

**Cascavel**
**2019**

**RAFAEL GOMES DA SILVA**

# Comparação do processo de derivação de Casos de Uso a partir de modelos BPMN e i* utilizando a ferramenta de apoio JGOOSE

Monografia apresentada como requisito parcial para obtenção do grau de Bacharel em Ciência da Computação, do Centro de Ciências Exatas e Tecnológicas da Universidade Estadual do Oeste do Paraná - Campus de Cascavel

Orientador: Prof. Dr. Victor Francisco Araya Santander

Cascavel
2019

**Rafael Gomes da Silva**

# Comparação do processo de derivação de Casos de Uso a partir de modelos BPMN e i* utilizando a ferramenta de apoio JGOOSE

Monografia apresentada como requisito parcial para obtenção do Título de Bacharel em Ciência da Computação, pela Universidade Estadual do Oeste do Paraná, Campus de Cascavel, aprovada pela Comissão formada pelos professores:

Prof. Dr. Victor Francisco Araya Santander
(Orientador)
Colegiado de Ciência da Computação,
UNIOESTE

Prof. Dr. Ivonei Freitas da Silva
Colegiado de Ciência da Computação,
UNIOESTE

Prof. Ms. Sidgley Camargo de Andrade
UTFPR

Cascavel, 16 de dezembro de 2019

*"Aquele que luta com monstros deve acautelar-se para não tornar-se também um monstro. Quando se olha muito tempo para o abismo, o abismo olha para você."*
*Friedrich Nietzsche.*

# AGRADECIMENTOS

Primeiramente, gostaria de agradecer aos meus pais e meu irmão pelo apoio incondicional em todos os momentos durante esses anos, vocês foram fundamentais para que isso se tornasse realidade. Também gostaria de agradecer ao meu orientador, professor Victor, que sempre esteve disposto a ajudar e ensinar, muito obrigado pela paciência e pelos ensinamentos concedidos. Aos amigos que criei aqui, obrigado pelos momentos, e aos que mesmo longe estiveram sempre comigo.

# Lista de Figuras

# Lista de Tabelas

# Lista de Abreviaturas e Siglas

BP2UC - *Business Process to Use Cases*
BPM - *Business Process Management*
BPMI - *Business Process Modeling Iniciative*
BPMN - *Business Process Model and Notation*
E4J - *Editor for JGOOSE*
GQM - *Goal Question Metric*
IEEE - *Institute of Electrical and Electronic Engineers*
JGOOSE - *Java Goal Into Object Oriented Standard Extension*
LES - Laboratório de Engenharia de Software
OMG - *Object Management Group*
SD - *Strategic Dependenncy*
SR - *Strategic Rationale*
UC - *Use Case*
UML - *Unified Modeling Language*
UNIOESTE - Universidade Estadual do Oeste do Paraná

# Sumário

# Resumo

Este trabalho apresenta um projeto de pesquisa cujo objetivo principal é realizar uma comparação entre modelos organizacionais i* e processos de negócio BPMN na derivação de casos de uso com suporte computacional. As propostas e ferramentas previamente desenvolvidas são apresentadas bem como o passo principal da pesquisa o qual consiste em realizar um quase-experimento. O impacto da pesquisa está em refutar ou confirmar hipóteses sobre a obtenção de casos de uso em contextos nos quais o ambiente organizacional e processos de negócios são previamente modelados e utilizados para descrever requisitos funcionais.

**Palavras-chave:** Casos de uso, processo de negócio, modelo organizacional, quase-experimento.

# Capítulo 1

# Introdução

Este capítulo tem a finalidade da apresentação do trabalho de uma forma geral. Na seção 1.1 é mostrado o contexto do problema. Na próxima seção 1.2 é apresentada a motivação para a realização do trabalho. Na seção 1.3 a proposta do trabalho é denotada de uma forma geral. Em 1.4 as contribuições esperadas que podem ser trazidas com esse trabalho são apresentadas.

## 1.1 Contexto

Projetar e construir um software é uma tarefa desafiadora que exige compreensão de várias subáreas da Engenharia de Software, dentre elas, destaca-se a Engenharia de Requisitos. A Engenharia de Requisitos abrange principalmente a fase inicial de desenvolvimento do sistema, fornecendo o mecanismo apropriado para entender aquilo que o cliente deseja, analisando as necessidades, avaliando a viabilidade, negociando uma solução razoável, especificando a solução sem ambiguidades, validando a especificação e gerenciando as necessidades à medida que são transformadas em um sistema operacional [1]. Pode ser considerada umas das áreas mais críticas para o alcance do sucesso e qualidade do projeto de software que será implementado, e sem ela, a possibilidade de os usuários finais ou stakeholders não terem suas necessidades atendidas é muito grande. Erros cometidos nessa etapa são mais difíceis de serem detectados e mais caros de serem corrigidos [2].

Engenharia de Requisitos pode ser vista como um processo que é dividido em duas fases, a fase de requisitos iniciais (*early requirements phase*) e a fase de requisitos detalhados (*late requirements phase*). A primeira fase que diz respeito aos requisitos iniciais faz análise e modelagem do ambiente a ser transformado em software, o contexto organizacional, os *stakeholders*,

objetivos e relacionamentos. A fase de requisitos detalhados concentra-se na modelagem do sistema em conjunto com o ambiente, determinando e ajustando as fronteiras do sistema, sendo possível identificar requisitos do software [3]. Os requisitos de software correspondem à descrições do que o sistema deve fazer, serviços que deve oferecer e restrições a seu funcionamento [4].

Na fase de requisitos iniciais (*early requirements*), esses requisitos podem ser especificados através de muitas técnicas, existem duas técnicas que representam bem esse tipo de conceito. Um deles é o framework i* (lê-se istar), o qual propõe que a representação de um ambiente organizacional incluindo um sistema computacional pretendido seja feita através da modelagem de atores e suas intencionalidades, relacionamentos entre membros da organização, possibilitando um melhor entendimento das relações que movem a determinada organização. De outro modo, outra técnica bastante utilizada é o conceito baseado em processos de negócio. Dentre as várias linguagens de modelagem de processos de negócios existentes destaca-se o Business Process Model and Notation (BPMN), uma notação gráfica para expressar processos de negócio em formato diagramático, muito utilizado nas áreas empresarial e acadêmica [7]. O BPMN foi publicado em 2004 pela Business Process Modeling Iniciative (BPMI) e, em 2006, foi adotado como padrão OMG (Object Management Group) [8], a especificação do BPMN se encontra na versão 2.0, publicada em 2011 e é a notação mais utilizada para modelar processos de negócios [9].

O objetivo do BPMN é fornecer suporte ao gerenciamento de processos de negócio através de uma notação de fácil compreensão, tanto por analistas que criam esboços iniciais do projeto, por desenvolvedores responsáveis pela implementação e também pela equipe de implantação e suporte do software [9]. Em uma organização pode existir a necessidade de automatizar seus processos para melhorar seu funcionamento e aumentar sua vantagem competitiva entre as demais empresas. Nesse caso, pode-se utilizar tanto modelos de processos de negócio usando BPMN quanto abordagens de modelagem organizacional como o framework i* como fonte para extração de requisitos. Portanto, entender de determinado tipo de técnica para extração de requisitos, seja ela i* ou BPMN é a chave para identificar os requisitos que o software deve atender.

Como mencionado, a Engenharia de Requisitos pode fazer o uso de duas técnicas (i* e

BPMN) para melhor compreensão do contexto organizacional e os processos de negócio, onde as informações são importantes e devem ser usadas para elicitar os requisitos do sistema de software que será implementado. Entretanto, elicitar esses requisitos utilizando as técnicas mencionadas não é uma tarefa fácil, pois é preciso ter conhecimento do domínio e do ambiente organizacional, dos relacionamentos entre atores/participantes da organização e das tarefas/atividades realizadas por eles. Além disso, as tarefas de modelagem de processos de negócios e contexto organizacional, bem como a tarefa de desenvolvimento de software são realizadas por diferentes grupos de pessoas, culminando na utilização de diferentes linguagens [7]. Como essas etapas (*early e late requirements*) acabam exigindo diferentes habilidades por parte dos desenvolvedores, essa culminação de várias linguagens diferentes acaba ocorrendo.

Na etapa de *late requirements* tem destaque o uso da UML (Unified Modeling Language) [11] [12] a qual vem com o propósito de fornecer uma linguagem padrão para a modelagem Orientada a Objetos (OO). A UML é definida como uma linguagem gráfica para visualização, especificação, construção e documentação de artefatos de sistemas de software. Nesta linguagem, a técnica mais utilizada para modelar requisitos funcionais é casos de uso. Casos de uso são utilizados amplamente no desenvolvimento orientado a objetos e são considerados ferramentas essenciais em várias metodologias de desenvolvimento de projetos de software. Contudo, a obtenção de casos de uso UML é um desafio e a comunidade tem buscado diferentes abordagens para realizar esta tarefa. Uma das abordagens para esse fim é considerar os modelos BPMN e i* como base para gerar os casos de uso.

## 1.2 Motivação

Conforme relatado na seção anterior, poder utilizar modelos BPMN e i* para derivar casos de uso pode ser vantajoso. Nesse contexto, cabe destacar alguns trabalhos que propõem iniciativas voltadas à derivação de casos de uso UML a partir de modelos organizacionais. Um desses trabalhos, apresentado por [13], propõe um processo que consiste na construção de modelos organizacionais através do framework i* e derivação desses modelos para casos de uso UML. Para a realização de um processo automático dessa derivação, foi desenvolvida uma ferramenta computacional denominada JGOOSE (*Java Goal Into Object Oriented Standard Extension*) [14], a qual está integrada ao E4J (*Editor for JGOOSE*), um editor de modelos organizacionais i* e

diagramas de casos de uso UML [15][16]. Após alguns anos, outros trabalhos foram realizados com objetivo de aprimorar a ferramenta JGOOSE, integrando a ela o BP2UC (*Business Process to Use Cases*), como apresentado em [17], que apresenta um E4J para criar modelos de processos de negócio (BPMN). A BP2UC implementa a proposta de derivação de casos de uso a partir de modelos BPMN apresentada em [17]. Esta proposta consiste de passos e diretrizes para mapear elementos no BPMN para casos de uso.

Desta forma, considerando os trabalhos em [13] e [17] é possível derivar Casos de Uso a partir de modelos de processos de negócio e modelos organizacionais de forma automatizada, adotando a ferramenta JGOOSE como suporte computacional. Contudo, há necessidade de investigar quais as vantagens e desvantagens dos dois modelos i* e BPMN na geração de casos de uso quando comparados em um determinado ambiente utilizando os editores da ferramenta.

Para esse fim, será realizado um quase experimento [18] considerando princípios da engenharia de software experimental. Optou-se por utilizar a abordagem Goal/Question/Metric (GQM) [16][47] para apoiar a fase inicial do quase experimento a qual nos ajuda tanto na definição quanto no planejamento do mesmo. O GQM é uma abordagem orientada a metas e utilizada na Engenharia de Software para medição de produtos e processos de software. O GQM é baseado em que toda coleta de dados deve ser baseada em um fundamento lógico, objetivo ou meta. A representação da estrutura no âmbito deste trabalho assume a seguinte forma:

• **Objetivo global:** Verificar a corretude e completude da rotina de mapeamento de i* e BPMN para gerar casos de uso usando a ferramenta de apoio JGOOSE. Dessa forma, será possível comparar detalhes diferentes que cada uma das técnicas gera com o apoio da ferramenta, como erros e fraquezas, diferenças entre seus diagramas, chegando, assim, a conclusão das vantagens ou desvantagens que cada uma das técnicas apresenta no âmbito de derivação de casos de uso com o JGOOSE.

• **Objetivo de estudo:**

**Analisar** os casos de uso textuais e diagramáticos gerados a partir do uso da ferramenta JGOOSE.

**Com a finalidade** de verificá-los.

**Com respeito à** corretude e completude dos casos de uso textuais seguindo o template proposto por [19] e diagrama de Casos de Uso gerado considerando a notação UML.

**Do ponto de vista dos** stakeholders.

**No contexto do** uso da ferramenta por discentes da disciplina de Processo de Engenharia de Software I do curso de Ciência da Computação da Unioeste.

Assim, avaliando as alternativas de pesquisa na Engenharia de Software, chegou-se a conclusão de que um quase-experimento seria viável para trabalhar na comparação dos dois modelos i* e BPMN na geração de casos de uso em determinado ambiente. Motiva-nos o fato de que com a realização deste experimento poderemos obter resultados que nos permitam apontar vantagens e desvantagens no uso das abordagens e ferramenta de derivação de casos de uso a partir de modelos i* e BPMN.

## 1.3 Proposta

Assim, neste trabalho realizou-se um quase experimento com os acadêmicos de Ciência da Computação da UNIOESTE, matriculados na matéria de Engenharia de Software I. Um quase-experimento é realizado frequentemente em laboratório, proporcionando maior nível de controle. Podem ser manipuladas uma ou mais variáveis mantendo outras fixas, avaliando o efeito resultante. Os experimentos são adequados para conhecimento convencional, observar relacionamentos, analisar predições dos modelos ou legitimar medidas. O presente trabalho realizou um diagnóstico mais profundo e sistemático sobre as técnicas i* e BPMN no âmbito da derivação de casos de uso usando a ferramenta JGOOSE. Objetivou-se com a realização do quase-experimento, coletar dados que nos permitam apresentar dados de uma comparação de cada uma das abordagens para derivação de casos de uso em determinado ambiente apresentado. Para esse fim, algumas hipóteses foram estabelecidas, as quais poderão ser confirmadas ou refutadas considerando os resultados do experimento.

Para a base desse quase experimento foram utilizadas técnicas e fases propostas por [18], sendo que as fases do quase-experimento contemplam a definição do escopo, planejamento, operação e por fim a analise dos resultados obtidos. Cada fase é detalhada no capítulo 4. A fase do planejamento deve ser uma parte do protocolo de experimentação que deve ser formulada com atenção, pois nessa fase, são definidas informações importantes como a formulação das hipóteses e seleção das variáveis do contexto. A formulação de hipóteses [18] é importante em um quase-experimento pois estas irão estabelecer suposições que podem ser confirmadas ou

refutadas, estas hipóteses são formuladas para nortear o experimentador na condução do quase-experimento. As hipóteses já foram formuladas assim como a seleção das variáveis e podem ser descritas da seguinte forma:

Hipótese 1: a utilização dos modelos organizacionais i* juntamente da ferramenta E4J (i*) para derivação de Casos de Uso influenciou o resultado final, sendo que a quantidade de casos de uso corretos bem como estereótipos («include», «extend» e «generalization») e atores encontrados por quem utilizou é maior do que a quantidade encontrada por quem utilizou BPMN juntamente da ferramenta E4J (BPMN). Ou, ao contrário, a utilização de modelos de processo de negócio BPMN juntamente da ferramenta E4J (BPMN) para derivação de Casos de Uso influenciou o resultado final, sendo que a quantidade de casos de uso corretos bem como estereótipos e atores encontrados por quem utilizou é maior à quantidade de casos de uso corretos encontrados por quem utilizou i* juntamente da ferramenta E4J (i*).

Hipótese 2: a completude dos casos de uso corretos gerados através da derivação i* -> casos de uso é superior a derivação BPMN -> casos de uso ou vice e versa. Este aspecto é mensurado pela completude das descrições textuais dos casos de uso.

Hipótese nula: a utilização de alguma das duas técnicas assim como da ferramenta de apoio não influenciou o resultado final, sendo que não há diferença entre as abordagens em relação à quantidade de casos de uso, atores e estereótipos corretos gerados no diagrama de Casos de Uso bem como em relação à completude das descrições textuais geradas.

Já a seleção de variáveis [18] indica um conjunto de elementos os quais serão usados para realizar o quase-experimento (variáveis independentes) ou serão obtidos com o quase-experimento (variáveis dependentes). Neste quase quase-experimento as variáveis dependentes são os casos de uso, atores e estereótipos gerados bem como as descrições textuais dos casos de uso mensurados através do GQM, obtidos pelos dois grupos que realizarão o experimento (Seção 4.2.5) e as variáveis independentes são as técnicas e a ferramenta que será utilizada no processo.

## 1.4 Contribuições obtidas

As principais contribuições desse trabalho foram:

• Apresentar dados de uma comparação sobre o uso das técnicas i* e BPMN no processo de

derivação de Casos de Uso.

• Permitir que interessados no tema possam ter acesso a informação do uso de i* e BPMN sendo comparados na derivação de Casos de Uso.

• Apresentar resultados que possam confirmar se o uso de i* e BPMN são viáveis no âmbito acadêmico.

• Servir de base para realizar outros experimentos no âmbito industrial.

• Servir de base para a partir do experimento ter idéias claras de melhorias na ferramenta JGOOSE.

## 1.5 Estrutura do trabalho

Na sequência, o trabalho encontra-se organizado da seguinte maneira:

• **Capítulo 2:** são apresentados os conceitos teóricos essenciais para o entendimento do contexto deste trabalho. Será apresentado o conceito de modelos organizacionais i* e o processo de negócio BPMN e Casos de Uso.

• **Capítulo 3:** é apresentado conceitos acerca da Engenharia de Sofware Experimental com foco em Experimento Controlado.

• **Capítulo 4:** é apresentado o protocolo de experimentação, bem como as hipóteses formuladas para a realização do mesmo, e as fases do quase-experimento.

# Capítulo 2

# Referencial teórico

Neste capítulo serão apresentados aspectos teóricos para o entendimento do contexto deste trabalho. Primeiramente, na seção 2.1 é introduzido o conceito de modelo organizacional, enfatizando a técnica i*. Na seção 2.2 é apresentado o conceito de processos de negócio e como são gerenciados por uma organização, incluindo a modelagem de processos de negócio BPMN. Na seção 2.3 é mostrado os conceitos necessários para o entendimento de Casos de Uso, os elementos que compõe o Diagrama de Casos de Uso UML e um *template* utilizado para obtenção de descrição textual de Casos de Uso. Na seção 2.4 as propostas de derivação de Casos de Uso são apresentadas contendo suas etapas. A seção 2.5 é apresenta a ferramenta JGOOSE e o editor E4J.

## 2.1 Modelo organizacional

Várias técnicas tradicionais que são aplicadas no desenvolvimento de *software* tratam de aspectos relacionados à funcionalidade do sistema, descrições de atividades e entidades, porém sem considerar aspectos amplos como objetivos da organização, as regras do negócio, as restrições e também aspectos não funcionais relacionados a qualidade, confiabilidade e a usabilidade. De acordo com [22], essas técnicas não são adequadas para buscar soluções alternativas para problemas da organização, e seu uso pode levar a realizar à implementação de sistemas computacionais que não satisfazem os problemas reais da organização. Os requisitos organizacionais não podem ser considerados como uma simples descrição de funcionalidade do sistema, pois trata do domínio onde o sistema está inserido e das restrições que podem existir no ambiente e no desenvolvimento. Nesse contexto, a Modelagem Organizacional facilita a compreensão do

ambiente empresarial e é considerada uma atividade valiosa pela Engenharia de Requisitos. O modelo organizacional é uma representação da estrutura, atividades, processos, informações e recursos, das restrições de empresas comerciais, governamentais e outros ambientes. Esse modelo ajuda a compreender as interações entre organizações e pessoas. Várias técnicas têm sido propostas para a modelagem organizacional [13]. Dentre estas técnicas destaca-se o framework i* [23] o qual tem uma forte comunidade de apoio [51] e vem sendo utilizado em âmbito acadêmico e industrial.

### 2.1.1 Framework i*

O framework de modelagem i* (i-estrela) foi originalmente proposto por Yu [23], trata da modelagem de contextos organizacionais tomando como base os relacionamentos entre dependências entre atores participantes.

O principal objetivo no i* é representar, através de modelos, os atores que participam e as dependências entre os mesmos, para que suas próprias metas sejam atingidas, recursos fornecidos, tarefas sejam realizadas e metas satisfeitas ou razoavelmente satisfeitas [23]. Mais além, o uso do referido framework possibilita o entendimento das razões internas dos atores, uma vez que as mesmas são expressas explicitamente, melhorando o entendimento por parte do engenheiro de requisitos durante a modelagem.

O framework i* apresenta dois modelos diferentes: o modelo SD (dependências estratégicas - *Strategic Dependency*) e o modelo SR (modelo de razões estratégicas - *Strategic Rationale*). Esses modelos são detalhados a seguir.

#### 2.1.1.1 O Modelo SD - *Strategic Dependency*

O modelo SD mostra os relacionamentos de dependência estratégica entre atores da organização, utilizando uma rede de nós, representando atores e arestas, que representam dependências entre os mesmos.

Neste modelo, uma dependência simboliza um relacionamento baseado em cooperação entre dois atores, onde o primeiro é denominado *depender* e o segundo *dependee*, unidos por um elo de dependência denominado *dependum*, representando uma meta a ser atingida, uma tarefa a ser realizada, um recurso a ser fornecido, ou uma meta flexível a ser satisfeita. Yu [23] define quatro tipos de *dependum* da seguinte forma:

Uma **meta** é uma condição ou estados de desejos que o ator deseja alcançar. Através das metas, podemos representar a intencionalidade dos atores organizacionais. A especificação de como o modo da meta deve ser alcançada não é especificado, dessa forma, possibilita uma série de alternativas.

Uma **tarefa** especifica uma maneira particular de realizar algo. Quando uma tarefa é especificada como subtarefa de outra, ela restringe a tarefa superior para uma ação em particular.

Um **recurso** é uma entidade (física ou informacional) que não é considerada problemática pelo ator. Sua principal característica é se está disponível (e por quem, caso seja uma dependência externa).

Uma **meta flexível (softgoal)** é uma condição ou estado no mundo que o ator deseja alcançar, porém, diferente de uma meta, o critério para a condição de ser alcançada não é definido a princípio, estando aberto a interpretação. Para metas flexíveis, utiliza-se o termo "razoavelmente satisfeita", pois servem para designar metas que não possuem critérios claramente definidos para sua satisfação.

Metas, tarefas, recursos e metas flexíveis fazem parte dos elos de dependência entre atores, com base na figura 2.1, pode-se perceber quatro tipos de dependências possíveis. Tais dependências serão descritas a seguir:

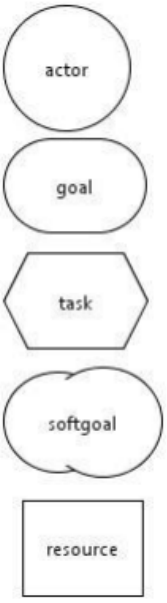



Figura 2.1: Tipos de dependência i*

• **Dependência por meta:** o ator (*depender* depende de outro ator (*dependee*) para alcançar um determinado estado. Cabe ao *dependee* toda e qualquer decisão para que a meta seja alcançada. O *depender* possui a habilidade de decidir que condição ou estado será alcançado. Porém, o *dependee* pode falhar na realização de tal condição, tornando assim o *depender* vulnerável.

• **Dependência por tarefa:** o ator (*depender*) depende de outro ator (*dependee*) para que uma tarefa seja desempenhada. Uma dependência por tarefa determina "como"e não "por quê"uma tarefa deve ser desempenhada. Neste caso o *depender* também fica vulnerável, pois o *dependee* pode falhar ao desempenhar uma tarefa.

• **Dependência por recurso:** o ator (*depender*) depende de outro ator (*dependee*) para a disponibilização de uma entidade (física ou computacional). Neste caso, o *depender* possui a habilidade de usar a entidade como um recurso. Porém, o *depender* torna-se vulnerável, pois a entidade pode ficar indisponível.

• **Dependência por meta flexível:** o ator (*depender*) depende de outro ator (*dependee*) para que este desempenhe alguma tarefa para "satisfazer a contento"uma meta flexível. Neste caso, o *depender* tem a decisão final em aceitar ou não a meta alcançada, usando o benefício de habilidades e conhecimentos do *dependee*. Diferentemente do que ocorre em relação às metas, as condições para que as metas flexíveis sejam alcançadas são elaboradas de acordo com o desempenho das tarefas.

Utilizaremos um exemplo simples para auxiliar no entendimento desse modelo e de seus elementos (Figura 2.2).

Explicando de forma mais detalhada a Figura 2.2, que trata de um sistema genérico de compra e venda, vemos que o ator "*Cliente*" depende do ator "*Funcionário*" para atingir o objetivo de "*Fazer Compra*" e o "*Funcionário*" por sua vez depende do ator "*Sistema*" para atingir o objetivo de "*Efetuar Venda*", ainda o "*Funcionário*" depende de uma solicitação do recurso "*Dados para Pagamento*" do ator "*Cliente*". O ator "*Funcionário*" tem uma dependência do "*Sistema*" para realizar o objetivo de "*Consultar Produtos*", também depende do "*Sistema*" para realizar o objetivo de "*Emitir Nota Fiscal*", que poderia estar pendente. Ainda o ator "*Funcionário*" depende do "*Sistema*" para atingir o objetivo-soft "*Rapidez*", o qual claramente representa um requisito não funcional que deve ser atendido pelo sistema.

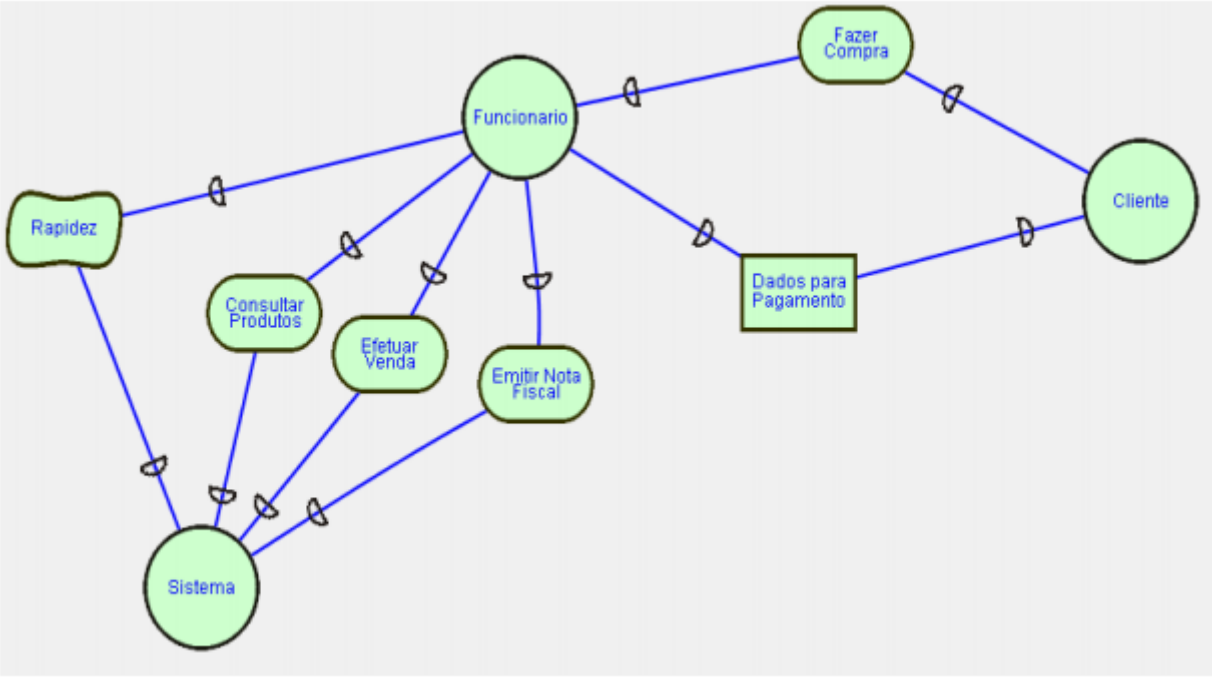



Figura 2.2: Exemplo Modelo SD

De acordo com [23], os tipos dependências refletem o grau de liberdade que existe no relacionamento. Na dependência por meta, cabe ao *dependee* toda e qualquer decisão para que a meta seja satisfeita. Na dependência por tarefa, o *dependee* executa a tarefa de acordo com o desejo do *depender*, cabendo assim ao *depender* tomar as decisões. No caso da dependência por recurso, o grau de liberdade é nulo, ou seja, o *dependee* fornece o recurso exatamente como o *depender* necessita. Por fim, em uma dependência por meta flexível, o *depender* toma a decisão final, sendo beneficiado pelo *know-how* (conhecimento prévio adequado) do *dependee*.

#### 2.1.1.2 O Modelo SR - *Strategic Rationale*

O modelo SR tem como objetivo representar o detalhamento das estratégias internas (*rationale*) dos atores participantes, em termos dos elementos do processo, das alternativas e das decisões por trás do processo.

Os elementos do processo usados para representar os relacionamentos intencionais internos aos atores são: os relacionamentos "*meio-fim*", que possuem o papel de explicitar as decisões que foram tomadas para que as metas do ator fossem alcançadas, a *decomposição de tarefas*, que detalha a elaboração e realização das tarefas, além de mostrar como os recursos são disponibilizados e utilizados, e os *relacionamentos de contribuição*, que exibem o tipo de contribuição

(positiva ou negativa) entre metas flexíveis. A seguir, os elementos serão detalhados.

**Decomposição por tarefa** - uma tarefa (definida anteriormente como uma maneira particular de realizar algo) é mapeada graficamente pela representação de nós subcomponentes ligados a tarefa superior através de um segmento de reta cortado. Os nós subcomponentes podem ser: metas, tarefas, recursos e metas flexíveis.

**Relacionamento meios-fim** - os relacionamentos do tipo meio fim são mapeados graficamente através de uma seta direcionada para o nó fim, que pode ser uma meta a ser alcançada, uma tarefa a ser desempenhada, um recurso a ser produzido ou uma meta flexível a ser "satisfeita a contento".

Para melhor visualizar e entender o modelo SR mostraremos a expansão do ator "*Sistema*" apresentado na Figura 2.2, bem como suas ligações e razões estratégicas, que serão vistas na Figura 2.3, logo a seguir.

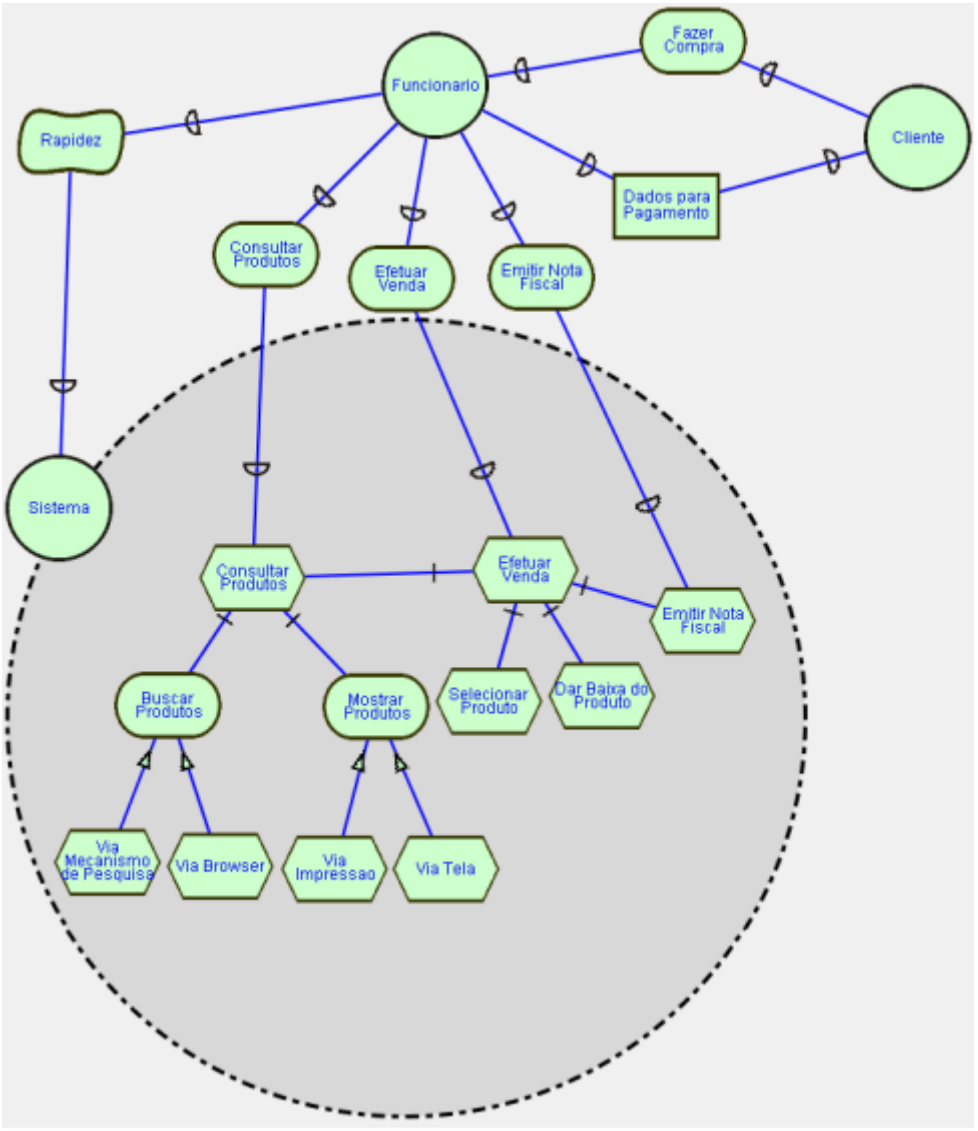



Figura 2.3: Exemplo Modelo SR

Analisando o modelo SR, vemos que as razões estratégicas por trás de cada dependência se revelam de forma mais detalhada, isso pode ser exemplificado no objetivo de "*Consultar Produtos*" que quando detalhado é decomposto por duas ligações de decomposição, demonstrando que para atingir o objetivo principal é necessário que se realize o sub-objetivo "*Buscar Produtos*" e posteriormente "Mostrar Produtos" resultantes da busca. A busca ainda pode ser feita de duas formas "*Via Mecanismo de Pesquisa*" e "*Via Browser*" demonstrados pelas ligações meio-fim, onde, "*Via Browser*", é um meio de se obter o fim de "*Buscar Produtos*", assim como, "*Via Mecanismo de Pesquisa*". O sub-objetivo "*Mostrar Produtos*" possui dois meios para ser alcançada, "*Via Impressão*" e "*Via Tela*". O objetivo "*Efetuar Venda*" é detalhado por meio

de ligações de decomposição, que como já mencionado acima, elas decompõem esse objetivo em "*Consultar Produtos*", "*Selecionar Produto*", "*Dar Baixa do Produto*" e "*Emitir Nota Fiscal*", onde essa ultima decomposição pode ser acessada de forma independente, demonstrada na tarefa "*Emitir Nota Fiscal*" solicitado pelo ator "*Funcionário*".

## 2.2 Modelagem de Processo de Negócio

De acordo com Davenport [24], um processo é uma ordenação específica das atividades de trabalho dentro do tempo e do espaço, possuindo um começo e um fim, além de entradas e saídas claramente identificadas. O processo define a estrutura para realizar tarefas que visam atingir os objetivos da organização. Um processo de negócio é um conjunto de atividades inter-relacionadas que são executadas por uma organização a fim de atingir um objetivo de negócio. Cada processo de negócio é executado por uma única organização, mas pode interagir com processos de negócio de outras organizações [25].

As organizações devem possuir um amplo entendimento dos processos internos e como estes são relacionados com o ambiente externo, com objetivo de aumentar a eficiência e qualidade dos produtos e serviços fornecidos, por isso muitas organizações adotam a abordagem de gerenciamento de Processos de Negócio (*Business Process Management* - BPM) [26].

A *Business Process Management* é uma abordagem que se refere ao gerenciamento completo do ciclo de vida dos processos de negócio, incluindo a análise, projeto, implementação, execução e melhoria contínua dos processos de negócio de uma organização, a fim de alcançar os resultados desejados de forma consistente e alinhada com as metas estratégicas da organização [27][28]. A BPM adota modelos de processos de negócio, os quais representam uma abstração do funcionamento do processo, provendo uma visão simplificada de sua estrutura, facilitando a comunicação, melhorias e inovações, além de definir requisitos necessários para sistemas de informação encarregados de apoiar a execução desse processo [29].

A modelagem dos processos de negócio é formada por técnicas que tem por objetivo explicar atividades feitas pela organização e como essas atividades se relacionam com recursos do negócio, com o intuito de alcançar o objetivo do processo. Eriksson e Penker [29] definem várias razões para a adoção da modelagem de processos de negócio pelas organizações, dentre elas:

• Aumentar a compreensão do processo e facilitar a comunicação;

• Atuar como base para sistemas de informação que suportem o processo;

• Identificar melhorias no processo atual.

A modelagem de processos de negócio torna-se ferramenta essencial para compreender o processo, já que permite uma visão mais clara e serve como um meio de documentação das atividades executadas no processo [30]. São várias técnicas, dentre elas se destaca a *Business Process Model and Notation* (BPMN), definida como o padrão mundial para esse fim e amplamente utilizada nas áreas industrial e acadêmica [31].

### 2.2.1 Business Process Model and Notation (BPMN)

A *Business Process Model and Notation* (BPMN) é uma notação gráfica para representação de processos de negócio em formato diagramático, cujo principal objetivo é fornecer uma notação universal compreensível por todos os usuários de negócio, incluindo profissionais encarregados de implementar ferramentas para apoiar a execução dos processos de negócio [32].

Originalmente denominada *Business Process Modeling Notation*, a primeira versão da BPMN foi disponibilizada em 2004 pela *Business Process Management Iniciative* (BPMI). Em 2005 a BPMI e a *Object Management Group* (OMG) fundiram-se, desde então, a BPMN é mantida pela OMG, organização internacional que especifíca padrões para aplicações orientadas a objetos e, devido a isso, a BPMN passou a ser considerada o padrão para modelagem de processos de negócio [33].

Em janeiro de 2011 foi disponibilizada a versão atual da notação, BPMN 2.0, a qual introduziu vários elementos adicionais e novos tipos de diagramas, com consequência na redução da necessidade de uso de anotações e culminando em uma maior rapidez para criação e interpretação dos modelos [34]. Neste trabalho será adotada a versão 2.0 da BPMN. Três tipos de diagramas são suportados na atual versão [32].

• **Diagrama de Processos de Negócio (Orchestration):** representa a relação entre as atividades e eventos do processo, com elementos que apoiam o entendimento do fluxo do processo. A subdivisão é feita da seguinte forma:

**- Processo de negócio privado executável:** processo interno a uma organização específica e modelado com o propósito de ser executado de acordo com a semântica da BPMN [32].

**- Processo de negócio privado não executável:** processo interno a uma organização e modelado com a finalidade de documentar o comportamento do processo em um nível de detalhe definido pelo modelador [32].

**- Processo de negócio público:** processo que representa a interação entre um processo de negócio privado e algum outro processo ou participante [32].

• **Diagrama de Coreografia:** retrata a sequência de interações entre os participantes de um processo, retratando a dinâmica da comunicação entre eles [32].

• **Diagrama de Colaboração:** representa a interação entre duas ou mais entidades de negócio [32].

No presente trabalho será utilizado o Diagrama de Processos de Negócio. Através dele pode-se modelar o fluxo do trabalho da organização, com definição de quais atividades serão executadas, bem como o responsável pela execução e como as interações entre os participantes através de mensagens são feitas.

A BPMN fornece um pequeno conjunto de categorias para que o usuário possa facilmente identificar os tipos básicos dos elementos e entender o diagrama [32]. Na figura 2.4 são apresentados os principais elementos gráficos da BPMN 2.0. Abaixo são apresentadas as categorias e os elementos básicos.

**1. Objetos de fluxo:** definem o comportamento do processo de negócio. Tem a subdivisão em três categorias da seguinte forma:

(a) **Eventos:** indicam algo que ocorreu durante a execução de um processo, afetando o fluxo do mesmo. Usualmente possuem uma causa (gatilho) ou um impacto (resultado). São divididos em três tipos:

i. **Eventos de início:** indicam onde um processo inicia. São representados por um círculo com uma borda simples.

ii. **Eventos intermediários:** são representados por um círculo duplo e indicam a ocorrência de um evento entre o início e o final da execução do processo.

iii. **Eventos de fim:** indicam quando um processo termina. São representados por um círculo de borda simples, mas, mais espessa que a borda dos eventos de início.

(b) **Atividade:** representa um trabalho realizado pela organização. Subdivide-se em:

i. **Tarefa:** é uma atividade atômica, ou seja, um trabalho que já se encontra no maior nível de detalhamento. É representada por um retângulo arredondado.

ii. **Subprocesso:** é uma atividade não-atômica, que pode ser decomposta através de outros elementos a fim de definir seu funcionamento interno. É representado por um retângulo arredondado com um símbolo + na parte central inferior.

(c) **Gateway:** controle a convergência e divergência de fluxos de sequência em um processo. É representado por um losango.

**2. Dados:** representam as informações utilizadas ou produzidas por uma atividade. São representados por quatro elementos: Objetos de dados, dados de entrada, dados de saída e bases de dados.

**3. Conectores:** conecta um objeto de fluxo com outro objeto de fluxo ou outra informação. Há três tipos de conectores:

(a) **Fluxo de sequência:** usado para exibir a ordem em que as atividades são executadas no processo.

(b) **Fluxo de mensagem:** usado para exibir a troca de mensagem entre dois participantes do processo.

(c) **Associação:** usado para conectar informações e artefatos com outros elementos gráficos BPMN.

**4. Swinlanes:** são utilizadas para agrupar, organizar e categorizar as atividades executadas no fluxo de execução do processo. Subdivide-se em dois elementos:

(a) **Pool (piscina):** representa um processo de negócio ou um participante do processo de negócio, responsável pela realização de atividades.

(b) **Lane (raia):** subdivisão dentro de uma Pool, utilizada para organizar e categorizar atividades, caracterizando os participantes envolvidos na realização das atividades do processo.

**5. Artefatos:** são usados para fornecer informações adicionais sobre o processo. São subdivididos em dois elementos:

(a) **Grupos:** são utilizados para agrupar elementos gráficos que pertencem a uma mesma categoria.

(b) **Anotações de texto:** constituem um mecanismo para o modelador fornecer informações adicionais para o leitor do diagrama.

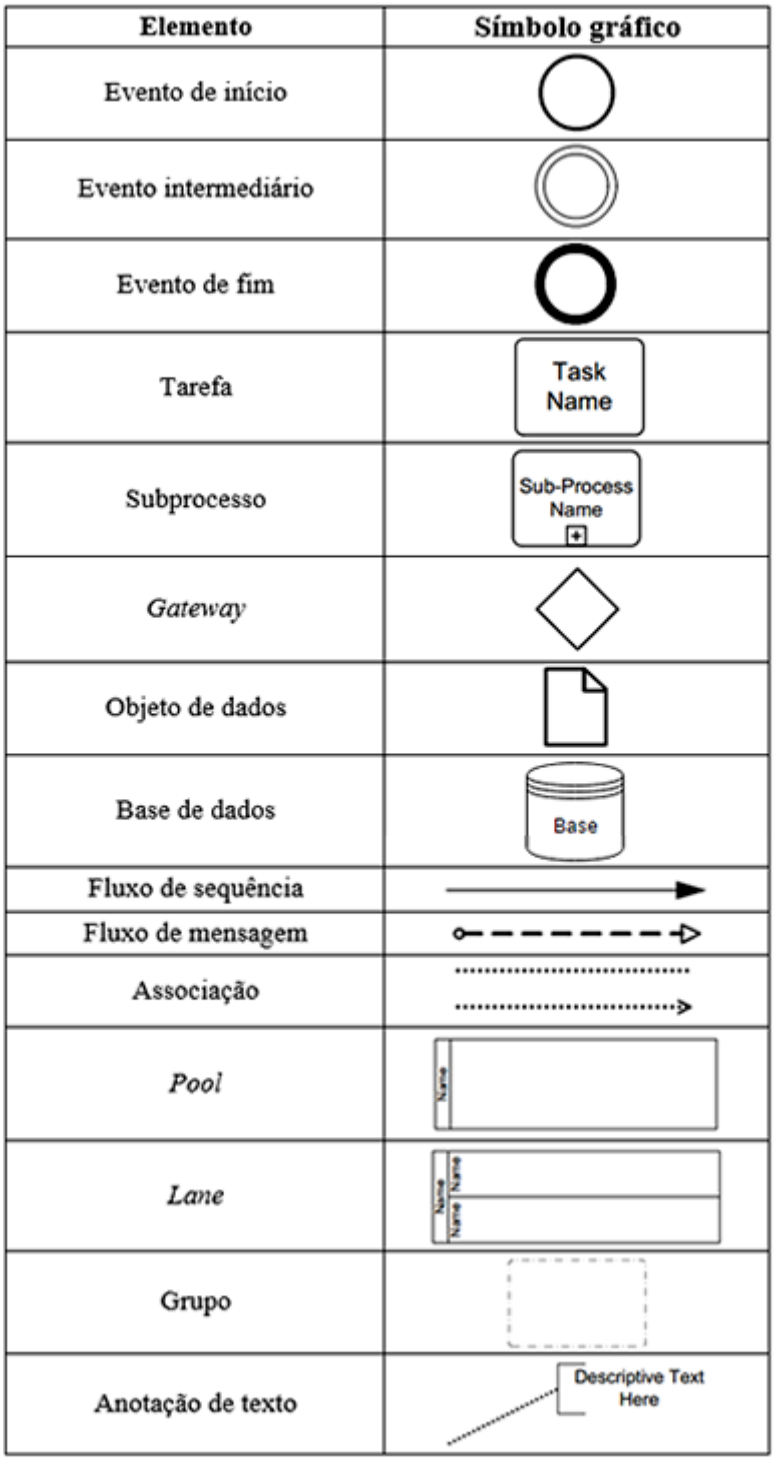



Figura 2.4: Elementos básicos BPMN

## 2.3 Casos de Uso

Casos de Uso foram introduzidos por Jacobson [35], com objetivo de mostrar o comportamento do sistema pela visão do usuário. Um Caso de Uso especifica o comportamento de um

sistema, ou de parte dele, sendo uma descrição de uma sequência de ações, incluindo as possíveis variações nestas ações, que produzem um resultado observável para um ator [36].

Para Cockburn [19], um Caso de Uso captura um contato dos *stakeholders* do sistema e seu comportamento, mapeia o escopo do sistema, torna a comunicação com o usuário mais fluída e contribui para o gerenciamento do projeto. Um Caso de Uso pode ser definido como um conjunto de cenários, onde cada cenário contém uma sequência de passos que representa uma interação entre o usuário e o sistema.

A especificação de Casos de Uso é uma das estratégias de documentação que apoiam e auxiliam o desenvolvedor a especificar as funcionalidades do sistema de forma mais coesa, tendo a representatividade da intenção do usuário, dessa forma, facilitando a compreensão dos requisitos funcionais, o que é um fator primordial para a qualidade do software resultante. Devido a isso, a modelagem de Casos de Uso tornou-se uma técnica amplamente utilizada para especificação de requisitos, tanto na área acadêmica como na industrial [37].

Casos de Uso podem ser representados de duas formas, através do Diagrama de Casos de Uso e também com descrições textuais. Essas formas possuem objetivos diferentes, a primeira mostra a relação entre os Casos de Uso e os atores, e a segunda apresenta a representação do comportamento dos Casos de Uso usando um texto estruturado escrito em linguagem natural [37].

### 2.3.1 Diagrama de Casos de Uso UML

O conceito de Casos de Uso é adotado pela Unified Modeling Language (UML) [11] desde o final da década de 1990, o que contribuiu para a popularização da utilização de Casos de Uso [37]. Segundo [36], a UML é uma linguagem gráfica para visualização, especificação, construção e documentação de artefatos de sistemas de software orientados a objeto, a qual define padrões para projeto de sistemas. Um desses padrões especificado pela UML é o Diagrama de Casos de Uso.

Um Diagrama de Casos de Uso reúne vários Casos de Uso, procurando descrever o funcionamento do sistema [36]. Com o diagrama é possível descrever os relacionamentos e dependências entre Casos de Uso e os atores que estão envolvidos no processo. Para a elaboração do diagrama e relacionamentos é utilizado um conjunto de elementos, que podem ser visualizados

na figura 2.5.

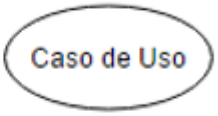
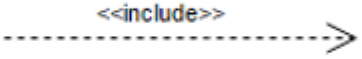
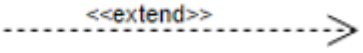
| Nome | Definição | Notação gráfica |
|---|---|---|
| Ator | Representa um conjunto coerente de papéis que os usuários de Casos de Uso desempenham quando interagem com esses Casos de Uso. |  Ator |
| Caso de Uso | Especifica o comportamento de um sistema ou de parte dele. |  Caso de Uso |
| Associação | Representa a comunicação entre Atores e Casos de Uso. |  |
| Inclusão | Representa que um Caso de Uso é utilizado durante a execução de outro Caso de Uso. Indica que um Caso de Uso base incorpora explicitamente o comportamento de outro Caso de Uso em um determinado local especificado no Caso de Uso base. Possui o estereótipo <<include>> associado. |  <<include>> |
| Extensão | Indica que um Caso de Uso base incorpora implicitamente o comportamento de outro Caso de Uso em um determinado local especificado indiretamente no Caso de Uso estendido. É utilizado para indicar um comportamento opcional. Possui o estereótipo <<extend>> associado. |  <<extend>> |
| Generalização | Pode ocorrer entre Atores ou Casos de Uso. Indica que o elemento filho herda o comportamento e o significado do elemento pai, acrescentando ou sobrescrevendo o comportamento do pai. |  |

Figura 2.5: Elementos de um Diagrama de Casos de Uso

### 2.3.2 Descrição textual

Um Caso de Uso pode ser representado na forma textual, utilizando linguagem natural, descrevendo como o sistema se comporta e representando ações que devem ser realizadas entre o

usuário e o sistema para concluir um determinado objetivo [36]. Porém, o uso de linguagem natural pode acarretar em confusão, dificuldades para compreensão do mesmo e formas variadas de interpretação. Para tentar minimizar o problema foram criadas várias *guidelines* e estruturas para a definição de padrões para a especificação textual [37]. Dentre várias estruturas propostas para obter a especificação textual dos Casos de Uso, as quais são chamadas de *templates*, destaca-se o conjunto proposto por Cockburn [19], que tem por objetivo capturar um conjunto completo de informações, incluindo o nome do Caso de Uso, atores, escopo, contexto, pré-condições, cenário de sucesso, entre outras informações. Neste experimento será adotado uma adaptação do *template* que tem o formato *fully dressed form*, definindo explicitamente os objetivos do Caso de Uso, enfatizando a especificação de informações detalhadas, como a sequência de passos executada no cenário principal, objetivando facilitar a compreensão e a comunicação entre os *stakeholders* [37][19]. A figura 2.6 apresenta esse *template*. Uma versão adaptada é utilizada na JGOOSE.

**Caso de Uso:** <nome - objetivo descrito com uma frase curta contendo um verbo na voz ativa>

---

***INFORMAÇÃO CARACTERÍSTICA***

***Objetivo no contexto:*** <uma sentença mais longa do objetivo do caso de uso, se for necessário>

***Escopo:*** <qual sistema está sendo considerado (por exemplo, organização ou sistema computacional) >

***Pré-condições:*** <o que é necessário que já esteja satisfeito para realizar o caso de uso>

***Condição final de sucesso:*** <o que ocorre/muda após a obtenção do objetivo do caso de uso>

***Condição final de falha:*** <o que ocorre/muda se o objetivo é abandonado>

***Ator primário:*** <o nome do papel para o ator primário, ou descrição>

---

***CENÁRIO PRINCIPAL DE SUCESSO***

<coloque aqui os passos do cenário necessários para obtenção do objetivo>

<passo #> <descrição da ação>

---

***EXTENSÕES***

<coloque aqui as extensões, uma por vez, cada uma referenciando o passo associado ao cenário principal>

<passo alterado> <condição> : <ação ou subcaso de uso>

---

***INFORMAÇÕES RELACIONADAS***

<informações adicionais necessárias para melhor descrever o Caso de Uso>

Figura 2.6: Template de Cockburn no formato *fully dressed form* para especificação da descrição textual. Adaptado de [19].

## 2.4 Propostas de derivação de Casos de Uso

A ferramenta JGOOSE a ser avaliada no experimento, suporta dois processos de obtenção de casos de uso. O processo de derivação de casos de uso a partir de i* proposto em [13] e define diretrizes para essa finalidade. Já o processo de derivação de casos de uso a partir de modelos BPMN foi proposto por [17] e também apresenta um conjunto de diretrizes para essa finalidade. Estas propostas são apresentadas brevemente nas próximas seções.

### 2.4.1 Derivação de casos de uso a partir de i*

Esta proposta consiste essencialmente em 3 passos os quais permitem derivar respectivamente, os atores de casos de uso, os casos de uso e finalmente, as descrições textuais dos casos de uso gerados. Para esse fim são analisados tanto os elementos dos modelos SD e SR em i* [13].

***1º Passo da Proposta: Descoberta de atores***

Diretriz 1: todo ator em i* deve ser analisado para um possível mapeamento para ator em caso de uso;

Diretriz 2: inicialmente, deve-se analisar se o ator em i* é externo ao sistema computacional pretendido. Caso o ator seja externo ao sistema, o mesmo é considerado candidato a ator em Casos de Uso;

Diretriz 3: deve-se garantir que o ator em i* é candidato a ator em Caso de Uso, através da seguinte análise, conforme segue:

SubDiretriz 3.1: verificar se existe pelo menos uma dependência do ator analisado em relação ao ator em i* que representa o sistema computacional pretendido;

Diretriz 4: atores em i*, relacionados através do mecanismo IS-A nos modelos organizacionais e mapeados individualmente para atores em casos de uso (aplicando diretrizes 1, 2 e 3), serão relacionados no diagrama de casos de uso através do relacionamento do tipo «generalização».

***2º Passo da Proposta: Descoberta de Casos de Uso.***

Diretriz 5: para cada ator descoberto para o sistema pretendido no passo 1, devemos observar todas as suas dependências (***dependum***) do ponto de vista do ator como ***dependee***, em relação ao ator sistema computacional pretendido (sistema computacional -> *dependum* -> ator), visando descobrir casos de uso para o ator analisado;

SubDiretriz 5.1: deve-se avaliar as dependências do tipo objetivo associadas com o ator;

SubDiretriz 5.2: deve-se avaliar as dependências do tipo tarefa, associadas com o ator;

SubDiretriz 5.3: deve-se avaliar as dependências do tipo recurso, associadas com o ator;

SubDiretriz 5.4: deve-se avaliar as dependências do tipo objetivo-soft, associadas com o ator;

Diretriz 6: analisar a situação especial na qual um ator de sistema (descoberto seguindo as diretrizes do passo 1) possui dependências (como ***depender*** em relação ao ator em i* que representa o sistema computacional pretendido ou parte dele. (ator -> *dependum* -> sistema computacional).

Diretriz 7: classificar cada caso de uso de acordo com seu tipo de ojetivo associado (objetivo contextual, objetivo de usuário, objetivo de subfunção).

***3º Passo da Proposta: Descoberta e descrição do fluxo principal e alternativas dos Casos de Uso.***

Diretriz 8: analisar cada ator e seus relacionamentos no Modelo de Razões Estratégicas para extrair informações que possam conduzir à descrição de fluxos principal e alternativos, bem como pré condições e pós-condições dos casos de uso descobertos para o ator.

Diretriz 8.1: analisar os sub-componentes em uma ligação de ***decomposição de tarefa*** em um possível mapeamento para passos na descrição do cenário primário (fluxo principal) de casos de uso.

Diretriz 8.2: analisar ligações do tipo *meio-fim* em um possível mapeamento para passos alternativos na descrição de casos de uso.

Diretriz 8.3: analisar os relacionamentos de dependências de sub-componentes no modelo de Razões Estratégicas em relação a outros atores do sistema. Estas dependências podem originiar pré-condições e pós-condições para os casos de uso descobertos.

Diretriz 9: Investigar a possibilidade de derivar novos objetivos de casos de uso apartir da observação dos passos nos cenários (fluxos de eventos) dos casos de uso descobertos. Cada passo de um caso de uso deve ser analisado para verificar a possibilidade do mesmo ser refinado em um novo caso de uso.

Diretriz 9.1: Inicialmente deve-se averiguar qual é o objetivo que se quer atingir com a ação que o passo representa na descrição do caso de uso;

Diretriz 9.2: Descoberto o objetivo, deve-se averiguar a necessidade de decompor/refinar o objetivo para que o mesmo possa ser alcançado;

Diretriz 9.3: Um novo caso de uso será gerado a partir do objetivo analisado, se pudermos definir os passos que representam a necessidade de interações adicionais entre o ator e o sistema para se atingir o sub-objetivo desejado;

Diretriz 9.4: Adicionalmente, pode-se encontrar novos objetivos e cenários com base na observação dos obstáculos de objetivos já descobertos.

Diretriz 10: Desenvolver o diagrama de casos de uso utilizando os casos de uso descobertos, bem como observando os relacionamentos do tipo «include», «extend» e «generalization» usados para estruturas as especificações dos casos de uso.

### 2.4.2 Derivação de casos de uso a partir de BPMN

A seguir, é apresentada a propostas de derivação de casos de uso para o modelo BPMN. O processo a ser utilizado para a derivação dos casos de uso foi definido por [17], sendo composto por representação diagramática, através do Diagrama de Casos de Uso UML e de descrição textual. Essa abordagem usa como base diretrizes e conceitos estabelecidos nos trabalhos de [7], [47] e [48], que se assemelham ao mapear participantes para atores e atividades para casos de uso.

**1º PASSO: OBTENÇÃO DO DIAGRAMA DE CASOS DE USO**

A primeiro passo da abordagem é responsável pela identificação dos atores e casos de uso e posterior associação entre eles. Como resultado dessa etapa é gerado um Diagrama de Casos de Uso da UML.

Para aplicar o processo de derivação devem ser considerados alguns conceitos e premissas. É utilizado o termo instância para referir-se a um fluxo de execução do processo de derivação, o qual é originado a partir de um evento de início, subprocesso ou elemento com múltiplas opções de fluxos. Para auxiliar a derivação é utilizada uma lista de instâncias, a qual armazena, para cada instância, o número identificador, elemento que a originou e o próximo elemento a ser analisado. Quando uma nova instância é criada, ela é inserida no início dessa estrutura. Assim que a avaliação de uma instância tiver sido finalizada, deve-se remover a mesma da estrutura de instâncias e continuar o processo de derivação na instância localizada no início da estrutura. Caso não houver mais instâncias a serem avaliadas o processo é finalizado.

Para determinar o próximo elemento a ser analisado adota-se o sentido determinado pelos fluxos de sequência e de mensagem. Ao analisar um elemento deve-se marcá-lo como visitado. Cada caso de uso obtido deve ser armazenado em uma estrutura composta pelos atributos número, o qual identifica a ordem de obtenção, nome, diretriz utilizada, instância originadora e

casos de uso incluídos.

As diretrizes utilizadas no processo de obtenção do Diagrama de Casos de Uso são apresentadas a seguir, divididas em três categorias, sendo a primeira categoria responsável pela identificação dos atores, a segunda pela identificação dos casos de uso e a terceira pela associação dos casos de uso aos atores e pela definição de relacionamentos de inclusão entre os casos de uso.

**Categoria 1 - Identificação dos atores**

**DRD1:** Cada pool presente no diagrama BPMN será representada por um ator no Diagrama de Casos de Uso. O nome do ator será o nome da pool.

**DRD2:** Cada lane originará um ator no Diagrama de Casos de Uso. O nome do ator será o nome da lane. O ator obtido será uma especialização do ator que representa a pool na qual a lane está inserida, obtendo-se com isso uma hierarquia de atores.

**Categoria 2 - Identificação dos atores**

**DRD3**: Cada evento de início que seja pertencente ao processo principal e não seja destinatário de algum fluxo de mensagem, originará uma instância de execução do processo de derivação.

**DRD4:** Se o elemento atualmente analisado tiver sido avaliado e não for uma atividade que possui o recebimento de fluxo de mensagem, deve-se finalizar a instância atual.

**DRD5:** Se o elemento atualmente analisado for uma tarefa que não tenha sido avaliada anteriormente e a instância atual de análise não tiver sido originada pela avaliação de um fluxo de mensagem, será originado um caso de uso. O nome do caso de uso será o nome da tarefa

**DRD6:** Se o elemento atualmente analisado for uma tarefa que não tenha sido avaliada anteriormente e a instância tiver sido originada por um fluxo de mensagem, será originado um caso de uso com o mesmo nome da tarefa e no eu campo Casos de Uso Incluídos na descrição textual deve ser inserido o nome do Caso de Uso obtido a partir da atividade que possui um fluxo de mensagem destinado à tarefa atualmente analisada.

**DRD7:** Se o elemento do fluxo de sequência atualmente analisado for um subprocesso que não tenha sido avaliado, será originado um caso de uso com o nome do subprocesso. A instância atual é finalizada e deve-se criar uma nova instância para cada fluxo de mensagem ou de sequência saindo do subprocesso. Posteriormente, deve-se expandir o subprocesso e criar

uma nova instância, que deve possuir o marcador especial SP, indicando que é proveniente de um subprocesso, cujo primeiro elemento da instância será o elemento inicial do subprocesso expandido. A análise deve continuar a partir desse elemento.

**DRD8:** Se o elemento atualmente analisado for uma atividade (A1) que tenha sido avaliada anteriormente e que seja proveniente de instância originada a partir de fluxo de mensagem, deve-se inserir no campo Casos de Uso Incluídos do caso de uso obtido a partir de A1, o nome do caso de uso gerado pela atividade que possui um fluxo de mensagem destinado à A1. Após, devese finalizar a execução da instância atual.

**DRD9:** Se o elemento analisado for um evento de fim, deve-se finalizar a execução da instância atual

**DRD10:** Caso haja múltiplas opções para continuar o fluxo da execução devese finalizar a instância atual e inserir uma instância de análise para cada saída, continuando o processo de derivação no próximo elemento da última instância criada.

**DRD11:** Se o elemento atualmente analisado não se enquadrar nas diretrizes especificadas anteriormente, deve-se analisar o próximo elemento.

**Categoria 3: Associação dos casos de uso**

**DRD12:** Um caso de uso será associado ao ator que representa a swimlane na qual a atividade relacionada com o caso de uso está inserida.

**DRD13:** Caso uma atividade (A1) possua fluxo de mensagem para uma atividade de outro participante (P1), o caso de uso que representa A1 será associado com o ator originado a partir de P1.

**DRD14:** Cada caso de uso obtido a partir de uma instância originada pela expansão de um subprocesso será incluído (via estereótipo «include») ao caso de uso que representa o subprocesso.

**2º PASSO: OBTENÇÃO DA DESCRIÇÃO TEXTUAL**

A segunda etapa da abordagem é responsável pela obtenção de uma descrição textual para cada caso de uso identificado na etapa anterior. As diretrizes presentes nessa etapa são utilizadas no processo de avaliação dos elementos associados com as atividades que originaram os casos de uso, procurando obter informações adicionais que sejam de fundamental importância para o entendimento do caso de uso. As informações obtidas são inseridas em um template obtido

a partir do template utilizado na JGOOSE, o qual é uma adaptação do proposto por Cockburn [19].

Adotou-se o conjunto de diretrizes apresentado em [31] como base, sendo utilizado para definição de um processo algorítmico para obtenção das informações. Cabe destacar que em [31] é adotado um template simplificado para o mapeamento das informações, ocasionando a perda de detalhes importantes como a sequência de atividades realizadas. Para possibilitar a utilização do estereótipo «include» alterou-se a DRT8. Já para obter um maior detalhamento das ações realizadas no caso de uso, foi empregado um template mais completo e elaborou-se o processo de aplicação das diretrizes adotando o conceito de sequência de ações executadas no cenário principal. As diretrizes pertencentes à segunda etapa foram divididas em categorias de acordo com o tipo do elemento e são apresentadas a seguir.

***Categoria 1: Associação com elementos de dados***

Uma associação entre um elemento de dados e uma atividade resulta em uma sentença que será anexada ao cenário da descrição do caso de uso que representa a respectiva atividade. Para gerar a sentença são utilizadas as diretrizes DRT1 a DRT6, as quais são apresentadas na Figura 2.7.

***Categoria 2: Associação com anotação***

**DRT7:** Quando uma associação conecta uma anotação de texto com uma atividade, o conteúdo da anotação de texto deve ser transcrito ao campo Informação adicional do caso de Uso que representa a atividade.

***Categoria 3: Fluxo de mensagem***

**DRT8:** Quando uma atividade recebe uma mensagem, deve ser inserida a sentença Recebeu <mensagem> de <nome do participante emissor> no cenário de seu caso de uso. Caso o emissor da mensagem for uma atividade, deverá ser anexada à sentença obtida o trecho «include» <nome do caso de uso obtido a partir do emissor da mensagem>, indicando que aquele passo do cenário será associado com outro caso de uso.

**DRT9:** Quando uma atividade envia uma mensagem, deve ser inserida a sentença Enviou <nome da mensagem> para <nome do participante receptor> no cenário de seu caso de uso.

| Diretriz | Descrição | Representação gráfica | Sentença originada |
|---|---|---|---|
| DRT1 | Atividade obtém/recebe um *Data Object*. | Nome → Atividade<br>ou<br>Nome → Atividade | <nome do elemento> recebido. |
| DRT2 | Atividade produz um *Data Object*. | Atividade → Nome<br>ou<br>Atividade → Nome | <nome do elemento> produzido/enviado. |
| DRT3 | Atividade recebe uma coleção de objetos de dados. | Nome → Atividade | Recebe uma coleção de <nome do elemento>. |
| DRT4 | Atividade produz/envia uma coleção de objetos de dados. | Atividade → Nome | Envia uma coleção de <nome do elemento>. |
| DRT5 | Atividade acessa um *Data Store*. | Nome → Atividade | Obtém informações da base de dados <nome do elemento> |
| DRT6 | Atividade altera uma *Data Store*. | Atividade → Nome | Altera informações na base de dados <nome do elemento> |

Figura 2.7: Sentenças originadas a partir de elementos de dados. Adaptado de [31]

***Categoria 4: Fluxo de sequência***

DRT10: Se uma atividade estiver conectada a outra atividade por um fluxo de sequência será originada uma précondição no caso de uso originado pela atividade alvo, com a sentença Atividade <nome da atividade de origem> deve ser concluída. Para fluxos de sequência entre duas atividades com um gateway como elemento intermediário deve ser utilizada uma das diretrizes da Figura 2.8. A sentença obtida deve ser inserida no caso de uso referente à atividade alvo.

***Categoria 5: Eventos***

Para cada evento de início deve-se aplicar uma das diretrizes da Figura 2.9. Já para eventos intermediários e eventos de fim deverá ser aplicada a DRT24.

DRT24: Quando uma atividade possui como próximo elemento de um fluxo de sequência um evento intermediário ou final, será originada uma sentença com o conteúdo O evento

| Diretriz | Tipo do evento | Notação | Sentença originada |
|---|---|---|---|
| DRT17 | Mensagem |  | A mensagem <nome da mensagem> chegou de <nome do participante emissor>. |
| DRT18 | Temporizador |  | O tempo/data <nome/definição do evento> foi alcançado. |
| DRT19 | Condicional |  | A condição <expressão> tornou-se verdadeira. |
| DRT20 | Sinal |  | O sinal <nome/definição do evento> chegou. |
| DRT21 | Múltiplo |  | O evento <nome/definição do evento> [ou <nome/definição do evento> ocorreu. |
| DRT22 | Múltiplo paralelo |  | O evento <nome/definição do evento> [e <nome/definição do evento> ocorreu. |
| DRT23 | Outro tipo ou tipo não especificado | - | O evento <nome/definição do evento> ocorreu. |

Figura 2.8: Sentenças originadas a partir de *gateways*. Adaptado de [31]

<nome/definição do evento> ocorre, que deve ser incluída como Condição Final de Sucesso no caso de uso derivado a partir da atividade (ver na figura 2.7).

***Processo para obtenção textual dos casos de uso***

Utilizando as diretrizes para representação textual especificadas anteriormente, foi definido um algoritmo contendo o processo de obtenção de informações detalhadas acerca dos casos de uso, o qual é composto por dez passos descritos a seguir e que deve ser aplicado a cada caso de uso obtido anteriormente. O processo de obtenção das descrições textuais é finalizado quando todas as atividades que originaram casos de uso tiverem sido analisadas.

Processo de aplicação das diretrizes para representação textual:

Passo 1 - Preenchimento do cabeçalho: Completa-se o cabeçalho do template, inserindo o respectivo número e nome do caso de uso.

Passo 2 – Associação com atores: O campo Ator primário será preenchido com o nome da lane ou pool na qual a atividade que originou o caso de uso está localizada. Caso a atividade possua fluxo de mensagem com atividades de outros participantes, os nomes dos atores que representam tais participantes deverão ser inseridos no campo Atores secundários.

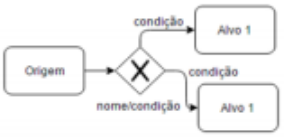
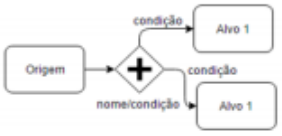
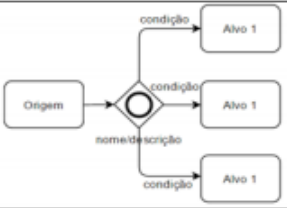
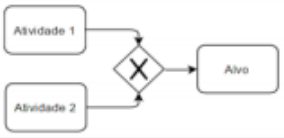
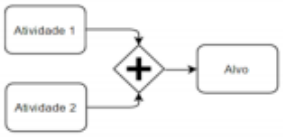
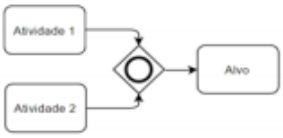
| Diretriz | Descrição | Representação gráfica | Sentença originada |
|---|---|---|---|
| DRT11 | Divergência utilizando *Exclusive Gateway*. Execução segue em apenas um dos caminhos (o que possuir a condição verdadeira). |  | <nome/descrição do gateway> ser <condição do fluxo de sequência>. |
| DRT12 | Divergência utilizando *Parallel Gateway*. Representa a divisão do fluxo em dois ou mais fluxos que são executados de maneira paralela. |  | <atividade origem> ter sido concluída. |
| DRT13 | Divergência utilizando *Inclusive Gateway*. É realizada avaliação da condição de cada fluxo e o fluxo pode prosseguir por uma ou mais saídas. |  | No *gateway* <nome/descrição do *gateway*> o fluxo deve seguir pela ramificação <condição do fluxo de sequência>. |
| DRT14 | Convergência usando *Exclusive Gateway*. Quando um dos fluxos chegar ao gateway a sequência continua no fluxo de saída. |  | Uma das atividades <nome da atividade 1> ou <nome da atividade 2> [ou <nome da atividade n>] tiver sido concluída. |
| DRT15 | Convergência usando *Parallel Gateway*. Garante a conclusão de todos os fluxos de chegada para só então dar continuidade ao fluxo de saída. |  | A <nome da atividade 1> e <nome da atividade 2> [e <nome da atividade n>] tiver sido concluída. |
| DRT16 | Convergência usando *Inclusive Gateway*. Garante que todos os fluxos em execução sejam concluídos antes de dar sequência ao fluxo de saída. |  | A <nome da atividade 1> e <nome da atividade 2> [e <nome da atividade n>] tiver sido concluída. |

Figura 2.9: Sentenças originadas por eventos de início. Adaptado de [31]

Passo 3 – Obtenção de gatilho: Analisa-se eventos de início associados com a atividade que originou o caso de uso, aplicando a diretriz apropriada da Figura 2.9. A informação obtida será inserida no campo Gatilho.

Passo 4 – Análise de fluxo de sequência: Avalia-se a existência de fluxo de sequência entre uma atividade e a atividade atualmente avaliada, aplicando a DRT10 e inserindo a sentença obtida como uma pré-condição.

Passo 5 – Análise de gateways: Avalia-se gateways entre uma atividade origem e a atividade que representa o caso de uso atualmente analisado, verificando qual das diretrizes da Figura 2.8 deverá ser aplicada e inserindo a sentença no campo Pré-condições.

Passo 6 – Avaliação de eventos intermediários e de fim: Deve-se verificar se a atividade está relacionada com eventos intermediários ou de fim, aplicando então a diretriz DRT24 e inserindo

a sentença no campo Condição final de sucesso.

Passo 7 – Obtenção do cenário principal: Preenche-se o Cenário principal do caso de uso. Para determinar a ordem correta de inserção das sentenças no cenário, vários elementos são avaliados e há várias diretrizes envolvidas, devido a isso o passo é dividido em subpassos.

Passo 7.1 – Recebimento de mensagem: Aplica-se a diretriz DRT8.

Passo 7.2 – Recebimento/obtenção de dados: Aplicam-se as diretrizes DRT1, DRT3 e DRT5.

Passo 7.3 - Alteração/envio de dados: Aplicam-se as diretrizes DRT2, DRT4 e DRT6.

Passo 7.4 – Envio de mensagem: Aplica-se a diretriz DRT9.

Passo 8 – Inclusão em outros casos de uso: Deve-se avaliar a estrutura de armazenamento de casos de uso obtida na primeira etapa da abordagem, verificando se o caso de uso que corresponde a atividade atual é incluído por algum outro caso de uso. Caso isso ocorra, os nomes dos casos de uso que incluem o atualmente analisado devem ser inseridos no campo Casos de uso pai.

Passo 9 – Casos de usos incluídos: Analisa-se novamente a estrutura de casos de uso, verificando se o caso de uso que corresponde a atividade atual inclui outros casos de uso, inserindo os seus respectivos nomes no campo Casos de uso incluídos.

Passo 10 – Informações adicionais: Para cada associação com anotação de texto aplica-se a diretriz DRT7.

**3º PASSO: REFINAMENTO MANUAL**

Para obter um modelo de casos de uso de forma mais acurada e com maior correspondência com a situação expressa no modelo BPMN, pode ser necessário realizar um refinamento manual, complementando o resultado obtido através da derivação com informações que não puderam ser obtidas com aplicação das diretrizes. Isso pode ocorrer devido ao modelo BPMN estar incompleto, possuir inconsistências e/ou não ter sido construído utilizado boas práticas de modelagem. Além disso, em várias propostas, tais como [48], [53] e [52], defende-se a realização de uma etapa posterior para aprimorar o resultado obtido através da transformação direta. Devido a isso, torna-se relevante complementar a abordagem com recomendações para revisão dos Casos de uso obtidos, as quais constituem a terceira etapa da abordagem.

Nessa etapa não são definidas diretrizes, mas são apresentadas orientações acerca de pos-

síveis situações que podem ser avaliadas e modificadas a fim de melhorar o modelo obtido. Recomenda-se que sejam consultados indivíduos que conheçam o processo de negócio, tais como futuros usuários, especialistas no domínio da aplicação, além do engenheiro de requisitos, o que irá facilitar a adição, remoção ou alteração de informações contidas no modelo. Aconselha-se também que os nomes dos atores e casos de uso sejam avaliados e alteradospara nomes mais significativos, se necessário. Deve-se verificar se os atores obtidos correspondem a participantes reais do processo de negócio. Caso o ator tiver sido obtido através de uma pool que corresponder a um processo de negócio, deverá ser avaliada a possibilidade de substituição pelo profissional que executa as atividades nesse processo ou então tal ator deve ser removido. Atores que representem indivíduos ou sistemas que não interagem com o sistema a ser representado devem ser descartados, bem como os casos de uso que se relacionem unicamente com tais atores. Devem ser mantidos apenas os casos de uso que correspondem às tarefas relacionadas com a execução do sistema. Por fim, recomendase que sejam complementados os campos do template de representação textual com informações adicionais não obtidas diretamente dos modelos BPMN.

## 2.5 JGOOSE

É relevante a apresentação da ferramenta JGOOSE (*Java Goal into Object Oriented Standard Extension*) pois se tem como objetivo utilizá-la para a realização do experimento.

A JGOOSE é uma ferramenta de auxílio para o mapeamento para modelos funcionais. Atualmente a ferramenta permite a elaboração de modelos organizacionais do framework i*, através do editor E4J i*, a criação de Diagramas de Casos de Uso e a obtenção automática de Casos de Uso a partir de modelos i*. Também conta com a elaboração de modelos de processo de negócio BPMN, através do editor E4J BPMN, a criação de Diagramas de Casos de Uso e a obtenção automática de Casos de Uso a partir de modelos BPMN. Os Casos de Uso obtidos são apresentados através de Diagrama de Casos de Uso UML e descrições textuais, utilizando uma versão baseada no *template* proposto por Cockburn [19], apresentada na figura 2.6.

A primeira versão da ferramenta foi desenvolvida em 2006 [42], com a reimplementação, utilizando Java, da ferramenta JGOOSE, a qual foi implementada na linguagem *Rational Rose Scripting* [41]. Ao longo dos anos a ferramenta passou por várias melhorias e aprimoramentos,

realizados por acadêmicos vinculados do Laboratório de Engenharia de Software (LES) da Universidade Estadual do Oeste do Paraná (UNIOESTE), através de projetos de iniciação científica e de Trabalhos de Conclusão de Curso. Dentre as mudanças realizadas estão a refatoração do código-fonte, correção de bugs, implementação de novas funcionalidades e alterações na interface gráfica do usuário [43] [40].

A partir da implementação do editor gráfico do Diagrama de Casos de Uso, denominado E4J Use Cases, possibilitou-se a criação e manipulação de Diagrama de Casos de Uso diretamente no ambiente de trabalho proposto pela JGOOSE [16][39]. Para a implementação do E4J Use Cases adotou-se a JGraphX, uma biblioteca Java para visualização de grafos, a qual consiste de um conjunto de estruturas e funcionalidades que possibilitam a criação de aplicações interativas voltadas para a manipulação de diagramas [37].

Após alguns anos, em um trabalho de conclusão de curso apresentado em [17], a ferramenta foi aprimorada e passou a contar com um editor BPMN (E4J BPMN) e uma ferramenta para derivar automaticamente Casos de Uso a partir de modelos BPMN. Com isso, a JGOOSE passou a possibilitar a obtenção de Casos de Uso a partir de modelos i* e modelos BPMN, contribuindo para a especificação de requisitos de um sistema de informação. É vantajoso para o engenheiro de requisitos construir Casos de Uso a partir de requisitos extraídos de diversas fontes que modelam o mesmo propósito e, posteriormente, que é o objetivo deste trabalho, compará-los em determinado contexto, para verificar vantagens e desvantagens de cada uma das propostas.

Para melhor compreensão das técnicas utilizadas na ferramenta JGOOSE, serão mostradas figuras das telas do software a seguir. A figura 2.10 representa a tela inicial do JGOOSE, onde podemos perceber que cada módulo é dividido pela sua tarefa, temos o editor BPMN (E4J BPMN) que no botão do lado direito podemos mapear esse modelo BPMN para Casos de Uso, também temos o editor i* (E4J i*), que pode ser mapeado para Casos de Uso com o botão ao lado direito.

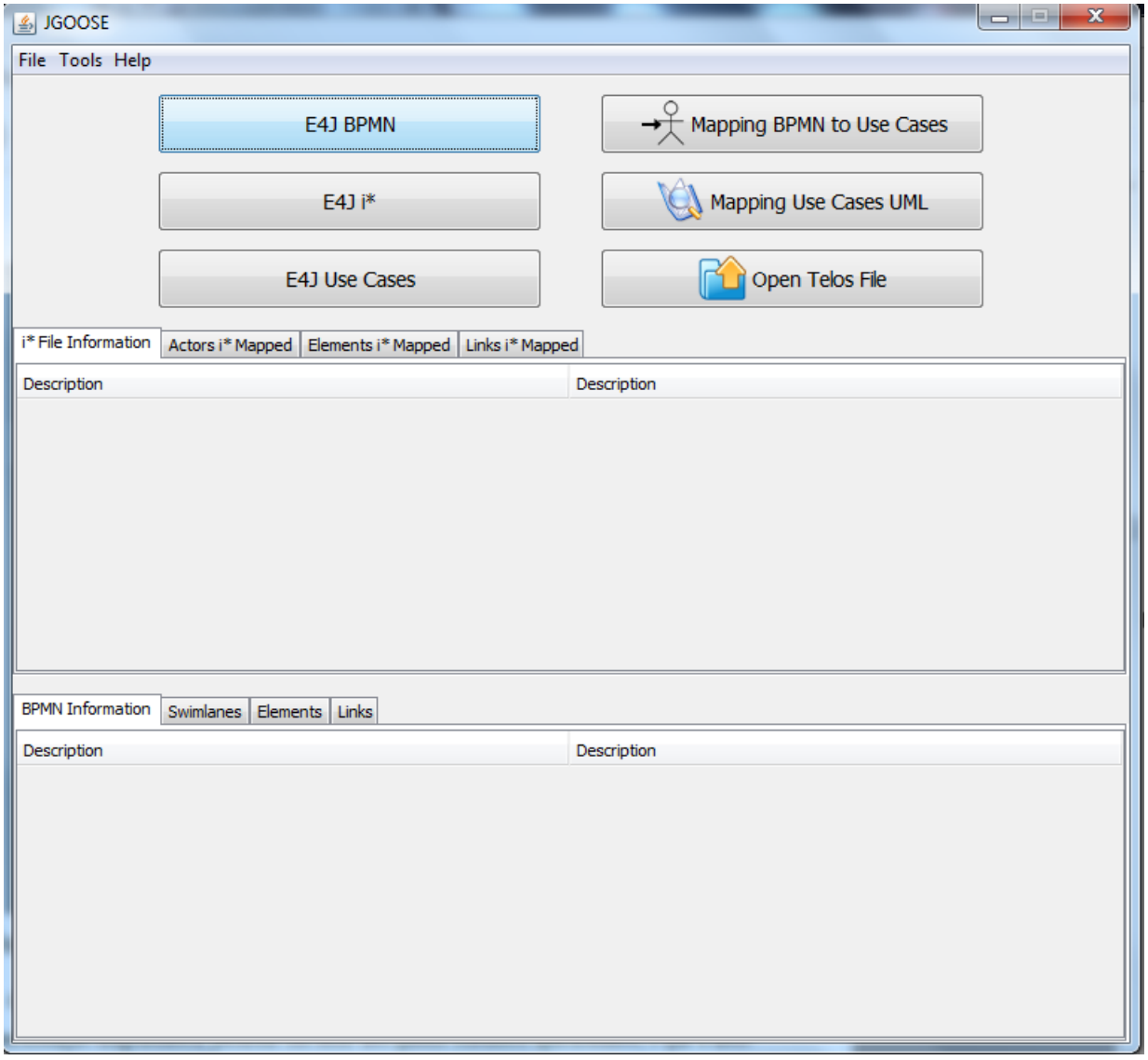



Figura 2.10: Tela Inicial JGOOSE.

A figura 2.11 mostrada a seguir, apresenta a tela inicial do E4J i*, na qual podemos construir modelos organizacionais SD e SR. Temos a área de trabalho na qual o usuário pode inserir, visualizar e manipular os elementos pertencentes ao modelo. Para inserir um novo elemento deve-se arrastar o elemento da paleta para a área de desenho. Temos a barra de menus, que é o conjunto de oito menus que permitem acessar funções envolvendo a aplicação. Os itens do menu são:

• File: funções envolvendo o arquivo, como Abrir, Salvar e Imprimir. É também através desse menu que é acessada a opção para obter os Casos de Uso correspondentes ao modelo;

• Edit: contém as opções Copiar, Colar, Recortar, Seleção e Refazer/Desfazer;

• View: possibilita alterações na área de desenho, como alteração do zoom;

• Format: permite personalizar os elementos selecionados;

• Shape: opções relacionadas com agrupamento de elementos;

• Model: permite realizar alterações na área de desenho, como a cor de fundo;

• Options: possibilita alterar opções gerais da aplicação;

• Help: exibe informações úteis ao usuário.

Podemos visualizar a barra de ferramentas, que é o conjunto de atalhos para as funções mais utilizadas durante a edição de modelos, como Abrir, Salvar, Copiar, Colar, Desfazer e Refazer, e ferramentas para personalização dos elementos.

Abaixo da barra de ferramentas ao lado esquerdo temos a paleta de elementos contendo cinco abas que agrupam os elementos i* em categorias. As abas e seus respectivos elementos são:

• Actors: Actor, Agent, Role, Position.

• Actor Associations: ISA, Is-part-off, Plays, Covers, Occupies e Instance of.

• Dependency Elements: Goal, Task, SoftGoal e Resource.

• Relationship Links: Dependency, Means-end, e Task-decomposition.

• Contribution Links: Make, Some+, Help, Break, Some-, Hurt, Unknown, And e Or.

E abaixo da paleta de elementos temos o minimapa, que é uma miniatura do modelo inteiro, sua função é servir de orientação em modelos muito grandes.

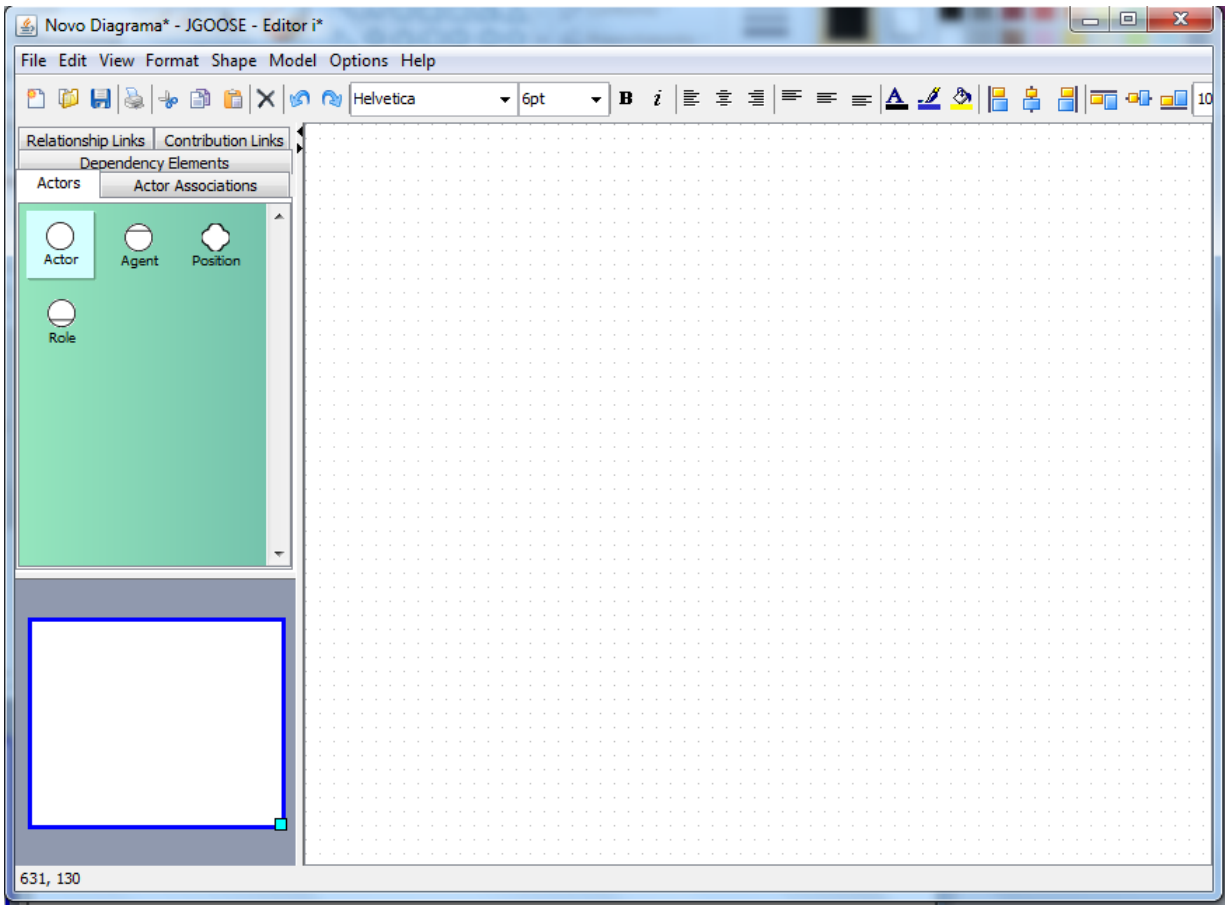



Figura 2.11: Tela Inicial Módulo i*.

Para a demonstração do uso foi construído o modelo SR da figura 2.12. Esse modelo representa a primeira etapa da proposta de derivação de Casos de Uso a partir de i* na qual o usuário deve construir o modelo que será usado para derivar os casos de uso na forma de diagrama e descrição textual. Este modelo SR representa uma organização que deseja obter um sistema computacional que apoie seu processo de compra e venda. Para acionar o JGOOSE para o mapeamento de casos de uso, basta seguir os passos presentes na Fig. 2.12: (1) clicar no menu "File" e em seguida clicar no submenu (2) "Generate Use Cases". O fluxo de atividades do E4J i* se encerra e o JGOOSE é acionado exibindo uma tela para seleção do ator que representa o sistema computacional. Para o exemplo da Fig. 2.12, deve-se selecionar o ator "Sistema". Após essa seleção, é possível visualizar informações sobre os atores, elementos e ligações existentes no modelo SR.

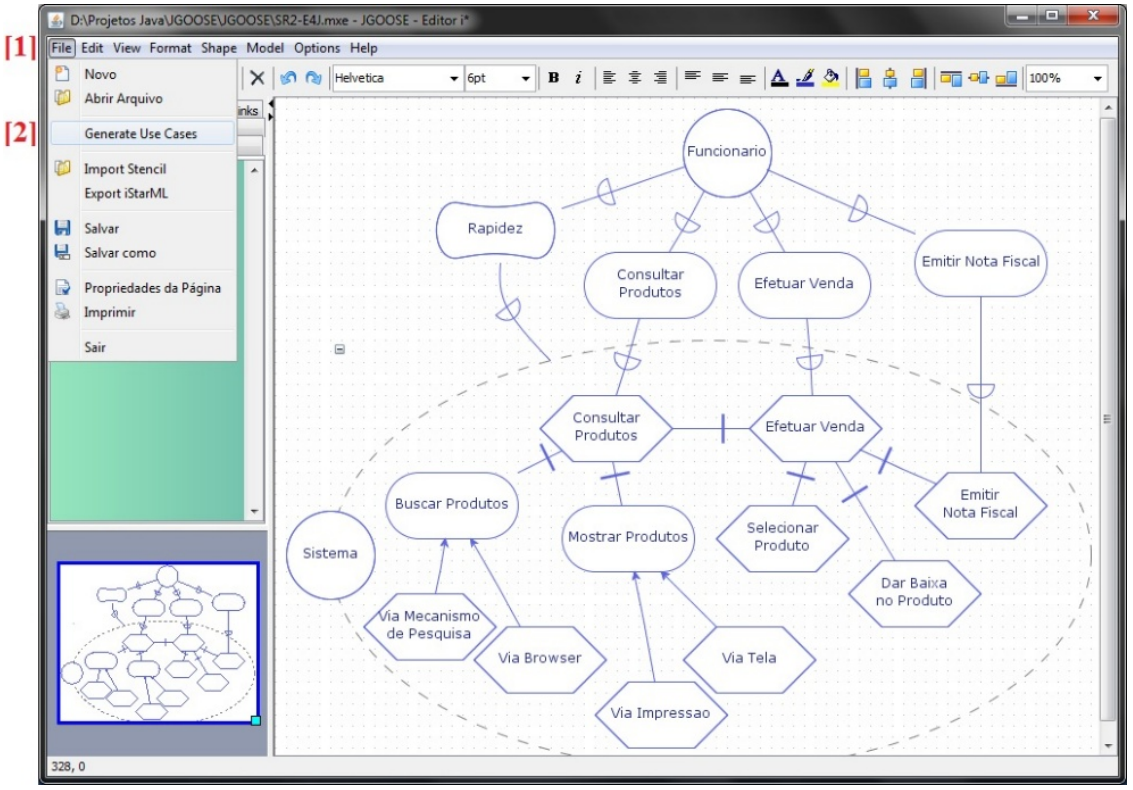



Figura 2.12: Tela Modelo i*.

O próximo passo é mapear os casos de uso UML. Quando o usuário clicar no botão “Mapping Use Cases UML” (Fig. 2.10), serão aplicadas as diretrizes presentes em [6] e [13] e ativada a tela da Fig. 2.13, que é a segunda etapa da proposta de derivação de Casos de Uso, "Obtenção da Descrição Textual". É possível visualizar nesta figura que o único ator mapeado para o diagrama é o ator “Funcionário” e que os casos de uso “Consultar Produtos”, “Efetuar Venda” e “Emitir Nota Fiscal”, associados ao mesmo, correspondem às dependências entre este ator e o sistema computacional pretendido (ver Fig. 2.12). Na Fig. 2.13 também está presente o botão “Diagram”, que quando acionado, o fluxo de atividades do JGOOSE se encerra, a rotina de mapeamento descrita anteriormente é executada e o E4J Use Cases é acionado contendo o diagrama de casos de uso gerado.

Para melhor detalhar a janela, dividiu-se a figura em oito áreas, comentadas a seguir.

• Barra de menus: permite ao usuário salvar os Casos de uso obtidos (File) ou exibe informações de auxílio ao usuário (Help);

• Use Cases Mapped: tabela contendo informações básicas dos Casos de Uso obtidos;

• Especificação do Caso de Uso: exibe o template de descrição textual para o Caso de Uso

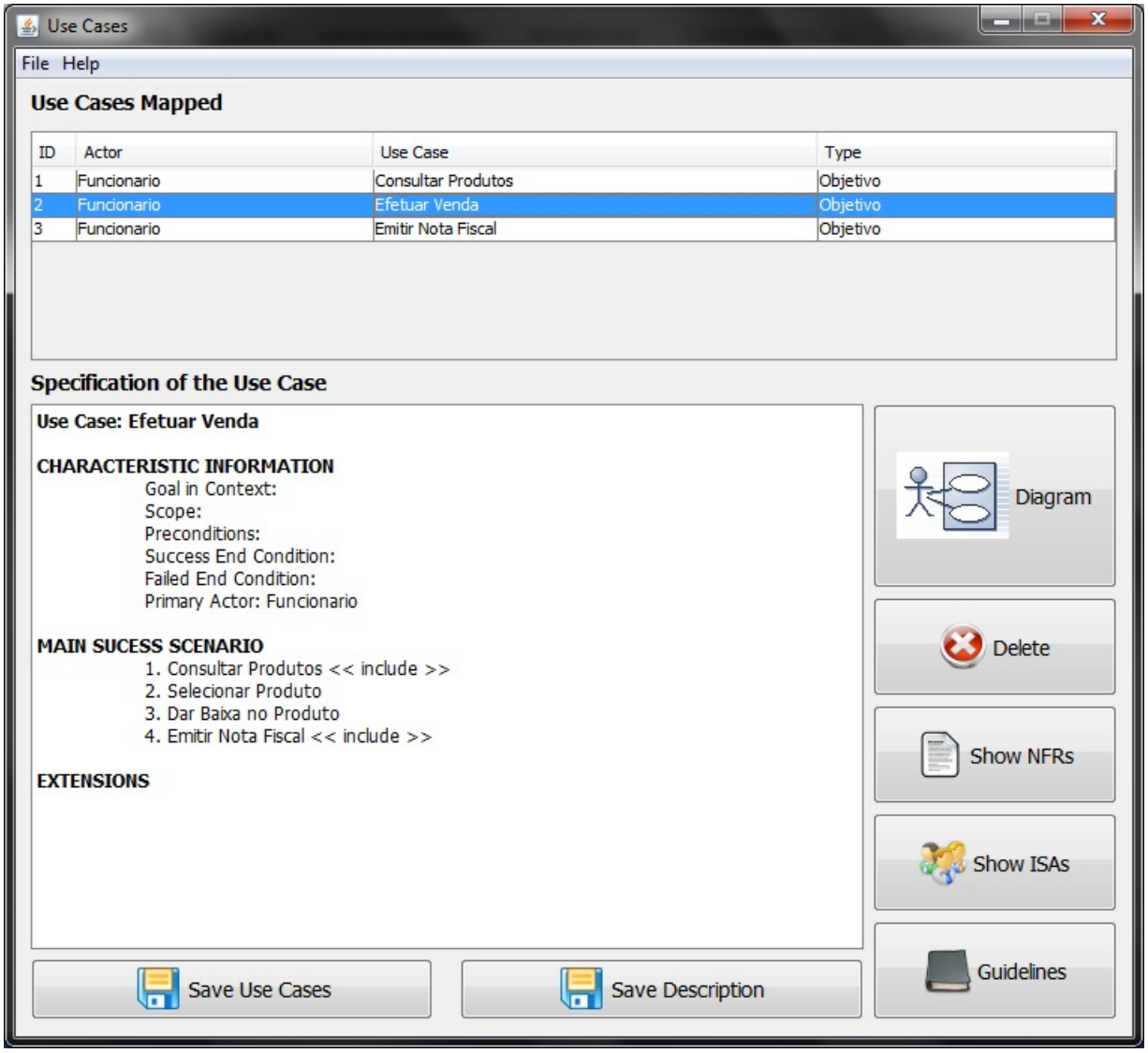



Figura 2.13: Tela Use Cases i*.

selecionado na tabela. Permite alterar as informações obtidas;

• Diagram: aciona o E4J Use Cases para exibir o Diagrama de Casos de Uso obtido, permitindo ao usuário realizar alterações no diagrama;

• Delete: remove o Caso de Uso selecionado na tabela;

• Guidelines: exibe ao usuário as diretrizes utilizadas no processo de mapeamento de Casos de Uso;

• Save Use Cases: armazena o Diagrama de Casos de Uso em local definido pelo usuário;

A tela principal do E4J BPMN é apresentada na Figura 2.14 e para melhor descrevê-la, a mesma é comentada a seguir:

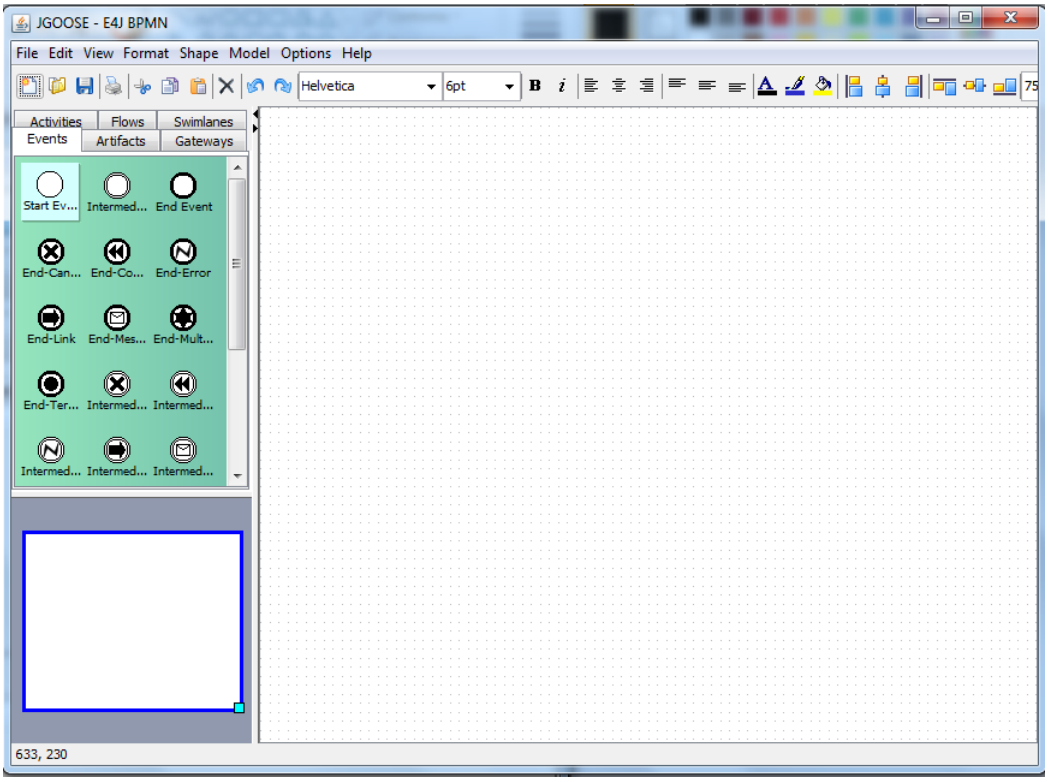



Figura 2.14: Tela Inicial E4J BPMN.

Temos a área de trabalho na qual o usuário pode inserir, visualizar e manipular os elementos pertencentes ao modelo. Para inserir um novo elemento deve-se arrastar o elemento da paleta para a área de desenho. Temos a barra de menus, que é o conjunto de oito menus que permitem acessar funções envolvendo a aplicação. Os itens do menu são:

• File: funções envolvendo o arquivo, como Abrir, Salvar e Imprimir. É também através desse menu que é acessada a opção para obter os Casos de Uso correspondentes ao modelo;

• Edit: contém as opções Copiar, Colar, Recortar, Seleção e Refazer/Desfazer;

• View: possibilita alterações na área de desenho, como alteração do zoom;

• Format: permite personalizar os elementos selecionados;

• Shape: opções relacionadas com agrupamento de elementos;

• Model: permite realizar alterações na área de desenho, como a cor de fundo;

• Options: possibilita alterar opções gerais da aplicação;

• Help: exibe informações úteis ao usuário.

Também temos a barra de ferramentas que é o conjunto de atalhos para as funções mais utilizadas durante a edição de modelos, como Abrir, Salvar, Copiar, Colar, Desfazer e Refazer, e ferramentas para personalização dos elementos. A Paleta de elementos que agrega seis abas com os elementos BPMN em categorias. As abas e seus respectivos elementos são:

• Activities: tarefa e subprocesso;

• Flows: fluxo de sequência, fluxo de mensagem e associação;

• Swimlanes: pool e lane;

• Events: eventos agrupados de acordo com o tipo (início, intermediário ou final);

• Artifacts: anotação de texto, objeto de dados, base de dados e grupo;

• Gateways: paralelo, exclusivo, complexo, padrão (sem símbolo associado), inclusivo e baseado em evento.

E também é mostrado o minimapa, que corresponde a uma miniatura do modelo apresentado na área de desenho. Possui função de auxiliar a manipulação de modelos cujo tamanho é maior que a porção exibida na área de desenho. Contém um retângulo azul que representa a porção do modelo que está sendo exibida na área de desenho;

Para exemplificação da abordagem será adotado o modelo de processo de negócio correspondente à escolha dos laureados com o Prêmio Nobel de Medicina. A construção do modelo BPMN deve ser realizada neste editor BPMN para que as diretrizes apresentadas nas seção 2.4.2 possam ser aplicadas de forma automatizada na JGOOSE. A descrição do processo e o correspondente modelo BPMN foram baseados no material disponibilizado em [17]. O modelo BPMN é apresentado na Figura 2.15 e na Figura 2.16 é o resultado do processo de derivação.

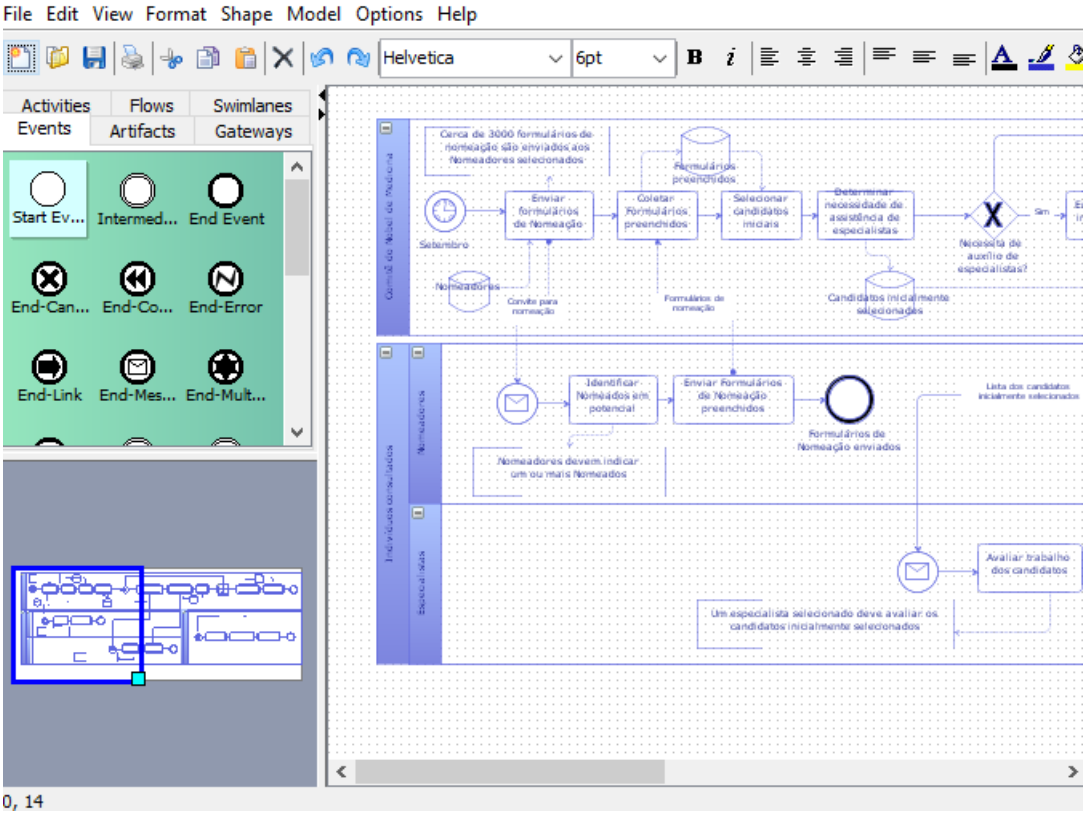



Figura 2.15: Tela com Modelo BPMN.

Para melhor detalhar a tela da Figura 2.16, que representa a segunda etapa da proposta de derivação de Casos de Uso, a mesma foi dividida em oito áreas, comentadas a seguir:

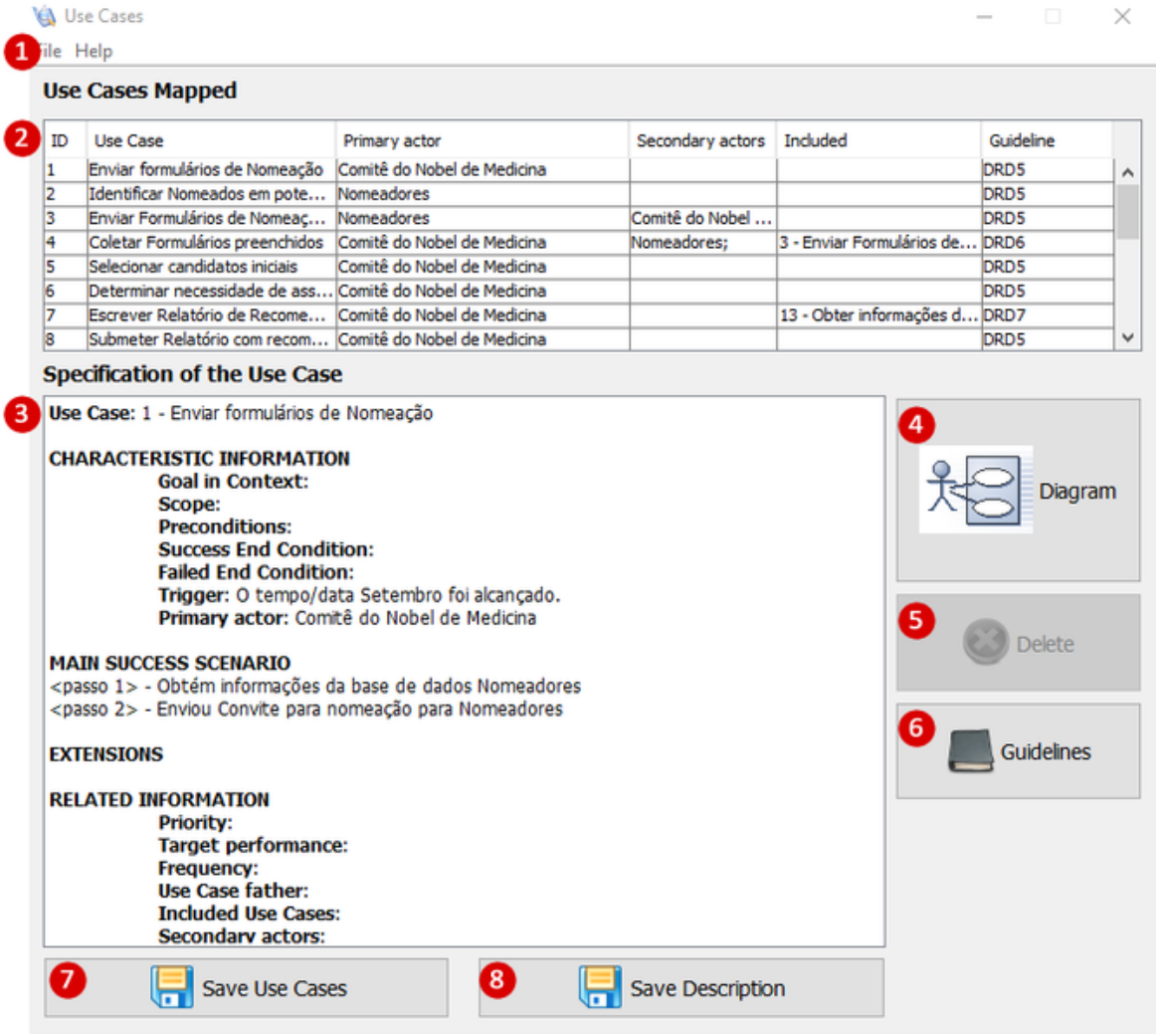



Figura 2.16: Tela Use Cases BPMN.

Acima temos a barra de menus (1), que permite ao usuário salvar os Casos de uso obtidos (File) ou exibe informações de auxílio ao usuário (Help). Abaixo temos a janela (2) "Use Cases Mapped", que é a tabela contendo informações básicas dos Casos de Uso obtidos, abaixo temos a especificação do Caso de Uso (3) que exibe o template de descrição textual para o Caso de Uso selecionado na tabela, permite alterar as informações obtidas. Botão Diagram (4) a direita, que aciona o E4J Use Cases para exibir o Diagrama de Casos de Uso obtido, permitindo ao usuário realizar alterações no diagrama. Botão delete (5), que remove o Caso de Uso selecionado na tabela. Também temos a opção Guidelines (6), que exibe ao usuário as diretrizes utilizadas no

processo de mapeamento de Casos de Uso. Por último temos os botões para salvar. O primeiro é o Save Use Cases (7), que armazena o Diagrama de Casos de Uso em local definido pelo usuário, e o Save Description (8), que armazena a descrição textual dos Casos de Uso em local definido pelo usuário.

## 2.6 Considerações finais

Nesse capítulo foram apresentados conceitos sobre os métodos de modelagem que serão utilizados no experimento. Na seção 2.1 o framework i* e seus modelos (SD e SR) são mostrados através de exemplos práticos. Na seção 2.2 o modelo BPMN é explicado também com exemplos práticos. Na seção 2.3 é abordado o conceito de casos de uso bem como o diagrama UML e a descrição textual dos mesmos. Na seção 2.4 são apresentadas as propostas de derivação que serão utilizadas como base para a análise dos resultados finais do experimento e por último, na seção 2.5, apresenta-se a ferramenta de apoio JGOOSE, editor E4J i*, o editor E4J BPMN, funcionalidades para derivação de Casos de Uso a partir de um modelo i* e BPMN. A interface gráfica do usuário foi detalhada, bem como os passos necessários.

Com o E4J, a JGOOSE torna-se um ambiente *standalone* que permite a construção de modelos organizacionais i*, diagramas de Casos de Uso UML e modelos de processo de negócio BPMN. Com a funcionalidade "BP2UC"e "Mapping Use Cases UML"o processo de aplicação é automatizada. A visualização do Diagrama de Casos de Uso gerado é feita através do E4J Use Cases, demonstrando a integração entre as funcionalidades presentes na JGOOSE.

# Capítulo 3

# Engenharia de Software Experimental

Neste capítulo são apresentados os conceitos básicos sobre a engenharia de software experimental com foco na descrição do método experimental. Na seção 3.1 são apresentados conceitos gerais básicos sobre a engenharia de software experimental. Na seção 3.2 são apresentados conceitos básicos sobre experimentos na engenharia de software. Na seção 3.3, dentre as 4 estratégias de experimentação apresentadas por [18], uma delas foi escolhida descrevendo o motivo pelo qual isso aconteceu. Na seção 3.4, as considerações finais do capítulo são descritas.

## 3.1 Conceitos gerais

Experimentação pode ser considerada uma das partes mais importantes do processo científico. É através de experimentos que as teorias são verificadas, fatores são explorados para que novas teorias possam ser formuladas ou refeitas. Com a experimentação obtêm-se vantagens para a avaliação da atividade humana. Portanto, é necessário avaliar pesquisas comparando com as existentes.

De acordo com Wohlin [18], as metodologias específicas são necessárias para ajudar a estabelecer uma base de engenharia e de ciência para a Engenharia de Software. Existem quatro métodos relevantes para condução de experimentos na área de Engenharia de Software: **científico, de engenharia, experimental, e analítico**.

O **método científico** sugere o modelo ou a teoria de comportamento, mede e analisa, verifica as hipóteses do modelo ou da teoria. Esse método pode ser utilizado quando o produto de software, ambiente, é analisado. Ele faz a avaliação se o modelo é realmente representativo para o fenômeno que está sob observação.

O **método de engenharia** observa as soluções existentes, sugere as soluções mais adequadas, mede e analisa, e replica alterações até o momento que as melhorias sejam todas solucionadas. É uma abordagem orientada à melhoria iterativa que sugere a existência de um modelo, processo ou produto, e a partir disso, tem o propósito de melhora a cada modificação.

O **método experimental** apresenta o modelo, desenvolve o método qualitativo e/ou quantitativo, aplica um experimento, avalia o modelo e dessa forma, o processo é repetido. O processo se inicia com o levantamento de um modelo novo, não necessariamente baseado em um modelo já existente, tentando obter resultados satisfatórios com o efeito do processo ou produto sugerido pelo modelo novo. É importante deixar claro que para o método experimental diversas técnicas devem ser aplicadas, bem como deve ser definido um protocolo de experimentação, conforme será explicado no Capítulo 4.

O **método analítico (ou matemático)** sugere uma teoria formal, desenvolve a teoria, encontra os resultados e se possível, realiza uma comparação com as observações empíricas. É um método dedutivo que não se comporta de forma a ser um projeto experimental no sentido estatístico, mas serve de base para o desenvolvimento de modelos.

É importante observar que experimentos não oferecem prova com 100% de certeza do resultado, eles verificam uma realidade teórica próxima ao real.

## 3.2 O método experimental

Os objetivos relacionados à execução de experimentos em Engenharia de Software são a caracterização, avaliação, previsão, controle e melhoria a respeito de produtos, processos, recursos, modelos, teorias entre outros.

A literatura oferece várias definições dos objetivos da experimentação em Engenharia de Software. Em [49] são listados:

- Para compreender a natureza dos processos da informação os pesquisadores devem observar o fenômeno, encontrar explicação, formular a teoria, e verificá-la.

- A experimentação pode ajudar a construir uma base de conhecimento confiável e reduzir assim incerteza sobre quais teorias, ferramentas, e metodologias são adequadas.

- A observação e experimentação podem levar a novos e úteis meios da introspecção, e abrir novas áreas de investigação. A experimentação pode encontrar novas áreas onde a engenharia

age lentamente.

- A experimentação pode acelerar o processo eliminando abordagens inúteis e suposições errôneas. A experimentação ajuda também a orientar a engenharia e a teoria nas direções promissoras de pesquisa.

- Os experimentos podem ser custosos, mas um experimento significativo geralmente pode se encaixar no orçamento de um pequeno laboratório. Por outro lado, um experimento caro pode valer a pena muito mais do que seu custo e, por exemplo, oferecer à companhia liderança de três, quatro ou cinco anos sobre a competição.

- O crescimento do número de trabalhos científicos com uma validação empírica significativa possui a boa chance de acelerar o processo de formação da Engenharia de Software como ciência. As idéias duvido sãs serão rejeitadas mais rapidamente e os pesquisadores poderão concentrar-se nas abordagens promissoras.

- A tecnologia vem se modificando rapidamente. As mudanças sempre trazem ou eliminam as suposições. Os pesquisadores devem então antecipar as mudanças nas suposições e aplicar os experimentos para explorar as conseqüências dessas mudanças.

Um experimento é realizado através de um processo, como a maioria dos casos que envolvem o desenvolvimento de um software. Uma das principais vantagens de um experimento é o controle, por exemplo, sujeitos, objetos e instrumentação. O ponto inicial é de que temos uma ideia de uma relação de causa e efeito, ou seja, existe uma relação entre uma construção de causa e uma construção de efeito. A partir disso, Wohlin [18] afirma que se temos uma teoria somos capazes de formular uma hipótese, que nada mais significa que temos uma ideia, e dessa forma temos um relacionamento, ainda podendo declarar formalmente uma hipótese. Para avaliar determinado assunto, podemos usar um experimento, ele é criado por exemplo, para testar uma teoria ou uma hipótese. Na concepção do experimento, temos um número de tratamentos sobre o qual temos controle. O experimento é realizado e somos capazes de observar o resultado. Isso significa que testamos a relação entre o tratamento e o resultado, se a experiência estiver configurada corretamente, poderemos desenhar conclusões sobre a relação entre a causa e o efeito final.

Os elementos principais do experimento são as **variáveis, os objetos, os participantes, o**

**contexto do experimento, hipóteses, e o tipo de projeto do experimento**. Há dois tipos de variáveis do experimento: dependentes e independentes. As **variáveis independentes** referem-se à entrada do processo de experimentação. Essas variáveis também se chamam "fatores"e apresentam a causa que afeta o resultado do processo de experimentação. O próprio valor de um fator se chama "tratamento". As **variáveis dependentes** referem-se à saída do processo de experimentação. Essas variáveis apresentam o efeito que é causado pelos fatores do experimento. O próprio valor de uma variável dependente se chama "resultado". A figura 3.1 representa o processo de um experimento.

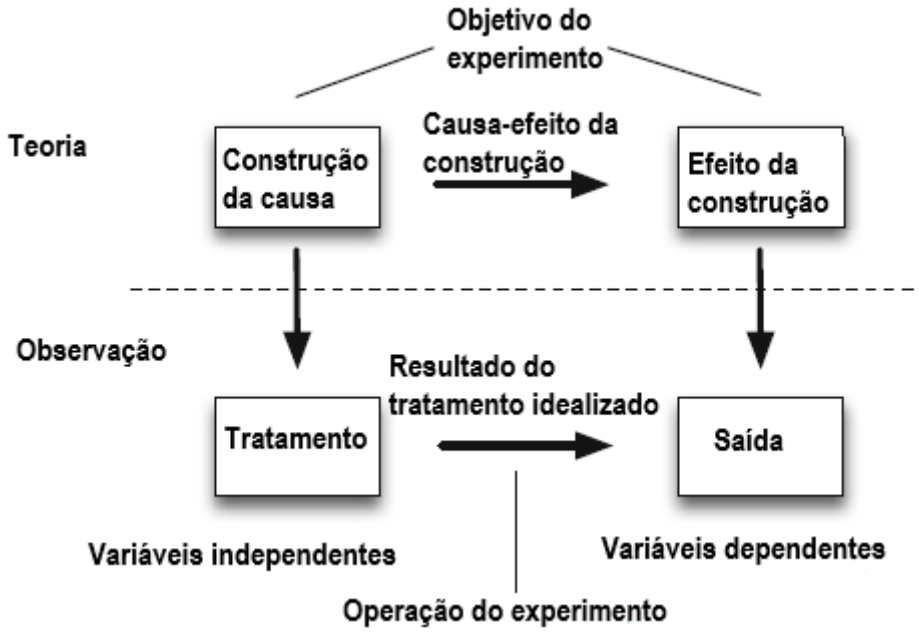



Figura 3.1: Processo de um experimento [18].

O **objeto** é uma ferramenta usada para verificar o relacionamento causa-efeito numa teoria. Durante a execução do experimento os tratamentos estão sendo aplicados ao conjunto dos objetos e assim o resultado está sendo avaliado. Os objetos junto com o sistema de medição e diretrizes da execução do experimento compõem a instrumentação do experimento.

Os **participantes** são os indivíduos que foram selecionados da população sob interesse para conduzir o experimento. Isso significa que para generalizar os resultados de um experimento a uma população desejada, o conjunto de participantes deve ser representativo para aquela população.

O **contexto do experimento** é composto das condições em que o experimento está sendo

executado.

Um experimento geralmente é formulado através de **hipóteses**. A hipótese principal se chama hipótese nula e declara que não há nenhum relacionamento estatístico significante entre a causa e o efeito. O objetivo principal do experimento é, então, rejeitar a hipótese nula a favor de uma ou algumas hipóteses alternativas, que neste caso é a hipótese 1 ou 2.

O processo de experimentação não é um processo simples, é necessário preparar, conduzir e analisar corretamente, esse processo representa etapas de uma atividade. Os processos são importantes, pois podem ser utilizados como listas de verificação e diretrizes de como e o que fazer. Para a realização de um experimento vários passos devem ser tomados e eles têm que estar em uma certa ordem, dessa forma, um processo de como realizar experimentos é necessário. Abaixo, na figura 3.2 é apresentado uma visão geral sobre o processo de experimento controlado, dividido em algumas etapas consideradas importantes para o sucesso da pesquisa.

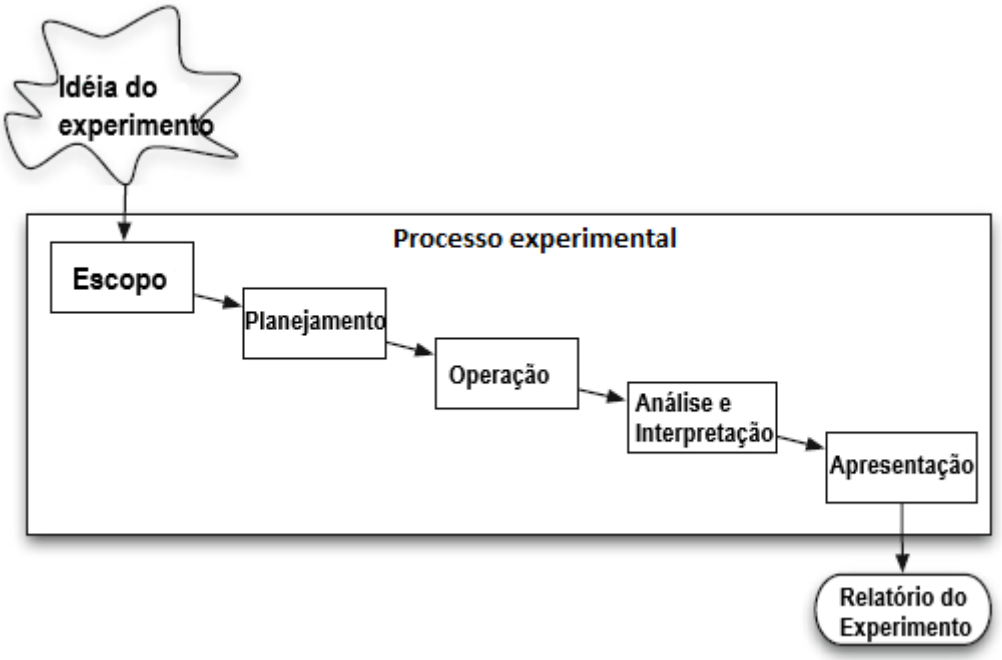



Figura 3.2: Visão geral do experimento[18].

De acordo com Wohlin [18] um experimento é constituído de cinco fases, nas seguintes atividades principais. **Escopo** é o primeiro passo, em que podemos analisar o experimento em termos de problema e objetivos. O **planejamento** vem a seguir, onde o design do experimento é determinado, a instrumentação é considerada e as ameaças ao experimento são avaliadas. Na

atividade **operacional**, são recolhidas medições que depois são analisadas e avaliadas na **análise e interpretação**. Finalmente, os resultados são apresentados e embalados em **apresentação e pacote**.

No processo não se assume que uma atividade é necessariamente finalizada antes que a próxima seja iniciada. A ordem de atividades no processo indica principalmente a ordem de início das atividades. Em outras palavras, o processo é parcialmente iterativo e pode ser necessário voltar atrás e refinar uma atividade anterior antes de continuar com o experimento. A principal exceção é quando um processo do experimento começou, então não é possível voltar atrás para o escopo e planejamento do experimento. Dessa forma, pode ser observado que os sujeitos são influenciados pelo experimento, e se de alguma forma o processo voltar atrás, existe o risco de ser impossível usar os mesmos sujeitos ao retornar a fase de operação do processo experimental. No Capítulo 4, será apresentado o protocolo de experimentação, com todas as fases bem detalhadas ao leitor.

## 3.3 Estratégia a ser aplicada

Em [18] são descritas 4 estratégias de experimentação em engenharia de software, sendo elas:

• **Pesquisa** (*survey*), que é um sistema para coletar informação de ou sobre pessoas para descrever, comparar ou explicar seu conhecimento, atitudes ou comportamento.

• **Estudo de caso** (*case study*), que é um inquérito empírico que se embasa em diversas fontes de evidência para investigar uma instância de um fenômeno da engenharia de software dentro de seu contexto em vida real.

• **Experimento** (*experiment*), é o inquérito empírico que manipula um fator ou variável do contexto estudado. Baseado em aleatoriedade, diferentes tratamentos são aplicados para ou por diferentes sujeitos, enquanto se mantém as outras variáveis constantes e medindo os efeitos nas variáveis resultado.

• **Quase-Experimento** (*quasi-experimen*t), é um inquérito empírico similar a um experimento, onde a atribuição de tratamentos aos sujeitos não pode ser feito de maneira aleatória, emergindo das características dos sujeitos ou objetos em si.

Neste trabalho a estratégia escolhida foi o Quase-Experimento. O motivo de escolha foi a

dificuldade de realização de um Estudo de Caso ou Experimento, o que necessitaria um longo período de acompanhamento assim como cooperação de membros da indústria, pois seria necessário aleatoriedade e diferentes sujeitos.

Para isso será comparado o resultado obtido pelos participantes em um experimento considerando dois sistemas de informação cujos requisitos devem ser definidos (ver capítulo 5), os casos de uso serão comparados em relação ao resultado previamente elaborado consistente de descrição textual e um diagrama de casos de uso UML. A comparação entre os resultados obtidos dos participantes do experimento em relação aos resultados previamente elaborados para os problemas tratados serão baseados essencialmente na corretude e completude dos casos de uso com o objetivo de se fazer uma análise do quase-experimento, dessa forma, respondendo as questões e métricas buscadas.

## 3.4 Considerações finais

Neste capítulo foram apresentados conceitos e técnicas da Engenharia de Software Experimental. Na seção 3.1, conceitos gerais e métodos da Engenharia de Software Experimental foram discutidos. Na seção 3.2 a descrição de como o Quase-Experimento será realizado é apresentado, objetivos relacionados à execução de experimentos, bem como o processo de acordo com [18]. Na seção 3.3 a justificativa da estratégia a ser utilizada é descrita, apontando motivos que foram encontrados para isso.

# Capítulo 4

# Protocolo de Experimentação

O experimento a ser desenvolvido tem por objetivo avaliar a corretude e completude da derivação de Casos de Uso a partir de modelos i* e BPMN utilizando a ferramenta JGOOSE [16],[17] e [42]. Como método de avaliação esse experimento foi definido visando obter dados necessários a uma análise posterior permitindo obter conclusões a respeito dos resultados obtidos utilizando a referida referida ferramenta na geração de Casos de Uso, tanto no aspecto de corretude e completude de elementos textuais e gráficos gerados.

O experimento possui como objetivo responder a pergunta principal da pesquisa conforme segue:

• Quais são as diferenças na comparação das técnicas i* e BPMN na derivação e obtenção de Casos de Uso a partir do software JGOOSE com suas respectivas ferramentas automáticas no processo de derivação (E4J i*, E4J BPMN).

Essa é uma pergunta difícil de ser respondida, já que isso depende muito de qual ambiente o sistema está inserido, quais as variáveis que estão em questão e fatores externos como os indivíduos participantes (experiência, motivação, etc). Contudo, um experimento permite definir variáveis dependentes e independentes (Seção 4.2.3) as quais podem ser controladas de forma a obter os resultados desejados em um contexto específico industrial ou acadêmico.

A seguir descreve-se o protocolo de experimentação conforme proposto em [18].

## 4.1 Definição do Escopo do Experimento

Nesta seção apresenta-se a definição do escopo do quase-experimento, onde é abordada a definição do objetivo e suas características, apresentando qual o objetivo geral do quase-

experimento, também é apresentado o objeto de estudo onde são relacionados as técnicas (i* e BPMN), o propósito do quase-experimento e sua perspectiva no ponto de vista de visualizar qual técnica apresenta melhores resultados na derivação de casos de uso, o foco de qualidade no aspecto prático e o contexto onde esse trabalho é inserido, bem como o resumo do escopo.

### 4.1.1 Definição do Objetivo

O estudo a ser realizado tem como objetivo geral encontrar através de uma comparação entre duas técnicas i* e BPMN, quais as vantagens e desvantagens no âmbito de derivação automática de Casos de Uso utilizando a ferramenta JGOOSE, levando em consideração a corretude e completude desses Casos de Uso. O quase-experimento é motivado pela necessidade de se obter Casos de Uso com a maior corretude e completude possível, referentes ao uso de processos BPMN e i* no desenvolvimento de software, assim como avaliar o funcionamento dos módulos E4J i* e BPMN dentro da ferramenta JGOOSE.

**Objeto de estudo:** Os objetos de estudos são os modelos organizacionais i* e BPMN, e a influência que o uso destes traz na obtenção de casos de uso, assim como os editores i* e BPMN da ferramenta JGOOSE juntamente com as funcionalidades que permitem derivar casos de uso de forma automatizada.

**Propósito:** O propósito do quase-experimento foi obter dados qualitativos e quantitativos a respeito do uso de modelos organizacionais i* e BPMN em obtenção de casos de uso UML. O experimento permitiu avaliar as vantagens e desvantagens do uso das técnicas e ferramenta JGOOSE na obtenção de casos de uso.

**Perspectiva:** A perspectiva é do ponto de vista do pesquisador que quer saber se o uso de uma técnica apresenta melhores resultados em relação à outra ou ainda se as mesmas são complementares na obtenção de casos de uso.

**Foco de Qualidade:** O efeito principal a ser estudado no quase-experimento é a obtenção, por parte dos sujeitos, dos casos de uso correspondentes ao tipo de técnica utilizada, i* ou BPMN. O aspecto a ser avaliado não é o formal, mas sim o prático.

**Contexto:** O contexto de quase-experimento é a realização de um projeto de trabalho de conclusão de curso, que é uma atividade obrigatória dentro do curso de Ciência da Computação,

o quase-experimento será realizado em um dos laboratórios do curso. Os participantes do quase-experimento serão os alunos de Ciência da Computação cursando a disciplina de Processo de Engenharia de Software I no ano de 2019.

### 4.1.2 Resumo do Escopo

Para resumir o escopo do experimento utilizamos o formato proposto pela técnica GQM [16][47]. Assim, o escopo do experimento envolve:

**Analisar** os casos de uso textuais e diagramáticos gerados a partir do uso da ferramenta JGOOSE.

**Com a finalidade** de verificá-los.

**Com respeito** à corretude e completude do número de casos de uso textuais seguindo o template proposto por [19] e diagrama de Casos de Uso gerado considerando a notação UML.

**Do ponto de vista** dos stakeholders.

**No contexto** do uso da ferramenta JGOOSE e editores E4J por discentes da disciplina de Processo de Engenharia de Software I do curso de Ciência da Computação da Unioeste.

## 4.2 Planejamento

Nesta seção o foco está em descrever o planejamento do experimento, este planejamento descreve em detalhes o contexto no qual o experimento é feito, a formulação das hipóteses a serem consideradas conforme as questões da pesquisa, a seleção das variáveis dependentes e independentes que influenciam o experimento e a seleção dos sujeitos considerados no experimento, neste caso, acadêmicos de um curso de graduação em Ciência da Computação. Se descreve também o *design* do experimento definindo entre outros pontos os grupos de controle do experimento bem como a instrumentação do experimento na qual se define quais os instrumentos a serem usados na experimentação. Finalmente são apresentados os elementos que podem ser uma ameaça à validade do experimento bem como aqueles que podem fortalecer a confiabilidade dos resultados obtidos, como por exemplo, as métricas que serão utilizadas para avaliar os resultados através da abordagem GQM (*Goal Question Metric*).

### 4.2.1 Seleção de Contexto

O contexto do experimento é o desenvolvimento de um trabalho de conclusão de curso dentro do curso de Ciência da Computação da UNIOESTE campus Cascavel, sendo que o experimento será conduzido por um aluno de graduação. O experimento visa gerar dados sobre dois modelos já estabelecidos (i* e BPMN), assim como utilização de ferramenta desenvolvida por membros deste curso, que visa automatizar parte do processo de desenvolvimento de software (obtenção de Casos de Uso).

### 4.2.2 Formulação das Hipóteses

De acordo com Wohlin [18], na fase de planejamento é necessário definir algumas hipóteses as quais devem ser utilizadas como ponto de partida do experimento. Para essas hipóteses terem validade, observa-se que a corretude e completude devem ser medidas. A completude foi medida através do template de casos de uso proposto por Cockburn [19], que também é apresentado na figura 2.6, considerando as características do template a serem preenchidas e após isso comparando cada template de caso de uso obtido de cada uma das duas técnicas (i* e BPMN), representando o quão completa está a descrição textual do caso de uso. Já corretude dos casos de uso obtidos foi medida através da comparação de três parâmetros, entre o obtido pelos sujeitos e o esperado, os quais foram definidos conforme segue:

- Primeiro: Existência de Caso de Uso correspondente ao esperado, sendo que para verificação da existência ou não da correspondência foi utilizada a descrição textual do Caso de Uso. Assim, se o Caso de Uso descoberto quando comparado ao esperado representar uma funcionalidade similar, este foi considerado correto.

- Segundo: Relação ao ator correto. Na análise de parâmetro, o Caso de Uso obtido esteve relacionado ao mesmo ator que o esperado conforme obtido no primeiro parâmetro. Cabe destacar que, caso o Caso de Uso obtido não satisfaça o primeiro parâmetro é impossível que satisfaça o segundo.

- Terceiro: Relação aos demais Casos de Uso. Foi avaliada a presença de relação entre o Caso de Uso obtido e demais Casos de Uso, sendo que a avaliação teve como foco averiguar a presença de relações do tipo <*include*> e <*extend*> de acordo com o esperado. Observa-se

que o terceiro parâmetro apenas foi avaliado após os dois primeiros parâmetros terem sido avaliados em todos os Casos de Uso obtidos, pois apenas foram considerados como Casos de Uso válidos para avaliação de relação, aqueles que satisfizerem os dois primeiros parâmetros. Do mesmo modo, apenas relações esperadas e existentes entre Casos de Uso válidos foram consideradas como tendo satisfeito o terceiro parâmetro.

Deve observar-se que um determinado Caso de Uso obtido no decorrer do experimento pode estar incorreto, parcialmente correto ou correto, por outro lado, a completude não foi mensurada de tal forma, e sim comparando elemento a elemento dos templates de caso de uso resultantes de cada uma das técnicas (i* e BPMN) utilizadas, portanto, não existiu um caso de uso parcialmente completo ou completo e sim um mais ou menos completo que o outro. Essa avaliação deve ser realizada separadamente para obter-se resultados para cada uma das avaliações que estão sendo propostas, corretude e completude.

No que diz respeito à corretude, já foram citadas acima que são 3 parâmetros que podem envolver os casos de uso derivados:

Caso de Uso correto: o caso de uso gerado existe e faz sentido em relação ao que se espera bem como suas relações com o ator e demais casos de uso são todas corretas;

Caso de Uso parcialmente correto: o caso de uso gerado está correto pois existe no resultado esperado mas nem todas as relações com o ator e demais casos de uso são corretas;

Caso de Uso incorreto: o caso de uso gerado não existe nos resultados esperados e por consequência as suas relações não cabem a ser avaliadas.

No que diz respeito à completude, foram investigadas na literatura algumas informações sobre as características que vários templates utilizam, conforme em [54] as tabelas 4.1 e 4.2 mostra como alguns autores consideram a construção de templates que auxiliam na especificação de casos de uso.

| Autor | [56] | [57] | [58] | [59] | [60] | [61] | [62] | [19] | [63] | [64] |
|---|---|---|---|---|---|---|---|---|---|---|
| Nome caso de uso | Y | Y | Y | Y | Y | Y | Y | Y | Y | Y |
| Objetivo no contexto | Y | | | | | | Y | Y | Y | |
| Trigger | | | | | | | Y | Y | | |
| Escopo | | | | | | | | | | |
| Sumário | | | Y | Y | Y | Y | Y | Y | | |
| Nível de abstração | | | | | | | | Y | | |
| Stakeholders | | | | | | | | Y | | |
| Stakeholders e outros interessados | | | | | | | | Y | | |
| Ator primário | | Y | Y | Y | Y | Y | Y | Y | | Y |
| Atores secundários | | | | | | | | | | Y |
| Requisito especial | | | | | | | | Y | | |
| Frequência | | | | | | | | Y | | |
| Prioridade | | | | | | | | Y | | |
| Objetivo de performance | | | | | | | | Y | | |
| Pré-condição | Y | Y | Y | Y | Y | Y | Y | Y | Y | |
| Pós-condição (Sucesso) | | Y | Y | | Y | Y | Y | Y | Y | |
| Pós-condição (Falha) | | | | | | | | Y | | |
| Data final | | Y | | | | | | | | |
| Super caso de uso | | | | | | | | Y | | Y |
| Dependência | | | | | | | | | | |
| Generalização | | | | | | | | | | |
| Cenário principal | Y | Y | Y | Y | Y | Y | Y | Y | Y | Y |
| Sub cenário | Y | | | | | | | | | Y |
| Cenário alternativo | Y | | Y | Y | Y | Y | Y | Y | Y | Y |
| Curso excepcional | Y | Y | | | | | | | | |
| Fluxo excepcional | | | | | | | | Y | | |
| Pontos de extensão | | | Y | Y | Y | | | Y | | Y |
| Variações | | | | | | | | Y | | |
| Referência cruzada p/ requisitos | | | | | | | | | | |
| Questões abertas | | | | | | | | Y | | |
| Uso de palavras | | | | | | | | Y | | Y |
| Constraints | | | | | | | | | | Y |
| Observações | | | | | | | | | | |

Tabela 4.1: 1 - Características de templates UC por cada autor.

| Autor | [65] | [66] | [67] | [68] | [69] | [70] | [71] | [72] | [73] | [74] |
|---|---|---|---|---|---|---|---|---|---|---|
| Nome caso de uso | Y | Y | Y | Y | Y | Y | Y | Y | Y | Y |
| Objetivo no contexto | Y | | Y | Y | Y | Y | Y | Y | | Y |
| Trigger | | | | | | | Y | Y | | |
| Escopo | | | | | | | | Y | | |
| Sumário | | | | | | | | | | |
| Nível de abstração | | | | | | | | Y | | |
| Stakeholders | | | | | | | | Y | | |
| Stakeholders e outros interessados | | | | | | | | Y | | |
| Ator primário | Y | Y | Y | | Y | Y | Y | Y | Y | Y |
| Atores secundários | Y | Y | Y | | Y | Y | Y | | Y | Y |
| Requisito especial | | | Y | | | | Y | Y | | |
| Frequência | | | | | | | | | | |
| Prioridade | | | | | | | | | | |
| Objetivo de performance | | | | | | | | | | Y |
| Pré-condição | Y | Y | | Y | Y | Y | Y | Y | | Y |
| Pós-condição (Sucesso) | Y | Y | | Y | Y | Y | Y | Y | | Y |
| Pós-condição (Falha) | | | | | | | | Y | | |
| Data final | | | | | | | | | | |
| Super caso de uso | | | | | | | | | | |
| Dependência | | Y | | | | Y | | | | |
| Generalização | | Y | | | | Y | | | | |
| Cenário principal | Y | Y | Y | Y | Y | Y | Y | Y | Y | Y |
| Sub cenário | | Y | | | | | | | | |
| Cenário alternativo | Y | Y | Y | Y | Y | Y | Y | Y | | Y |
| Curso excepcional | | | | | | | | | | |
| Fluxo excepcional | | Y | | | | | | | | |
| Pontos de extensão | Y | Y | | | | Y | Y | Y | Y | |
| Variações | Y | Y | | | | | | Y | | |
| Referência cruzada p/ requisitos | | | Y | | | | | | | |
| Questões abertas | | | | | | | | Y | | |
| Uso de palavras | | Y | | | Y | Y | | Y | Y | Y |
| Constraints | | Y | | | | | | | | |
| Observações | Y | | | | | | | | | |

Tabela 4.2: 2 - Características de templates UC por cada autor.

Nas tabelas 4.1 e 4.2, na primeira coluna constam os elementos considerados importantes em uma descrição textual de casos de uso. Assim, para cada proposta de template (primeira linha apresentando as referências aos trabalhos propostos) são preenchidos com "Y"quais elementos cada template contempla. Por exemplo, os templates que consideram uma maior quantidade de elementos de descrição textual são o trabalho de [72] e [19].

Neste trabalho, a descrição textual adotada para avaliar a completude foi uma adaptação do template que tem o formato *fully dressed form* proposto por Cockburn [19], visto que tem o objetivo de capturar um conjunto completo de informações. A decisão de optar pelo template de Cockburn também leva em consideração que em comparação com outros autores ele se apresenta mais informações a serem preenchidas, portanto, tornando o caso de uso mais detalhado. Para a validade da completude de cada caso de uso, considera-se que foram comparados os resultados obtidos das derivações das duas técnicas (i* e BPMN) entre si, para cada um dos problemas apresentados e propostos no Apêndice A. Para chegar a um resultado consistente, cada informação do template (Nome, objetivo no contexto, escopo, etc) do caso de uso resultante para o primeiro problema em i* foi comparado com cada informação do template (Nome, objetivo no contexto, escopo, etc) do caso de uso resultante para o primeiro problema em BPMN, da mesma forma, o processo foi replicado para o segundo problema. Algumas variações de linguística podem ocorrer dependendo do modo de como a modelagem é feita, dessa forma, consideramos conforme [55] que comparar elemento a elemento dos diferentes templates obtidos trouxe uma completude mais significativa.

Ao se comparar a quantidade de casos de uso corretos derivados no grupo 1 e 2 (Tabela 4.1), foi possível realizar uma análise sob o ponto de vista de influência que a modelagem organizacional i* tem na quantidade de casos de uso corretos e completos, o que serve como indício do quão eficaz é o uso de modelos organizacionais i* pela maturidade das diretrizes na derivação de casos de uso com o auxílio do editor E4J (i*) da ferramenta JGOOSE, observando se ela trás benefício ou não. Também foi possível analisar o modelo de processos de negócio BPMN, mostrando o quão eficaz é o uso deste modelo na derivação de casos de uso com o auxílio do editor E4J (BPMN) da ferramenta JGOOSE.

Ao se comparar a quantidade de casos de uso corretos e completos derivados entre os dois grupos foi possível avaliar o desempenho considerando a eficácia da ferramenta de apoio

JGOOSE e qual das duas técnicas se sobressai à outra em determinado contexto que está sendo inserido. Desta forma, as seguintes hipóteses foram usadas como base para analise dos resultados.

- Hipótese 1: a utilização dos modelos organizacionais i* juntamente da ferramenta E4J (i*) para derivação de Casos de Uso influenciou o resultado final, sendo que a quantidade de casos de uso corretos bem como estereótipos («include», «extend» e «generalization») e atores encontrados por quem utilizou é maior do que a quantidade encontrada por quem utilizou BPMN juntamente da ferramenta E4J (BPMN). Ou, ao contrário, a utilização de modelos de processo de negócio BPMN juntamente da ferramenta E4J (BPMN) para derivação de Casos de Uso influenciou o resultado final, sendo que a quantidade de casos de uso corretos bem como estereótipos e atores encontrados por quem utilizou é maior à quantidade de casos de uso corretos encontrados por quem utilizou i* juntamente da ferramenta E4J (i*).

- Hipótese 2: a completude dos casos de uso corretos gerados através da derivação i* -> casos de uso é superior a derivação BPMN -> casos de uso ou vice e versa. Este aspecto é mensurado pela completude das descrições textuais dos casos de uso.

- Hipótese nula: a utilização de alguma das duas técnicas assim como da ferramenta de apoio não influenciou o resultado final, sendo que não há diferença entre as abordagens em relação à quantidade de casos de uso, atores e estereótipos corretos gerados no diagrama de Casos de Uso bem como em relação à completude das descrições textuais geradas.

A medida a ser utilizada para análise é a diferença da quantidade de casos de uso corretos, parcialmente corretos e a diferença entre a completude entre os casos de uso obtidos pelas duas técnicas que foram coletadas em média pelos diferentes grupos a serem descritos na seção correspondente.

### 4.2.3 Seleção de Variáveis

As variáveis independentes são as técnicas e a ferramenta que serão utilizadas no processo de obtenção de Casos de Uso e as variáveis dependentes são as quantidades de Casos de Uso errados, parcialmente corretos e corretos obtidos pelos dois grupos.

### 4.2.4 Seleção de Sujeitos

Os sujeitos selecionados são estudantes do curso de Ciência da Computação e matriculados na disciplina de Processo de Engenharia de Software I no ano de 2019.

### 4.2.5 Design do Experimento

Grupos de controle:

A ferramenta JGOOSE/E4J será a variável a ser controlada, levando em consideração o suporte para o processo de derivação dos casos de uso a partir de modelos i* e BPMN. Os sujeitos foram divididos em dois grupos de controle com dois problemas diferentes a serem modelados, foi utilizado o método *crossover* para se obter uma melhor representação dos dois grupos. Primeiramente o Grupo 1 realizou a derivação do problema 1 utilizando modelos organizacionais i* enquanto o grupo 2 utilizou modelos de processo de negócio BPMN para modelar o mesmo problema. Após isso, o grupo 1 realizou a modelagem do problema 2 utilizando modelos de processo de negócio BPMN enquanto o grupo 2 utilizou modelos organizacionais i*, ou seja, cada sujeito experimental aplica todos os tratamentos, mas diferentes sujeitos aplicam tratamentos em uma ordem diferente. Em um *design crossover*, os grupos experimentais apresentam-se respeitando uma ordem em que os sujeitos aplicam tratamentos.

Os tempos em que cada tratamento é aplicado são conhecidos como períodos. Por exemplo, na Tabela 4.1 existem duas seqüências de aplicação de tratamento: AB e BA. Além disso, também existem dois períodos: período 1, onde os sujeitos na seqüência AB aplicam o tratamento A e os indivíduos na sequência BA aplicam o tratamento B; período 2, onde os sujeitos da sequência AB aplicam o tratamento B e os sujeitos na sequência BA aplicam o tratamento A.

Apesar das vantagens do tratamento *crossover*, é preciso tomar cuidado diante de tal experimento, para que os resultados não sejam impactados por problemas até mesmo invalidando o experimento. O *carryover* é um efeito que pode ocorrer quando se há sequência de tratamento onde o conhecimento prévio adquirido anteriormente pode ser utilizado de forma a influenciar um resultado de um tratamento que possa fazer de uma sequência. Consequentemente, os tratamentos administrados por último podem parecer mais eficazes do que aqueles administrados primeiro se o primeiro tratamento estiver aumentando a eficácia do segundo, ou menos eficazes

se o primeiro tratamento está diminuindo a eficácia do segundo. O efeito carryover pode ter um grande impacto sobre o experimento e até invalidar os resultados finais de uma experiência.

Desta forma, visando reduzir os efeitos negativos no nosso experimento usando o método crossover, serão consideradas algumas boas práticas conforme proposto em [50].

• Defina períodos. Decida quantas vezes os sujeitos vão repetir a tarefa experimental e estudar as implicações.

• Definir seqüências. Especifique as ordens em que os tratamentos devem ser aplicados.

• Lide com a transição no tempo de design. Selecione a estratégia para ser usada para contabilizar o carryover.

• Leve em consideração a variabilidade do assunto. Os dados escolhidos e a técnica de análise devem ser capazes de explicar medidas dependentes (medidas repetidas no mesmo assunto).

• Lide com o carryover no tempo de análise. A maneira em que o carryover é contabilizado na análise deve coincidir com a decisão de design sobre o carryover.

Os *designs de crossover* também têm grandes vantagens, tais como exigir menos sujeitos ou reduzir a variabilidade devido às diferenças entre os sujeitos.

Tabela 4.3: Grupos de controle

| | **Problema 1 (Período 1)** | **Problema 2 (Período 2)** |
|---|---|---|
| **Grupo 1** Sequência AB (13 participantes) | (**A**) Ferramenta modeladora de modelos organizacionais i* (E4J i*) e ferramenta geradora de casos de uso (Módulo do JGOOSE para derivar casos de uso) | (**B**) Ferramenta modeladora de modelos de processo de negócio BPMN (E4J BPMN) e ferramenta geradora de casos de uso (Módulo do JGOOSE para derivar casos de uso) |
| **Grupo 2** Sequência BA (11 participantes) | (**B**) Ferramenta modeladora de modelos de processo de negócio BPMN (E4J BPMN) e ferramenta geradora de casos de uso (Módulo do JGOOSE para derivar casos de uso) | (**A**) Ferramenta modeladora de modelos organizacionais i* (E4J i*) e ferramenta geradora de casos de uso (Módulo do JGOOSE para derivar casos de uso) |

Como parte que compõe o design do experimento, tem-se algumas características definidas como segue:

**Aleatoriedade:** Os sujeitos foram divididos em grupo de dois ou três participantes conforme disponibilidade de voluntários em sala, sendo que a divisão dos grupos assim como a alocação do problema será feita de maneira aleatória.

**Bloqueio:** No momento que os sujeitos são avaliados em grupos ao contrário de individualmente é obtida uma maneira de diminuir a influência de fatores que possam contribuir para a experiência prévia de cada sujeito, contrariando expectativas para o resultado do experimento.

**Balanceamento:** Como os sujeitos são todos estudantes do mesmo curso e da mesma disciplina, por conta de falta de recursos para a pesquisa se torna impossível obter variação para balancear os dados obtidos.

A princípio, o tempo reservado para a derivação dos casos de uso duração foi de 210 minutos (3 horas e 30 minutos), onde foram destinados 105 minutos para cada período, porém, os números foram menores, como serão mostrados na seção 4.4. O pesquisador apenas respondeu dúvidas frequentes referentes ao material fornecido não influenciando o processo de nenhuma outra maneira. Previamente à execução do experimento foram ministradas aulas aos participantes relativas aos dois modelos (i* e BPMN) e sobre a linguagem UML. As aulas foram ministradas pelo docente da disciplina.

O experimento foi realizado em um dos laboratórios no bloco do curso de Ciência da Computação. Para execução do experimento a ferramenta auxiliar no processo JGOOSE que inclui o editor E4J (i* e BPMN) foi apresentada aos possíveis participantes pelo docente da disciplina e por pesquisas anteriores que envolviam BPMN, considerando assim que os participantes tem familiaridade com a mesma, mas como houve dúvidas, o pesquisador foi responsável por saná-las.

### 4.2.6 Instrumentação

Segue a descrição dos instrumentos a serem utilizados durante a execução do experimento, os quais serão fornecidos a cada grupo. Cabe ressaltar que os itens 1, 2, 3 podem ser consultados nos Apêndices A, B e C:

1 - Questionários de abertura dos dois projetos conforme proposto por em [20] (Ver Apêndice A);

2 - Glossários de termos próprios e técnicos detalhando alguns dos termos utilizados nas

outras documentações (Ver Apêndice B).

3 - Template de descrição textual de casos de uso baseado em [19] sendo que os grupos terão de entregar um para cada caso de uso.

4 - 1 Computador para cada dupla ou trio (dependendo do número de integrantes).

5 - Ferramenta JGOOSE e editores E4J i*, Use Cases e BPMN.

### 4.2.7 Avaliação de Validade

**Validade Interna:** Pelo número de estudantes (ver tabela 4.1) a participar considera-se que a validade interna será bem atendida dentro de seu escopo, no entanto a falta de variedade nos sujeitos pode causar grande variação em condições diferentes.

**Validade Externa:** Considerando o nível no que diz respeito a experiência dos sujeitos, o resultado obtido no experimento será representativo de desenvolvedores de software com pouca experiência e treinamento em Processo de Engenharia de Software, na utilização de modelos i* e BPMN e em Casos de Uso em geral.

**Validade de Conclusão:** Como os dados obtidos serão quantitativos e qualitativos, a conclusão foi dificultada o que torna como principal ameaça para sua validade o processo de quantificação em si. Para se diminuir o risco, será necessário o fornecimento por parte dos sujeitos da descrição textual de cada Caso de Uso de maneira a estes serem analisados de maneira individual pelo pesquisador, facilitando a sua avaliação e qualificação.

**Validade de Construção:** Como ameaça principal temos que as medidas adotadas podem não ser representativas de corretude e completude. Por esse motivo, foi dividido o resultado final esperado em três parâmetros para a corretude, sendo estes parâmetros partes do todo para se obter uma medida de eficiência o mais próximo possível da realidade, e para a completude foram avaliados os casos de uso entre si, comparando item a item do template proposto por Cockburn [19], dos resultados obtidos entre os dois métodos utilizados (i* vs. BPMN) para mensurar a completude dos casos de uso resultantes.

De modo geral, o experimento pode sofrer alguns tipos de limitações para esse modo de planejamento. O escopo se torna estreito, o que pode dificultar a generalização para outros contextos no âmbito de desenvolvimento de software. No entanto, a obtenção de dados quantitativos possibilita a obtenção de dados que fornecem uma fácil visualização entre os

diferentes tratamentos fornecendo um indício do real estado da ferramenta avaliada bem como uma ponte para realizar experimentos em contextos industriais. Também como é um experimento facilmente reproduzível torna-se possível sua ampliação futura.

### 4.2.8 Uso da abordagem GQM para apoiar a medição dos resultados obtidos no experimento

• **Questões e métricas:**

O uso da GQM [16][47] pode auxiliar a nortear os elementos a serem medidos no experimento definindo questões e métricas as quais são definidas para atender os objetivos da pesquisa. Desta forma, no presente experimento, as seguintes questões e métricas serão utilizadas.

**Q1** Qual é a quantidade de casos de uso corretos gerados pela abordagem i*-> casos de uso?

**M1** Quantidade de casos de uso corretos gerados.

**Q2** Qual é a quantidade de casos de uso corretos gerados pela abordagem BPMN-> casos de uso?

**M2** Quantidade de casos de uso corretos gerados.

**Q3** O uso dos estereótipos do tipo include e extend está correto pela abordagem i*-> casos de uso?

**M3** O uso dos estereótipos do tipo include e extend está correto.

**Q4** O uso dos estereótipos do tipo include e extend está correto pela abordagem BPMN-> casos de uso?

**M4** O uso dos estereótipos do tipo include e extend está correto.

**Q5** Há diferenças entre o diagrama de casos de uso gerado pelo E4J Use Cases i* e o diagrama de casos de uso E4J Use Cases BPMN?

**M5** Houve diferenças entre o diagrama de casos de uso gerado pelo E4J Use Cases i* e o diagrama de casos de uso E4J Use Cases BPMN.

**Q6** Esses modelos i* e BPMN construídos para um mesmo problema da organização levam aos mesmos casos de uso?

**M6** Os modelos i* e BPMN construídos para um mesmo problema da organização não levam aos mesmos casos de uso.

**Q7** Qual é a quantidade de casos de uso parcialmente corretos gerados pela abordagem i*->

casos de uso?

**M7** Quantidade de casos de uso parcialmente corretos gerados.

**Q8** Qual é a quantidade de casos de uso parcialmente corretos gerados pela abordagem BPMN-> casos de uso?

**M8** Quantidade de casos de uso parcialmente corretos gerados.

**Q9** Qual é a quantidade de elementos da descrição textual de casos de uso preenchidos corretamente para cada casos de uso gerado pela abordagem BPMN?

**M9** Quantidade de casos de uso considerando a completude dos mesmos.

**Q10** Qual é a quantidade de elementos da descrição textual de casos de uso preenchidos corretamente para cada casos de uso gerado pela abordagem i*

**M10** Quantidade de casos de uso considerando a completude dos mesmos.

## 4.3 Operação

Segundo Wohlin [18], a operação do experimento é dividida em três etapas distintas, são elas:

- Preparação: fase inicial do experimento que consiste no preparo do local do experimento e recepção dos sujeitos;

- Execução: conduzir o experimento de acordo com o que foi planejado;

- Validação dos dados: tomada de ações que garantam que a informação coletada seja representativa do experimento realizado.

O quase-experimento foi realizado no dia 17 de outubro de 2019. O quase-experimento foi iniciado às 8h30m e teve seu término às 11h15m, sendo que nesse intervalo entre os horários os dois problemas [Apêndice A] foram modelados, conforme os grupos finalizavam o primeiro problema no período 1 já eram orientados a começar a modelagem do segundo problema no período 2, contando apenas com 10 minutos de intervalo entre um problema e outro. Todos os alunos tinham um conhecimento prévio das técnicas (i* e BPMN) decorrente da disciplina de Processo de Engenharia de Software I.

Os alunos foram divididos em 1 grupo de 13 participantes e 1 grupo de 11 participantes, totalizando 24 participantes. Desses 2 grupos foram feitos 11 subgrupos, onde o design *crossover* referente a Tabela 4.1 foi aplicado, contando com 6 subgrupos modelando o problema 1 em i* e

5 subgrupos modelando o problema 1 em BPMN, em outro período, os 6 subgrupos modelaram o problema 2 em BPMN e os 5 subgrupos modelaram o problema 2 em i*.

Antes do início do quase-experimento foram distribuídos questionários com algumas questões referentes aos problemas que foram modelados, ao conhecimento dos alunos em relação as técnicas i* e BPMN, ao uso da ferramenta JGOOSE e também com um campo para contabilizar o horário de início e término da modelagem dos problemas para ambas as técnicas [Apêndice C]. Também foram distribuídos os problemas [Apêndice A] para a leitura dos participantes onde nesse momento antes da real obtenção dos requisitos pela descrição, os participantes puderam esclarecer dúvidas referentes aos problemas e a ferramenta JGOOSE.

Após o início do quase-experimento os participantes foram instruídos a não discutir entre eles os documentos referentes aos problemas a serem modelados. Foi realizada essa recomendação devido a evitar troca de informações.

## 4.4 Análise e Interpretação

A análise dos casos de uso obtidos no quase experimento foram baseadas conforme o design do experimento, a corretude foi avaliada conforme descrito na seção 4.2.2, onde os 3 aspectos a serem avaliados são:

Primeiro: Existência de Caso de Uso correspondente ao esperado ao caso base já modelado pelo pesquisador, sendo que para verificação da existência ou não da correspondência será utilizada a descrição textual do Caso de Uso. Assim, se o Caso de Uso descoberto quando comparado ao esperado representar uma funcionalidade similar, este será considerado correto.

Segundo: Relação ao ator correto. Na análise de parâmetro, o Caso de Uso obtido deverá estar relacionado ao mesmo ator que o esperado conforme obtido no primeiro parâmetro. Cabe destacar que, caso o Caso de Uso obtido não satisfaça o primeiro parâmetro é impossível que satisfaça o segundo.

Terceiro: Relação aos demais Casos de Uso. Será avaliada a presença de relação entre o Caso de Uso obtido e demais Casos de Uso, sendo que a avaliação terá como foco averiguar a presença de relações do tipo <include> e <extend> de acordo com o esperado. Observando novamente que o segundo parâmetro foi avaliado somente para os casos onde o primeiro parâmetro foi satisfeito, da mesma maneira ocorre para o terceiro parâmetro, onde só foi avaliado

para os casos onde o primeiro e o segundo parâmetro foram validados.

Outro detalhe que vale a pena ser ressaltado é a existência de um modelo base de ambas as técnicas (i* e BPMN) dos problemas feito pelo orientando deste trabalho de conclusão de curso para o começo da avaliação dos dados, já que os problemas propostos também foram modelados pelo próprio orientando. Através dessa análise obteve-se os seguintes dados: No total, foram gerados 211 casos de uso considerando as duas técnicas (BPMN e i*). Para a modelagem BPMN foram gerados 147 casos de uso, dos quais 2 foram corretos, 66 parcialmente corretos e 79 incorretos. Considerando os aspectos avaliados, a técnica BPMN gerou 24 casos de uso parcialmente corretos que não satisfazem o segundo aspecto, que diz respeito a relação ao ator correto e 42 casos de uso parcialmente corretos que não satisfazem o terceiro aspecto, que é a relação do tipo <include> e <extend>. Para a modelagem i* foram gerados 64 casos de uso, dos quais 1 foi correto, 51 parcialmente corretos e 12 incorretos. Considerando os aspectos a serem avaliados, a técnica i* gerou 6 casos de uso parcialmente corretos que não satisfazem o segundo aspecto, que tem como objetivo avaliar a relação ao ator correto, também gerou 45 casos de uso que não satisfazem o terceiro aspecto, que avalia a presença de relações do tipo <include> e <extend>. Os gráficos dos dados obtidos são apresentados a seguir para uma melhor visualização dos resultados.

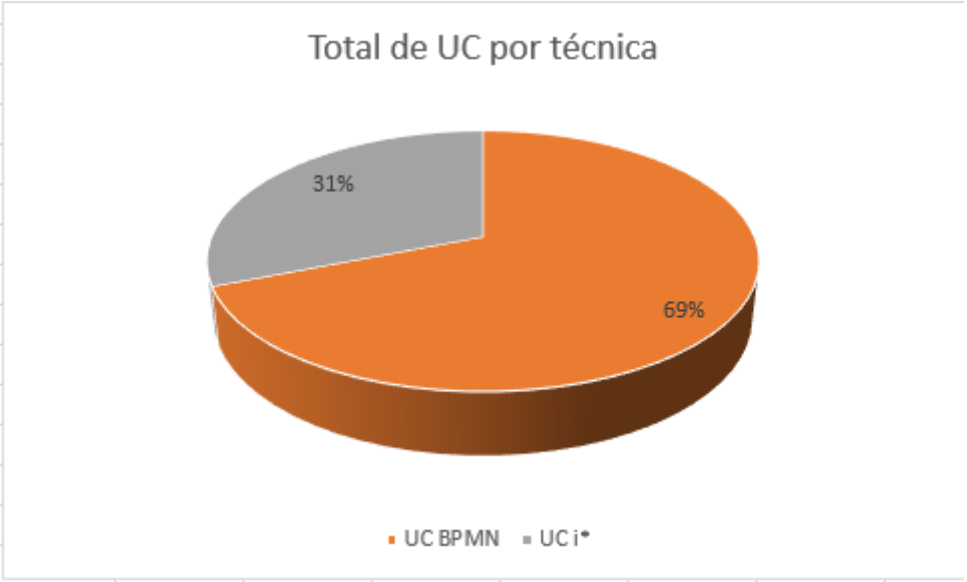



Figura 4.1: Número de Casos de Uso obtidos pelas duas técnicas.

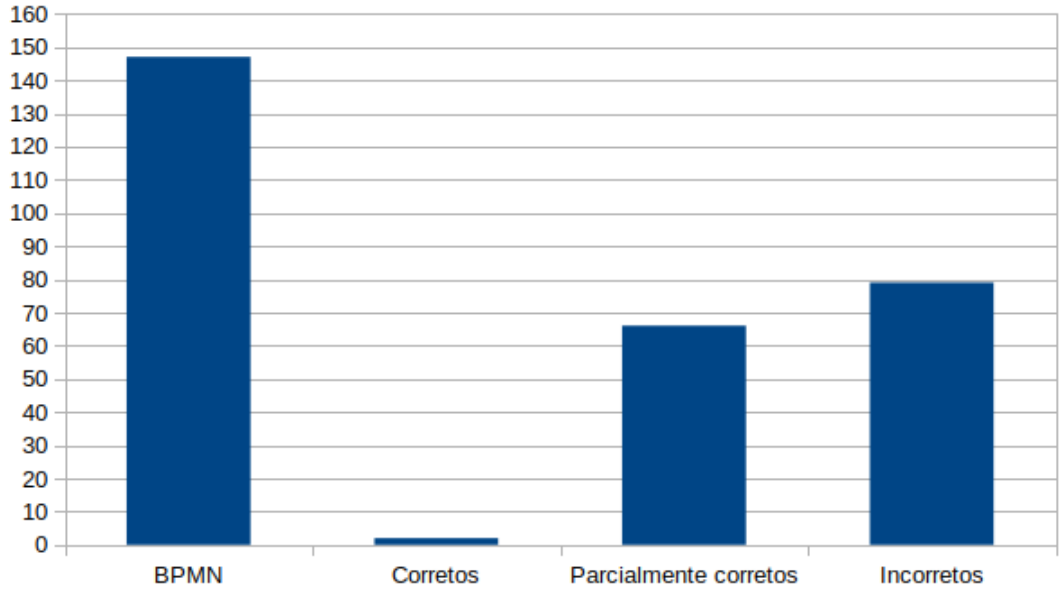



Figura 4.2: Número de Casos de Uso obtidos pela técnica BPMN.

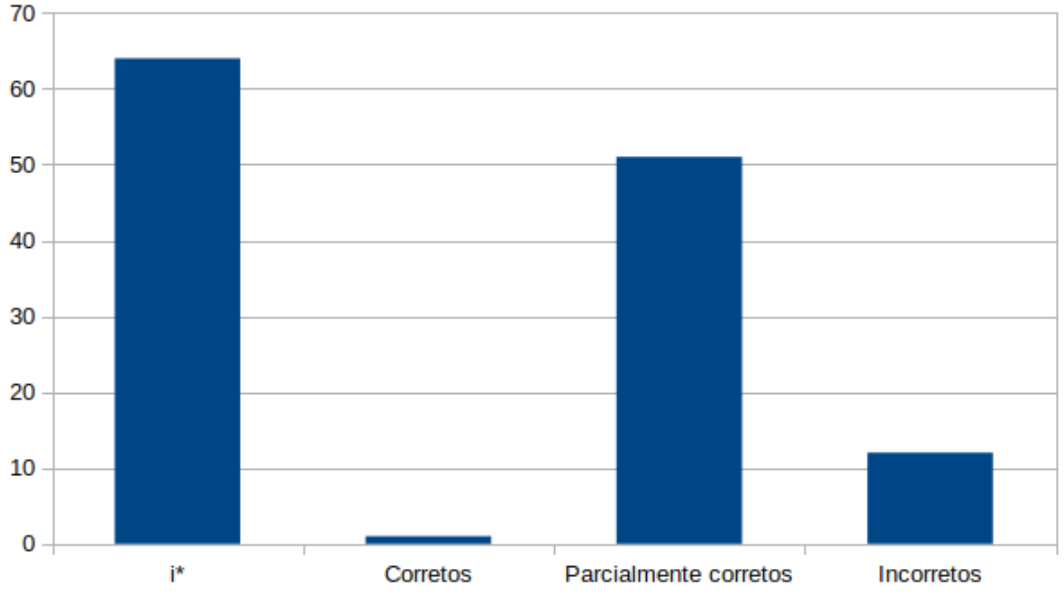



Figura 4.3: Número de Casos de Uso obtidos pela técnica i*.

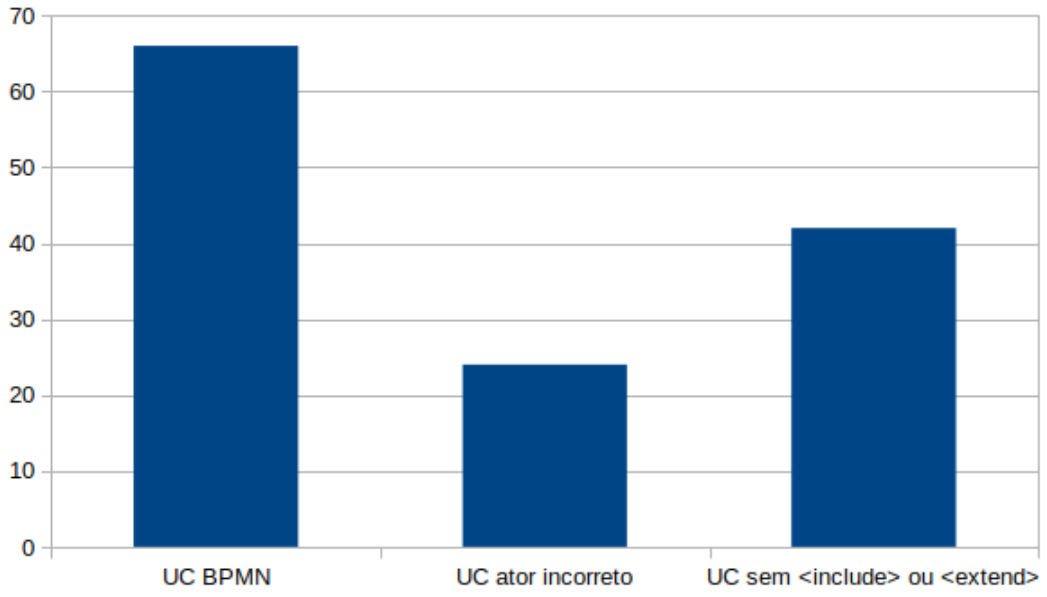



Figura 4.4: Casos de Uso BPMN parcialmente corretos por aspecto avaliado.

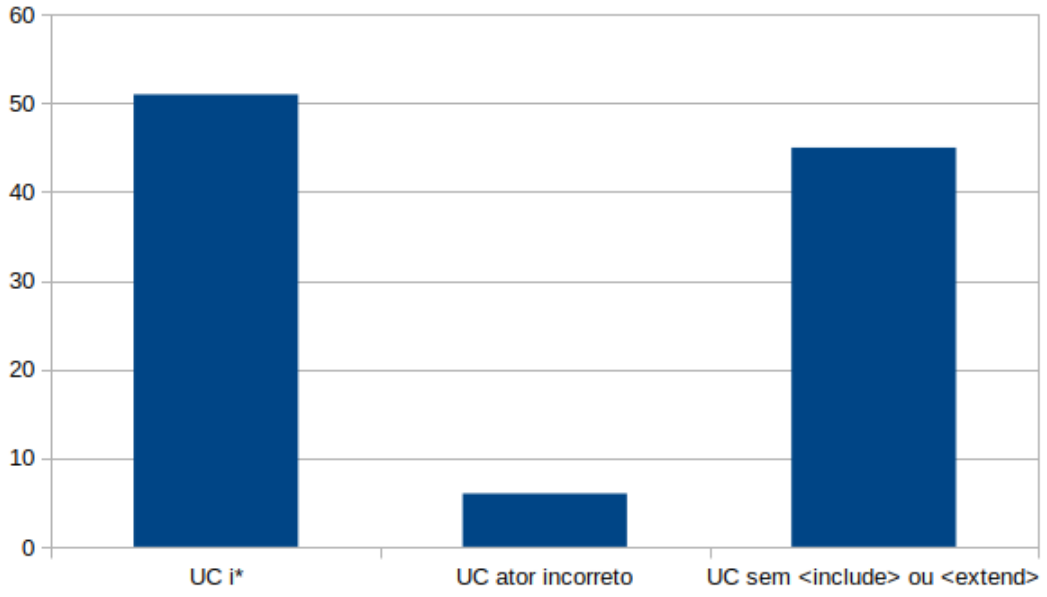



Figura 4.5: Casos de Uso i* parcialmente corretos por aspecto avaliado.

Foram feitos gráficos para os dois problemas que foram modelados, considerando a quantidade de casos de uso totais, casos de uso corretos, parcialmente corretos e incorretos. Considerando o problema SiStagios [Apêndice A], os números obtidos para a técnica BPMN foram os seguintes: 80 casos de uso no total, 1 correto, 32 parcialmente corretos e 47 incorretos. A Tabela 4.2 mostra estes dados por equipe tanto no 1º quanto no 2º período. Para a técnica i* foram

obtidos os seguintes números: 42 casos de uso no total, 0 corretos, 34 parcialmente corretos, 32 incorretos. A Tabela 4.4 mostra estes dados por equipe tanto no 1º quanto no 2º período. A Figura 4.7 resume essas informações.

Para o problema Coneu [Apêndice A] considerando a técnica BPMN, foram obtidos os seguintes resultados: 67 casos de uso no total, 1 correto, 34 parcialmente corretos e 32 incorretos. A Tabela 4.3 mostra estes dados por equipe tanto no 1o quanto no 2o período. Em relação a técnica i*, foram obtidos os seguintes números: 22 casos de uso no total, 1 correto, 17 parcialmente corretos, 4 incorreto. A Tabela 4.5 mostra estes dados por equipe tanto no 1o quanto no 2o período. A Figura 4.8 resume essas informações.

| Grupo | Correto | Parcialmente correto | Incorreto |
|---|---|---|---|
| Grupo 1 | 0 | 2 | 0 |
| Grupo 2 | 1 | 3 | 4 |
| Grupo 3 | 0 | 7 | 14 |
| Grupo 4 | 0 | 20 | 29 |
| Grupo 8 | 0 | 0 | 0 |

Tabela 4.4: UC BPMN SiStagios.

| Grupo | Correto | Parcialmente correto | Incorreto |
|---|---|---|---|
| Grupo 5 | 1 | 5 | 4 |
| Grupo 6 | 0 | 8 | 3 |
| Grupo 7 | 0 | 3 | 1 |
| Grupo 9 | 0 | 9 | 7 |
| Grupo 10 | 0 | 2 | 5 |
| Grupo 11 | 0 | 7 | 12 |

Tabela 4.5: UC BPMN Coneu.

| Grupo | Correto | Parcialmente correto | Incorreto |
|---|---|---|---|
| Grupo 5 | 0 | 5 | 2 |
| Grupo 6 | 0 | 1 | 0 |
| Grupo 7 | 0 | 0 | 5 |
| Grupo 9 | 0 | 6 | 1 |
| Grupo 10 | 0 | 20 | 0 |
| Grupo 11 | 0 | 2 | 0 |

Tabela 4.6: UC i* SiStagios.

| Grupo | Correto | Parcialmente correto | Incorreto |
|---|---|---|---|
| Grupo 1 | 1 | 1 | 0 |
| Grupo 2 | 0 | 2 | 0 |
| Grupo 3 | 0 | 5 | 2 |
| Grupo 4 | 0 | 8 | 2 |
| Grupo 8 | 0 | 1 | 0 |

Tabela 4.7: UC i* Coneu.

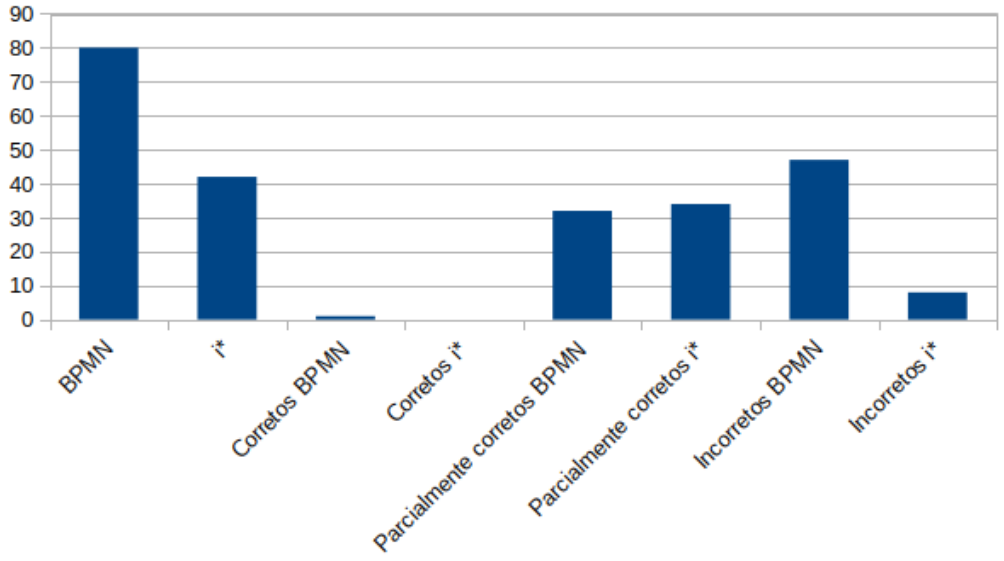



Figura 4.6: Número de Casos de Uso obtidos no problema SiStagios.

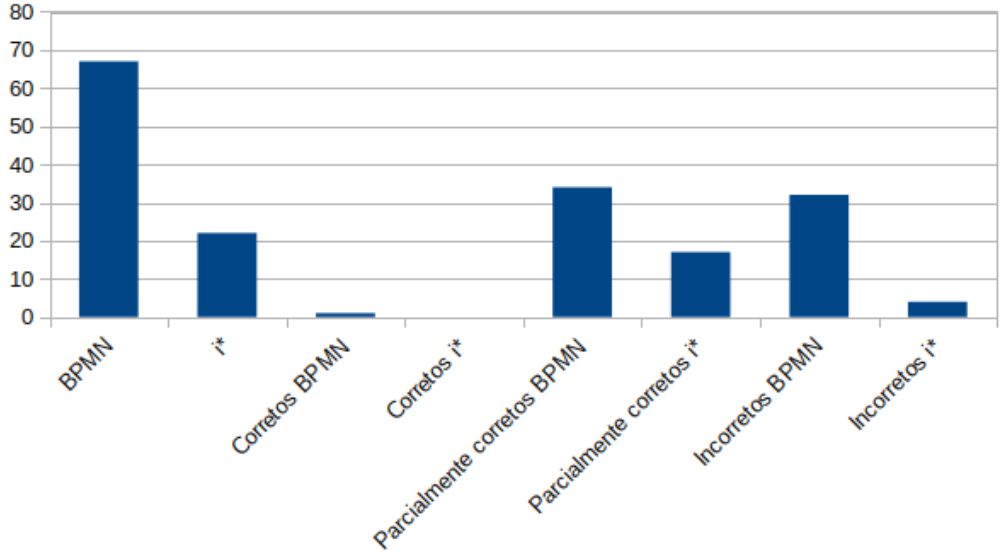



Figura 4.7: Número de Casos de Uso obtidos no problema Coneu.

No que diz respeito à completude, foram analisados os templates dos casos de uso obtidos para cada um dos problemas propostos com as respectivas técnicas de modelagem (i* e BPMN). O template a ser considerado é descrito na seção 4.2.2, que foi avaliado comparando os templates dos casos de uso obtidos dos dois problemas para ambas as técnicas, sendo assim, os templates dos casos de uso do problema 1 referentes a técnica BPMN foram comparados com os templates dos casos de uso da técnica i* e o mesmo processo foi feito para o problema 2. Para efetivamente obter um resultado, cada caso de uso correspondente a técnica BPMN, foi referenciado um caso de uso equivalente na técnica i*, os demais casos de uso que não tinham correspondências entre as técnicas foram avaliados de forma isolada, dessa forma, não interferindo no conjunto de casos de uso equivalentes.

No total obteve-se 6 conjuntos para os problemas 1 e 2 [Apêndice A] com casos de uso equivalentes, no total, 41 casos de uso foram equivalentes, do total, 26 do problema SiStagios sendo 12 casos de uso BPMN e 14 casos de uso i*, no problema Coneu foram gerados 15 casos de uso no total, sendo 9 da técnica BPMN e 6 da técnica i*. Cada conjunto não teve o mesmo número de casos de uso, por exemplo, no conjunto equivalente 4 para o problema SiStagios a técnica BPMN contempla 3 casos de uso e a técnica i* contempla 4 casos de uso, mas todos com o mesmo objetivo. Foram feitos gráficos onde a primeira e segunda coluna representam o total de casos de uso por técnica em determinado conjunto equivalente, a terceira e quarta coluna representam a quantidade de elementos inseridos no template por cada técnica, esses números são baseados na quantidade dos casos de uso por cada uma das técnicas nesse conjunto equivalente, ou seja, o total de elementos no template é baseado no total de casos de uso do determinado conjunto. Os números foram inseridos nos gráficos 4.9, 4.10, 4.11, 4.12, 4.13, e 4.14 a seguir:

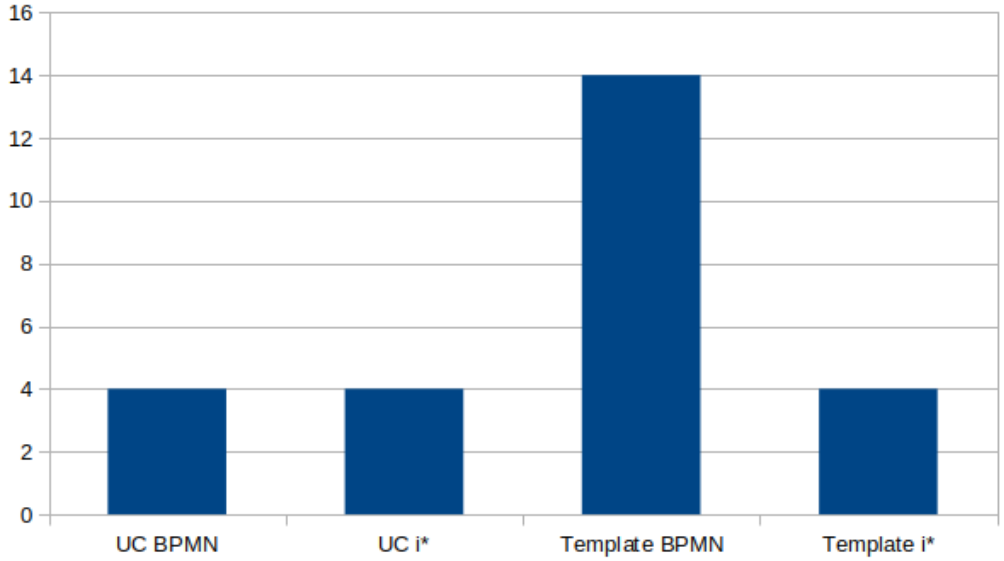



Figura 4.8: Conjunto equivalente 1 SiStagios - Número de elementos no template por técnica.

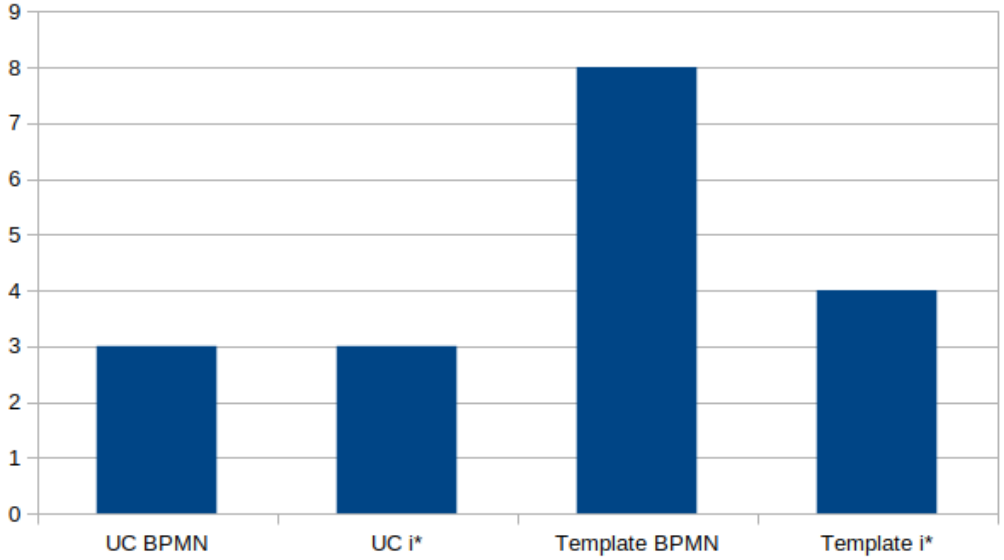



Figura 4.9: Conjunto equivalente 2 SiStagios - Número de elementos no template por técnica.

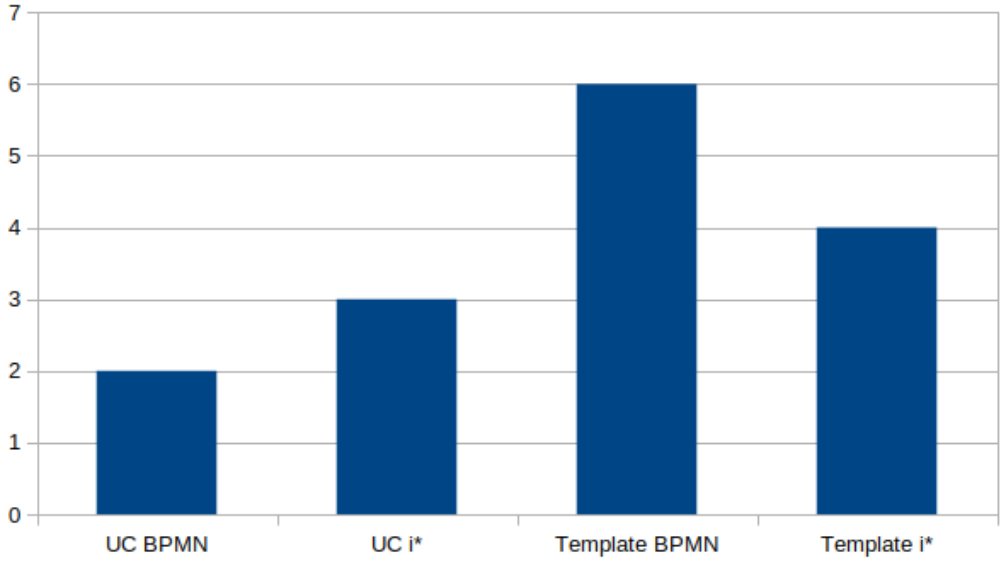



Figura 4.10: Conjunto equivalente 3 SiStagios - Número de elementos no template por técnica.

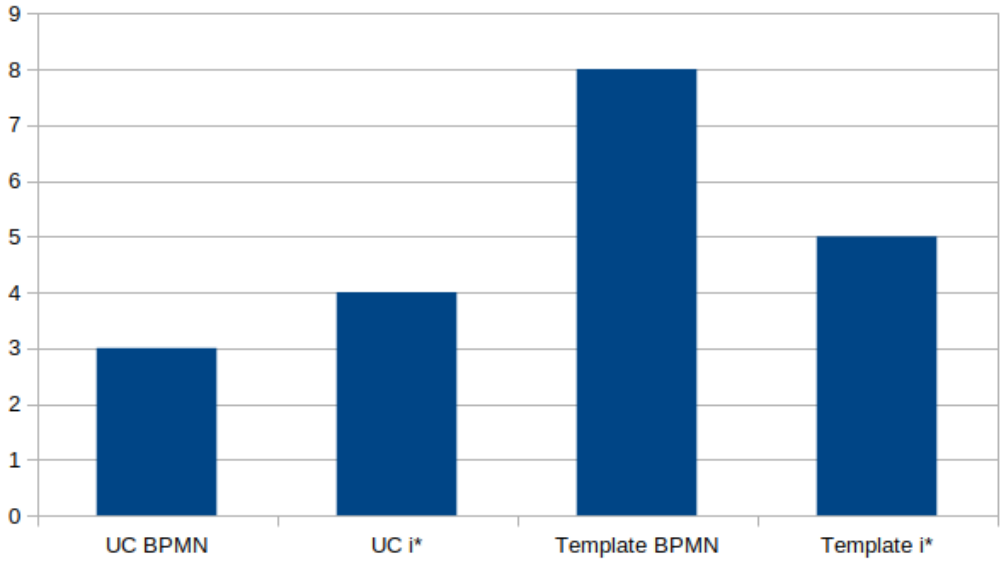



Figura 4.11: Conjunto equivalente 4 SiStagios - Número de elementos no template por técnica.

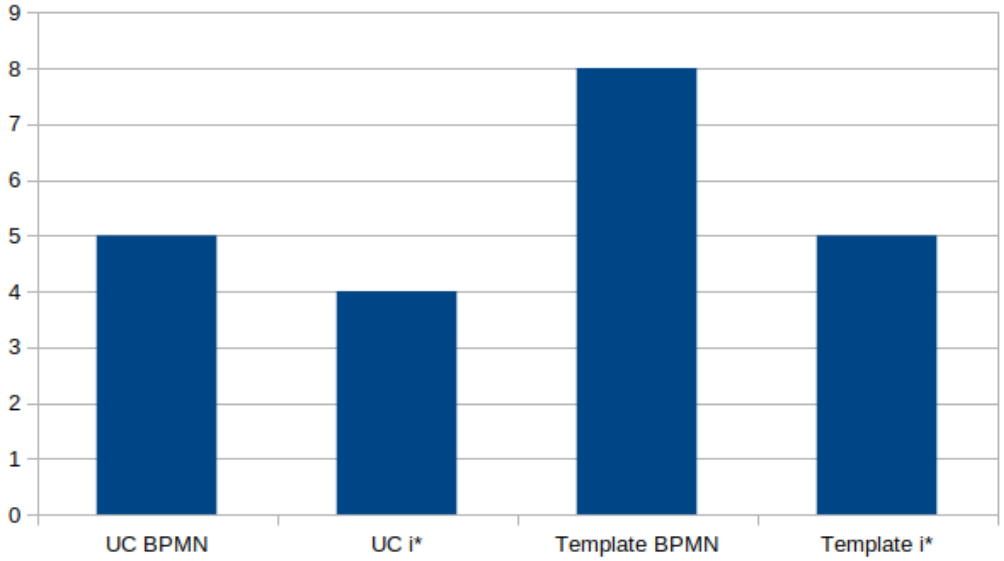



Figura 4.12: Conjunto equivalente 1 Coneu - Número de elementos no template por técnica.

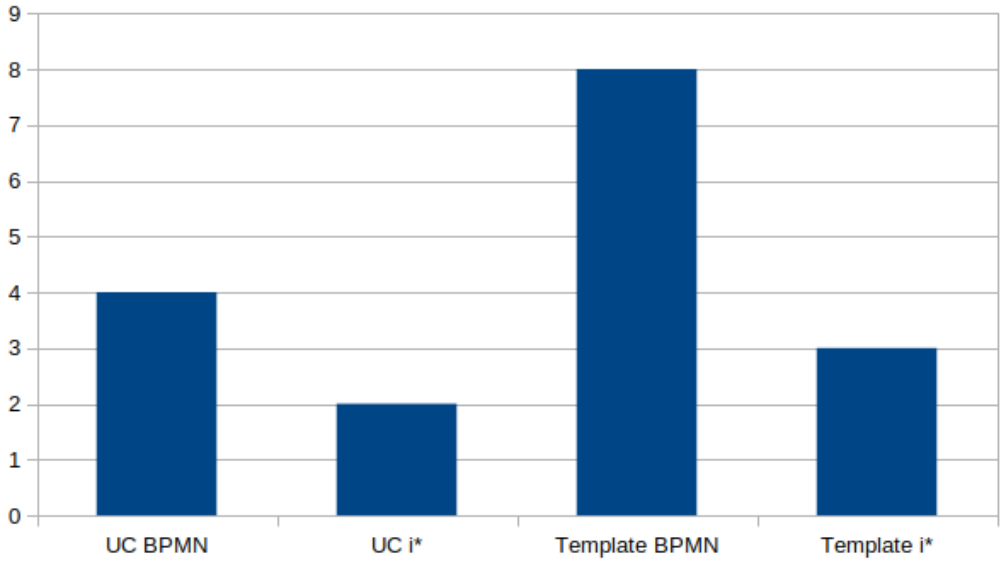



Figura 4.13: Conjunto equivalente 2 Coneu - Número de elementos no template por técnica.

Para os casos de uso que não tiveram correspondências entre as técnicas e foram avaliados de forma isolada ao demais casos de uso que tiveram correspondências, os números foram de 79 casos de uso no total, sendo 41 deles pertencentes ao problema SiStagios e 38 pertencentes ao problema Coneu. No problema SiStagios o número de casos de uso correspondentes à técnica BPMN foi de 21, contendo 42 elementos preenchidos no template, enquanto para a técnica i* foram 20 casos de uso com 20 elementos no template. Os números são apresentados nos

gráficos a seguir:

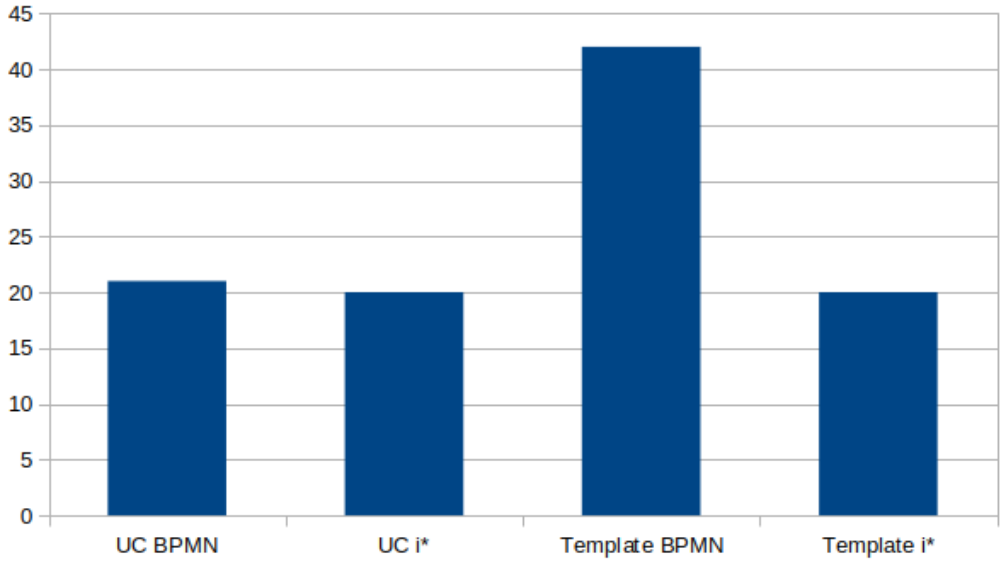



Figura 4.14: Problema SiStagios - Casos de Uso sem correspondência por técnica.

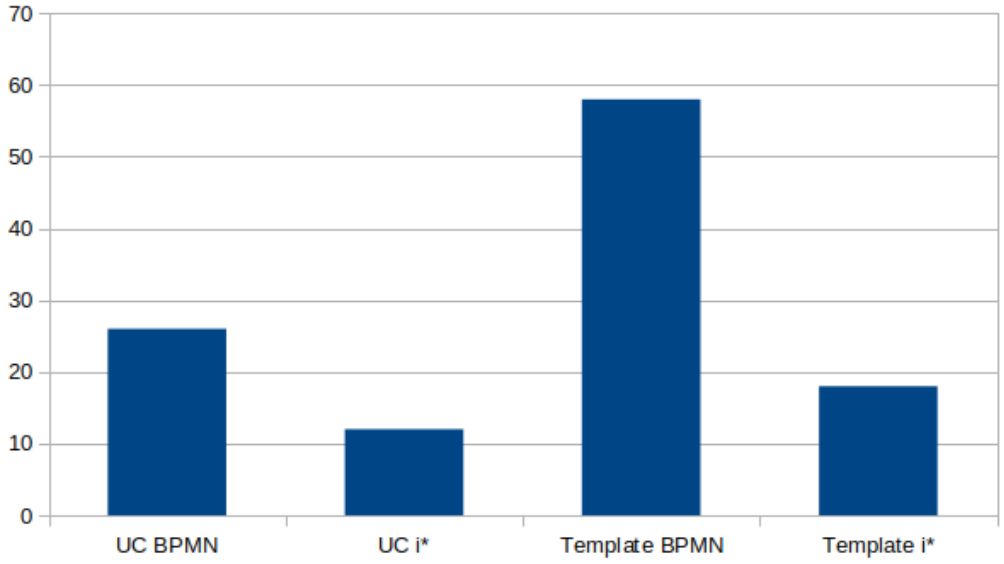



Figura 4.15: Problema Coneu - Casos de Uso sem correspondência por técnica.

## 4.5 Apresentação dos Resultados

Após feitas as duas análise de corretude e completude mencionadas na seção 4.4, pode-se responder as questões e métricas mencionadas na seção 4.2.8 da abordagem GQM [16][47].

**Q1** Qual é a quantidade de casos de uso corretos gerados pela abordagem i*-> casos de uso?

1.

**Q2** Qual é a quantidade de casos de uso corretos gerados pela abordagem BPMN-> casos de uso?

2.

**Q3** O uso dos estereótipos do tipo include e extend está correto pela abordagem i*-> casos de uso?

Foi gerado 1 estereótipo e está correto.

**Q4** O uso dos estereótipos do tipo include e extend está correto pela abordagem BPMN-> casos de uso?

Foi gerado 1 estereótipo e está correto.

**Q5** Há diferenças entre o diagrama de casos de uso gerado pelo E4J Use Cases i* e o diagrama de casos de uso E4J Use Cases BPMN?

Houve diferenças entre os diagrama de casos de uso gerados, foram gerados 20 diagramas de caso de uso BPMN e 11 diagramas de caso de uso i*. Os diagramas BPMN apresentam menos concordância que os diagramas i*, já que gerou atores que são incorretos, na maioria dos casos apresentou atores do tipo "sistema"que se refere ao problema em questão ou apresentou atores referenciando o próprio requisito do sistema, em alguns casos referenciou relações entre atores corretos, porém, os diagramas i* que foram 11 no total, apresentaram relações corretas entre atores que também estavam corretos, apenas em 1 caso de uso i* as relações estavam incorretas sem exceções. Todos os diagramas podem ser visualizados em [77].

**Q6** Esses modelos i* e BPMN construídos para um mesmo problema da organização levam aos mesmos casos de uso?

Foram poucos os casos onde os modelos i* e BPMN construídos para um mesmo problema levaram aos mesmos casos de uso, pode-se notar isso nos números apresentados anteriormente, onde esse número de casos de uso equivalentes foi menor do que os de casos de uso sem correspondência entre as técnicas, por outro lado, existiu alguns casos de uso onde ele se repetiu para a mesma técnica, porém, como não tinha correspondência da outra técnica foi considerado como não equivalente.

**Q7** Qual é a quantidade de casos de uso parcialmente corretos gerados pela abordagem i*->

casos de uso?

51.

**Q8** Qual é a quantidade de casos de uso parcialmente corretos gerados pela abordagem BPMN-> casos de uso?

66.

**Q9** Qual é a quantidade de elementos da descrição textual de casos de uso preenchidos corretamente para cada caso de uso gerado pela abordagem BPMN?

Os elementos que compõem a descrição textual dos casos de uso preenchidos corretamente são os seguintes: Preconditions, Success End Condition, Trigger, Primary actor, Main Scenario. Esses elementos estão presentes em apenas 3 descrições textuais BPMN, a parcela maior contempla 3 elementos, onde são eles: Preconditions, Success End Condition, Primary actor. A parcela intermediária de elementos apenas contém o elemento Primary actor.

**Q10** Qual é a quantidade de elementos da descrição textual de casos de uso preenchidos corretamente para cada caso de uso gerado pela abordagem i*?

Os elementos que compõem a descrição textual dos casos de uso preenchidos corretamente são os seguintes: Preconditions, Primary actor, Main scenario. Esses elementos estão presentes em apenas 1 descrição textual i*, a parcela maior contempla 1 elemento, que são eles: Primary actor. A parcela intermediária contempla os elementos Primary actor, Main scenario.

Vale também ressaltar detalhes sobre o questionário aplicado para os participantes, a primeira e segunda pergunta se referiam à experiência dos participantes diante das técnicas utilizadas para modelar os problemas. Todas as respostas dos 11 grupos foram direcionadas apenas a experiência adquirida durante a disciplina de Processo de Engenharia de Software I. A terceira e quarta pergunta se referiam a clareza e objetividade dos problemas, dos 11 grupos 10 deles responderam que o problema SiStagios estava claro e objetivo considerando o primeiro contato com o cliente e 1 respondeu que estava ambíguo. No que diz respeito ao problema Coneu, dos 11 grupos 8 responderam que o problema estava claro e objetivo quando considerado ao primeiro contato com o cliente, 3 grupos responderam que o problema Coneu não tinha tantos detalhes quando comparado ao SiStagios.

A quinta pergunta do questionário se referia a utilização da ferramenta JGOOSE, se ela foi de fácil utilização ou apresentou algum problema. As respostas foram um pouco divergentes

pois também foi solicitado o que poderia ser melhorado. O grupo 1 respondeu que a JGOOSE não foi de fácil utilização, a interface poderia ser mais intuitiva, principalmente quanto à relação entre tarefas e objetivos. O grupo 2 respondeu que a ferramenta executa todas as funções necessárias para a elaboração dos diagramas, porém ela necessita que o usuário realize uma série de ações repetitivas e que o sistema poderia fazer sozinho, como é o caso de programas similares como Bizagi, dentre tais ações pode-se citar criação de raias e piscinas, conexão das setas de relacionamento e disposição espacial das setas. O grupo 3 respondeu que também não foi de fácil utilização, apresentando alguns bugs e falta de atalhos no teclado. O grupo 4 respondeu que também não foi de fácil utilização, a ferramenta possui vários bugs que dificultam a usabilidade da mesma, principalmente nas ligações das relações entre tarefas e atores. O grupo 5 respondeu que a ferramenta JGOOSE não gerou os casos de uso do sistema Coneu (BPMN), além da dificuldade do texto não seguir a movimentação das ligações. O grupo 6 respondeu que o detalhe foi a exceção de usabilidade do programa, como falta de atalhos de teclado e lenta manipulação de componentes.

O grupo 7 respondeu que não foi de fácil utilização, não tinha teclas de atalho, que também encontraram problemas ao gerar os casos de uso do modelo BPMN onde além de não gerar, também a apagou a lane superior do modelo projetado. O grupo 8 respondeu que a ferramenta apresenta alguns problemas de funcionalidade, por exemplo o delete para remover um objeto da tela, foi comentado a falta de algumas teclas de atalho. O grupo 9 respondeu que poderia haver atalhos de teclado para melhor utilização, algumas ferramentas foram difíceis de serem encontradas no software. O grupo 10 respondeu que é fácil de utilizar, porém tem muitos bugs e isso é o maior problema. Por exemplo, quando mudaram o zoom para detalhar melhor o que fizeram, todas as ligações/setas saíram do lugar e isso foi o que mais atrasou o desenvolvimento do modelo. O grupo 11 respondeu que a ferramenta funcionou corretamente.

Levando em consideração as questões e métricas respondidas através dos resultados obtidos, pode-se comparar esses resultados com as hipóteses utilizadas para o quase-experimento.

Considerando a hipótese 1 e as questões GQM respondidas anteriormente, a utilização de modelos de processo de negócio BPMN juntamente da ferramenta E4J (BPMN) para derivação de Casos de Uso influenciou o resultado final, sendo que a quantidade de casos de uso corretos bem como estereótipos e atores encontrados por quem utilizou modelos de processo de negócio

BPMN juntamente da ferramenta E4J (BPMN) é maior à quantidade de casos de uso corretos encontrados por quem utilizou i* juntamente da ferramenta E4J (i*).

Considerando a hipótese 2 e as questões GQM respondidas anteriomente, a completude dos casos de uso gerados através da derivação BPMN -> casos de uso é superior a derivação i* -> casos de uso.

Considerando a hipótese nula, ela pode ser refutada, pois a utilização das duas técnicas assim como da ferramenta influenciaram o resultado final, os fatores relacionados a essa influência são citados na seção 4.6.

## 4.6 Conclusão

Pode-se perceber que a técnica BPMN obteve uma vantagem nos números na derivação dos casos de uso utilizando a ferramenta JGOOSE, porém, pontos importantes devem ser destacados, por exemplo, observando os gráficos das figuras 4.2, 4.3 e 4.4 pode-se ver que a técnica BPMN conseguiu derivar um número maior de casos de uso corretos e parcialmente corretos, porém, com mais casos de uso incorretos que no caso eram passos ou tarefas intermediárias para chegar efetivamente no caso de uso esperado. Por outro lado, a técnica i* derivou um número menor de casos de uso corretos e parcialmente corretos mas com um número menor de casos de uso incorretos. Neste caso, é importante ressaltar a importância das diretrizes que são utilizadas no processo de derivação tanto em BPMN como em i*. A técnica BPMN apresentou casos de uso incorretos que eram passos intermediários entre o caso de uso esperado, como a modelagem BPMN é baseada em passos necessários para se realizar um determinado processo, isso apresenta a possível causa de existirem casos de uso incorretos sendo passos intermediários do efetivo caso de uso, que é o amadurecimento das diretrizes utilizadas no processo de derivação da técnica BPMN. Um exemplo de um dos resultados em que isso ocorre é mostrado a seguir:

Use Case: 1 - Solicita dados do aluno
CHARACTERISTIC INFORMATION
Goal in Context:
Scope:

Preconditions:

Success End Condition:

Failed End Condition:

Trigger: O evento Cadastrar aluno ocorreu.

Primary actor: Coordenadora da central

Use Case: 2 - Informa dados

CHARACTERISTIC INFORMATION

Goal in Context:

Scope:

Preconditions: Atividade Solicita dados do aluno deve ser concluída;

Success End Condition:

Failed End Condition:

Trigger:

Primary actor: Aluno

Use Case: 3 - Insere dados no sistema

CHARACTERISTIC INFORMATION

Goal in Context:

Scope:

Preconditions: Atividade Informa dados deve ser concluída;

Success End Condition:

Failed End Condition:

Trigger:

Primary actor: Coordenadora da central

Onde neste caso apresentado o caso de uso correto seria somente o caso de uso 3 - Insere dados no sistema. As diretrizes utilizadas no processo de derivação da técnica i* se mostram

mais sólidas neste sentido, os números mostram a quantidade de casos de uso incorretos obtidos por i*, possívelmente pelo modelo ter um viés de metas apresentando uma visão mais abrangente do processo, ressaltando que em BPMN é utilizado uma descrição passo a passo do processo no modelo.

Os casos de uso que foram analisados separadamente foram os casos de uso que tiveram presença em BPMN e não tiveram em presença em i*, ou, por outro lado, estiveram presentes em i* e não estiveram presentes em BPMN, um exemplo é apresentado a seguir:

BPMN:
Use Case: 1 - Acessa o sistema (fazer login)
CHARACTERISTIC INFORMATION
Goal in Context:
Scope:
Preconditions:
Success End Condition:
Failed End Condition:
Trigger: O evento Start Event ocorreu.
Primary actor: Gerenciador de estoque.
MAIN SUCCESS SCENARIO

i*:
Use Case: Gerar Nota Fiscal
CHARACTERISTIC INFORMATION
Goal in Context:
Scope:
Preconditions:
Success End Condition:
Failed End Condition:
Primary Actor: Empresa

MAIN SUCCESS SCENARIO

Essas informações mostram que as técnicas acabam obtendo casos de uso diferentes e válidos, pois nos casos onde o caso de uso aparece em i* e não em BPMN, ou, ao contrário, são casos de uso esperados e considerados parcialmente corretos, que fazem parte dos processos dos problemas, até mesmo se complementando em alguns casos. Outro detalhe importante foi o conhecimento relatado pelos participantes no questionário aplicado do começo ao fim do quase-experimento, em todos os casos a experiência dos participantes sobre as técnicas foi somente a adquirida em sala de aula, portanto, o conhecimento dos participantes é o básico. Nas tabelas 4.2, 4.3, 4.4 e 4.5 pode-se visualizar que o número de casos de uso obtidos por cada grupo apresenta isso de forma mais detalhada, considerando a experiência e o questionário relatado na seção anterior após as perguntas GQM, onde apenas foram gerados 3 casos de uso corretos considerando as duas técnicas (BPMN e i*). Como relatado no questionário também se pode perceber que o problema SiStagios gerou mais casos de uso que o problema Coneu, conforme os grupos relataram que o problema SiStagios estava mais claro que o problema Coneu. Também foi marcado o tempo de execução dos problemas considerando a técnica utilizada, conforme relatado pelos participantes o tempo para realização do modelo em BPMN foi maior do que em i*, levando uma média de 65 minutos, enquanto o i* levou uma média de 43 minutos. Os números são apresentados na tabela a seguir:

| Grupo | Tempo BPMN | Tempo i* |
|---|---|---|
| Grupo 1 | 60 minutos | 60 minutos |
| Grupo 2 | 45 minutos | 25 minutos |
| Grupo 3 | 80 minutos | 25 minutos |
| Grupo 4 | 90 minutos | 20 minutos |
| Grupo 5 | 75 minutos | 70 minutos |
| Grupo 6 | 75 minutos | 46 minutos |
| Grupo 7 | 90 minutos | 72 minutos |
| Grupo 8 | 30 minutos | 24 minutos |
| Grupo 9 | 45 minutos | 39 minutos |
| Grupo 10 | 55 minutos | 42 minutos |
| Grupo 11 | 70 minutos | 50 minutos |

Tabela 4.8: Tempo de execução das técnicas.

Como relatado na Seção 4.5, seguindo os questionários respondidos pelos participantes os dois problemas estavam bem especificados considerando o primeiro contato com o cliente, esse primeiro contato com o cliente representa uma conversa informal entre as partes interessadas no primeiro encontro. Em alguns casos, foi relatado que o problema SiStagios estava mais claro e objetivo.

No que diz respeito à ferramenta JGOOSE, o maior problema segundo os participantes foi a falta de atalho de algumas funcionalidades e alguns bugs que ocorreram na hora de gerar os casos de uso, em BPMN na hora da geração alguns elementos sumiam como as lanes da pool e tarefas com ligações. Em i* as tarefas sumiam na hora da geração dos casos de uso. Desta forma, foi realizado um refinamento manual para que pudesse gerar os casos de uso conforme os modelos que os participantes construíram. Em praticamente todos os casos que precisaram de refinamento em BPMN o problema foi na ausência de uma lane na hora da geração ou o problema foi modelado sem uma pool, dessa forma não gerando os casos de uso. Nos outros casos de uso BPMN que foram refinados o problema estava um vários modelos no mesmo arquivo (mais de uma pool no mesmo arquivo). No caso da técnica i* a maioria dos modelos que não geraram casos de uso foi pelo motivo dos participantes não terem feito um modelo SR que englobava as funcionalidades do problema. Nos outros modelos i* que não geraram foi pelo motivo de bug na ferramenta, onde na hora da geração sumiam as ligações entre tarefas. Lembrando um ponto muito importante, que os modelos que foram refinados manualmente não sofreram nenhuma alteração na forma como foram feitos, todos foram mantidos exatamente como os participantes fizeram, somente esses detalhes mencionados foram refeitos.

A partir desses resultados e informações não é possível dizer que uma técnica ou processo de derivação se sobressaiu a outra, a técnica BPMN gerou mais casos de uso corretos e parcialmente corretos mas também acabou gerando mais casos de uso incorretos. A técnica i* gerou um número de casos de uso corretos e parcialmente corretos menor, porém a eficácia da técnica se mostra na quantidade de casos de uso incorretos, que foi menor que a técnica BPMN, mostrando solidez em suas diretrizes e alertando para uma possível revisão das diretrizes BPMN. A experiência dos participantes também foi um fator relevante, pois a maioria dos casos de uso que foram considerados parcialmente corretos, só foram considerados nessa faixa de corretude pois não apresentaram informações do tipo <include> ou <extend>, o que mostra

o conhecimento básico na utilização e modelagem das técnicas i* e BPMN dos participantes no quase-experimento. Esse trabalho também mostra a importância de uma atualização na ferramenta JGOOSE, onde alguns bugs na hora da geração dos casos de uso aconteceram e foram mencionados pelos participantes. Deve-se enfatizar a importância das duas técnicas na modelagem e obtenção de casos de uso, neste contexto, não é correto dizer que uma técnica é melhor que a outra, e sim, que cada uma tem suas particularidades e sua relevância, podendo serem utilizadas de maneira correta a determinado problema conforme a necessidade do Engenheiro de Requisitos.

## 4.7 Considerações finais

Neste capítulo foram apresentadas as etapas para a realização do experimento controlado. Na seção 4.1 é apresentada a definição do escopo do experimento, onde é abordada a definição do objetivo e suas características, mostrando o objetivo geral do trabalho, também é apresentado o objeto de estudo onde as técnicas (i* e BPMN) são relacionadas, o propósito do experimento e sua perspectiva, o foco de qualidade no aspecto prático e o contexto onde esse trabalho é inserido e o resumo do escopo. Na seção 4.2 é apresentado o planejamento do experimento, esse planejamento descreve em detalhes o contexto no qual o experimento é feito, formulação das hipóteses, seleção de variáveis dependentes e independentes e sujeitos que participarão do experimento. Se descreve o design do experimento bem como instrumentação que serão usados. Após isso é mostrada elementos para a validade do experimento e outros fatores que podem contribuir para a confiabilidade dos resultados como o uso da abordagem GQM para apoiar a medição dos resultados obtidos no experimento. Nas seções, 4.3, 4.4 e 4.5 são apresentados outras fases do experimento que são, operação, análise e interpretação e apresentação dos resultados, na seção 4.6 é feita uma conclusão com base nos resultados obtidos.

# Apêndice A

# Problemas a serem modelados

**Problema 1: Desenvolvimento de um software desktop para a automação de uma loja de pneus (Gerenciamento de clientes, fornecedores, compras, vendas e produtos).**

**Introdução**

**R:** O presente projeto tem por finalidade o desenvolvimento de um software para automação de uma loja de comércio de pneus, sendo responsável pelo gerenciamento do estoque de produtos assim como parte do processo de venda.

**Informações sobre o cliente**

**R:** O cliente é uma empresa que vêm atuando na área do comércio de pneus. É uma revendedora de pneus, que hoje conta com 25 funcionários e recebe pedidos para compra de pneus novos, semi novos e recapados de diversas cidades do sul do Brasil.

**Nome do Produto**

R: Nome do produto a ser definido, nome temporário a fim de identificação será "Coneu".

**Detalhamento do Problema:**

**Objetivo do Projeto**

**R:** O sistema desktop a ser desenvolvido deverá ser capaz de realizar o controle de estoque dos produtos da empresa, sendo necessário possibilitar a sua manipulação pelo **Gerente de Estoque e Vendedor** assim como a atualização automática do estoque ao ser realizada uma compra por um cliente cadastrado. A compra de produtos dos **Fornecedores** por parte da **Empresa Cliente** faz parte do sistema a ser desenvolvido. Deve ser responsável também por parte do processo de vendas da empresa, não sendo incluso nisto o processo de pagamento (cartão, dinheiro, etc) que será realizado pelo **Cliente** durante seu cadastro.

**Breve descrição da necessidade**

**R:** O sistema representa uma mudança no modelo de negócio da **Empresa Cliente**, partindo de uma atuação localizada com vendas sendo feitas através de loja física. O objetivo final da **Empresa Cliente** é diminuir o tempo gasto necessário para consultas de produtos, clientes, fornecedores, compras e vendas, automatizando a maior parte do processo que a empresa realiza, dessa forma, agilizando os processos da mesma e armazenando os dados de forma consistente e segura.

**Funcionamento Atual**

R: Atualmente a **Empresa Cliente** conta com uma planilha eletrônica (Excel) para auxiliá-lo no controle de estoque, toda que vez que uma venda é realizada o **Vendedor** é encarregado a realizar a venda daquele determinado produto, igualmente para uma compra, mas neste caso a compra é realizada pelo **Gerente de Estoque**, ambas tarefas podem ser canceladas durante o processo. As vendas são cadastradas manualmente pelo **Vendedor** em um caderno de anotações assim como as compras realizadas pelo **Gerente de Estoque**. Em uma venda, o **Cliente** solicita o produto ao **Vendedor** e o mesmo verifica se este consta em estoque, se houver, o vendedor anota a venda constando a quantidade, produto, cliente e o valor unitário do produto em uma planilha do excel, se o cliente não está inserido no sistema o **Vendedor** pode realizar o cadastro desse **Cliente**. Em uma compra, o **Gerente de Estoque** verifica a existência do produto em estoque e solicita o produto ao **Fornecedor**, o **Gerente de Estoque** anota a compra constando a quantidade, produto, fornecedor e o preço unitário do produto em uma planilha do excel, se o fornecedor ou produto não está inserido no sistema o **Gerente de Estoque** pode realizar o cadastro desse Fornecedor ou produto. O cadastro de **Fornecedores** no presente momento é feito apenas com uma agenda de anotações, assim como o cadastro de **Cliente**. A **Empresa Cliente** deseja um software capaz de controlar o estoque da empresa e gerenciar as vendas realizadas pela mesma, bem como as compras feitas com o fornecedor e vendas feita para clientes.

**Dependências do Processo**

**R:** Depende do sistema da receita federal para legalização das vendas através de Nota Fiscal e, de acordo com os desejos da **Empresa Cliente**, os dados do sistema (base de dados) serão

armazenados em um banco de dados integrado ao sistema.

**Usuários chave envolvidos**

**R:** Ao chegar produtos novos na **Empresa Cliente** as notas fiscais serão encaminhadas ao setor administrativo da empresa onde o **Gerente de Estoque** irá adicioná-los no estoque usando o sistema Coneu assim como arquivará uma cópia da nota fiscal de compra equivalente no sistema. O **Gerente de Estoque** deve ser capaz de gerenciar produtos, fornecedores, e compras. O **Vendedor** deve ser capaz de gerenciar clientes e vendas. Os usuários mais importantes de acordo com a **Empresa Cliente** são: **Gerente de Estoque** e **Vendedor**.

**Dados a serem armazenados**

**R:** Informações de compra e venda devem ser gerenciadas, dados dos **Clientes Cadastrados** e **Fornecedores** também devem ser armazenados, assim como os dados referentes aos produtos e sua quantidade em estoque.

**Qual a frequência de utilização do sistema?**

**R:** Constante.

**Prioridade do cliente**

**R:** Foi destacado pelo cliente que ele espera que o sistema seja simples de operar causando o menor transtorno possível para usuários **Gerente de Estoque** e **Vendedor**, como a **Empresa Cliente** já está em funcionamento é de preferência do cliente que o produto apenas seja lançado após ter sua usabilidade validada junto a representante da **Empresa Cliente.**

---

**Problema 2: Desenvolvimento de uma aplicação web para gerenciar o processo de estágio de uma instituição (Gerenciamento de alunos, estágio, vagas, empresa e curso)**

**Introdução**

**R:** O presente projeto tem por finalidade o desenvolvimento de uma aplicação web, sendo responsável pelo gerenciamento de estágios dos alunos de uma instituição de forma a satisfazer

as necessidades dos interessados no projeto.

**Informações sobre o cliente**

**R:** O cliente é um centro educacional profissionalizante que conta com a parceria de várias empresas que atuam em diversas áreas. De uma forma geral, a **Empresa Cliente** é responsável pelo ensino técnico profissionalizante de estudantes com interesse em diversas áreas de atuação.

**Nome do Produto**

**R:** Nome do produto a ser definido, nome temporário a fim de identificação será "SiStagios".

**Detalhamento do Problema:**

**Objetivo do Projeto**

**R:** O sistema web a ser desenvolvido deverá ser capaz de gerenciar os alunos cadastrados e seu curso, bem como o estágio e as vagas oferecidas pela empresa sendo necessário possibilitar a sua manipulação pela **Coordenadora da Central**. O sistema proposto permitirá à **Coordenadora da Central** cadastrar, consultar, atualizar e excluir dados de alunos, empresas, cursos, vagas e estágios correntes, que hoje são criados manualmente. Também permitirá aos **Alunos** interessados consultarem vagas disponíveis para eventuais interesses.

**Breve descrição da necessidade**

**R:** O sistema representa uma melhora no modelo de negócio do centro educacional, abrindo a possibilidade de realizar tarefas para a organização do processo de estágios da **Empresa Cliente (Centro Educacional)**. Partindo de uma atuação localizada no ensino profissionalizante, o objetivo é a facilitação e flexibilização do processo burocrático de estágio, promovendo confiabilidade, agilidade, integridade e organização.

**Funcionamento Atual**

**R:** Atualmente a empresa tem um sistema que apresenta algumas falhas. Este sistema apresenta problemas de segurança, má organização e falta de documentação, sendo assim, necessário a substituição do mesmo. Hoje, a **Coordenadora da Central** utiliza o sistema existente para gerenciar alunos, empresas, vagas, estágios e curso. Quando uma empresa deseja ser

vinculada ao sistema ela deve entrar em contato com a **Coordenadora da Central**, assim cadastrando a empresa que quer ser parceira do Centro Educacional. O mesmo ocorre para quando uma nova vaga de estágio é aberta na empresa parceira, ela deve entrar em contato com a **Coordenadora da Central** para que essa vaga seja vinculada a empresa. A partir do momento que o **Aluno** for cadastrado no sistema pela **Coordenadora da Central**, seu curso também é vinculado e ele poderá consultar as vagas disponibilizadas pelas empresas parceiras. Dessa forma, o **Aluno** de determinado curso poderá consultar as vagas que são disponibilizadas pela empresa parceira. Esse processo é realizado pela **Coordenadora da Central**.

**Dependências do Processo**

**R:** Depende do sistema das empresas para a disponibilização de vagas conforme a demanda das mesmas, a partir da informação das empresas no contato com o centro educacional as vagas podem ser disponibilizadas entre os alunos.

**Usuários chave envolvidos**

**R:** Praticamente todo o processo é realizado pela **Coordenadora da Central**, pois ela é quem cadastra os alunos, empresas, vagas, estágios e cursos. Ao chegar novas vagas da empresa parceira no centro educacional, a **Coordenadora da Central** cadastra as novas vagas no sistema, podendo editar, excluir e consultar, assim como se alguma empresa nova quiser ser parceira do centro educacional, poderá ser incluída no sistema pela **Coordenadora da Central**, da mesma forma ocorre com alunos novos e o curso do aluno. Um **Aluno** vinculado ao sistema poderá consultar as vagas de estágio abertas conforme seu curso.

**Dados a serem armazenados**

**R:** Informações de vagas e estágios, curso de Alunos, bem como dados de alunos e empresas devem ser gerenciados.

**Qual a frequência de utilização do sistema?**

**R:** Constante.

**Prioridade do cliente**

**R:** Foi destacado pela **Empresa Cliente** que ele espera que o sistema seja simples de operar causando o menor transtorno possível para usuários **Coordenadora da Central** e **Aluno**, como a **Empresa Cliente** já está em funcionamento é de preferência do cliente implantar o sistema somente quando estiver sido desenvolvido por completo após realizado os testes necessários.

# Apêndice B

# Glossário de termos próprios e técnicos

### Glossário de Termos Próprios

**Empresa Cliente:** A empresa vendedora de livros que requisitou o desenvolvimento do sistema.

**Cliente Visitante:** usuário utilizando o sistema sem ter sido cadastrado.

**Cliente Cadastrado:** cliente que passou pelo processo de cadastro automatizado podendo realizar compras.

**Gerente:** funcionário da Empresa Cliente que trabalha na administração deste, sendo parte de seu processo de tomada de decisão.

**Gerente de Estoque:** funcionário da Empresa Cliente responsável pelo estoque da empresa, sendo responsável por cadastrar novos itens (livros) no estoque e, se necessário, modificar itens já existentes.

Somente haverá um possível Gerente de Estoque cadastrado.

**Fornecedores:** empresa que fornecem os livros a Empresa Cliente.

### Glossário de termos técnicos

**API (Application Programming Interface):** Um API nada mais é do que um software desenvolvido para atuar como mediador entre um software pai e os desenvolvedores que gostariam de utilizar determinada funcionalidade deste software pai em seu próprio software. É, portanto o papel do API agir como mensageiro entre os dois softwares.

**Cloud (Cloud Computing)**: É o fornecimento de serviços relacionados a computadores, tais como armazenamento e processamento, através da internet. Serviços esses fornecidos por um provedor mediante pagamento.

**Notas Fiscais:** A lei brasileira define vários tipos de documentação que servem como modo de legalização perante o governo, a que interessa neste ambiente é a Nota Fiscal de Consumidor Eletrônica. Este documento não é impresso, existindo apenas digitalmente. Sua geração é de responsabilidade da Empresa que fez a venda, assim como seu envio para o órgão governamental competente.

# Apêndice C

# Questionário aplicado

**Grupo:**

**Metodologia i***

Tempo de início:

Tempo de término:

**Metodologia BPMN**

Tempo de início:

Tempo de término:

1) Qual seu conhecimento na utilização e modelagem da técnica i*? (Tempo de experiência, projetos, etc)

2) Qual seu conhecimento na utilização e modelagem da técnica BPMN? (Tempo de experiência, projetos, etc)

3) O modelo do problema (SiStágios) estava claro e objetivo no que diz respeito à entrevista no primeiro contato com o cliente proprietário? Caso a resposta seja não, na sua opinião, o que poderia ser melhorado?

4) O modelo do problema (Coneu) estava claro e objetivo no que diz respeito à entrevista no primeiro contato com o cliente proprietário? Caso a resposta seja não, na sua opinião, o que poderia ser melhorado?

5) A ferramenta JGOOSE foi de fácil utilização ou apresentou algum problema? Se sim, na sua opinião, o que poderia ser melhorado?

6) Na sua opinião, qual das técnicas foi mais fácil de se modelar o problema 1 e o problema 2? Por quê?

7) Qual sua opinião diante dos casos de uso gerados tanto pela técnica i* como BPMN?

# Referências Bibliográficas


[1] THAYER, R. H., and M. Dorfman, Software Requirements Engineering, 2d ed., IEEE Computer Society Press, 1997.

[2] BOEHM, B. Software Engineering Economics (Englewood Cliffs, NJ: Prentice Hall, 1981).

[3] LAPOUCHNIAN, A. Goal-Oriented Requirements Engineering: An Overview of the Current Research. Toronto. 2005.

[4] SOMMERVILLE, I. Engenharia de Software. 2011.

[5] LAMSWEERDE, A. Goal-Oriented Requirements Engineering: A Guided Tour, 5th IEEE International Symposium on Requirements Engineering, Toronto, Canada, 2001.

[6] YU, E. S. K. Towards modelling and reasoning support for early-phase requirements engineering. In: 3RD IEEE INTERNATIONAL SYMPOSIUM ON REQUIREMENTS ENGINEERING - RE97, 1997. Washington D.C., USA: [s.n.], (1997). p.226–235.

[7] CRUZ, E. F. et al. Advances in enterprise engineering viii: 4th enterprise engineering working conference, eewc 2014, funchal, madeira island, Portugal, may 5-8, 2014. proceedings.In: . [S.l.]: Springer International Publishing, 2014. cap. From Business Process Models to Use Case Models: A Systematic Approach, p. 167–181.

[8] OMG. Business Process Model and Notation (BPMN) Specification. 2009. v.2.0 (online). Disponível em: http://www.bpmn.org. Acesso em: 26 de março de 2018.

[9] CHINOSIA, M.; TROMBETTA, A. BPMN: An introduction to the standard. Computer Standards and Interfaces, v. 34, p. 124-134, 2012.

[10] SHISHKOV, B.; XIE, Z.; LIU, K.; DIETZ, J. L, "Using norm analysis to derive use cases from business processes," in Proc. 5th Workshop On Organiz. Semiotics, 2002.

[11] UML. Disponível em: <http://www.uml.org/>. Consultado na Internet em: 26/03/2018.

[12] BOOCH, G.; RUMBAUGH, J. JACOBSON, I. UML- Guia de Usuário. Vol. 2. Rio de Janeiro: Campus, 1999.

[13] SANTANDER, Victor Francisco Araya; CASTRO, J. F. B. Deriving Use Cases from Organizational Modeling. In: IEEE Joint International Requirements Engineering Conference - RE 02, 2002, Essen. IEEE Joint International Requirements Engineering Conference, RE 02, University of Essen, Germany, September, 9-13., 2002.

[14] BRISCHKE, M. ; SANTANDER, VICTOR F. A ; SILVA, I. F. . Melhorando a Ferramenta JGOOSE. In: 15th Workshop on Requirements Engineering, 2012, Buenos Aires, 24 a 27 de Abril. Anais do 15th Workshop on Requirements Engineering, 2012.

[15] MERLIN, L. P. ; SILVA, A. L. B. ; SANTANDER, V. F. A. ; SILVA, I. F. ; CASTRO, J. Integrating the E4J editor to the JGOOSE tool. In: XVIII Workshop on Requirements Engineering, 2015, Lima. XVIII Ibero-American Conference on Software Engineering. Lima: UCSP, 2015. p. 646-659.

[16] PELISER, D. E4J Use Cases: um editor de diagrama de casos de uso integrado à ferramenta JGOOSE. Cascavel - PR: [s.n.], Dezembro 2014.

[17] GIROTTO, Alysson N.; Derivando Casos de Uso a partir de Modelos BPMN, Trabalho de Conclusão de Curso – Universidade Estadual do Oeste do Paraná, Cascavel – PR, 2016.

[18] WOHLIN, C; RUNESON, P; OHLSSON, M, C; WESSLÉN, A; HOST, M; REGNELL, B. Experimentation in Software Engineering. New York – 2012.

[19] COCKBURN, A. Writing Effective Use Cases. Boston, MA, USA: Addison-Wesley Longman Publishing Co., Inc., 2000.

[20] LES. Laboratório de Engenharia de Software. Consultado na internet em 29/03/2019. https://www.dropbox.com/s/4gkd7bjxst70z28/DocumentosProblemasQuase_Experimento Cibse2019.rar?dl=0

[21] SOUZA, C.; SANTANDER, V. Uma Proposta de Elicitação e Análise de Requisitos no Contexto de Médias e Pequenas Empresas de Desenvolvimento de Software. 14th Workshop On Requirements Engineering. Rio de Janeiro, Brasil, 2011.

[22] ALENCAR, F. M. R. Mapeando a Modelagem Organizacional em especificações precisas. 1999. p. 304 Tese (Doutorado) - Centro de Informática, Universidade Federal de Pernambuco, Recife, 1999.

[23] YU, E. et al. Social Modeling for Requirements Engineering. Mit Press, 2011. (Cooperative Information Systems).

[24] DAVENPORT, T. Reengenharia de Processos: Como Inovar na Empresa Através da Tecnologia da Informação. 5. ed. Rio de Janeiro: Campus, 1994.

[25] KO, R. K. L. A computer scientist's introductory guide to business process management (bpm). Crossroads, ACM, New York, NY, USA, v. 15, n. 4, p. 4:11–4:18, jun. 2009. ISSN 1528-4972.

[26] BANDARA, W. et al. Major issues in business process management: An expert perspective. In: ÖSTERLE, H.; SCHELP, J.; WINTER, R. (Ed.). ECIS. [S.l.]: University of St. Gallen, 2007. p. 1240–1251.

[27] SMITH, H.; FINGAR, P. Business Process Management: The Third Wave. [S.l.]: Meghan-Kiffer Press, 2003.

[28] ABPMN. Guia para o Gerenciamento de Processes de Negócio Corpo Comum de Conhecimento. [S.l.], 2009.

[29] ERIKSSON, H.-E.; PENKER, M. Business Modeling With UML: Business Patterns at Work. 1st. ed. New York, NY, USA: John Wiley & Sons, Inc., 1998.

[30] FRANCO, C. R. da R. Um Catálogo De Boas Práticas, Erros Sintáticos E Semânticos Em Modelos BPMN. Dissertação (Monografia) — Universidade Federal de Pernambuco, Recife, Março 2014.

[31] CRUZ, E. F.; MACHADO, R. J.; SANTOS, M. Y. Bridging the gap between a set of interrelated business process models and software models. In: ICEIS 2015 - Proceedings of the 17th International Conference on Enterprise Information Systems, Volume 2, Barcelona, Spain, April, 2015. [S.l.: s.n.], 2015. p. 338–345.

[32] BPMN. Business Process Model and Notation (BPMN) Version 2.0.2. [S.l.], December 2013.

[33] OMG. Object Management Group. 2016. Consultado na INTERNET: http://www.omg.org, 2018.

[34] GALO, J. Comparativo Entre as Versões 1.2 e 2.0 da Notação BPMN e Sua Aplicação em Diagrama de Processos de Negócios. Dissertação (Monografia) — Universidade Tecnológica Federal do Paraná, Medianeira, 2012.

[35] JACOBSON, I. Object-oriented development in an industrial environment. In: Conference Proceedings on Object-oriented Programming Systems, Languages and Applications. New

York, NY, USA: ACM, 1987. (OOPSLA '87), p. 183–191.

[36] BOOCH, G.; RUMBAUGH, J.; JACOBSON, I. Unified Modeling Language User Guide, The (2Nd Edition) (Addison-Wesley Object Technology Series). [S.l.]: Addison-Wesley Professional, 2005. ISBN 0321267974.

[37] TIWARI, S.; GUPTA, A. A systematic literature review of use case specifications research. Inf. Softw. Technol., Newton, MA, USA, v. 67, n. C, p. 128–158, 2015.

[38] ALDER, G. Design and implementation of the jgraph swing component. Techinical Report, v. 1, n. 6, 2002.

[39] PELISER, D. et al. Integrating the e4j use cases to the jgoose tool. In: Proceedings of the 35th International Conference of the Chilean Computer Science Society. [S.l.: s.n.], 2016.

[40] PELISER, D.; SANTANDER, V. F. A.; MERLIM, L. Jgoose: Melhorias realizadas. V EPAC - Encontro Paranaense de Computação, Cascavel - PR, Setembro 2013.

[41] BRISCHKE, M. Desenvolvimento de uma ferramenta para integrar modelagem organizacional e modelagem funcional na engenharia de requisitos. Cascavel, 2005.

[42] VICENTE, A. A. JGOOSE: Uma ferramenta de Engenharia de Requisitos para Integração da Modelagem Organizacional i* com a Modelagem Funcional de Casos de Uso UML. Dissertação (Monografia) — Universidade Estadual do Oeste do Paraná, Cascavel, 2006.

[43] BRISCHKE, M. Melhorando a Ferramenta JGOOSE. Dissertação (Monografia) — Universidade Estadual do Oeste do Paraná, Cascavel, 2012.

[44] IEEE: IEEE standard glossary of software engineering terminology. Technical Report, IEEE Std 610.12-1990, IEEE (1990).

[45] SHAW, M. What Makes Good Research in Software Engineering. International Journal of Software Tools for Technology Transfer, vol 4. 2002.

[46] TRAVASSOS, G, H; GUROV, D; AMARAL, E, A, G. Introdução a Engenharia de Software Experimental. Relatório Técnico RT-ES-590/02. Rio de Janeiro – 2002.

[47] PESSINI, T. BP2UC: Técnica para derivar Casos de Uso a partir de modelos BPMN. Dissertação (Monografia) — Universidade Estadual do Oeste do Paraná, Cascavel, 2014.

[48] SIQUEIRA, F. L.; SILVA, P. S. M. Transforming an enterprise model into a use case model in business process systems. Journal of Systems and Software, v. 96, p. 152 – 171, 2014.

[49] Conradi, R. , Basili, V., Carver, J., Shull, F., Travassos, G., "A Pragmatic Document Stan-

dards for an Experience Library: Roles, Documents, Contents and Structure.", Technical Report CS-TR-4235, University of Mary land, April 2001.

[50] VEGAS, S., APA, C., JURISTO, N. - Crossover Designs in Software Engineering Experiments: Benefits and Perils. IEEE TRANSACTIONS ON SOFTWARE ENGINEERING, VOL. 42, NO. 2, February 2016.

[51] https://en.wikipedia.org/wiki/I*. Acesso em 01/05/2019.

[52] VARA, J. L. de la; SÁNCHEZ, J. Improving requirements analysis through business process modelling: A participative approach. In: Business Information Systems: 11th International Conference, BIS 2008, Innsbruck, Austria, May 5-7, 2008. Proceedings. Berlin, Heidelberg: Springer Berlin Heidelberg, 2008. p. 165–176.

[53] RODRÍGUEZ, A.; GUZMAN, A.; MEDINA, E.; PIATTINI, M. Semi-formal transformation of secure business processes into analysis class and use case models: An mda approach. Inf. Softw

[54] TIWARI, S.; GUPTA, A. A systematic literature review of use case specifications research: Published in Information Software Technology 2015. p. 128-158.

[55] MENZEL, I.; MUELLER, M.; GROSS, A.; DOERR, J. An Experimental Comparison regarding the Completeness of Functional Requirements Specifications.

[56] J.C.S. Leite, G.D.S. Hadad, J.H. Doorn, G.N. Kaplan, A scenario construction process, Require. Eng. 5 (1) (2000) 38–61. http://dx.doi.org/10.1007/PL00010342.

[57] A.D. Toro, B.B. Jiménez, A.R. Cortés, M.T. Bonilla, A requirements elicitation approach based in templates and patterns, in: WER, 1999, pp. 17–29.

[58] G. Schneider, J. Winters, Applying Use Cases A Practical Guide, Addison-Wesley, 1998.

[59] R.J. Harwood, Use case formats: requirements, analysis, and design, in: Journal of Object-Oriented Programming, 1997.

[60] L. Mattingly, H. Rao, Writing effective use cases and introducing collaboration cases, in: Journal of Object-Oriented Programming, 1998.

[61] A. Jaaksi, Our cases with use cases, in: Journal of Object-Oriented Programming, 1998.

[62] D. Kulak, E. Guiney, Use Cases: Requirements in Context, Addison-Wesley, 2012.

[63] D. Liu, K. Subramaniam, B. Far, A. Eberlein, Automating transition from use cases to class model, in: Proceedings of the Canadian Conference on Electrical and Computer Engineering,

IEEE CCECE 2003, vol. 2, 2003, pp. 831–834.

[64] M. Misbhauddin, M. Alshayeb, Extending the uml use case metamodel with behavioral information to facilitate model analysis and interchange, Softw. Syst. Model, 2013, 1–26, http://dx.doi.org/10.1007/s10270-013-0333-9.

[65] W. Fleisch, Applying use cases for the requirements validation of component-based real-time software, in: Proceedings of the 2nd IEEE International Symposium on Object-Oriented Real-Time Distributed Computing, ISORC '99, 1999, pp. 75–84.

[66] S. Tiwari, S. Rathore, S. Gupta, V. Gogate, A. Gupta, Analysis of use case requirements using SFTA and SFMEA techniques, in: Proceedings of the 17th International Conference on Engineering of Complex Computer Systems, ICECCS 2012, July 2012, pp. 29–38.

[67] E. Insfŕan, O. Pastor, R. Wieringa, Requirements engineering-based conceptual modelling, Require. Eng. 7 (2) (2002) 61–72. http://dx.doi.org/10.1007/s007660200005.

[68] I. Jacobson, M. Christerson, P. Jonsson, G. Overgaard, Object-Oriented Software Engineering: A Use-Case Driven Approach, Addison-Wesley, Reading, MA, 1992. 1992 Edition.

[69] S.S. Somé, Supporting use case based requirements engineering, Inf. Softw. Technol. 48 (1) (2006) 43–58. <http://www.sciencedirect.com/science/article/pii/S0950584905000285>.

[70] T. Yue, L.C. Briand, y. Labiche, Facilitating the transition from use case models to analysis models: Approach and experiments, ACM Trans. Softw. Eng. Methodol. 22 (1) (2013) 5:1–5:38. http://doi.acm.org/10.1145/2430536.2430539.

[71] P. Kruchten, The Rational Unified Process: An Introduction, Addison-Wesley, Boston, 2003.

[72] J. Kettenis, Getting Started With Use Case Modeling: White Paper, Oracle Corporation, USA, 2007.

[73] M. El-Attar, J. Miller, A subject-based empirical evaluation of SSUCDs performance in reducing inconsistencies in use case models, Emp. Softw. Eng. 14 (5) (2009) 477–512. http://dx.doi.org/10.1007/s10664-008-9101-9.

[74] P. Haumer, Use Case-Based Software Development, IBM Rational Rose, 2004, 1–27.

[75] Link para os Diagramas de Casos de Uso e questionários respondidos pelos participantes. https://drive.google.com/drive/folders/1e4aBsbjBg2SOnKgS5xpy8YrRhvy6zC3g?usp=sharing.