Pós-Graduação em Ciência da Computação

Sarah Moniky Silva Ribeiro

Desenvolvimento de uma extensão da linguagem de modelagem iStar para Sistemas Críticos de Segurança - iStar4Safety

Universidade Federal de Pernambuco
posgraduacao@cin.ufpe.br
http://cin.ufpe.br/~posgraduacao

Recife
2019

Sarah Moniky Silva Ribeiro

Desenvolvimento de uma extensão da linguagem de modelagem iStar para Sistemas Críticos de Segurança - iStar4Safety

Trabalho apresentado ao Programa de Pós-graduação em Ciência da Computação do Centro de Informática da Universidade Federal de Pernambuco como requisito parcial para obtenção do grau de Mestre em Ciência da Computação.

Área de Concentração: engenharia de requisitos
Orientador: Jaelson Freire Brelaz de Castro

Recife
2019

Catalogação na fonte
Bibliotecária Monick Raquel Silvestre da S. Portes, CRB4-1217

R484d Ribeiro, Sarah Moniky Silva
Desenvolvimento de uma extensão da linguagem de modelagem iStar para sistemas críticos de segurança – iStar4Safety / Sarah Moniky Silva Ribeiro. – 2019.
200 f.: il., fig., tab.

Orientador: Jaelson Freire Brelaz de Castro.
Dissertação (Mestrado) – Universidade Federal de Pernambuco. CIn, Ciência da Computação, Recife, 2019.
Inclui referências e apêndices.

1. Engenharia de software. 2. Engenharia de requisitos. 3. Sistemas críticos de segurança. I. Castro, Jaelson Freire Brelaz de (orientador). II. Título.

005.1 CDD (23. ed.) UFPE-FQ 2019-047

Sarah Moniky Silva Ribeiro

“**Desenvolvimento de uma extensão da linguagem de modelagem iStar para Sistemas Críticos de Segurança - iStar4Safety”**

Dissertação de Mestrado apresentada ao Programa de Pós-Graduação em Ciência da Computação da Universidade Federal de Pernambuco, como requisito parcial para a obtenção do título de Mestre em Ciência da Computação.

Aprovado em: 19/02/2019.

BANCA EXAMINADORA

_____

Profa. Dra. Carla Taciana Lima Lourenço Silva Schuenemann
Centro de Informática / UFPE

_____

Prof. Dr. Johnny Cardoso Marques
Divisão de Ciência de Computação / ITA

_____

Prof. Dr. Jaelson Freire Brelaz de Castro
Centro de Informática / UFPE
(**Orientador**)

Dedico essa dissertação à meu pai, minha mãe, Sarah, Fernanda e Luiza que são meu pilar e o motivo de minhas lutas.

AGRADECIMENTOS

Agradeço à meus pais, Maria José e Zilmar por terem através de seu esforço e suor, me sustentado por todo esse mestrado: financeiramente e emocionalmente. Espero um dia poder retribuir tudo isso.

Agradeço à minhas irmãs, Sarah e Luiza e à minha sobrinha Fernanda por sempre estarem a meu lado, mesmo longe por tanto tempo. Pelo seu amor e paciência comigo.

Agradeço à minha família recifense, Dea e Lilia, por sua amizade, cuidados e por me permitirem fazer parte do lar de vocês por tanto tempo. Amo vocês, meninas, obrigada! E a dona Telma e dona Dedé por suas orações.

Agradeço também a todos que de alguma forma estiveram a meu lado nessa jornada, com palavras de incentivo e amizade: Jonas, Telma, Júnior, Luciana, Dany, Laurinha, Folha. Obrigada gente!

Meu muito obrigada a Camilo que contribuiu muito para a evolução desse trabalho com suas dicas valiosas. Agradeço também a Carla, Mariana, Jéssyka, Ênyo, João, Danyllo por toda a ajuda na realização desse trabalho. Muito obrigada!

Agradeço ao pessoal do laboratório, seus conselhos, sua amizade, brincadeiras: obrigada pessoal! Obrigada, Elaine por sua amizade e por sua ajuda.

Agradeço ao meu orientador, Jaelson por sua orientação durante todo esse trabalho e pelo seu incentivo para os meus próximos passos. Obrigada professor!

Enfim, não é possível citar todos os nomes mas o sentimento nesse momento é de um agradecimento profundo à todos e todas que me ajudaram nessa jornada. Obrigada!

RESUMO

Sistemas Críticos de Segurança (do inglês, Safety-Critical Systems - SCS) são caracterizados por serem sistemas que caso falhem ou não se comportem como esperado, podem levar à danos ou até perdas de vidas, destruição de propriedade, perdas de missões e/ou dano ambiental. Logo, gastos financeiros e esforços para garantir seu bom funcionamento são altamente justificáveis e aplicáveis. Observa-se que há uma maior probabilidade de erros relacionados à segurança (do inglês, safety) estarem associados às fases iniciais do processo de desenvolvimento, do que às fases finais (tais como a codificação). Os requisitos iniciais de segurança são definidos no documento da Análise Preliminar de Segurança (do inglês, Preliminary Safety Analysis - PSA) que é a base para outras análises. Percebeu-se a falta de uma linguagem orientada à objetivos que permitisse a modelagem de requisitos iniciais de segurança no domínio de Sistemas Críticos de Segurança. Esse trabalho tem por objetivo criar uma extensão da linguagem iStar para possibilitar a modelagem de requisitos de segurança desde os estágios iniciais de desenvolvimento do sistema, adotando uma abordagem orientada a objetivos. A extensão de iStar foi desenvolvida seguindo o processo PRISE (A PRocess to Conduct iStar Extensions). A nova linguagem permitirá uma definição dos requisitos iniciais de segurança. Foi proposta a linguagem **iStar4Safety**, com diversos construtores específicos para modelar segurança. Além disso, foi definido o **metamodelo** da linguagem acompanhado de regras de validação. A ferramenta piStar Tool de modelagem foi adaptada, originando a ferramenta piStar-4Safety Tool, que permite que modelos em iStar4Safety sejam criados. A nova linguagem de modelagem foi avaliada através de um método empírico. Um Sistema Crítico de Segurança, que visa controlar o cruzamento de ferrovias, foi modelado com a nova linguagem iStar4Safety. De acordo com o survey aplicado, a linguagem iStar4Safety é conservativa, ou seja, mantém todas as características de iStar 2.0. Adicionalmente, ela permite,de forma satisfatória, que os requisitos iniciais de segurança do Sistema Crítico de Segurança em questão sejam modelados.

Palavras-chaves: Engenharia de Requisitos. Sistemas Críticos de Segurança. iStar. Extensão. Modelagem.

ABSTRACT

Safety-Critical System (SCS) is known to be a system that, if it fails or does not perform as expected, can lead to damage or loss of life, property, missions, and / or environmental damage. Therefore, financial expenses and efforts to ensure its proper functioning are highly justifiable and applicable. It is observed that there is a greater probability that safety-related errors are associated with the early stages of the development process, rather than the later phases (such as the coding). The early safety requirements are defined in the Preliminary Safety Analysis (PSA) document that is the basis for further analysis. It was noticed the lack of an goal-oriented language that allowed the modeling of initial safety requirements in the field of Safety-Critical Systems. This work aims to create an extension of the iStar language to enable the modeling of safety requirements from the early stages of system development, adopting a goal-oriented approach. The iStar extension was developed following the PRISE process (A PRocess to Conduct iStar Extensions). The new language will allow a definition of the initial safety requirements. The iStar4Safety language was proposed, with several specific constructors for modeling safety. In addition, the metamodel of the language was defined along with validation rules. The piStar Tool modeling tool has been adapted, originating the piStar-4Safety Tool to enable iStar4Safety models to be created. The new modeling language was evaluated through an empirical method. A critical safety system, which aims to control the crossing of railways, was modeled with the new iStar4Safety language. According to the survey applied, the iStar4Safety language is conservative, that is, maintains all the features of iStar 2.0e. Moreover, it satisfactorily supports the modeling of the initial safety requirements of the given Safety-Critical System.

Keywords: Requirements Engineering. Critical Security Systems. iStar. Extension. Modeling.

LISTA DE ILUSTRAÇÕES

LISTA DE TABELAS

LISTA DE ABREVIATURAS E SIGLAS

ER	Engenharia de Requisitos
FDA	*Food and Drug Administration* - Admin. de Drogas/Alimentos dos EUA
GORE	*Goal-oriented Requirements Engineering* - Engenharia de Requisitos Orientada a Objetivos
iStar	Modeling Framework iStar
LCCS	*Level Crossing Control System* - Sistema de Controle de Cruzamento
MOF	*Meta-Object Facility* - Facilidade Meta-Objeto
OCL	*Object Constraint Language* - Linguagem para Especificação de Restrições em Objetos
PSA	*Preliminary Safety Analysis* - Análise Preliminar de Segurança
SD	*Strategic Dependency* - Dependência Estratégica
SR	*Strategic Rationale* - Raciocínio Estratégico
UML	*Unified Modeling language* - Linguagem de Modelagem Unificada

SUMÁRIO

1 INTRODUÇÃO

Esse capítulo apresenta o contexto no qual esta dissertação foi desenvolvida, bem como as motivações para sua escolha. Além disso, são elencados os objetivos e alguns trabalhos relacionados à esse trabalho que visa o desenvolvimento de uma linguagem de modelagem de requisitos para Sistemas Críticos de Segurança. Esta nova linguagem será chamada de iStar4Safety, sendo uma extensão da linguagem iStar (YU, 1995). Logo após esses tópicos, é apresentada a metodologia de pesquisa utilizada e os itens que estão fora do escopo deste. Por fim, a estrutura da dissertação é definida.

1.1 CONTEXTO

Sistemas Críticos de Segurança são definidos como sistemas que, caso falhem ou se comportem de maneira inesperada, podem levar à acidentes que resultarão em danos à pessoas ou propriedade, perdas de vidas, grandes perdas financeiras e/ou danos ao ambiente (LEVESON, 1995; LEVESON, 2011). Portanto, grande parte desses sistemas são complexos, não-triviais, exigindo assim um nível maior de acurácia em seu desenvolvimento, a fim de evitar comportamentos inaceitáveis ou indesejados. Deve-se ainda ter em mente que com a evolução do hardware e seu melhor custo/benefício, os computadores ocuparam um papel crucial e primordial em diversas áreas. Consequentemente, o software vem executando mais e mais funções críticas, seja nas áreas médica, robótica, de energia nuclear ou aviação, tais como em Sistemas de Bomba de Infusão de Insulina, Sistemas de Tráfego Aéreo, Sistemas para Robôs Sociais, mas também em áreas tidas como não inerentemente críticas (KNIGHT, 2002).

É importante salientar que o termo segurança pode se referir tanto ao termo *safety* quanto à *security*. Para atender à propriedade de *security*, que não é o foco desse trabalho, o sistema deve ser livre de ações maliciosas (por agentes não-autorizados), seja de vazamento, destruição e/ou modificação de informações (BERRY, 1998). Questões relacionadas à *security* dizem respeito, por exemplo, à acessos não autorizados ao sistema para obtenção de senhas de usuários. Nessa dissertação o termo segurança será usado como sinônimo de *safety*.

Preocupações e inserção de conceitos relacionados à segurança (do inglês *safety*) devem acontecer desde o início do processo de desenvolvimento do software (LEVESON, 2011) estendendo-se até o fim de sua vida útil, tanto no desenvolvimento, quanto na documentação. Tal afirmação se aplica então, aos requisitos de segurança que devem se manter o mais completos e atualizados quanto possível.

É importante ainda salientar que a propriedade de segurança não pode ser garantida considerando-se componentes (software, hardware, dados, processos, pessoas) isolados no

sistema. Isso se deve ao fato da mesma ser uma propriedade emergente. Isso significa que, para sua avaliação, é necessária uma verificação holística do sistema, ou seja, o sistema deve ser pensado como sendo mais do que a união de seus componentes: devem ser consideradas também interfaces entre os componentes e subcomponentes (LEVESON, 2011).

Para o desenvolvimento de sistemas em geral, algumas fases são necessariamente executadas, sendo a fase de Engenharia de Requisitos (ER) uma das primeiras e a mais importante das fases. Isto se deve ao fato de que requisitos mal definidos e/ou faltosos irão inevitavelmente afetar a qualidade do produto desenvolvido para pior e, como já pontuado, falhas na definição de requisitos são as maiores causas de falhas em segurança. É através da engenharia de requisitos que serão definidos os *stakeholders* (que são pessoas ou organizações que tem interesse legítimo em um projeto ou entidade) e as necessidades de cada um deles. Essas necessidades, tratadas aqui como objetivos, serão os futuros requisitos para o sistema, levando em conta também para uma definição mais completa, questões de compromissos (*tradeoffs*) e priorização de requisitos (LAPOUCHNIAN, 2005).

Na fase inicial da Engenharia de Requisitos, o uso de objetivos tem provado ser bastante útil pois provê um melhor meio de organizar e justificar os requisitos de software (ANTON, 1996). Nesse sentido, a *Goal-oriented Requirements Engineering* - Engenharia de Requisitos Orientada a Objetivos (GORE) tem sido uma grande aliada para a definição de requisitos de forma mais assertiva e holística (ANTON, 1996). As linguagens de modelagem orientadas à objetivos focam em atividades anteriores à formulação de requisitos de software, já que as mesmas descrevem os requisitos do sistema através de uma visão de mais alto-nível, isto é, através de objetivos associados ao responsável pela sua realização. Nesta dissertação usaremos uma abordagem GORE para modelar requisitos de segurança o mais cedo possível no processo de desenvolvimento de software.

De acordo com Nuseibeh e Easterbrook (2000), devido à natureza dinâmica do desenvolvimento, requisitos podem variar e conflitar entre si, podendo ser difíceis de entender, além de estarem sujeitos a restrições variadas que venham a ser definidas no contexto de seu desenvolvimento. Em se tratando de sistemas críticos, onde a complexidade é comum e algumas falhas intoleráveis, muita atenção deve ser dada para a fase de requisitos, visto que falhas na especificação de requisitos do software, ao contrário de erros de codificação, tem sido identificados como a principal causa de vários acidentes e catástrofes dos últimos vinte anos (MEDIKONDA; PANCHUMARTHY, 2009).

A linguagem de modelagem iStar tem sido usada na especificação de requisitos iniciais de variados sistemas (YU, 1995). A linguagem iStar é bastante usada em conjunto com outros formalismos devido a alguns fatores não cobertos pela linguagem, como: necessidade de conectar a engenharia de requisitos iniciais à engenharia de requisitos finais, para customizar a linguagem à algum domínio específico, para combinar a linguagem com outros paradigmas que a tornem mais eficiente, entre outros.

Várias extensões tem sido propostas na literatura a fim de possibilitar a construção de modelos iStar que contemplem áreas específicas, como por exemplo, extensões para a modelagem de Sistemas Adaptativos (MORANDINI et al., 2017; Lei; Ben; He, 2015), para Linhas de Produtos de Software (MELLADO; MOURATIDIS; FERNáNDEZ-MEDINA, 2014; ASADI et al., 2011). Porém, um ponto preocupante é que a maioria das extensões de iStar não é criada com o intuito de integrar-se à trabalhos da comunidade iStar, fato que pode atrapalhar sua difusão e utilidade (FRANCH et al., 2011).

Nesta dissertação, temos a intenção de que a proposta da extensão da linguagem iStar seja apropriada para segurança e esteja de acordo com as melhores práticas para desenvolvimento de extensões propostas por pesquisadores da área (GONÇALVES et al., 2018a; GONÇALVES et al., 2018c).

1.2 MOTIVAÇÃO

Sistemas Críticos de Segurança devem ser construídos visando garantir a mitigação de perigos, à fim de prevenir acidentes. É sabido que tais acidentes podem levar à grandes perdas financeiras, danos ambientais, ferimentos e até mortes (LEVESON, 2011).

Verificou-se em um trabalho prévio realizado por Vilela et al. (2017b), a falta de construtores em linguagens GORE para a modelagem da Análise Preliminar de Segurança. Os autores ainda concluem que, como linguagens promissoras para extensão de segurança estão as linguagens iStar e KAOS.

Deve-se ter em mente a natureza intencional de iStar, e definir como inserir elementos não-intencionais sem alterar a linguagem original.

KAOS apresenta a ideia de obstáculos como elementos que impedem a realização dos objetivos. Através da análise de obstáculos, pode-se identificar os motivos pelos quais um objetivo do sistema poderia não ser alcançado (LAMSWEERDE; LETIER, 2004). Neste trabalho, consideraremos os perigos (do inglês, *hazards*) como obstáculos para a segurança do sistema.

Extensões tem sido propostas para o iStar (GONÇALVES et al., 2018a) visando possibilitar seu uso em outros domínios ((MORANDINI et al., 2017; MELLADO; MOURATIDIS; FERNáNDEZ-MEDINA, 2014; MOURATIDIS; GIORGINI, 2007; MOURATIDIS et al., 2013; GHANAVATI; AMYOT; RIFAUT, 2014), entre outras). Nesse sentido, o processo PRISE foi proposto à fim de definir a melhor forma de criar extensões de qualidade para a linguagem iStar (GONÇALVES et al., 2018b). Entre as boas práticas, observadas em Gonçalves et al. (2018c) é importante que as extensões propostas preservem a linguagem original do iStar, ou seja, que sejam conservativas, bem como elas sejam o mais simples possível.

Tendo esses pontos em mente, define-se na próxima seção, os objetivos à serem alcançados pelo desenvolvimento dessa dissertação.

1.3 OBJETIVOS

Baseado nesse contexto, a principal questão de pesquisa a ser investigada por essa dissertação é:

Questão de Pesquisa *Quais construtores uma linguagem orientada à objetivos deveria ter para modelar requisitos iniciais de segurança sem comprometer a simplicidade da linguagem?*

Para responder à questão de pesquisa, foram definidos os seguintes objetivos específicos:

- O1 - Criar uma extensão que seja simples, conservativa e adicione a menor quantidade possível de novos construtores à linguagem iStar;
- O2 - Definir quais conceitos necessita-se modelar com a extensão criada;
- O3 - Definir um metamodelo da extensão, conservando o metamodelo nativo de iStar 2.0;
- O4 - Definir a sintaxe concreta da extensão;
- O5 - Criar/adaptar uma ferramenta de modelagem que permita a modelagem utilizando a extensão criada;
- O6 - Ilustrar o uso da extensão através da modelagem de um Sistema Crítico de Segurança;
- O7 - Avaliar a extensão através de um método empírico.

1.4 TRABALHOS RELACIONADOS

Salienta-se que não foram encontrados trabalhos que tratam da criação de uma extensão de iStar para modelagem de Sistemas Críticos de Segurança. Portanto, alguns trabalhos que tem relação com o que foi desenvolvido nesta dissertação são apresentados abaixo.

Outras linguagens de modelagem orientadas à objetivos permitem a modelagem parcial de requisitos não intencionais. A linguagem KAOS, do inglês *Knowledge Acquisition in autOmated Specification* ou *Keep All Objects Satisfied* proposta em (LAPOUCHNIAN, 2005; LAMSWEERDE; LETIER, 2004; DARDENNE; LAMSWEERDE; FICKAS, 1993) é uma linguagem de modelagem orientada à objetivos com um conjunto de técnicas de análise formal bem estabelecidas. Um importante conceito na linguagem KAOS é a ideia de obstáculo (LAMSWEERDE; LETIER, 2000) que permite a representação de situações onde o objetivo, expectativa ou requisito pode ser obstruído por algum fato.

Outra abordagem proposta na literatura vem do trabalho de (MOURATIDIS; GIORGINI, 2007), Secure Tropos, uma extensão da metodologia Tropos. Secure tropos adiciona o conceito de restrição de segurança (do inglês *security*), além de estender os conceitos de dependência, objetivo, tarefa e recurso da linguagem nativa para o tipo *secure*. As ameaças à objetivos de segurança são circunstâncias que podem causar perdas, representadas por construtores ameaças (threats).

Em Martins e Gorschek (2016), os autores apresentam, através de uma revisão de literatura sistemática, as abordagens que tem sido propostas para elicitar, modelar, especificar e validar os requisitos relacionados à segurança no contexto de Sistemas Críticos de Segurança. Como resultado do trabalho, os autores definiram a necessidade de uma maior integração entre profissionais de Engenharia de Requisitos e de segurança. Além disso, concluíram que existe uma necessidade de validações na indústria. Também ressaltaram a falta de evidências de utilidade e usabilidade na maioria das abordagens e sugerem que novos estudos investiguem como melhorar a comunicação entre participantes do processo de desenvolvimento de Sistemas Críticos de Segurança.

1.5 METODOLOGIA DE PESQUISA

O presente trabalho foi desenvolvido através dos métodos empírico e de engenharia apresentados por Glass (1994) e Wohlin et al. (2012). O método de engenharia trata-se da observação das soluções já existentes, propondo soluções melhores, ou mesmo construindo ou desenvolvendo novas soluções, através de medições e análises de forma iterativa até que seja verificado que não há mais melhorias a serem implementadas. Já o método empírico consiste em propor um método para realizar a avaliação da proposta, através de casos de uso, experimentos ou *surveys*, por exemplo (GLASS, 1994; WOHLIN et al., 2012).

Utilizou-se como base para a definição das características necessárias para atender aos objetivos dessa proposta, o trabalho de Vilela et al. (2017b). Nesse trabalho, os autores definem, através de uma vasta pesquisa bibliográfica em relação a normas e outros trabalhos relevantes relacionados à segurança, as características que devem ser modeladas em sistemas críticos à fim de atender ao que é necessário para a definição da Análise Preliminar de Segurança.

Prezando pela qualidade e eficiência no processo de desenvolvimento da extensão da linguagem iStar, utilizamos o processo PRISE, proposto por Gonçalves (2018). Ele visa sistematizar o desenvolvimento de extensões de iStar, definindo os passos necessários e recomendados para o desenvolvimento de uma extensão de qualidade. Sendo assim, seguimos as seguintes etapas:

1. Análise da necessidade da extensão para a área de Sistemas Críticos de Segurança;
2. Descrição de conceitos da extensão de iStar para Sistemas Críticos de Segurança;

3. Desenvolvimento da extensão iStar4Safety;

   Desenvolvimento da ferramenta para criação de modelos da extensão.

4. Validação e avaliação da extensão iStar4Safety.

5. Publicação da extensão iStar4Safety no catálogo CATIE.

É importante ressaltar que a realização do subprocesso cinco (Checagem de novos construtores à serem inseridos), realizada após a etapa de validação e avaliação da extensão, não foi necessária, devido à não terem sido identificados novos construtores após a avaliação e validações da extensão. No entanto, todos os demais cinco subprocessos foram executados.

1.6 FORA DO ESCOPO

Como a abordagem proposta integra um contexto maior, alguns aspectos não irão compor o escopo desse trabalho. Deve-se atentar ao fato de que a Análise Preliminar de Segurança visa ser uma análise base para as outras análises. À partir da Análise Preliminar de Segurança, os requisitos relacionados à segurança do sistema serão tratados em outros níveis pelas outras análises. Logo, os artefatos gerados à fim de modelar as definições dessa análise irão também poder ser usados ou evoluídos por outras análises. Apesar disso, os seguintes tópicos não serão diretamente tratados nessa dissertação:

1. **Análise de Segurança do Sistema**: Essa análise é usada para refinar e definir causas para os perigos encontrados na Análise Preliminar de Segurança. Algumas respostas quanto aos comportamentos dos requisitos de segurança em relação ao sistema integrado serão consideradas. A modelagem de definições de tal análise não será abordada por esse trabalho;

2. **Análise de Segurança do Subsistema**: Trata-se de uma sub-análise da Análise de Segurança do Sistema, preocupando-se em examinar como os comportamentos dos componentes, ou subsistemas, irão afetar a segurança do sistema no geral. Essa análise também não será foco da construção de construtores desse trabalho;

3. **Análise de Segurança de Operação e Suporte**: Nessa análise são identificados perigos e procedimentos de redução de riscos durante as fases de operação do sistema e suporte ao mesmo. Essa fase dura até o fim da vida útil do sistema. O trabalho atual não foca em modelar esse artefato;

4. **Representação das Restrições da Linguagem com OCL**: É sabido que a melhor forma de definir regras de validação é através de uma linguagem formal, como *Object Constraint Language* - Linguagem para Especificação de Restrições em

Objetos (OCL). Porém, devido à restrições de tempo e baseando-nos em como os autores da linguagem nativa definem as suas restrições, essa representação não será escopo dessa dissertação.

1.7 ESTRUTURA DA DISSERTAÇÃO

Essa dissertação encontra-se dividida em sete capítulos.

O **capítulo 1** apresenta a introdução desse trabalho, concentrando-se em mostrar o contexto, a motivação para realização da dissertação, os objetivos, bem como trabalhos que se relacionam à essa pesquisa. Além disso, apresenta-se a metodologia de pesquisa usada e os itens que estão fora do escopo desse trabalho são elencados.

Já o **capítulo 2** apresenta o referencial teórico com conceitos importantes para um melhor entendimento do estudo realizado, apresentando uma descrição do sistema usado como exemplo para ilustração durante o desenvolvimento do trabalho, temas relacionados à engenharia de requisitos, além do processo PRISE.

O **capítulo 3** apresenta conceitos relacionados à segurança, além da ferramenta de modelagem adotada.

O **capítulo 4** consiste na descrição dos procedimentos realizados para criação da extensão iStar4Safety, seguindo os passos indicados pelo processo PRISE.

Quanto ao **capítulo 5**, nele é mostrada a ilustração do uso da extensão, realizada através da modelagem de um Sistema Crítico de Segurança: um Sistema de Cruzamento de Ferrovias. As 6 (seis) diretrizes para a modelagem com iStar4Safety foram seguidas, mostrando como realizar uma iteração completa. Os modelos finais com iStar4Safety do sistema foram apresentados e a verificação de sua completude foi realizada.

O **capítulo 6** mostra o método empírico aplicado, uma modelagem de um cenário mais a coleta d a opinião dos participantes através de um *survey*, realizado com o objetivo de avaliar a linguagem iStar4Safety quanto à qualidade dos modelos desenhados, eficiência, usabilidade e eficácia.

Finalmente, o **capítulo 7** apresenta uma conclusão do trabalho, apresentando suas limitações e contribuições geradas. Além disso, são apresentados os trabalhos futuros que poderão ser desenvolvidos tomando esse trabalho como base.

2 REFERENCIAL TEÓRICO

Nesse capítulo apresenta-se os conceitos necessários para o entendimento desse trabalho. Na seção 2.1, o Sistema Crítico de Segurança que será usado para ilustrar a dissertação, um Sistema de Bomba de Infusão de Insulina, é apresentado. Logo após, a Engenharia de Requisitos Orientada à Objetivos é detalhada. Nas seções 2.3 e 2.4 apresentamos as duas linguagens orientadas à objetivos que contribuíram para o desenvolvimento do trabalho: KAOS e iStar. A seguir, o processo utilizado para a criação da linguagem iStar4Safety, PRISE, é explicado. Por fim, a ferramenta de apoio usada para modelagem dos diagramas desse trabalho complementa esse capítulo.

2.1 BOMBA DE INFUSÃO DE INSULINA

Com o intuito de ilustrar os conceitos utilizados na dissertação, é apresentado nessa seção o Sistema de Bomba de Infusão de Insulina. Trata-se de um Sistema Crítico de Segurança, ou seja, caso se comporte de forma não esperada ou não desejada pode levar à acidentes que podem ocasionar perdas financeiras, danos ao ambiente, injúrias e até mortes. No contexto de tal Sistema, acidentes podem ser a aplicação de uma super dosagem de insulina, que pode levar o paciente à um estado de coma, ou ainda o acidente de baixa dosagem de insulina, que pode levar o paciente à cegueira.

Zhang, Jones e Jetley (2010) chamam a atenção de que a terminologia à ser usada em segurança muitas vezes dificulta análises, mas que a prioridade deve ser a de aplicar algum método disciplinado à fim de avaliar como determinado projeto pode causar danos.

Esse sistema foi escolhido devido à sua complexidade e criticidade de seu uso, além de ser bastante utilizado pela literatura para esse fim (MARTINS et al., 2015; VILELA et al., 2017b; SOMMERVILLE, 2010). Em 2008, o *Food and Drug Administration* - Admin. de Drogas/Alimentos dos EUA (FDA) relatou mais de cinco mil eventos adversos que foram reportados e estavam relacionados ao uso de bombas de infusão de insulina (ZOUGHBI; BRIAND; LABICHE, 2011).

O exemplo foi retirado e adaptado de Martins et al. (2015) e complementado pelos trabalhos de Zhang, Jones e Jetley (2010), Zhang et al. (2011) que apresentam, respectivamente, o desenvolvimento de uma bomba de infusão de insulina de baixo custo, um modelo genérico de bomba de infusão de insulina e uma análise de perigos para tal modelo. O nível de abstração do desenvolvimento de um sistema irá definir quão profunda será a análise e, quanto mais profunda, mais perigos, causas e estratégias para solucionar, evitar ou controlar efeitos de tais perigos serão encontrados (ZOUGHBI; BRIAND; LABICHE, 2011). Como o intuito nessa seção é ilustrar a extensão criada utilizando um Sistema Crítico de Segurança, não houve a intenção de analisar exaustivamente todos os requisitos da bomba

de insulina ou seus perigos, mas sim utilizar um subconjunto suficiente para possibilitar a validação da extensão.

A bomba de infusão de insulina foi criada para auxiliar no tratamento da Diabetes *Mellitus* tipo 1. Segundo Martins et al. (2015) essa bomba automatizada de infusão de insulina dá maior flexibilidade ao tratamento, na medida em que substitui a ação humana em várias etapas da infusão, simulando o comportamento do corpo humano. A bomba de insulina pode aplicar dosagens de insulina de ação rápida (*bolus*) ou contínua (basal).

Conforme descrito pelos autores, citando o trabalho de Zhang, Jones e Jetley (2010), um modelo genérico da bomba deve incluir uma interface com o usuário, que será uma tela LCD e avisos em forma de alarmes sonoros. Além disso, a bomba é composta por outros dispositivos de hardware: microprocessador, bateria, mecanismo de infusão, cateter e reservatório de insulina.

2.1.1 Requisitos Gerais da Bomba de Infusão de Insulina

Uma bomba de infusão de insulina trata-se de um sistema cujo objetivo é aplicar dosagens de insulina de ação rápida através de um cateter colocado sob a pele do usuário. Ela é usada para tratar a diabetes Mellitus do tipo I, mantendo o nivelamento adequado de glicose no sangue do paciente.

A bomba de insulina é composta de alguns componentes, a saber:

- Bateria não-recarregável;
- Reservatório de insulina – Seringa comum;
- Motor de passos;
- Interface com o usuário – Tela LCD, quatro botões e um alarme;
- Conjunto de infusão; e
- Software controlador da bomba.

O dispositivo poderá ser usado 24 (vinte e quatro) horas por dia, durante os sete dias da semana. Não possui restrições quanto à faixa etária, porém é necessário que o paciente saiba usá-la, sendo treinado para tal. Ao utilizar a bomba de insulina, deve-se atentar à necessidade de mobilidade do paciente. A bomba de infusão deve ser desenvolvida minimizando custos. Logo uma seringa comum deve ser usada como reservatório.

A bomba permite dois tipos de aplicação de insulina: Basal (contínua) e *Bolus* (rápida). A aplicação basal é realizada de forma contínua durante as 24 horas do dia. Já a aplicação *bolus* é uma dose adicional, aplicada de forma mais rápida que a dose basal.

O Controlador da Bomba de Infusão de Insulina representa uma abstração do Software da Bomba de Insulina (ZHANG; JONES; JETLEY, 2010) e deve realizar algumas funções básicas:

- Interpretar comandos do usuário e entradas recebidas através dos dispositivos de entrada e agir de acordo;
- Manter os perfis de entrega de insulina definidas pelo usuário;
- Recomendar doses apropriadas de *bolus* para corrigir baixo nível de glicose no sangue;
- Codificar e enviar instruções para o mecanismo de entrega de insulina permitindo a administração precisa de insulina baseada nos perfis de entrega configurados pelo usuário;
- Enviar informações para mecanismos de saída permitindo o monitoramento do status de todo processo pelo paciente;
- Instruir dispositivo de saída no envio de alarmes e alertas que sejam perceptíveis ao usuário; e
- Armazenar dados e eventos importantes durante o uso da bomba para futuras análises.

É importante salientar que o dispositivo pode sofrer interferências derivadas do ambiente físico que o circunda como: temperatura, som, pressão e radiação.

O principal objetivo de um paciente com diabetes que utiliza a bomba de insulina é receber a dosagem correta de insulina a fim de que o nível de glicose em seu sangue esteja satisfatório. Após inicializar o sistema, o usuário da bomba de insulina poderá verificar as configurações gerais da bomba: verificar a tela LCD, verificar alarmes, verificar o status de uso motor de passos, e os níveis de bateria e insulina restantes no reservatório. O paciente pode realizar a troca de bateria. O paciente pode ainda trocar o conjunto de infusão, sendo necessário para isso, interromper o processo de infusão e recolher o êmbolo. O status de bateria e insulina devem ser apresentados em uma tela separada para garantir maior precisão na análise.

O usuário da bomba pode configurar os perfis de infusão basal (contínua). Os perfis devem ser armazenados, podendo ser alterados quando necessário. O usuário deseja receber a infusão basal, o que é realizado pela bomba de insulina.

O usuário pode necessitar receber auxílio ao utilizar a bomba de infusão, o que deve ser feito pelo fornecedor da bomba.

O usuário pode ainda configurar a quantidade desejada da infusão *bolus* (rápida), que irá complementar a infusão basal (contínua). A infusão *bolus* pode ser alterada. O usuário deseja receber a infusão *bolus*, o que é realizado pela bomba de insulina.

Ao iniciar qualquer tipo de infusão, a mesma deve ser ativada e deve ser realizado o gerenciamento do motor de passos, que é realizado pelo controlador da bomba de insulina e significa acionar o motor de passos e realizar a contagem dos passos.

A bomba de infusão de insulina deve realizar o tratamento do paciente de forma adequada. Para isso, o software de controle da bomba deve gerenciar a aplicação das infusões *bolus* e basal: iniciar as infusões, ou reativar caso a mesma tenha sido interrompida. Além disso, a bomba pode interromper a infusão basal ou ainda cancelar a infusão *bolus*.

O software de controle deve também monitorar os sensores e manter a tela LCD atualizada. Além disso ele tem a função de reinicializar a bomba corretamente, o que é um objetivo crítico para evitar perigos. O controlador de sensores da bomba de infusão de insulina é o responsável por realizar o controle dos sensores, verificando os níveis de bateria e insulina no reservatório. Além de atualizar a tela LCD e ativar alarmes quando necessário.

Para melhor ilustrar o sistema, em adição é apresentado o fornecedor da bomba de insulina, que deve fornecer suporte ao uso da bomba, sendo responsável por treinar os usuários ou tirar as dúvidas que os mesmos possuam, bem como a realização de manutenções no dispositivo.

2.1.2 Requisitos de Segurança da Bomba de Infusão de Insulina

O uso da bomba de infusão está associado a alguns perigos, seja de mau-funcionamento, mal-uso ou interferências do ambiente ao redor do mesmo. Apresenta-se portanto, esses perigos, com o intuito de modelar os requisitos de segurança da bomba.

O paciente deve receber a dosagem correta de insulina, o que é um objetivo crítico. Ou seja, deve-se evitar que aconteça a superdosagem (*overdose*) ou uma subdosagem (*underdose*), que seriam resultados de dosagens erradas e podem causar acidentes.

A superdosagem de insulina é um acidente de mais alto impacto e pode acontecer devido à: configuração de valor maior que o correto para dosagem de infusão (pode ser causado por paciente destreinado), fluxo livre de insulina (que pode acontecer devido ao motor de passos estar liberando mais insulina que desejado), aplicações excessivas de infusão *bolus* (que pode acontecer se o usuário faz requisição de *bolus* e não se alimenta, ou ainda se o usuário é destreinado), e por falha no sistema de entrega de insulina .

Já a subdosagem de insulina, que pode levar o paciente à cegueira, é considerado um acidente de impacto **muito severo**, e pode acontecer devido ao fato do usuário configurar um valor abaixo do correto para dosagem de infusão (o que pode acontecer devido ao usuário não ser treinado), vazamento de insulina (o paciente se move, causando a desconexão do cateter ou o paciente não tem consciência de que a bomba está desconectada, ou ainda o reservatório de insulina estar quebrado), o reservatório de insulina fica vazio (que pode ter como causa não ter insulina próxima para recarregar ou o reservatório de insulina estar quebrado) e também caso haja falha no sistema de entrega de insulina.

Outro objetivo crítico é que o usuário não receba choque elétrico ao utilizar a bomba de infusão, que pode causar danos de nível **considerável** ao paciente, como queimaduras. O choque elétrico pode acontecer devido à bomba não ter uma proteção contra descarga

elétrica. Ainda, caso o dispositivo entre em contato com água ou umidade o que pode gerar descargas elétricas. Falhas de componentes ou circuitos elétricos podem ser outras causas para choques elétricos.

O paciente pode receber a dosagem menor ou maior que o esperado caso a bomba volte inadvertidamente ao estado de fábrica. Essa volta inadvertida pode ser causada por falhas de *software* ou *hardware*.

O paciente deseja também não sofrer infecção ao trocar a bateria e ao trocar o conjunto de infusão. Caso aconteça uma infecção, o impacto do acidente pode ser **muito severo**. A infecção pode ocorrer caso a bomba seja exposta à substâncias nocivas, que pode ser causado pela bomba ser compartilhada ou pela bomba ser esterilizada incorretamente.

Se a bomba não for reinicializada corretamente, pode causar um acidente de nível de impacto **catastrófico**, já que pode levar à aplicação de uma dosagem errada de insulina, que no pior caso pode levar à morte. O perigo que impede o reinício correto da bomba pode ser uma leitura errada do histórico de perfis armazenados.

2.1.2.1 Estratégias de Mitigação de Acidentes

Algumas estratégias de segurança devem ser definidas com o objetivo de tratar os perigos. Esse tratamento pode se dar por três formas: eliminar o perigo, reduzir sua probabilidade de ocorrência ou reduzir os danos causados por ele. A seguir, apresenta-se algumas estratégias que poderão ser usadas.

Para evitar que o usuário destreinado configure um valor superior ou inferior ao valor desejado, que pode ocasionar um perigo, pode-se limitar os valores da infusão em tela. Assim o intervalo de valores possível será limitado. Também pode-se realizar o treinamento do usuário. O treinamento deve ser ministrado pelo fornecedor da bomba.

Pode-se evitar que válvulas quebradas no caminho de entrega levem à um acidente, solicitando que o usuário verifique constantemente o caminho de entrega da insulina. Essa estratégia também, irá servir para casos de desconexão da bomba.

Caso o usuário solicite a infusão *bolus*, sem se alimentar, correndo o risco de superdosagem, a solução poderá ser, quando o paciente solicitar *bolus*, o controlador da bomba enviar um aviso alertando sobre a necessidade de se alimentar antes da aplicação.

Caso o sistema de entrega falhe, ou aconteça falha de algum componente, a solução pode ser o controlador da bomba enviar um alerta sonoro à fim de deixar o usuário ciente da falha. Além disso, o usuário pode receber suporte do fornecedor da bomba para lidar com esse perigo.

Em relação ao perigo do paciente se mover e desconectar o cateter, a solução mais segura pode ser solicitar que ele verifique constantemente o caminho de entrega. Para tratar o fato do reservatório estar quebrado, deve-se solicitar que o usuário verifique constantemente o nível de insulina. Se não houver insulina em local próximo para repor

o reservatório, deve-se acionar o serviço médico responsável. Caso o reservatório esteja quebrado, a solução pode ser trocar o reservatório.

Para mitigar problemas com o compartilhamento da bomba, deve-se acessar o documento “Políticas de Compartilhamento”. Além disso, para evitar o problema da esterilização ser realizada de forma errada, o paciente deve acessar o Manual de esterilização".

Os perigos que podem gerar uma descarga elétrica, podem ser evitados através de um suporte especializado fornecido pelo fornecedor da bomba de infusão e a interrupção momentânea de uso do dispositivo.

Para problemas oriundos de falta de treinamento, a estratégia será o usuário receber suporte para uso, que é dado pelo fornecedor da bomba de infusão, que pode ser um treinamento para uso ou somente para tratar dúvidas do usuário.

Ao reinicializar a bomba de infusão, para evitar o perigo da bomba voltar inadvertidamente ao estado de fábrica e para evitar que uma leitura errada do histórico de perfis armazenados aconteça, deve-se solicitar que o usuário verifique as configurações após cada reinicialização.

2.2 ENGENHARIA DE REQUISITOS ORIENTADA À OBJETIVOS

A Engenharia de Requisitos define o propósito do sistema a ser desenvolvido, considerando e identificando os *stakeholders* e as necessidades que os mesmos buscam que sejam atendidas pelo sistema, documentando-as de uma forma passível de análise, comunicação e implementação (NUSEIBEH; EASTERBROOK, 2000).

Nesta dissertação estaremos seguindo os princípios da Engenharia de Requisitos Orientada à Objetivos, que pode abarcar tanto as fases iniciais como tardias do desenvolvimento. Os objetivos dos *stakeholders* são tipicamente modelados na forma de objetivos de alto nível e abstratos na fase inicial de requisitos, o que possibilita uma elicitação e análise mais conveniente para esse estágio. Ela considera as opções de desenvolvimento, ao contrário da fase tardia que já é orientada por requisitos mais bem definidos. Os modelos criados na fase inicial, devem sofrer refinamentos conforme o desenvolvimento do sistema evolui (HORKOFF; YU, 2016).

A Engenharia de Requisitos Orientada à Objetivos (do inglês Goal-oriented Requirements Engineering) - GORE), considera uma perspectiva mais ampla do que o sistema à ser criado. Ela permite a modelagem de preocupações de mais alto-nível no contexto de sistemas complexos, considerando portanto não apenas o sistema, mas também o ambiente ao seu redor. Foca em atividades anteriores à formulação de requisitos do sistema de software e normalmente compreende as atividades de elicitação de objetivos, refinamento dos objetivos, análises variadas de objetivos e a definição de agentes responsáveis por cada um deles (LAPOUCHNIAN, 2005). Um dos seus principais benefícios é prover suporte à análise de requisitos das fases iniciais do desenvolvimento.

Os objetivos definem quais metas o sistema deve atender, descrevendo os motivos pelos quais o sistema é necessário, além de guiar decisões em vários níveis organizacionais (ANTON, 1996). Os objetivos no processo de elicitação de requisitos, irão de objetivos de alto-nível à objetivos de mais baixo nível, através de sucessivos refinamentos (NUSEIBEH; EASTERBROOK, 2000). O raciocínio de objetivos através de árvores de refinamento permite a rastreabilidade entre objetivos estratégicos de alto-nível e requisitos de nível mais baixo (LAPOUCHNIAN, 2005). Uma outra vantagem da modelagem através de objetivos é a comunicação de requisitos aos *stakeholders*, devido ao nível de abstração fornecido por essa abordagem.

Existem várias linguagens de modelagem de requisitos orientadas à objetivos. Entre estas encontram-se as linguagens KAOS (DARDENNE; LAMSWEERDE; FICKAS, 1993) e iStar (YU, 1995; DALPIAZ; FRANCH; HORKOFF, 2016) que serão apresentadas nas próximas seções.

2.3 KAOS

KAOS do inglês *Knowledge Acquisition in autOmated Specification* ou *Keep All Objects Satisfied* é uma linguagem de modelagem orientada à objetivos com um conjunto de técnicas de análise formal bem estabelecidas (LAPOUCHNIAN, 2005; LAMSWEERDE; LETIER, 2004; DARDENNE; LAMSWEERDE; FICKAS, 1993).

Trata-se de um paradigma semi-formal que modela e estrutura objetivos. A linguagem combina redes semânticas para modelagem conceitual de objetivos, premissas, agentes, objetos e operações do sistema seguindo uma lógica temporal baseada na definição de objetivos e objetos.

Os refinamentos de objetivos em KAOS terminam quando o sub-objetivo é realizável por algum agente individual associado ao mesmo.

Requisitos em KAOS são objetivos sobre responsabilidade de um agente no sistema e expectativas são objetivos sobre responsabilidade de um agente no ambiente.

KAOS possui quatro modelos básicos: (1) *Goal Model* (Modelo de Objetivos), (2) *Responsability Model* (Modelo de Responsabilidade), (3) *Operation Model* (Modelo de Operação) e o (4) *Object Model* (Modelo de Objeto). Os modelos e suas relações são apresentados na figura 1 (OBJECTIVER, 2007).

No modelo de objetivos, os *objetivos* do sistema são modelados. Eles são representados graficamente por paralelogramas da cor azul. Os **objetivos** podem ser refinados em sub-objetivos através de setas com círculos amarelos. Os objetivos podem se conflitar, o que é representado pelo link ***conflicts***, representado pelo link com o elemento vermelho raio, entre os dois objetivos na figura 1. **Expectativas** (*Expectation*) também são representadas nesse modelo através de um paralelograma de cor amarela. Já as **Propriedades do Domínio** (*Domain Property*) representam propriedades relevantes ao domínio da aplicação. **Requisitos** (*Requirements*) são representados graficamente por paralelogramas com

as bordas grossas e devem estar associados ao ator que o concretiza. O ator se relacionará ao requisito pelo relacionamento de *responsabilidade* (*Responsibility*), uma seta com um círculo vermelho, caso o agente seja responsável por ele. No entanto, quando o agente é responsável por alguma expectativa, a sua ligação à ela é feita por meio do link de **responsabilidade**, representado por uma seta com um círculo rosa. O modelo de objetivos estará completo quando todos os seus objetivos-folha forem ou uma expectativa, ou um requisito ou uma propriedade do domínio. Todos os construtores são apresentados na figura 1 (OBJECTIVER, 2007; DARDENNE; LAMSWEERDE; FICKAS, 1993; LAPOUCHNIAN, 2005).

2.3.1 Obstáculos

Um importante conceito na linguagem KAOS é a ideia de obstáculo. O construtor obstáculo representa situações onde o objetivo, expectativa ou requisito pode ser obstruído por algum fato. Esta noção de obstáculo será de grande utilidade na modelagem de Sistemas Críticos de Segurança. Ao considerar o refinamento de um obstáculo, deve-se considerar que caso um obstáculo seja violado, é pelo fato de ao menos um de seus sub-objetivos ter sido violado.

Obstáculos são ligados ao objetivo que obstruem através do link de **obstrução**. Obstáculos podem ser refinados assim como objetivos através de refinamentos E ou OU (OBJECTIVER, 2007).

Os **obstáculos** podem ser resolvidos por várias formas. A primeira diz respeito à definir novos requisitos que previnam a ocorrência do obstáculo, ligando esse novo requisito ao obstáculo através do link de **resolução** (link com seta da cor verde na figura 1). A segunda forma é restaurar o objetivo obstruído, criando um requisito de restauração. Uma terceira alternativa seria mitigar efeitos causados pela ocorrência do obstáculo.

Nessa dissertação, um obstáculo será considerado como uma negação de um objetivo, já que a realização de um obstáculo irá impedir que o objetivo aconteça. Um exemplo de obstáculo ao objetivo "Paciente não receber descarga elétrica"pode ser uma "Falha de componentes ou circuitos elétricos", já que tal falha pode ser um impeditivo à execução do objetivo.

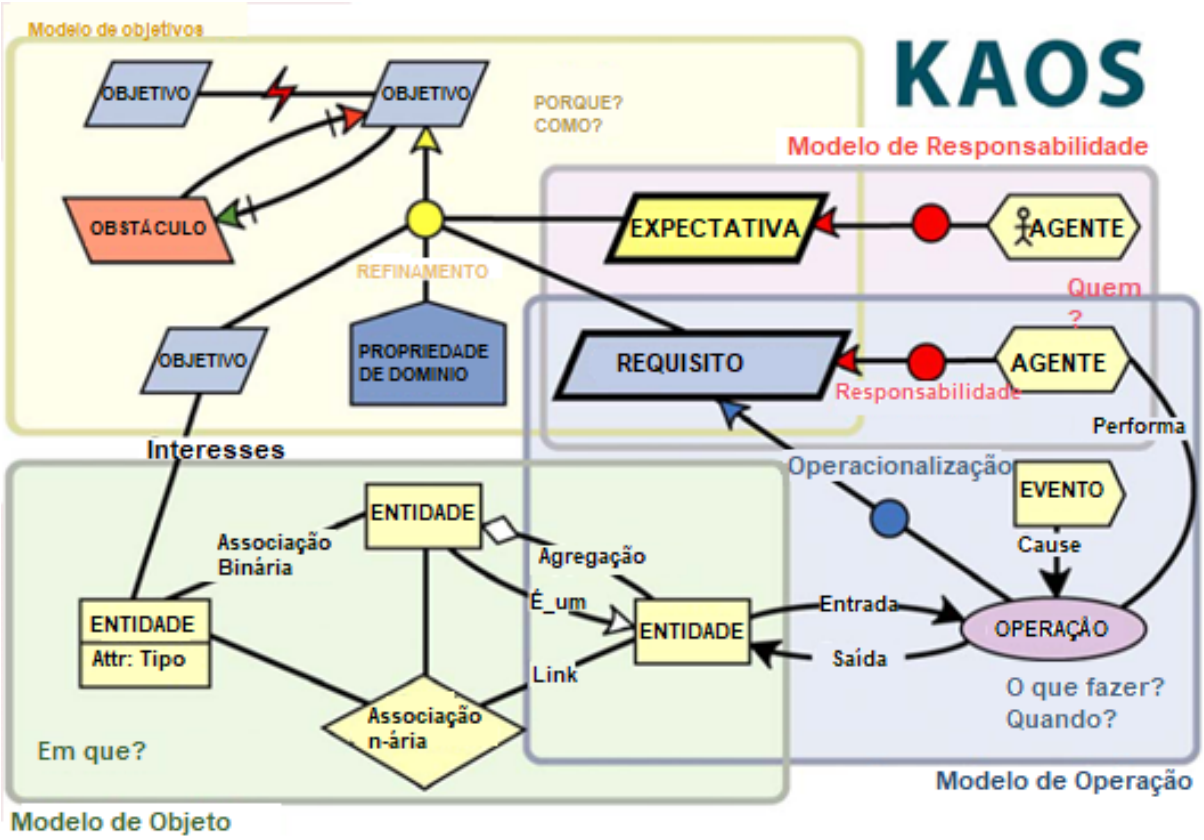


Figura 1 – Modelos de KAOS

Fonte:Adaptado (OBJECTIVER, 2007).

2.4 ISTAR 2.0

A linguagem iStar proposta por Yu (1995) foca na modelagem social de atores relacionados ao sistema e suas interdependências. Essa abstração se dá através da descrição do relacionamento entre atores. A linguagem tem sido usada na especificação de requisitos iniciais de variados sistemas. Ela atualmente está na sua versão 2.0 (DALPIAZ; FRANCH; HORKOFF, 2016) (veja tabela 2).

Tabela 2 – Tabela comparativa entre construtores de iStar e iStar 2.0.

Nós e links	iStar 1.0	iStar 2.0
Atores	Atores gerais, Papéis, posições e agentes	Atores gerais Papéis, agentes
Links de atores	*is-a* *is-part-of, plays, occupies, covers* *INS*	*is-a* *participates-in* -
Elementos intencionais	*Goal, task, resource* *Softgoal*	*Goal, task, resource* *Quality*
Links entre elementos intencionais	*Means-end, task decomposition* *Contribution*	*refinement* *Contribution* *Qualification, neededBy*

Os modelos em iStar (HORKOFF; YU, 2016; DALPIAZ; FRANCH; HORKOFF, 2016) podem ser de três tipos, à depender da visão desejada pelo analista: o modelo de Dependência Estratégica (do inglês *Strategic Dependency* - Dependência Estratégica (SD)) apresenta uma visão de cada ator no sistema e as dependências estratégicas entre eles. Um exemplo de modelo SD pode ser visto na figura 2. No diagrama SD apresentado por essa figura, pode-se visualizar um cenário de reembolso de passagens aéreas. O ator estudante depende da agência de viagens para comprar passagens e agendar sua viagem. O estudante depende ainda da universidade para ter o formulário online de viagem processado.

O modelo de Raciocínio Estratégico (do inglês *Strategic Rationale* - Raciocínio Estratégico (SR)) mostra todos os detalhes referentes ao modelo, podendo incluir todos os construtores nativos da linguagem, e apresentando dentro dos limites de cada ator o raciocínio estratégico. Um exemplo de modelo SR pode ser visto na figura 3. Esse modelo representa o cenário de reembolso de viagem estendido, definindo as possibilidades de agendamento da viagem por parte do estudante além das atividades que devem ser realizadas pela universidade à fim de processar o formulário enviado pelo estudando. Ainda pode-se visualizar a forma que a agência pretende reservar a viagem.

A terceira visão possível para a modelagem em iStar 2.0 é o modelo híbrido que permite uma combinação entre os modelos SD/SR, permitindo portanto alguns atores expandidos e outros não.

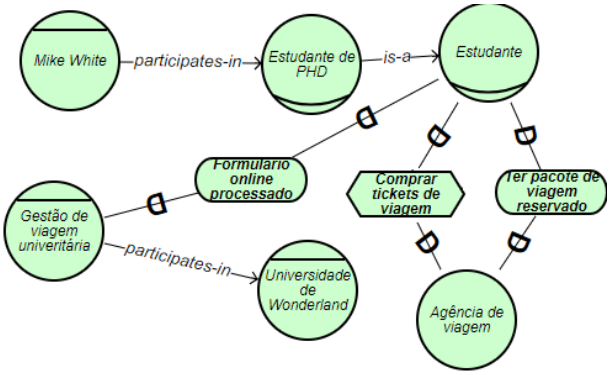


Figura 2 – Exemplo de modelo SD de iStar 2.0 - Cenário de reembolso de viagem

Fonte: Adaptado (DALPIAZ; FRANCH; HORKOFF, 2016).

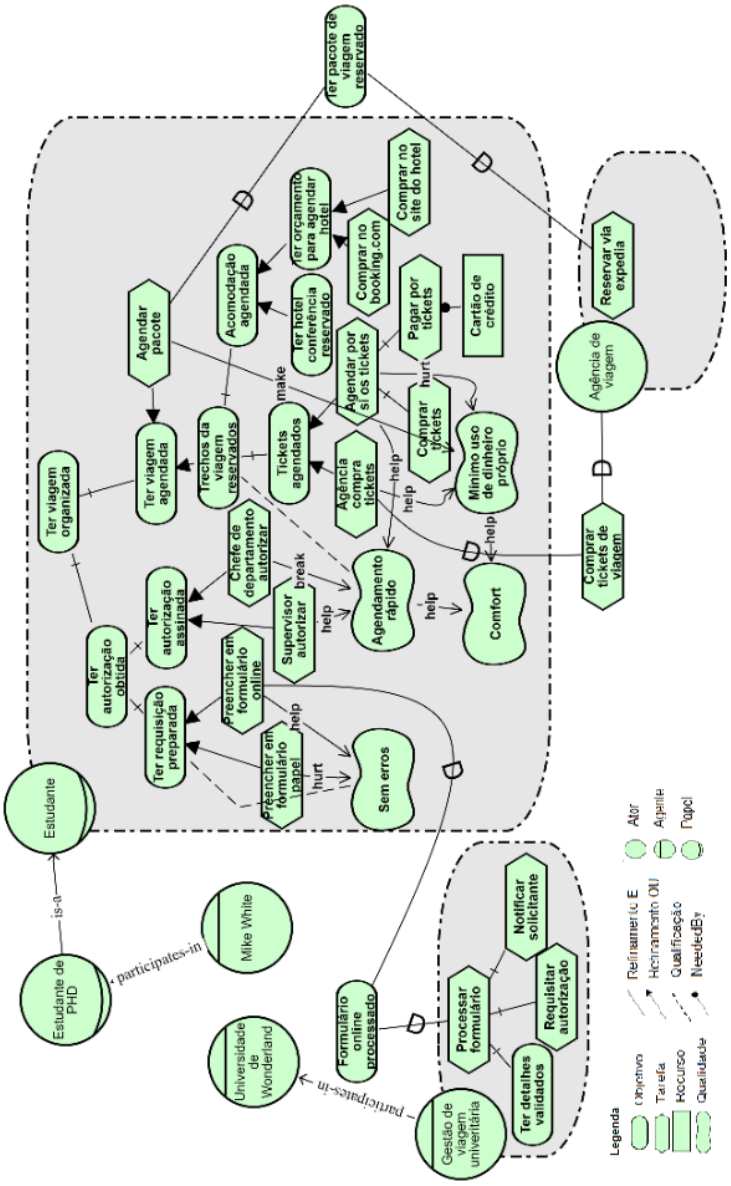


Figura 3 – Exemplo de modelo SR de iStar 2.0 - Cenário de reembolso de viagem

Fonte: Adaptado (DALPIAZ; FRANCH; HORKOFF, 2016).

Os atores de iStar, representados pela figura 4 podem ser de três tipos diferentes:

- **Ator**: Um ator geral. Deve ser usado quando nenhuma especificação sobre o ator é necessária ou desejada;
- **Agente:** Um ator com manifestações concretas, físicas; e
- **Papel:** Uma caracterização abstrata de um ator, posicionando-o dentro de um determinado contexto ou ambiente.

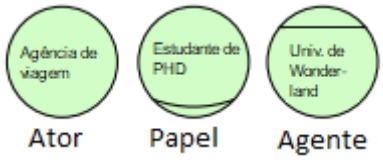


Figura 4 – Tipos de atores em iStar

Fonte: Adaptado (DALPIAZ; FRANCH; HORKOFF, 2016).

Atores podem ainda se inter-relacionar através dos links: ***participates-in*** ou ***is-a***. O link *is-a* representa a relação de generalização/especialização. Deve-se atentar ao fato que agentes não podem ser especializados, pois são instanciações concretas (DALPIAZ; FRANCH; HORKOFF, 2016). Já o link *participates-in* denota relações que não são generalizações/especializações. Todo ator pode ser relacionado à múltiplos atores através da relação *participates-in*.

Para denotar o caráter social e de intencionalidade da linguagem, elementos intencionais estão presentes como protagonistas em iStar. Esses elementos podem aparecer dentro e fora dos limites do ator, sendo eles:

- **Objetivo (*Goal*):** Representa o conjunto de objetivos que o ator deseja alcançar e tem critérios claros de satisfação;
- **Qualidade (*Quality*):** É um atributo que denota um desejo de caráter não-funcional do ator e possui níveis de satisfação diferenciados. Através das qualidades é possível denotar a forma que se deseja alcançar algo e/ou o quanto aquela qualidade afeta determinado elemento;
- **Tarefa (*Task*):** As tarefas representam ações concretas que o ator quer que sejam executadas; e
- **Recurso (*Resource*):** Uma entidade física ou informacional que é desejada pelo ator à fim de obter a realização de uma tarefa.

A figura 5 representa cada um dos elementos intencionais.

Figura 5 – Tipos de elementos intencionais em iStar

Fonte: (Autora, 2018).

No modelo SR, os atores são representados juntamente à sua fronteira que possui construtores intencionais que combinados denotam as estratégias daquele ator.

Em iStar as relações entre atores são representadas por dependências. As dependências compreendem cinco elementos, a saber: (1) ***depender*** (2) ***dependerElmt*** (3) ***dependum*** (4) ***dependee*** e (5) ***dependeeElmt***.

A figura 6 representa a relação de dependência e seus elementos. Observe que o *depender* é o ator que deseja que o *dependee* satisfaça algo. Esse algo a ser satisfeito é representado pelo elemento *dependum*, que pode ser qualquer um dos quatro elementos intencionais apresentados anteriormente (objetivo, qualidade, tarefa ou recurso). O *dependerElmt* trata-se do elemento intencional dentro do limite do ator *depender* de onde a relação de dependência se inicia. Já o *dependeeElmt* é o elemento intencional presente no ator *dependee* que representa como o ator irá satisfazer à relação de dependência.

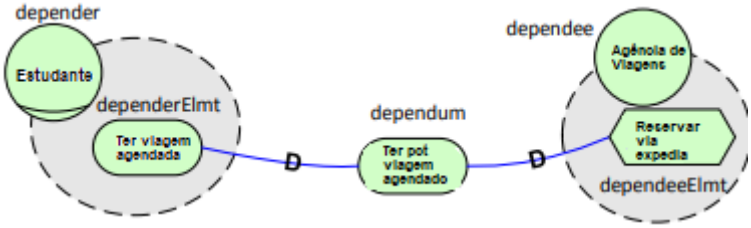


Figura 6 – Elementos de uma relação de dependência em iStar.

Fonte: Dalpiaz, Franch e Horkoff (2016).

Os elementos intencionais podem se relacionar e links existem para atender à esse fim. A tabela 3 mostra os links da linguagem iStar.

Tabela 3 – Relação links x construtores de iStar

		Ponta da seta apontando para			
		Goal	*Quality*	*Task*	*Resource*
Links começando de	Objetivo	Refinamento	Contribuição	Refinamento	n/a
	Qualidade	Qualificação	Contribuição	Qualificação	Qualificação
	Tarefa	Refinamento	Contribuição	Refinamento	n/a
	Recurso	n/a	Contribuição	*NeededBy*	n/a

O link de **refinamento** liga os elementos **tarefa** e **objetivo**, através de uma estrutura hierárquica. Nesse relacionamento, um elemento pai pode ter até N filhos, e pode participar de somente um relacionamento de refinamento. Os refinamentos possíveis são do tipo **E** do tipo **OU**. Quando elementos são refinados por relacionamentos **E**, todos os seus filhos devem ser atendidos para que o pai se concretize. Quando o relacionamento é do tipo **OU**, se ao menos um filho for atendido, o pai é atendido. Em um relacionamento de somente um filho, usa-se o refinamento **OU**. A figura 7 mostra exemplos de refinamentos possíveis em iStar.

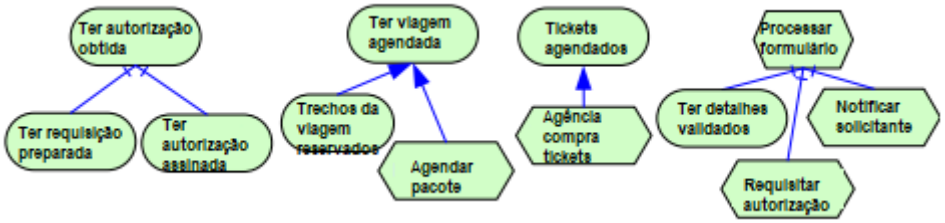


Figura 7 – Exemplos de links de refinamento

Fonte: (DALPIAZ; FRANCH; HORKOFF, 2016).

Já o link **NeededBy** relaciona uma tarefa à um recurso indicando portanto, o ativo necessário para que o ator realize a tarefa (DALPIAZ; FRANCH; HORKOFF, 2016). O link **neededBy** pode ser visto na figura 8.

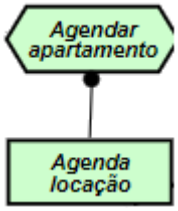


Figura 8 – Exemplos de relacionamento *neededBy*.

Fonte: (Autora, 2018).

Os links de contribuição, apresentados na figura 9, ligam os elementos intencionais às qualidades, indicando qual o nível de contribuição do primeiro para a satisfação do segundo. Essa contribuição pode ser de quatro tipos, a saber:

- *Help:* Indica que o construtor auxilia de forma fraca a satisfação da qualidade.
- *Make:* Indica que o construtor satisfaz suficientemente à qualidade à qual contribui;
- *Hurt:* Indica que o construtor é capaz de impedir de forma fraca a satisfação da qualidade; e
- *Break:* Indica que o construtor impede suficientemente a satisfação da qualidade.

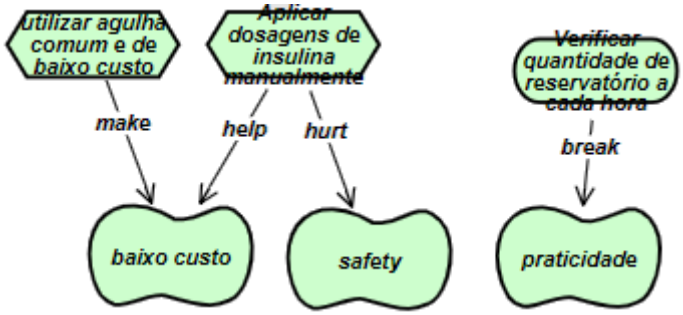


Figura 9 – Exemplos de cada tipo de contribuição em iStar.

Fonte: (Autora, 2018).

Quando deseja-se que um construtor seja realizado atendendo à determinada qualidade, diz-se que a qualidade **qualifica** o referido construtor. A figura 10 apresenta um exemplo em que a tarefa de *aplicar dosagem de insulina* deve ser feita *sem erros*.

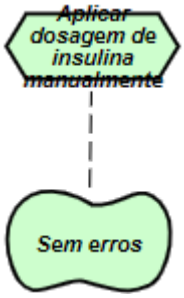


Figura 10 – Exemplo de relacionamento de qualificação.

Fonte: (Autora, 2018).

Nesta dissertação nos basearemos na linguagem iStar para propor uma extensão que seja adequada para a modelagem de Sistemas Críticos de Segurança.

2.4.1 Estendendo a Linguagem iStar

A linguagem iStar é bastante usada em conjunto com outros formalismos, devido a alguns fatores como: necessidade de conectar a engenharia de requisitos iniciais (*early requirements engineering*) à engenharia de requisitos finais (*later requirements engineering*), para customizar a linguagem à algum domínio específico (MORANDINI et al., 2017; MELLADO; MOURATIDIS; FERNáNDEZ-MEDINA, 2014; MOURATIDIS; GIORGINI, 2007; MOURATIDIS et al., 2013; GHANAVATI; AMYOT; RIFAUT, 2014) ou para combinar a linguagem com outros paradigmas que a tornem mais eficiente, entre outros. A revisão sistemática da literatura realizada por Gonçalves et al. (2018a) identificou 96 extensões de iStar e 307 construtores criados, sendo as áreas sociais e de sistemas inteligentes com mais extensões definidas.

Apesar de que extensões tem sido propostas na literatura à fim de possibilitar a construção de modelos iStar que contemplem áreas específicas, um ponto preocupante é que a maioria das extensões dessa linguagem não são criadas com o intuito de integrar-se à

trabalhos da comunidade da mesma, fato que pode atrapalhar sua difusão e utilidade (FRANCH et al., 2011; GONÇALVES; ARAÚJO; CASTRO, 2018). Sendo assim, deve-se buscar formas de permitir que as extensões criadas possam ser visíveis e disponíveis para profissionais de requisitos e comunidade iStar.

Estudos realizados por Gonçalves et al. (2018c) entre pesquisadores que criaram extensões para a linguagem iStar, mostram que representações gráficas escolhidas devem ser simples a ponto de serem desenhadas à mão e, devem seguir o padrão da linguagem nativa à fim de não destoar de sua forma de representação (DALPIAZ; FRANCH; HORKOFF, 2016).

Neste trabalho, pretende-se que a nova linguagem proposta seja o mais simples possível.

Extensões, assim como as linguagens que as mesmas estendem, devem ter seus metamodelos e sintaxes abstratas bem definidas. Explica-se, na próxima seção, abaixo cada um desses termos.

2.4.2 Metamodelagem

De acordo com Kelly e Tolvanen (2007) uma linguagem de modelagem deve ser formalizada através do uso de um metamodelo. Metamodelos são usados para descrever a estrutura e dados dos modelos, mapeando os conceitos da área de aplicação, suas propriedades e conexões e especificando seus links e funções de seus elementos (GONÇALVES et al., 2018c). O padrão MOF da OMG pode ser utilizado para esse fim.

Meta-Object Facility - Facilidade Meta-Objeto (MOF), trata-se de um padrão da OMG para a engenharia dirigida à modelos. Atualmente encontra-se na versão 2.4.1 (OMG, 2013). Através do padrão MOF, é possível definir e estender novos e existentes metamodelos além de modelos de infraestrutura de software. Conceitos padrão da modelagem de classes da UML são usados para extensibilidade de MOF 2.0. No contexto do padrão, uma classe de objeto representa a natureza do mesmo (OMG, 2013).

2.4.3 Sintaxe concreta

É uma prática recomendada que os construtores à serem criados para a sintaxe concreta, atendam aos princípios de claridade semiótica, proposto por Moody, Heymans e Matulevičius (2010) e baseado na teoria de símbolos de Goodman (1968). Esse princípio visa garantir que exista uma correspondência de 1:1 entre construtores semânticos e símbolos gráficos, evitando assim que qualquer uma de quatro anomalias abaixo ocorra (MOODY; HEYMANS; MATULEVIČIUS, 2010; GONÇALVES et al., 2018c):

- Deficiência de símbolo: Quando um construtor semântico não tem símbolo o representando;

- Redundância de símbolo: Quando um construtor semântico é representado por mais de um símbolo;
- Sobrecarga de símbolo: Quando o mesmo símbolo é usado para representar múltiplos construtores; e
- Excesso de símbolo: Quando um símbolo não representa qualquer construtor semântico.

Para a proposta da nova linguagem de modelagem adotaremos o processo PRISE proposto por Gonçalves Enyo (2019)[1].

2.5 PROCESSO PRISE

O PRISE - A PRocess to conduct iStar Extensions - (GONÇALVES et al., 2018b; GONÇALVES, 2018) visa definir um processo para criação de extensões para a linguagem iStar. Esse processo adota as melhores práticas para criação de extensões de iStar propostos pela comunidade da linguagem (GONÇALVES et al., 2018c). Estas boas práticas foram agrupadas em forma de 9 (nove) diretrizes:

- D1 - Preservar a sintaxe original da linguagem iStar;
- D2 - Desenvolver extensões consistentes, completas e sem conflitos e seguir um processo/método para criá-las;
- D3 - Realizar uma revisão da literatura, consultar especialistas no domínio e em iStar e modelar sistemas da área de aplicação antes de estender a linguagem;
- D4 - Descrever uma definição clara dos conceitos que irão ser inseridos na extensão;
- D5 - Propor as sintaxes concreta e abstrata da extensão;
- D6 - Checar a consistência entre as sintaxes abstrata e concreta;
- D7 - Relacionar conceitos introduzidos pela extensão com os construtores nativos de iStar;
- D8 - Definir extensões com o menor número possível de modificações e novas representações à fim de não complicar o uso da linguagem de modelagem iStar; e
- D9 - Propor representações gráficas simples e meticulosas, aptas à serem desenhadas no papel sem o auxílio de uma ferramenta.

[1] Artigo sob revisão.

A figura 11 apresenta uma visão geral do processo PRISE.

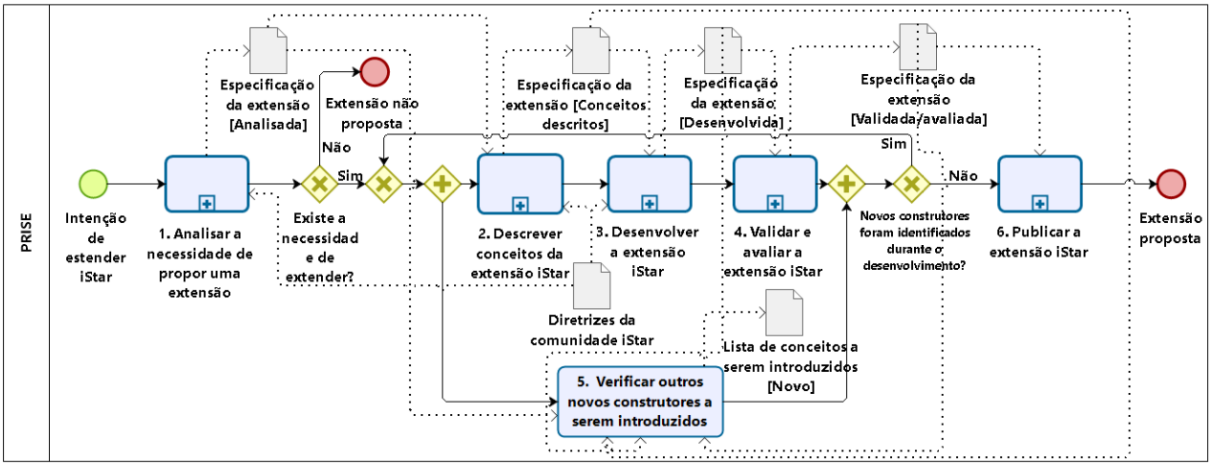


Figura 11 – Processo PRISE - Visão Geral

Fonte: Adaptado de SITE PRISE (GONÇALVES, 2018).

O passo inicial "Analisar a necessidade de propor uma extensão", visa analisar se há uma real necessidade de propor uma extensão para a área escolhida. Assim, no nosso caso, a área de segurança deve ser estudada, seus conceitos definidos, especialistas contactados e realizada uma busca por extensões que já atendam à segurança. Além disso, o subprocesso preconiza a necessidade de modelar um exemplo de um sistema que possua os conceitos desejados, à fim de validar se a linguagem nativa realmente não suporta a sua modelagem. Por fim, é definido se a criação da extensão é necessária.

O próximo passo "Descrever conceitos da extensão iStar", trata-se de um subprocesso para descrever o significado dos conceitos de segurança obtidos no subprocesso anterior. Nesse momento, uma análise de construtores a serem reusados, tanto de extensões já criadas como de construtores nativos da linguagem iStar é realizada, a fim de atender à guias propostos pela própria comunidade da linguagem e elencados em Gonçalves et al. (2018c).

No subprocesso 3 "Desenvolver a extensão iStar", o metamodelo e as regras de validação da linguagem serão criados. Além disso, é desenvolvida também a sintaxe concreta da extensão. Uma ferramenta pode ser proposta aqui à fim de auxiliar na aplicabilidade da nova extensão.

Já o subprocesso 4 diz respeito à "Validar e avaliar a extensão iStar". Nesse momento a extensão será usada para modelar um sistema não-trivial, que atenda e possua os conceitos que fazem parte da extensão. Após consultas à especialistas e melhorias aplicadas, a extensão deverá ser validada através de um método empírico.

O subprocesso 5, "Verificar outros novos construtores a serem inseridos"não foi executado, devido à não terem sido identificados novos construtores após a avaliação e validações da extensão.

Por fim, o subprocesso 6 "Publicar a extensão iStar"diz respeito a publicação da extensão no catálogo de extensões de iStar. Tal catálogo é apresentado na próxima seção (GONÇALVES et al., 2018b).

2.5.1 Catálogo de Extensões CATIE

Um catálogo de extensões de iStar, CATIE, foi proposto por Gonçalves et al. (2018b), com o intuito de reunir extensões criadas e seus construtores, figurando-se como um importante repositório para auxiliar na criação de extensões para iStar.

O catálogo parte de uma pesquisa realizada pelos autores à fim de investigar as melhores formas de propor extensões de iStar (GONÇALVES et al., 2018c). O catálogo possibilita a verificação de atendimento ao princípio de claridade semiótica (MOODY; HEYMANS; MATULEVIČIUS, 2010).

O CATIE apresenta uma lista de extensões já propostas em iStar. Os construtores de cada extensão são também disponibilizados e detalhados, incluindo seu conceito, área de aplicação e imagem. Os conflitos encontrados nas extensões já existentes são apresentados no menu **conflitos**. Por fim, o catálogo permite que novas extensões sejam reportadas.

Para adoção da nova linguagem proposta por esse trabalho será importante ter o apoio de uma ferramenta de modelagem que já implemente a linguagem nativa, iStar 2.0. Nesse sentido, apresenta-se na próxima seção uma ferramenta de modelagem para modelos iStar 2.0.

2.6 FERRAMENTA PISTAR

A ferramenta piStar (PIMENTEL; CASTRO, 2018) foi desenvolvida com o intuito de gerar os modelos (SR, SD e híbridos) de iStar 2.0, atendendo às regras de validação e disponibilizando todos os construtores e links da linguagem.

Um dos objetivos de piStar é prover um ambiente que possibilite a inserção de novas extensões, de forma prática. Por isso, a ferramenta é considerada por seus autores, uma plataforma para facilitar o desenvolvimento de iStar. Para esse fim, o código fonte e a documentação da ferramenta foram disponibilizados em um repositório de código aberto.

A figura 12 mostra a interface da ferramenta Pistar. Na figura, podemos ver que a parte **A**, representa o palete de modelagem, a parte **B** é a área de desenho, onde podem ser inseridos os construtores presentes no palete de modelagem, possibilitando a construção dos modelos. Já a parte **C** da figura mostra as propriedades de um construtor que tenha sido previamente selecionado. A parte **D**, por fim, apresenta o menu principal da ferramenta, onde os modelos podem ser salvos (em formato ".txt"ou de imagem), ou abertos, através de arquivos ".txt"já salvos previamente.

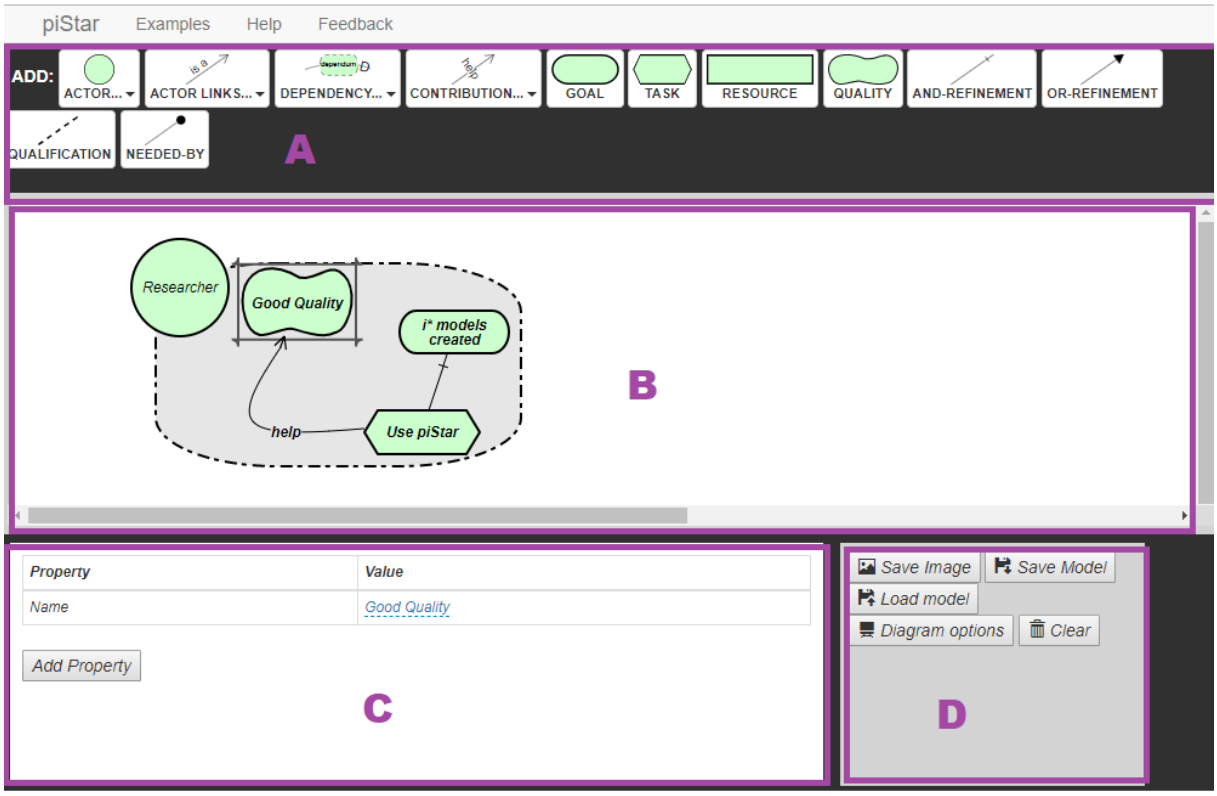


Figura 12 – Interface da ferramenta Pistar.

Fonte: Adaptado (PIMENTEL; CASTRO, 2018).

Nesta dissertação, a nova linguagem de modelagem de requisitos de segurança, iStar4Safety, será apoiada por uma versão estendida da ferramenta piStar: PiStar-4Safety.

2.7 CONCLUSÃO

O capítulo 2 apresentou uma revisão bibliográfica dos conceitos gerais usados por esse trabalho. Primeiramente foi apresentado o Sistema de Bomba de Infusão de Insulina e seus requisitos que serão usados para ilustrar exemplos necessários ao entendimento dessa dissertação. O próximo tema tratado foi Engenharia de Requisitos Orientada à Objetivos e as duas linguagens GORE usadas por esse trabalho, KAOS e iStar. O processo PRISE foi apresentado na seção 2.5 e por fim, a ferramenta piStar, que apoiou a modelagem usada na dissertação, completou a revisão da literatura. No próximo capítulo, apresenta-se os conceitos relacionados à segurança.

3 SEGURANÇA

Nesse capítulo serão apresentados os principais conceitos relacionados à segurança usados nesse trabalho.

Segundo Firesmith (2004), segurança é o grau no qual danos acidentais são apropriadamente tratados seja prevenindo-os, identificando-os, reagindo a eles ou se adaptando à presença de tais danos.

Analisar a segurança de Sistemas Críticos de Segurança é antecipar acidentes e suas causas através da identificação de perigos que possam ocorrer durante todo o ciclo de vida do sistema, na busca de eliminá-los ou, no pior caso, controlá-los. Utilizar dados históricos e experiências prévias obtidas no desenvolvimento de outros sistemas para o desenvolvimento de sistemas críticos, se mostra menos útil do que esperado, devido a maior especificidade tanto dos sistemas quanto dos acidentes que podem acontecer. Tendo em vista a criticidade das interfaces, é difícil pensar que os comportamentos de tais interações entre componentes aconteça de igual forma em outros sistemas (LEVESON, 1995). Segurança precisa estar inserida em todo processo de desenvolvimento de sistemas (LEVESON, 2011).

A Análise de segurança de sistemas é divida em quatro estágios, à depender do momento em que são realizadas e dos seus objetivos (LEVESON, 1995). São elas:

- Análise Preliminar de Segurança (Preliminary Safety Analysis - PSA): Também chamada de Análise Preliminar de Perigos. Trata-se de um processo iterativo. Essa análise é usada nos estágios iniciais do ciclo de vida para identificar funções críticas do sistema e perigos gerais relacionados à ele;

- Análise de Segurança do sistema (System Safety Analysis - SSA): Essa análise é usada para refinar e definir causas para os perigos encontrados na Análise Preliminar de Segurança. Na SSA[1], respostas quanto à comportamentos dos requisitos de segurança em relação ao sistema integrado serão consideradas;

- Análise de Segurança do Subsistema (Subsystem Safety Analysis - SSSA): Trata-se de uma sub-análise da SSA, preocupando-se em examinar como os comportamentos dos componentes, ou subsistemas, irão afetar a segurança do sistema no geral; e

- Análise de Segurança de Operação e Suporte (Operating and Support Safety Analysis - OSSA): Nessa análise são identificados perigos e procedimentos de redução de riscos durante as fases de operação do sistema e suporte ao mesmo. Essa fase dura até o fim da vida útil do sistema.

[1] O SSA também é encontrado na literatura como SHA (Análise de Perigos de Segurança - Safety Hazard Analysis), veja Leveson (1995).

A documentação gerada pelo sistema, em especial a análise de perigos, precisa ser constantemente atualizada. Leveson (2011) chama a atenção ao fato de que acidentes que poderiam ter sido prevenidos, frequentemente ocorrem devido à mudanças no sistema ou no seu ambiente que não atendem ou até violam premissas definidas na Análise Preliminar de Segurança. A documentação relativa à segurança deve estar sempre atualizada e precisa ser utilizada durante todo o processo de desenvolvimento indicando objetivos e requisitos de segurança que devem ser priorizados e atendidos à fim de que seja construído um sistema seguro.

Além disso, durante o processo de análise de segurança de um Sistema Crítico de Segurança, é importante o envolvimento dos seus *stakeholders* principalmente no processo de avaliação da segurança. Logo, o método aplicado para definir esses requisitos deve ser fácil de aprender, entender e aplicar (STÅLHANE; MYKLEBUST, 2016).

Nas próximas seções detalhamos mais sobre requisitos de segurança, que são o foco da modelagem de iStar4Safety. Além disso, os principais conceitos de segurança, incluindo os conceitos base usados para definição dos construtores da extensão, estão listados e conceituados na seção 3.2.

3.1 SISTEMAS CRÍTICOS E SEUS REQUISITOS DE SEGURANÇA

Um software crítico é "[...]qualquer software que possa, direta ou indiretamente contribuir para a ocorrência de um estado perigoso do sistema."(LEVESON, 1995, p.156).

O software, sendo um componente do sistema, não é por si só perigoso mas pode contribuir para o aumento da segurança do sistema ou mesmo colocá-lo em um estado perigoso (BROOMFIELD; CHUNG, 1997; LUTZ, 2000). A engenharia de software para Sistemas Críticos de Segurança requer um especial entendimento sobre o papel do software no sistema e de todas as suas interações dentro do mesmo, ou seja, suas interfaces com outros sistemas.

Para tratar de requisitos de Sistemas de Segurança Críticos é necessário um cuidado especial, devido à especificidade dos mesmos quanto à necessidade de evitar comportamentos indesejados.

Segundo Leveson (2011) as linguagens de especificação auxiliam na resolução de problemas, permitindo o (1)raciocínio sobre propriedades particulares, (2) possibilidade de construção de um software que atenda ao sistema do qual fará parte e (3) para validar se as qualidades pretendidas para o software estão sendo atendidas. Documentar e rastrear perigos e suas soluções é requisito básico para qualquer programa de segurança.

A elicitação de requisitos para Sistemas Críticos é um processo bastante oneroso, devido à necessidade de capturar os comportamentos de todos os subsistemas envolvidos e lidar com todas as restrições definidas (BROOMFIELD; CHUNG, 1997). Somado à isso, muitas evidências apontam que erros relacionados à segurança estão mais relacionados à erros em requisitos do que na codificação (LEVESON, 1995).

A maioria das especificações de requisitos são incompletas tendo em vista que não especificam requisitos para eliminar perigos mais improváveis ou mitigar suas consequências. Além disso, são incompletas devido à tipicamente não considerar todos os possíveis eventos e combinações de eventos possíveis bem como todas as interações possíveis entre seus componentes. Somado a isso, por mais que fosse possível cobrir todos os eventos na especificação de requisitos, ainda sim pode-se ter requisitos ambíguos, incorretos ou implementados inconsistentemente (FIRESMITH, 2004).

A Engenharia de Requisitos tem um importante papel durante o processo de certificação, de um Sistema Crítico de Segurança. Contudo, seu custo é bem alto no caso de sistemas críticos (LUTZ, 2000).

Tanto as abordagens de Engenharia de Software, como as da Engenharia de Segurança devem ser tratadas no processo de Engenharia de Requisitos, permitindo a participação ativa e comunicação entre os diversos profissionais interessados no desenvolvimento de software seguro e confiável (BROOMFIELD; CHUNG, 1997). Porém, é importante salientar a dificuldade na interação entre as áreas de segurança. A busca por uma linguagem comum entre os profissionais dessas áreas é tratada como um desafio. Leveson (1995) afirma que definir uma terminologia comum em algum campo é difícil porém de grande importância, já que facilita a comunicação entre os envolvidos, o que é um fator crucial para o sucesso de qualquer empreitada. Portanto, a próxima seção apresenta um conjunto de conceitos à fim de permitir uma maior integração no entendimento de termos relacionados à segurança.

É importante salientar que todo Sistema Crítico de Segurança controlado por software precisa ser certificado quanto à segurança (MARTINS; GORSCHEK, 2016). Existem diversos padrões internacionais preocupados com o desenvolvimento de software crítico. O padrão RTCA DO-178, por exemplo, é o padrão internacional de facto usado para certificação de softwares para programas aeroespaciais militares e comerciais (ZOUGHBI; BRIAND; LABICHE, 2007; Bowen; Stavridou, 1993). Outro padrão internacional, o IEC 61508-3, lida com a segurança funcional de sistemas eletrônicos.

3.2 CONCEITOS DE SEGURANÇA

Leveson (1995) traz luz ao fato de que as terminologias usadas entre os envolvidos no desenvolvimento de um sistema crítico devem ser equivalentes, visto que a falta de uma comunicação comum entre profissionais de diferentes tipos pode ser um gargalo nesse processo. Vilela et al. (2017b), Vilela et al. (2017a) apresentam um conjunto de definições importantes a serem utilizadas durante o desenvolvimento de sistemas críticos, mais especificamente cobrindo os conceitos necessários para a criação do documento da Análise Preliminar de Segurança, que será o escopo de modelagem da extensão proposta. Alguns conceitos, dentre eles os que serão utilizados para a criação da extensão, estão apresentados na próxima seção.

3.2.1 Segurança *(Safety)*

O termo Segurança *Safety*, é definido como a liberdade de acidentes ou perdas. Leveson (1995) aponta que apesar de algumas definições definirem *safety* como um subconjunto de *security* ou confiabilidade, ela é uma qualidade distinta que deve ser tratada separadamente, apesar das interações e características em comum entre essas qualidades.

3.2.2 Sistema de Software Seguro (Safe Software System)

Um sistema de software seguro é aquele software que irá executar como um componente de um sistema sem contribuir para que perigos aconteçam.

É importante salientar que o software por si só não tem a propriedade de ser seguro ou inseguro. A segurança do software deve ser considerada dentro do contexto no qual o sistema está sendo desenvolvido. Isso também se justifica por segurança ser um propriedade emergente, logo deve ser pensada de forma holística (LEVESON, 1995).

3.2.3 Software Crítico de Segurança *(Safety Critical Software)*

Um software crítico de segurança é aquele que pode de forma direta ou indireta contribuir para que o sistema entre em um estado perigoso (LEVESON, 1995). O software, mesmo não controlando dispositivos críticos de hardware, pode atuar como provedor de dados críticos usados em tomada de decisões por *stakeholders* (FIRESMITH, 2004).

3.2.4 Análise Preliminar de Segurança (*Preliminary Safety Analysis*)

Apresenta-se nessa seção, a definição de conceitos importantes à modelagem de requisitos iniciais de Sistemas Críticos de Segurança (VILELA et al., 2017b). Ressalta-se que as definições não tem o intuito de cobrir todas as questões relacionadas à segurança de sistemas, mas clarificar os conceitos relevantes para o entendimento desta dissertação.

3.2.4.1 Acidente (*Accident*)

Um acidente é um evento não planejado e nem desejado, mas não necessariamente não esperado, que resulta em, no mínimo, um nível específico de perda (LEVESON, 1995; BERRY, 1998). O critério para especificar eventos como acidentes é que as perdas com tal evento são tão importantes que precisam ter um papel central no processo de desenvolvimento e na definição de compromissos. Ainda segundo Leveson (1995), é importante definir um evento de acidente pois ele irá influenciar as abordagens usadas para aumentar a segurança do sistema, possibilitando a definição de ações de prevenção e medidas corretivas. Um acidente pode ou não ser previsto.

Os acidentes são afetados pelo ambiente no qual estão presentes. Tomando como exemplo, na bomba de insulina, o perigo "bomba sem proteção de descarga elétrica", um aci-

dente poderia ser: "Paciente sofre descarga elétrica". Caso o acidente aconteça, o paciente vai sofrer danos indesejáveis.

Quando um perigo se concretiza, e por algum motivo, não há perdas ou as perdas são danos desconsideráveis em relação ao que se esperava, chama-se o evento de incidente ou quase perda. No exemplo da descarga elétrica, caso a descarga liberada não tivesse sido nem sentida pelo paciente, a situação não se configuraria como um acidente, mas sim como um incidente.

3.2.4.2 Perigo (*Hazard*)

Um perigo trata-se de "[...]um estado ou conjunto de condições de um sistema (ou objeto) que, junto com outras condições no ambiente do sistema (ou objeto) irá inevitavelmente levar à um acidente (evento de perda)."Leveson (1995, p.177). O acidente é consequência de um perigo. O ambiente no qual o sistema faz parte influencia na definição de perigos e o limite de até onde os perigos serão tratados deve existir, apesar de que o estado mais seguro seria modelar e lidar com todo e qualquer perigo relacionado ao sistema (LEVESON, 1995). Como exemplo de perigo, tem-se a falha do sistema de entrega de insulina. Tal perigo pode levar ao acidente de superdosagem de insulina, que pode levar o paciente à morte.

Um perigo deve conter somente os elementos suficientes e necessários para que um acidente ocorra. O acidente de superdosagem de insulina, tem como perigos: fluxo livre de insulina, configuração de insulina maior que o permitido, aplicações excessivas de *bolus* e falha no sistema de entrega. Cada uma dessas situações são suficientes para a ocorrência do acidente. Para o escopo desse trabalho, será considerado que o exemplo apresentado representa todas as possibilidades para tal acidente.

Para a identificação de perigos, que não é um processo trivial, várias técnicas podem ser usadas. Ericson et al. (2015) apresenta algumas boas práticas para ajudar nesse reconhecimento:

1. Utilizar os componentes do chamado triângulo de perigos:
    a) Utilizar checklists de componentes/elementos perigosos.
    b) Avaliar eventos de disparo e causas de perigos.
    c) Avaliar ameaças e acidentes possíveis.
2. Utilizar conhecimento adquirido em experiências passadas.
3. Analisar boas práticas de desenvolvimento.
4. Revisar conceitos de desenvolvimento geral de segurança.
5. Revisar e analisar as causas de perigos, de perigos genéricos.

6. Questões-chave quanto à estados de falha.

7. Avaliar e refinar acidentes com uma abordagem de mais alto-nível e funções críticas.

Um perigo é definido pela combinação do nível do perigo combinado com a exposição ao perigo mais a probabilidade do perigo levar à um acidente.

Os perigos podem ser tratados de quatro formas, organizadas abaixo por ordem de precedência, sendo a primeira a forma mais segura de lidar com o perigo, ou seja, eliminá-lo(LEVESON, 1995; BERRY, 1998):

- Eliminação do perigo;
- Redução do perigo;
- Controle do perigo; e
- Minimização de danos causados pelo perigo.

A identificação do perigo é importante a fim de definir modos de mitigá-lo. A estratégia mais segura para mitigar o perigo é eliminar as possibilidades do mesmo ocorrer. É imprescindível, portanto, que os perigos sejam encontrados e modelados no início do desenvolvimento, ou seja, na fase de definição de requisitos iniciais (BERRY, 1998).

3.2.4.3 Causa do Perigo (*Cause of Hazard*)

Uma causa do perigo é representada por uma condição que sozinha ou associada à outras, é/são suficiente(s) para o perigo relacionado à ela(s) ocorrer. As causas do perigo podem ser controladas ou até eliminadas em alguns casos (BERRY, 1998; VILELA et al., 2017b; LEVESON, 1995). No trabalho de Vilela et al. (2018), encontra-se uma classificação de quatro tipos de causas para um perigo, a saber:

- Perigo procedural (*Procedural Hazard*): Relacionado ao processo e como essas ações usuais ou não de realizar atividades pode causar perigos. Ex.: Compartilhamento da bomba. O compartilhamento pode ser considerado usual, mas pode levar o paciente à uma acidente, ser infeccionado.

- Perigo de interface (*Interface Hazard*): Devido à sua importância, a análise de interfaces é um dos componentes básicos do processo de Engenharia de Sistemas. A interface entre componentes é tida como grande responsável por erros relacionados à segurança do sistema. Os vários componentes de um sistema (Software, Hardware, processos, dados, pessoas), estão interconectados e segurança é uma propriedade emergente. Portanto a segurança precisa ser analisada também nas interações entre os componentes, que é um dos focos de erros relacionados à mesma. Um exemplo de uma causa do tipo perigo de interface para a bomba de insulina, seria uma falha da leitura de algum sensor físico.

- Fator humano (*Human Factor*): Leveson (1995) dedica dois capítulos, o cinco e seis deste livro, à explanações sobre humanos e sua relação com perigos. Considerando o ser humano como um componente do sistema, sua interação com os outros componentes trata-se de uma interface, que é tida como uma grande fonte de erros relacionados à segurança (LUTZ, 1993; LEVESON, 2011). Um exemplo de fator humano poderia ser, na bomba de infusão de insulina: "Paciente configura valor alto de insulina", que pode causar o acidente de overdose de insulina. Deve-se atentar ao fato de que associar acidentes à erros humanos é de fato muito comum Contudo deve ser feito com cuidado, pois corre-se o risco de ter como único objetivo a busca de um "culpado", principalmente por questões legais e sociais, perdendo assim de vista a real necessidade de uma ação mais segura que, nesse caso, é a compreensão do motivo do acidente para aprender com ele e evitar novos eventos indesejados. Associar o humano como causa, leva por vezes à ignorar a característica de multicausalidade de um acidente. No exemplo de acidente citado acima, poderiam ter sido desenvolvidos mecanismos no projeto para não permitir que o paciente ultrapassasse o valor adequado de dose de insulina, ou outra solução que tornasse durante seu desenvolvimento, o dispositivo mais seguro (LEVESON, 1995).

- Perigo Ambiental (*Environmental Hazard*): Causas de perigos classificadas como perigos ambientais são aquelas relacionadas a perigos no ambiente que circunda o sistema e que são uma ameaça ao mesmo. A porção do ambiente a ser considerada deve levar em conta o limite estabelecido para o sistema. Um exemplo de perigo ambiental poderia ser a precipitação de chuva, que faria o dispositivo ficar em contato com água dando margem ao acidente de descarga elétrica (ZHANG; JONES; JETLEY, 2010).

- Causa do Sistema (*System Cause*): Uma causa de perigo pode ser um evento gerado pelo próprio sistema. Tendo em mente que o sistema é mais que a soma de seus componentes, as causas do sistema podem ser especializadas em:

  - Falha (*Failure*): Falha é a não-realização ou inabilidade do sistema realizar uma função que foi definida para o mesmo, sobre o tempo e condições ambientais definidas (LEVESON, 1995). Nem toda falha provoca um acidente ou perigo. Uma falha possui a propriedade de Probabilidade de Falha (*FailureProbability*) que pode assumir os valores de: (1) Frequente (*Frequent*), (2) Provável (*Probable*), (3) Ocasional (*Occasional*), (4) Remota (*Remote*) ou (5) Improvável (*Improbable*). A falha ainda pode ser de dois tipos: *Software* ou *Hardware*. Uma falha de *hardware*, por sua vez, pode ser mecânica ou eletrônica.
  - Sistema não executa comportamento (*SystemMisBehavior*): O comportamento à nivel de sistema não é executado, podendo então levar à um perigo quando outras condições necessárias acontecerem em conjunto.

Um exemplo, na bomba de insulina de causa de perigo de sistema, poderia ser a falha de software, que pode causar o perigo da bomba voltar inadvertidamente ao estado de fábrica.

3.2.4.4 Condições Ambientais (*Environmental Condition*)

Condições ambientais tratam-se de um conjunto de componentes e suas propriedades, incluindo elementos físicos, culturais, demográficos, econômicos, políticos, regulatórios ou tecnológicos que, apesar de não serem parte do sistema, podem afetar seu comportamento. Deve-se atentar ao fato de que muitas informações sobre o ambiente que circunda o sistema podem ser identificadas no início do desenvolvimento. Porém outras só poderão ser identificadas conforme o andamento do projeto onde novas decisões de desenvolvimento, características, e requisitos serão encontrados. Algumas dessas afirmações somente estarão claras após detalhadas decisões de desenvolvimento e análise de segurança (LEVESON, 1995; LEVESON, 2011; VILELA et al., 2017b). Na bomba de infusão de insulina, condições ambientais que podem interferir em segurança podem ser: chuva, ambiente molhado, ambiente sem recarga de insulina, falta de fonte de energia, etc.

3.2.4.5 Requisitos Funcionais de Segurança (*Functional safety Requirement*)

Requisitos funcionais de segurança são os requisitos usados para mitigar ou prevenir os efeitos de falhas identificadas na análise de segurança. Um exemplo de tal conceito na bomba de infusão de insulina seria a ação de "trocar reservatório"para mitigar o perigo do "Reservatório quebrado".

3.2.4.6 Restrição (*Constraint*)

Restrições se referem à qualquer decisão de engenharia que tenha sido selecionada para ser imposta como requisito, descrevendo assim como o software deve ser desenvolvido e implementado (VILELA et al., 2017b; FIRESMITH, 2004). As restrições de segurança de alto-nível sempre se originam de perigos identificados para o sistema (LEVESON, 2011). Morandini et al. (2017) ainda aponta que restrições adicionam um grau de incerteza ao desenvolvimento em questão. Uma restrição no Sistema de Infusão de Insulina poderia ser que o paciente não pode receber infusão de insulina, caso o reservatório esteja sendo trocado.

3.2.4.7 Obstáculo (*Obstacle*)

Obstáculos são comportamentos ou outros objetivos que bloqueiam ou previnem a satisfação de um dado objetivo. Definem comportamentos excepcionais e indesejados para o alcance de objetivos. Denotam a razão pela qual um objetivo falha e devem ser refinados, com refinamentos do tipo e/ou. Obstáculos são classificados como duais dos objetivos.

Pela dificuldade em especificar o que não é desejado em detrimento do que é desejado, o indicado é que obstáculos sejam identificados à partir dos refinamentos dos objetivos, ou seja, suas folhas. (VILELA et al., 2017b; LAPOUCHNIAN; LESPÉRANCE, 2006; ANTON, 1996; LAMSWEERDE; LETIER, 2000). Em Sistemas Críticos de Segurança, obstáculos são identificados como perigos já que estes impedem que o objetivo crítico seja alcançado.

3.2.4.8 Pré/Pós-condição (*Pre/Post condition*)

Uma pré-condição caracteriza o estado que precisa existir para a satisfação de um objetivo ser possível, ou seja, devem ser atendidas para possibilitar que o objetivo que as mesmas condicionam possa ser também atendido. Ainda segundo Morandini et al. (2017), pré-condições são modeladas para restringir a adoção de um objetivo para um contexto particular. Já uma pós-condição caracteriza o estado do sistema uma vez que o objetivo é concretizado (VILELA et al., 2017b; ANTON, 1996). Um exemplo de pré-condição na Bomba de Infusão de Insulina, poderia ser: "bomba desligada", para o objetivo de trocar reservatório ser alcançado. A pós-condição seria a ação concretizada, ou seja, reservatório trocado.

3.2.4.9 Nível de Criticidade de Elementos Críticos de Segurança (*Criticality level of safety-critical elements*)

O nível de criticidade de um elemento de segurança crítica, define o grau de criticalidade desse elemento em alguma escala pré-definida. . A criticalidade é definida através da análise do elemento em relação ao sistema e do nível de controle que ele exerce sobre as funcionalidades e o quanto o elemento pode contribuir para acidentes e perigos. Um exemplo de escala é a proposta pelo padrão IEC 61508, que apresenta quatro níveis de integridade de segurança - SILs (do inglês Safety Integrity Levels). O SIL 4 correponde ao mais alto nível de integridade de segurança. Através da definição do SIL da função de segurança é definida a medida que será tomada no desenvolvimento de sistemas relacionados à segurança (BELL, 2006).

3.2.4.10 Estratégias de Segurança (*Safety Strategies*)

Estratégias de segurança são ações que visam mitigar as consequências de um possível acidente. O objetivo dessas ações é eliminar ou reduzir o risco associado a uma situação perigosa. Cada mitigação tem um custo para sua realização, que na maioria das vezes consume algum recurso (VILELA et al., 2017b; MIL-STD-882E. . . , 2012; ASNAR; GIORGINI; MYLOPOULOS, 2011). No exemplo da bomba de infusão de insulina, uma estratégia segurança para mitigar uma falha de software, pode ser o paciente receber alerta sobre a falha e/ou receber suporte para tais falhas, sendo esse suporte dado pelo fornecedor da bomba.

3.2.4.11 Recursos de Segurança (*Safety resources*)

Em se tratando de Sistemas Críticos de Segurança, recursos (que podem ser tanto entidades físicas como informacionais) são os ativos necessários para o correto funcionamento de requisitos críticos, sendo portanto especializações de recursos, indicando assim a sua criticidade em relação à outros recursos (VILELA et al., 2017b). Um recurso de segurança na bomba de infusão de insulina seria as políticas de compartilhamento, usadas para mitigar o perigo da bomba ser compartilhada.

3.2.4.12 Nível de Impacto do Acidente (*Accident Impact Level*)

O impacto define quão crítico é o acidente em relação à segurança do sistema. O nível de impacto é classificado em cinco categorias (LEVESON, 1995; MIL-STD-882E..., 2012; ZOUGHBI; BRIAND; LABICHE, 2011):

1. Catastrófico (*Catastrophic*);
2. Muito Severo (*Hazardous/Severe-major*);
3. Considerável (*Major*);
4. Pequeno (*Minor*); e
5. Sem efeito (*No effect*).

Na bomba de infusão de insulina, uma superdosagem (*overdose*) de insulina é um acidente de impacto catastrófico, pois pode levar à morte. Já o acidente de subdosagem (*underdose*) possui menor impacto, tendo nível de perigo muito severo (*Hazardous/Severe-major*).

3.3 CONCLUSÃO

Este capítulo definiu conceitos importantes em segurança que guiarão o entendimento dessa dissertação. Primeiramente apresentou-se uma visão geral sobre Sistemas Críticos de Segurança. Depois, foi definido o que são softwares críticos e requisitos de segurança. Por fim foram conceituados os termos que serão usados para construção dos construtores da extensão.

No próximo capítulo, apresenta-se a linguagem iStar4Safety e seu processo de desenvolvimento, realizado através do processo PRISE.

4 ISTAR4SAFETY

Neste capítulo é apresentada a proposta de uma extensão da linguagem iStar para modelar requisitos iniciais de segurança. O objetivo é que iStar4Safety possibilite a modelagem de requisitos de segurança o mais cedo possível no desenvolvimento de Sistemas Críticos de Segurança e seja de fácil aprendizado e uso, atendendo assim à questão de pesquisa proposta na seção 1.3.

A seção 4.1 apresenta uma análise para verificar a necessidade de uma nova extensão para modelar requisitos de segurança. Nesta seção, os conceitos candidatos à integrar a extensão são apresentados. Um Sistema Crítico de Segurança, a bomba de infusão de insulina apresentada na seção 2.1, será modelada somente com a linguagem iStar 2.0 para confirmar a necessidade da nova extensão. Por último, será realizada uma busca por extensões já propostas em iStar para modelar requisitos de segurança.

Em seguida, a seção 4.2 apresenta o subprocesso dois do PRISE que tem por finalidade conceituar iStar4Safety. Esta seção inclui uma pesquisa por construtores já existentes e que possam ser reusados pela nova extensão. Os conceitos candidatos à integrar iStar4Safety serão definidos em maiores detalhes. Finalmente, será realizada uma análise de integração entre construtores nativos de iStar e KAOS aos conceitos que devem ser modelados pela nova extensão.

A seção 4.3 mostra o processo de desenvolvimento de iStar4Safety. O metamodelo da extensão será apresentado. As regras de validação, tanto de iStar 2.0 quanto da extensão são elencadas. Logo após, a sintaxe concreta de iStar4Safety será apresentada seguida de uma verificação de completude, consistência e conflitos dos construtores da extensão. Como parte desse subprocesso, uma ferramenta de modelagem será proposta com o intuito de possibilitar a criação de modelos que incluam os construtores criados. Por fim, são propostas diretrizes para guiar a modelagem de requisitos de segurança com iStar4Safety.

Já na seção 4.4, o Sistema de Bomba de Infusão de Insulina(seção 2.1) será novamente modelado, com iStar4SAfety, validando assim a extensão criada.

Por fim, a seção 4.5 mostra o processo de publicação da extensão no catálogo CATIE, finalizando os passos para o desenvolvimento de iStar4Safety indicados pelo PRISE.

É importante salientar que, conforme preconiza o processo, especialistas em iStar, segurança e no PRISE foram contactados nos momentos apropriados, para sanar dúvidas que surgiram durante o desenvolvimento da extensão.

4.1 ANALISAR A NECESSIDADE DE PROPOR UMA EXTENSÃO

A primeira etapa do processo PRISE diz respeito a analisar a necessidade de propor uma nova extensão para a linguagem iStar. Esse subprocesso pode ser visualizado através

do diagrama BPMN apresentado na figura 13 e disponível em Subprocesso 1 - PRISE (GONÇALVES ENYO, 2019). Todos os passos do subprocesso foram realizados.

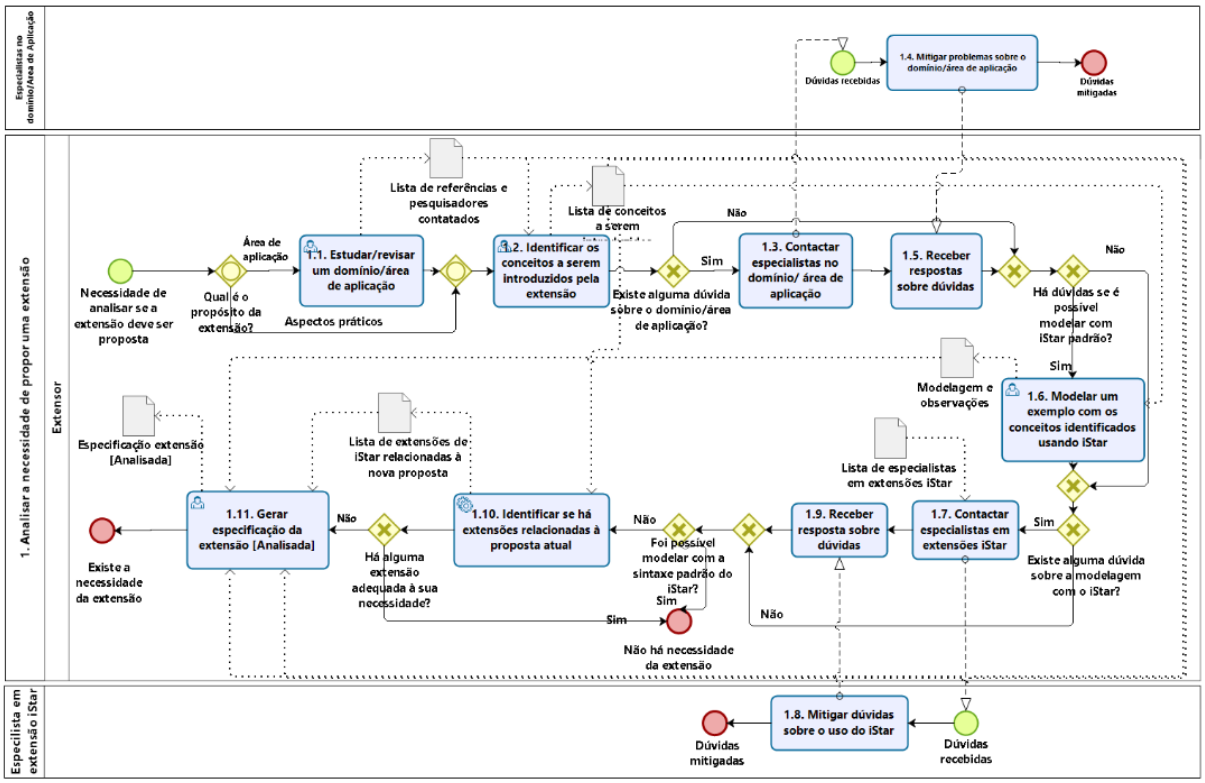


Figura 13 – Subprocesso 1 do PRISE - Análise da necessidade de proposta de extensão

Fonte: Adaptado de Site Prise (GONÇALVES, 2018).

Como ponto de partida, identificou-se que a proposta de extensão estava relacionada à uma área de aplicação, segurança. Esta área foi escolhida devido à falta de uma linguagem de modelagem que possibilitasse a modelagem de conceitos de segurança relacionados à Engenharia de Requisitos Iniciais de Sistemas de Segurança Críticos (VILELA et al., 2017b).

4.1.1 Estudar/Revisar uma Área de Domínio/Aplicação

Conforme a figura 13, o subprocesso 1.1 do subprocesso 1 apresenta a necessidade de estudar/revisar a área de domínio/aplicação. Para isso, foi realizada uma vasta pesquisa bibliográfica , a fim de investigar a área, os tipos de análises de segurança e identificar conceitos candidatos à integrar a nova extensão. A criação da extensão parte da necessidade de atender à características preconizadas em Vilela et al. (2017b) ou seja, características necessárias para a modelagem da fase inicial de requisitos, que em segurança se relacionam às informações disponíveis no documento da Análise Preliminar de Segurança. A pesquisa bibliográfica teve como ponto inicial o livro de Leveson (1995) que define e trata de vários aspectos relacionados à segurança em relação à sistemas, figurando-se como material essencial para compreendermos essa área. Vários outros trabalhos contribuiriam para esse

subprocesso e consequentemente para a criação da extensão. Tais trabalhos estão integrados às referências dessa dissertação (LEVESON, 1995; VILELA et al., 2017a; MARTINS; GORSCHEK, 2016; FIRESMITH, 2004; MIL-STD-882E..., 2012; ZOUGHBI; BRIAND; LABICHE, 2011; LAMSWEERDE, 2004; LAMSWEERDE; LETIER, 2000).

4.1.2 Identificar os Conceitos a Serem Introduzidos por *iStar4Safety*

Para a identificação de conceitos à serem introduzidos na nova extensão para modelagem inicial de conceitos de segurança de Sistemas de Segurança Críticos, utilizou-se como base o trabalho de Vilela et al. (2017b). Nele, os autores apresentam uma lista de características, definidas através de vasta pesquisa bibliográfica, que deveriam ser modeladas quando o objeto de modelagem fosse a Análise Preliminar de Segurança de Sistema Críticos de Segurança.

Inicialmente, todas as características apontadas no trabalho de Vilela et al. (2017b) foram consideradas candidatas à integrar a nova extensão e encontram-se apresentadas na tabela 4. Tal tabela foi adaptada para inserção da versão 2.0 de iStar, já que sua versão original trata de iStar 1.0. O termo "N"indica que a linguagem não possui construtores aptos à modelar o conceito, "S"indica que a linguagem possui construtores aptos à modelar o conceito e o termo "P"indica que a linguagem permite a modelagem parcial do conceito.

A tabela original possui ainda as linguagens GRL (Linguagem de Requisitos Orientada à Objetivos - *Goal-oriented Requirement Language*) e NFR Framework(Framework de Requisitos Não-funcionais - *Non-Functional Requirements Framework*) e pode ser consultada em Vilela et al. (2017b).

Tabela 4 – Adaptação da lista de características para modelagem de uma Análise Preliminar de Segurança presentes nas linguagens GORE a partir de Vilela et al. (2017b).

	Característica	iStar 1.0	iStar 2.0	KAOS
1	Modelagem de acidentes	N	N	P(Obstáculo)
2	Modelagem de perigos	N	N	P(Obstáculo)
3	Modelagem de causas de perigo	N	N	P(Sub-obstáculo)
4	Modelagem de condições ambientais	N	N	S(Condição de trigger)
5	Modelagem de requisitos funcionais de segurança	S(Tarefas)	S(Tarefas)	S(Tarefas)
6	Representar restrições	S(Links de contribuição)	S(Links de contribuição)	S(Links de contribução)
7	Representar obstáculos	N	N	S(Obstáculo)
8	Representar pre/pós condições	N	N	S(Par Pre/Pós Condição)
9	Representar os relacionamentos de contrutores à perigos	N	N	N
10	Representar como um evento afeta a segurança de um sistema	S(Softgoals e links de contribuição)	S(Qualidades e links de contribuição)	S(Softgoals e links de contribuição)
11	Representar o nível de criticidade de um elemento crítico ou suas contribuições a situações de falha	N	N	N
12	Modelagem e raciocínio de estratégias de segurança	S(Softgoals e links de contribuição)	S(Qualidades e links de contribuição)	S(Operacionalizalizações e links de contribuição)
13	Modelagem de recursos	S(Recurso)	S(Recurso)	S(Operacionalizações)
14	Modelagem do nível de impacto de um acidente	N	N	N
15	Apoio de descrição textual de requisitos de segurança	N	N	N

Todos os conceitos apresentados na tabela 4, à exceção dos itens 9, 10, 11 e 15 encontram-se definidos na subseção 3.2. Os itens não definidos foram considerados auto-explicativos.

Os cinco primeiros construtores foram definidos por Vilela et al. (2017b) como conceitos **chave** (*core*) endereçados por uma Análise Preliminar de Segurança de Sistemas Críticos de Segurança.

Salienta-se que a intenção da modelagem não é incluir todos os conceitos, mas sim, permitir a modelagem consistente de requisitos iniciais de um Sistema Crítico de Segurança, definindo, portanto, uma extensão que cubra as necessidades de modelagem de uma Análise Preliminar de Segurança e, ao mesmo tempo, atenda às características necessárias para a criação de extensões iStar consistentes.

Portanto, usou-se também as diretrizes propostas por Gonçalves et al. (2018c) e apresentado em 2.5. O item D9 das diretrizes aponta a necessidade de definir extensões com o menor número de modificações e novas representações à fim de não tornar o uso da linguagem de modelagem complicada. Além disso, Stålhane e Myklebust (2016) aponta a necessidade de que a análise de segurança deve possibilitar a participação dos *stakeholders* do sistema, a fim de que estes participem da avaliação de segurança, o que preconiza o uso de linguagens que sejam fáceis de entender, aprender e aplicar. A aceitação e uso da extensão poderia ser afetada negativamente se a mesma destoasse das características de iStar, ou seja, facilidade de uso e quantidade limitada de construtores.

Após a seleção dos conceitos candidatos, foi definido um subconjunto de conceitos à serem modelados, através tanto de pesquisas bibliográficas como de reuniões com os autores do trabalho,Vilela et al. (2017b), que propuseram as características de segurança necessárias para a modelagem de uma Análise Preliminar de Segurança, especialistas em iStar e proponentes de extensões de iStar.

Nesse sentido, concluiu-se que um subconjunto de nove conceitos atenderia à modelagem de requisitos iniciais em iStar sem ferir as características propostas inicialmente bem como as diretrizes do processo PRISE. O objetivo dessa extensão é permitir uma modelagem consistente da Análise Preliminar de Segurança que contribuía para o entendimento dos requisitos iniciais de segurança de um Sistema Crítico. Os conceitos selecionados após análise foram os seguintes:

1. Modelagem de Acidentes (Conceito chave);
2. Modelagem de Perigos (Conceito chave);
3. Modelagem de Causa de Perigos (Conceito chave);
4. Modelagem de Condições Ambientais (Conceito chave);
5. Modelagem de Requisitos Funcionais de Segurança (Conceito chave);
6. Representação do relacionamento entre os construtores relacionados à perigo: É necessário que os conceitos chave se relacionem para permitir a racionalização do processo de modelagem de requisitos relacionados à segurança;
7. Modelagem e raciocínio de Estratégias de Segurança: Tal conceito é necessário devido à necessidade de propor estratégias para a resolução de situações de perigo e

assim obter uma modelagem que capture a intenção de manter o sistema o mais seguro possível;

8. Habilidade de modelar Recursos: Essa característica é importante devido ao fato de que recursos podem também ser artefatos críticos. Logo, visualizou-se a necessidade de modelá-lo; e

9. Modelagem do Nível de Impacto do Acidente: O nível de impacto do acidente deve ser considerado em uma análise preliminar para segurança. Esse nível pode ter cinco valores, conforme já informado: (1) Catastrófico (2) Muito Severo (3) Considerável (4) Pequeno (5) Sem Efeito.

Conforme apresentado pelo trabalho original, o conceito 10, que diz respeito à representar como um evento afeta segurança, já é modelado pela linguagem através da inserção da qualidade "segurança"e da relação de contribuição entre ela e os demais elementos.

Os demais conceitos apontados por Vilela et al. (2017b) foram modelados através de outros conceitos ou não foram cobertos pela extensão. Uma explicação sobre o motivo dos conceitos não terem sido inseridos na nova linguagem é apresentada abaixo:

- Representação de restrições: Apesar da possibilidade apontada por Vilela et al. (2017b) de desenhar a restrição através de links de contribuição (*hurt, break, help, make*). Leveson (1995) cita a importância em separar restrições de segurança de restrições não relacionadas à essa propriedade, quando se trata de sistemas críticos. Porém, os links existentes permitem somente a associação entre elementos de qualidade e outros elementos, não possibilitando modelar restrições que podem ser representadas por outros construtores além de qualidades. Apesar disso, devido à não ser um elemento chave na modelagem de uma Análise Preliminar de Segurança, esse conceito não foi inserido na nova extensão.

- Representação de obstáculos: Um obstáculo em segurança (*safety*) à um objetivo de segurança é representado pelo conceito de perigo (LAMSWEERDE; LETIER, 2000). Portanto o conceito foi representado no domínio de segurança através da modelagem de perigo.

- Representação de pre/pós-condições: iStar é uma linguagem que não representa a temporalidade em sua modelagem. Portanto não foi observada a necessidade de modelar pré/pós-condições.

- Representação como um evento afeta a segurança do sistema: Conforme indicado no trabalho que apresenta as características necessárias para modelagem de uma Análise de Segurança do Sistema (VILELA et al., 2017b), através dos links de contribuição (*make, help, hurt, break*) é possível definir o nível de contribuição de dado evento para segurança.

- Representar o nível de criticidade de um elemento crítico ou suas contribuições à situações de falha: Esse item não foi modelado diretamente. Para o nível de modelagem à que a nova extensão propõe atender, ou seja, a engenharia de requisitos inicial, acredita-se que a inserção dos novos elementos indicando situações de perigo, tarefas e recursos de segurança e objetivos de segurança já demonstre quais itens estão relacionados à Análise de Segurança do Sistema. Além disso, as contribuições de um elemento à situações de falhas podem ser indicados pelos links de contribuição.

- Apoio de descrição textual de requisitos de segurança: Considerando que a extensão trata da modelagem de requisitos inicias de segurança, e que a linguagem nativa iStar 2.0 não permite a inserção de descrição textual, não consideramos necessário para essa fase inserir descrições textuais dos elementos.

4.1.3 Modelagem de um Exemplo com iStar 2.0

Como passo importante para definir a necessidade de criação da extensão, um exemplo de Sistema Crítico de Segurança foi modelado. O processo PRISE orienta que seja realizada a modelagem de um sistema completo que ilustre o domínio da proposta da extensão e assim, todos os conceitos previamente definidos como necessários. Além disso, o exemplo pode trazer luz à novos conceitos e características que não foram considerados.

O sistema escolhido foi o de uma Bomba de Infusão de Insulina, devido à sua criticidade e por ser um exemplo bastante utilizado/modelado pela comunidade acadêmica para a área de segurança (VILELA et al., 2017b; MARTINS et al., 2015; ZHANG; JONES; JETLEY, 2010; ZHANG et al., 2011). Detalhes sobre o sistema e seus requisitos encontram-se na seção 2.1.

Com base nos requisitos propostos e considerando que o objetivo das linguagens GORE é atuar na fase inicial da Engenharia de Requisitos, foram desenvolvidos os modelos SD (figura 14) e SR (figura 15) usando a linguagem Istar 2.0 para o Sistema de Bomba de Insulina, com o apoio da ferramenta de modelagem *piStar*(PIMENTEL; CASTRO, 2018). O modelo SR por sua vez , devido ao seu tamanho, foi dividido em quatro partes, separadas por atores, à fim de possibilitar uma melhor visualização. A figura 16 representa o ator Paciente, a figura 17 representa o Controlador da Bomba de Insulina. Já a figura 18 apresenta a modelagem do ator Fornecedor da Bomba e, por fim, a figura 19 mostra a parte do modelo que se refere ao ator Controlador de Sensores.

É importante salientar que esse modelo utilizou somente construtores da linguagem padrão de iStar 2.0, o que permite observar, já previamente, a limitação em modelar características relacionadas à segurança com os construtores nativos da linguagem.

Conforme pode ser visualizado no modelo SD representado pela figura 14 e que modela os requisitos apresentados na seção 2.1.2.1, os atores Paciente, Fornecedor da bomba de

insulina, Controlador da bomba e Controladores de sensores, dependem um do outro para a realização dos objetivos modelados.

Neste modelo SD, está representado que o ator Paciente depende do ator Fornecedor da Bomba para "Ser auxiliado no uso da bomba". O paciente por sua vez, depende do ator Controlador da Bomba de Insulina para ter as infusões basal e *bolus* gerenciadas. Além disso, o Controlador da Bomba de Insulina depende do ator Controlador de Sensores para atingir os objetivos de "Ter sensores monitorados"e "Manter tela atualizada".

Na figura 15, que apresenta o modelo SR da Bomba de Infusão de Insulina, visualizamos a falta de elementos relacionados à segurança, devido à não possibilidade de modelá-lo com a versão nativa de iStar 2.0.

Devido às limitações de iStar 2.0, não foi possível representar os seguintes conceitos: perigo, causa do perigo e condições ambientais. Uma forma de modelar acidentes foi através da negação de um objetivo. Recursos podem ser modelados através do construtor recurso, porém não é possível especificar a natureza crítica dos mesmos. Requisitos Funcionais de Segurança também podem ser modelados através de tarefas e recursos, porém não é possível associá-los aos perigos que são o motivo de sua existência e nem demonstrar que são especificamente relacionados à segurança. É possível modelar como eventos afetam a segurança do sistema, através de links de contribuição e do construtor "segurança"sendo uma qualidade, o que permitiu definir que um objetivo é crítico, ou seja, associando-o através do link de contribuição à qualidade "segurança".

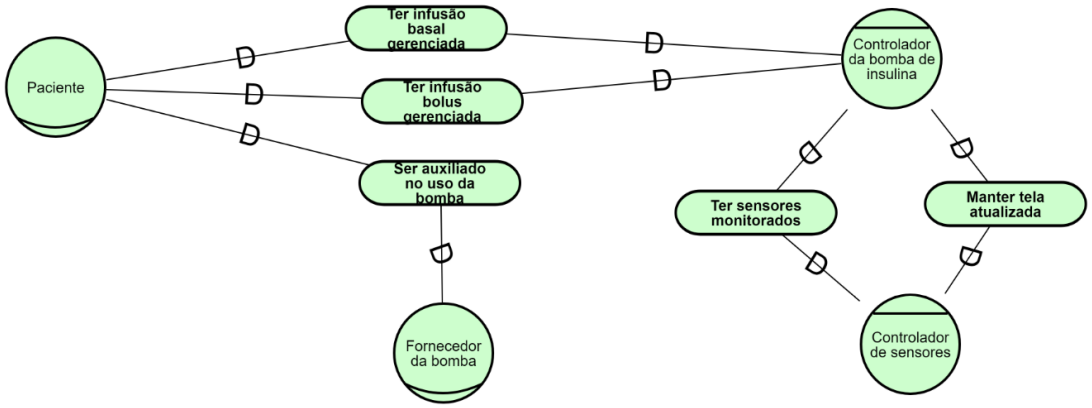


Figura 14 – Modelo SD de iStar 2.0 do exemplo do Sistema de Bomba de Insulina, desenvolvido com a ferramenta piStar.

Fonte: (Autora, 2018).

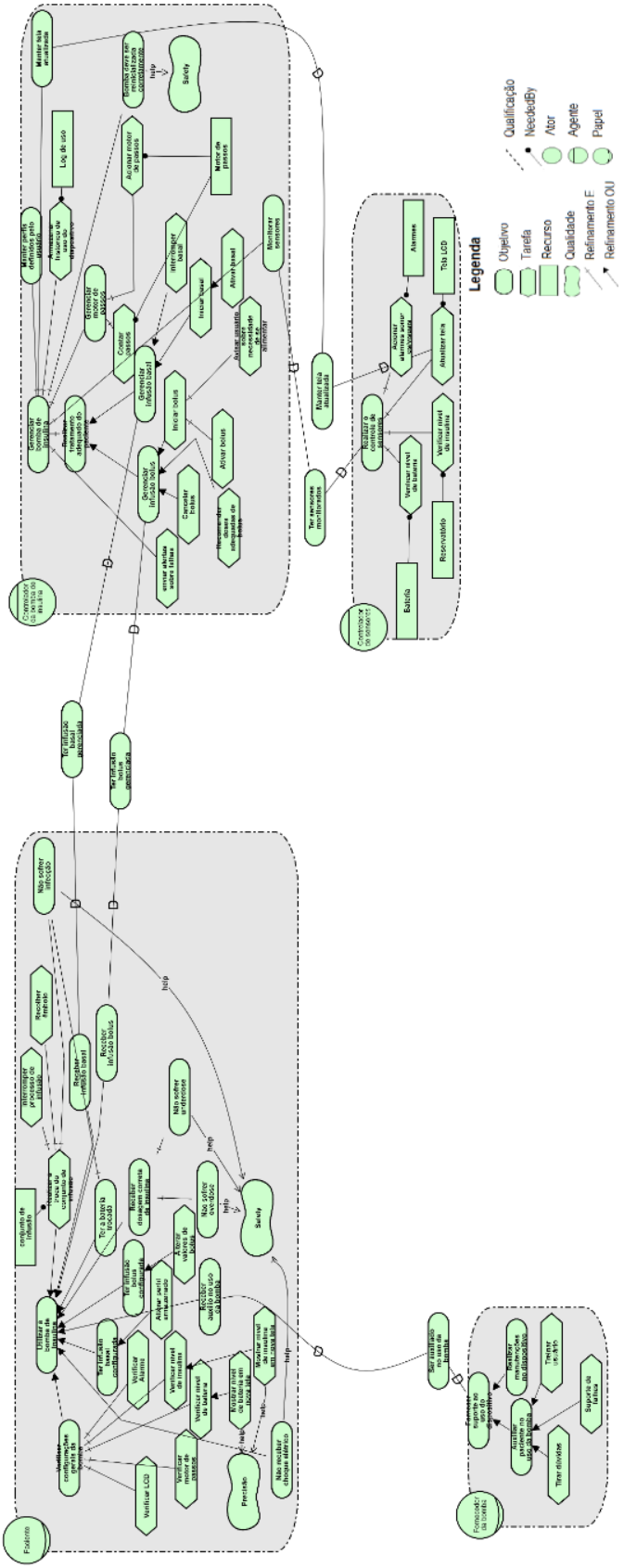


Figura 15 – Modelo SR, em iStar, do exemplo do Sistema de Bomba de Insulina, desenvolvido com a ferramenta piStar.

Fonte: (Autora, 2018).

Explica-se nas próximas seções, com mais detalhes, a modelagem do modelo SR de cada ator separadamente.

4.1.3.1 Ator Paciente

O ator **paciente** tem como principal objetivo, utilizar a Bomba de Infusão de insulina. Para tanto ele pode: "Verificar configurações gerais da bomba", "Ter infusão basal configurada", "Ter infusão *bolus* configurada", pretende "Não receber choque elétrico"(O que auxilia em manter a segurança do mesmo), "Receber auxílio no uso da bomba", "Receber dosagem correta de insulina", "Ter bateria trocada", "Realizar a troca do conjunto de infusão", além de receber infusão basal e *bolus*.

Ao verificar as configurações gerais da bomba, o **paciente** irá verificar também a tela LCD, o motor de passos e o alarme. O **paciente** também irá verificar o nível de bateria e de insulina em nova tela o que irá ajudar na precisão de uso.

Para ter a infusão basal configurada, o **paciente** pode alterar o perfil basal armazenado. Por sua vez, ao configurar a infusão *bolus*, ele pode alterar os valores de *bolus* configurados.

Para atender ao objetivo de receber dosagem correta de insulina, o **paciente** deseja "Não sofrer overdose"e "Não sofrer *underdose*", o que são objetivos que incrementam a segurança do uso do dispositivo.

Para o ator **paciente** realizar a troca do conjunto de infusão, ele necessita do recurso "conjunto de infusão". Ao mesmo tempo, para realizar a troca, ele deve interromper qualquer processo de infusão que esteja ativo e recolher o êmbolo.

Tanto ao realizar a troca da bateria, quanto ao trocar o cartucho, o **paciente** deseja "Não sofrer infecção"à fim de ajudar a manter seguro o uso do dispositivo.

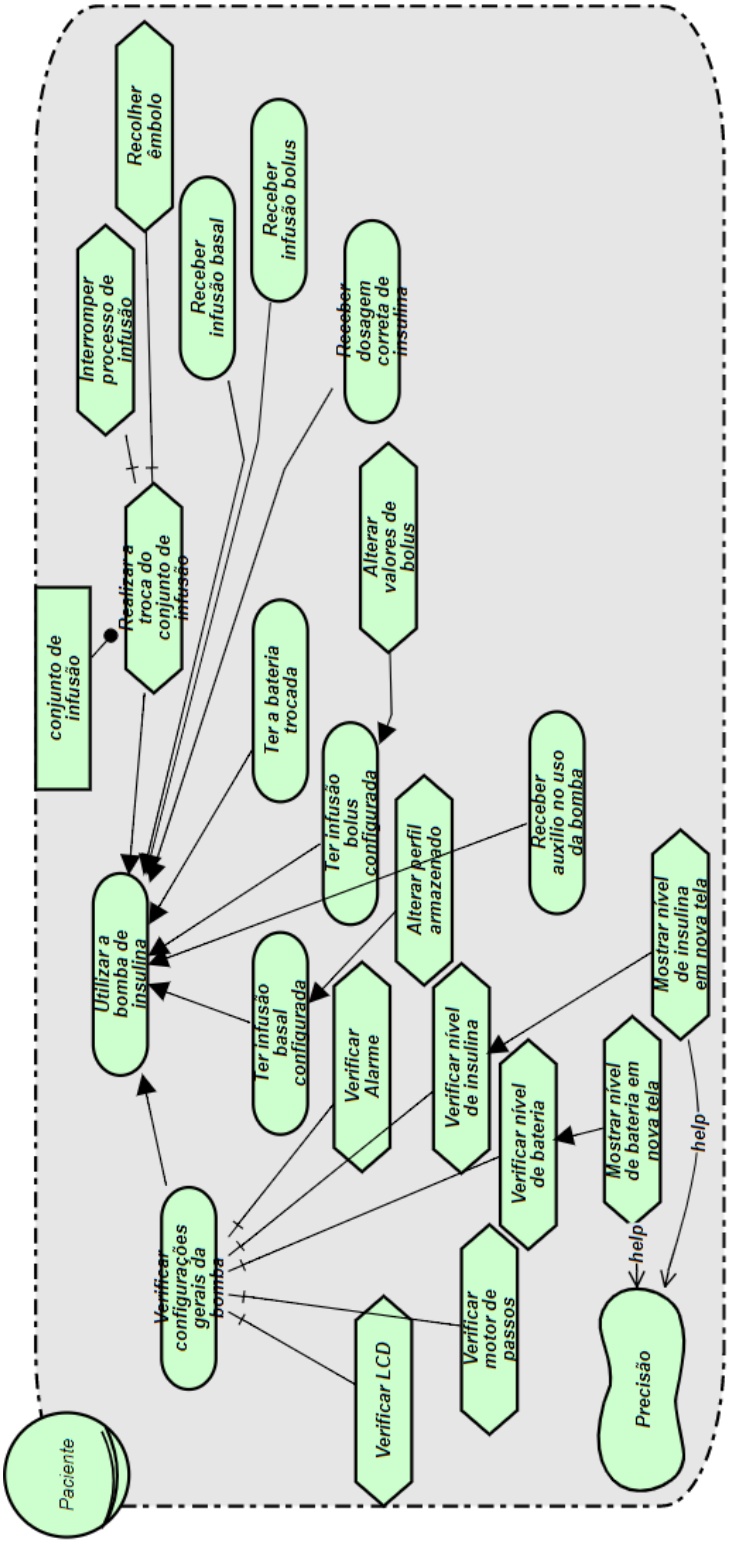


Figura 16 – Ator paciente - Parte 1 do modelo SR, em iStar, do exemplo do Sistema de Bomba de Insulina, desenvolvido com a ferramenta piStar.

Fonte: (Autora, 2018).

4.1.3.2 Ator Controlador da Bomba de Insulina

O ator **controlador da bomba de insulina** tem como principal objetivo gerenciar a bomba de insulina. Para tanto, ele é responsável por: "Enviar alertas sobre falhas", "Realizar tratamento adequado do paciente", "Monitorar sensores", "Gerenciar motor de passos", reinicializar a bomba corretamente, o que incrementa a segurança. Além disso, ele deve armazenar o histórico de uso do dispositivo (necessitando do arquivo de *log* de uso para tal), "Manter a tela atualizada"e "Manter perfis definidos pelo usuário".

Para realizar o tratamento adequado do **paciente**, o controlador da bomba de insulina deve gerenciar as infusões *bolus* ou basal. O gerenciamento de *bolus* consiste em poder "Cancelar *bolus*", "Recomendar doses adequadas de *bolus*"ou ainda "Iniciar *bolus*". Ao iniciar a infusão o controlador deve tanto ativar a infusão como avisar ao paciente sobre a necessidade de se alimentar. Quando gerenciando a infusão basal, o **controlador da bomba** pode iniciar a infusão basal ou interrompe-la.

Quando gerenciando o motor de passos, o **controlador da bomba** de insulina deverá acionar o motor e contar os passos.

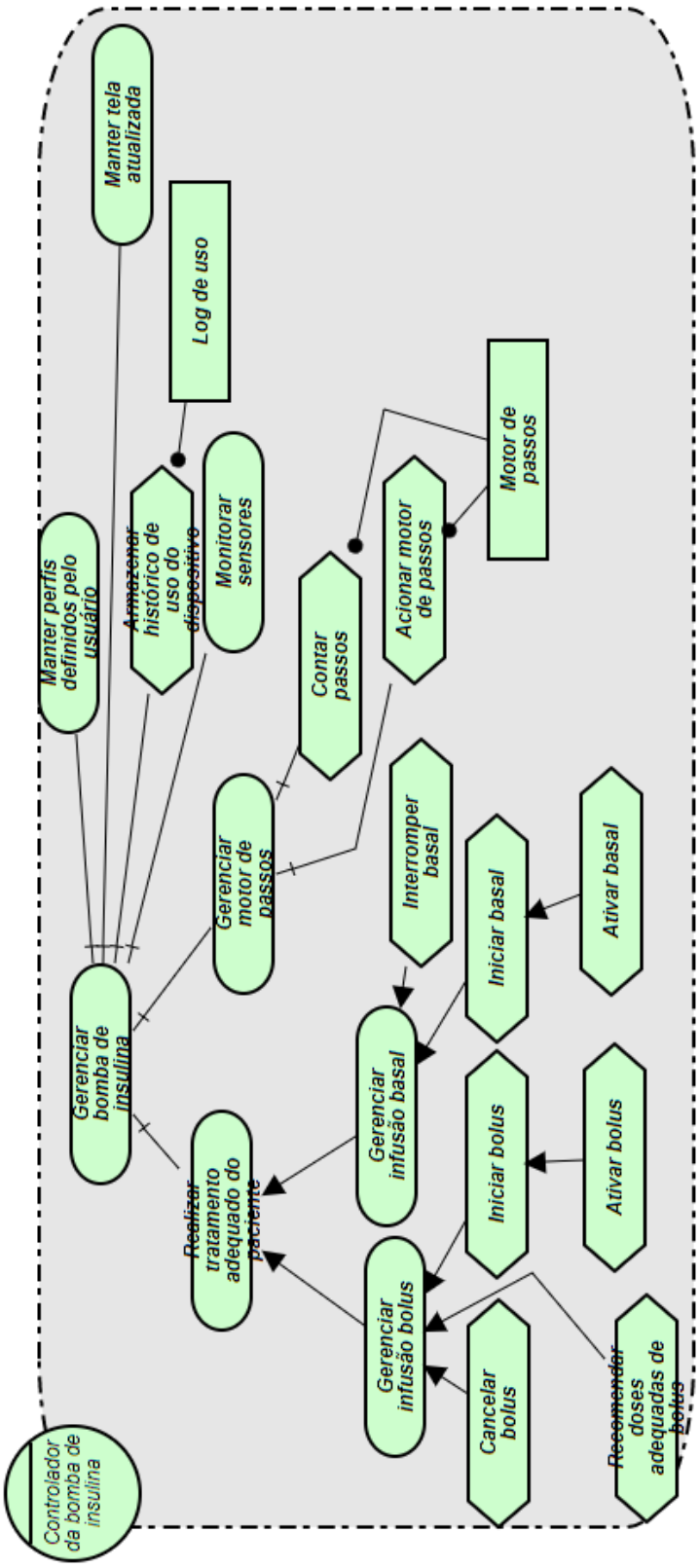


Figura 17 – Ator Controlador da Bomba de Insulina - Parte 2 do modelo SR, em iStar, do exemplo do Sistema de Bomba de Insulina, desenvolvido com a ferramenta piStar.

Fonte: (Autora, 2018).

4.1.3.3 Ator Controlador de Sensores

O ator **controlador de sensores** tem como principal objetivo controlar os sensores do dispositivo de infusão de insulina. Á fim de realizar esse objetivo, o ator deve verificar os níveis de bateria e insulina, atualizar a tela e deve acionar alarmes sonoros/visuais quando necessário.

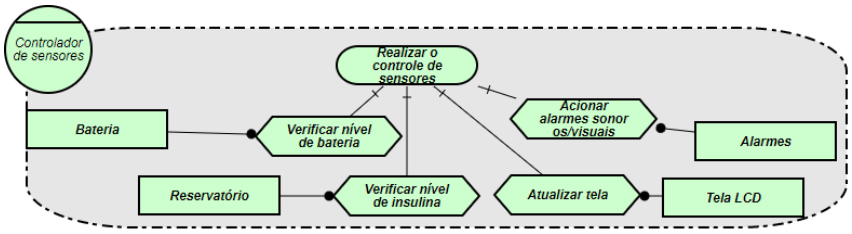


Figura 18 – Ator Controlador de sensores - Parte 3 do modelo SR, em iStar, do exemplo do Sistema de Bomba de Insulina, desenvolvido com a ferramenta piStar.

Fonte: (Autora, 2018).

4.1.3.4 Ator Fornecedor da Bomba

O ator **fornecedor da bomba** tem como principal objetivo "Fornecer suporte ao uso do dispositivo". Esse suporte se dá através de manutenções no dispositivo ou no auxílio à pacientes no uso da bomba. O **fornecedor da bomba** pode auxiliar o paciente tirando dúvidas, dando suporte à falhas do dispositivo, ou mesmo treinando-os.

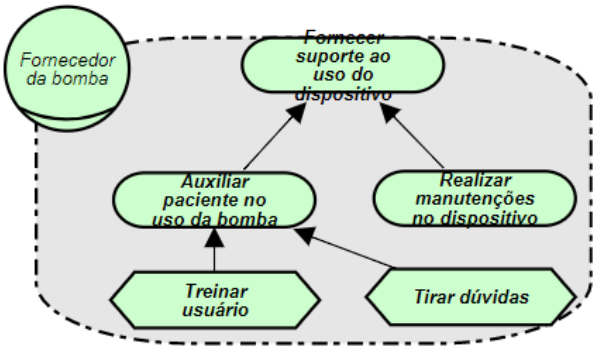


Figura 19 – Ator Fornecedor da bomba - Parte 4 do modelo SR, em iStar, do exemplo do Sistema de Bomba de Insulina, desenvolvido com a ferramenta piStar.

Fonte: (Autora, 2018).

Por fim conclui-se que a modelagem do exemplo confirmou a necessidade de criação de uma extensão à fim de modelar requisitos relacionados à segurança em Sistemas Críticos de Segurança.

4.1.4 Busca por Extensões Relacionadas à Proposta de *iStar4Safety*

Foi realizada uma busca no catálogo CATIE (GONÇALVES et al., 2018b) de extensões de iStar, com o intuito de verificar se haviam extensões criadas para o domínio desejado. Essa busca não retornou nenhuma extensão relacionada à segurança.

4.2 CONCEITUALIZAÇÃO DA EXTENSÃO DE ISTAR

Nesta seção iremos apresentar os passos realizados propostos pelo subprocesso dois do PRISE, que pode ser visto na figura 20. Primeiramente, uma busca e seleção por construtores foi realizada no catálogo CATIE (GONÇALVES et al., 2018b), seguido pela descrição dos conceitos candidatos à integrar iStarSafety. Logo após, uma possibilidade de de integração entre construtores nativos de iStar e conceitos candidatos, além de uma discussão sobre os construtores que farão parte da extensão são apresentados na subseção 4.2.3.

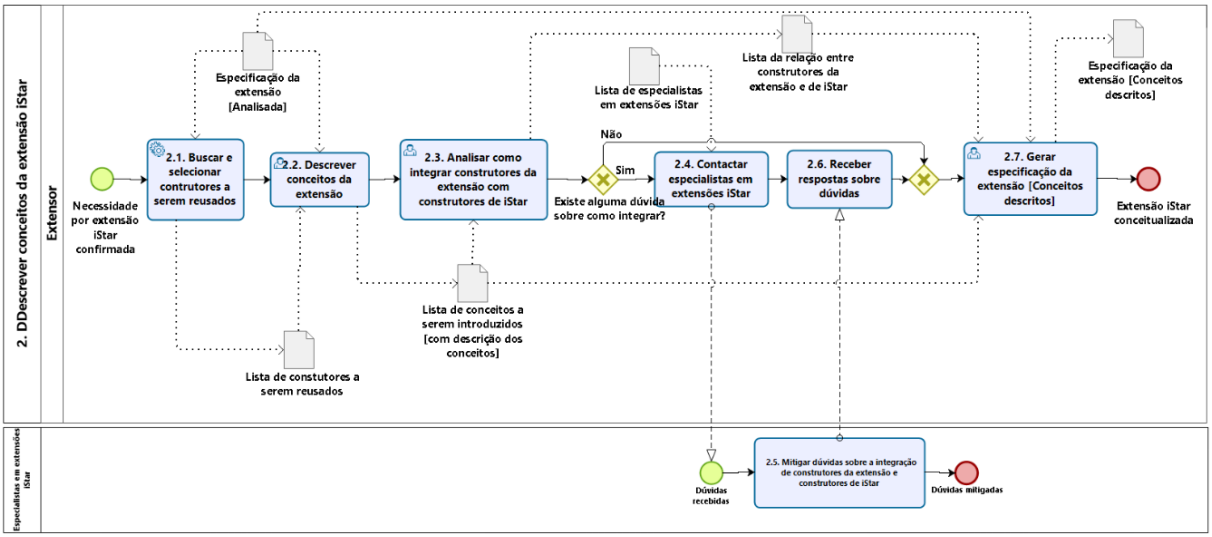


Figura 20 – Subprocesso 2 do PRISE - Descrição de conceitos para a extensão de *iStar*

Fonte: Adaptado de Site Prise (GONÇALVES, 2018).

4.2.1 Pesquisa por Construtores já Existentes em Outras Extensões

Seguindo o processo indicado pelo PRISE foi realizada uma pesquisa no catálogo de extensões, CATIE, em busca de construtores já propostos, e que pudessem ser reusados na proposta atual da extensão iStar4safety. Essa prática é recomendada a fim de evitar redundâncias de construtores, além de melhorar o entendimento e aceitação de novas extensões (GONÇALVES et al., 2018c).

Após as buscas no catálogo CATIE[1], não foi encontrado nenhum construtor de outra extensão proposta que pudesse ser reusado por esse trabalho.

[1] O catálogo CATIE encontra-se na página <https://istarextensions.cin.ufpe.br/catalogue/>.

4.2.2 Definição de Conceitos de iStar4Safety

À fim de descrever o significado dos conceitos que serão inseridos na extensão iStar4Safety, realizou-se o subprocesso de definição de conceitos da extensão.

Os conceitos à serem modelados são definidos como:

1. **Acidente**: É um evento não planejado e nem desejado, mas não necessariamente não esperado, que resulta em, no mínimo, um nível específico de perda (LEVESON, 1995; BERRY, 1998). O termo foi detalhado na seção 3.2.4.1.

2. **Perigo**: Trata-se de "[...]um estado ou conjunto de condições de um sistema (ou objeto) que, junto com outras condições no ambiente do sistema (ou objeto) irá inevitavelmente levar à um acidente (evento de perda)."Leveson (1995, p.177). O acidente é consequência de um perigo. Para uma descrição mais detalhada, veja a seção 3.2.4.2.

3. **Causa de Perigo**: É representada por uma condição que sozinha ou associada à outras, é/são suficiente(s) para o perigo relacionado à ela(s) ocorrer. As causas do perigo podem ser controladas ou até eliminadas em alguns casos (BERRY, 1998; VILELA et al., 2017b; LEVESON, 1995). O conceito foi mais detalhado em seção 3.2.4.3.

4. **Condição Ambiental**: Tratam-se de um conjunto de componentes e suas propriedades, incluindo elementos físicos, culturais, demográficos, econômicos, políticos, regulatórios ou tecnológicos que, apesar de não serem parte do sistema, podem afetar seu comportamento.Para maiores detalhes veja seção 3.2.4.4.

5. **Requisitos Funcionais de Segurança**: São os requisitos funcionais usados para mitigar ou prevenir os efeitos de falhas identificadas na análise de segurança.

6. **Estratégias de Segurança**: São ações que visam mitigar as consequências de um possível acidente. O objetivo dessas ações é eliminar ou reduzir o risco associado a uma situação perigosa. Cada mitigação tem um custo para sua realização, que na maioria das vezes envolve o consumo algum recurso (VILELA et al., 2017b; MIL-STD-882E..., 2012; ASNAR; GIORGINI; MYLOPOULOS, 2011).

7. **Recursos**: Na linguagem iStar recursos são apresentados como entidades informacionais ou físicas que são requeridas pelo ator à fim de realizar uma tarefa (DALPIAZ; FRANCH; HORKOFF, 2016). Ou seja, em se tratando de Sistemas Críticos de Segurança, recursos são os ativos necessários para o correto funcionamento de requisitos críticos. Portanto são especializações de recursos, indicando assim a sua criticidade em relação à outros recursos (VILELA et al., 2017b).

8. **Nível de Impacto do Acidente**: Define quão crítico é o acidente em relação à segurança do sistema. A seção 3.2.4.12 apresenta maiores detalhes sobre esse conceito.

É importante salientar que o item 6 apresentado na seção 4.1.2, "Representação do relacionamento entre os construtores relacionados à perigos"não continuou presente na lista de conceitos, devido ao fato de que identificou-se que tal ideia, trata-se de um link entre elementos, logo não necessita de conceitualização.

4.2.3 Analisar como Integrar Construtores de iStar4Safety com Construtores de iStar e KAOS

Esse subprocesso define a necessidade de verificar se existem construtores da linguagem nativa, iStar 2.0, que poderão ser reusados na nova extensão, seja através de generalização de construtores já existentes ou do uso do próprio construtor.

A verificação para reuso de construtores preconizada pelo processo PRISE trata apenas daqueles construtores presentes em iStar ou nas suas extensões. Porém, considerando sugestões de especialistas em iStar e em segurança, além do trabalho de Vilela et al. (2017b) que também avalia a linguagem KAOS (DARDENNE; LAMSWEERDE; FICKAS, 1993) como apta à modelar conceitos da Análise Preliminar de Segurança de Sistemas Críticos de Segurança, optou-se por considerar os construtores como candidatos à modelar os conceitos identificados para Sistemas Críticos de Segurança na linguagem iStar. A tabela 4, adaptada de Vilela et al. (2017b) apresenta uma relação dos conceitos presentes tanto em KAOS como em iStar, definindo quais deles são passíveis de representação através das linguagens.

É importante salientar que Vilela et al. (2017b) indicam que KAOS é a única linguagem de modelagem GORE, dentre as quatro mais usadas (iStar,KAOS, GRL, NFR *Framework*), que permite a modelagem de todos os cinco conceitos chave: acidente, perigo, causa de perigo, condição ambiental e requisitos funcionais de segurança.

Portanto, a abordagem usada por KAOS para a modelagem dos conceitos-chave foi escolhida e adaptada para ser usada na extensão iStar4Safety.

Como pode ser observado, para KAOS, obstáculo é indicado como um construtor apto à modelar acidentes, perigos e causas de perigos. Segundo Lamsweerde e Letier (2000), obstáculos apresentam uma ideia contrária de objetivos: um objetivo trata de condições desejadas e um obstáculo de condições indesejadas, já que o mesmo obstrui a satisfação de uma condição desejada. Logo, são definições duais. Se o obstáculo acontecer, o objetivo pode não ser realizado.

Um obstáculo é um conjunto de comportamentos não desejados. Assim, um obstáculo, quando acontece, pode impedir a concretização de um objetivo. Portanto, a negação de tais obstáculos são precondições necessárias para o objetivo ser alcançado satisfatoriamente.

Lamsweerde e Letier (2000) dizem ainda, de forma direta que perigos podem obstruir objetivos de segurança. Firesmith (2004) já apresentou uma definição para objetivo de segurança, considerado um tipo de objetivo de qualidade. Segundo o autor, um objetivo de segurança exprime a importância em alcançar o objetivo em relação à atender algum fator alvo de segurança, como saúde dos *stakeholders* ou evitar grandes perdas monetárias.

Lamsweerde e Letier (2000) citam ainda a necessidade de completude na definição de obstáculos. Principalmente no caso de objetivos de alta-prioridade. Mas também salientam a ideia de que a completude está diretamente relacionada ao conhecimento que se tem sobre o domínio em questão.

Logo, chegou-se à conclusão de que perigos seriam representados como obstáculos à Objetivos de Segurança. Nessa dissertação, decidiu-se que o tratamento quanto à perigos se daria somente no domínio de objetivos de segurança (FIRESMITH, 2004) devido à alta-prioridade no tratamento desses objetivos em um Sistema Crítico de Segurança. Um objetivo de segurança, quando obstruído por um perigo pode causar um acidente, que é exatamente o que deseja-se evitar ao se utilizar um Sistema Crítico de Segurança.

Quanto à acidentes, os mesmos serão conceitos subentendidos na modelagem de objetivos de segurança. A negação de um objetivo de segurança irá causar o acidente, ou ainda, a concretização de um perigo pode causar o acidente já que poderá fazer com que o objetivo de segurança não seja satisfeito.

As causas de perigos nada mais são que o refinamento de perigos. Logo, tratam-se também de perigos, portanto, serão representados através do mesmo conceito: perigos.

No caso do conceito de condições ambientais, chegou-se à conclusão de que uma condição ambiental é também uma causa para um perigo. Conforme pode ser visualizado na subseção 3.2.4.3, uma causa do perigo pode ser de quatro tipos, sendo um deles o perigo ambiental. Nesse sentido, na extensão iStar4Safety, o conceito de causa de perigo será representado através do mesmo construtor escolhido para modelar perigos.

O refinamento de objetivos em KAOS termina quando todos os objetivos-folha são realizáveis de forma concreta por algum agente individual responsável por ele. Assim, quando o objetivo é apresentado em termos de condições que são monitoráveis e controláveis pelo agente. Logo, pensando à nível de um objetivos de segurança, que pode ser obstruído por um perigo, que pode ser refinado em causas de perigo, que serão os perigos-filhos, precisaremos definir estratégias para lidar com os perigos encontrados, à fim de possibilitar uma modelagem que preze pela segurança do sistema e do seu ambiente.

Utilizando do processo usado pela linguagem KAOS para modelagem de obstáculos, chegamos à conclusão de que a melhor forma de definir estratégias de segurança é através de **recursos de segurança** e **tarefas de segurança** associadas aos perigos-folha, definindo assim as estratégias para mitigação do perigo (ou seja, construtores que indicam ações concretas). Logo, o conceito de requisito de segurança funcional será atendido por essas duas especializações de construtores nativos de *iStar* 2.0: recurso e tarefa, sendo

que recursos de segurança devem estar associados à tarefas de segurança (ou tarefas), conforme a linguagem nativa.

Portanto, ao final desse subprocesso, chegou-se à conclusão de que é possível reutilizar três construtores nativos de iStar 2.0 na extensão iStar4Safety, a saber: Objetivo, Recurso e Tarefa.

À fim de simplificar a modelagem e para possibilitar uma visualização melhor e mais compreensível das estratégias de tratamento de perigos, os elementos da extensão iStar4Safety somente serão modelados dentro dos atores. Assim, os construtores que serão definidos para a extensão deverão ser modelados dentro das fronteiras dos atores, não podendo portanto, serem elementos *dependum* e não sendo, portanto representados nos modelos SD.

Um refinamento de um objetivo significa que todos os subobjetivos que impliquem o objetivo-pai são condições suficientes para que o pai aconteça. Assim sendo, se ao menos um dos seus subobjetivos for violado, o pai também o será. Logo, trazendo essa ideia à nível de perigos, caso algum perigo que obstrui o objetivo aconteça, o pai será violado, sendo portanto obstruído e podendo causar um acidente.

A tabela 5 condensa os conceitos que serão reusados e sua relação com *iStar* e KAOS. A linguagem KAOS foi escolhida para essa comparação, devido ao fato de ser a linguagem que mais atende à modelagem de conceitos identificados por Vilela et al. (2017b) para a modelagem de requisitos de segurança necessários à modelagem de uma Análise Preliminar de Perigos.

Na próxima seção, serão descritos os passos seguidos para o desenvolvimento da extensão, incluindo seu metamodelo, regras de validação, sintaxe concreta e a ferramenta de apoio à modelagem. Além disso, apresenta-se também as diretrizes indicadas para a modelagem de requisito de segurança com iStar4Safety.

Tabela 5 – Relacionamento entre conceitos e linguagens iStar e KAOS.

Conceito de Segurança	*iStar4Safety*	KAOS	Observações
Acidente	-	-	Acidente será subentendido no modelo pois é uma consequência do perigo obstruir um Objetivo de Segurança.
Perigo	Especialização de Objetivo	Obstáculo	O Perigo, ao contrário de Objetivo não é intencional. Logo, é um objetivo que deseja-se que não se concretize.
Causa de Perigo	Especialização de Objetivo	Obstáculo	Causa de Perigo é um perigo-filho.
Condição Ambiental	Especialização de Objetivo	Obstáculo	Condição Ambiental é uma causa do perigo, portanto é também representado como um perigo-filho.
Requisito de Segurança Funcional/ Estratégia de Segurança	Especialização de Tarefa e Recurso	Requisito	Estratégia de Segurança será a estratégia que será usada para lidar com o perigo. Sendo assim, pode ser uma união de Requisitos Funcionais de Segurança, à fim de explicar a estratégia de segurança que será usada.
Relacionamento entre construtores relacionados à perigos.	Link obstrui e refinamentos E/OU	-	O link obstrui será usado para associar objetivos de segurança e perigos. Perigos e Estratégias de Segurança serão associados pelos links de refinamento E/OU.

4.3 DESENVOLVIMENTO DA EXTENSÃO ISTAR4SAFETY

Nesse momento, após a conceitualização de iStar4Safety ter sido realizada, o processo propõe o desenvolvimento da extensão. O subprocesso 3 (três) do PRISE é apresentado nessa seção e pode ser visto na figura 21. A primeira ação desse subprocesso diz respeito à criar o metamodelo da extensão. Logo após a definição do metamodelo, são definidas as regras de validação que definirão as regras para uma modelagem correta com iStar4Safety.

Em seguida, a sintaxe concreta da extensão é apresentada, além de uma verificação da completude, consistência e conflitos entre os construtores da extensão. O PRISE sugere, à fim de facilitar o uso da extensão, que uma ferramenta de modelagem seja criada para auxiliar no uso da linguagem. Nesse trabalho, a ferramenta piStar (PIMENTEL; CASTRO, 2018) é estendida e a versão criada apresentada na subseção 4.3.5.

Por fim, as diretrizes de uso da ferramenta são apresentadas para orientar a modelagem de requisitos de segurança com iStar4Safety e de forma a auxiliar na completude da modelagem.

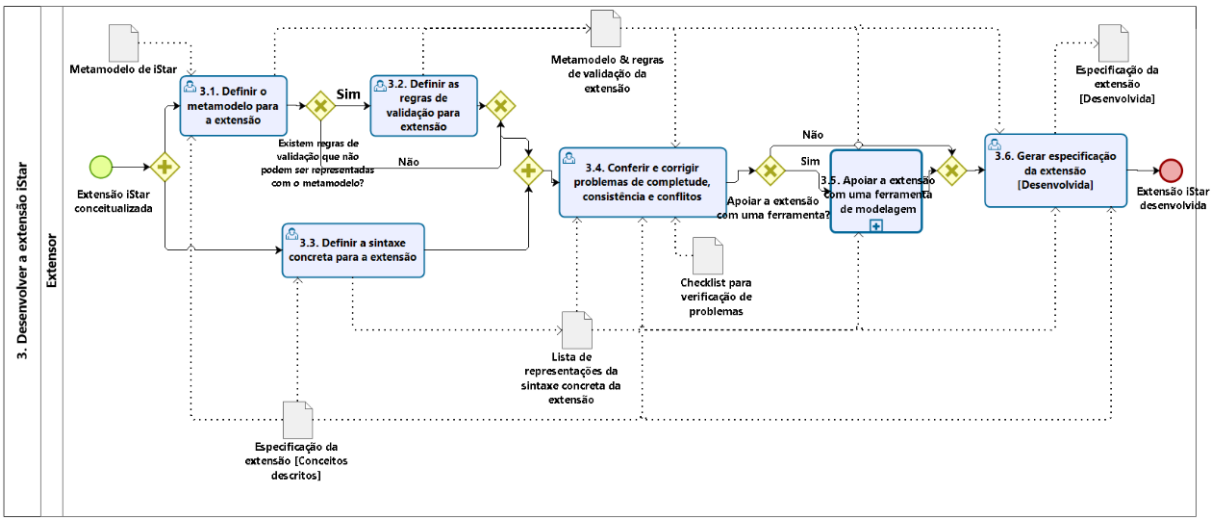


Figura 21 – Subprocesso 3 do PRISE - Desenvolvimento da extensão de *iStar*

Fonte: Adaptado de Site Prise (GONÇALVES, 2018).

4.3.1 Metamodelo

Nessa seção apresenta-se o metamodelo criado para iStar4Safety. O metamodelo é uma abstração dos conceitos usados e de seus relacionamentos, sendo então útil para definir a linguagem de modelagem(BRAMBILLA; CABOT; WIMMER, 2012).

Os construtores criados para as novas extensões devem ser representados no metamodelo da linguagem. Como iStar4Safety estende iStar 2.0, o metamodelo de iStar deve então ser preservado e acrescido pelos elementos da nova extensão e nenhuma definição da linguagem nativa deve ser alterada (GONÇALVES; ARAÚJO; CASTRO, 2018), prezando por uma extensão conservativa.

Portanto, o metamodelo de iStar 2.0 foi mantido, preservando assim todas as características originais da linguagem. A figura 22 apresenta o metamodelo criado para iStar4Safety. Note que os construtores representados pela cor amarela são originais da linguagem iStar 2.0 e não sofreram quaisquer alterações. Os novos construtores de iStar4Safety estão representados pela cor roxa. O metamodelo foi representado através do

diagrama de classes de UML e desenvolvido na ferramenta ASTAH[2], um ambiente para criação de modelos UML. A representação do metamodelo através de diagrama de classes da UML está de acordo com o padrão MOF 2.4.1 da OMG (OMG, 2013).

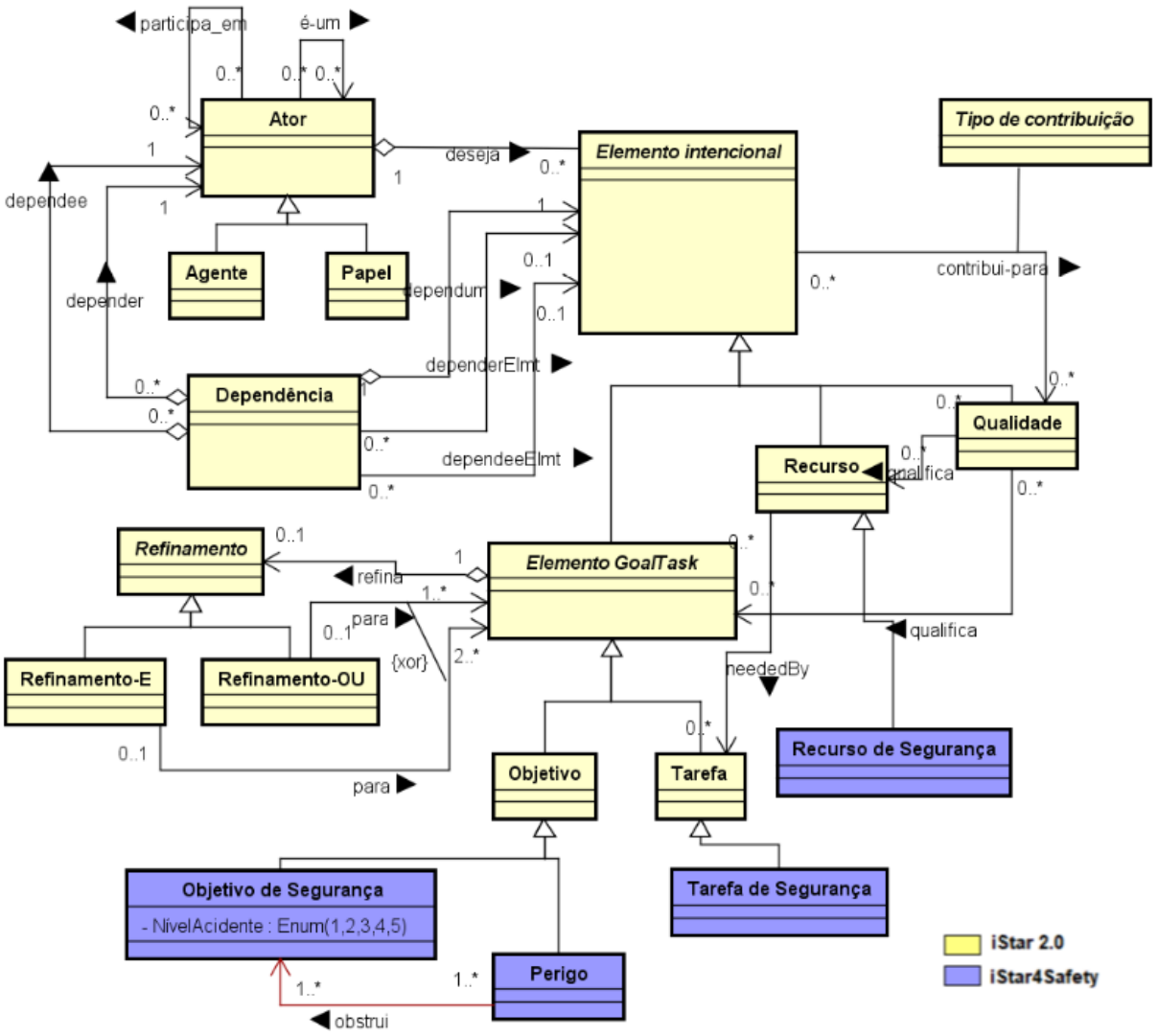


Figura 22 – Metamodelo de iStar4Safety.

Fonte: (Autora, 2018).

A parte amarela do metamodelo, representa os construtores nativos de iStar 2.0 (DALPIAZ; FRANCH; HORKOFF, 2016). Conforme pode ser visualizado, a classe **Ator** tem como especialização as classes **Agente** e **Papel**, representando assim, os três possíveis tipos de atores de iStar.

Já a classe abstrata **Elemento Intencional** tem como especialização todos os elementos intencionais de iStar: recurso, qualidade, tarefa e objetivo. A classe abstrata **Elemento GoalTask** que generaliza as classes **Objetivo** e **Tarefa**, relaciona-se com a classe

[2] Link para ferramenta Astah: <http://astah.net/>

abstrata **refinamento**, mostrando que seus elementos filho podem ser refinados, através dos relacionamento E/OU.

A classe **Qualidade** pode qualificar, conforme o metamodelo indica, elementos da classe **Recurso** e da classe **Elemento goaltask**. Qualidades podem ainda receber contribuições de quaisquer elementos filhos da classe abstrata **Elemento Intencional**.

Os recursos se relacionam à classe **tarefa** através da relação de ***NeededBy***. Já a classe dependência cria relacionamentos entre os atores e os elementos intencionais.

Os elementos de iStar4Safety estão representados pela cor roxa. Pode-se visualizar que a classe **Recurso de Segurança** é uma especialização de um recurso. Os elementos **Tarefa de segurança** serão, por sua vez, especializações da classe **Tarefa**. Esses dois elementos, associados ou não (considerando a necessidade do recurso de segurança estar associado à uma Tarefa de segurança ou Tarefa), podem solucionar os perigos, sendo portanto, estratégias de segurança.

As classes **Objetivo de Segurança** e **Perigo** são especializações da classe **Objetivo**. Os elementos da classe **Perigo** podem **obstruir** construtores **objetivos de segurança**. Elementos da classe **Perigo** relacionam-se entre si, e quando isso acontece, são ditos ser causas do perigo ao qual estão associados. Os elementos da classe **Objetivo de Segurança**, possuem a propriedade de nível do acidente que pode assumir um de cinco valores de acordo com o nível de acidente que acontecerá se o objetivo de segurança for obstruído.

4.3.2 Regras de Validação

Deve-se ter em mente que os metamodelos podem ter restrições quanto à representação de todas as especificidades da linguagem, sendo necessárias geralmente descrições adicionais em linguagem natural ou através de alguma linguagem formal. Como iStar4Safety é uma extensão conservativa, as restrições de iStar 2.0 foram mantidas e são apresentadas na seção 4.3.2.1 em linguagem natural, conforme relatado em (DALPIAZ; FRANCH; HORKOFF, 2016). As restrições adicionais relacionadas à extensão, encontram-se em 4.3.2.2 e são também apresentadas em linguagem natural.

4.3.2.1 Regras de Validação de iStar 2.0

As regras de validação definem as restrições que não puderam ser representadas através do metamodelo. A seguir apresentamos as regras de validação para a modelagem com iStar 2.0. Tais regras foram definidas por Dalpiaz, Franch e Horkoff (2016) complementando as características e restrições do metamodelo.

- O relacionamento **é-um** aplica-se somente a pares de **papéis** ou **atores**;
- Não deverá haver ciclos do relacionamento ***é-um***;
- Não deverá haver ciclos do relacionamento **participa-em**;

- Um par de **atores** podem ser relacionados por até um link de atores: não é possível conectar dois **atores** via **é-um** e **participa-em**;
- Em uma **dependência** D, se o ***dependerElmt*** x existe, então o **ator** que quer x é o mesmo **ator** que é o ***depender*** de D;
- Em uma **dependência** D, se o ***dependeeElmt*** y existe, então o **ator** que quer y é o mesmo **ator** que é o ***dependee*** de D;
- Os elementos ***depender*** e ***dependee*** de uma **dependência** deverão ser **atores** diferentes;
- Para uma **dependência**, se um ***dependerElmt*** x existe, então x não pode ser refinado ou contribuído;
- O relacionamento de refinamento não deve levar à ciclos de refinamento (Ex.: G refinado por **OU** para G1, e G1 refinado por **OU** para G, G refinado por **OU** para G, etc.);
- O relacionamento entre elementos intencionais (**contribuiPara, qualifica, *neededBy*, refina**) aplicam-se somente à elementos que são desejados pelo mesmo **ator**;
- Um elemento intencional e uma **qualidade** podem ser relacionados ou por um relacionamento **contribuiPara** ou por um relacionamento **qualifica**, mas não por ambos; e
- Não é possível que uma **qualidade** contribua consigo mesma.

4.3.2.2 Regras de Validação de iStar4Safety

À fim de complementar as informações do metamodelo da extensão iStar4Safety, seguem listadas as regras de validação que devem ser atendidas ao utilizá-la. Ressaltamos que todas as regras definidas para iStar 2.0 em Dalpiaz, Franch e Horkoff (2016) e listadas na subseção acima se aplicam também à extensão, já que a mesma é conservativa.

- Regra de Validação 01 - Os construtores da extensão iStar4Safety não podem ser elementos ***dependum***;
- Regra de Validação 02 - Um **objetivo de segurança** só pode ser refinado ou por **objetivos de segurança** ou por **perigos**, não podendo ser pelos dois elementos ao mesmo tempo;
- Regra de Validação 03 - Somente **perigos**-raiz podem estar relacionados à **objetos de segurança**;

- Regra de Validação 04 - **Perigos**-filhos se associam à seus pais pelos relacionamentos de refinamento;
- Regra de Validação 05 - Somente **perigos**-folha podem se associar à estratégias de segurança: **tarefas de segurança** associadas ou não à um **recurso de segurança**;
- Regra de Validação 06 - Todo **perigo**-folha deve ter pelo menos uma estratégia de segurança associada à ele;
- Regra de Validação 07 - **Recursos de segurança** associam-se pelo relacionamento ***neededBy*** à **tarefas de segurança** ou ainda à **tarefas**; e
- Regra de Validação 08 - Toda estratégia de segurança deve estar relacionada ao **ator** que a realiza, exceto aquelas realizáveis pelo próprio **ator** que possui o elemento em sua fronteira.

4.3.3 Sintaxe Concreta de iStar4Safety

A sintaxe concreta define elementos gráficos ou textuais usados para a criação dos elementos do modelo através de editores de modelagem ou mesmo manualmente (BRAMBILLA; CABOT; WIMMER, 2012).

A definição da sintaxe concreta de iStar4Safety foi escolhida visando a simplicidade de modelagem, ou seja, construir extensões com a menor quantidade possível de novos construtores e representações à fim de não prejudicar o uso da linguagem e propor representações gráficas simples que possibilitem a modelagem sem ferramenta, quando necessário (GONÇALVES et al., 2018c). Ao mesmo tempo, proporcionar a modelagem satisfatória de requisitos iniciais de Sistemas Críticos de Segurança à fim de possibilitar que os conceitos do documento da Análise Preliminar de Segurança sejam modelados.

Portanto, optou-se por utilizar o mecanismo de extensão de peso-leve, usado também em extensões da linguagem UML. O estereótipo textual, que trata-se do nome do construtor entre os símbolos « », é a opção peso-leve mais utilizada para representar nós especializados, conforme pode ser visto no trabalho de Gonçalves et al. (2018a).

Para a criação dos construtores gráficos, foram utilizadas as mesmas representações dos nós pais, associadas à estereótipos com o nome do construtor especializado. Essa prática é também usada em UML para representar a especialização de nós (GONÇALVES et al., 2018a) representando assim um novo conceito através do nó especializado.

Outra diferença entre os construtores da linguagem nativa e da extensão iStar4Safety está na seleção das cores: os elementos de segurança são representados pela cor rosa e o elemento **perigo** pela cor vermelha.

Portanto, como resultado de todo o processo de definição de construtores gráficos, as seguintes representações foram escolhidas e sua representação gráfica pode ser vista na figura 23.

- Objetivo de Segurança: O elemento (A) da figura 23 representa uma especialização do elemento **objetivo**, conforme pode ser visualizado no metamodelo. Portanto, a representação do pai, ou seja, **objetivo**, é mantida com a adição de um estereótipo **«Objetivo de Segurança»** («SafetyGoal»). A cor do construtor é rosa;
- Perigo: O elemento (B) da figura 23 representa uma especialização do elemento **objetivo**, conforme pode ser visualizado no metamodelo. Portanto, a representação do pai, ou seja, **objetivo**, é mantida com a adição de um estereótipo **«Perigo»** («Hazard»). A cor do construtor é vermelha;
- Tarefa de Segurança: O elemento (C) da figura 23 representa uma especialização do elemento **tarefa**, conforme pode ser visualizado no metamodelo. Portanto, a representação do pai, ou seja, **tarefa**, é mantida com a adição de um estereótipo **«Tarefa de segurança»** («SafetyTask»). A cor do construtor é rosa;
- Recurso de Segurança: O elemento (D) da figura 23 representa uma especialização do elemento **recurso**, conforme pode ser visualizado no metamodelo. Portanto, a representação do pai, ou seja, **recurso**, é mantida com a adição de um estereótipo **«Recurso de Segurança»** («SafetyResource»). A cor do construtor é rosa; e
- Link Obstrui: O elemento (E) da figura 23, representa o link **obstrui** (obstructs) que foi adicionado tendo em vista a necessidade de relacionar os construtores **perigo** e **objetivo de segurança**, permitindo assim a representação de **perigo** como um elemento não-intencional, em uma linguagem que é de natureza intencional.

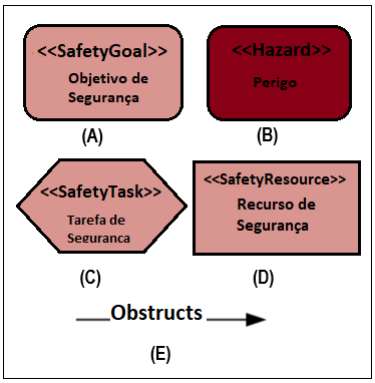


Figura 23 – Construtores gráficos adicionados pela extensão *iStar4Safety*

Fonte: (Autora, 2018).

Somado a isso, durante todo o processo de criação da sintaxe gráfica, foram realizadas buscas no catálogo de extensões de iStar, atividade proposta por Gonçalves et al. (2018b). Essa busca verificou se os princípios de claridade semiótica, proposto por Moody, Heymans e Matulevičius (2010) e baseado na teoria de símbolos de Goodman (1968), estavam sendo atendidos. O resultado foi satisfatório e esse subprocesso considerado concluído.

Por fim, apresenta-se nos próximos tópicos um maior detalhamento sobre cada construtor criado para iStar4Safety:

4.3.3.1 Objetivo de Segurança

Objetivos de segurança representam objetivos críticos ao sistema sendo modelado. É de fundamental importância para a segurança do sistema que tais objetivos possam ser analisados e tratados. Portanto, os **atores** desejam a satisfação do **objetivo de segurança** pois sua não-realização pode causar um acidente.

Um **objetivo de segurança** pode ser refinado em mais **objetivos de segurança**. Isso acontece caso seja necessário especializar o construtor, e pode ser feito através de refinamentos **E** ou **OU**. Para um exemplo dessa situação em relação à bomba de infusão, considere o **objetivo de segurança**: "Receber dosagem correta de insulina". É sabido que, uma dosagem errada pode ter dois contextos: não receber uma dosagem maior que o desejado, ou seja "Não sofrer overdose", e também não receber uma dosagem menor que o desejado, ou seja, "Não sofrer *underdose*". Logo, em iStar4Safety, esse refinamento é representado conforme modelado na figura 24.

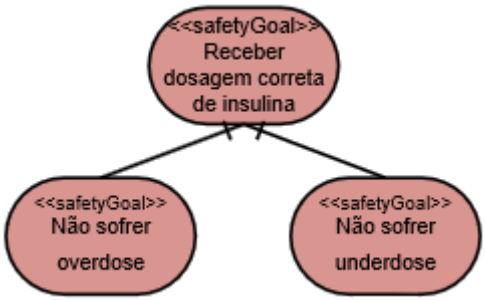


Figura 24 – Exemplo de refinamento de objetivo de segurança em mais objetivos de segurança.

Fonte: (Autora, 2018).

4.3.3.2 Link Obstrui

O link **obstrui** é um relacionamento n-ário, podendo relacionar o pai à um ou mais filhos. O link **obstrui** indica que o **perigo** é um obstáculo à satisfação de um **objetivo de segurança**. Essa relação não indica qual o nível de obstrução provocada pelo **perigo**. A relação possível entre o link **obstrui** e os demais construtores é apresentada na tabela 6

.

Tabela 6 – Relação links x Construtos de *iStar* e *iStar4Safety*

		Ponta da seta apontando para				
		Objetivo	*Qualidade*	*Tarefa*	*Recurso*	*Perigo*
Links começando de	*Objetivo*	*Refinamento*	*Contribuição*	*Refinamento*	*n/a*	*n/a*
	Qualidade	*Qualificação*	*Contribuição*	*Qualificação*	*Qualificação*	*Qualificação*
	Tarefa	*Refinamento*	*Contribuição*	*Refinamento*	*n/a*	*Refinamento*
	Recurso	*n/a*	*Contribuição*	*NeededBy*	*n/a*	*Refinamento*
	Perigo	*Obstrui*	*Contribution*	n/a	*n/a*	*Refinamento*

4.3.3.3 Perigo

Perigos são obstáculos à realização de um **objetivo de segurança**. Logo, caso um **perigo** obstrua um **objetivo de segurança**, cria-se uma situação de potencial acidente.

Um **objetivo de segurança** pode ser obstruído por **um** ou mais **perigos**. Diz-se que o **perigo obstrui** o **objetivo de segurança**, já que o **perigo** é um obstáculo à realização do **objetivo de segurança**. A figura 25 representa os **perigos** que obstruem o **objetivo de segurança** "Não sofrer overdose". Assim, caso aconteça um "fluxo livre de insulina", uma "configuração maior de insulina", "aplicações excessivas de *bolus*"ou "falha no sistema de entrega", o **objetivo de segurança** "Não sofrer overdose"pode ser obstruído, podendo causar o acidente relacionado à superdosagem de insulina.

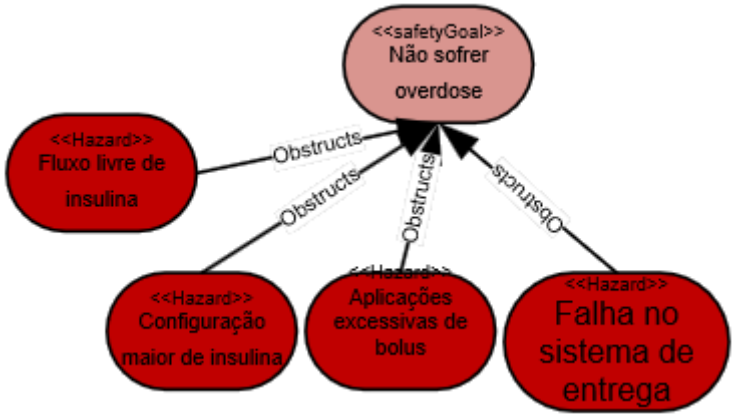


Figura 25 – Exemplo de relacionamento entre objetivos de segurança e perigos.

FFonte: (Autora, 2018).

O construtor **perigo** também é usado para representar as **causas dos perigos** e **condições ambientais**. Conforme já mencionado, causas ambientais são um tipo de causa de perigos. Causas de perigos são também **perigos**. Portanto os perigos-filho serão as causas do perigo-pai. No exemplo da figura 26, o **objetivo de segurança** "Não sofrer infecção"pode ser obstruído pelo **perigo** "Bomba exposta à substância nova". Esse **perigo** por sua vez tem como causas a "bomba compartilhada"ou "bomba não esterilizada corretamente".

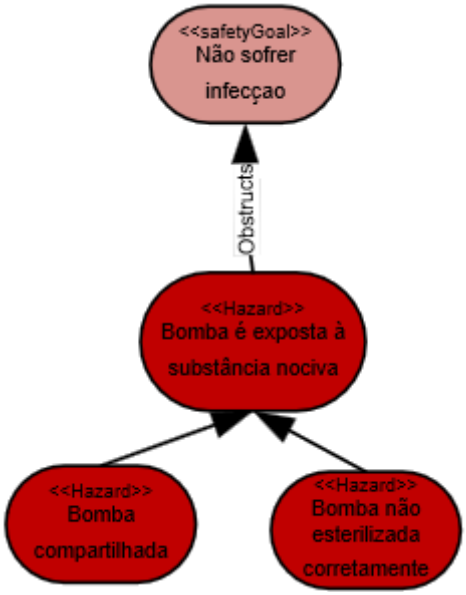


Figura 26 – Exemplo de perigo e refinamento de perigos em causas de perigo.

Fonte: (Autora, 2018).

4.3.3.4 Tarefa de Segurança

O construtor **tarefa de segurança** é um dos construtores que podem formar uma estratégia de segurança. É usado para representar ações concretas que serão tomadas à fim de mitigar o **perigo**. Os perigos-folha serão tratados, visto que o tratamento aplicado aos filhos, consequentemente tratará o pai.

A figura 27 representa o refinamento OU de um **perigo** em **tarefas de segurança**. Nesse exemplo, o perigo-folha "Bomba sem proteção de descarga elétrica"pode ser tratado por ao menos uma das duas ações concretas, **tarefas de segurança**, apresentadas. Como a **tarefa de segurança** "Receber suporte especializado"é concretizada pelo ator "Fornecedor da bomba", deve existir uma relação de dependência entre os dois atores, que no caso é: o "paciente"depende do "fornecedor da bomba"para "Receber suporte especializado". A outra **tarefa de segurança**, "Interromper uso da bomba"é realizado pelo próprio paciente, portanto não possui dependências com outros atores.

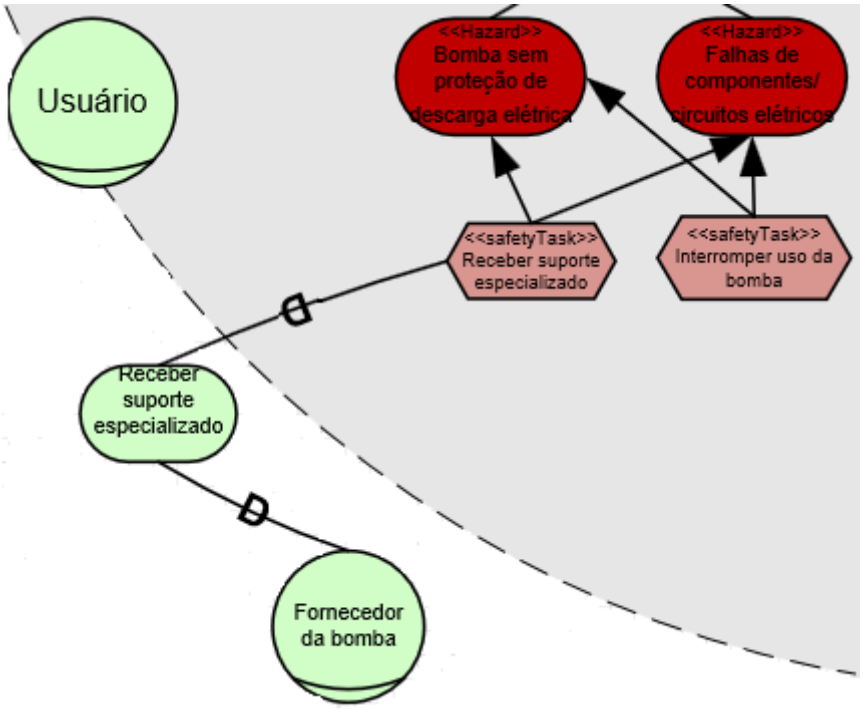


Figura 27 – Exemplo de refinamento de perigos em tarefas de segurança.

Fonte: (Autora, 2018).

Já a figura 28 representa a possibilidade de um **recurso de segurança** ser necessário para a realização da **tarefa de segurança**. Observe que uma solução de mitigação para o **perigo** "Reservatório quebrado"será a **tarefa de segurança** "Trocar Reservatório"que precisa do ativo "Reservatório"disponível para ser concretizada.

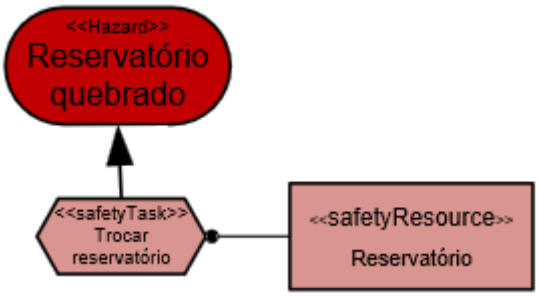


Figura 28 – Exemplo de refinamento de perigos em tarefas de segurança e recursos de segurança

Fonte: (Autora, 2018).

Explica-se mais o construtor **recurso de segurança** na próxima seção.

4.3.3.5 Recurso de Segurança

O construtor **recurso de segurança** irá modelar algum ativo que tem a característica de ser crítico para a mitigação de **perigo** no Sistema Crítico de Segurança modelado. Como

já apresentado o construtor pode ser associado à uma **tarefa de segurança**.

É importante ressaltar que, devido a um ativo ser um objeto concreto, normalmente uma instância de algo, caso seja definido como seguro, o recurso será desse tipo em todo o modelo. Ou seja, uma vez declarado como **recurso de segurança**, o recurso será desse tipo em todo o modelo, exceto na posição de *dependum*, visto que a modelagem SD não é contemplada pela extensão.

4.3.4 Análise da Qualidade da Linguagem iStar4Safety

Após a criação da modelagem, é necessário verificar se a linguagem proposta está adequada. Para isso podemos verificar:

- **Completude**: A linguagem iStar4Safety é capaz de modelar todos os conceitos de segurança identificados na seção 4.2.2. Os conceitos estão representados no metamodelo. A linguagem também inclui restrições que definem a sintaxe concreta da mesma. iStar4Safety foi definida seguindo os conceitos de claridade semiótica propostos em (MOODY; HEYMANS; MATULEVIČIUS, 2010);
- **Consistência**: Verificou que todos os conceitos estão representados tanto nos níveis de sintaxe abstrata quanto concreta. iStar4Safety tem a representação de todos os conceitos em ambos os níveis;
- **Presença de nós e links da sintaxe padrão de iStar**: Também verificou-se que estão presentes os construtores nativos de iStar 2.0. iStar4Safety manteve todos os construtores padrão da linguagem nativa iStar 2.0, ou seja, a extensão é conservativa; e
- **Conflitos:** Nesse item é verificado se existe conflitos entre construtores da extensão proposta nesta dissertação e as extensões já existentes. Essa verificação foi feita levando em consideração as extensões presentes no catálogo CATIE (GONÇALVES et al., 2018b). Observamos que iStar4Safety não possui nenhum conflito com construtores de extensões já existentes.

O artefato gerado pela verificação desse subprocesso, encontra-se no anexo A.

Para facilitar a adoção da nova linguagem de modelagem iStar4Safety, é necessário que seja disponibilizada uma ferramenta para confecção de modelos que incluam seus construtores.

4.3.5 Uma Ferramenta de Modelagem para iStar4Safety

O subprocesso 3.5, apresentado na figura 29, propõe o desenvolvimento de uma ferramenta de modelagem para apoiar a modelagem de sistemas com a extensão criada. Os passos indicados para o subprocesso foram todos seguidos.

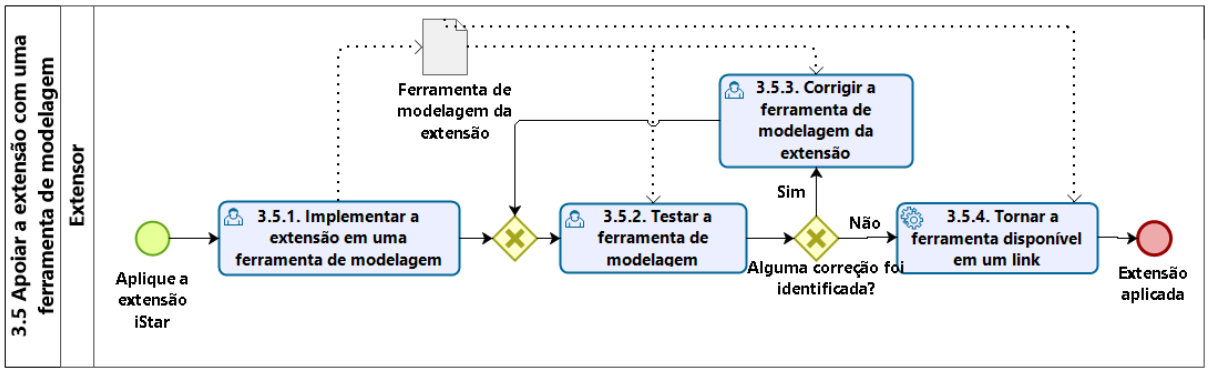


Figura 29 – Subprocesso 3.5 do PRISE - Suporte da extensão com uma ferramenta de modelagem

Fonte: Adaptado de Site Prise (GONÇALVES, 2018).

Foi utilizada uma ferramenta já existente, piStar (PIMENTEL; CASTRO, 2018). piStar foi então estendida sendo criada uma versão que possibilita a modelagem com iStar4Safety. Essa nova versão da ferramenta foi definida como piStar-4Safety[3].

Conforme pode ser visualizado pela figura 30, somente o construtor **perigo** (*hazard*) e o link **obstrui** (*obstructs*) foram adicionados à paleta de construtores da ferramenta.

Para a criação dos construtores de segurança, ou seja, **Objetivo de segurança**, **tarefa de segurança** e **recurso de segurança**, um botão chamado **alternar para segurança** (*Toggle Safety*) foi inserido, conforme mostrado na figura 31. Portanto, para modelagem dos construtores **Objetivo de segurança**, **tarefa de segurança** e **recurso de segurança**, os construtores originais devem ser inseridos (**Objetivo**, **tarefa** ou **recurso**) e o modelador deve clicar sobre o botão "Toggle Safety"para assim convertê-lo no respectivo construtor de segurança.

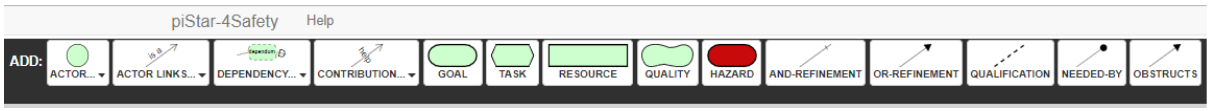


Figura 30 – Paleta de construtores de PiStar-4Safety

Fonte: Ferramenta PiStar-4Safety.

O nível de impacto do acidente deve ser inserido nas propriedades do construtor **objetivo de segurança**. A propriedade é **NívelImpactoAcidente** ("*accidentImpactLevel*"), que pode ser vista na figura 31, que já será anexada ao elemento assim que o mesmo se tornar de segurança.

[3] A ferramenta piStar-4Safety encontra-se em: <http://www.cin.ufpe.br/~jhcp/pistar/4safety/>.

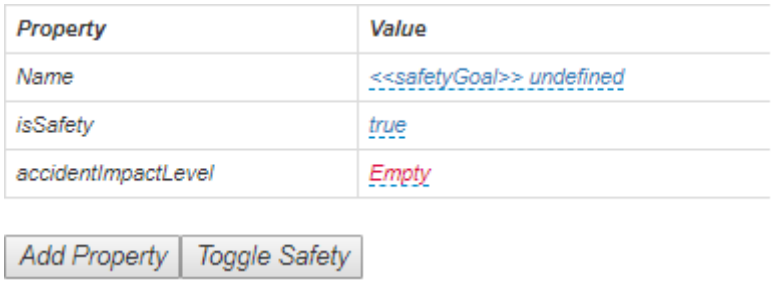


Figura 31 – Botão *Toggle Safety* e propriedade *accidentImpactLevel* de piStar-4Safety

Fonte: Ferramenta PiStar-4Safety

A próxima subseção apresenta o passo a passo definido para utilização da extensão *iStar4Safety* na ferramenta *PiStar-4Safety*.

4.3.6 Diretrizes para Modelagem com *iStar4Safety*

Para a criação do modelo, utilizando a extensão iStar4Safety foi definido que o seguinte conjunto de passos deve ser executado, visando a completude dos requisitos :

- **Diretriz 1** - Modelar as funcionalidades relacionadas à iStar padrão (parte do tipo não-segurança):
  1. Leia a descrição dos requisitos do seu sistema; e
  2. Modele todas as funcionalidades de seu sistema, não se preocupando ainda com a modelagem de requisitos de segurança. **Obs.**: Atente-te que, nesse momento, alguns **objetivos** que deveriam ser **objetivos de segurança** podem vir à ser inseridos. O próximo passo tratará essa questão.
- **Diretriz 2** - Modelar o **objetivo de segurança**: Um **objetivo de segurança** é um objetivo crítico que, caso não ocorra como descrito pode provocar um acidente.
  1. Leia a descrição de requisitos do seu sistema (a parte a ser modelada diz respeito à requisitos críticos de segurança);
  2. Se o **objetivo de segurança** já tiver sido modelado como **objetivo**, altere-o para ser representado como **objetivo de segurança**. Se estiver usando a ferramenta PiStar-4Safety, altere o **objetivo** clicando sobre "***Toggle Safety***";
  3. Se o **objetivo de segurança** ainda não estiver no modelo, insira-o;
  4. Caso necessário refina o **objetivo de segurança** em mais objetivos de segurança; e
  5. Insira, o valor da propriedade "Nível de Impacto do Acidente" no **objetivo de segurança**. (Se estiver usando a ferramenta piStar-4Safety, essa propriedade é do tipo *String*). Você deve inserir o valor desejado, que pode ser um dos cinco abaixo:

  - Catastrófico (MAIOR IMPACTO);
  - Muito Severo;
  - Considerável;
  - Menor; e
  - Sem Efeito (SEM IMPACTO) .

  Obs1: O nível de impacto do acidente do **objetivo de segurança**-pai será o maior nível de impacto dos **objetivo de segurança**-filhos.

  Obs2: Existem outras classificações possíveis e utilizadas para o nível de impacto do acidente. A classificação proposta é uma sugestão, mas pode ser alterada.

- **Diretriz 3** - Inserir todos os **perigos** para o **objetivo de segurança** modelado.

  Um perigo à um **objetivo de segurança** é um perigo que pode fazer com que o **objetivo de segurança** não se concretize. Podendo consequentemente causar um acidente.

  1. Leia a descrição de requisitos do seu sistema (a parte a ser modelada diz respeito à requisitos críticos de segurança);
  2. Verifique quais são os perigos que obstruem o **objetivo de segurança**;
  3. Insira o construtor **perigo**; e
  4. Ligue o **perigo** ao **objetivos de segurança** que ele obstrui através do link **obstrui**.

  Conforme já apresentado na subseção 3.2.4.2, **perigos** podem ser identificados através de técnicas, como por exemplo a de Ericson et al. (2015), já que a identificação desses construtores não é um processo trivial, porém crucial para uma definição de conceitos de segurança preliminares do sistema, que seja completos e de qualidade.

- **Diretriz 4** - Identificar todas as causas para cada perigo identificado.

  Uma causa de um perigo é algo que causa o perigo. Logo, causas nada mais são que **perigos**.

  1. Leia a descrição de requisitos do seu sistema (a parte a ser modelada diz respeito à requisitos de segurança);
  2. Verifique quais são as causas para cada **perigo** identificado no passo anterior;
  3. Insira o construtor **perigo**;
  4. Ligue o **perigo**-filho ao **perigo**-pai através dos links **E** ou **OU** (o link deve partir do filho); e

5. Refine, caso necessário, os **perigos**-filho em novos **perigos**, ou seja, suas causas.

- **Diretriz 5** - Definir a estratégia de mitigação para cada **perigo**-folha.

  Estratégias de mitigação são ações concretas para mitigar perigo ou suas consequências. Portanto, **tarefas de segurança** e/ou **recursos de segurança** devem, combinados ou não, modelar as estratégias para: (1) eliminar o **perigo**, (2) reduzir o **perigo**, (3) controlar o **perigo**, ou ainda (4) minimizar os danos criados pelo **perigo**.

  1. Leia a descrição de requisitos do seu sistema (a parte a ser modelada diz respeito à requisitos críticos de segurança);
  2. Verifique quais são as estratégias para cada **perigo**-folha modelado;
  3. Insira o construtor desejado.(No caso da ferramenta *PiStar-4Safety* Tarefa associada ou não a um Recurso, e altere-os clicando sobre "***Toggle Safety***"; e
  4. Associe a estratégia de segurança ao **perigo** relacionado, através dos links **E** ou **OU** (partindo da estratégia).

- **Diretriz 6** - Associar a estratégia de mitigação a um ator que ficará responsável por sua realização.

  Como estratégias de segurança são ações concretas, as mesmas devem ser associadas ao ator que as concretiza. Pode haver o caso em que as estratégias são realizadas pelo próprio ator. Caso sejam realizadas por outro ator, devem ser ligadas através de relações de dependência.

  1. Leia a descrição de requisitos do seu sistema (a parte a ser modelada diz respeito à requisitos críticos de segurança);
  2. Identifique quais são os atores responsáveis pela realização de cada **tarefa de segurança** ou pela disponibilização do **recurso de segurança**; e
  3. Modele essa dependência (Lembre-se que construtores de iStar4Safety não são modelados fora dos limites do ator, ou seja, não podem ser elementos *dependum*).

- Repita as diretrizes de **1** à **6** até modelar todos os seus requisitos de segurança.

  Recomenda-se que o processo de modelagem de requisitos de segurança utilizando a extensão iStar4Safety seja realizado de forma iterativa, visto a necessidade de prezar por completude dos modelos criados. Porém, o modelador pode optar por realizar as diretrizes de outra forma. Ao final da modelagem, recomenda-se fortemente que

seja verificada a completude do modelo criado. A próxima subseção apresenta os itens que devem ser verificados à fim de atender à completude da análise.

4.3.6.1 Conferindo a Completude do Modelo Criado com a Extensão *iStar4Safety*

Verifique se os itens abaixo foram contemplados na sua modelagem:

- Todos os **objetivos de segurança** tem a propriedade **Nível de Impacto do Acidente** com um dos valores predefinidos para esse campo inserido;
- Todos os **objetivos de segurança**-raiz tem um ou mais **perigos** associados. Caso não sejam raiz, os **objetivos de segurança** são refinados por um ou mais **objetivos de segurança**;
- Todos os **perigos**-raiz estão ligados à um ou mais **objetivos de segurança**-folha através do link **obstrui**;
- Todos os **perigos**-folha estão ligados à uma ou mais estratégias de segurança;
- Todas as estratégias de segurança estão associadas ao ator que as realizam, menos no caso de serem realizadas pelo próprio ator em que estão definidas; e
- Um recurso que é de segurança será em todo modelo desse tipo, exceto fora dos limites dos atores (*dependum*), em que não podem ser modelados construtores iStar4Safety.

4.4 VALIDAÇÃO E AVALIAÇÃO DA EXTENSÃO ISTAR4SAFETY

A fim de realizar a validação e avaliação da extensão iStar4Safety desenvolvida no subprocesso 4.3, os passos apresentados na figura 32 foram executados e são descritos nessa seção.

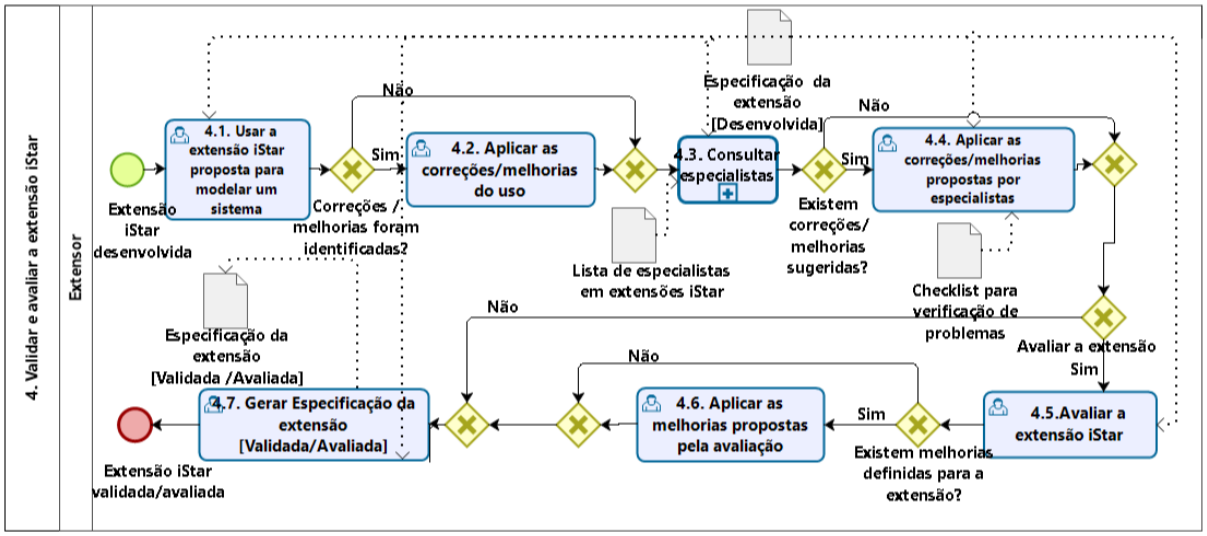


Figura 32 – Subprocesso 4 do PRISE - Validação e avaliação da extensão de *iStar*

Fonte: Adaptado de Site Prise (GONÇALVES, 2018).

4.4.1 Modelagem de um Sistema Usando a Extensão *iStar4Safety*

Para validar e avaliar a extensão criada, o subprocesso 4 (quatro) inicia-se preconizando a necessidade de modelagem de um sistema que não seja trivial e seja real. Para esse fim, foi modelado o Sistema de Bomba de Infusão de insulina. Para tanto, foi utilizado a descrição do Sistema Crítico de Segurança da Bomba de Infusão de Insulina, seção 2.1. Observe que este Sistema Crítico de Segurança já havia sido previamente modelado em iStar 2.0 na seção 4.1.3. Contudo a linguagem nativa não foi capaz de representar todos os conceitos de segurança necessários.

A modelagem do Sistema de Bomba de Insulina foi realizada de forma iterativa. Ela foi iniciada durante a fase do desenvolvimento do metamodelo (vide seção 4.3.1). Tal prática se mostrou bastante útil para refinarmos o metamodelo e a representação gráfica. Além disso, permitiu um *feedback* maior por parte dos especialistas que foram consultados. A ferramenta usada para a criação dos modelos iniciais foi o aplicativo de criação de diagramas Microsoft Visio © 2016[4].

Após a implementação da extensão iStar4Safety na ferramenta de modelagem piStar, começou-se a utilizar a nova versão, piStar-4Safety, para as demais modelagens. Por esse motivo, os construtores criados com a nova ferramenta estão com os estereótipos em inglês, saber:

- Objetivos de segurança são representados por um elemento **objetivo** da cor rosa, com o estereótipo **safetyGoal**;

[4] A ferramenta Visio pode ser encontrada em: <https://products.office.com/pt-br/visio/flowchart-software>

- Tarefas de segurança são representados por um contrutor **tarefa** da cor rosa, com o estereótipo **safetyTask**;
- Recursos de segurança são representados por um elemento **recurso** da cor rosa, com o estereótipo **safetyTask**;
- Perigos são representados por um elemento **objetivo** da cor vermelha, com o estereótipo **hazard**; e
- Os links **obstrui** são representados por links **obstructs**.

Portanto, após a criação da extensão foram gerados os modelos SD e SR da bomba de infusão de insulina. A figura 33 apresenta o modelo SD,e a figura 34 apresenta o modelo SR da bomba de infusão de insulina modelados com iStar4Safety.

Na figura 33, que apresenta o modelo SD do Sistema de Bomba de Infusão de Insulina pode-se verificar o relacionamento entre os atores do sistema. Observe que, na representação do SD sem iStar4Safety, apresentado na seção 4.1.3, não estão presentes alguns elementos *dependum*. Como dependências entre Paciente e Controlador da bomba, os novos *dependum* são: "Ter configurações verificadas"(objetivo), "Receber alertas de falhas"(Tarefa), "Receber alerta sobre alimentação"(Tarefa), "Valores de insulina em tela serem limitados"(Tarefa). Já como novas dependências entre o Paciente e o Fornecedor da Bomba, temos como novos elementos *dependum*: "Receber suporte de falhas"(Tarefa), "Receber treinamento de uso"(Tarefa) e "Receber suporte especializado". Isso se deve à necessidade de representar qual o ator realiza/fornece a estratégia de segurança para mitigação da causa do perigo.

O modelo SR desse Sistema Crítico de Segurança está representado pela figura 34, que representa o modelo completo. Para permitir uma melhor visualização, cada ator foi separado em imagens distintas e são apresentados nas próximas subseções.

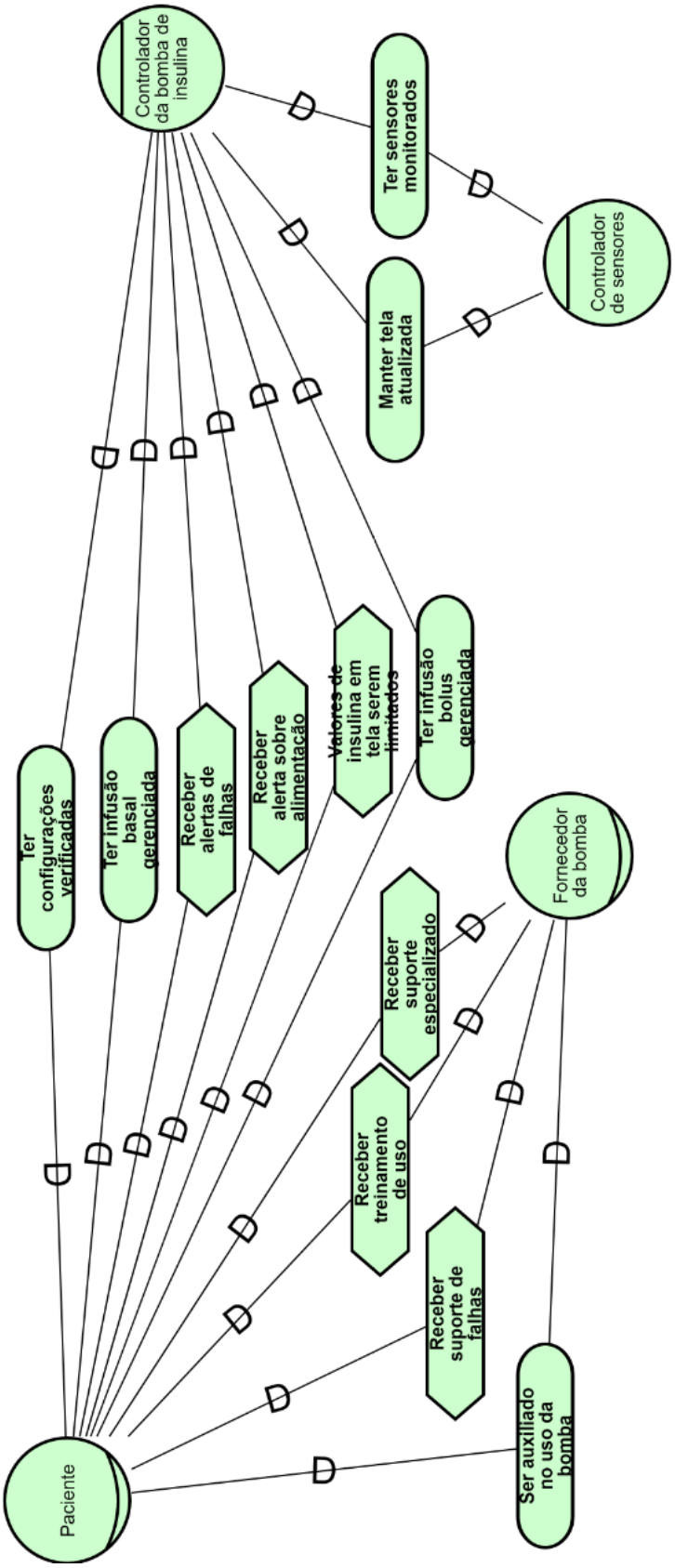


Figura 33 – Modelo SD da bomba de infusão de insulina modelada com safety.

Fonte: (Autora, 2018).

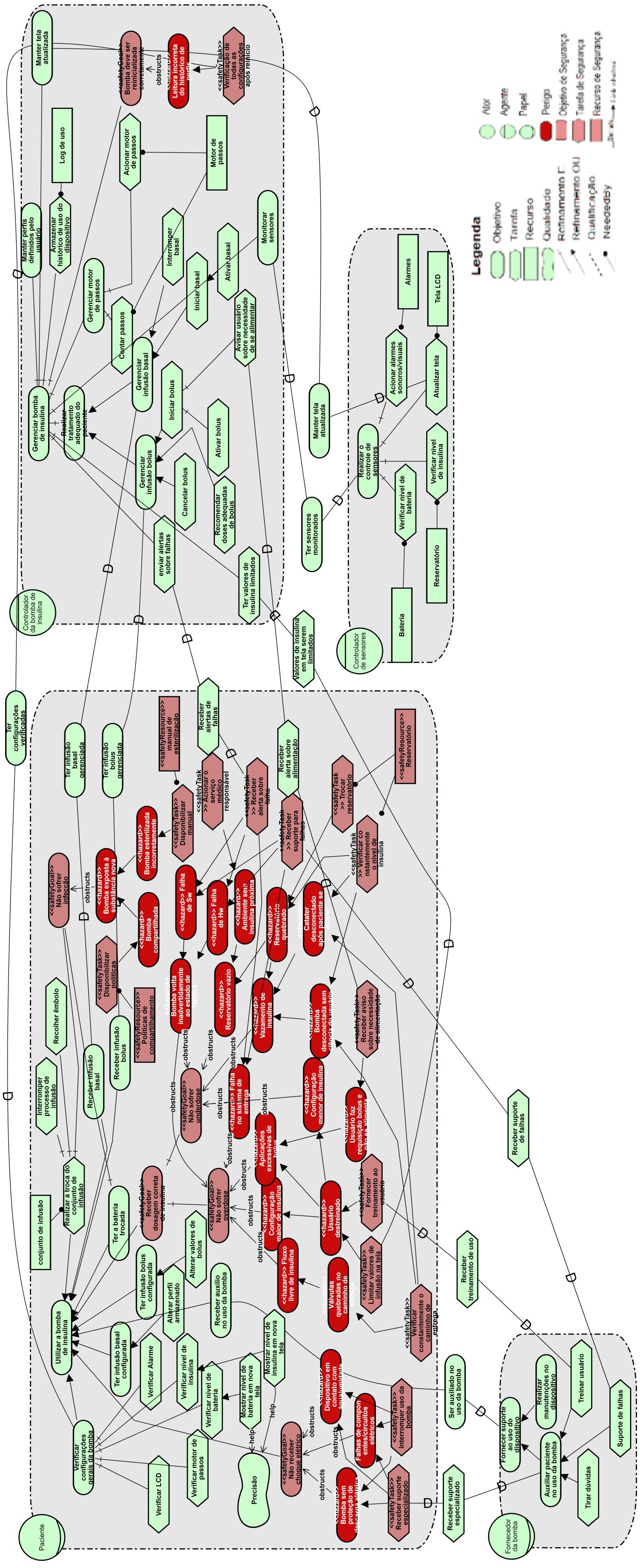


Figura 34 – Modelo SR da bomba de infusão de insulina modelada com safety.

Fonte: (Autora, 2018).

4.4.1.1 Ator Paciente

A figura 35, representa o ator **Paciente** da Bomba de Infusão de Insulina. Observe que os requisitos de segurança se concentram nesse ator na modelagem de requisitos iniciais desse sistema. Isso se deve ao fato de ele ser o principal usuário da bomba de insulina e assim, depender de outros atores que mitigam alguns dos perigos definidos, o que justifica também o aumento dos elementos *dependum*, conforme já mencionado.

À modelagem do modelo SR do ator sem requisitos de segurança, apresentado na subseção 4.1.3.1, foram acrescentados os elementos de segurança. O **objetivo** "Receber dosagem correta de insulina"se tornou um **objetivo de segurança**. Outro novo **objetivo de segurança** adicionado ao ator foi o de "Não receber choque elétrico".

Os **perigos** que podem impedir que o paciente não receba choque elétrico são os de: "Bomba sem proteção de descarga elétrica", "Falhas de componentes/circuitos elétricos"e "Dispositivo em contato com água ou umidade". Esses três **perigos** podem ser mitigados pelas **tarefas de segurança** de "Receber suporte especializado"(que é esperado do ator **Fornecedor**) ou "Interromper o uso da bomba".

O **objetivo de segurança** "Receber dosagem correta de insulina"significa que o paciente não deve sofrer *underdose* e nem overdose. Os **perigos** que podem causar o acidente de overdose são: "Fluxo livre de insulina", "Configuração maior de insulina"do que o apropriado, "Aplicações excessivas de *bolus*"e falhas no sistema de entrega da insulina.

O **perigo** de "Fluxo livre de insulina"tem como causa "válvulas quebradas no caminho de entrega". A forma de mitigar esse **perigo** é: verificar constantemente o caminho de entrega de insulina ou limitar em tela os valores de infusão (eliminando o perigo).

Já o **perigo** de "Configuração maior de insulina"pode ser causado por uso da bomba por "Usuário destreinado". As soluções para tratar esse **perigo** podem ser limitar os valores de insulina em tela ou fornecer treinamento ao usuário, o que depende do ator **fornecedor**.

O perigo de "Aplicações excessivas de *bolus*"pode ser causado por usuário destreinado ou pelo fato do paciente solicitar a infusão *bolus* e não se alimentar. A forma de mitigar esse último perigo seria o Sistema de Controle da Bomba enviar alertas sobre a necessidade de alimentação.

Quanto ao **perigo** de "Falha no sistema de entrega"que pode ocasionar tanto uma overdose quanto o **perigo** de *underdose*, pode ser mitigado através de envio de alerta de falha do controlador da bomba para o paciente.

Já quanto ao **objetivo de segurança** "Não sofrer *underdose*", os **perigos** que podem obstruí-lo são: "Falha no sistema de entrega", "Vazamento de insulina"o reservatório de insulina estar vazio, ou ainda as configurações da bomba de insulina voltarem inadvertidamente ao estado de fábrica.

À fim de mitigar o perigo de bomba voltar ao estado de fábrica inadvertidamente, deve-se tratar suas causas que são os **perigos** de falha de software ou de falha de hardware.

Para mitigar tais **perigos**, o **paciente** deve receber alertas sobre falhas pelo **controlador da bomba** ou ainda "receber suporte para falhas"pelo **fornecedor**.

O perigo do "Reservatório vazio"pode ser causado pelo ambiente em que o paciente está não ter insulina em acesso próximo, ou ainda pelo reservatório estar quebrado. À fim de mitigar sa consequências de não ter insulina disponível nas proximidades o **paciente** deve "Acionar o serviço médico responsável". Caso o reservatório esteja quebrado, deve-se trocar o mesmo por outro.

O **perigo** de "Vazamento de insulina"pode ser causado pelos **perigos** da "Bomba desconectada sem ciência do usuário", ou pelo cateter desconectado após paciente se mover", ou ainda pelo reservatório estar quebrado. Para mitigar os **perigos** da bomba ou o cateter se desconectarem, deve-se verificar constantemente o caminho de entrega e o nível de insulina.

Ao realizar as trocas da bateria e do conjunto de infusão, o **paciente** deseja não sofrer infecção, que é um objetivo de **segurança**. O **perigo** que podem obstruir esse objetivo é o fato da bomba ser exposta à alguma substância nociva. As causas para esse perigo podem ser o fato da bomba ser compartilhada ou a bomba ser esterilizada incorretamente. Para mitigar o perigo de compartilhamento da bomba, deve-se disponibilizar o **recurso de segurança** "Políticas de compartilhamento". Já o perigo de esterilização errada pode ser mitigado pela disponibilização de um manual de esterilização.

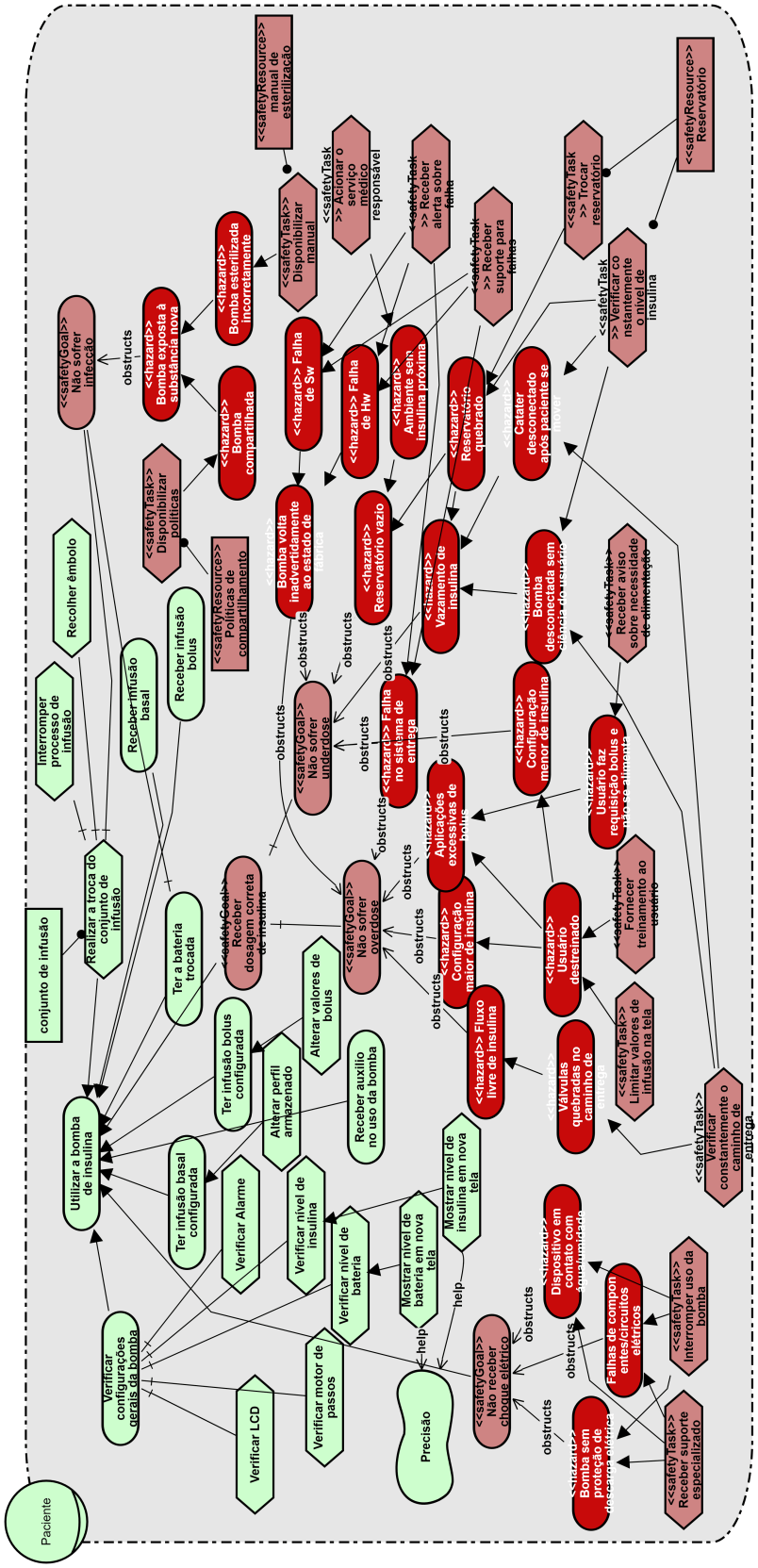


Figura 35 – Ator paciente - Parte 1 do modelo SR, com iStar4Safety, do exemplo do Sistema da Bomba de Insulina, desenvolvido com a ferramenta piStar-4Safety.

Fonte: (Autora, 2018).

4.4.1.2 Ator Controlador da Bomba de Insulina

A figura 36 apresenta a parte do modelo SR relativa ao ator Controlador da Bomba. A modelagem apresentada na subseção 4.1.3.2 foi alterada pela adição de pouco construtores.

Foram adicionadas ao gerenciamento da bomba as **tarefas** de "enviar alertas sobre falhas"e "Ter valores de insulina limitados". Além disso, foi adicionada a **tarefa** de avisar o usuário sobre a necessidade do mesmo se alimentar ao iniciar a infusão *bolus*.

Outro novo objetivo, é o **objetivo de segurança** "Bomba deve ser reinicializada corretamente". O **perigo** que pode obstruir esse objetivo é uma "Leitura incorreta do histórico de perfis". Para mitigar esse perigo, é **tarefa de segurança** do controlador verificar todas as configurações da bomba após qualquer reinicialização realizada.

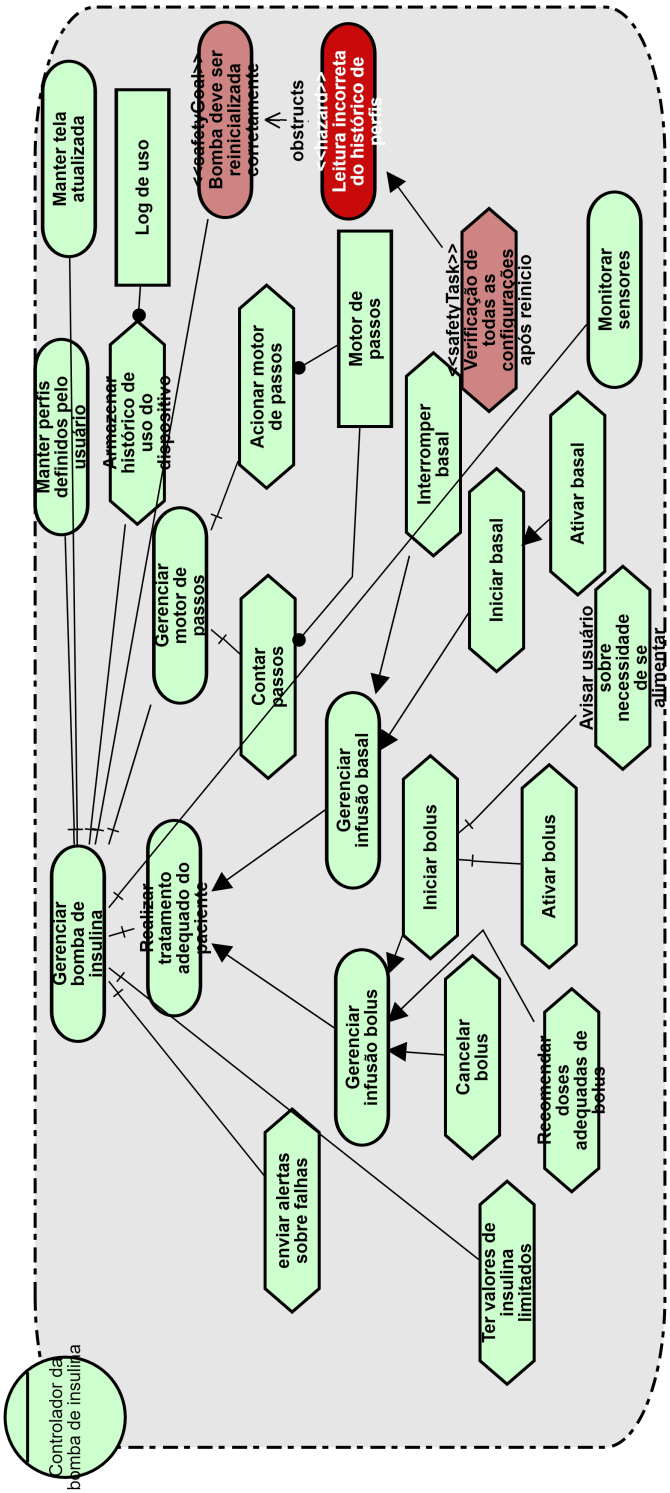


Figura 36 – Ator Controlador da Bomba de Insulina - Parte 2 do modelo SR, com iStar4Safety, do exemplo do Sistema da Bomba de Insulina, desenvolvido com a ferramenta piStar-4Safety.

Fonte: (Autora, 2018).

4.4.1.3 Ator Controlador de Sensores

A figura 37 por sua vez, mostra a parte do modelo SR relacionada ao ator **controlador de sensore**s da bomba. Conforme pode ser visto na subseção 4.1.3.3, o ator continua realizando as mesmas funções que anteriormente.

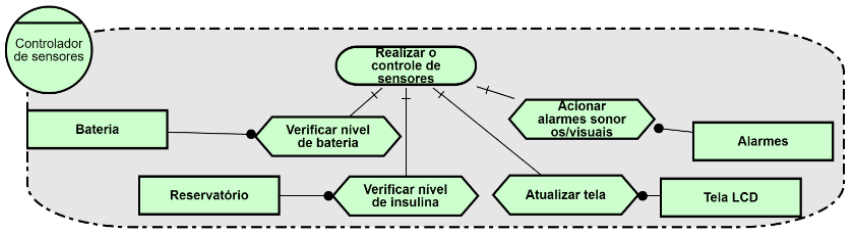


Figura 37 – Ator Controlador de sensores - Parte 3 do modelo SR, com iStar4Safety, do exemplo do Sistema da Bomba de Insulina, desenvolvido com a ferramenta piStar-4Safety.

Fonte: (Autora, 2018)..

4.4.1.4 Ator Fornecedor da Bomba

Representamos na figura 38, a parte do modelo SR relativa ao ator **fornecedor da bomba**.

O modelo mostra que comparado ao modelo sem iStar4Safety (vide subseção 4.1.3.4) somente a tarefa "Suporte de falhas"foi adicionada à fronteira do ator. Isso se deve ao fato do **fornecedor** precisar fornecer esse novo suporte específico, agora de falhas, para o ator **paciente**.

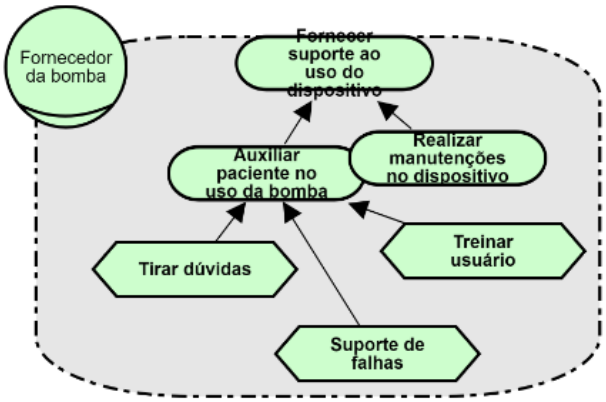


Figura 38 – Ator Fornecedor da bomba - Parte 4 do modelo SR, com iStar4Safety, do exemplo do Sistema da Bomba de Insulina, desenvolvido com a ferramenta piStar-4Safety.

Fonte: (Autora, 2018).

Além da modelagem do Sistema de Infusão da Bomba de Insulina, o capítulo 5 apresenta a ilustração da modelagem com iStar4Safety de um Sistema de Cruzamento de Ferrovias, complementado a validação da extensão.

4.4.2 Avaliação da Extensão iStar4Safety

O subprocesso de desenvolvimento orienta, por fim, a realização da avaliação da extensão. O processo PRISE recomenda que essa avaliação seja realizada através de um experimento, estudo de caso ou survey.

O método empírico escolhido para avaliação de iStar4Safety foi o survey. Essa avaliação está detalhada no capítulo 6.

4.5 PUBLICAÇÃO NO CATÁLOGO CATIE DA EXTENSÃO ISTAR4SAFETY

Finalmente, a extensão deve ser publicada no catálogo CATIE de extensões criado para iStar4Safety (GONÇALVES et al., 2018b) conforme indicado pela Figura 39 que representa o subprocesso. O título, link da extensão, além do *e-mail* da criadora da extensão foram enviados à fim da extensão ser endossada.

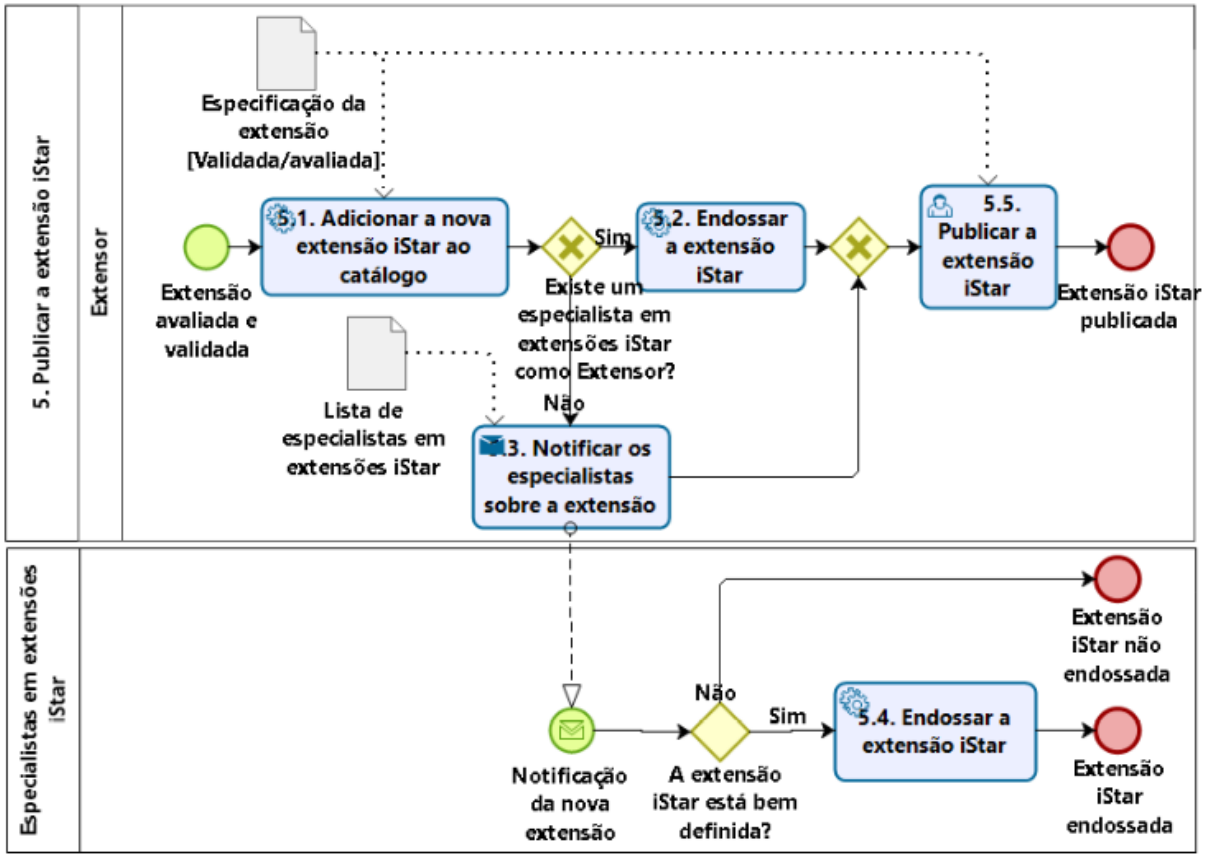


Figura 39 – Subprocesso 5 do PRISE - Publicação da extensão de *iStar*

Fonte: Adaptado de Site Prise (GONÇALVES, 2018).

Para que a publicação seja avaliada, é solicitado, conforme já mencionado, um documento descritivo da extensão. O documento disponibilizado foi o artefato final do processo,

a especificação da extensão[5], que apresenta de forma resumida todos os procedimentos realizados para o desenvolvimento de iStar4Safety.

Conforme indicado pelo processo, a extensão foi endossada e publicada no catálogo. Uma visão geral de iStarSafety no CATIE pode ser vista na figura 40.

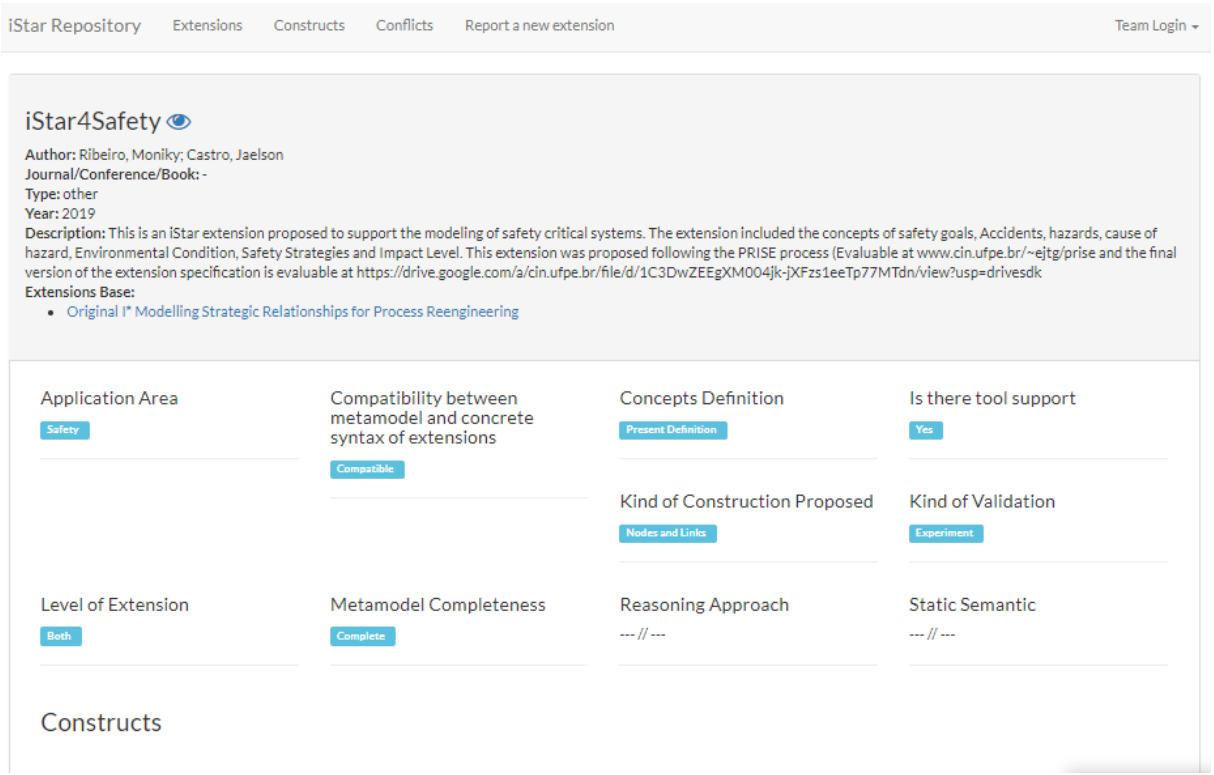


Figura 40 – Interface do catálogo CATIE representando que iStar4Safety foi publicada.

Fonte: Site do catálogo CATIE (GONÇALVES et al., 2018b).

4.6 CONCLUSÃO

Nesse capítulo foi apresentado o processo de criação de iStar4Safety. Primeiramente foi confirmada a necessidade de uma linguagem que possibilitasse a modelagem de requisitos de segurança. Partindo dessa necessidade, os conceitos relacionados à segurança e candidatos à fazer parte da extensão foram apresentados, tendo por base o trabalho de Vilela et al. (2017b). Um exemplo de um Sistema Crítico de Segurança, Sistema de Bomba de Infusão de Insulina, foi modelado somente com construtores de iStar 2.0. A modelagem, tanto do modelo SD como SR, permitiu a confirmação da necessidade de uma nova abordagem para modelagem de requisitos de segurança, já que não foi possível modelar perigos, causas de perigos e estratégias de segurança somente com iStar. Somado à isso, após uma busca realizada no catálogo CATIE, não foram encontradas extensões que modelassem segurança em iStar.

[5] A especificação da extensão encontra-se disponível em <https://drive.google.com/file/u/1/d/1C3DwZEEgXM004jk-jXFzs1eeTp77MTdn/view?usp=drivesdk>

O próximo passo realizado para o desenvolvimento de iStar4Safety foi a definição dos conceitos de segurança candidatos à integrar a extensão. A primeira atividade foi a busca por construtores de outras extensões que pudessem ser reusados por iStarSafety, no catálogo CATIE. Essa busca não retornou nenhum construtor que pudesse ser reusado. Seguindo esse processo, os conceitos candidatos foram definidos. Uma análise quanto à possibilidade de integrar conceitos necessários à nova extensão e construtores já existentes em iStar e KAOS foi realizada. Chegamos à conclusão de que a ideia de **obstáculos** usada em KAOS poderia ser adaptada para a extensão. Os obstáculos em iStar4Safety são representados por **perigos**, sendo estes portanto, obstáculos à realização dos **objetivos de segurança**. Foi definido que o conceito candidato **acidente** não seria representado explicitamente, devido à ser subentendido através dos construtores **perigo** e **objetivo de segurança**. Outra decisão foi a de que o contruto **perigo** representaria os conceitos de perigo, causas de perigo e condições ambientais. As estratégias de segurança serão representadas através dos construtores **tarefa de segurança** e **recurso de segurança** que são realizáveis de forma concreta por atores. Por fim, foi definido que os construtores **objetivo**, **tarefa** e **recurso** seriam especializados, sendo portanto, reutilizados.

No subprocesso de desenvolvimento, foi criado o metamodelo de iStar4safety. As regras de validação da linguagem foram elencadas. Na sequência foi apresentada a sintaxe concreta da extensão, com a definição de cada representação gráfica dos construtores criados. Além disso, os novos construtores foram apresentados em detalhes e exemplos modelados na ferramenta Microsoft Visio © 2016, foram apresentados. Uma análise de qualidade da linguagem, preconizada pelo PRISE foi apresentada. Logo após, uma ferramenta para modelagem de requisitos de segurança, piStar-4Safety, foi apresentada. Para finalizar esse subprocesso, um total de 6 (seis) diretrizes para a modelagem com iStar4Safety forma definidas, além de um conjunto de itens para facilitar a verificação da completude de modelos criados com a extensão.

O próximo subprocesso apresentado foi o de avaliação e validação da extensão. Na validação, o Sistema de Bomba de Infusão de Insulina foi novamente modelado, agora com iStar4Safety, e com o apoio da ferramenta piStar-4Safety. A linguagem iStar4Safety permitiu a modelagem de todos os conceitos definidos para a extensão. Uma segunda validação foi realizada, através da modelagem de um Sistema de Controle de Cruzamento de Ferrovias, e encontra-se no próximo capítulo. Para avaliar iStar4Safety foi realizado um estudo empírico, um survey, que encontra-se detalhado no capítulo 6.

Finalizando este capítulo, iStarSafety foi publicada no catálogo CATIE. O processo de publicação foi detalhado na seção 4.5.

No próximo capítulo, a segunda validação da extensão iStar4Safety é apresentada. Os requisitos de um Sistema de Controle de Cruzamento de Ferrovias são elencados e os modelos SD e SR são desenvolvidos com o apoio da ferramenta piStar-4Safety.

5 MODELAGEM DE UM SISTEMA DE CONTROLE DE CRUZAMENTO DE FERROVIA COM ISTAR4SAFETY

Esse capítulo apresenta a modelagem de um Sistema Crítico de Segurança, um Sistema de Cruzamento de Ferrovias, visando ilustrar a modelagem de requisitos com o uso da linguagem de modelagem iStar4Safety. Para de ilustrar a aplicabilidade da extensão iStar4Safety, apresenta-se nesse capítulo uma ilustração de um Sistema Crítico de Segurança, um *Level Crossing Control System* - Sistema de Controle de Cruzamento (LCCS).

O estudo de caso usado foi adaptado dos trabalhos de Schnieder (2004), Arabestani, Bitsch e Gayen (2004) que apresentam respectivamente um estudo de caso de um Sistema de Controle de Tráfego de Ferrovias e uma definição de especificação de requisitos de segurança através da notação UML.

Considerando a complexidade dos sistemas de cruzamento de ferrovias, o intuito não é esgotar todas as situações críticas, mas sim apresentar um subconjunto delas que possibilite demonstrar exemplo de uso da linguagem iStar4Safety.

O sistema modelado está no domínio de sistemas de controle de tráfego, sendo uma aplicação de controle à nível de travessia de uma ferrovia. Schnieder (2004) ressalta que o intuito não é apresentar o sistema real com toda a sua complexidade inerente, mas sim aplicar simplificações para criar um estudo de caso viável.

Para a modelagem desse sistema, na seção 5.1 são elencados os requisitos gerais relacionados ao Sistema de Cruzamento de Ferrovias. Na seção 5.2 os requisitos de segurança do sistema são apresentados. A modelagem do sistema encontra-se na seção 5.3, onde uma iteração com cada um das 6 (seis) diretrizes propostas para a modelagem com iStar4safety é elencada, à fim de mostrar o uso da extensão com as diretrizes e gerar os modelos SD e SR finais do sistema.

5.1 REQUISITOS GERAIS DO SISTEMA DE CRUZAMENTO DE FERROVIA

O sistema modelado, trata-se de um Sistema de Controle para Cruzamento de Ferrovia, onde há o cruzamento da linha férrea com a rodovia, sendo ponto de passagem do tráfego padrão de pessoas, automóveis, ciclistas, etc. A área de intersecção entre as duas vias é chamada zona de perigo, devido ao fato da criticidade envolvida na passagem de usuários e trens.

A travessia é composta da sinalização de alerta que é um semáforo constituído pelas luzes de alerta amarela e vermelha, e por barreiras de segurança, de um lado e do outro do cruzamento. Um protótipo para validação do sistema foi feito por Schnieder (2004) à fim de validar o sistema completo e foi utilizado aqui para ilustrar o sistema, veja a figura 41.

Figura 41 – Experimento simulando o Sistema de Controle de Travessia de Ferrovia criado para validação.

Fonte: Schnieder (2004).

As luzes amarelas indicam que a travessia deve ser evitada, mas pode ser feita. A luz vermelha indica que as barreiras estão fechadas para passagem e usuários da via não podem entrar no espaço de linha da ferrovia. Luzes desligadas indicam passagem livre. Mesmo com as barreiras abaixadas, é possível que o usuário adentre a linha férrea apesar de ser proibido pelas regulamentações existentes. O Sistema de Controle de Travessia deve monitorar o tempo de fechamento máximo para tentar evitar o perigo do usuário adentrar a rodovia quando impedido. Avisos sobre perigos devem ser mostrados na zona de perigo.

Quem controla as barreiras e o semáforo é o Sistema de Controle de Cruzamento. Assim que um trem se aproxima do nível de travessia,esse sistema é ativado . Quando ativado, o sistema realiza algumas ações à fim de permitir a travessia de forma segura. A saber:

1. Ativa a luz amarela do semáforo;
2. Depois de um tempo de 3seg troca a luz do semáforo para a cor vermelha;
3. Depois de 9seg começa a abaixar as barreiras de segurança;
4. Se em até 6seg as barreiras estiverem completamente abaixadas, o sistema envia um sinal de que a via está segura, permitindo a passagem do trem no cruzamento;
5. Assim que o trem terminar de passar pelo cruzamento, a área de perigo pode ser aberta para tráfego rodoviário. O semáforo é então apagado e as barreiras levantadas; e
6. Por fim, o sistema de controle volta para o estado de desativado.

Os componentes de interesse principal desse sistema serão: (1) Sistema de controle do trem (2) Sistema de Controle de Cruzamento na Via (3) Centro de operações (4) Usuários da via - pedestres, outros veículos, ciclistas. A comunicação entre esses sistemas deve ser possível. Outros atores adicionados à modelagem, serão a equipe de manutenção, necessários para auxiliar na segurança, e o operador do trem.

Em sistemas de controle de trens modernos, o trem se comunica por rádio com o sistema de controle de cruzamento, através de ordens de ativação baseadas na localização do trem.

Portanto, o sistema de controle do trem deverá detectar sua localização. Quando o trem estiver próximo de um nível de cruzamento, o sistema de controle do trem envia uma solicitação para que o sistema de controle de cruzamento ative as luzes do semáforo e abaixe as barreiras de segurança. O sistema do trem também irá ativar o modo de frenagem, que possibilitará a parada do trem caso aconteça alguma situação de falha.

Após receber o sinal de que o Sistema de Cruzamento está ciente de sua aproximação, o Sistema de controle do trem irá esperar por um tempo para procedimentos serem realizados e depois faz o envio de uma requisição á fim de verificar se tudo está certo para sua travessia. O Sistema de Controle de Cruzamento relata que o estado de travessia é seguro (caso realmente seja), o trem cancela o estado de frenagem e executa a travessia do cruzamento. Um sensor do sistema irá indicar que o trem passou por completo e o sistema de cruzamento irá então desligar o semáforo, levantar as barreiras de segurança e desativar o controle.

5.2 REQUISITOS DE SEGURANÇA DO SISTEMA DE CRUZAMENTO DE FERROVIA

Considerando a criticidade do sistema apresentado, que pode, no pior caso causar perdas de vidas e grandes perdas financeiras, uma análise dos perigos deve ser realizada à fim de mitigar possíveis falhas e situações perigosas.

Os usuários da via (pedestres, ciclistas, motoristas) desejam não serem atropelados ao atravessar a via. Uma colisão teria impacto catastrófico, para todos os envolvidos no acidente, podendo levar à morte e/ou grandes perdas financeiras. A colisão pode acontecer devido ao usuário estar na zona de perigo quando o cruzamento estiver fechado, porque ele não percebeu que está lá ou porque ele realmente ultrapassou a barreira intencionalmente. Para mitigar o perigo de estar na zona de perigo inconscientemente,a equipe de manutenção da ferrovia deve manter alertas visuais na zona de perigo.

Caso o usuário esteja na via conscientemente, esse perigo pode ser tratado por definir o tempo máximo de bloqueio da via, que vai mitigar uma espera indefinida, e o usuário deve ter acesso à legislação de travessia de cruzamentos.

O Sistema de Controle do Cruzamento deseja que o trem cruze a travessia somente com todas as condições de segurança favoráveis.

As falhas que serão consideradas (SCHNIEDER, 2004) serão:

- Falha da luz amarela do semáforo;
- Falha da luz vermelha do semáforo;
- Falha das barreiras;
- Falha dos sensores do veículo; e
- Falha na recepção de chegada do trem.

As luzes do semáforo e os sensores do veículo são monitorados constantemente e quaisquer defeitos devem ser rapidamente reportados para o Sistema de Controle de Cruzamento. Assim que recebe o estado de falha, o Sistema de Controle de Cruzamento envia o alerta da falha para o Centro de operações, que deve repará-la e depois deve reiniciar o Sistema de Controle de Cruzamento.

Falhas das barreiras serão detectadas pelo Sistema de Cruzamento caso as mesmas não abaixem ou levantem no tempo correto. O sistema irá declarar a falha e o Sistema de Controle de Cruzamento deverá ser reinicializado pelo Centro de operações.

Caso o Sistema de Controle do Trem não receber o status de seguro, enviado pelo Sistema de Controle de Cruzamento, devido à falhas de comunicação, é solicitado ao operador do trem que ele verifique e informe se é possível que o trem possa atravessar de forma segura o cruzamento. Se a resposta for afirmativa, os freios são cancelados e a travessia é permitida.

Caso a luz amarela esteja com defeito, a forma de mitigar esse perigo é habilitar a luz vermelha. Se a luz vermelha falhar, o procedimento de fechamento da via deve ser cancelado e ter a falha tratada pelo Centro de Operações.

Caso o sensor do veículo esteja com defeito, o Sistema de Controle de Cruzamento não deverá ser desativado após a passagem do trem, pois não haverá certeza de que o mesmo passou pelo cruzamento com sucesso. Para tratar esse perigo, será solicitado ao Centro de Operações que trate esse bloqueio.

Para evitar que as barreiras fiquem constantemente fechadas, o Sistema de Controle de Cruzamento irá bloquear qualquer requisição de passagem caso a luz vermelha esteja ativa por mais de 240seg. O bloqueio deverá ser tratado pelo Centro de Operações.

5.3 MODELANDO O SISTEMA DE CRUZAMENTO DE FERROVIA USANDO ISTAR4SAFETY

Para modelar os requisitos de segurança do Sistema de Cruzamento da Ferrovia, utilizando a linguagem iStar e a extensão iStar4Safety na ferramenta PiStar-4Safety, seguimos as diretrizes descritas na subseção 4.3.6:

5.3.1 Diretriz 1 - Modelar as Funcionalidades Relacionadas à iStar 2.0 Padrão

De acordo com a primeira diretriz, os requisitos gerais descritos para o Sistema de Cruzamento da Ferrovia foram modelados. A figura 42 representa o modelo de Dependência Estratégica (SD) do sistema, e a figura 43 representa o modelo de Raciocício Estratégico (SR). É importante salientar que nessa etapa os requisitos de segurança ainda não foram considerados.

Na figura 42, são modeladas as dependências entre os atores **Usuário da Via**, **Operador**, **Sistema Controlador de Cruzamento** e **Sistema Controlador do Trem**. Se comparado ao modelo SD final do sistema modelado com iStarSafety, pode-se observar que o modelo com a extensão possui mais elementos *dependum*, ou seja, dependências. Isso se deve ao fato de que as estratégias de segurança que ainda serão modeladas para mitigar perigos, devem ser associadas aos atores que as realizam, aumentando assim a quantidade de dependências.

Quanto à figura 43, os atores se relacionam e as relações de dependência são definidas em maiores detalhes. Este modelo possui os requisitos que não são do tipo segurança, relacionados ao Sistema de Controle de Cruzamento sendo modelado, conforme indicado pela diretriz 1.

Observe que também, em ambos os modelos, dois atores (Centro de Operações e Equipe de Manutenção da ferrovia) não foram modelados na versão inicial da modelagem de requisitos do sistema em questão. Isso se deve ao fato de que o papel deles aqui é auxiliar na mitigação de perigos. Logo, serão incorporados no momento em que os requisitos de segurança forem levados em consideração, visto que devem existem dependências entre os atores.

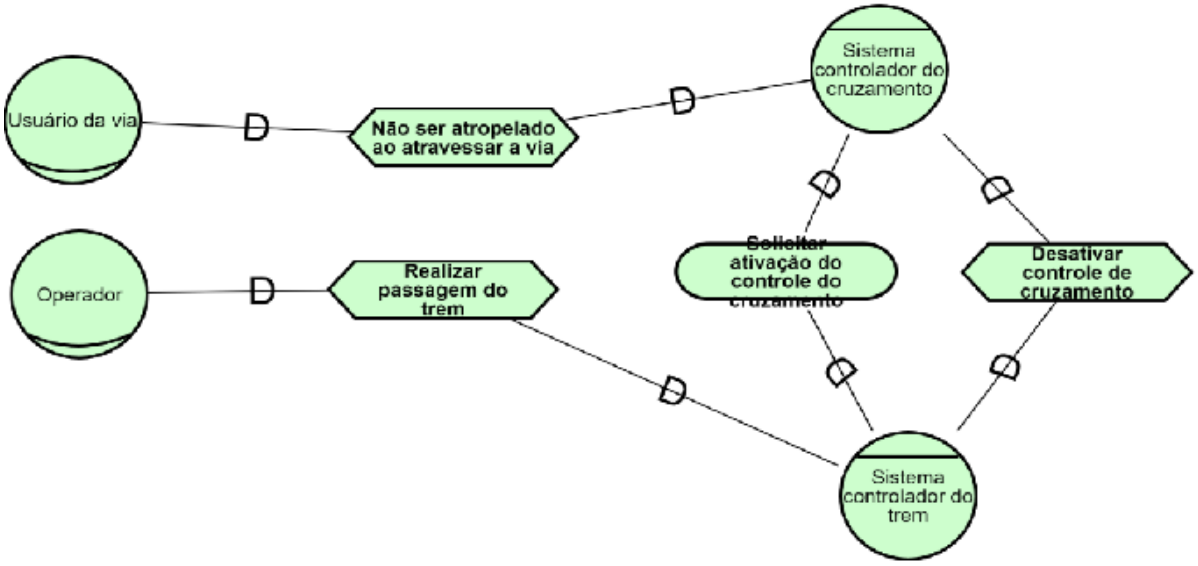


Figura 42 – Modelo SD do Sistema de Controle de Travessia da Ferrovia modelado sem aspectos de segurança.

Fonte: (Autora, 2018).

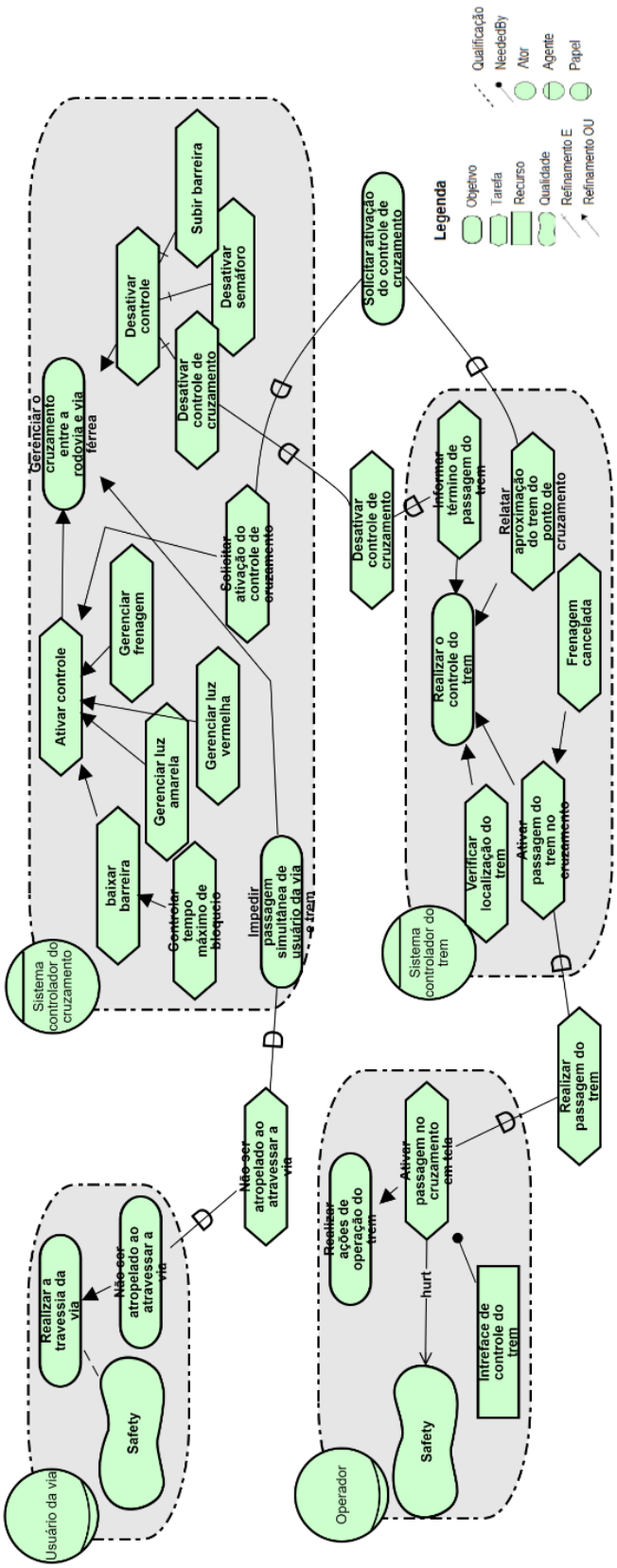


Figura 43 – Modelo SR do Sistema de Controle de Travessia da Ferrovia modelado sem aspectos de segurança.

Fonte: (Autora, 2018).

Para melhor entendimento do modelo, cada ator será detalhado na seção 5.3.7 que apresenta o resultado das iterações das diretrizes realizadas

5.3.2 Diretriz 2 - Modelar o Objetivo de Segurança

O próximo passo será modelar o **objetivo de segurança**. Para essa iteração foi escolhido o **objetivo de segurança "Não ser atropelado ao atravessar a via"**,sendo esse o requisito de segurança modelado, que está relacionado ao ator Usuário da via. Como o objetivo já está no modelo, iremos atualizar o seu atributo para **objetivo de segurança** e acrescentar o nível de impacto **Catastrófico** nas suas propriedades. O objetivo não foi ainda refinado em mais **objetivos de segurança**. A figura 44 mostra a modelagem gerada com o objetivo de segurança modelado.

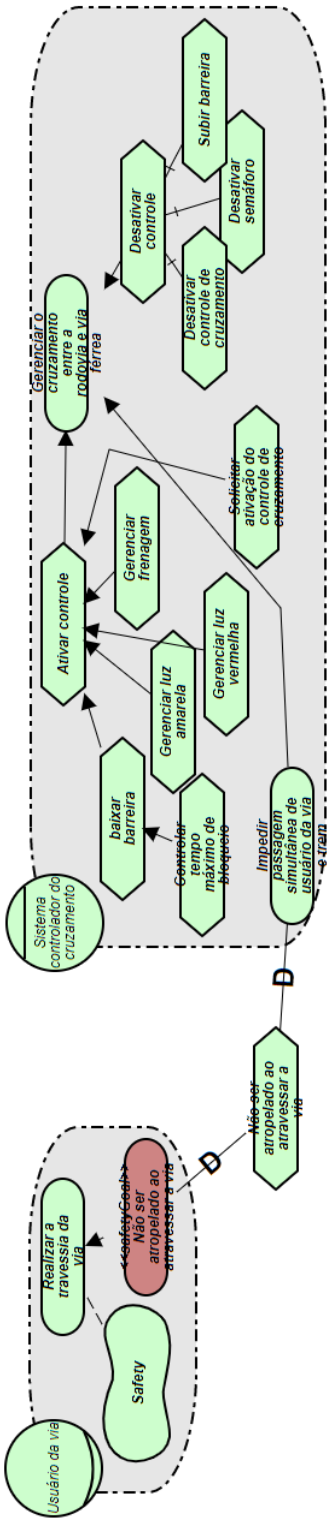


Figura 44 – Uso da diretriz 2 para modelagem de requisitos de segurança do Sistema de Controle de Travessia da Ferrovia com iStar4Safety.

Fonte: (Autora, 2018).

5.3.3 Diretriz 3 - Inserir Todos os Perigos para o Objetivo de Segurança Modelado

O próximo passo consiste em inserir todos os perigos que podem obstruir a concretização do**objetivo de segurança**. Os obstáculos para o **objetivo de segurança "Não ser atropelado ao atravessar a via"**é o perigo do usuário estar na zona de perigo com o cruzamento estando fechado. A modelagem resultante é mostrada na figura 45, onde o objetivo de segurança e o perigo que o bloqueia de ser concretizado foram inseridos.

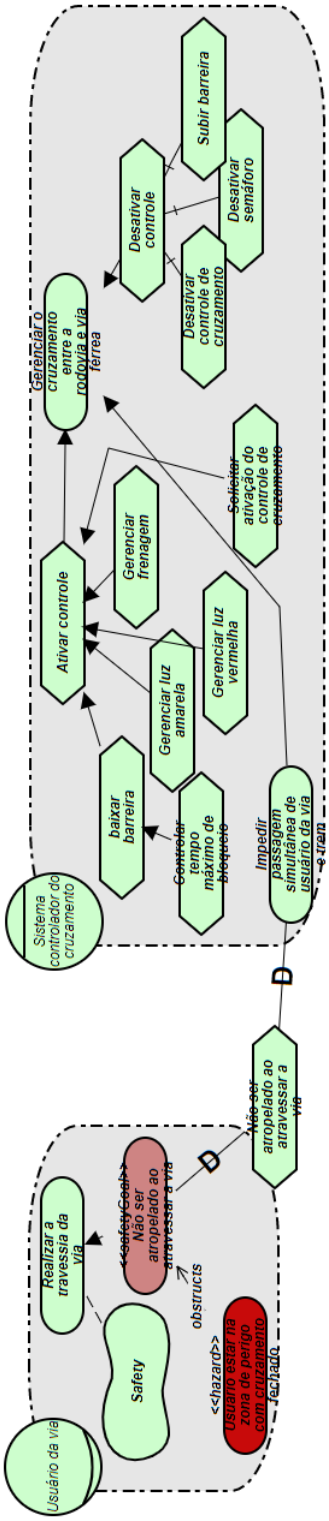


Figura 45 – Uso da diretriz 3 para modelagem de requisitos de segurança do Sistema de Controle de Travessia da Ferrovia com iStar4Safety.

Fonte: (Autora, 2018).

5.3.4 Diretriz 4 - Identificar Todas as Causas para cada Perigo Identificado

Nesse momento, todas as causas de **perigos** modelados no passo anterior devem ser modelados. As causas para o perigo do usuário estar na zona de perigo com cruzamento fechado podem ser: (1) usuário não percebe que está na zona de perigo, ou (2) usuário ultrapassa barreiras de segurança conscientemente. A figura 46 mostra essa modelagem, na qual percebe-se a inserção de novos perigos-filho indicando a causa do perigo que bloqueia o objetivo de segurança.

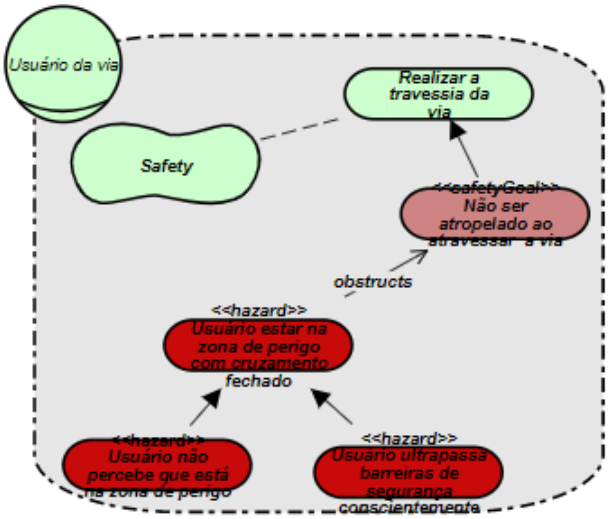


Figura 46 – Uso da diretriz 4 para modelagem de requisitos de segurança do Sistema de Controle de Travessia da Ferrovia iStar4Safety.

Fonte: (Autora, 2018).

5.3.5 Diretriz 5 - Definir as Estratégias de Mitigação para cada Perigo-folha

Para atender à diretriz 5 as estratégias de mitigação para cada perigo-folha devem ser definidas e associadas por refinamento **E/OU** aos **perigos** que são mitigados por elas. A figura 47 representa a modelagem das estratégias de mitigação que incluem: ser alertado por alertas visuais, inserindo-os na zona de perigo, para mitigar o perigo do usuário estar inconscientemente no ambiente possibilitando que o mesmo se oriente por tais alertas. Caso ele esteja conscientemente na zona de perigo, deverá ser disponibilizado o documento da legislação de travessia de cruzamentos de linhas férreas para que o usuário posso acessá-lo bem como ter uma definição de tempo máximo de bloqueio da via definido.

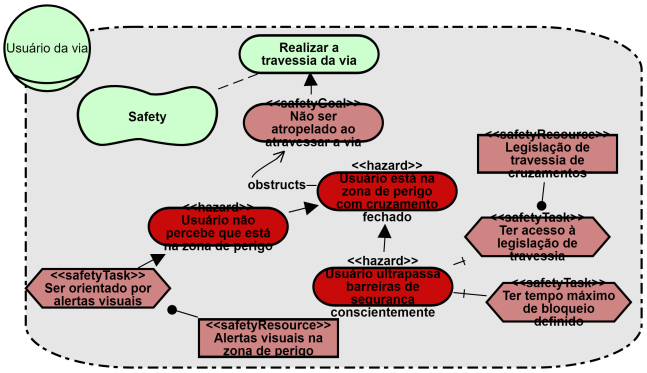


Figura 47 – Uso da diretriz 5 para modelagem de requisitos de segurança do Sistema de Controle de Travessia da Ferrovia com iStar4Safety.

Fonte: Fonte: (Autora, 2018).

5.3.6 Diretriz 6 - Associar a Estratégia de Mitigação a um Ator que Ficará Responsável por sua Realização

Para definir de quais atores se depende para a concretização das estratégias de segurança, essa diretriz orienta à associar tais estratégias aos seus respectivos executores. Logo, será inserido o ator **Equipe de Manutenção da Ferrovia** que é o responsável por inserir alertas visuais na zona de perigo. Já o Sistema de Controle da Ferrovia será o responsável por definir o tempo de bloqueio máximo. É tarefa do usuário estar consciente da legislação vigente de travessia de cruzamentos.

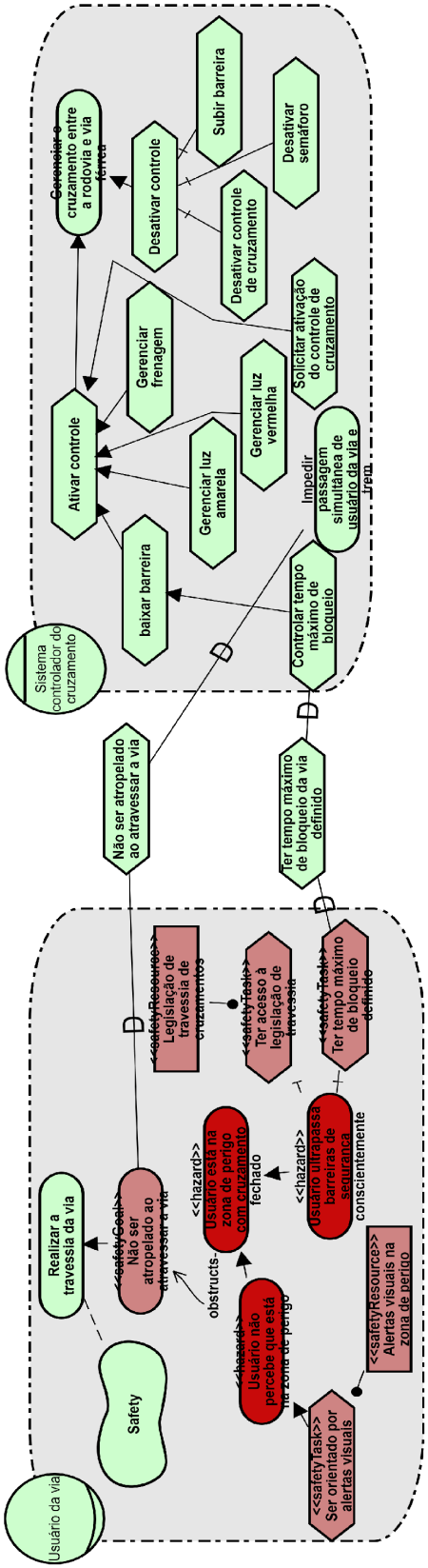


Figura 48 – Diretriz 6 para modelagem de requisitos de segurança do Sistema de Controle de Travessia da Ferrovia com iStar4Safety.

Fonte: (Autora, 2018).

5.3.7 Resultado das Iterações do Passo a Passo

Após a realização das outras iterações para inserção de toda a modelagem de requisitos de segurança, o diagrama SD, representado pela figura 49 foi gerado. Nesse modelo de Dependência estratégica percebe-se a adição de mais seis elementos "dependum", a saber:

- O recurso "alertas visuais na zona de perigo";
- O objetivo "Ter passagem ativada manualmente";
- E quatro tarefas: (1) "Ter tempo máximo de bloqueio da via definida", (2) "Ter bloqueio tratado", (3) "Ter falha reparada"e (4) "Ser reinicializado".

O objetivo: "Não ser atropelado ao atravessar à via", se tornou um **objetivo de segurança** e está na fronteira do ator usuário da via, não sendo representado mais como elemento "dependum".

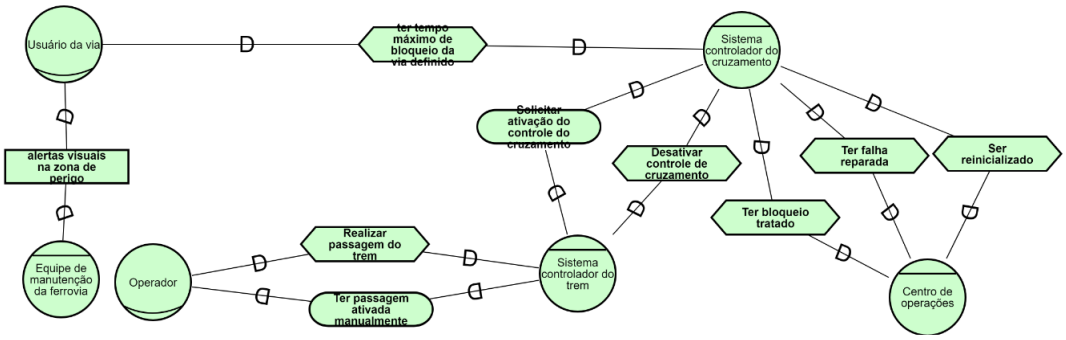


Figura 49 – Modelo SD do Sistema de Controle de Travessia da Ferrovia modelado com iStar4Safety.

Fonte: (Autora, 2018).

O detalhamento do sistema isto é, o modelo SR, foi gerado e é apresentado pela figura 50. Cada ator será detalhado, logo depois.

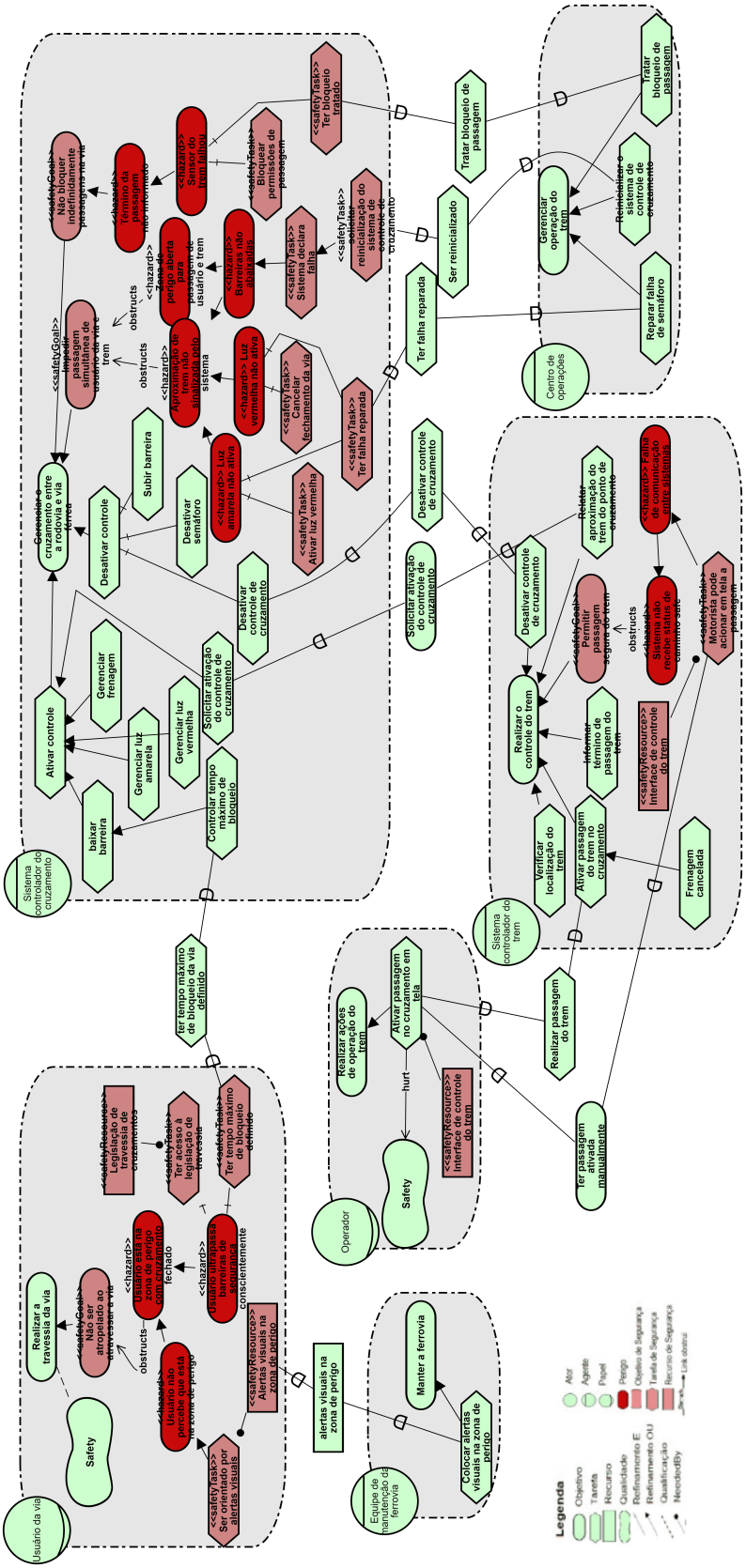


Figura 50 – Modelo SR do Sistema de Controle de Travessia da Ferrovia modelado com iStar4Safety.

Fonte: (Autora, 2018).

5.3.7.1 Ator Usuário da Via

A figura 51 mostra o ator **Usuário da Via** e sua fronteira. O **objetivo** de não ser atropelado ao atravessar a via, se tornou um **objetivo de segurança** na modelagem com iStar4Safety.

O perigo que pode impedir que esse **objetivo de segurança** aconteça, é "Usuário estar na zona de perigo com cruzamento fechado". As causas para esse perigo são, ou o **usuário** não perceber que está na zona de perigo, ou o fato do **usuário** atravessar as barreiras de segurança de forma consciente.

Para mitigar o **perigo** do **usuário** não perceber que está na zona de perigo, o **recurso de segurança** "alertas visuais na zona de perigo"deve ser disponibilizado pela **equipe de manutenção da ferrovia**, outro ator do sistema que surgiu dessa necessidade de dependência. Já o perigo de que o usuário ultrapasse a via conscientemente, pode ser mitigado com a disponibilização do recurso de segurança "Legislação de travessia de cruzamentos"ou pela tarefa de segurança "Ter tempo máximo de bloqueio da definido". Essa última tarefa será definida como de responsabilidade do ator **Sistema Controlador do Cruzamento**.

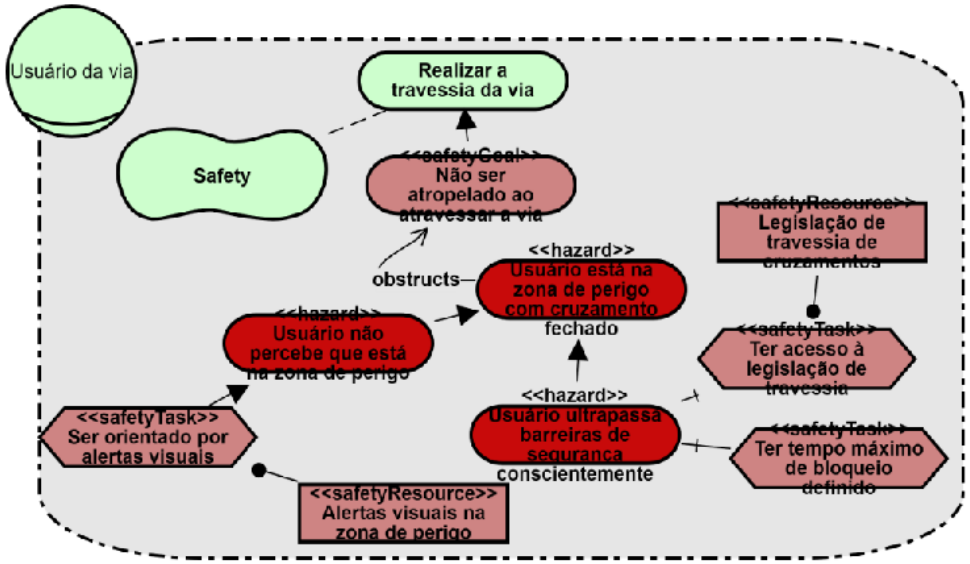


Figura 51 – Ator Usuário da Via - Parte 1 do modelo SR do Sistema de Controle de Travessia da Ferrovia modelado com iStar4Safer.

Fonte: (Autora, 2018).

5.3.7.2 Ator Sistema Controlador do Cruzamento

O ator S**istema Controlador do Cruzamento**, representado pela figura 52, possui dois **objetivos de segurança**.

O **objetivo de segurança** "Impedir passagem simultânea de usuário da via e trem", pode ser obstruído pelos **perigos** de "Aproximação de trem não sinalizada pelo sistema"e

pela "zona de perigo aberta para passagem de usuário e trem". As causas para a não-sinalização de aproximação do trem, serão a luz amarela ou vermelha de sinalização não estarem ativas ou as barreiras não estarem abaixadas. Para mitigar o **perigo** da luz amarela não estar ativa, deve-se realizar a tarefa de segurança "ativar a luz vermelha"e que o ator centro de operações repare a falha. Para mitigar o perigo da luz vermelha não estar ativa, deve-se "Cancelar o fechamento da via", além de ter a falha reparada pelo Centro de Operações. Quanto ao perigo das barreiras não serem abaixadas, á fim de mitigá-lo, o sistema deve declarar a falha e realizar a **tarefa de segurança** "Solicitar restauração do Sistema de Controle de Cruzamento"para o ator Centro de Operações. O perigo das barreiras abertas pode causar também o **perigo** da "Zona de perigo aberta para passagem de usuário e trem".

Já o **objetivo de segurança** de "Não bloquear indefinidamente passagens na via", pode ser violado ao sistema não ser informado sobre o término da passagem do trem. O que pode causar esse perigo, é o fato do sensor do trem falhar. Nesse caso, para sanar esse perigo, o Sistema de Controle de Cruzamento deve bloquear permissões de passagem e o Centro de Operações deve tratar o bloqueio.

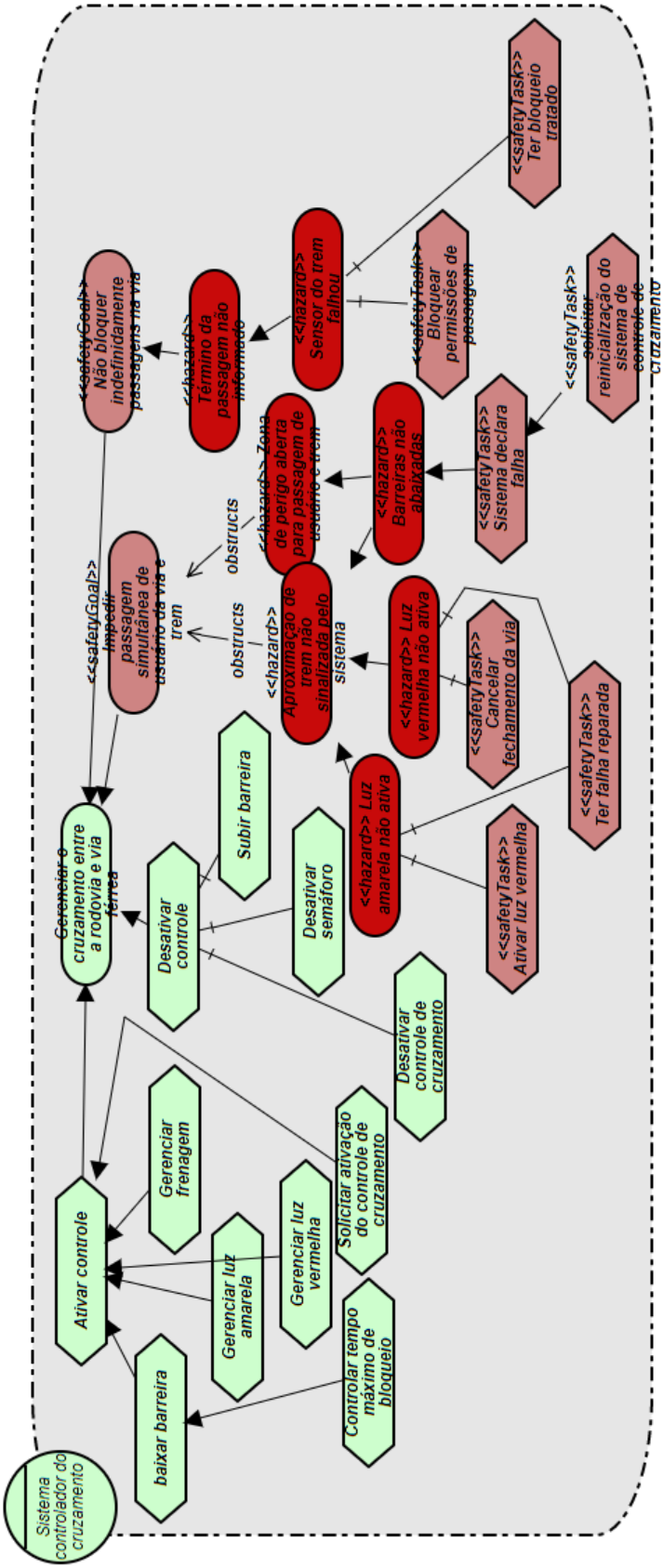


Figura 52 – Ator Sistema Controlador do Cruzamento - Parte 2 do modelo SR do Sistema de Controle de Travessia da Ferrovia modelado com iStar4Safety.

Fonte: (Autora, 2018).

5.3.7.3 Ator Equipe de Manutenção da Ferrovia

A **Equipe de Manutenção da Ferrovia**, representada pelo trecho do ator do modelo SR na figura 53, irá ser responsável pela tarefa de "Colocar alertas visuais na zona de perigo". Note que essa tarefa, apesar de ser crítica para o ator Usuário da Via, não tem essa natureza para a equipe de manutenção.

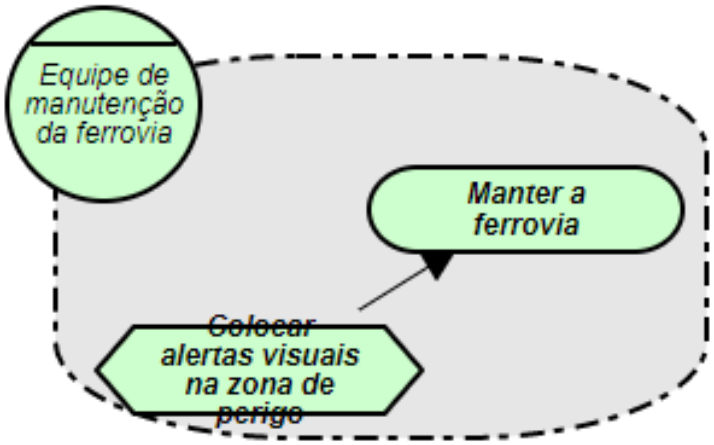


Figura 53 – Ator Equipe de Manutenção da Ferrovia - Parte 3 do modelo SR do Sistema de Controle de Travessia da Ferrovia modelado com iStar4Safety.

Fonte: (Autora, 2018).

5.3.7.4 Ator Operador

A figura 54 representa o trecho do modelo SR referente ao **Operador do trem**. A tarefa de "Ativar passagem do cruzamento em tela", apesar de não ter sido modelada como crítica para o ator em questão, é de execução necessária para sanar o perigo (relacionado ao ator Sistema Controlador do Trem) de "Falhas de comunicação entre sistemas". Note que o recurso "Interface de controle do trem"se tornou um **recurso de segurança** no novo modelo, devido ao mesmo ser um **recurso de segurança** no ator **sistema controlador do trem**.

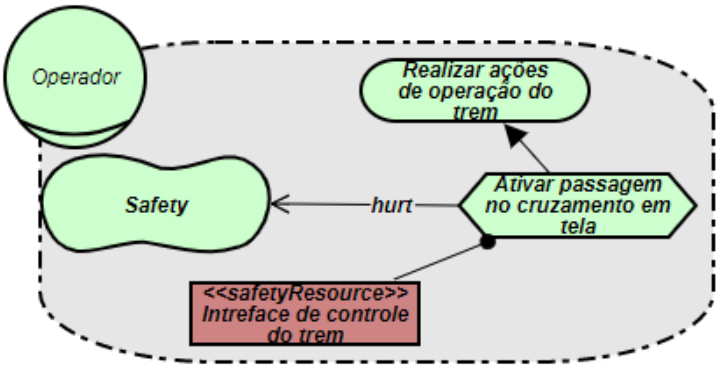


Figura 54 – Ator Operador - Parte 4 do modelo SR do Sistema de Controle de Travessia da Ferrovia modelado com iStar4Safety.

Fonte: (Autora, 2018).

5.3.7.5 Ator Sistema Controlador do Trem

O ator Sistema Controlador do Trem, vide figura 55, tem o **objetivo de segurança** de "Permitir passagem segura do trem". A passagem não será segura, caso o **perigo** "Sistema não recebe status de caminho seguro"aconteça. Esse **perigo** é causado por "falhas de comunicação entre sistemas". Para sanar esse perigo, espera-se que o ator Operador realize a **tarefa de segurança** de acionar em tela a passagem do trem.

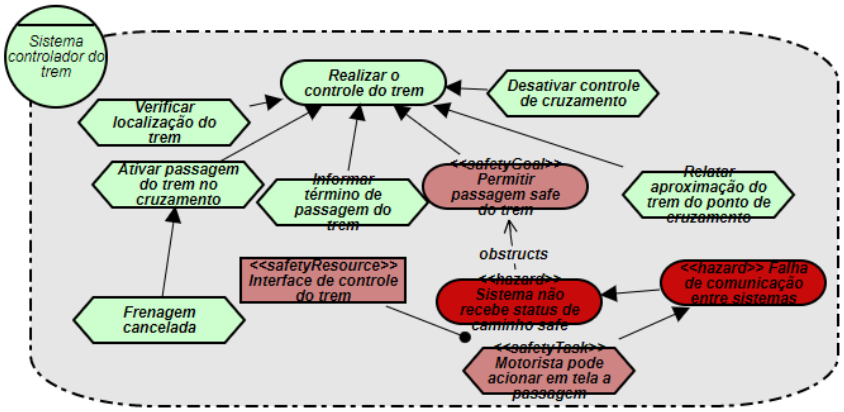


Figura 55 – Ator Sistema Controlador do Trem - Parte 5 do modelo SR do Sistema de Controle de Travessia da Ferrovia modelado com iStar4Safety.

Fonte: (Autora, 2018).

5.3.7.6 Ator Centro de Operações

Por fim, o ator Centro de operações, representado pela parte 6 (seis) do modelo SR na figura 54, é um novo ator, que não existia no modelo SR sem requisitos de segurança modelados. Ele é responsável por reinicializar o reparar falhas do semáfaro, além de reinicializar o sistema de controle de cruzamento ou ainda tratar o bloqueio da passagem, que são ambas **tarefas de segurança** para o ator Sistema Controlador do Cruzamento.

Note que esse ator só se relaciona ao sistema por ser responsável por realizar estratégias de segurança para o ator **Sistema Controlador de Cruzamento**.

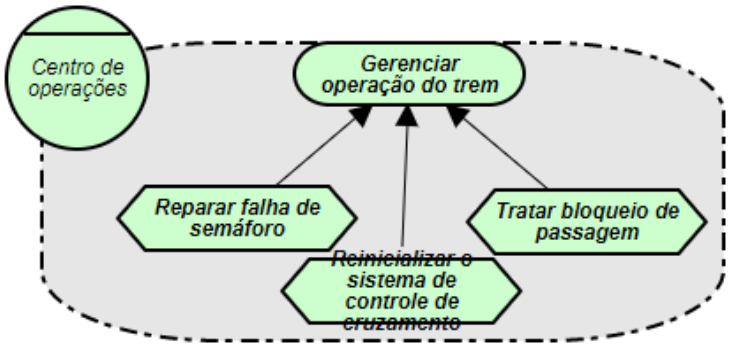


Figura 56 – Ator Centro de Operações - Parte 6 do modelo SR do Sistema de Controle de Travessia da Ferrovia modelado com iStar4Safety.

Fonte: (Autora, 2018).

5.3.8 Conferindo a Completude do Modelo Criado

Para verificar se os requisitos de segurança foram inseridos de forma satisfatória atendendo às regras de completude, a verificação de completude preconizada pelo guia de modelagem (veja seção 4.3.6.1), foi realizada conforme descrito na Tabela 7:

Tabela 7 – Conferindo completude do modelo criado pra Sistema de Controle de Travessia de Ferrovia com iStar4Safety.

Diretriz de Completude	Nome da Diretriz	Status
1	Todos os objetivos de segurança tem a propriedade Nível de Impacto do Acidente com um dos valores predefinidos para esse campo inserido	Realizado na seção 5.3.1
2	Todos os objetivos de segurança-raiz tem um ou mais perigos associados. Caso não sejam raiz, os objetivos de segurança são refinados por um ou mais objetivos de segurança	Realizado na seção 5.3.2
3	Todos os perigos-raiz estão ligados à um ou mais objetivos de segurança-folha através do link obstrui	Realizado na seção 5.3.3
4	Todos os perigos-folha estão ligados à uma ou mais estratégias de segurança	Realizado na seção 5.3.4
5	Todas as estratégias de segurança estão associadas ao ator que as realizam, menos no caso de serem realizadas pelo próprio ator em que estão definidas	Realizado na seção 5.3.5
6	Um recurso que é de segurança será em todo modelo desse tipo, exceto fora dos limites dos atores (dependum), em que não podem ser modelados construtores iStar4Safety	Realizado na seção 5.3.6

5.4 CONCLUSÃO

Este capítulo apresentou a modelagem de um Sistema de Controle de Cruzamento de Ferrovia, para validar a extensão criada com um Sistema Crítico de Segurança não-trivial e complexo, conforme solicitado pelo processo PRISE.

Primeiramente, na seção 5.1, os requisitos gerais do Sistema de Cruzamento foram elencados. Depois disso, os requisitos de segurança necessários para garantir que o sistema execute de forma segura foram apresentados. Na seção 5.3, o Sistema de Cruzamento da Ferrovia foi modelado, seguindo as diretrizes propostas para melhor utilização de iStar4Safety. Por fim, os modelos de Dependência Estratégica (SD) e de Raciocínio Estratégico (SR) são apresentados como resultado final dessa ilustração da modelagem com iStar4Safety.

A avaliação de iStar4Safety, realizada através da análise da modelagem de um Sistema Crítico de Segurança com iStar4Safety e pela aplicação de um survey presencial é detalhada, além da análise dos resultados obtidos.

6 AVALIAÇÃO DA LINGUAGEM ISTAR4SAFETY

Segundo Wohlin et al. (2012), a realização de um estudo empírico é essencial para a avaliação de processos e atividades realizadas por humanos. O método survey, trata-se de um método empírico de pesquisa para coletar informações para descrever, comparar ou explicar conhecimentos, atitudes e comportamento (KITCHENHAM; PFLEEGER, 2008).

Os principais pontos a serem traçados quando utilizando a avaliação empírica através desse método são (KITCHENHAM; PFLEEGER, 2008):

- Configurar os objetivos;
- Definir o projeto do survey;
- Desenvolver os instrumentos do survey;
- Avaliar o instrumento do survey;
- Obter dados válidos;
- Analisar os dados obtidos.

6.1 OBJETIVOS DO ESTUDO

O intuito desse estudo foi: **Analisar** a modelagem de Sistemas Críticos de Segurança em conjunto com a extensão iStar4Safety, **para o propósito de** avaliação, **com respeito à** usabilidade, efetividade, eficiência e avaliação da modelagem **do ponto de vista de** engenheiros de software, **no contexto de** estudantes de graduação da disciplina de Especificação de Requisitos e Validação de Sistemas, do Centro de Informática da UFPE[1].

Para atender ao objetivo do estudo foi solicitado aos participantes que desenvolvessem a modelagem de um Sistema Crítico de Segurança, uma Bomba de Infusão de Insulina, conforme documento de requisitos fornecido aos mesmos e disponível no anexo B.1. Os modelos criados pelos participantes foram avaliados, para verificar os quesitos de corretude sintática, complexidade estrutural e similaridade comportamental da modelagem do Sistema Crítico de Segurança proposto.

Para avaliar a opinião dos participantes, foi implementado um questionário com questões abertas e fechadas que deveriam ser respondidas pelos mesmos, após a modelagem do Sistema Crítico de Segurança. Para a análise do questionário, quatro questões de pesquisa foram formuladas. O foco era avaliar se a nova extensão atende ao quesito usabilidade (o quanto a extensão é fácil de usar e aprender), efetividade (se a extensão realiza o que ela se propõe a fazer: modelar requisitos de segurança de uma Análise Preliminar de Segurança)

[1] Universidade Federal de Recife

e eficiência (se a utilização da extensão possibilita o menor desperdício de esforço). As seguintes questões de pesquisa foram então propostas:

RQ1: *Eficiência* - Quanto esforço é demandado para a modelagem através do uso da extensão?

RQ2.1: *Usabilidade* - Os sujeitos possuem dificuldade em entender ou aplicar a extensão iStar4Safety?

RQ2.2: *Usabilidade* - Os sujeitos se mostraram satisfeitos ao utilizar a extensão proposta?

RQ3: *Efetividade* - A extensão iStar4Safety auxilia na modelagem de requisitos de segurança relacionados à modelagem de uma Análise Preliminar de Segurança levando em conta o processo proposto pela mesma?

As questões que fazem parte do formulário B.3 foram desenvolvidas tomando por base as questões de pesquisa acima definidas (KITCHENHAM; PFLEEGER, 2002b).

6.2 PROJETO DO ESTUDO

O método utilizado para avaliação da extensão iStar4Safety foi do tipo transversal (*cross-sectional*), aplicado de forma supervisionada, já que foi realizado em ambiente controlado, e um mesmo horário, com todas as atividades e participantes tendo recebido o mesmo tratamento.

Para responder as questões de pesquisa, o estudo foi dividido em três fases (LUCASSEN et al., 2016): (1) Os participantes receberam aulas sobre os temas relevantes à aplicação do *survey* (segurança, linguagem iStar, extensão iStar4Safety, Sistemas Críticos de Segurança e sobre o sistema de Bomba de Insulina a ser modelado), considerando que já tinham conhecimento prévio sobre iStar e conhecimento básico em segurança; (2) Dezesseis (16) alunos graduandos da disciplina de Especificação de Requisitos e validação de sistemas da UFPE realizaram a modelagem de um Sistema Crítico de Segurança (Bomba de Infusão de Insulina) utilizando a extensão iStar4Safety na ferramenta piStar-4Safety. Os participantes estavam no mesmo ambiente e em mesmo horário; (3) Após a modelagem, os participantes responderam presencialmente um questionário que coletou opiniões relevantes sobre os mesmos e opiniões sobre o uso da extensão.

É importante salientar que Svahnberg, Aurum e Wohlin (2008) trouxeram luz ao fato de que as percepções sobre o entendimento e avaliação de múltiplas perspectivas envolvidas na seleção de processos, são comparáveis entre estudantes e profissionais experientes da indústria.

6.3 DESENVOLVIMENTO DOS INSTRUMENTOS DO ESTUDO

Os instrumentos do estudo foram desenvolvidos através dos seguintes passos: (KITCHENHAM; PFLEEGER, 2002a):

- Busca na literatura relevante;
- Construção dos instrumentos;
- Avaliação dos instrumentos; e
- Documentação dos instrumentos.

Portanto, para satisfazer ao objetivo do estudo, os instrumentos desenvolvidos e utilizados por esse estudo encontram-se listados abaixo:

1. Um **termo de consentimento**, visando apresentação das regras de participação e garantir a confidencialidade de participação para os participantes envolvidos na avaliação;
2. Um **guia** impresso e online com as diretrizes para a modelagem utilizando a extensão iStar4Safety, orientando a modelagem em conjunto com a ferramenta piStar-4Safety (anexo B.2);
3. Uma **descrição de requisitos do Sistema Crítico de Segurança**, a Bomba de Infusão de Insulina, englobando requisitos gerais e requisitos de segurança relacionados à Análise Preliminar de Segurança do sistema (Anexo B.1);
4. Um **pré-modelo do modelo SR da Bomba de Infusão de Insulina**, em formato ".txt"(formato suportado para *upload* de modelos existentes, na ferramenta PiStar-4Safety), contendo a modelagem de requisitos gerais já inseridos, visto que o objetivo do exercício foi avaliar a modelagem da extensão para segurança. Logo, devido à complexidade do sistema e do tempo disponível para modelagem, foi solicitada apenas a inclusão dos requisitos de segurança do sistema proposto (Anexo B.5);
5. Uma **descrição geral da Bomba de Infusão de Insulina**, disponibilizada *online* em forma de slides. (Adaptado da aula, anexo B.4);
6. Um **resumo da aula** aplicada englobando conceitos de segurança, iStar4Safety e da ferramenta piStar-4Safety, disponibilizada *online* em forma de slides. (Adaptado da aula, anexo B.4); e
7. Um **questionário *online*** criado e disponibilizado no aplicativo *google forms* contendo (1) questões de perfil pessoal e acadêmico sobre os participantes e (2) questões abertas e fechadas para avaliação do uso da extensão e construtores da mesma (Anexo B.3).

É importante salientar que os requisitos do sistema à ser modelado, ou seja, a Bomba de Infusão de Insulina, é uma junção de requisitos encontrados em alguns trabalhos (ZHANG; JONES; JETLEY, 2010; ZHANG et al., 2011; MARTINS et al., 2015; VILELA et al., 2017b;

SOMMERVILLE, 2010) que foram refinados para representar um subconjunto de requisitos que fosse passível de modelagem, considerando:

- O nível acadêmico dos estudantes;
- o tempo disponível para aplicação do estudo; e
- o conhecimento em segurança, iStar, iStar4Safety e no próprio Sistema de Bomba de Infusão de Insulina.

Um outro ponto importante a ser esclarecido é que, no momento de aplicação do experimento, uma versão anterior da extensão estava em uso e foi utilizada pelos participantes. A diferença dessa versão do experimento para a atual é que os recursos de segurança podiam estar associados diretamente aos perigos através do link *neededBy*, além de, como é feito atualmente, á tarefas e tarefas de segurança.

6.4 AVALIAÇÃO DOS INSTRUMENTOS DA AVALIAÇÃO

As duas principais formas de avaliar os instrumentos de um survey, que integra essa avaliação, são através de grupos focais e estudos pilotos (KITCHENHAM; PFLEEGER, 2002a).

Portanto, para melhor atender ao objetivo da avaliação, os instrumentos do survey foram avaliados primeiramente por especialistas em cada uma das áreas relacionadas à extensão: na área de segurança, foi obtido o apoio de uma pesquisadora doutora que estuda requisitos críticos de segurança e é uma das autoras do trabalho-base (VILELA et al., 2017b) utilizado para escolha de conceitos à serem inseridos na extensão. Além disso, em relação à extensões de iStar, o criador do processo PRISE, doutorando da UFPE auxiliou na avaliação de tais instrumentos. O orientador desse trabalho e os demais pesquisadores citados são especialistas na linguagem iStar e na área de Engenharia de Requisitos. Após a avaliação, os ajustes relevantes foram aplicados aos instrumentos. Posteriormente, foi realizado um piloto do estudo que é descrito na próxima subseção.

6.4.1 Piloto da Avaliação

Para avaliarmos a aplicabilidade do método desenhado, foi realizado um piloto em uma turma de pós-graduação onde os participantes foram 07 (sete) alunos de mestrado e doutorado da UFPE, inscritos na disciplina de Engenharia de Requisitos do mesmo semestre.

O objetivo da avaliação-piloto foi avaliar, sobre a visão de pós-graduandos, os instrumentos, material de apoio e questionário que seriam aplicados. Com os resultados, aplicou-se melhorias ao processo de avaliação e adaptações nos instrumentos de apoio e questionário.

Após a aplicação do piloto, definiu-se que o tempo de resposta de 25 (vinte e cinco) minutos para o survey e de 1 (uma) hora para a modelagem, seriam suficientes para a realização das tarefas.

6.5 DOCUMENTAÇÃO DA AVALIAÇÃO

Antes de aplicarmos o estudo, os participantes receberam aulas sobre os temas de interesse desse estudo, a saber:

- iStar: 8 (oito) horas de aula sobre a linguagem iStar 2.0, que foi a base da extensão proposta. Estão incluídas também as aulas sobre a ferramenta piStar;
- Segurança: 4 (quatro) horas de aula, sendo 2 (duas) horas para conceitos de segurança gerais e as demais 2 (duas) horas para apresentação do artigo sobre os conceitos de segurança necessários à modelagem da Análise Preliminar de Segurança;
- iStar4Safety: 3 (três) horas para apresentação da extensão iStar4Safety. Somado à extensão, foi apresentada também a ferramenta piStar-4Safety;
- Bomba de Infusão de Insulina: 1 (uma) hora de aula sobre o sistema exemplo, sem incluir detalhes de modelagem ou de requisitos à fim de não interferir no estudo; e
- Aula prática: 2 (duas) horas de prática da modelagem da extensão, utilizando a ferramenta piStar-4Safety, para a modelagem de um exemplo de Sistema Crítico de Segurança, um Sistema de Radioterapia.

No total, foram ministradas 18 horas/aula de conteúdo relacionado à extensão. É importante salientar que as oito horas de aula relacionadas à iStar e as quatro primeiras horas/aula de segurança foram ministradas durante a disciplina cursada pelos alunos, mas anteriormente ao período de aplicação do survey. Isso se deve ao fato de que os alunos estavam inscritos na disciplina e receberam aulas de iStar á fim de desenvolver os projetos relacionados à disciplina.

Foi apresentado aos participantes do estudo, para estimular sua participação, as seguintes informações (KITCHENHAM; PFLEEGER, 2002a)

- O propósito do estudo;
- Em que o estudo é relevante aos participantes;
- Porque a participação de cada participante é importante;
- O porque da escolha dos participantes; e
- Como a confidencialidade seria preservada.

Para evitar viés nas respostas, procuramos (KITCHENHAM; PFLEEGER, 2002a):

- Criar questões neutras, não utilizando perguntas que influenciassem as respostas dos entrevistados;
- Definir as perguntas de forma a cobrir adequadamente o tópico de pesquisa;
- A ordem das questões foi devidamente analisada, a fim de não influenciar respostas nas próximas questões;
- Buscou-se criar categorias de respostas do tipo não enviesadas, exaustivas e mutuamente exclusivas; e
- Buscou-se fornecer instruções claras e não enviesadas.

Para a execução do estudo foi solicitado aos participantes que utilizassem os itens disponibilizados tanto eletronicamente como impressos, utilizados para modelar os requisitos de segurança do sistema proposto.

O questionário online aplicado tinha por objetivo avaliar a percepção dos graduandos sobre o uso da extensão iStar4Safety.

As perguntas do questionário foram de dois tipos: abertas e fechadas. Questões abertas são mais complexas de analisar, porém evitam a restrição de respostas ao entrevistado, como acontece com as questões fechadas. Considerando o objetivo de avaliar a extensão, um total de 9 (nove) questões abertas e 15 (quinze) questões fechadas foram disponibilizadas no questionário, totalizando 24 (vinte e quatro) questões, além do pré-questionário com 10 (dez) questões para identificar o perfil e conhecimento dos participantes em relação aos temas da extensão. A aplicação utilizada, *Google Forms*[2], fornece a interface visual adequada (KITCHENHAM; PFLEEGER, 2002b). Procurou-se também atender ao checklist proposto por Kitchenham e Pfleeger (2002b) para informações e instruções sobre as melhores práticas no desenvolvimento do questionário. O questionário aplicado está disponível no anexo B.3.

O questionário começou com uma seção com 4 (quatro) questões para análise de perfil dos participantes e mais 6 (seis) questões sobre o conhecimento e uso de linguagens de modelagem, requisitos, linguagem iStar 2.0 e segurança.

A próxima seção do questionário teve por meta coletar informações sobre o uso da extensão: sua facilidade de uso e entendimento. No total, 24 (vinte e quatro) questões foram disponibilizadas para esse fim.

Questões do tipo escala *likert* foram utilizadas. Seis questões eram do tipo (concordo/-discordo), com as seguintes definições: (1) Discordo totalmente, (2) Discordo parcialmente, (3) Indiferente, (4) Concordo parcialmente e (5) Concordo totalmente. Outra definição para escala *likert* foi necessária em 3 (três) questões para definir o nível de contribuição de

[2] A plataforma Google forms encontra-se disponível em: <https://www.google.com/forms/about/>

cada construtor criado. Essa escala compreendia do valor 0 (zero) representando nenhuma contribuição, ao valor 5 (cinco) representando total contribuição.

6.5.1 Aplicação da Avaliação

Para a avaliação da extensão criada, os participantes foram convidados, após assinarem o termo de consentimento e confidencialidade, à modelarem um Sistema Crítico de Segurança, que no caso foi uma Bomba de Infusão de Insulina. Para isso, cada participante recebeu uma descrição dos requisitos do sistema. De posse da descrição e de um pré-modelo contendo as funcionalidades gerais do sistema modelado em iStar 2.0, os participantes deveriam modelar os requisitos de segurança propostos na descrição. Para essa atividade foi disponibilizado o tempo de uma hora e cinco minutos.

Logo após a modelagem, os participantes deveriam responder um questionário *online* disponibilizado na plataforma *google forms*. O tempo estipulado para respostas foi de 25 (vinte e cinco) minutos.

Ao fim da aplicação do formulário, foi disponibilizado um campo para inserção dos artefatos produzidos pela atividade, a saber: (1) Modelo SR do Sistema de Bomba de Insulina com iStar4Safety em ".txt"e (2) Modelo SR do Sistema de Bomba de Insulina com iStar4Safety em ".png"que seriam posteriormente analisados.

Após a realização da atividade de modelagem e da submissão do questionário, a participação dos estudantes foi considerada concluída.

6.6 OBTENÇÃO DE DADOS VÁLIDOS

A amostra da população foi selecionada de forma a evitar viés, prover adequação e atender ao melhor custo-benefício. O intuito com o uso do subconjunto de indivíduos que compreende a amostra é generalizar a população-alvo (KITCHENHAM; PFLEEGER, 2002b; KITCHENHAM; PFLEEGER, 2008).

A população-alvo foram Engenheiros de Software com conhecimento em segurança, sendo estes os profissionais que realizam a modelagem de requisitos em ambientes reais.

O subconjunto representativo da população-alvo foram alunos de graduação dos cursos de Ciência da Computação e Engenharia da Computação, que haviam sido previamente treinados em: (1) segurança, (2) iStar, (3) na ferramenta de modelagem piStar, (4) nas funcionalidades básicas do Sistema Crítico de Segurança aplicado para modelagem e (4) na extensão iStar4Safety e sua ferramenta de suporte, piStar-4Safety.

6.7 ANÁLISE E INTERPRETAÇÃO DOS DADOS

Kitchenham e Pfleeger (2003) trazem luz ao fato de que são necessários cuidados ao analisar questões abertas, devido ao fato da subjetividade inerente às respostas das mesmas. Categorias de respostas devem, portanto, ser construídas.

É importante salientar que a possibilidade de definir a obrigatoriedade de resposta das questões ao desenvolver o formulário no aplicativo *google forms* possibilitou que não houvesse questões sem resposta. Apresenta-se, a seguir, a análise e interpretação dos dados obtidos.

6.7.1 Perfil Acadêmico e Nível de Conhecimento

O total de 16 (dezesseis) estudantes tiveram seus dados analisados, apesar do estudo ter se iniciado com 18 (dezoito) sujeitos, sendo dois deles removidos do estudo por não terem comparecido à todos os treinamentos requeridos.

A primeira seção do formulário compreendia dez questões que tinham o intuito de possibilitar tanto a análise do perfil quanto do conhecimento dos entrevistados nas áreas fins do estudo: segurança, iStar e Engenharia de Requisitos. A tabela 8 resume os dados de perfil acadêmico dos participantes.

Tabela 8 – Perfil acadêmico dos entrevistados.

Participante	Período do curso	Curso
#1	10	Engenharia da computação
#2	9	Ciência da computação
#3	10	Ciência da computação
#4	13	Engenharia da computação
#5	12	Engenharia da computação
#6	7	Ciência da computação
#7	12	Engenharia da computação
#8	6	Ciência da computação
#9	8	Ciência da computação
#10	12	Engenharia da computação
#11	9	Engenharia da computação
#12	9	Engenharia da computação
#13	8	Ciência da computação
#14	12	Ciência da computação
#15	9	Ciência da computação
#16	9	Engenharia da computação

Para a maioria dos entrevistados, a primeira experiência com modelagem se deu atra-

vés da disciplina. Cinco (5) participantes relataram ter mais que seis meses de experiência com Engenharia de Requisitos. O tempo de experiência dos entrevistados é representado no gráfico da Figura 57. Além disso, dez (10) participantes declararam médio conhecimento em Engenharia de Requisito, enquanto seis (6) participantes declararam baixo conhecimento. Quanto ao conhecimento em iStar, os participantes declararam em sua maioria (treze participantes) ter conhecimento de nível médio. Somente um participante declarou ter baixo conhecimento e três disseram ter alto conhecimento em iStar. Quanto à segurança, sete (7) participantes declararam ter baixo conhecimento e nove (9) médio conhecimento, e somente um participante declarou ter alto conhecimento.

Considera-se, portanto, que a população da amostra utilizada por esse experimento é homogênea em relação ao perfil acadêmico e conhecimento nas áreas fins do estudo.

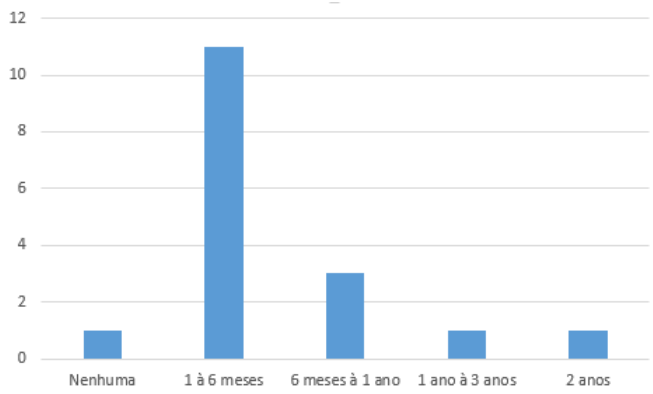


Figura 57 – Tempo de experiência dos participantes em Engenharia de Requisitos.

Fonte: (Autora, 2018).

6.7.2 Avaliação da Eficiência

Para avaliar a **eficiência** na modelagem com a extensão, definiu-se o esforço necessário para modelagem através do uso de iStar4Safety, respondendo assim à questão de pesquisa **RQ1**. Para tanto foi calculado o tempo médio gasto pelos participantes. É importante salientar que a diferença entre o menor e o maior tempo de modelagem do Sistema Crítico de Segurança proposto foi de 58 (cinquenta e oito) minutos. O tempo médio de modelagem foi de 41 (quarenta e um) minutos. O tempo gasto por cada participante pode ser visto na tabela 9.

Tabela 9 – Tempo gasto para modelagem de requisitos de segurança com iStar4Safety.

Participante	Tempo (hh:min)	Participante	Tempo (hh:min)
#1	00:29	#9	00:34
#2	00:20	#10	00:41
#3	00:31	#11	00:49
#4	00:40	#12	00:45
#5	00:31	#13	00:55
#6	00:40	#14	01:00
#7	00:31	#15	00:47
#8	00:30	#16	01:18

Os participantes afirmaram, no questionário, em sua maioria, que acreditavam que o tempo gasto para modelagem de requisitos de segurança sem o uso da extensão, seria maior.

6.7.3 Avaliação da Usabilidade

Para avaliar a **usabilidade** da extensão iStar4Safety, foram propostas as questões de pesquisa **RQ2.1** e **RQ2.2**.

Com o intuito de responder à questão de pesquisa 2.1, ou seja, verificar se os participantes tinham dificuldades em entender ou aplicar a extensão, algumas questões foram propostas para esse fim e são abaixo analisadas.

O nível de dificuldade declarado pelos participantes quanto ao entendimento de iStar4Safety foi de fácil à muito fácil. Mais da metade dos participantes declararam que seria mais difícil modelar requisitos de segurança sem a extensão proposta, conforme pode ser visto na Figura 58. Dois participantes (#10 e #14) declararam que seria mais fácil modelar somente com iStar padrão. No entanto, o participante #10 declarou em outro momento, que não visualizava uma forma de modelar os requisitos de segurança com iStar padrão. Já o participante #14 comentou que acredita ser totalmente possível, de uma forma mais complexa, porém sem detalhar como seria a modelagem.

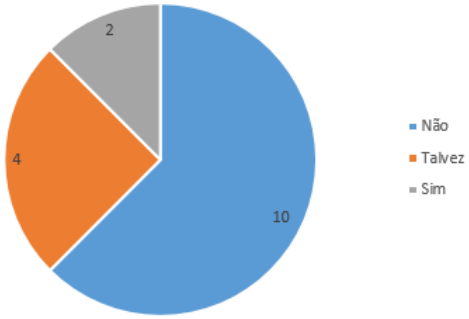


Figura 58 – A modelagem de requisitos de segurança seria mais fácil sem a extensão.

Fonte: (Autora, 2018).

Para auxiliar no uso da extensão, foi disponibilizado o guia de modelagem que pode ser visualizado no anexo B.2. O guia pretende ser um tutorial para a modelagem com a extensão, e ainda, apresenta um checklist para verificar a completude da modelagem criada. Quando indagados sobre o guia, os participantes julgaram as diretrizes definidas para modelagem como satisfatórias. Somente o participante #13 se declarou indiferente. Porém, quando indagado sobre sua percepção e sugestão de alterações no guia, declarou: "está tranquilo". Portanto, não podemos concluir que sua resposta realmente define insatisfação com o artefato.

Quanto à dificuldade no uso dos construtores, o mais citado por sua dificuldade de uso foi o construtor **recurso de segurança**, sendo também o menos usado nas modelagens desenvolvidas na proposta do Sistema Crítico de Segurança. A maioria dos participantes justificou a dificuldade por não saber como conectá-lo à outros construtores ou por limitação de conhecimento pessoal. Portanto, considera-se ser uma questão relacionada à subjetividade de modelagem e não a um erro e também à dificuldades oriundas já do uso da linguagem nativa iStar. Um participante citou a dificuldade em saber quando parar de modelar **perigos**. Apesar disso, os participantes foram orientados à modelar a quantidade de camadas de perigo necessárias para criar o documento de requisitos proposto. Acredita-se que mais treinamentos sobre a extensão mitigariam as dificuldades apresentadas.

Quanto a questão de pesquisa 2.2, que mede especificamente o quanto os sujeitos se mostraram satisfeitos ao utilizar a extensão, todos os participantes declararam que utilizariam a extensão para modelar requisitos iniciais do seu Sistema Crítico de Segurança. Quanto à opinião dos participantes, os pontos mais recorrentes foram condensados nos seguintes tópicos:

1. A extensão é importante para o fim à que se destina (16);
2. A extensão é fácil de usar (5); e
3. Aprender sobre a extensão é mais fácil para quem já conhece iStar (3).

A facilidade de utilização pode atender à um dos objetivos ao criar-se a extensão: simplicidade. Isso se deve ao fato de que, se os participantes conseguiram no tempo do estudo modelar os requisitos propostos para um sistema crítico em uma nova linguagem, e considerando que todas as funcionalidades foram modeladas, ao menos de forma incompleta, acredita-se, do ponto de vista do autor, que tal extensão seja simples.

É importante salientar que os participantes em sua maioria (14 participantes) declararam que na sua opinião era realmente necessária uma proposta de extensão da linguagem iStar para modelagem de Sistemas Críticos de Segurança na fase inicial de requisitos.

6.7.4 Avaliação da Efetividade

Para analisarmos a efetividade da extensão proposta, analisamos se os participantes julgavam que a extensão os auxiliou na modelagem de requisitos de segurança de uma Análise Preliminar de Segurança (PSA) (**RQ3**).

Os participantes disseram que o uso da extensão contribuiu para a modelagem de perigos, suas causas e respectivas estratégias de solução, conforme pode ser visualizado na Figura 59. Os participantes também concordaram com a afirmação de que os construtores criados foram suficientes para modelagem dos requisitos chave de um documento de Análise Preliminar de Perigos de um Sistema Crítico de Segurança.

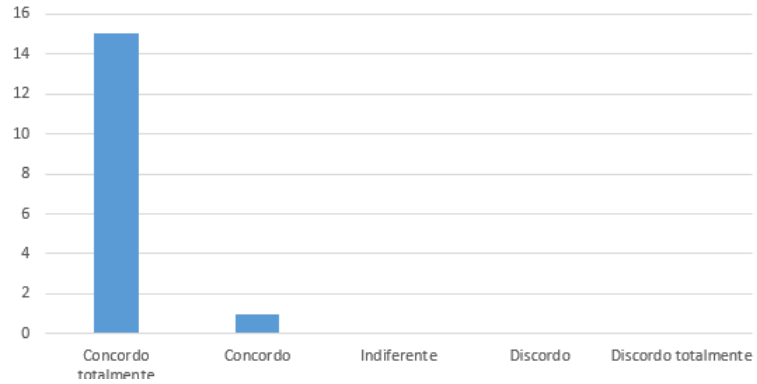


Figura 59 – Contribuição da extensão para modelagem de aspectos relacionados à segurança.

Fonte: (Autora, 2018).

Os participantes foram indagados sobre a contribuição de cada construtor apresentado para modelagem dos conceitos de segurança modelados. O gráfico 60 mostra a avaliação de cada participante. A escala de avaliação foi de 0 (zero), significando *nenhuma contribuição* à 5 (cinco), que significava *total contribuição*. Os construtores com menor média de contribuição (4,5) foram os de estratégias de segurança. Foi solicitada a opinião dos participantes sobre os construtores em uma questão aberta. O participante #11 sugeriu que fosse criado um outro tipo de construtor para a representação do elemento **perigo**, sem proposta de nenhum, mas salientou que a diferença de cores já contribuiu para di-

ferenciação do construtor perigo para outros construtores. O participante #15 acredita serem necessários os refinamentos **E/OU** entre **objetivos de segurança** e **perigos**. Essa proposta não se aplica, porque qualquer **perigo** trata-se de um obstáculo suficiente para a não concretização do **objetivo de segurança**. As demais opiniões no geral refletem satisfação com os construtores criados.

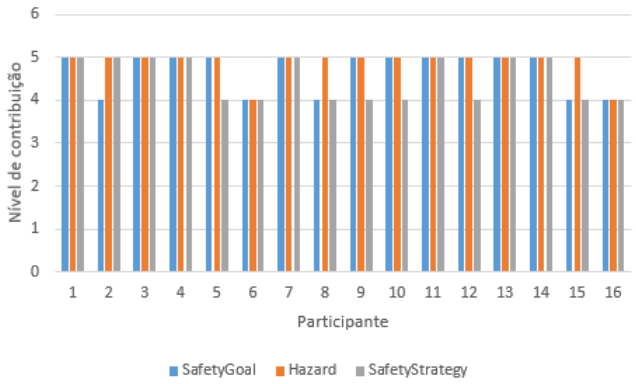


Figura 60 – Contribuição de cada construtor da extensão para modelagem de aspectos relacionados a segurança.

Fonte: (Autora, 2018).

Quando questionados sobre para qual nível profissional os participantes indicariam a extensão, as respostas foram, em sua maioria, tanto para Engenheiros de Software com experiência em segurança quanto para aqueles sem experiência em segurança, conforme apresentado pela Figura 61. Acredita-se que essa opinião reflita a ideia de que a extensão consegue ter construtores fáceis de atender, considerando que a falta de conhecimento em segurança não impediria a indicação de uso da extensão.

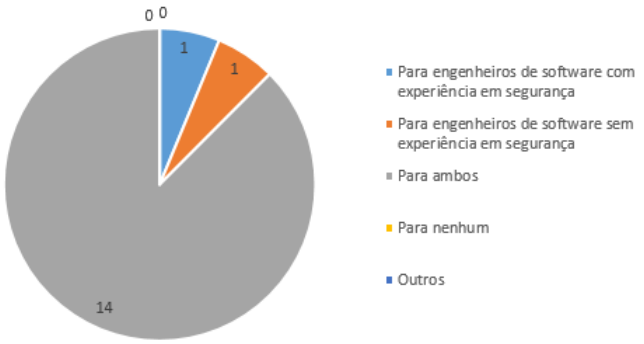


Figura 61 – Profissionais aos quais se indicaria o uso da extensão iStar4Safety.

Fonte: (Autora, 2018).

Solicitou-se que os participantes propusessem mudanças ou melhorias para a extensão. Algumas opiniões diziam respeito à ferramenta de apoio. Como o intuito dessa questão era avaliar a extensão, desconsideramos tais respostas. É importante dizer que a ferramenta de apoio é uma ameaça à validade do estudo devido ao fato de que a mesma não implementava

algumas regras de validação. Além disso, a ferramenta base, pi-Star vem passando por um processo de adaptações para inserção de melhorias. Apresenta-se abaixo, um resumo das propostas de melhoria apontadas pelos participantes:

1. Alteração da propriedade **Nível de Impacto do Acidente**; e
2. Cores mais significativas.

Apesar de não ser o foco desse estudo, foram solicitadas também opiniões sobre a ferramenta utilizada. É importante dizer que a ferramenta ainda não implementava as regras de validação de iStarSafety. Além disso a ferramenta base, piStar, está em processo de atualização e alguns erros apontados e relacionados à ferramenta base foram repassados aos seus desenvolvedores, para contribuir nesse processo. As propostas relacionadas a versão com a extensão, indicam principalmente a necessidade de mudança da propriedade **nível de impacto do acidente**. Isso se dá, pelo item ser uma lista finita de valores, o que não pode atualmente ser implementado pela ferramenta. Nesse caso, essa melhoria será postergada para a nova versão. As outras propostas dizem respeito às regras de validação não implementadas. Essas questões serão tratadas em uma nova versão futura da ferramenta.

6.7.5 Avaliação da Qualidade dos Modelos Criados

Para avaliar a modelagem do Sistema Crítico de Segurança realizada pelos participantes do estudo, foram considerados três aspectos: corretude sintática, complexidade estrutural e similaridade comportamental (VILELA; CASTRO; PIMENTEL, 2016; MIRANDA; GENERO; PIATTINI, 2005; DIJKMAN et al., 2011). A seleção dessas propriedades foi baseada no trabalho de Vilela, Castro e Pimentel (2016) que as propõe para análise de resultados de modelos criados pelo experimento proposto pelos autores. Apresenta-se cada um dos aspectos e sua análise nas próximas seções.

6.7.5.1 Corretude Sintática

Nessa avaliação, os modelos criados pelos participantes foram analisados, para verificação da corretude sintática dos mesmos. Para tanto, todos os modelos gerados pelos participantes através da ferramenta piStar-4Safety foram analisados. É importante salientar que a ferramenta não implementava as regras de validação da extensão. Portanto devido à tal característica, foi possível avaliar a quantidade de erros sintáticos. A tabela 10 mostra a quantidade de erros sintáticos por participante. Nota-se que o máximo de erros encontrados por participante foi quatro (participante #12). Além disso, dez participantes cometeram até um erro sintático, somente.

Tabela 10 – Quantidade de erros sintáticos por participante na modelagem de iStar4Safety.

Participante	Erros sintáticos	Participante	Erros sintáticos
#1	0	#9	3
#2	1	#10	1
#3	3	#11	1
#4	1	#12	4
#5	0	#13	1
#6	1	#14	1
#7	2	#15	0
#8	3	#16	2

Os erros sintáticos encontrados, bem como a porcentagem de ocorrência dos mesmos são apresentados na Figura 62.

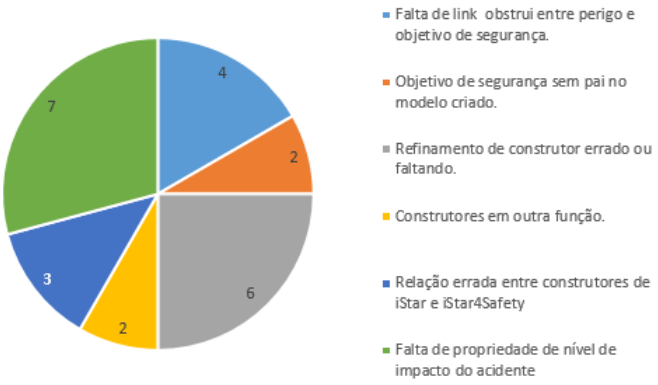


Figura 62 – Tipos de erros sintáticos presentes na modelagem.

Fonte: (Autora, 2018).

Observa-se que a quantidade de erros sintáticos foi baixa. O erro mais cometido, "Falta de propriedade **nível de impacto de acidente**", aconteceu geralmente quando o participante modelou um **objetivo de segurança** que não foi definido na descrição de requisitos proposta inicialmente, ou ainda quando o **objetivo de segurança** é um pai de outros **objetivos de segurança** e, portanto, não possuíam o nível de acidente definido no documento de requisitos. Acredita-se que um tratamento para esse erro é conferir a completude do documento de requisitos quanto à definições de segurança. Ainda, caso os modeladores tenham autonomia e conhecimento, podem inserir o valor da propriedade em tempo de modelagem. Quanto aos **objetivos de segurança**-pais, conforme já mencionado, a ação indicada é que eles possuam o nível de impacto mais alto entre seus filhos.

O segundo erro mais recorrente, "Refinamento de construtor errado ou faltando", engloba questões sobre erros relacionados ao refinamento de construtores da extensão: seta

de refinamento com direção errada (1), falta de refinamento de algum **objetivo de segurança** (2), **objetivo de segurança** com refinamento **OU** e link **obstrui** ao mesmo tempo (2), e, por fim, um **objetivo de segurança** sendo refinado em elementos não relacionados à segurança (1). Esses erros sintáticos podem ser tratados futuramente pela ferramenta, através da implementação das restrições definidas para a extensão, apresentadas na subseção 4.3.2.2.

Já o terceiro maior tipo de erro sintático cometido, "Falta de link **obstrui** entre **perigos** e **objetivos de segurança**", parece se justificar por ter faltado atenção pontual por parte dos participantes, visto que os sujeitos que cometeram tal erro modelaram corretamente o link para os outros três **objetivos de segurança**, errando na modelagem de somente um **objetivo de segurança**. Esse erro também poderia ser impedido pela implementação dessa restrição na ferramenta usada.

Quanto ao erro "**Objetivo de segurança** sem pai no modelo criado", tal situação aconteceu com dois participantes e em relação ao mesmo **objetivo de segurança**: "Não receber descarga elétrica". Esse erro provavelmente está associado à não compreensão desse ponto no documento de requisitos proposto.

O erro de "Relação errada entre construtores iStar e iStar4Safety", aconteceu três vezes por dois participantes e observa-se que houve uma má compreensão dessa restrição. Já o erro de "Construtores em outra função", ou seja, a inserção de construtores errados para indicar algo que é representado por outro construtor, cometido por um participante, mostra a não-familiaridade do mesmo com iStar. O participante declarou seu conhecimento na linguagem como nível médio, além de ter declarado não ter experiência com Engenharia de Requisitos.

6.7.5.2 Complexidade Estrutural

A complexidade estrutural visa verificar a quantidade de elementos que compõem o modelo criado pelo participante. Logo, foi calculado aqui a quantidade de elementos utilizados pelos participantes para modelar os requisitos de segurança com iStar4Safety.

A Tabela 11 representa a quantidade de construtores usados por cada participante para modelagem do Sistema de Bomba de Infusão de Insulina, à partir do documento de requisitos que os mesmos receberam. Vale salientar que, devido à subjetividade de modelagem de cada indivíduo, a diferença de quantidade de construtores é aceitável e normal. Porém, alguns casos devem ser discutidos.

Tabela 11 – Quantidade de construtores de iStar4Safety por participante.

Participante	Objetivos de Segurança	Perigos	Links Obstrui	Nível de Impacto do Acidente	Recurso de Segurança	Tarefa de Segurança
#1	5	12	6	5	0	6
#2	5	10	5	4	0	7
#3	6	12	7	4	0	5
#4	5	14	4	5	1	5
#5	4	12	7	4	3	3
#6	5	12	7	5	0	6
#7	6	12	6	5	1	6
#8	5	11	4	5	0	7
#9	5	12	5	5	1	5
#10	4	14	4	4	0	6
#11	6	13	7	6	2	4
#12	6	10	6	4	1	6
#13	5	11	7	4	1	5
#14	5	12	7	5	1	6
#15	5	12	7	5	0	7
#16	5	13	4	5	1	5
Média	5,12	12	5,81	4,68	0,75	5,56

Ao analisar a média dos construtores de iStar4Safety modeladas por participante, observa-se que a quantidade de construtores usados foi equivalente, com poucas variações, que ocorreram em sua maior parte provavelmente devido à subjetividade inerente à prática da modelagem. As maiores variações aconteceram na modelagem do link **obstrui** em que o menor número de links é quatro e o maior é sete. A quantidade de links **obstrui** deve ser a quantidade de **perigos**-raiz existentes, multiplicando cada um pela quantidade de **objetivos de segurança** que o mesmo obstrui. Quatro participantes não se atentaram à essa regra cometendo erros sintáticos. É importante salientar que todo **objetivo de segurança** também deveria possuir um valor definido para o **nível de impacto de acidente**, porém, observa-se a diferença da quantidade dos dois construtores entre o mesmo participante, o que se configura, conforme já apresentado, como um erro sintático.

Outro ponto a ser observado é a resistência à modelar **recursos de segurança**. Sete (7) participantes declararam ser o construtor mais difícil de modelar. As estratégias de segurança foram modeladas em no mínimo seis e no máximo sete estratégias, considerando que essas etratégias são uma soma da quantidade de **tarefas de segurança** e *recursos de segurança* modelados. Somente um participante, #3, modelou menos de seis estratégias de segurança.

6.7.5.3 Similaridade Comportamental

A similaridade comportamental, segundo Dongen, Dijkman e Mendling (2013) visa analisar o comportamento do indivíduo segundo suas especificidades de experiências e conhecimento. Em relação à validação nesse estudo, a similaridade comportamental foi avaliada através da análise da quantidade de funcionalidades completas modeladas por participante.

Por funcionalidades completas, entendem-se que todos os construtores relacionados à um requisito de segurança, ou seja, um **objetivo de segurança**, tenham sido modelados e a análise da situação perigosa e suas estratégias possam ser compreendidas, ou seja:

- O **objetivo de segurança** deve ter sido corretamente modelado;
- Os **perigos** associados ao **objetivo de segurança** devem ter sido corretamente modelados;
- As estratégias de segurança devem ter sido corretamente modeladas; e
- As ligações entre os construtores de iStarSafety devem ter sido corretamente modelados.

Todos esses pontos buscam garantir a modelagem de um sistema orientado à segurança. Sendo assim, foram verificadas se as funcionalidades, representadas pelos **objetivos de segurança** apresentados na tabela 12 foram corretamente modeladas, permitindo o correto entendimento dos mesmos. Os pontos distribuídos foram: (a) 0 (zero) ponto se a funcionalidade não foi modelada (b) 0,5 ponto se a funcionalidade foi modelada de forma incompleta e, (3) 1 (um) ponto se a funcionalidade foi modelada corretamente. O total de pontos máximo é quatro pontos e representa que todas as funcionalidades foram modeladas e estão corretamente representadas. A Figura 63 mostra a quantidade de funcionalidades modeladas por participante.

Tabela 12 – Funcionalidades à serem representadas no modelo.

Objetivo de Segurança
Não receber quantidade maior de insulina
Não receber quantidade menor de insulina
Não ser infectado
Não receber descargas elétricas

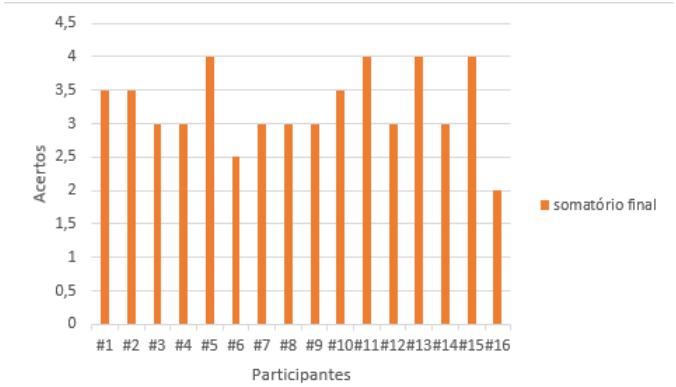


Figura 63 – Total de funcionalidades modeladas pelos participantes.

Fonte: (Autora, 2018).

Verificou-se que todas as funcionalidades foram representadas, ao menos parcialmente, por todos os participantes. Somente quatro participantes modelaram completamente e corretamente as funcionalidades do sistema proposto. O motivo da incompletude de modelagem por parte dos demais participantes foi, principalmente, devido a erros sintáticos que fazem com que a leitura da funcionalidade não seja correta.

6.8 AVALIAÇÃO DA VALIDADE DO ESTUDO

A validade de um estudo irá mostrar o quão confiáveis os seus resultados podem ser considerados, definindo em qual medida os mesmos podem ter sido ou não enviesados pela visão do pesquisador. Sendo assim, tendo em vista a confiabilidade do estudo, é importante definir tais ameaças. Quatro tipos de ameaças à validade foram consideradas para esse survey, a saber (WOHLIN et al., 2012):

- **Validade interna**: As ameaças à validade interna podem influenciar a variável independente afetando-as com respeito à causalidade, ameaçando a conclusão da relação entre o tratamento e os resultados. Como o *survey* não teve a presença de um grupo de controle, algumas ameaças podem influenciar seu resultado por consequência disso. Logo, foram consideradas nesse estudo, as seguintes ameaças à validade interna de grupo único:

    1. *Maturação*: Essa ameaça diz respeito ao comportamento do indivíduo conforme o tempo se passa. O indivíduo pode ficar cansado ou entediado. À fim de tratar essa ameaça, um tempo suficientemente aplicável para a modelagem sem excessos foi estipulado. Além disso, buscou-se a criação de um questionário sucinto e os requisitos propostos para modelagem foram definidos de forma a não cansarem os participantes. Logo, acreditamos que essa ameaça foi tratada e não afetará os resultados;

2. *Instrumentação*: Para tratar a ameaça de influências negativas por parte da instrumentação, os materiais foram revisados por pesquisadores nas áreas fim do estudo, além da aplicação do estudo piloto. Acredita-se que tais ações possibilitaram a criação de um conjunto de instrumentos de qualidade. A ferramenta de modelagem usada como base é uma ameaça à validade, devido ao fato de que os alunos tiveram dificuldades em usá-la e reportaram tais dificuldades no questionário proposto; e
3. *Mortalidade*: Essa ameaça diz respeito à possibilidade de participantes abandonarem o estudo antes de sua conclusão. Para mitigar essa ameaça, os participantes foram contactados e foi apresentado a eles um resumo de como seria o estudo e da necessidade de sua participação por inteiro, apesar da não-obrigatoriedade de sua participação. Além disso, o estudo foi aplicado em horários de aula onde comumente estão presentes os participantes.

- **Validade externa**: Ameaças à validade externa compreendem aquelas que limitam a habilidade de generalizar os resultados do experimento à prática industrial. As seguintes ameaças foram tratadas:

  1. Interação de seleção e tratamento: Para mitigar a ameaça de não selecionarmos o grupo representativo da população-alvo que usaria na indústria a extensão iStarSafety, fornecemos treinamento à todos os participantes, contemplando todos os temas relacionados ao uso da extensão iStar4Safety. Além disso, os indivíduos selecionados estão todos há no mínimo 3 (anos) em cursos de graduação na área de Tecnologia da Informação (Engenharia da Computação e Ciência da Computação). Acreditamos, portanto, que a amostra selecionada aproxima-se da população-alvo; e
  2. Interação de configuração e tratamento: Essa ameaça diz respeito ao material usado pelo estudo e configuração do mesmo não atender ao que é usado na prática industrial. O exemplo usado para modelagem, a Bomba de Infusão de Insulina, foi especificada visando atender às configurações reais da mesma, apesar da necessidade de tratar um subconjunto de funcionalidades devido ao tempo. Acredita-se que o material usado no experimento representa o mini-mundo do que seria utilizado na prática.

- **Validade de conclusão**: A validade de conclusão se preocupa com a habilidade de desenhar conclusões corretas sobre o tratamento e os resultados do estudo.

  1. *Heterogeneidade de sujeitos*: Um grupo muito heterogêneo representa uma ameaça ao estudo devido à diferença de comportamento durante o estudo. Portanto, o grupo escolhido foi de estudantes de graduação que estavam em uma mesma disciplina, tendo formação e conhecimento similares;

2. *Experiência dos sujeitos*: Sujeitos inexperientes podem enviesar as conclusões do experimento. Para mitigar a falta de experiência dos participantes, foram ministrados treinamentos sobre os temas tratados e de relevância para o estudo; e
3. *Confiabilidade da medição* A instrumentação de má qualidade pode criar resultados não confiáveis. Com o intuito de tratar essa ameaça, os instrumentos desse estudo foram avaliados por especialistas envolvidos em nosso grupo de pesquisa, quatro no total, e reavaliados através do estudo piloto aplicado.

Validade de Construtores: A validade de construtores se preocupa em generalizar o resultado do estudo para a teoria que está por trás do mesmo. Algumas ameaças estão relacionadas ao projeto do estudo e outras a fatores sociais.

1. *Viés de mono-operação:* Essa ameaça diz respeito ao uso de somente uma variável independente, sujeito ou tratamento, refletindo na completude de resultados. Nesse estudo utilizamos um único tratamento. Porém, considerando a complexidade de Sistemas Críticos de Segurança somado ao tempo de aplicação do estudo e nível de conhecimento dos participantes, acredita-se que foi o melhor caminho a ser tomado. Além disso, à fim de mitigar esse problema, as questões do formulário visam adicionar mais conteúdo de análise;
2. *Interação de testes e tratamento:* Esta ameaça trata da possibilidade de testes de tratamento enviesarem o resultado final do experimento. Para evitar essa ameaça, evitamos utilizar requisitos do Sistema de Bomba de Insulina que seria modelado no estudo como exemplos de uso da extensão nas aulas dadas;
3. *Adivinhação de hipótese:* Os sujeitos podem tentar descobrir o propósito do estudo e enviesar seu comportamento de respostas e participação à fim de contribuir positivamente ou negativamente para o alcance do mesmo. Para mitigar esse efeito, os participantes foram incentivados à serem imparciais nas suas respostas, além de estarem cientes de que sua participação seria tratada anonimamente; e
4. *Apreensão sobre avaliação:* Os participantes podem se sentir apreensivos por estarem participando do estudo e estarem sendo avaliados. Para mitigar essa ameaça, um acordo de confidencialidade foi assinado e foi ratificado que a participação não traria ônus na disciplina cursada.

6.9 CONCLUSÃO

O capítulo seis mostrou a avaliação da extensão iStar4Safety realizada através de um método empírico, modelagem de um cenário mais coleta de opiniões dos participantes através de um survey, aplicado à alunos de graduação do Centro de Informática da UFPE.

O estudo realizado foi apresentado em detalhes e teve o objetivo de analisar a modelagem de Sistemas Críticos de Segurança, através da modelagem de um Sistema de Bomba de Infusão de Insulina.

O projeto do estudo foi, portanto, definido. Logo após, conforme pode ser visto na seção 6.3, os instrumentos do estudo (disponibilizados nos apêndices) foram desenvolvidos, visando uma maior qualidade na execução e nos resultados desse estudo. Para avaliar o estudo e seus instrumentos, um piloto foi aplicado a uma turma de pós-gradução da UFPE. Tanto os participantes do estudo como os participantes do piloto estavam inscritos em aulas de disciplinas de Engenharia de Requisitos no período do estudo. O piloto contribuiu para melhoria tanto do projeto como dos instrumentos usados.

A população alvo recebeu aulas dos temas associados ao estudo (18 horas no total): linguagens iStar e iStar4Safety, segurança de sistemas, ferramenta piStar-4Safety e ainda sobre o Sistema referente à modelagem.

Após a aplicação da avaliação, que consistiu na modelagem de um Sistema de Bomba de Infusão de Insulina e à um questionário sobre a extensão, os dados foram analisados.

Quanto à análise de resultados, primeiramente foram verificados o perfil acadêmico e nível de conhecimento dos estudantes quanto aos temas relacionados ao estudo. Em seguida avaliamos a eficiência de iStar4Safety. Outro ponto avaliado foi a usabilidade da extensão. A efetividade de iStar4Safety foi analisada de acordo com a visão dos participantes. Por fim, os modelos desenvolvidos pelos participantes do estudo foram avaliados quanto à corretude sintática, complexidade estrutural e similaridade comportamental.

Apresentou-se também uma avaliação sobre a validade do estudo aplicado, para demonstrar o nível de confiabilidade do mesmo.

Após a apresentação da a avaliação de iStar4Safety, é necessário apresentar as conclusões obtidas com essa pesquisa. No capítulo seguinte, tais conclusões são discutidas e discorremos sobre as limitações encontradas. As contribuições do trabalho são por fim elencadas e trabalhos futuros são sugeridos.

7 CONCLUSÃO

Este capítulo apresenta as conclusões obtidas com esse trabalho de pesquisa. São discutidos os itens realizados, bem como as limitações encontradas. Além disso, apresenta-se as contribuições do trabalho, bem como os trabalhos que podem adicionar valor ao que foi até então produzido.

7.1 DISCUSSÃO

O objetivo principal do trabalho realizado foi responder à questão de pesquisa: "Quais construtores uma linguagem orientada à objetivos deveria ter para modelar requisitos iniciais de segurança sem comprometer a simplicidade da linguagem?". Com o intuito de responder à essa questão, definindo assim o conjunto de novos construtores que integrariam a extensão, foi utilizado então o processo PRISE, proposto por Gonçalves, Araújo e Castro (2018) que apresenta seis subprocessos para a criação de uma extensão de iStar que atenda à melhores práticas propostas por especialistas (GONÇALVES et al., 2018c).

A extensão deveria ser simples, possuindo poucos construtores, em concordância com a quantidade de construtores de iStar, deveria ser conservativa e ao mesmo tempo apoiar a modelagem inicial necessária à sistemas tão complexos como são os Sistemas Críticos de Segurança. Este foi o primeiro objetivo específico.

Para a definição dos conceitos necessários para modelagem de requisitos iniciais, ou seja, o objetivo específico dois, foi usado como base o trabalho de Vilela et al. (2017b) que propôs os conceitos necessários para a análise inicial de requisitos de segurança, atendendo à Análise Preliminar de Perigos(PSA), realizada para Sistemas Críticos de Segurança. Dos quinze conceitos definidos, foram priorizados os conceitos-chave devido ao fato dos mesmos serem utilizados para a criação de documentos da Análise Preliminar de Segurança, sendo eles: acidente, perigo, causa de perigo, condição ambiental e requisito funcional de *safety*. Além desses conceitos, concluiu-se que seria necessário a inserção de mais 3 conceitos à fim de possibilitar uma modelagem mais completa de requisitos de segurança iniciais: recursos, estratégias de segurança além do nível de impacto do acidente.

Sendo assim, foi realizada uma busca por construtores que poderiam ser reusados. Por fim, foi definido que a extensão reusaria a ideia de obstáculo, originalmente proposta pela linguagem KAOS (LAMSWEERDE; LETIER, 2000), para a modelagem de perigos. Seriam então adicionados quatro novos construtores, um link e uma propriedade:

- Um objetivo do tipo segurança - **Objetivo de Segurança**;
- Um obstáculo ao objetivo de segurança, que seria portanto um anti-objetivo, ou seja, um obstáculo para a satisfação do objetivo - **Perigo**;

- Ações concretas para mitigar os perigos, formando estratégias de segurança - **Tarefas de Segurança** e **Recursos de Segurança**.
- Um link que relacionaria perigos à objetivos de segurança que os mesmos obstruem - **Link Obstrui**;
- Uma propriedade no construtor objetivo de segurança, que definiria o maior nível de impacto de um acidente provocado pela não-realização de tal objetivo de segurança - **nível de impacto do acidente**.

Para atender ao terceiro objetivo específico, e assim melhor definir os elementos da nova linguagem, foi especificado o metamodelo de iStar4Safety. O metamodelo da linguagem iStar 2.0 foi preservado, devido ao fato da extensão ser conservativa. Para complementar as definições dos novos construtores, foram definidas, textualmente, as regras de validação que não puderam ser representadas no metamodelo.

O objetivo específico 4 (quatro), consistiu na criação da sintaxe concreta da extensão, ou seja, os elementos gráficos de iStar4Safety. Adotou-se uma abordagem de peso-leve (*light-weight*) no desenvolvimento da extensão. Os elementos foram definidos preservando o formato dos elementos estendidos do iStar 2.0 correspondentes, tendo apenas o acréscimo de estereótipos e mudança de cores. Os **perigos** serão da cor vermelha e os demais elementos de segurança serão representados pela cor rosa, acrescidos, conforme já mencionado, do nome do construtor na forma de um estereótipo.

Chegou-se a conclusão de que **perigos** obstruem elementos do tipo **objetivo de segurança**, podendo ser refinados para incluir as causas e condições ambientais que o provocam. Para mitigar o surgimento de **perigos**, suas causas deveriam ser tratadas. Portanto, foi definido que estratégias de segurança seriam associadas à **perigos**-folha para definir como mitigar perigos. As estratégias de segurança são então formadas por ações concretas, sendo representadas por **tarefas de segurança** e/ou **recursos de segurança**. Devido à natureza social de iStar, a estratégia de segurança deve então ser associada por dependência ao ator que a realiza, salvo no caso em que ela é realizada pelo ator em que está contida. Além disso, caso o **objetivo de segurança** seja obstruído, um **acidente** pode acontecer. O nível de impacto de tal acidente deve então ser indicado como uma propriedade presente no elemento **objetivo de segurança**.

Como preconizado pelo PRISE, foi realizada uma verificação sobre completude, consistência e falta de conflitos entre a extensão iStar4Safety e a linguagem iStar 2.0 e demais extensões. O resultado gerado mostrou que iStar4Safety atendeu satisfatoriamente à verificação.

Como passo adicional, conforme preconizado pelo objetivo específico 5 (cinco), foi proposta uma ferramenta para modelagem de iStar4Safety. A ferramenta criada, piStar-4Safety foi uma extensão adicionada à ferramenta piStar que modela a linguagem iStar 2.0. É importante salientar que tal ferramenta foi tratada como uma ameaça à validade

da avaliação empírica aplicada, devido à algumas dificuldades citadas pelos participantes ao utilizarem a ferramenta.

Um conjunto de diretrizes foi proposta para guiar a modelagem de requisitos de segurança iniciais utilizando a extensão iStar4Safety, com dicas associadas ao uso da ferramenta. Como parte do guia, foram propostos, ainda, alguns pontos à serem verificados para garantir a completude da modelagem do ponto de vista da sintaxe.

Uma ilustração da extensão, à fim de atender ao objetivo específico 6 (seis), foi então realizada. Foi realizada uma ilustração da aplicabilidade da linguagem iStar4Safety na modelagem de um novo Sistema Crítico de Segurança: um Sistema de Cruzamentos de Ferrovias. Foram utilizadas para a modelagem, as diretrizes propostas no guia de modelagem com a extensão, sendo realizada uma iteração para ilustrar o processo de modelagem com a linguagem iStar4Safety.

Por fim, atendendo ao objetivo específico 7 (sete), realizou-se uma avaliação de iStar4Safety com a aplicação de um survey em uma turma de graduação formada por 16 (dezesseis) alunos de Engenharia da Computação e Ciência da Computação, do sexto ao décimo terceiro período do seu respectivo curso que estavam cursando a disciplina de Especificação de Requisitos e Validação de Sistemas do Centro de Informática da UFPE. O objetivo do survey foi avaliar a extensão através da modelagem de um Sistema Crítico de Segurança, a Bomba de Infusão de Insulina, em uma versão mais simples e através de respostas ao survey aplicado após a modelagem.

A modelagem resultante da avaliação foi verificada em relação à corretude sintática, complexidade estrutural e similaridade comportamental dos modelos gerados por cada participante. Quanto à corretude sintática, verificou-se que a taxa de erros foi de 0 à 4 erros por participante, tendo somente 4 participantes (25%) com 3 à 4 erros. Os erros encontrados, 24 (vinte e quatro) no total, foram classificados em categorias e analisados.

Quanto à complexidade estrutural, foi verificado que a média de construtores usada pelos participantes foi geralmente similar e as diferenças se explicam por estilo de modelagem próprio ou por falta de conhecimento sobre o construtor, no caso de **recursos de segurança**.

No caso da propriedade de similaridade comportamental, observou-se que todos os participantes modelaram todas as funcionalidades propostas para segurança, porém devido aos erros sintáticos já apresentados, algumas funcionalidades não foram modeladas corretamente. Sendo assim, acreditamos que um tempo maior de treinamento no exemplo usado e a adição das regras de validação na ferramenta teriam contribuído bastante para que mais funcionalidades fossem modeladas corretamente. Demais detalhes sobre as modelagens realizadas são explicados na subseção 6.7.5.

As propriedades de eficiência, usabilidade e efetividade também foram verificadas. A avaliação desses itens se deu pelas respostas obtidas através do questionário aplicado somado à análise dos modelos desenvolvidos pelos participantes.

Quanto à eficiência, verificamos que o tempo médio gasto pelos participantes foi similar.

Quanto à usabilidade, a extensão foi bem avaliada pelos participantes, sendo considerada de fácil uso. Essa era uma das metas à serem alcançadas.

No quesito efetividade, foram avaliados cada construtor de iStar4Safety. No geral, os participantes avaliaram bem cada construtor, sendo as estratégias de segurança os construtores menos bem avaliados, com média 4,5/5. A maioria dos participantes disseram que indicariam a extensão tanto para profissionais com experiência como para aqueles sem experiência em segurança. Os participantes propuseram algumas mudanças, tais como: melhoria da propriedade do nível de impacto do acidente e melhoria das cores usadas. A melhoria para a propriedade de nível de impacto do acidente pode ser implantada futuramente na ferramenta. A alteração de cores, será analisada para aplicabilidade em uma versão futura de iStar4Safety.

7.2 LIMITAÇÕES

Algumas limitações foram percebidas durante o desenvolvimento desse trabalho. Apesar do trabalho utilizado como base para a definição dos conceitos candidatos á modelagem de uma Análise Preliminar de Perigos propor quinze conceitos para a modelagem de Sistemas Críticos de Segurança, tais conceitos não foram todos inseridos, devido à alguns fatores. Os conceitos propostos pelo trabalho de Vilela et al. (2017b) que não foram modelados ou foram modelados por outros conceitos foram: representação de restrições, obstáculos, pré/pós-condições, como um evento afeta a segurança do sistema, nível de criticidade de um elemento crítico e descrição textual de requisitos de segurança. Ressalta-se, porém, que o conceito de obstáculo é coberto pela representação de um perigo, sendo perigo o obstáculo para a satisfação de um objetivo de segurança. Quanto aos conceitos de representação de restrições, como um evento afeta a segurança de um sistema e representação do nível de criticidade de um elemento crítico, tais conceitos podem ser completamente ou parcialmente representados através dos links de contribuição. Maiores detalhes sobre os motivos dos conceitos não fazerem parte de iStar4Safety podem ser encontrados na seção 4.1.2.

Outro ponto a ser considerado, foram as regras de validação. É sabido que a melhor forma de definir tais regras é através de uma linguagem formal, como a Linguagem para Restrições de Objetos (Object Constraint Language - OCL) (BRAMBILLA; CABOT; WIMMER, 2012). Porém, devido à restrições de tempo e baseado em como os autores da linguagem iStar 2.0 definem as suas restrições, também foi utilizada a linguagem natural para definir as regras de validação da extensão.

Quanto à ferramenta piStar-4Safety, não foram implementadas as regras de validação da extensão. Porém trata-se de uma melhoria que será realizada na próxima versão

da ferramenta. Ressalta-se que a ferramenta trata-se de um adicional para auxiliar na modelagem de requisitos de segurança provida pela extensão.

Quanto ao estudo avaliativo realizado por meio de um survey, este possibilitou uma análise restrita devido à aplicação de um único tratamento. Porém, sua escolha se justifica pelo tempo disponível para avaliação e por esse tipo de método empírico ser indicado pelo PRISE. Houve o cuidado de criar um ambiente controlado, aplicar o estudo de forma presencial e dar aos participantes o mesmo tratamento, além da utilização, como objetos de avaliação, tanto dos modelos gerados como das respostas do questionário com as avaliações dos participantes. Outro ponto a ser considerado é que todo o estudo foi respaldado pela literatura e planejado à fim de possibilitar evidências com o menor viés possível. Questões relacionadas à validade do estudo foram apresentadas na seção 6.8.

7.3 CONTRIBUIÇÕES

Essa dissertação apresentou a criação de iStar4Safety - uma extensão da linguagem iStar 2.0 para a modelagem de requisitos de segurança iniciais relacionados à Análise Preliminar de Segurança de Sistemas Críticos de Segurança. A nova linguagem proposta é simples, como pode ser observado na avaliação realizada através do survey aplicado, no qual os participantes não apresentaram dificuldades ao modelar o sistema proposto. Além disso, conseguiu-se responder satisfatoriamente a Questão de Pesquisa definida: "Quais construtores uma linguagem orientada à objetivos deveria ter para modelar requisitos iniciais de segurança sem comprometer a simplicidade da linguagem?", com a adição de uma quantidade pequena de novos construtores: quatro construtores gráficos, um link e uma propriedade, reforça a ideia de simplicidade de iStar4Safety. Apresentamos abaixo algumas outras contribuições desse trabalho, demonstrando assim que os objetivos específicos foram atendidos:

- A extensão criada, iStar4Safety. é simples, conservativa e adiciona a menor quantidade possível de novos construtores à linguagem iStar (Objetivo Específico 1).
- Foram selecionados e definidos os conceitos necessários para a modelagem de uma Análise de Segurança Preliminar na fase de Engenharia de Requisitos iniciais (Objetivo Específico 2);
- Foi criado um metamodelo de iStar, preservando o metamodelo nativo de iStar 2.0, além das regras de validação que permitem a implementação da extensão iStar4Safety em ferramentas de modelagem de iStar, tais como piStar Pimentel e Castro (2018), conservando tanto o metamodelo quanto as regras de validação de iStar 2.0; (Objetivo Específico 3);
- A sintaxes concreta da nova linguagem foi definida (Objetivo Específico 4);

- Foi adaptada e disponibilizada uma ferramenta - piStar-4Safety - para modelagem de requisitos iniciais de segurança de Sistemas Críticos de Segurança com a extensão criada (Objetivo Específico 5);
- A extensão foi ilustrada através da modelagem de dois exemplos ilustrativos, modelando um Sistema de Bomba de Infusão de Insulina e um Sistema de Cruzamento de Ferrovias, permitindo a visualização da aplicabilidade de iStar4Safety (Objetivo Específico 6);
- A extensão foi avaliada através de uma avaliação realizada com a modelagem de um cenário mais coleta das opiniões dos participantes através de um survey presencial (Objetivo Específico 7);
- Foi criado um guia com diretrizes de como utilizar a nova linguagem iStar4Safety.
- Através do desenvolvimento da extensão, foi realizada uma avaliação do processo PRISE proposto por (GONÇALVES et al., 2018b).

7.4 TRABALHOS FUTUROS

Alguns trabalhos a serem realizados para complementar ou dar prosseguimento à proposta desta dissertação são listados:

- Analise de como integrar a modelagem das demais análises de segurança (posteriores à Análise Preliminar de Segurança) à Engenharia de Requisitos;
- Implementação das regras de validação na ferramenta piStar-4safety;
- Avaliação da extensão através de experimentos controlados, comparando a extensão criada à outras formas de modelar requisitos iniciais de Sistemas Críticos de Segurança;
- Avaliar os construtores gráficos criados (sintaxe gráfica), realizando estudos empíricos que verificassem sua efetividade e, caso necessário, aplicar melhorias.
- Avaliar a forma de representação do nível de impacto do acidente; e
- Definir uma forma adequada de representar o conceito de restrições, separando as restrições relacionadas à segurança de outras restrições.
- Possíveis adaptações da extensão visando atender normas específicas de segurança de sistemas críticos como as normas: ARP 4761, MIL-STS-882 e IEC61508.

REFERÊNCIAS

ANTON, A. I. Goal-based requirements analysis. In: *Proceedings of the 2Nd International Conference on Requirements Engineering (ICRE '96)*. Washington, DC, USA: IEEE Computer Society, 1996. (ICRE '96), p. 136–. ISBN 0-8186-7252-8. Disponível em: <http://dl.acm.org/citation.cfm?id=850944.853130>.

ARABESTANI, S.; BITSCH, F.; GAYEN, J.-T. Precise definition of the single-track level crossing in radio-based operation in uml notation and specification of safety requirements. In: _____. *Integration of Software Specification Techniques for Applications in Engineering: Priority Program SoftSpez of the German Research Foundation (DFG), Final Report*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2004. p. 119–144. ISBN 978-3-540-27863-4. Disponível em: <https://doi.org/10.1007/978-3-540-27863-4_9>.

ASADI, M.; BAGHERI, E.; GASEVIC, D.; HATALA, M.; MOHABBATI, B. Goal-driven software product line engineering. In: *Proceedings of the 2011 ACM Symposium on Applied Computing*. New York, NY, USA: ACM, 2011. (SAC '11), p. 691–698. ISBN 978-1-4503-0113-8. Disponível em: <http://doi.acm.org/10.1145/1982185.1982336>.

ASNAR, Y.; GIORGINI, P.; MYLOPOULOS, J. Goal-driven risk assessment in requirements engineering. *Requir. Eng.*, Springer-Verlag New York, Inc., Secaucus, NJ, USA, v. 16, n. 2, p. 101–116, jun. 2011. ISSN 0947-3602. Disponível em: <http://dx.doi.org/10.1007/s00766-010-0112-x>.

BELL, R. Introduction to iec 61508. In: *Proceedings of the 10th Australian Workshop on Safety Critical Systems and Software - Volume 55*. Darlinghurst, Australia, Australia: Australian Computer Society, Inc., 2006. (SCS '05), p. 3–12. ISBN 1-920-68237-6. Disponível em: <http://dl.acm.org/citation.cfm?id=1151816.1151817>.

BERRY, D. M. The safety requirements engineering dilemma. In: *Proceedings of the 9th International Workshop on Software Specification and Design*. Washington, DC, USA: IEEE Computer Society, 1998. (IWSSD '98), p. 147–. ISBN 0-8186-8439-9. Disponível em: <http://dl.acm.org/citation.cfm?id=857205.858307>.

Bowen, J.; Stavridou, V. Safety-critical systems, formal methods and standards. *Software Engineering Journal*, v. 8, n. 4, p. 189–209, July 1993. ISSN 0268-6961.

BRAMBILLA, M.; CABOT, J.; WIMMER, M. *Model-Driven Software Engineering in Practice*. 1st. ed. [S.l.]: Morgan & Claypool Publishers, 2012. ISBN 1608458822, 9781608458820.

BROOMFIELD, E.; CHUNG, P. Safety assessment and the software requirements specification. *Reliability Engineering and System Safety*, v. 55, n. 3, p. 295–309, 1997. ISSN 0951-8320. Disponível em: <http://www.sciencedirect.com/science/article/pii/S0951832096001019>.

DALPIAZ, F.; FRANCH, X.; HORKOFF, J. istar 2.0 language guide. *CoRR*, abs/1605.07767, 2016. Disponível em: <http://arxiv.org/abs/1605.07767>.

DARDENNE, A.; LAMSWEERDE, A. van; FICKAS, S. Goal-directed requirements acquisition. *Science of Computer Programming*, v. 20, n. 1, p. 3 – 50, 1993. ISSN 0167-6423. Disponível em: <http://www.sciencedirect.com/science/article/pii/016764239390021G>.

DIJKMAN, R.; DUMAS, M.; DONGEN, B. van; KääRIK, R.; MENDLING, J. Similarity of business process models: Metrics and evaluation. *Information Systems*, v. 36, n. 2, p. 498 – 516, 2011. ISSN 0306-4379. Special Issue: Semantic Integration of Data, Multimedia, and Services. Disponível em: <http://www.sciencedirect.com/science/article/pii/S0306437910001006>.

DONGEN, B. V.; DIJKMAN, R.; MENDLING, J. Measuring similarity between business process models. In: _____. *Seminal Contributions to Information Systems Engineering: 25 Years of CAiSE*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2013. p. 405–419. ISBN 978-3-642-36926-1. Disponível em: <https://doi.org/10.1007/978-3-642-36926-1_33>.

ERICSON, C. A. et al. *Hazard analysis techniques for system safety*. [S.l.]: John Wiley & Sons, 2015.

FIRESMITH, D. Engineering safety requirements, safety constraints, and safety-critical requirements. *Journal of object technology*, v. 3, n. 3, p. 27–27, 2004. ISSN 1660-1769. Disponível em: <http://dx.doi.org/10.5381/jot.2004.3.3.c3>.

FRANCH, X.; MATE, A.; TRUJILLO, J. C.; CARES, C. On the joint use of i* with other modelling frameworks: A vision paper. In: *Proceedings of the 2011 IEEE 19th International Requirements Engineering Conference*. Washington, DC, USA: IEEE Computer Society, 2011. (RE '11), p. 133–142. ISBN 978-1-4577-0921-0. Disponível em: <http://dx.doi.org/10.1109/RE.2011.6051642>.

GHANAVATI, S.; AMYOT, D.; RIFAUT, A. Legal goal-oriented requirement language (legal grl) for modeling regulations. In: *Proceedings of the 6th International Workshop on Modeling in Software Engineering*. New York, NY, USA: ACM, 2014. (MiSE 2014), p. 1–6. ISBN 978-1-4503-2849-4. Disponível em: <http://doi.acm.org/10.1145/2593770.2593780>.

GLASS, R. L. The software-research crisis. *IEEE Softw.*, IEEE Computer Society Press, Los Alamitos, CA, USA, v. 11, n. 6, p. 42–47, nov. 1994. ISSN 0740-7459. Disponível em: <https://doi.org/10.1109/52.329400>.

GONÇALVES, E. *Prise Prise Process*. 2018. Disponível em: <http://www.cin.ufpe.br/~ejtg/prise/#diagram/815dfa2b-a56e-4d0a-bf31-4a14f1b7c710>.

GONÇALVES, E.; ARAÚJO, J.; CASTRO, J. Towards extension mechanisms in istar 2.0. In: *iSTAR@ CAiSE*. [S.l.: s.n.], 2018.

GONÇALVES, E.; CASTRO, J.; ARAúJO, J.; HEINECK, T. A systematic literature review of istar extensions. *Journal of Systems and Software*, v. 137, p. 1 – 33, 2018. ISSN 0164-1212. Disponível em: <http://www.sciencedirect.com/science/article/pii/S0164121217302741>.

GONÇALVES, E.; HEINECK, T.; ARAúJO, J.; CASTRO, J. A catalogue of istar extensions. In: WER18. Rio de Janeiro, 2018.

GONÇALVES, E.; OLIVEIRA, M. A. de; MONTEIRO, I.; CASTRO, J.; ARAÚJO, J. Understanding what is important in istar extension proposals: the viewpoint of researchers. *Requirements Engineering*, Jul 2018. ISSN 1432-010X. Disponível em: <https://doi.org/10.1007/s00766-018-0302-5>.

GONÇALVES ENYO, A. J. C. J. Prise: A process to conduct istar extensions. 2019.

GOODMAN, N. *Languages of art: An approach to a theory of symbols*. [S.l.]: Hackett publishing, 1968.

HORKOFF, J.; YU, E. Interactive goal model analysis for early requirements engineering. *Requir. Eng.*, Springer-Verlag New York, Inc., Secaucus, NJ, USA, v. 21, n. 1, p. 29–61, mar 2016. ISSN 0947-3602. Disponível em: <http://dx.doi.org/10.1007/s00766-014-0209-8>.

KELLY, S.; TOLVANEN, J.-P. *Domain-Specific Modeling*. USA: John Wiley & Sons, Inc., 2007. ISBN 0470036664.

KITCHENHAM, B.; PFLEEGER, S. L. Principles of survey research part 4: Questionnaire evaluation. *SIGSOFT Softw. Eng. Notes*, ACM, New York, NY, USA, v. 27, n. 3, p. 20–23, maio 2002. ISSN 0163-5948. Disponível em: <http://doi.acm.org/10.1145/638574.638580>.

KITCHENHAM, B.; PFLEEGER, S. L. Principles of survey research: Part 5: Populations and samples. *SIGSOFT Softw. Eng. Notes*, ACM, New York, NY, USA, v. 27, n. 5, p. 17–20, sep 2002. ISSN 0163-5948. Disponível em: <http://doi.acm.org/10.1145/571681.571686>.

KITCHENHAM, B.; PFLEEGER, S. L. Principles of survey research part 6: Data analysis. *SIGSOFT Softw. Eng. Notes*, ACM, New York, NY, USA, v. 28, n. 2, p. 24–27, mar. 2003. ISSN 0163-5948. Disponível em: <http://doi.acm.org/10.1145/638750.638758>.

KITCHENHAM, B. A.; PFLEEGER, S. L. Principles of survey research part 2: Designing a survey. *SIGSOFT Softw. Eng. Notes*, ACM, New York, NY, USA, v. 27, n. 1, p. 18–20, jan. 2002. ISSN 0163-5948. Disponível em: <http://doi.acm.org/10.1145/566493.566495>.

KITCHENHAM, B. A.; PFLEEGER, S. L. Principles of survey research: Part 3: Constructing a survey instrument. *SIGSOFT Softw. Eng. Notes*, ACM, New York, NY, USA, v. 27, n. 2, p. 20–24, mar. 2002. ISSN 0163-5948. Disponível em: <http://doi.acm.org/10.1145/511152.511155>.

KITCHENHAM, B. A.; PFLEEGER, S. L. Personal opinion surveys. In: _____. *Guide to Advanced Empirical Software Engineering*. London: Springer London, 2008. p. 63–92. ISBN 978-1-84800-044-5. Disponível em: <https://doi.org/10.1007/978-1-84800-044-5_3>.

KNIGHT, J. C. Safety critical systems: Challenges and directions. In: *Proceedings of the 24th International Conference on Software Engineering*. New York, NY, USA: ACM, 2002. (ICSE '02), p. 547–550. ISBN 1-58113-472-X. Disponível em: <http://doi.acm.org/10.1145/581339.581406>.

LAMSWEERDE, A. V. Elaborating security requirements by construction of intentional anti-models. In: *Proceedings of the 26th International Conference on Software Engineering*. Washington, DC, USA: IEEE Computer Society, 2004. (ICSE '04), p. 148–157. ISBN 0-7695-2163-0. Disponível em: <http://dl.acm.org/citation.cfm?id=998675.999421>.

LAMSWEERDE, A. V.; LETIER, E. Handling obstacles in goal-oriented requirements engineering. *IEEE Trans. Softw. Eng.*, IEEE Press, Piscataway, NJ, USA, v. 26, n. 10, p. 978–1005, out. 2000. ISSN 0098-5589. Disponível em: <https://doi.org/10.1109/32.879820>.

LAMSWEERDE, A. van; LETIER, E. From object orientation to goal orientation: A paradigm shift for requirements engineering. In: WIRSING, M.; KNAPP, A.; BALSAMO, S. (Ed.). *Radical Innovations of Software and Systems Engineering in the Future*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2004. p. 325–340. ISBN 978-3-540-24626-8.

LAPOUCHNIAN, A. Goal-oriented requirements engineering: An overview of the current research. 01 2005.

LAPOUCHNIAN, A.; LESPÉRANCE, Y. Modeling mental states in agent-oriented requirements engineering. In: DUBOIS, E.; POHL, K. (Ed.). *Advanced Information Systems Engineering*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2006. p. 480–494. ISBN 978-3-540-34653-1.

Lei, Y.; Ben, K.; He, Z. A framework for self-adaptive software based on extended tropos goal model. In: *2015 7th International Conference on Intelligent Human-Machine Systems and Cybernetics*. [S.l.: s.n.], 2015. v. 2, p. 533–536.

LEVESON, N. G. *Safeware: System Safety and Computers*. New York, NY, USA: ACM, 1995. ISBN 0-201-11972-2.

LEVESON, N. G. *Engineering a Safer World: Systems Thinking Applied to Safety*. Massachusetts, London, England: Mit Press, 2011. ISBN 9-780-26201662-9.

LUCASSEN, G.; DALPIAZ, F.; WERF, J. M.; BRINKKEMPER, S. The use and effectiveness of user stories in practice. In: *Proceedings of the 22Nd International Working Conference on Requirements Engineering: Foundation for Software Quality - Volume 9619*. Berlin, Heidelberg: Springer-Verlag, 2016. (REFSQ 2016), p. 205–222. ISBN 978-3-319-30281-2. Disponível em: <https://doi.org/10.1007/978-3-319-30282-9_14>.

LUTZ, R. R. Targeting safety-related errors during software requirements analysis. *SIGSOFT Softw. Eng. Notes*, ACM, New York, NY, USA, v. 18, n. 5, p. 99–106, dez. 1993. ISSN 0163-5948. Disponível em: <http://doi.acm.org/10.1145/167049.167069>.

LUTZ, R. R. Software engineering for safety: A roadmap. In: *Proceedings of the Conference on The Future of Software Engineering*. New York, NY, USA: ACM, 2000. (ICSE '00), p. 213–226. ISBN 1-58113-253-0. Disponível em: <http://doi.acm.org/10.1145/336512.336556>.

MARTINS, L. E. G.; FARIA, H. d.; VECCHETE, L.; CUNHA, T.; OLIVEIRA, T. d.; CASARINI, D. E.; COLUCCI, J. A. Development of a low-cost insulin infusion pump: Lessons learned from an industry case. In: *Proceedings of the 2015 IEEE 28th International Symposium on Computer-Based Medical Systems*. Washington, DC, USA:

IEEE Computer Society, 2015. (CBMS '15), p. 338–343. ISBN 978-1-4673-6775-2. Disponível em: <http://dx.doi.org/10.1109/CBMS.2015.14>.

MARTINS, L. E. G.; GORSCHEK, T. Requirements engineering for safety-critical systems: A systematic literature review. *Information and Software Technology*, v. 75, p. 71–89, 2016. ISSN 0950-5849. Disponível em: <http://www.sciencedirect.com/science/article/pii/S0950584916300568>.

MEDIKONDA, B. S.; PANCHUMARTHY, S. R. A framework for software safety in safety-critical systems. *SIGSOFT Softw. Eng. Notes*, ACM, New York, NY, USA, v. 34, n. 2, p. 1–9, fev. 2009. ISSN 0163-5948. Disponível em: <http://doi.acm.org/10.1145/1507195.1507207>.

MELLADO, D.; MOURATIDIS, H.; FERNáNDEZ-MEDINA, E. Secure tropos framework for software product lines requirements engineering. *Comput. Stand. Interfaces*, Elsevier Science Publishers B. V., Amsterdam, The Netherlands, The Netherlands, v. 36, n. 4, p. 711–722, jun 2014. ISSN 0920-5489. Disponível em: <http://dx.doi.org/10.1016/j.csi.2013.12.006>.

MIL-STD-882E System Safety. [S.l.], 2012. Disponível em: <http://www.everyspec.com/MIL-STD/MIL-STD-0800-0899/MIL_STD_882D_934/>.

MIRANDA, D.; GENERO, M.; PIATTINI, M. Empirical validation of metrics for uml statechart diagrams. In: CAMP, O.; FILIPE, J. B. L.; HAMMOUDI, S.; PIATTINI, M. (Ed.). *Enterprise Information Systems V*. Dordrecht: Springer Netherlands, 2005. p. 101–108. ISBN 978-1-4020-2673-7.

MOODY, D. L.; HEYMANS, P.; MATULEVIČIUS, R. Visual syntax does matter: improving the cognitive effectiveness of the i* visual notation. *Requirements Engineering*, v. 15, n. 2, p. 141–175, Jun 2010. ISSN 1432-010X. Disponível em: <https://doi.org/10.1007/s00766-010-0100-1>.

MORANDINI, M.; PENSERINI, L.; PERINI, A.; MARCHETTO, A. Engineering requirements for adaptive systems. *Requir. Eng.*, Springer-Verlag New York, Inc., Secaucus, NJ, USA, v. 22, n. 1, p. 77–103, mar. 2017. ISSN 0947-3602. Disponível em: <https://doi.org/10.1007/s00766-015-0236-0>.

MOURATIDIS, H.; GIORGINI, P. Secure tropos: A security-oriented extension of the tropos methodology. *Int. J. Soft. Eng. Knowl. Eng.*, World Scientific Publishing Co., v. 17, n. 02, p. 285–309, apr 2007. Disponível em: <http://dx.doi.org/10.1142/s0218194007003240>.

MOURATIDIS, H.; ISLAM, S.; KALLONIATIS, C.; GRITZALIS, S. A framework to support selection of cloud providers based on security and privacy requirements. *J. Syst. Softw.*, Elsevier Science Inc., New York, NY, USA, v. 86, n. 9, p. 2276–2293, set. 2013. ISSN 0164-1212. Disponível em: <http://dx.doi.org/10.1016/j.jss.2013.03.011>.

NUSEIBEH, B.; EASTERBROOK, S. Requirements engineering: A roadmap. In: *Proceedings of the Conference on The Future of Software Engineering*. New York, NY, USA: ACM, 2000. (ICSE '00), p. 35–46. ISBN 1-58113-253-0. Disponível em: <http://doi.acm.org/10.1145/336512.336523>.

OBJECTIVER. *A KAOS Tutorial*. 1. ed. [S.l.], 2007.

OMG. *OMG Meta Object Facility (MOF) Core Specification, Version 2.4.1*. 2013. Disponível em: <http://www.omg.org/spec/MOF/2.4.1>.

PIMENTEL, J.; CASTRO, J. pistar tool – a pluggable online tool for goal modeling. In: . [S.l.: s.n.], 2018. p. 498–499.

SCHNIEDER, E. Specification methodology, case studies, and experiments – an introduction to the subject area of traffic control systems. In: _____. *Integration of Software Specification Techniques for Applications in Engineering: Priority Program SoftSpez of the German Research Foundation (DFG), Final Report*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2004. p. 89–95. ISBN 978-3-540-27863-4. Disponível em: <https://doi.org/10.1007/978-3-540-27863-4_7>.

SOMMERVILLE, I. *Software Engineering*. 9th. ed. USA: Addison-Wesley Publishing Company, 2010. ISBN 0137035152, 9780137035151.

STÅLHANE, T.; MYKLEBUST, T. Early safety analysis. In: *Proceedings of the Scientific Workshop Proceedings of XP2016*. New York, NY, USA: ACM, 2016. (XP '16 Workshops), p. 19:1–19:6. ISBN 978-1-4503-4134-9. Disponível em: <http://doi.acm.org/10.1145/2962695.2962714>.

SVAHNBERG, M.; AURUM, A.; WOHLIN, C. Using students as subjects - an empirical evaluation. In: *Proceedings of the Second ACM-IEEE International Symposium on Empirical Software Engineering and Measurement*. New York, NY, USA: ACM, 2008. (ESEM '08), p. 288–290. ISBN 978-1-59593-971-5. Disponível em: <http://doi.acm.org/10.1145/1414004.1414055>.

VILELA, J.; CASTRO, J.; MARTINS, L. E. G.; GORSCHEK, T. Integration between requirements engineering and safety analysis. *J. Syst. Softw.*, Elsevier Science Inc., New York, NY, USA, v. 125, n. C, p. 68–92, mar. 2017. ISSN 0164-1212. Disponível em: <https://doi.org/10.1016/j.jss.2016.11.031>.

VILELA, J.; CASTRO, J.; MARTINS, L. E. G.; GORSCHEK, T.; SILVA, C. Specifying safety requirements with gore languages. In: *Proceedings of the 31st Brazilian Symposium on Software Engineering*. New York, NY, USA: ACM, 2017. (SBES'17), p. 154–163. ISBN 978-1-4503-5326-7. Disponível em: <http://doi.acm.org/10.1145/3131151.3131175>.

VILELA, J.; CASTRO, J.; MARTINS, L. E. G.; GORSCHEK, T. Safe-re: A safety requirements metamodel based on industry safety standards. In: *Proceedings of the XXXII Brazilian Symposium on Software Engineering*. New York, NY, USA: ACM, 2018. (SBES '18), p. 196–201. ISBN 978-1-4503-6503-1. Disponível em: <http://doi.acm.org/10.1145/3266237.3266242>.

VILELA, J.; CASTRO, J.; PIMENTEL, J. A systematic process for obtaining the behavior of context-sensitive systems. *Journal of Software Engineering Research and Development*, v. 4, n. 1, p. 2, May 2016. ISSN 2195-1721. Disponível em: <https://doi.org/10.1186/s40411-016-0028-3>.

WOHLIN, C.; RUNESON, P.; HST, M.; OHLSSON, M. C.; REGNELL, B.; WESSLN, A. *Experimentation in Software Engineering*. [S.l.]: Springer Publishing Company, Incorporated, 2012. ISBN 3642290434, 9783642290435.

YU, E. S.-K. *Modelling Strategic Relationships for Process Reengineering*. Tese (Doutorado), Toronto, Ont., Canada, Canada, 1995. AAINN02887.

ZHANG, Y.; JETLEY, R.; JONES, P. L.; RAY, A. Generic safety requirements for developing safe insulin pump software. *Journal of Diabetes Science and Technology*, v. 5, n. 6, p. 1403–1419, 2011. PMID: 22226258. Disponível em: <https://doi.org/10.1177/193229681100500612>.

ZHANG, Y.; JONES, P. L.; JETLEY, R. A hazard analysis for a generic insulin infusion pump. *Journal of Diabetes Science and Technology*, v. 4, n. 2, p. 263–283, 2010. PMID: 20307387. Disponível em: <https://doi.org/10.1177/193229681000400207>.

ZOUGHBI, G.; BRIAND, L.; LABICHE, Y. A uml profile for developing airworthiness-compliant (rtca do-178b), safety-critical software. In: *Proceedings of the 10th International Conference on Model Driven Engineering Languages and Systems*. Berlin, Heidelberg: Springer-Verlag, 2007. (MODELS'07), p. 574–588. ISBN 3-540-75208-0, 978-3-540-75208-0. Disponível em: <http://dl.acm.org/citation.cfm?id=2394101.2394153>.

ZOUGHBI, G.; BRIAND, L.; LABICHE, Y. Modeling safety and airworthiness (rtca do-178b) information: Conceptual model and uml profile. *Softw. Syst. Model.*, Springer-Verlag New York, Inc., Secaucus, NJ, USA, v. 10, n. 3, p. 337–367, jul. 2011. ISSN 1619-1366. Disponível em: <http://dx.doi.org/10.1007/s10270-010-0164-x>.

APÊNDICE A – ARTEFATOS DO PROCESSO PRISE

Apresentamos o checklist de verificação de completude, consistência e conflitos que é um artefato do processo PRISE.

Checklist de verificação de completude, consistência e conflitos

Extensão: *iStar4Safety* para Sistemas Críticos de Segurança.

Autora: Sarah Moniky S. Ribeiro

Completude			
Em quais níveis a extensão de *iStar* foi aplicada? [X]Definição de conceitos[X] Metamodelo [X] Regras de validação[X] Sintaxe concreta			
Consistência			
Liste abaixo os conceitos e sua representação em cada nível			
Conceito	**Definição do conceito**	**Metamodelo**	**Sintaxe concreta**
Objetivo de Segurança	[X]	[X]	[X]
Tarefa de Seegurança	[X]	[X]	[X]
Recurso de Segurança	[X]	[X]	[X]
Perigo	[X]	[X]	[X]
Link Obstrui	[X]	[X]	[X]

Presença de nós e links da sintaxe padrão de iStar 2.0.			
Construto		**Metamodelo**	**Sintaxe concreta**
Objetivo		[X]	[X]
Qualidade		[X]	[X]
Recurso		[X]	[X]
Tarefa		[X]	[X]
Ator		[X]	[X]
Agente		[X]	[X]
Papel		[X]	[X]
Refinemento		[X]	[X]
Qualificação		[X]	[X]
Contribuição (make, help, break, hurt)		[X]	[X]
NeededBy		[X]	[X]
Dependência		[X]	[X]
Links de Associação	é-um	[X]	[X]
	Participa-em	[X]	[X]

Conflitos				
Liste os conceitos e verifique a ocorrência de conflitos entre os construtos da extensão e construtos de extensões existentes.				
Conceito	**Conflitos**			
	Um construto com dois ou mais símbolos	Dois ou mais construtos com somente um símbolo	Representação errada de construtos de *iStar*	Construto que não é parte da extensão
Objetivo de Segurança	[0]	[0]	[0]	[0]
Tarefa de Segurança	[0]	[0]	[0]	[0]
Recurso de Segurança	[0]	[0]	[0]	[0]
Perigo	[0]	[0]	[0]	[0]
Link Obstrui	[0]	[0]	[0]	[0]
Detalhes sobre problemas: ----				

APÊNDICE B – MATERIAL DE SUPORTE DO SURVEY

Apresenta-se nessa seção o material de apoio utilizado para a avaliação da extensão iStar4Safety,a saber:

- Documento de requisitos da Bomba de Infusão de Insulina para criação do modelo SD pelos participantes do survey - Anexo B.1;
- Um guia para modelagem com iStar4Safety, que serviu de material de apoio para modelagem do estudo empírico - Anexo B.2;
- O questionário no aplicativo "google forms"sobre o uso da extensão, após a modelagem realizada - Anexo B.3;
- Um resumo das aulas dadas em sala de aula, á fim de instruir os participantes sobre os temas relevantes ao estudo - Anexo B.4;
- Um modelo SR base da Bomba de Infusão de Insulina sem construtos de iStar4Safety, disponibilizado aos participantes da avaliação - Anexo B.5;
- Um modelo final SR da Bomba de Infusão de Insulina com construtos de iStar4Safety, usado para comparação de modelos criados durante aplicação do survey - Anexo B.6;

B.1 REQUISITOS PARA MODELAGEM DA BOMBA DE INFUSÃO DE INSULINA

O seguinte documento de requisitos foi disponibilizado para ser modelado em sala de aula, como parte do Survey que foi aplicado, visando avaliar a extensão iStar4Safety.

Modelagem de sistema *safety-critical* utilizando iStar4Safety

Data: 30/11/2018

Nome: _____

Horário Inicial: _____

Horário Final: _____

O objetivo dessa atividade é modelar os conceitos *safety* relacionados à bomba de infusão de insulina. Portanto, considere a especificação abaixo de tal sistema. A modelagem em iStar dos requisitos da seção 1 já foi criada e será fornecida em um pré-modelo com construtos iStar jpá inseridos.

Você deve modelar os requisitos safety (ferramenta Pistar) apresentados na **seção 2**. Siga o passo-a-passo que disponibilizamos para auxiliá-lo(a) na sua modelagem.

Salve seu modelo (.txt e .png) pois eles serão enviados no próximo passo.

1 – Requisitos gerais da bomba de infusão de insulina (Pré-modelo)

Bomba de infusão de insulina - Requisitos

Uma bomba de infusão de insulina trata-se de um dispositivo que objetiva aplicar dosagens de insulina de ação rápida através de um cateter colocado sob a pele do paciente, à fim de tratar a diabetes Melitus tipo I, mantendo o nivelamento adequado de glicose no sangue do paciente. A bomba se configura como um sistema *safety-critical*. Nesse exemplo usaremos um subconjunto das funcionalidades de uma bomba de insulina, tendo como base principal dois trabalhos: [MARTINS ET. AL., 2015[1]] e [ZHANG, ET.AL. ,2010[2]].

A bomba de insulina é um sistema composto por uma bateria, um reservatório de insulina (seringa), uma interface com o usuário, um conjunto de infusão e pelo software que faz o controle da bomba.

Um objetivo do **paciente** ao **utilizar a bomba de infusão de insulina**, será **verificar as configurações gerais da mesma**. Ele deverá **verificar a tela**, **alarmes**, **nível de bateria**, e **nível de insulina**. Á fim de prover maior **precisão**, os níveis de bateria e insulina devem ser **apresentados em telas separadas**.

O paciente deseja **ser tratado** através de dois tipos de infusão: **basal** e **bolus**. A infusão basal é aplicada em pequenas doses durante todo o dia. A infusão bolus é uma infusão de aplicações extras, quando necessário.

O paciente deve **configurar a infusão basal** e pode **alterar os perfis armazenados** para ela. Assim como o paciente pode **configurar a infusão bolus** e **alterar as configurações** desse tipo de aplicação.

Ao utilizar a bomba de insulina, o paciente pode **trocar o cartucho do conjunto de infusão**. Para isso, ele precisa **interromper as infusões ativas** e **recolher o êmbolo**.

O paciente pode ainda querer **trocar a bateria do dispositivo**.

Já o **software controlador da bomba de infusão de insulina** deve **gerenciar a bomba de insulina**. Para isso, ele deve **realizar o tratamento adequado do paciente**, **manter tela**

[1] Martins, L.E.G.,de Faria, H.,Vecchete, T.C., de Oliveira, T.,Casarine, D.E., Colucci, J.A. 2015. Development of a Low-Cost Insulin Infusion Pump: Lessons Learned from an Industry Case.

[2] Zhang, Y., Jones, P. L., Jetley, R. 2010. A Hazard Analysis for a Generic Insulin Infusion Pump.

atualizada, **gerenciar alertas (**incluindo **um alerta sobre reservatório vazio)** e **monitorar sensores**.

Para realizar o tratamento adequado do paciente, o software controlador da bomba deve **gerenciar a infusão bolus** e **gerenciar a infusão basal**. O gerenciamento da infusão bolus corresponde à **iniciar bolus** ou **cancelar bolus**. O gerenciamento da infusão basal, significa **iniciar basal** ou **interromper basal**.

O **fornecedor da bomba** tem por objetivo **fornecer suporte ao uso do dispositivo**. Esse suporte é feito por **auxiliar o paciente no uso da bomba** ou por **realizar manutenções no dispositivo e por disponibilizar manuais e políticas de uso e compartilhamento**.

2 – Requisitos *safety* da bomba de infusão de insulina

O principal objetivo do **paciente** ao utilizar a bomba de infusão de insulina é **ter sua diabetes tratada**, já que pessoas com diabetes não tem a produção necessária de insulina pelo pâncreas. À fim de combater esse problema, o paciente **necessita receber as dosagens corretas de insulina**. Um objetivo do paciente é não **receber dosagem de insulina maior que a correta** (overdose), pode ter como consequência mais severa a morte, o que é considerado um acidente de impacto **Catastrófico**. Enquanto outro objetivo é **não receber dosagem menor de insulina do que o correto** (underdose), já que se isso acontecer o paciente pode ficar cego, que é considerado de impacto **Perigoso/Grave**.

O que pode levar o paciente à receber uma dosagem maior de insulina do que o desejado é: **fluxo livre de insulina**. O fluxo livre de insulina pode ser causado por **válvulas no caminho de entrega estarem quebradas**. Á fim de tratar o fato das válvulas estarem quebradas, o paciente deve **verificar constantemente o caminho de entrega da insulina**.

O que pode levar o paciente a receber uma dosagem menor de insulina, é caso aconteça um **vazamento de insulina**, **ou** ainda que o **reservatório de insulina esteja vazio**. **Receber alerta sonoro** ou **verificar reservatório** são as soluções para evitar que o reservatório de insulina fique vazio. Para receber esse alerta *o paciente depende do software enviá-lo*. O vazamento de insulina pode ser causado pelo fato do **paciente se mover e desconectar a bomba sem perceber**. Nesse caso, o tratamento é **verificar constantemente o caminho de entrega da insulina,** que é de responsabilidade do paciente.

O paciente pode receber uma dosagem maior ou menor de insulina também pelo fato do **sistema de entrega de insulina da bomba falhar.** O sistema de entrega pode falhar por **problemas de software ou problemas no hardware**. A solução para problemas tanto de hardware como de software, será **receber suporte de uso**, que é fornecido pelo **Fornecedor da bomba**. Para o suporte de uso, o paciente depende do fornecedor da bomba.

Tanto ao **trocar a bateria**, como ao **trocar o conjunto de infusão**, o paciente **não deseja ser infectado**. Ser infectado por levar a morte, logo é um acidente de impacto **Catrastrófico**. O fato que obstrui esse objetivo é caso a **bomba seja exposta à alguma substância tóxica**. O que pode levar a bomba à ser exposta à alguma substância tóxica, é o fato da **bomba ser compartilhad**a. Á fim de mitigar problemas com o compartilhamento da bomba, deve-se disponibilizar o **documento "Políticas de Compartilhamento"**. Esse documento deve ser disponibilizado pelo **fornecedor da bomba.**

Ao utilizar a bomba, o paciente deseja **não receber descargas elétricas**. Um acidente de choque elétrico teria impacto **Maior**. O que obstrui esse objetivo é o perigo do **dispositivo estar em contato com água**. Uma causa para esse contato seria **chuva**. Para acabar com esse problema, a solução seria **não expor o dispositivo à chuva,** que é de responsabilidade do paciente.

Obrigada por sua participação!

B.2 GUIA PARA MODELAGEM COM ISTAR4SAFETY

O Guia pra modelagem em iStar4Safety apresenta as diretrizes para modelagem de Sistemas Críticos de Segurança utilizando a extensão iStar4Safety com a ferramenta pistar-4Safety.

Guia para modelagem em iStar4Safety

(Para os demais materiais, acesse Material de Aula/Experimento: http://www.cin.ufpe.br/~if716/experimento.php)

Para iniciar sua modelagem em iStar na ferramenta PiStar4Safety:
-> Abra o pré-modelo já disponibilizado!
-> Siga o passo a passo (O passo 1 já foi realizado no pré-modelo!)
-> Salve seu arquivo em "Save Model" e envie o .txt para: smsr@cin.ufpe.br
-> Responda o questionário
-> Pronto!

Esse guia em 6 passos irá auxiliá-lo na modelagem dos requisitos do sistema crítico proposto. Com essa sugestão de passos, pretendemos guiá-lo da melhor forma na inserção de requisitos relacionados à segurança na modelagem de um Sistema Crítico de Segurança. O foco foi para uso na ferramenta piStar-4Safety. Considere portanto que na ferramenta os construtores inseridos possuem nomes/estereótipos em inglês.

Vamos lá!

DIRETRIZ 1: Modelar Funcionalidades de iStar padrão (parte não relacionada à segurança)

01. Leia a descrição dos requisitos do seu sistema

Modele todas as funcionalidades de seu sistema, não se preocupando ainda com a modelagem de requisitos *de* segurança.

OBS: Atente-te que, nesse momento, alguns **Objetivos** que deveriam ser **Objetivos de Segurança** podem vir à ser inseridos. Não se preocupe. No próximo passo, resolveremos essa questão.

DIRETRIZ 2: Modelar o Objetivo de Segurança

Um **objetivo de segurança** é um objetivo crítico que, caso não ocorra como descrito pode provocar um acidente.

01. **Leia a descrição de requisitos do seu sistema (a parte a ser modelada diz respeito à requisitos de segurança).**

02. **Se o objetivo de segurança já tiver sido modelado como objetivo, altere-o para ser representado como objetivo de segurança, clicando sobre "Toggle Safety".**
03. **Se o objetivo de segurança ainda não estiver no modelo, insira o objetivo de segurança desejado.**
04. **Refine o objetivo de segurança caso seja necessário em mais objetivos de segurança.**
05. **Insira, o valor da propriedade "Accident Impact Level" no "SafetyGoal". Essa propriedade é do tipo *String*. Você deve inserir o valor desejado, que pode ser um dos cinco abaixo:**
    Catastrófico (Catastrophic) (MAIOR IMPACTO)
    Muito Severo (Hazardous/Severe-Major)
    Considerável (Major)
    Menor (Minor)
    Sem efeito (No Effect). (SEM IMPACTO)

Obs1.: Somente objetivos de segurança-raiz serão refinados em perigos.
Obs2.: O nível de impacto do acidente do objetivo de segurança-pai será o maior nível de impacto dos seus objetivos de segurança-filhos

DIRETRIZ 3: Inserir todos os perigos para o objetivo de segurança modelado.

Um **perigo** à um **objetivo de segurança** é um perigo que pode fazer com que o **objetivo de segurança** não se concretize. Podendo assim causar um acidente!.

01. **Leia a descrição de requisitos do seu sistema (a parte a ser modelada diz respeito à requisitos de segurança).**
02. **Verifique quais são os perigos que obstruem o objetivo de segurança.**
03. **Insira o construto de perigo e altere seu nome.**
04. **Ligue o perigo ao objetivo de segurança que ele obstrui através do link obstrui. perigos>objetivo de segurança.**
05. **OBSERVAÇÃO: Somente perigos-RAIZ se ligam à objetivo de segurança.**

DIRETRIZ 4 : Identifique todas as causas para cada perigo identificado.

Uma causa de um perigo é algo que causa o perigo! Logo, causas nada mais são que **perigos.**

01. **Leia a descrição de requisitos do seu sistema (a parte a ser modelada diz respeito à requisitos de segurança).**
02. **Verifique quais são as causas para cada perigo identificado no passo anterior.**
03. **Insira o construto e altere seu nome.**

04. **Ligue o perigo-filho ao perigo-pai através dos links E ou OU (Começando do filho).**
05. **Refine, caso necessário, os perigos-filhos em novos perigos, ou seja, suas causas.**

DIRETRIZ 5: Definir a estratégia de mitigação para cada perigo-folha.

Estratégias de segurança são ações concretas para mitigar **perigos.** As estratégias são representadas por **tarefas de segurança** e/ou **recursos de segurança**.

01. **Leia a descrição de requisitos do seu sistema (a parte a ser modelada diz respeito à requisitos *d*e segurança).**
02. **Verifique quais são as estratégias para cada perigo-FOLHA modelado.**
03. **Insira o construto desejado: "Task" ou "Resource" e altere seu nome.**
04. **Altere esse construto para ele se tornar do tipo de segurança, clicando sobre "Toggle Safety".**
05. **Ligue a estratégia de segurança ao perigoque ela mitiga, através dos links E ou OU (indo da estratégia para o perigo).**

DIRETRIZ 6 : Associe a estratégia de mitigação ao ator que a realiza.

Como estratégias de segurança são ações concretas, as mesmas devem ser associadas ao ator que as concretiza. Pode haver o caso em que as estratégias são realizadas pelo próprio ator. Caso sejam realizadas por outro ator, devem ser ligadas por dependência.

01. **Leia a descrição de requisitos do seu sistema (a parte a ser modelada diz respeito à requisitos de segurança).**
02. **Identifique quais são os atores responsáveis pela realização de cada tarefa de segurança ou pela disponibilização de cada recurso de segurança.**
03. **Modele essa dependência (Lembre-se que construtos de iStar4Safety não são modelados fora dos limites do ator, ou seja, não podem ser elementos *dependum*).**

REPITA O PROCESSO DE 1 À 6 ATÉ MODELAR TODOS OS ELEMENTOS DE SEGURANÇA!

Confira a completude de seu modelo!

Marque aqui se todas as regras de completude foram atendidas:

1. ☐ Todos os **objetivos de segurança** tem uma propriedade de **nível de impacto do acidente** com um valor inserido.
2. ☐ Todos os **objetivos de segurança** tem **um** ou **mais perigos** associados ou **objetivos de segurança** que nesse caso são seus refinamentos.
3. ☐ Todos os **perigos-raiz** estão ligados à **um** ou **mais objetivos de segurança -folha** pelo link **obstrui.**
4. ☐ Todos os **perigos-folha** estão ligados à **uma** ou **mais estratégias de segurança.**
5. ☐ Todas as **estratégias de segurança** estão associadas ao ator que as realizam, menos no caso de serem realizadas pelo próprio ator em que estão definidas.
6. ☐ Atente-se que: caso um **recurso** seja **um recurso de segurança,** ele será em **todo o** modelo desse tipo, menos fora dos **limites** dos atores em que não tem elementos iStar4Safety**.**

B.3 QUESTIONÁRIO SOBRE USO DA EXTENSÃO ISTAR4SAFETY

O questionário aplicado para avaliar a extensão iStar4Safety foi desenvolvido na ferramenta *Google forms* e disponibilizado para ser respondido após a realização da modelagem por parte dos participantes do survey.

Questionário sobre uso da extensão iStar4Safety

Esse questionário foi desenvolvido para avaliar o entendimento e uso da extensão iStar4Safety pelos participantes que a utilizaram.

Todas as informações providas por você são confidenciais. Nós estamos interessados em obter feedback sobre a extensão iStar4Safety para a modelagem de requisitos iniciais de sistemas críticos, mantendo o anonimato dos participantes.

Os participantes foram escolhidos devido à estarem cursando disciplinas de Especificação de Requisitos e Validação de Sistemas, e terem participado de aulas tanto de safety como da linguagem iStar 2.0.

Esse questionário conta com 2 (duas) seções. A primeira seção é um pré-questionário, e a seção 2 possui questões relacionadas a extensão. Após responder a segunda seção, você deverá "Submeter" as respostas clicando no botão de mesmo nome.

O tempo estimado do questionário completo é de 20min:
(1) - 5 (cinco) minutos para a seção de pré-questionário
(2) - 15 (quinze) minutos para a seção de análise da extensão iStar4Safety

Agradecemos a sua participação!

S. Moniky Ribeiro e Jaelson Castro

* Required

1. **Email address** *

Pré-questionário - Extensão iStar4Safety

Essa seção compreende um pré-questionário para captar informações básicas sobre os entrevistados e relevantes à nossa pesquisa.

O tempo estimado de resposta é de 5 minutos.

2. **1 - Qual a sua idade?** *

3. **2 - Qual o seu sexo?** *
   Mark only one oval.
   - ◯ Feminino
   - ◯ Masculino
   - ◯ Prefiro não dizer

4. **3 - Qual o seu curso?** *
   Mark only one oval.
   - ◯ Ciência da Computação
   - ◯ Engenharia da computação
   - ◯ Sistemas de Informação

5. **4 - Em qual período você está?** *

6. **5 - Você tem alguma experiência profissional em modelagem de sistemas? Se sim, quanto tempo e utilizando qual linguagem de modelagem?** *

7. **6 - Você já utilizou outra linguagem de modelagem, além de iStar? Se sim, qual? (ex. UML, Tropos, Kaos)** *

8. **7 - Qual é o seu tempo de experiência com engenharia de requisitos?** *
   Mark only one oval.
   - ◯ Nenhuma
   - ◯ De um à seis meses
   - ◯ De seis meses à um ano
   - ◯ De um ano à três anos
   - ◯ Mais do que três anos
   - ◯ Other: _____

9. **8 - Como você definiria seu nível de conhecimento em engenharia de requisitos?** *
   Mark only one oval.
   - ◯ Nenhum
   - ◯ Baixo
   - ◯ Médio
   - ◯ Alto
   - ◯ Muito alto

10. **9 - Como você definiria seu nível de conhecimento em iStar?** *
    Mark only one oval.
    - ◯ Nenhum
    - ◯ Baixo
    - ◯ Médio
    - ◯ Alto
    - ◯ Muito alto

11. **10 - Como você definiria seu nível de conhecimento em Safety?** *
Mark only one oval.

- ◯ Nenhum
- ◯ Baixo
- ◯ Médio
- ◯ Alto
- ◯ Muito alto

Análise da extensão iStar4Safety

Nessa seção, objetivamos avaliar o nível de entendimento e facilidade de uso da EXTENSÃO iStar4Safety para modelagem de requisitos iniciais de sistemas críticos.

O tempo estimado de resposta é de 15 minutos.

Obs.: As respostas na escala likert correspondem à:

(1) Discordo Totalmente
(2) Discordo Parcialmente
(3) Indiferente
(4) Concordo Parcialmente
(5) Concordo Totalmente

12. **1 - É necessária uma proposta de extensão da linguagem iStar para modelar sistemas críticos na fase inicial de requisitos.** *
Mark only one oval.

	1	2	3	4	5	
Discordo Totalmente	◯	◯	◯	◯	◯	Concordo Totalmente

13. **2 - Qual a sua opinião sobre a extensão iStar4Safety?** *

_____

_____

_____

_____

_____

14. **3 - Qual o nível de dificuldade em entender a extensão iStar4Safety?** *
Mark only one oval.

- ◯ Muito fácil
- ◯ Fácil
- ◯ Médio
- ◯ Difícil
- ◯ Muito difícil

15. **4 - Imagine um cenário (ex. bomba de insulina) em que você NÃO utilizaria a extensão iStar4Safety para modelar requisitos de safety, mas sim iStar padrão. A modelagem de requisitos de safety seria mais fácil?** *
Mark only one oval.

- ◯ Sim
- ◯ Não
- ◯ Talvez.

16. **5 - Imagine o cenário (ex. bomba de insulina)em que você NÃO utilizaria a extensão iStar4Safety para modelar requisitos de safety, mas sim iStar padrão. O tempo para modelagem seria maior?** *
Mark only one oval.

- ◯ Sim
- ◯ Não
- ◯ Talvez.

17. **6 - Qual a sua percepção sobre o guia disponibilizado para utilizar a extensão iStar4Safety? Você identificou alterações necessárias (Se sim, descreve-las abaixo)?** *

_____

_____

_____

_____

_____

18. **7 - O guia proposto para modelar requisitos safety iniciais de sistemas críticos com iStar4Safety foi satisfatório.** *
Mark only one oval.

	1	2	3	4	5	
Discordo Totalmente	◯	◯	◯	◯	◯	Concordo Totalmente

19. **8 - O uso de iStar4Safety contribuiu para a modelagem de hazards, suas causas e respectivas estratégias de solução.** *
Mark only one oval.

	1	2	3	4	5	
Discordo Totalmente	◯	◯	◯	◯	◯	Concordo Totalmente

20. **9 - Os novos construtores criados (SafetyGoal, hazard, Safetytask, SafetyResource e o link Obstructs)foram suficientes para modelar os requisitos core iniciais de um sistema crítico (hazard, cause of hazard, environmental conditions e requisitos funcionais de safety).** *
Mark only one oval.

	1	2	3	4	5	
Discordo Totalmente	◯	◯	◯	◯	◯	Concordo Totalmente

21. **10 - Qual o valor da escala abaixo representa o nível que iStar4Safety contribuiu para modelar os SafetyGoals encontrados?** *
Mark only one oval.

	0	1	2	3	4	5	
Nenhuma	◯	◯	◯	◯	◯	◯	Total

22. **11 - Qual o valor da escala abaixo representa o nível que iStar4Safety contribuiu para modelar os Hazards encontrados?** *
Mark only one oval.

	0	1	2	3	4	5	
Nenhuma	◯	◯	◯	◯	◯	◯	Total

23. **12 - Qual o valor da escala abaixo representa o nível que iStar4Safety contribuiu para modelar as SafetyStretegies (Safetytasks e/ou SafetyResources) encontradas?** *
Mark only one oval.

	0	1	2	3	4	5	
Nenhuma	◯	◯	◯	◯	◯	◯	Total

24. **13 - Qual a sua opinião sobre os construtores da extensão(SafetyGoal, Hazard, SafetyTask, SafetyResource e link Obstructs)?** *

25. **14 - Quais os conceitos mais fáceis e os mais difíceis de serem modelados (SafetyGoal, Hazard, SafetyTask, SafetyResource e link Obstructs)? Por quê?** *

26. **15 - Eu usaria iStar4Safety para modelar os requisitos iniciais do meu sistema crítico.** *
Mark only one oval.

- ◯ Sim
- ◯ Não
- ◯ Talvez

27. **16 - O uso de iStar4Safety não contribuiu para a modelagem de hazards, suas causas e respectivas estratégias de solução.** *
Mark only one oval.

	1	2	3	4	5	
Discordo Totalmente	◯	◯	◯	◯	◯	Concordo Totalmente

28. **17 - Os construtos utilizados para a modelagem de requisitos relacionados à safety foram graficamente representativos.** *
Mark only one oval.

	1	2	3	4	5	
Discordo Totalmente	◯	◯	◯	◯	◯	Concordo Totalmente

29. **18 - Você visualiza a possibilidade de modelar os requisitos iniciais de safety para um sistema crítico somente com a linguagem iStar 2.0 padrão? Se sim, como seria?** *

30. **19 - Engenheiros de requisitos podem se beneficiar ao utilizar iStar4Safety para modelar os requisitos iniciais de um sistema crítico?** *
Mark only one oval.

- ◯ Sim
- ◯ Não
- ◯ Talvez

31. **20 - Para qual nível de profissionais você recomendaria iStar4Safety?** *
Mark only one oval.

- ◯ Para engenheiros de software com experiência em safety.
- ◯ Para engenheiros de software sem experiência em safety.
- ◯ Para ambos.
- ◯ Para nenhum.
- ◯ Outros.

32. **21 - Quais são os pontos fortes da extensão iStar4Safety?** *

33. **22 - Quais são os pontos fracos da extensão iStar4Safety?** *

34. **23 - Você tem alguma proposta de mudança, melhoria ou comentários relacionados à extensão iStar4Safety?** *

35. **24 - Você tem alguma proposta de mudança, melhoria ou comentários relacionados à ferramenta PiStar4Safety (em relação à Safety)?** *

Anexar modelo SR da bomba de infusão de insulina

36. **Adicione aqui o modelo SR da bomba de insulina: Adicione o ".txt" e a imagem ".png".** *
    Files submitted:

B.4 AULA GERAL PARA PREPARAÇÃO PARA O EXPERIMENTO

O material condensa as aulas dadas para aplicação do *survey*.

Extensão da linguagem iStar para modelagem de sistemas críticos - iStar4Safety

- Moniky Ribeiro (smsr@cin) -

Requirements Engineering Lab
http://cin.ufpe.br/~ler
Advisor: Prof. Jaelson Freire de Castro

Outline

- Introdução
- Conceitos para modelagem de um PSA.
- Visão Geral de iStar4Safety
- Relembrando iStar 2.0
- iStar 2.0+Istar4Safety
- Passo a passo para modelagem com iStar4Safety
- Ferramenta Pistar + iStar4Safety
- Bomba de infusão de insulina

Introdução

- Sistemas críticos - Safety-Critical - são aqueles compostos de um conjunto de **hardware**, **software**, **processos**, **dados** e **pessoas**, cuja falha pode resultar em **acidentes** que podem causar **danos ao ambiente**, **perda financeira**, **injúrias** e até **mortes**.

Exemplos: Bomba de infusão de insulina, elevador, sistema robótico, sistema de tráfego aéreo.

Conceitos para a modelagem de um PSA

- **PHA(Preliminary Hazard Analysis) - Análise Preliminar de Perigos:**

  Essa análise acontece no início do ciclo de vida do software e irá identificar os perigos de modo mais geral e alto nível do sistema, além de identificar funções críticas.

- SHA(System Hazard Analysis) - Análise de Perigos do Sistema.
- SSHA(Subsystem Hazard Analysis) - Análise de Perigos do Subsistema.
- OSHA(Operating and Support Hazard Analysis) - Análise de Perigos de Operação e Suporte.

Conceitos para a modelagem de um PSA

Os 5 primeiros conceitos são considerados CORE.

- Alguns conceitos foram propostos por Vilela et.al (2017) para modelar o PSA de um sistema *safety-critical*. A saber:

1. **Modeling of Accident**
2. **Modeling of Hazard**
3. **Modeling of Cause of hazard**
4. **Modeling of Environmental condition**
5. **Modeling of Functional Safety Requirements**
6. Representation of constraint
7. Representation of obstacle
8. Representation of pre and post condition

Conceitos para a modelagem de um PSA

- Continuando lista de conceitos:

9. Relationships between safety elements
10. Ability to specify how a particular event affects system safety
11. Criticality level of safety-critical elements
12. Model and resoning of safety strategies.
13. **Modeling of resources**
14. **Modeling of accident impact level**
15. Support of textual description

Visão geral de iStar4Safety

- Após a nossa análise, chegamos à conclusão de que seria necessário e aplicável a modelagem de conceitos CORE e mais alguns outros, através do uso de 4 novos construtos e um novo link, sendo os construtos especializações de construtos já criados. São eles:
    1. SafetyGoal
    2. Hazard
    3. SafetyTask
    4. SafetyResource
    5. Link Obstructs

Visão geral de iStar4Safety

- Sendo assim, as seguintes características podem ser modeladas através de iStar+iStar4Safety:

1. **Modeling of Accident:** O **accident** é a **consequência** de um **hazard**. Logo, subentende-se o **accident** quando modelamos o **hazard**. **A modelagem deve ser feita de forma que a negação de um *safetyGoal* leve à um accident. Ex.: SafetyGoal:** Porta do elevador não abrir quando não permitido. **Accident:** Usuário sofre uma queda fatal.

Visão geral de iStar4Safety

2. **Modeling of Hazards: É representado pelo construto HAZARD.**

    O **hazard** é a um **obstacle** à realização de um **safetyGoal.** de um **hazard**. **Ex.: SafetyGoal:** Porta do elevador não abrir quando não permitido. **Hazard:** Sistema executa de forma errada e abre porta fora do local correto.

3. **Modeling of Cause of hazards: É representado pelo construto HAZARD.**

    Cause of hazards são causas que levam o hazard à acontecer. Logo os filhos no refinamento do hazard-pai compreendem suas causas.

Visão geral de iStar4Safety

4. **Modeling of Environmental Conditions: É representado pelo construto HAZARD.**

    Condições ambientais são causas do tipo ambientais que causam/contribuem para o *hazard* acontecer. Logo,. São também representadas em iStar4safety como hazards.

5. **Modeling of Functional Safety Requirement: É representado pelos construtos SAFETYTASK e SAFETYRESOURCE.**

    Requisito funcionais de segurança irão, em conjunto ou não, formarem as Safety Strategies (Conceito 13). Através deles, iremos representar as formas de mitigar o hazard.

Visão geral de iStar4Safety

10. **Ability to specify how a particular event affects system safety: É representado através da quality Safety mais links de contribuição .**

    A forma com que um evento afeta a Safety de um sistema pode ser representado por construtos nativos de iStar 2.0.

13. **Modeling of Resources: É representado pelos construtos RESOURCE e SAFETYRESOURCE.**

    Recursos críticos podem ser representadados pela especialização de Resources, que é SAFETYRESOURCE.

Visão geral de iStar4Safety

14. **Accident impact level. É representado através de uma propriedade "nível de impacto" em um safetyGoal.**

    O nível de impacto de um *accident* poderá ser inserido como propriedade de um *safetyGoal*.

Relembrando iStar 2.0 - Visão Geral

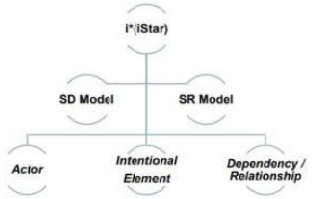


Figura 1 - Overview de iStar. Fonte: Dalpiaz, Franch, Horkoff, 2016.

Relembrando iStar 2.0 - Modelo SR

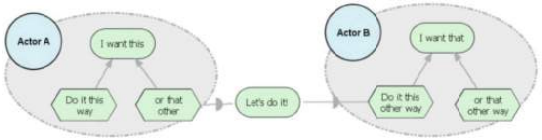


Figura 2 - Relação entre atores no modelo SR em iStar. Fonte: [13]

Relembrando iStar 2.0 - Dependência

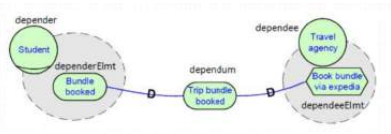


Figura 3 -Relação de dependência em iStar. Fonte[13]

Relembrando iStar 2.0 - Refinamentos

Figura 4 -Relação de dependência em iStar. Fonte: [13]

Relembrando iStar 2.0 - Needed-By

Figura 5 - Relação Needed-By, de Resource para Task em iStar. Fonte: [13]

Relembrando iStar 2.0 - Links de contribuição

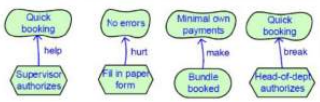


Figura 6 - Links de contribuição em iStar. Fonte: [13]

Relembrando iStar 2.0 - Link de qualificação

Figura 7 - Qualification Link em iStar. Fonte: [13]

iStar 2.0 + iStar4Safety

A extensão iStar4Safety não adicionou nenhum novo ator!

- Atores
  - **Ator**: Uma entidade autônoma, ativa.
  - **Agente** : Um ator com manifestações concretas, físicas
  - **Papel**: Uma caracterização abstrata do comportamento do ator social

Figura 8 - Um ator, um agente e um papel. Própria autora (Pistar), 2018

iStar 2.0 + iStar4Safety

- Elementos Intencionais
  - **Goal/Task/Quality e Resource**

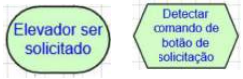


Figura 9 - Um Goal e uma Task. Própria autora(Visio). 2018

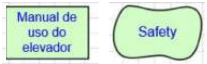


Figura 10 - Um Resource e uma Quality. Própria autora(Visio). 2018

SAFETYGOAL

iStar 2.0 + **iStar4Safety**

- Elementos Intencionais - **iStar4Safety**
  - **SafetyGoal**: Um SafetyGoal é um Goal Crítico, ou seja, pode contribuir para a ocorrência de acidentes, caso algum *hazard* associado à ele aconteça

Figura 11 - Um exemplo de SafetyGoal para a modelagem de um elevadorTask. Própria autora(Visio). 2018

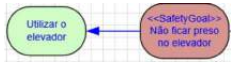


Figura 12 - Um exemplo de associação entre Goal e SafetyGoal. Própria autora(Visio). 2018

iStar 2.0 + **iStar4Safety**

- Elementos Intencionais - **iStar4Safety**
  - **SafetyGoal**: Cada **SafetyGoal** pode ser **obstruído** por de **1 à N Hazards**

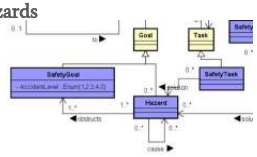


Figura 13 - Parte do metamodelo - relação SafetyGoal e Hazard. Própria autora(Astah), 2018

HAZARD

iStar 2.0 + iStar4Safety

- Elementos Intencionais - **iStar4Safety**
  - **Hazard**: Um hazard é um obstáculo para que o SafetyGoal se concretize. Ou seja, caso o hazard aconteça, um acidente pode acontecer.

Figura 14 - Um exemplo de Hazard para a modelagem de um elevador, Própria autora(Visio), 2018

Figura 15 - Um exemplo de Hazard obstruindo um SaftyGoal. Própria autora(Visio). 2018

iStar 2.0 + iStar4Safety

- Elementos Intencionais - **iStar4Safety**
  - Cada **Hazard** pode **obstruir** de **1 à N SafetyGoals**
  - **Cada Hazard** pode ser refinado por de **0 à N hazards**-filhos **(causas do hazard)**
  - **Os Hazards** folha podem possuir de **1 a N SafetyStrategies (SafetyTasks** e/ou **SafetyResources)**

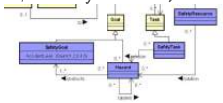
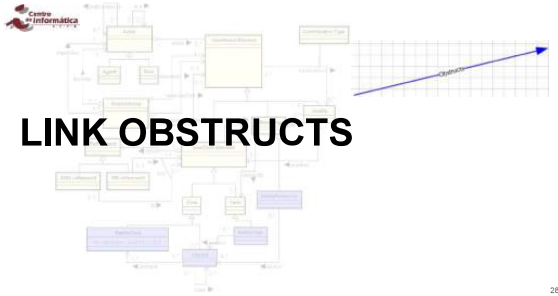


Figura 16 - Parte do metamodelo - relacionamentos com Hazard., Própria autora(Astah), 2018

LINK OBSTRUCTS



iStar 2.0 + iStar4Safety

- Elementos Intencionais - **iStar4Safety**
  - **Obstacle (LINK)**: SafetyGoal e Hazard são ligados através do link **OBSTRUCTS.** Ou seja, um hazard obstrui que um objetivo se concretize de forma *safety*, podendo levar à um acidente.

Figura 17 - Um exemplo do uso do link Obstructs conectando SafetyGoalk e Hazard.. Própria autora (Visio). 2018

iStar 2.0 + iStar4Safety

- Elementos Intencionais - **iStar4Safety**
  - Links **Obstructs** podem ser usados partindo de **hazards** para **safetygoal.**

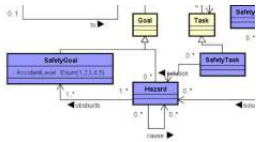


Figura 18 - Parte do metamodelo - relação SafetyGoal e Hazard. Própria autora(Astah). 2018

HAZARDS
(CAUSE OF HAZARDS)

iStar 2.0 + iStar4Safety

- Elementos Intencionais - **iStar4Safety**
  - **Hazard**: Outra questão importante é que hazards podem ter **causas**.
  - Causas de Hazards, são hazards-filhos e podem ser refinados em AND ou OR.

iStar 2.0 + iStar4Safety

- Elementos Intencionais - **iStar4Safety - HAZARD**
  - **Causa**-> Perigo procedural: Relacionado ao processo e como essas ações usuais ou oficiais de realizar atividades pode causar perigos.

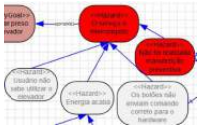


Figura 19 - Um exemplo de causa: perigo procedural para Hazard. Própria autora (Visio), 2018

iStar 2.0 + iStar4Safety

- Elementos Intencionais - **iStar4Safety - HAZARD**
  - **Causa**-> Perigo de Interface: Relacionado à interface entre os componentes do sistema. (Software, Hardware, Pessoas, Processos, dados)

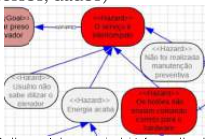


Figura 20 - Um exemplo de causa: perigo de interface para Hazard. Própria autora (Visio), 2018

iStar 2.0 + iStar4Safety

- Elementos Intencionais - **iStar4Safety - HAZARD**
  - **Causa**-> Fator Humano: Ações humanas podem causar *hazards* e consequentemente acidentes.

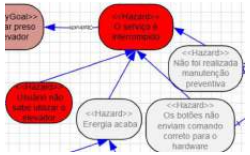


Figura 21 - Um exemplo de causa: fator humano para Hazard. Própria autora (Visio), 2018

iStar 2.0 + iStar4Safety

- Elementos Intencionais - **iStar4Safety - HAZARD**
  - **Causa**-> Perigo Ambiental: São aquelas relacionadas à perigos no ambiente que circunda o sistema e que são uma ameaça ao mesmo, podendo interferir em seu funcionamento.

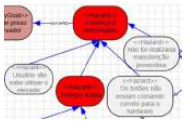


Figura 22 - Um exemplo de causa: Perigo ambiental para Hazard. Própria autora (Visio), 2018

iStar 2.0 + iStar4Safety

- Elementos Intencionais - **iStar4Safety - HAZARD**
  - **Causa**-> Causa do Sistema: Podem ser falhas ou falta de comportamento à nível do sistema.

  Ex.: O sistema não executa procedimento de acionamento de energia

iStar 2.0 + iStar4Safety

- Elementos Intencionais - **iStar4Safety**
  - Cada **Hazards** folha pode possuir de **1 a N SafetyStrategies (SafetyTasks** e/ou **SafetyResources)**

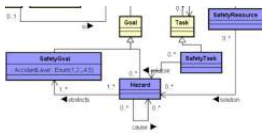


Figura 23 - Parte do metamodelo - relacionamentos com Hazard, Própria autora(Astah), 2018

SAFETY STRATEGIES - SAFETYTASKS & SAFETYRESOURCES

iStar 2.0 + iStar4Safety

- Elementos Intencionais - **iStar4Safety**
  - **Estratégias de *Safety***-> A fim de tratar os possíveis *hazards* que podem levar à não-realização de objetivos safety - OU ainda - A fim de tratar *hazards* que possam levar à acidentes, estratégias de mitigação devem ser definidas.

iStar 2.0 + iStar4Safety

- Elementos Intencionais - **iStar4Safety**
  - Estratégias de *Safety*-> Tais estratégias podem mitigar *hazards* de quatro formas:

1 - Eliminar o *Hazard (MAIS SAFE)*

2 - Reduzir o *Hazard*

3- Controlar o *Hazard*

4- Minimização de danos do *Hazard (MENOS SAFE)*

iStar 2.0 + iStar4Safety

- Elementos Intencionais - **iStar4Safety - Safety Strategies**
  - Os construtos Task e resource foram especializados para **SafetyTask** e **SafetyResource**.
  - A estratégia de Safety pode ser um desses construtos ou uma **união** deles.
  - Todo **hazard folha** deve possuir ao menos uma **estratégia safety**.

SAFETY STRATEGIES
- SAFETYTASKS -

iStar 2.0 + iStar4Safety

- Elementos Intencionais - **iStar4Safety - Safety Strategies**
  - **Safety Strategy->**SafetyTask: Uma SafetyTask representa ações *safety* que um ator quer que sejam executadas à fim de mitigar um *hazard.*

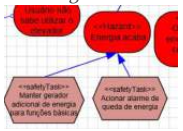


Figura 24 - Um exemplo de safetyTask: Estratégias para o perigo da queda de energia. Própria autora (Visio), 2018

SAFETY STRATEGIES
- SAFETYRESOURCE

iStar 2.0 + iStar4Safety

- Elementos Intencionais - **iStar4Safety - Safety Strategies**
  - **Safety Strategy->**SafetyResource: Um SafetyResource é uma entidade física ou informacional que um ator requer à fim de realizar uma tarefa safety. Um safetyResource pode ainda ser a única estratégia para mitigar o *hazard.*

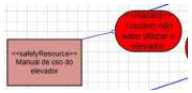


Figura 25 - Um exemplo de safetyResource: Estratégias para o perigo da falta de conhecimento do usuário. Própria autora (Visio), 2018

iStar 2.0 + iStar4Safety

- Elementos Intencionais - **iStar4Safety - Safety Strategies**
  - **Safety Strategy->**SafetyTask+SafetyResource: Um **SafetyResource** se liga por ***Needed-By*** às **Safetytasks**

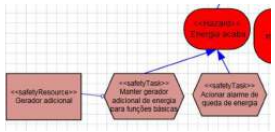


Figura 26 - Um exemplo de safetyResource + SafetyTask. Própria autora (Visio), 2018

- Accident impact level -

iStar 2.0 + iStar4Safety

- Elementos Intencionais - iStar4Safety
  - **Accident Impact Level:** Indicado como propriedade do SafetyGoal:

1 - Catastrophic (MAIOR IMPACTO)
2 - Hazardous/Severe-Major
3 - Major
4 -Minor
5 - No Effect. (SEM IMPACTO)

iStar 2.0 + iStar4Safety

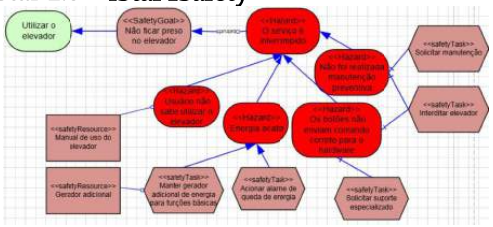


Figura 27 - Um exemplo do refinamento do SafetyGoal "Não ficar preso no elevador". Própria autora (Visio), 2018

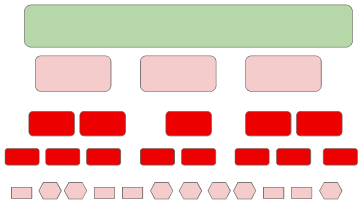
iStar 2.0 + iStar4Safety

Metamodelo da extensão iStar4Safety

(em amarelo, a versão original iStar 2.0
Em roxo a extensão IStar4Safety)

Figura 28 - Metamodelo da extensão iStar4Safety. Própria autora(Astah), 2018

iStar 2.0 + iStar4Safety - Visão em camadas



- iStar padrão
- Safety Goals
- Hazards
- Hazards (Cause of Hazards)
- Safety Strategies (Goal task e Goal resource)

Bomba de infusão de insulina
- Visão Geral -

Bomba de infusão de insulina

- Uma bomba de infusão de insulina (pump)

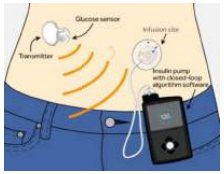


Figura 29 - Exemplo de bomba de infusão de insulina. Retirado de [2]

Bomba de infusão de insulina

- “A bomba de insulina é um dispositivo mecânico com comando eletrônico, do tamanho de um pager, pesando cerca de 80 a 100 g. Colocada externamente ao corpo, presa na cintura, pendurada por dentro da roupa ou no pescoço, a bomba de infusão deve ser usada ao longo das 24 horas do dia”. [9]

Bomba de infusão de insulina

- A bomba possibilita de forma geral 2 tipos de infusão:
  - BASAL: Aplicação que o paciente deve receber, de forma contínua, durante as 24 horas do dia.
  - BOLUS: Dose extra de insulina. Essas aplicações acontecem após alguma refeição ou para correção da glicemia.

Bomba de infusão de insulina

- Uma bomba de infusão de insulina (pump).
  - Vídeo de usuário retirado do youtube [6]: https://www.youtube.com/watch?v=vsEX1zDh5BM

Figura 30 - Um exemplo de peças da bomba de infusão de insulina. Retirado de [1]

Bomba de infusão de insulina

- Partes de uma bomba de infusão de insulina

Figura 31 - Conjunto de infusão: Cateter e seringa. Fonte [2]

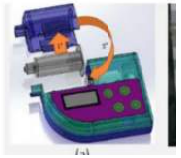


Figura 32 -Overview de um protótipo Fonte [8]

Bomba de infusão de insulina

- Os maiores perigos no uso da bomba são doses erradas de insulina aplicada, a saber:
  - OVERDOSE: O paciente recebe uma dosagem maior de insulina que o correto. -> Hipoglicemia
  - UNDERDOSE: O paciente recebe uma dosagem menor de insulina que o correto ->Hiperglicemia

Passo-a-passo para modelagem de sistemas críticos utilizando iStar4Safety.

PiStar: iStar 2.0 + **iStar4Safety**

- Apresentamos um guia propondo a melhor forma de modelar requisitos *safety* usando a extensão iStar4Safety.
- Os seguintes passos serão necessários e explicados em detalhes nos próximos slides:
  - 1º Passo - Modelar funcionalidades de iStar padrão (parte não-safety).
  - 2º Passo - Modelar o SafetyGoal.
  - 3º Passo - Inserir todos os hazards para o SafetyGoal modelado.
  - 4º Passo - Identifique todas as causas para cada hazard identificado.
  - 5º Passo - Definir a estratégia de mitigação pra cada hazard-folha.
  - 6º Passo - Associe a estratégia de mitigação ao ator que a realiza.

PiStar: iStar 2.0 + **iStar4Safety**

- **Passo-a-passo**

1º PASSO - Modelar funcionalidades de iStar padrão (parte não-safety).

- Ou seja, **não** se preocupe em modelar **ainda** os *hazards, suas causas e estratégias de mitigação.*
- **OBS**: Atente-te que nesse momento alguns Goals, que deveriam ser SafetyGoals podem vir à ser inseridos. Não se preocupe. No próximo passo, resolveremos essa questão.

PiStar: iStar 2.0 + **iStar4Safety**

- **Passo-a-passo**

2º PASSO - Modelar o **SafetyGoal**.

Para identificar um safetyGoal, leia a descrição dos requisitos *safety* do sistema. O safetyGoal pode ser refinado por outro SafetyGoal

- Um **SafetyGoal** é um objetivo crítico que, caso não ocorra como descrito pode provocar um accident.
- Se o **SafetyGoal** já tiver sido modelado como ***goal***, altere-o para **safetyGoal**, clicando sobre "Toggle Safety".
- Caso necessário refina o **safetyGoal** em mais **safety goals.**

PiStar: iStar 2.0 + **iStar4Safety**

- **Passo-a-passo**

Ainda 2º PASSO - Modelar o SafetyGoal.

- Insira, em propriedade do **SafetyGoal**, o **Accident Impact Level** relacionado à ele. Os níveis são:

  1 - Catastrophic (MAIOR IMPACTO)
  2 - Hazardous/Severe-Major
  3 - Major
  4 -Minor
  5 - No Effect. (SEM IMPACTO)

PiStar: iStar 2.0 + **iStar4Safety**

- **Passo-a-passo**

AINDA 2º PASSO - EXEMPLO

- Ex.: **SafetyGoal**: Receber radiação correta durante tratamento com máquina de raio X. Pode ser refinado em:
  - **SafetyGoal**: Não receber radiação em excesso.
- Observe que a **negação** de um **SafetyGoals** é o **accident**.
  - **Accident**: Receber radiação incorreta durante tratamento com máquina de raio X -> Pode levar à morte
  - **Accident Impact Level:** Catastrophic

PiStar: iStar 2.0 + **iStar4Safety**

- **Passo-a-passo**

3º PASSO - Inserir todos os hazards que podem obstruir o SafetyGoal modelado.

Para identificar um **hazard**, leia a descrição dos requisitos *safety* do sistema.

- Um **hazard** à um **SafetyGoal** é um perigo que pode fazer com que o **SafetyGoal** não se concretize. Podendo assim causar um accident!
- Logo um Hazard **obstructs** um safetyGoal.

PiStar: iStar 2.0 + **iStar4Safety**

- **Passo-a-passo**

AINDA 3º PASSO - EXEMPLO

- Ex.: **SafetyGoal**: Receber radiação correta durante tratamento com máquina de raio X. Pode ser refinado em:
  - **SafetyGoal**: Não receber radiação em excesso
- **Hazard**: Máquina configurada com valor maior de radiação

Hazard se liga ao SafetyGoal através do link **Obstructs**

Hazard - (obstructs) -> SafetyGoal

PiStar: iStar 2.0 + **iStar4Safety**

- **Passo-a-passo**

4º PASSO - Identifique todas as **Causas** para cada **hazard** identificado. **Causas são .. Hazards!**

Para identificar as Causas para um Hazard, leia a descrição dos requisitos *safety* do sistema.

- Uma causa de um perigo é algo que causa o hazard! Logo, causas nada mais são que **hazards** e estarão ligados por AND ou OR aos hazards que são causados, refinando-os. Um hazard pode causar de 0 à n outros hazards.

PiStar: iStar 2.0 + **iStar4Safety**

- **Passo-a-passo**

AINDA 4º PASSO - EXEMPLO

- Ex.: **Hazard**: Máquina configurada com valor maior de radiação. Pode ser refinado em:
- **Causa do Hazard->Hazard**: Usuário não treinado OU Máquina voltou às configurações de fábrica

Você pode criar quantos níveis de refinamento para hazard que desejar!

PiStar: iStar 2.0 + **iStar4Safety**

- **Passo-a-passo**

5º PASSO - Definir a estratégia de mitigação pra cada **hazard-folha.**

- Safety Strategies são modeladas através de SafetyTask e SafetyResources. Podem mitigar de **1 à N** hazards.
- Pense em: De qual ator dependo para a execução de tal estratégia?
  - Modele essa dependência. Mas lembre-se...

Elementos de iStar4Safety não fazem parte do SD, ou seja, não são dependum.

Só estão presentes internamente nos limites dos atores (SR).

PiStar: iStar 2.0 + **iStar4Safety**

- **Passo-a-passo**

AINDA 5º PASSO - EXEMPLO

- Ex.: **Hazard**:Máquina configurada com valor maior de radiação. Pode ser refinado em:
  - **Causa do Hazard->Hazard**: Usuário não treinado **OU** Máquina voltou às configurações de fábrica.
- **SafetyStrategy** para: "Usuário não treinado" será a **Safetytask**: "Receber treinamento".

PiStar: iStar 2.0 + **iStar4Safety**

- **Passo-a-passo**

6º PASSO - Associe a estratégia de mitigação ao ator que a realiza.

Como estratégias de Safety são ações concretas, as mesmas devem ser associadas ao ator que as concretiza. Pode haver o caso em que as estratégias são realizadas pelo próprio ator. Caso sejam realizadas por outro ator, devem ser ligadas por dependência.

PiStar: iStar 2.0 + **iStar4Safety**

- Passo-a-passo

6º PASSO - Associe a estratégia de mitigação ao ator que a realiza.

A SAFETYTASK: "Receber treinamento" deve ser associada ao ator **Fornecedor do Elevador.** Os construtos de iStar4Safety não podem ser *dependum*, ou seja, não aparecem fora dos limites do ator.

REPITA O PROCESSO DE 1 À 6 ATÉ MODELAR TODOS OS ELEMENTOS SAFETY!

- Ferramenta piStar + extensão iStar4Safety

PiStar: iStar 2.0 + **iStar4Safety**

- **IStar4Safety** pode ser modelada na ferramenta **PiStar**.
- É só acessar o link:
- **Hazard** e **Obstructs** são **novos construtos** no menu superior.

Figura 33 - Construto Hazard e link Obstructs em PiStar4safety

PiStar: iStar 2.0 + **iStar4Safety**

- **SafetyGoal**, **SafetyResource** e **SafetyTask** devem ser inseridos como **Goal**, **Resource** e **Task**. Altere seus nomes e clique no botão "Toggle Safety" na barra de propriedades.

PiStar: iStar 2.0 + **iStar4Safety**

- PARA **INSERIR** um SAFETYGOAL:
  - Insira um **Goal** padrão.
  - Insira o nome desejado.
  - Selecione o botão **"Toggle Safety"**, na barra inferior!
  - Pronto!

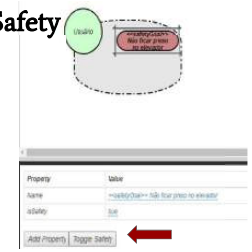


Figura 34 - Construto SafetyGoal em PiStar4Safety

PiStar: iStar 2.0 + **iStar4Sa**

- PARA alterar um GOAL para SAFETYGOAL:
  - Clique sobre o **Goal** desejado.
  - Selecione o botão "Toggle Safety", na barra inferior!
  - Pronto!

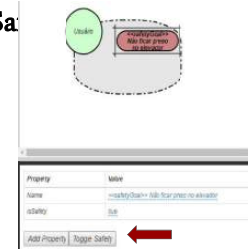


Figura 35 - Alterar Goal para SafetyGoal em PiStar4Safety

PiStar: iStar 2.0 + **iStar4Safety**

- PARA INSERIR O NÍVEL DE IMPACTO DO ACIDENTE:
  - Selecione o SafetyGoal desejado
  - Insira o nível em: "Accident Impact Level" em propriedades.
  - Insira o valor"

1 - Catastrophic
2 - Hazardous/Severe-Major
3 - Major
4 -Minor
5 - No Effect.

Figura 36 - Inserir "Accident Impact Level" em Pistar4Safety

PiStar: iStar 2.0 + **iStar4Safety**

- PARA INSERIR UM HAZARD:
  - Insira um construto HAZARD.
  - Insira o nome desejado, incluindo o estereótipo <<ha

Figura 37 - Inserir "Hazard" em Pistar4Safety

PiStar: iStar 2.0 + **iStar4Safety**

- PARA LIGAR **SAFETYGOAL** À SEUS **HAZARDS** :
  - Selecione o link "**Obstructs**"
  - Clique primeiro sobre o **Hazard**, depois sobre o **SafetyGoal**.

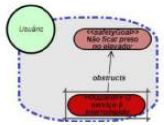


Figura 37 - Associar SafetyGoal e Hazard em Pistar4Safety

PiStar: iStar 2.0 + **iStar4Safety**

- PARA LIGAR **HAZARDS** À SEU **HAZARD-PAI** :
  - Selecione o refinamento desejado (AND/OR)
  - Clique primeiro sobre o **Hazard-filho**, depois sobre o **Hazard-Pai**.

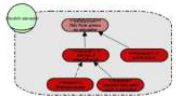


Figura 38 - Refinar Hazards em suas causas, em Pistar4Safety

PiStar: iStar 2.0 + **iStar4Safety**

- PARA INSERIR UMA SAFETYTASK:
  - Insira uma **Task** padrão.
  - Insira o nome desejado.
  - Selecione o botão "**Toggle Safety**", na barra inferior!
  - Pronto!

Figura 39 - Inserir SafetyTask, em Pistar4Safety

PiStar: iStar 2.0 + **iStar4Safety**

- PARA LIGAR **SAFETYTASK** AO **HAZARD** QUE O MESMO MITIGA :
  - Selecione o refinamento desejado (AND/OR)
  - Clique primeiro sobre a SafetyTask, depois sobre o **Hazard(folha)**.

Figura 40 - Refinar Hazard-folha em SafetyTask, em Pistar4Safety

PiStar: iStar 2.0 + **iStar4Safety**

- PARA INSERIR UM **SAFETYRESOURCE**:
  - Insira um **Resource**.
  - Insira o nome desejado.
  - Selecione o botão "**Toggle Safety**", na barra inferior!
  - Pronto!

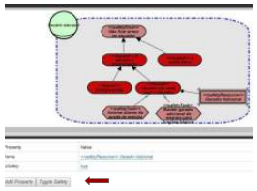


Figura 41 - Inserir SafetyResource, em Pistar4Safety

PiStar: iStar 2.0 + **iStar4Safety**

- PARA LIGAR **SAFETYRESOURCE** AO **HAZARD** QUE O MESMO MITIGA :
  - Selecione o refinamento Needed-By.
  - Clique primeiro sobre a SafetyResource, depois sobre o **Hazard**.

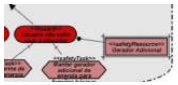


Figura 42 -Associar SafetyResource a Hazard por Needed-By em Pistar4Safety

PiStar: iStar 2.0 + **iStar4Safety**

- PARA LIGAR **SAFETYRESOURCE** A **SAFETYTASK** QUE PRECISA DELE :
  - Selecione o refinamento Needed-By.
  - Clique primeiro sobre a SafetyResource, depois sobre a SafetyTask.

Figura 43 -Associar SafetyResource a Hazard por Needed-By em Pistar4Safety

PiStar: iStar 2.0 + **iStar4Safety**

- Assim como no Pistar padrão, você tem as opções de:
  - **Salvar imagem** -> Em SVG ou PNG.
  - **Salvar Modelo** -> Salva modelo em .txt no seu computador.
  - **Ler Modelo Criado** -> Busca arquivo .txt salvo.
  - **Mudar opções do diagrama**.
  - **Limpar o diagrama** -> Limpa toda a tela.

Figura 44 -Opções adicionais em Pistar4Safety

References

1. Retirado de: http://medbox.iiab.me/modules/en-wikem/wiki/Insulin_infusion_device_complication.html. Acesso em: 19/11/2018
2. Retirado de: https://dwmdt.en.ec21.com/Syringe_Infusion_Set_for_Insulin--1180777_1180780.html. Acesso em: 19/11/2018
3. Zhang Y, Jetley R, Jones PL, Ray A. Generic safety requirements for developing safe insulin pump software. J Diabetes Sci Technol. 2011;5(6):1403-19. Published 2011 Nov 1. doi:10.1177/193229681100500612
4. Zhan, Fengjie & Wang, Xiaoyu & Huaxiao, Liu & Liu, Lei. (2015). A Kind of Safety Requirements Description Method of the Embedded Software Based on Ontology. 126-134. 10.1007/978-3-662-48634-4_9.

References

5. MARTINS, L. E. G. et al. Relatório Subprojeto Temático- 2º ano. São José dos Campos. 2013.
6. RODRIGUES, L. G. Troca de descartáveis da bomba de Insulina (Português). 2015. (09m49s). Disponível em: <www.youtube.com/watch?v=vsEX1zDh5BM8>. Acesso em: 19 nov. 2018.
7. MARTINS, L.E.G.; De Oliveira, T. A case study using a protocol to derive safety functional requirements from fault tree analysis. In Proceedings of the 22nd IEEE International Requirements Engineering Conference, Karlskrona, Sweden, 25–29 August 2014; pp. 412–419.
8. MARTINS,L. E. G. , H. de Faria, L. Vecchete, T. Cunha, T. de Oliveira, D. E. Casarini, J. A. Colucci, "Development of a low-cost insulin infusion pump: Lessons learned from an industry case", 2015 IEEE 28th International Symposium on Computer-Based Medical Systems, pp. 338-343, 2015.

References

9. MINICUCCI, W.J. Insulin pump therapy in patients with type 1 diabetes. Arq Bras Endocrinol Metabol, 52 (2008), pp. 340-348
10. Jéssyka Vilela, Jaelson Castro, Luiz Eduardo G. Martins, Tony Gorschek, and Carla Silva. 2017. Specifying Safety Requirements with GORE languages. In Proceedings of the 31st Brazilian Symposium on Software Engineering (SBES'17). ACM, New York, NY, USA, 154-163. DOI: https://doi.org/10.1145/3131151.3131175.
11. Dalpiaz, F., Franch, X., Horkoff, J.: iStar 2.0 language guide. CoRR abs/1605.07767 (2016).
12. LEVESON, N. 1995. Safeware: System Safety and Computers. ACM, New York, NY, USA.
13. Dalpiaz, F., Franch, X., Horkoff, J. iStar2.0: A Guided Tour. Disponível em: www.dropbox.com/s/4l2k4tbywb8wekk/iStar-tutorial-online.pdf?dl=0. Acessado em: 19/11/2017

References

14. Zhang Y, Jones PL, Jetley R. A hazard analysis for a generic insulin infusion pump. J Diabetes Sci Technol. 2010;4(2):263-83. Published 2010 Mar 1. doi:10.1177/193229681000400207.

Extensão da linguagem iStar para modelagem de sistemas críticos - iStar4Safety

- Moniky Ribeiro (smsr@cin) -

Requirements Engineering Lab
http://cin.ufpe.br/~ler
Advisor: Prof. Jaelson Freire de Castro

B.5 MODELO BASE DA BOMBA DE INFUSÃO DE INSULINA SEM ISTAR4SAFETY

O documento abaixo representa o modelo SR de iStar para modelagem do sistema proposto no survey aplicado. Os participantes o receberam e deveriam utilizar esse pré-modelo como base, acrescentando ao mesmo os requisitos de segurança de iStar4Safety.

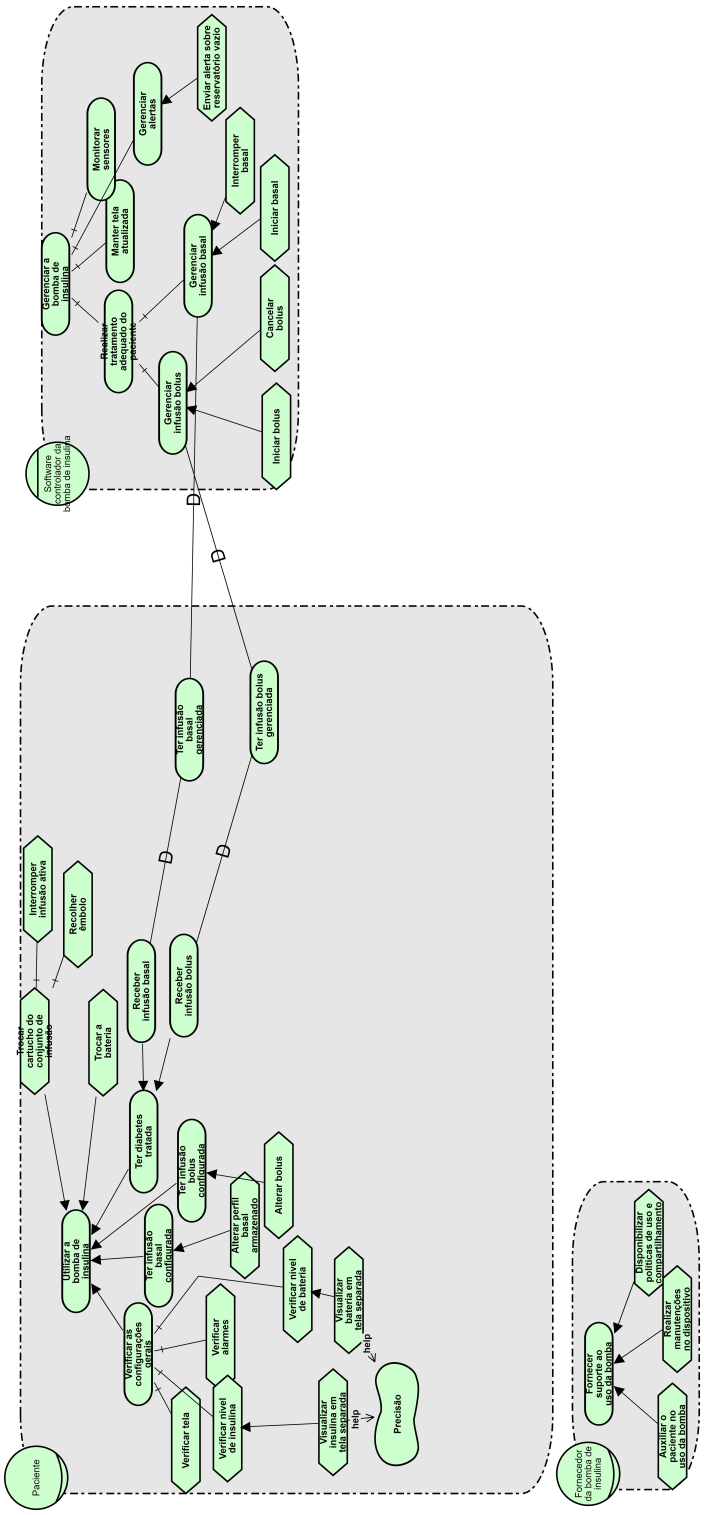


Figura 64 – Modelo SR dos requisitos gerais (sem requisitos de segurança) da Bomba de Infusão de Insulina.

Fonte: (Autora, 2018).

B.6 MODELO BASE DA BOMBA DE INFUSÃO DE INSULINA COM ISTAR4SAFETY

O documento abaixo representa o modelo de Raciocínio Estratégio (SR) de iStar mais iStar4Safety usado como base para verificação de completude dos modelos SR criados pelos participantes do survey.

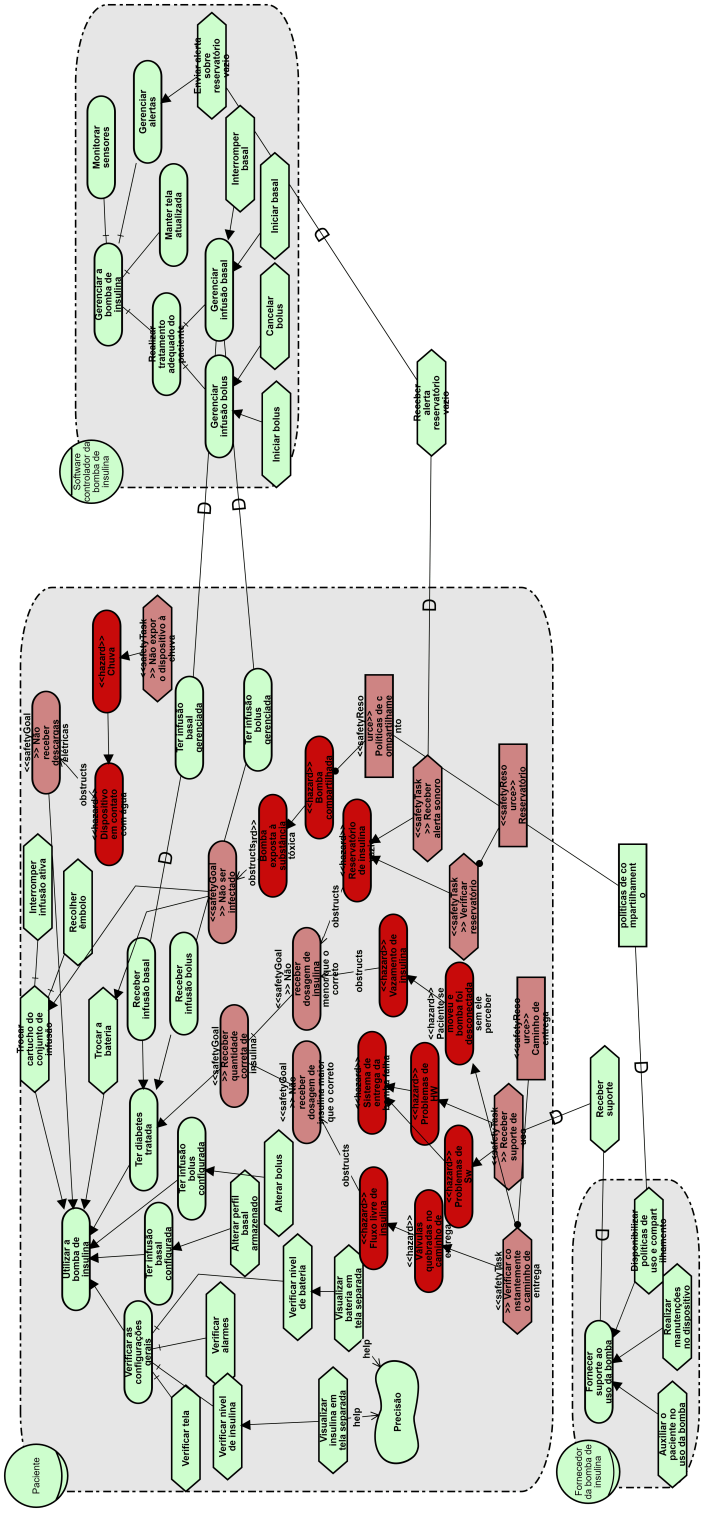


Figura 65 – Modelo SR de iStar4Safety da Bomba de Infusão de Insulina.

Fonte: (Autora, 2018).