

UNIVERSIDADE TECNOLÓGICA FEDERAL DO PARANÁ – UTFPR  
PROGRAMA DE PÓS-GRADUAÇÃO EM ENGENHARIA ELÉTRICA E  
INFORMÁTICA INDUSTRIAL

JONATHAN MORRIS SAMARA

**UM MODELO CONCEITUAL PARA CENÁRIOS DE  
ACIDENTES EM ATIVIDADES DE MANUTENÇÃO**

DISSERTAÇÃO

CURITIBA

2019

JONATHAN MORRIS SAMARA

# **UM MODELO CONCEITUAL PARA CENÁRIOS DE ACIDENTES EM ATIVIDADES DE MANUTENÇÃO**

Trabalho de Conclusão de Curso apresentado ao Departamento Acadêmico de Informática como requisito parcial para obtenção do grau de Bacharel no Curso Superior de Bacharelado em Sistemas de Informação da Universidade Tecnológica Federal do Paraná.

Orientador: Cesar Augusto Tacla

**CURITIBA**

**2019**

Dedico esse trabalho a minha esposa Franciele Saldanha de Quadros pelo seu amor e por me dar o apoio necessário para levar adiante os meus sonhos, como é o caso deste Mestrado.

Dedico, também, este trabalho a meu pai Maurice Jamil Samara, por me tornar o homem que eu sou.

Dedico, também, este trabalho a minha mãe Roseli Escarlete Almeida de Lima, que sempre me incentivou a autossuperação.

Dedico, também, este trabalho ao meu irmão David Morris Samara, por toda a felicidade e alegria que ele trouxe a minha vida.

*Dedico esse trabalho a Deus, Jeová, por ter me dado a dádiva da existência.*

*Dedico esse trabalho a minha esposa Franciele Saldanha de Quadros pelo seu amor e por me dar o apoio necessário para levar adiante os meus sonhos, como é o caso deste Mestrado.*

*Dedico, também, este trabalho a meu pai Maurice Jamil Samara, por me tornar o homem que eu sou.*

*Dedico, também, este trabalho a minha mãe Roseli Escarlete Almeida de Lima, que sempre me incentivou a autossuperação.*

*Dedico, também, este trabalho ao meu irmão David Morris Samara, por toda a felicidade e alegria que ele trouxe a minha vida.*

## Agradecimentos

Agradeço a Deus, Jeová, pela dádiva da existência e por ter me dado energia e força para concluir esse trabalho.

Agradeço a minha esposa Franciele Saldanha de Quadros, por fazer eu acreditar em mim mesmo.

Agradeço aos meus pais Maurice Jamil Samara e Roseli Escarlete Almeida de Lima, por ter me proporcionado a educação e cultura por meio de muito amor e carinho.

Agradeço ao meu irmão David Morris Samara, por ter me dado o incentivo e apoio necessário para concluir este trabalho.

Agradeço ao meu orientador Professor Dr. Cesar Augusto Tacla, por me dar a orientação, conhecimento e apoio necessário para concluir esse trabalho.

Agradeço ao Dr. Professor Gustavo Alberto Giménez Lugo, pelos conselhos e sugestões a respeito do trabalho.

Agradeço ao Dr. Professor Sebastião Ribeiro Junior, por ter me dado a oportunidade de entrar no grupo de pesquisa OneReal nos Institutos Lactec me possibilitando desenvolver esse trabalho. Eu também agradeço ao Dr. Professor Sebastião Ribeiro Junior pelas dicas valiosas a minha carreira.

Agradeço Dr. Professor Klaus de Geus, por me fornecer o suporte necessário ao desenvolvimento deste trabalho.

Agradeço a Universidade Tecnológica Federal do Paraná pelo conhecimento e pela estrutura fornecida.

Agradeço ao grupo OneReal Research Group no âmbito do projeto de P&D PD-6491-0299/2013, proposto pela Copel Geração e Transmissão S.A., sob os auspícios do Programa de P&D da Agência Nacional de Energia Elétrica (ANEEL).

Agradeço aos Institutos Lactec pela estrutura de ponta e pela Bolsa de Mestrado.

Agradeço ao Ignácio L. Del Hoyo pelo companheirismo ao longo do Mestrado.

*Quando a sabedoria entrar no seu coração e o conhecimento se tornar agradável para a sua alma, o raciocínio o guardará e o discernimento o protegerá - Provérbios 2:10-11*

## Resumo

Diversas pessoas são submetidas a algum tipo de risco ao executar atividades profissionais relacionadas à área da eletricidade, petroquímica, transportes e outras. Com base em entrevista com um Engenheiro de Manutenção, participação **in loco** em uma atividade de manutenção na área de transmissão de energia elétrica e revisão bibliográfica exploratória, esse estudo sintetiza as informações coletadas, constrói e avalia um modelo conceitual destinado à construção de sistemas computacionais de simulação de riscos e acidentes com fins diversos, tais como, treinamento de operadores ou estudo de riscos associados a uma atividade laboral. O modelo conceitual construído fundamenta-se nos conceitos de agentes, artefatos, sistemas multiagentes normas e riscos/consequências. Atividades podem ser expressas em termos de objetivos com pré-requisitos e efeitos. A preocupação central é a modelagem de riscos e de suas consequências. Deste modo, pré-requisitos de atividades que não estão presentes quando os agentes decidem executá-las levam a violações que podem causar consequências para o objetivo atual e/ou subsequentes (ex. parada) assim como para os agentes envolvidos (ex. acidente). O modelo, formalizado em Lógica de Primeira Ordem, foi implementado em PROLOG e aplicado a um estudo de caso permitindo avaliar se as inferências produzidas correspondiam ao esperado bem como compreender o estado do problema particular modelado. Além disso, esta formalização permitiu realizar uma análise comparativa (em relação a cenários de acidentes) entre quatro modelos, MOISE+, Modelo de Agentes Normativos de DASTANI, V3S e NORMMAS. Como conclusão, o modelo é uma síntese de vários modelos existentes reunindo em um só as características relevantes para tal tipo de modelagem. Também, se revelou capaz de representar diversos cenários atrelados a atividades profissionais de risco, em especial, mostrou-se capaz de produzir resultados coerentes com o esperado no estudo de caso.

**Palavras-chaves:** agentes, atividades, objetivos, normas, riscos, violação, consequências, modelo conceitual.

# Abstract

Many people are at risk when performing professional activities related to electricity, petrochemicals, transportation and others. Based on an interview with a Maintenance Engineer, on-site participation in a maintenance activity in the area of electricity transmission and exploratory literature review, this study synthesizes the information collected, builds and assesses a conceptual model for the construction of computer simulation systems. hazards and accidents for various purposes, such as operator training or study of risks associated with a labor activity. The constructed conceptual model is based on the concepts of agents, artifacts, multi-agent systems, norms and risks / consequences. Activities can be expressed in terms of goals with prerequisites and effects. The central concern is risk modeling and its consequences. Thus, prerequisites for activities that are not present at the time agents decide to perform them lead to violations that may have consequences for the current and / or subsequent goals (e.g. stopping) as well as for the agents involved (e.g. accident). The model, formalized in First Order Logic, was implemented in PROLOG and applied to a case study allowing to evaluate if the inferences produced corresponded to the expected ones as well as to assess the state of the problem modeled. This representation made it possible to perform a comparative analysis (in relation to accident scenarios modeling) between four models, MOISE +, DASTANI Normative Agents Model, V3S and NORMS. In conclusion, the model is a synthesis of several existing models bringing together in one the relevant characteristics for this type of modeling. Besides, if it was able to represent several scenarios related to risky professional activities, in particular, it was able to produce coherent results with the expected ones in the case study.

**Key-words:** agents, activities, goals, norms, risks, violation, consequences, conceptual model.

## Lista de ilustrações

Figura 1 – Árvore de objetivos definido pelo modelo MOISE+ (HÜBNER; SICHMAN; BOISSIER, 2002b) . . . . .	20
Figura 2 – Linguagem para descrever um programa de multiagentes normativos com a possibilidade de violações e sanções na notação EBNF segundo o texto (DASTANI et al., 2009). Nesta notação, <i>&lt; ident &gt;</i> é usado para denotar uma <i>string</i> e <i>&lt; int &gt;</i> inteiros. Os termos <i>&lt; b – prop &gt;</i> e <i>&lt; i – prop &gt;</i> são usados para designar dois tipos de conjuntos de proposições que são disjuntos entre si	22
Figura 3 – Um programa descrito na linguagem proposta neste estudo onde um agente representa um passageiro em uma estação de trem que pode entrar com ou sem um <i>ticket</i> na plataforma e no trem (DASTANI et al., 2009). . . . .	23
Figura 4 – Ontologia que descreve <i>Domain Model</i> no modelo V3S (BAROT et al., 2013)	27
Figura 5 – Gráfico de Bowtie do texto (CAMUS; LENNE; PLOT, 2012) . . . . .	30
Figura 6 – A estrutura geral das classes do modelo . . . . .	38
Figura 7 – Diagrama de classes do Modelo . . . . .	46
Figura 8 – Condição para definir se um dado objetivo foi atingido ou não . . . . .	55
Figura 9 – Diagrama de atividades do modelo (1) . . . . .	57
Figura 10 – Diagrama de atividades do modelo (2) . . . . .	58
Figura 11 – Diagrama de atividades do modelo (3) . . . . .	58
Figura 12 – Raciocínio 6 parte 1 . . . . .	134
Figura 13 – Raciocínio 6 parte 2 . . . . .	135



## Lista de tabelas

Tabela 1 – Construtores da linguagem <i>ACTIVITY-DL</i> (BAROT et al., 2013) . . . . .	28
Tabela 2 – As pré-condições possíveis para as atividades (BAROT et al., 2013) . . . . .	29
Tabela 3 – Os agentes que constituem uma manutenção . . . . .	73
Tabela 4 – Os papéis relevantes para a ocorrência da manutenção . . . . .	73
Tabela 5 – Relação $adoptsRole(ag_n, \rho_m)$ . . . . .	74
Tabela 6 – Definição de todos os artefatos presentes na manutenção . . . . .	74
Tabela 7 – Definição e descrição dos objetivos bem como dos respectivos pré-requisitos	75
Tabela 8 – Define as condições necessárias para que a manutenção tenha possibilidade de acontecer . . . . .	75
Tabela 9 – Descrição das entidades em função das relações . . . . .	78
Tabela 10 – Define o impacto que o erro em um relacionamento gera em outro relacio- namento por mudar a possibilidade de algo errado acontecer. . . . .	81
Tabela 11 – Objetivos que devem ser atingidos pelo agente que assumir um dada função	83
Tabela 12 – Exemplificação dos dados atrelados ao predicado $requiresCirc(goal_n, circ_m), requiresEntity(goal_n, e_j)$ serão organizados nas tabelas a seguir. . . . .	84
Tabela 13 – Especificação do predicado $requiresCirc(goal_i, circ_j)$ , do predicado $instanceOfRel(circ_n)$ e do predicado $instanceOfCond(circ_n)$ . . . . .	86
Tabela 14 – Especificação do estudo de caso atrelado ao predicado $requiresEntity(goal_i, e_j)$	88
Tabela 15 – Comparação entre o <i>MOISE+</i> e o modelo proposto neste estudo . . . . .	101
Tabela 16 – Comparação entre o <i>DASTANI+</i> e o modelo proposto neste estudo . . . . .	102
Tabela 17 – Comparação entre o <i>VS3</i> e o modelo proposto neste estudo . . . . .	104
Tabela 18 – Análise comparativa sobre a expressividade desses modelos no que tange aos objetivos deste estudo. . . . .	106

# 1 Introdução

Diversas pessoas são submetidas a algum tipo de risco na execução de atividades profissionais. Trabalhos como de eletricitas, bombeiros, área petroquímica, serviços de telecomunicações e de transporte (motoristas de ônibus e de carro) são apenas alguns em que os profissionais são expostos a condições de risco.

Acidentes nesses trabalhos ocorrem pelos mais diversos motivos. Investigar as cadeias de causalidade que resultam nessas situações pode trazer um entendimento de como lidar com essas circunstâncias. Consequentemente, isso traz um potencial para diminuir o número de acidentes. A computação é uma ciência que apresenta um grande potencial de contribuição para a resolução desse problema. Isso envolve criar representações que descrevam atividades com riscos de acidentes e que permitam realizar raciocínios sobre essas relações de causalidade.

Algo que exemplifica essa situação são as atividades praticadas por profissionais que trabalham com eletricidade em linha viva. Isso, pois esses trabalhadores realizam procedimentos de manutenção em componentes, estruturas e equipamentos elétricos energizados que demandam fluxo de potência elétrica (ex. transformadores, condutores). A particularidade nos processos de linha viva consiste no fato de que esses equipamentos e estruturas se mantêm energizados durante a ação dos profissionais. Essa situação expõe esses profissionais a perigos de acidentes com eletricidade causando mortes ou ferimentos extremamente sérios com danos permanentes à vida dos acidentados.

## 1.1 Motivação

A motivação desse estudo reside em obter uma compreensão do problema em voga por meio de um modelo conceitual concebido com base em dois fatores. O primeiro fator consiste numa averiguação de modelos computacionais que são aplicados em sistemas multiagentes (inclusive os normativos). O segundo fator consiste em averiguar um dado estudo de caso a fim de abstrair estruturas conceituais que possam ser do interesse dessa pesquisa.

## 1.2 Relevância

A computação pode contribuir de inúmeras maneiras por conceber sistemas que resolvessem de diferentes formas os problemas atrelados a acidentes de manutenção. Contudo, isso não pode ser feito sem um entendimento, em termos conceituais e computacionais, do estado deste problema. Por conta disso, a relevância desse estudo reside na apresentação de um modelo conceitual que defina em termos de classes e relacionamento uma estrutura lógico-descritiva de fatores ambientais e organizacionais que resultam em acidentes.

Esse estudo descreve a ocorrência de acidentes através de agentes, artefatos, violações e sanções. O que se define como inovador em relação aos demais estudos consiste no fato de que o esse modelo descreve a ocorrência de acidentes em termos de possibilidades em relação a uma violação. Em outros termos, isso define que, em determinados cenários, a ocorrência de uma violação em uma dada relação gera a possibilidade de ocorrer um acidente em algum outro momento futuro da manutenção.

### 1.3 Objetivos

Propor por intermédio de observações, de análises de documentos técnicos, de análises de modelos computacionais e de entrevista com profissionais da área, um modelo conceitual de cenários de ambientes de atividades, bem como os respectivos acidentes que neles podem acontecer, em que a validação ocorre por verificar se os raciocínios (para um dado estudo de caso do setor de energia elétrica) resultantes desse modelo são correspondentes com a realidade a fim de levantar um entendimento formal do problema para a comunidade acadêmica no que tange a que tipo de representação computacional é mais apropriada para determinado contexto.

#### 1.3.1 Objetivos Específicos

- Identificar os aspectos importantes que devem fazer parte da estrutura do modelo em relação aos riscos e consequências (acidentes) para os atores e atividades, que sejam relevantes na prática da atividade de manutenção, em caso de falha na operação.
- Construir um modelo conceitual que seja implementável computacionalmente e que produza as inferências que tratem aspectos consideráveis ao problema.
- Avaliar o modelo por aplicá-lo a um dado estudo de caso a fim de averiguar se os raciocínios produzidos nessa situação estão de acordo com a realidade.
- Analisar modelos computacionais em relação ao modelo conceitual desse estudo a fim de obter um levantamento formal do estado do problema.

### 1.4 Estrutura do Documento

O capítulo 2 apresenta os conceitos e estudos básicos que fundamentam essa pesquisa. O capítulo 3 descreve as etapas metodológicas adotadas pelo autor para conceber, verificar e analisar o modelo conceitual. O capítulo 4 mostra os resultados obtidos: o modelo conceitual em si (conjuntos, predicados e regras de inferência) e da aplicação deste para uma situação simples a fim de introduzir o leitor ao uso do modelo. O capítulo 5 exhibe o uso do modelo em um estudo de caso, apresentando como os conjuntos (classes sobre a ótica de orientação a objeto) são usados para especificar as instâncias dessa aplicação. Esse capítulo apresenta também os

raciocínios que podem ser feitos ao depurar a aplicação das regras. O capítulo 6 realiza uma análise do modelo conceitual deste estudo à luz de modelos computacionais bem verificados pela comunidade acadêmica. Nesse capítulo são feitas, também, discussões sobre a aplicação do modelo conceitual no estudo de caso verificando aspectos que ocorreram adequadamente bem como aspectos que devem ser analisados com mais profundidade. O capítulo 7 realiza uma verificação geral do estudo, mostra como o objetivo geral bem como os objetivos específicos foram atingido(s) e apresenta quais são os próximos passos.

## 2 Fundamentação Teórica

Este capítulo apresenta os estudos que fundamentam essa pesquisa fornecendo a base para consolidação do modelo aqui proposto, abrangendo seguintes áreas: Agentes, Artefatos, Sistemas Multiagentes, Normas (em conjunto com Violações e Sanções), Riscos e Lógica Modal.

A palavra *arcabouço* será usada em diversos momentos ao longo deste texto. O sentido desse termo deve ser interpretado como correspondente ao sentido do termo em inglês *framework*.

### 2.1 Agentes

Não existe uma definição universal para tratar o conceito de agente sendo que esse tópico se encontra em meio a debates e controvérsias. Contudo, existe um entendimento generalizado de que um comportamento *autônomo* é o cerne de noção que se tem por agente (JENNINGS; LÉSPRANCE, 2000). Apesar disso, a construção de um modelo computacional não pode ser feita sem uma definição. Assim sendo, nesse texto um agente é um sistema computacional que está situado em um dado ambiente e que apresenta comportamento autônomo (JENNINGS; LÉSPRANCE, 2000). Além disso um agente faz uso do seu comportamento autônomo com o propósito de atingir objetivos que a ele é designado (JENNINGS; LÉSPRANCE, 2000).

Como a definição de agente faz uso do conceito de ambiente, não é possível ter uma compreensão plena daquele sem considerar aspectos deste. Assim sendo, o termo ambiente é aquilo que apresenta as propriedades definidas na lista a seguir (RUSSELL; NORVIG, 2003; JENNINGS; LÉSPRANCE, 2000):

- *Acessibilidade vs Inacessibilidade*: um ambiente acessível é aquele em que um agente consegue ter informações claras, precisas e atualizadas no que tange a característica do ambiente.
- *Determinístico vs Não-Determinístico*: um comportamento determinístico é aquele em que uma ação possui um efeito claro e garantido, sem incertezas sobre o estado que irá resultar.
- *Episódico vs Não-Episódico*: um ambiente tende a ser o mais episódico possível tanto quanto o desempenho do agente estiver associado a um episódio discreto e específico no ambiente.

- *Estático vs Dinâmico*: um ambiente é estático se não houver outros processos em paralelo aos eventos associados ao agente.
- *Discreto vs Contínuo*: um ambiente é discreto se existir um número finito de ações e percepções.

Outro aspecto que está presente na definição de agente é o termo autônomo. Esse termo pode ser compreendido pela seguinte proposição: uma entidade que possui essa natureza é uma entidade que tem a capacidade de agir por si mesma. Essa entidade não precisa de nenhum outro fator externo (ex. ser humano) para efetuar as suas decisões (JENNINGS; LÉSPRANCE, 2000).

Há uma série de exemplos que se enquadram dentro dessa definição. Um termostato é um sistema de controle que está em um dado ambiente, como um quarto ou uma sala, (JENNINGS; LÉSPRANCE, 2000), que gera dois sinais de saída, (um desses sinais indica que a temperatura está ou baixa ou alta demais dependendo da aplicação, e o outro, demonstra que a temperatura está no nível aceitável). O comportamento autônomo do termostato é baseado nas seguintes regras:

- Se a temperatura estiver abaixo do nível de temperatura definido, então ligar o atuador.
- Se a temperatura estiver dentro do nível estabelecido, então desligar o atuador.

Outro exemplo de agentes consiste nos programas *Daemons* em sistemas *UNIX*. Esses algoritmos trabalham em segundo plano e monitoram um dado ambiente de *software*. Com base em certas regras, na ocorrência de um certo evento no ambiente, esses programas realizam uma determinada atuação (JENNINGS; LÉSPRANCE, 2000).

Os exemplos presentes neste texto são apropriados dentro do conceito de agentes. Contudo, esses exemplos não estão de acordo com a definição de agentes inteligentes (JENNINGS; LÉSPRANCE, 2000). Uma entidade que se enquadra dentro das características de um agente inteligente deve necessariamente respeitar a definição já apresentada e deve apresentar as seguintes propriedades: reatividade (capaz de perceber as mudanças que ocorrem no ambiente e responder a elas de maneira apropriada no que condiz aos objetivos do agente), proatividade (apresenta comportamento orientado a objetivos sendo que o agente toma as decisões a fim de satisfazer os objetivos em interesse) e habilidades sociais (capacidade de interagir com outros agentes a fim de poder satisfazer os próprios objetivos) (JENNINGS; LÉSPRANCE, 2000; RUSSELL; NORVIG, 2003).

Os agentes podem ser definidos em categorias. Uma dessas consiste nos agentes que são puramente reativos. Esses agentes tomam decisões considerando apenas informações que estão no instante presente. Por consequência, o comportamento deles ocorre por respostas diretas ao ambiente (JENNINGS; LÉSPRANCE, 2000). Os agentes que possuem estados estão

contidos em outra categoria. Esses agentes possuem uma dada estrutura interna de dados que são consideradas em tomadas de decisão (JENNINGS; LÉSPRANCE, 2000).

Uma outra maneira de analisar um agente se dá por meio das arquiteturas e modelos disponíveis para representar seus estados internos. Para o propósito do estudo que está sendo apresentando neste texto, é o suficiente considerar de forma sucinta quatro dessas arquiteturas. A primeira consiste nos agentes baseados em lógica que realizam deduções em função de uma decisão (GENESERETH; NILSSON, 1987), a segunda arquitetura consiste nos agentes reativos que tomam decisões com base em um mapeamento de uma certa situação em uma ação (BONASSO et al., 1995), a terceira arquitetura é *BDI* cujo comportamento ocorre da manipulação de estruturas de dados que representam crenças, desejos e intenções do agente (RAO; GEORGEFF, 1991) e a quarta arquitetura consiste em uma estrutura em camadas onde a tomada de decisão acontece por intermédio de diversos níveis de abstração a cerca do ambiente (WOOLDRIDGE; JENNINGS, 1995; JENNINGS; LÉSPRANCE, 2000).

Existe um modelo de agentes que faz uso da arquitetura *BDI* com propósito de representar situações de risco. Esse modelo é denominado por *MASVERP* que define o agente como tendo habilidades e objetivos a serem atingidos (EDWARD et al., 2008). Não só isso, mas o agente também pertence a certo círculo social. O autor deste estudo implementou o modelo *COCOM* que define parâmetros físicos e cognitivos a fim de que os agentes articulem suas ações no que tange a execução de objetivos (EDWARD et al., 2008). Esse modelo considera situações que delimitam quando um agente não tem condições de desempenhar uma certa atividade. Essas são: fome, sede, cansaço físico, carga cognitiva, cansaço em relação a vigilância, estresse, motivação para determinar agitação, motivação e condições regulares (EDWARD et al., 2008).

O *MASVERP* incorpora o conceito de *BTCU* (será tratado na seção 2.5). Em termos genéricos, *BTCU* consiste em condições limites que são aceitas para que um agente possa realizar a atividade. No *MASVERP* a decisão do agente no que tange a executar ou não uma atividade é parametrizada com base no *BTCU*.

## 2.2 Artefatos

A definição de um artefato pode ser feita por analisar antes o comportamento de um agente. De maneira genérica, existem duas formas que podem ser usadas para caracterizar o comportamento de um agente. Essas são: *goal-governed* e *goal-oriented* (CASTELFRANCHI et al., 2018) (RICCI; VIROLI; OMICINI, 2006).

*Goal-governed* são os agentes que apresentam capacidades cognitivas, que podem representar os seus respectivos objetivos e, portanto, são capazes de estruturar seu interesse (CASTELFRANCHI et al., 2018; RICCI; VIROLI; OMICINI, 2006). Em contraste, *goal-oriented* consiste nos agentes que são programados com a finalidade de alcançar um determinado ob-

jetivo (CASTELFRANCHI et al., 2018; RICCI; VIROLI; OMICINI, 2006). Em um sistema multiagente muitas vezes uma entidade não é adequadamente representada por nenhuma dessas duas categorias.

Assim sendo, artefatos são entidades que não se enquadram em *goal-governed* e não se enquadram em *goal-oriented*. Esses elementos são explicitamente projetados com o propósito de serem explorados pelos agentes para que possam alcançar seus objetivos individuais e sociais (RICCI; VIROLI; OMICINI, 2006; RICCI; VIROLI; OMICINI, 2007).

Para ilustrar, usando como referência a sociedade humana: se agentes (entidades autônomas) estão para seres humanos, então artefatos estão para as ferramentas (não autônomas) que são usadas com uma determinada finalidade (ex. um marceneiro, agente, usa um martelo (artefato) para pregar um prego (artefato)) (RICCI; VIROLI; OMICINI, 2006).

Uma outra distinção entre agentes e artefatos se torna claramente explícita quando verificada sob o ponto de vista conceitual e filosófico, pois os agentes apresentam a capacidade de se comunicar com outros agentes, em contraste, os artefatos não apresentam essa condição (RICCI; VIROLI; OMICINI, 2006).

Existem quatro elementos que podem ser usados para caracterizar um artefato (RICCI; VIROLI; OMICINI, 2006): *interface de uso*, *instruções de operação*, *funcionalidade* e *estrutura-comportamento*.

A *interface de uso* consiste no conjunto de operações que podem ser invocadas pelo agente a fim explorar as suas funcionalidades (RICCI; VIROLI; OMICINI, 2006). As *instruções de operação* consistem na descrição de como fazer o uso das funcionalidades do artefato, implicando nos protocolos de uso das operações que devem ser invocadas pelo agente (RICCI; VIROLI; OMICINI, 2006). A *Funcionalidade* do artefato consiste no propósito definido pelo programador do sistema (RICCI; VIROLI; OMICINI, 2006). A *estrutura-comportamento* consiste nos aspectos internos do artefato. Isso define como o artefato é implementado a fim de providenciar as suas funcionalidades (RICCI; VIROLI; OMICINI, 2006).

Existem arcabouços para especificar artefatos. Um desses recebe o nome de *Cartago* e será descrito com maior riqueza de detalhes na subseção que se segue.

### 2.2.1 Cartago

Cartago (*Common “Artefacts for Agents” Open Framework*) é um arcabouço usado para especificar as relações entre agentes e artefatos. Tendo em vista o fato de ser um dos principais *frameworks* na descrição dessas interações entre agentes e artefatos, não é possível falar no tema de artefatos em *SMA* sem ao menos comentar a estrutura conceitual presente no Cartago. Esse modelo é composto de três blocos, que são: *agent bodies*, *artefact* e *workspace* (RICCI; VIROLI; OMICINI, 2007).

*Agent bodies* é o que possibilita a inteligência do agente de se relacionar com o ambi-



ente. Para cada agente criado, há um *agent bodies* construído. Um *agent bodies* possui *effectors* (efetores) que tem o propósito de executar ações sobre o ambiente de trabalho e possui sensores (captadores de sinais do ambiente em sua volta). A relação completa entre agente-*agent bodies*-artefato é dada da seguinte forma: eventos observáveis são gerados pelos artefatos, sensores (componentes presentes no *agent bodies*) são sensibilizados, esta informação é transmitida para a inteligência do agente que, por sua vez, realiza os raciocínios e toma as decisões necessárias). O agente comunica ao *agent bodies* a ação que deve ser feita ao meio e por último, através do *effectors* uma ação é produzida no ambiente de trabalho.

*Artefacts* (artefatos) são os tijolos básicos na gerência do Cartago. Cada artefato apresenta um nome lógico e número de identificação (id) único que são definidos pelo *artifact creator* no momento da instanciação. O nome lógico consiste num caminho ágil para o agente poder referenciar e compartilhar o respectivo artefato com os demais agentes. O id é requisitado na identificação dos artefatos quando uma certa ação é executada. Os artefatos também apresentam nome completo que inclui o nome do(s) *workspace(s)* onde está logicamente localizados (RICCI; VIROLI; OMICINI, 2007).

A localização lógica dos artefatos fica dentro de *workspace*, que pode ser usado para definir a topologia do ambiente de trabalho. Neste âmbito, o *workspace* é feito com a finalidade de especificar nome lógico e id. Está dentro do escopo deste item definir uma topologia de ambiente possibilitando uma estrutura de interação entre agentes e artefatos. Assim sendo o agente só pode usar um artefato que está no mesmo *workspace* onde ele está contido.

## 2.3 Sistema Multiagente

Um sistema multiagente (SMA) organizado é aquele constituído por agentes autônomos que interagem visando um propósito em comum, tendo como consequência um comportamento global (HÜBNER; SICHMAN; BOISSIER, 2002b; ABBAS; SHAHEEN; AMIN, 2015). Assim sendo, uma organização com essas características deve ser capaz de manifestar conhecimento em comum, cultura, memória, história, distribuição de atividades e a capacidade de distinguir um agente em específico (ABBAS; SHAHEEN; AMIN, 2015). Deste fato é possível identificar o fenômeno *supra-individual* que implica em um comportamento que existe além dos comportamentos e atributos particulares no que diz respeito as entidades constituintes do sistemas.

Uma organização de um sistema multiagente deve conter relações sociais no que tange a agentes, institutos e grupos sociais (ABBAS; SHAHEEN; AMIN, 2015). Também deve apresentar uma *extensão de um espaço abstrato*. Isso implica uma representação dos seguintes conceitos: estrutura espacial, estrutura temporal, símbolos, semântica e capacidade de dedução. Há organizações que não se enquadram em todas essas restrições, contudo são suficientes para tratar o problema dentro de uma perspectiva computacional (ABBAS; SHAHEEN; AMIN, 2015).

A subseção *Conceitos Gerais de uma Organização de Sistemas Multiagentes* tem por

finalidade, detalhar melhor os elementos presentes da Teoria da Organização dentro do contexto de SMA em relação a esse estudo. Já a subseção *Formalização de Conceitos Específicos para SMA* tem por objetivo realizar uma verificação analítica dos elementos presentes no modelo de SMA denominado por *MOISE+*. Os conceitos que serão analisados são: objetivos, planos e papéis (ABBAS; SHAHEEN; AMIN, 2015).

### 2.3.1 Conceitos Gerais de uma Organização de Sistemas Multiagentes

A finalidade desta subseção consiste em trabalhar com uma maior riqueza de detalhes, todos os conceitos que constituem a ideia de uma organização de um sistema multiagente. Esses conceitos estão estruturados no texto que se segue.

**Divisão em tipos de atividades:** uma organização não é uniformemente estruturada. Isso, pois as atividades são distribuídas de forma desigual entre as diferentes entidades. Dentro do ponto de vista fenomenológico as atividades estão sujeitas a classificação, e ocorrem com diferentes frequências e em diferentes regiões dentro das definições espaciais da organização (ABBAS; SHAHEEN; AMIN, 2015).

**Integração:** dentro de uma organização ocorre a presença de interdependência entre diferentes espaços de atividades. Essas, por sua vez, estão relacionadas em uma estrutura única definida dentro de um contexto alinhado e integrado (ABBAS; SHAHEEN; AMIN, 2015).

**Composição:** uma organização é composta por elementos menores. No caso de uma SMA, os elementos atômicos que estruturam a organização são os agentes (ABBAS; SHAHEEN; AMIN, 2015).

**Estabilidade/Flexibilidade:** uma organização apresenta padrões de atividades. Esses padrões possuem características que podem ser enquadradas em dois aspectos: estáveis e flexíveis. No que tange as características estáveis, essas são constituídas por elementos/processos que definem o padrão em si mesmo. Em contraste com isso um comportamento flexível acontece quando o sistema é submetido a situações incomuns (ABBAS; SHAHEEN; AMIN, 2015).

**Coordenação:** todo sistema é dependente de algum recurso. Assim sendo, é preciso que esse recurso seja utilizado de forma inteligente a fim de que possa se manter ao longo do tempo. Para isso, se faz necessário que a organização se comporte como uma amplificadora de recursos a fim de que as estruturas operacionais tenham um comportamento cada vez mais organizado (ASHBY, 1962; FOERSTER, 2003; LENDARIS, 1964). Contudo as incertezas relacionadas aos efeitos combinados resultam em influências negativas nas eficiências. Portanto, para manter a eficiência organizacional se faz necessário a existência de elementos otimizadores sobre os padrões de atividades (ABBAS; SHAHEEN; AMIN, 2015).

**Recursividade:** uma organização é constituída por sub-organizações. Isso ocorre em diferentes níveis de estrutura e se dá por intermédio de um padrão recursivo (ABBAS; SHAHEEN; AMIN, 2015).

**Representação Multi-Nível e Causalidade:** a natureza recursiva das organizações resultam em atividades ocorrendo em diferentes escalas espaciais, temporais e estruturais. Como consequência disto, as cadeias causais presentes em estruturas organizacionais são processos multi-níveis (ABBAS; SHAHEEN; AMIN, 2015).

**Potenciais e Diferenciais:** diversos são os sistemas físicos onde as forças entre partículas são decorrentes de balanços de potenciais. Como esse comportamento está presente em diversos sistemas físicos, existem modelos abstratos de sistemas auto-organizáveis que levam em consideração a presença de forças potenciais e diferenciais em organizações (PRIGOGINE; NICOLIS, 1985). Esse conceito é trabalhado dentro de sistemas multiagentes. Um exemplo notório é o conceito de *Poder*, o qual é entendido como a capacidade de influenciar uma organização (ABBAS; SHAHEEN; AMIN, 2015).

**Regras e Gramáticas:** organizações podem ser compreendidas como potenciais configurações de atividades e processos. Essas configurações podem ser descritas usando gramática (PENTLAND, 1995; PENTLAND; RUETER, 1994). Tanto as gramáticas como as regras que compõem uma organização apresentam três interpretações, que são: como estruturas (especificações procedurais do que deve ser feito): como coação (as ações são definidas no que pode e não pode ser feito) e como um compilado das experiências (ABBAS; SHAHEEN; AMIN, 2015).

**Incerteza:** não é possível conceber o conceito de uma organização sem ao menos entendê-la como uma estrutura que distribui informação em si mesma. Sob esta ótica, a distribuição de informação inequivocamente implica geração de incerteza o que por sua vez se manifesta como um complicante no que tange a comunicação entre as partes bem como a atividade organizacional em si mesma.

### 2.3.2 Formalização de Conceitos Específicos para SMA

A apresentação desses conceitos será feita por intermédio de estudos relacionados ao *MOISE+*. Apesar de ser um arcabouço, o *MOISE+* trata a rigor acadêmico na ótica da computação clássica a tratativa dada para os conceitos em interesse a esse estudo. Assim sendo, uma análise aprofundada do modelo, bem como dos textos em referência, satisfazem com excelência os fundamentos teóricos para os estudos em desse texto.

A constituição do *MOISE+* é estruturada em três categorias de especificação, essas são: estrutural, funcional e deôntica.

A especificação estrutural acontece em três níveis: individual: social e coletivo. O nível individual trata de definir os papéis  $\rho$  dos agentes. A hereditariedade é uma relação que se dá entre dois papéis em que se  $\rho'$  é filho de  $\rho$  então  $\rho'$  é uma especialização de  $\rho$ . Um exemplo apropriado para isso é o jogo de futebol onde existe o papel jogador dado por  $\rho$  e existe o papel atacante dado por  $\rho'$  (HÜBNER; SICHTMAN; BOISSIER, 2002b; FERBER; GUTKNECHT,

1998; FOX; BARBUCEANU; GRUNINGER, 1996; CARRON; BOISSIER, 2001). Em termos formais, essa relação é dada por:

$$\rho_a \sqsubset \rho_b \quad (2.1)$$

O nível social estabelece relações de ligação dados pelo predicado  $link(\rho_s, \rho_d, t)$ . Existem três possíveis valores para  $t$ , os quais são  $t = \{aut, com, acq\}$ . O valor  $auth$  significa autoridade (neste caso  $\rho_s$  exerce autoridade sobre  $\rho_d$ ), o valor  $com$  significa comunicação (neste caso  $\rho_s$  pode se comunicar com  $\rho_d$ ) e o valor  $acq$  significa conhecimento ( $\rho_s$  tem conhecimento da existência de  $\rho_d$ ) (HÜBNER; SICHMAN; BOISSIER, 2002b; CARRON; BOISSIER, 2001). O MOISE+ define as seguintes relações de implicabilidade

$$\begin{aligned} link(\rho_s, \rho_d, auth) &\rightarrow link(\rho_s, \rho_d, com) \\ link(\rho_s, \rho_d, com) &\rightarrow link(\rho_s, \rho_d, acq) \end{aligned} \quad (2.2)$$

O modelo também determina como se dão as relações de hereditariedade para o predicado de  $link$  (HÜBNER; SICHMAN; BOISSIER, 2002b; CARRON; BOISSIER, 2001):

$$\begin{aligned} link(\rho_s, \rho_d, t) \wedge \rho'_s \sqsubset \rho_s &\rightarrow link(\rho'_s, \rho_d, t) \\ link(\rho_s, \rho_d, t) \wedge \rho'_d \sqsubset \rho_d &\rightarrow link(\rho_s, \rho'_d, t) \end{aligned} \quad (2.3)$$

O nível coletivo determina a existência de compatibilidade entre os papéis (HÜBNER; SICHMAN; BOISSIER, 2002b). Essa é uma relação reflexiva e o seu conceito se dá através da seguinte proposição: Se papel  $\rho_a$  possui a capacidade de realizar um determinado objetivo, então o papel  $\rho_b$  também tem essa capacidade. Em termos formais, essa relação se dá da seguinte forma (HÜBNER; SICHMAN; BOISSIER, 2002b; CASTELFRANCHI, 1995):

$$\rho_a \bowtie \rho_b \wedge \rho_a \neq \rho_b \wedge \rho_a \sqsubset \rho' \rightarrow \rho' \bowtie \rho_b \quad (2.4)$$

O nível coletivo também apresenta o conceito de grupo dado por  $gt$  o qual é constituído por:

$$gt = \langle R, SG, L^{intra}, L^{inter}, C^{intra}, C^{inter}, np, ng \rangle \quad (2.5)$$

Em que  $R$  é o conjunto dos papéis não abstratos,  $SG$  são subgrupos que estão contidos neste grupo,  $L^{intra}$  consiste dos *links* intra-grupos,  $L^{inter}$  dos links inter-grupos,  $C^{intra}$  das relações de compatibilidade intra-grupos e  $C^{inter}$  das relações de compatibilidade inter-grupos. O

símbolo  $np$  denota a cardinalidade mínima e máxima para uma dada função e o símbolo  $ng$  realiza o mesmo para os subgrupos (HÜBNER; SICHMAN; BOISSIER, 2002b).

A Especificação Funcional tem a finalidade descrever os objetivos a serem atingidos dentro de uma estrutura de árvore. A Figura 1 a seguir define como se dá esse tipo de especificação:

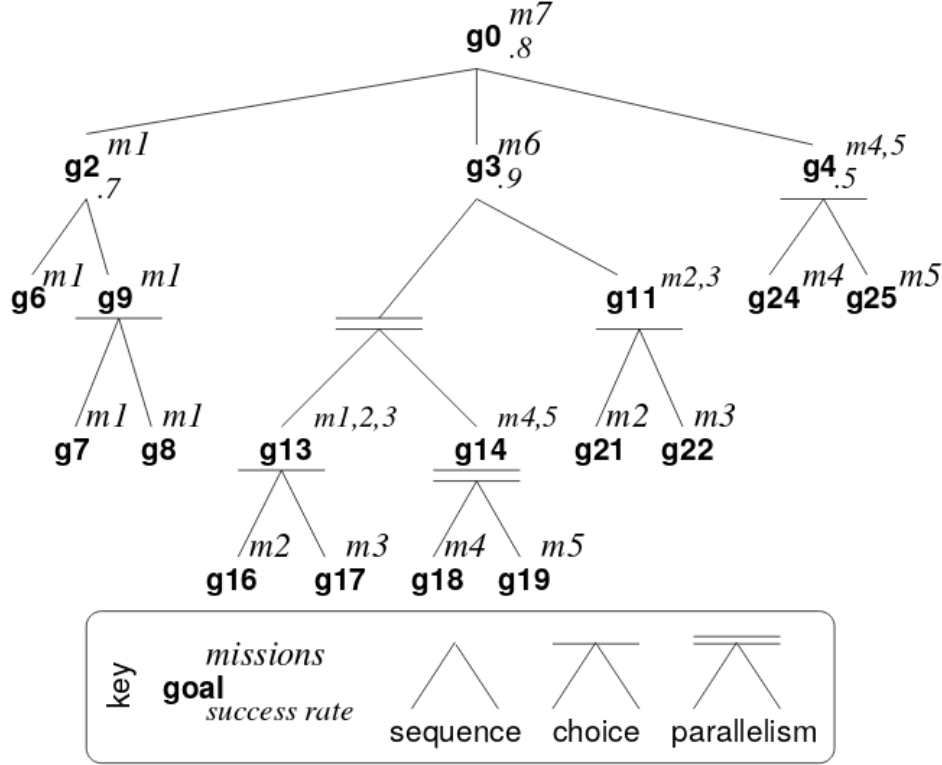


Figura 1 – Árvore de objetivos definido pelo modelo MOISE+ (HÜBNER; SICHMAN; BOISSIER, 2002b)

A Figura 1 define três tipos de relação de sub-objetivos: *sequence* onde todos os sub-objetivos devem necessariamente ser concluídos em sequência; *choice* onde o agente tem a possibilidade de escolher qual objetivo ele deseja seguir, e *parallelism* onde todos os objetivos devem ser concluídos, contudo sem uma sequência definida (EXPLORING..., 1996; SO; DURFEE, 1996). Esta parte do modelo é baseada em um arcabouço de distribuição de atividades denominado por *TAEMS* (GARVEY; LESSER, 1996).

Como é possível observar na Figura, os objetivos são agrupados em conjuntos de missões  $m$  (CARRON; BOISSIER, 2001). A relação a seguir define isso melhor:

$$m_k = \{g_n, \dots, g_m\} \quad (2.6)$$

A Especificação Deontica define predicados para estabelecer permissões e obrigações entre os papéis e as missões. Toda obrigação implica necessariamente em uma permissão, con-

forme define a relação a seguir (HÜBNER; SICHMAN; BOISSIER, 2002b; CASTELFRANCHI, 1995):

$$obl(\rho, m, tc) \rightarrow per(\rho, m, tc)$$

$$obl(\rho, m, tc) \wedge \rho \sqsubset \rho' \rightarrow obl(\rho', m, tc) \quad (2.7)$$

$$per(\rho, m, tc) \wedge \rho \sqsubset \rho' \rightarrow per(\rho', m, tc) \quad (2.8)$$

$$(2.9)$$

Onde o predicado *obl* define uma obrigação e o predicado *per* define permissão. O argumento *tc* define uma periodicidade de tempo para o qual a relação deôntica é válida.

## 2.4 Normas

Quando se trata de normas em sistemas multiagentes é de crucial importância definir claramente este termo. Isso se deve ao fato de que há diversos estudos que tratam o conceito de norma sobre perspectivas diferentes. Por exemplo, os estudos (ESTEVA; PADGET; SIERRA, 2002; MOSES; TENNENHOLTZ, 1995) apresentam normas, em sistemas multiagentes, para representar a presença de sociedades, institutos e organizações.

Há estudos que tratam normas como maneiras dos agentes trabalharem de forma coordenada com propósito de cumprir um objetivo global e também como uma maneira de obedecer as autoridades do sistema (LOPEZ; LUCK, 2003; LÓPEZ; LUCK, 2004).

No *MOISE+* normas são tratadas sob a ótica da lógica deôntica e é usada para especificar os agentes com dado papel no que condiz com as suas obrigações em relação as missões (HÜBNER; SICHMAN; BOISSIER, 2002b; HÜBNER; SICHMAN; BOISSIER, 2002a).

O estudo de (DASTANI et al., 2009) apresenta uma linguagem formal para especificar sistemas multiagentes normativos. Esta linguagem contém os conceitos de normas que são os mesmos usados neste texto. A Figura 2 apresenta a linguagem em notação EBNF (DASTANI et al., 2009).

```

N-MAS_Prog  := "Agents: " (<agentName> <agentProg> [<nr>])+ ;
              "Facts: " <bruteFacts>
              "Effects: " <effects>
              "Counts-as rules: " <counts-as>
              "Sanction rules: " <sanctions>;

<agentName> := <ident>;
<agentProg> := <ident>;
<nr>        := <int>;
<bruteFacts>:= <b-literals>;
<effects>   := ({<b-literals>} <actionName> {<b-literals>})+;
<counts-as> := ( <literals> ⇒ <i-literals> )+;
<sanctions> := ( <i-literals> ⇒ <b-literals> )+;
<actionName>:= <ident>;
<b-literals>:= <b-literal> {"," <b-literal>};
<i-literals>:= <i-literal> {"," <i-literal>};
<literals>  := <literal> {"," <literal>};
<literal>   := <b-literal> | <i-literal>;
<b-literal> := <b-prop> | "not" <b-prop>;
<i-literal> := <i-prop> | "not" <i-prop>;

```

Figura 2 – Linguagem para descrever um programa de multiagentes normativos com a possibilidade de violações e sanções na notação EBNF segundo o texto (DASTANI et al., 2009). Nesta notação, *< ident >* é usado para denotar uma *string* e *< int >* inteiros. Os termos *< b – prop >* e *< i – prop >* são usados para designar dois tipos de conjuntos de proposições que são disjuntos entre si

Os *Facts* implementados em forma de *brute facts* definem os estados iniciais do sistema dentro de um ambiente compartilhado por todos os agentes. O termo *Effects*, implementado por meio de *effects* define como se dá a transição de estados do sistema. A tag *< actionName >* representa os eventos que geram transição dos estados. Portanto uma norma é definida em termos de *Counts\_as rules*. O termo *< counts – as >* aponta para transição entre *< literals >* e *< i – literals >* representando os fatos que resultam em violações. Assim sendo em (DASTANI et al., 2009) as normas são descritas por intermédio de suas violações. Violação, dentro deste contexto, é dado como o descumprimento da norma (WRIGHT, 1969).

O termo *Sanction Rules* aponta para *< sanctions >* e esses, por sua vez, para a transição entre *< i – literals >* e *< b – literals >* sendo que esses *< i – literals >* através de *Counts\_as Rules*. Assim sendo, *Sanction Rules* define as consequências da violação. Essas consequências denotam caráter negativo ao agente tendo como foco uma natureza de ordem punitiva.

A Figura 3 apresenta um programa escrito nesta linguagem.



```

Agents:      passenger  PassProg  1
Facts:       {-at_platform, -in_train, -ticket}
Effects:     {-at_platform} enter {at_platform},
             {-ticket} buy_ticket {ticket},
             {at_platform, -in_train} embark {-at_platform, in_train}
Counts_as rules: {at_platform , -ticket}  $\Rightarrow$  {viol1},
                 {in_train , -ticket}  $\Rightarrow$  {viol1}
Sanction rules: {viol1}  $\Rightarrow$  {fined10}

```

Figura 3 – Um programa descrito na linguagem proposta neste estudo onde um agente representa um passageiro em uma estação de trem que pode entrar com ou sem um *ticket* na plataforma e no trem (DASTANI et al., 2009).

Este programa contém um agente chamado de *passenger* (especificações deste agente são detalhadas com maior rigor em *PassProg*). Os *Facts* deste programa são *-at\_plataforma* (agente não está na plataforma), *-in\_train* (agente não está no trem) e *-in\_ticket* (agente não possui o *-in\_ticket*). As regras de *Effects* apresentam dois *< actionName >*. O primeiro é denominado por *enter* que tem por finalidade alternar entre os estados *-at\_plataforma* e *at\_plataforma*, ou seja, é uma ação onde o agente muda o estado de, não está na plataforma para, está na plataforma. O segundo estado é o *buy\_ticket* que gera a transição entre os fatos *-ticket* para *ticket*, ou seja, na ocorrência *buy\_ticket* o agente passa a ter um *ticket* (DASTANI et al., 2009).

O programa especifica duas regras em *Counts\_as rules*. A primeira regra denota a ocorrência de uma violação para um agente que entra numa plataforma sem um *ticket*. A segunda regra define que uma violação que acontece para o caso de um agente entrar em um trem sem um *ticket* (DASTANI et al., 2009).

O programa também define uma regra para *Sanction rules*. Essa regra se aplica a *viol<sub>1</sub>*, que - se verdade - então resulta no fato *fined<sub>10</sub>*, cuja semântica denota um pagamento no valor de 10 unidades de moeda (DASTANI et al., 2009).

## 2.5 Riscos

O primeiro estudo teórico sobre acidentes de trabalho é feito no texto (RASMUSSEN, 1997). Essa pesquisa conclui que os erros em indústria não podem ser definidos apenas nas falhas de humanos, mas sim como consequência de um comportamento global da instituição. Ainda dentro deste âmbito, esse comportamento advém de uma forte pressão tendo em vista eficiência e otimização dos processos de produção (RASMUSSEN, 1997; FADIER; GARZA; DIDELOT, 2003).

Com base neste entendimento, o estudo (FADIER; GARZA; DIDELOT, 2003) apresenta um arcabouço a fim de identificar as redes de causalidade que resultam em acidentes de trabalho. Assim sendo, a gestão é feita com base nos seguintes fatores:

- Políticas: *leis, diretrizes, padrões e regras*.



- Corporativo: *regras, estratégias, políticas internas, gerenciamento.*
- Projeto de Equipamentos de Trabalho: *especificação, integração de segurança.*

O item política é mais relevante que os itens corporativo e projeto de equipamentos de trabalho. Esses dois últimos apresentam a mesma importância para uma estrutura de prevenção de acidentes bem sucedida.

A primeira análise a ser feita diz respeito ao nível corporativo-projeto de equipamentos de trabalho. Muitas vezes a equipe adota atividades paliativas a fim de otimizar os processos de produção. Isso envolve assumir níveis de tolerância no que diz respeito ao desempenho e a segurança. Esta situação está dentro do conceito, para o arcabouço, de *atividades limites*. Isto, pois tratam de situações que trabalham no limiar com os riscos. Assim sendo, as decisões feitas pelo profissional podem resultar muito facilmente em acidentes ou incidentes (FADIER; GARZA; DIDELOT, 2003). Dentro desta perspectiva é que se apresenta o ator *BATU* - *Boundary Activities Tolerated during Use* (Atividades Limites Toleradas Durante o Uso).

Existe dois tipos de *BATU* que devem ser verificados tendo como base os processos de trabalho: operacional e gerencial (administrativo - termo identificado no texto original: *managerial*). Aquele, faz referência as atividades relacionadas à melhoria da produtividade com o propósito de resultar em aumento das metas de produção, qualidade e segurança. Este, diz respeito às decisões administrativas independentes dos processos operacionais mas que os impactam.

Outro conceito presente em (FADIER; GARZA; DIDELOT, 2003) é o de *Boundary Conditions Tolerated by Use* - *BCTU* (Condições Limites Toleradas Durante o Uso). O termo condição faz referência a uma situação, estado ou a algumas circunstâncias externas às quais pessoas ou até mesmo entidades são afetados no que diz respeito a uma certa decisão. Assim sendo, *BCTU* consiste em uma série de elementos e circunstâncias (ambiental, material, humana, produtos) que por conta de sua natureza ou de como se relaciona com as demais entidades e processos apresenta um certo potencial na geração de situações particulares, tendo em vista causas decorrentes de operações dinâmicas. Tanto os *BATUs* bem como os *BCTUs* não podem ser analisados diretamente, mas por intermédio das ações e escolhas dos operadores e dos atores que constituem esse trabalho (FADIER; GARZA; DIDELOT, 2003).

Existem dois tipos de *BCTU*. O primeiro consiste no *BCTU* interno que se apresenta como uma concepção global de trabalhos e situações no que tange as relações de política da empresa. Nesta concepção, *BCTU* interno faz referência às diferentes hierarquias em termos de nível e decisões centrais. Em contraste com esse ponto, o *BCTU* externo aponta para o projeto da instalação. Como resultado, há o surgimento de quatro derivações, que são: soluções de segurança - funções de segurança (diz respeito as questões que podem fazer com que um dispositivo de segurança venha a falhar), soluções técnicas - requisições de trabalho (quando as soluções técnicas são incompatíveis com as requisições de trabalho), modelo de projeto - modelo de ins-

talação (ocorre quando a solução final não é ótima ou está degradada quando comparada com a solução inicial) e condições nominais preventivas - condições reais de operação (FADIER; GARZA; DIDELOT, 2003).

As relações entre *BATUs* e *BCTUs* são dinâmicas e são dependentes do processo. Para exemplificar, pode-se considerar o seguinte cenário: o projeto de uma máquina de dobra de papel, obriga o operador a adotar uma posição que o faz assumir riscos para acessar determinados pontos da máquina. Assim sendo, as escolhas do projeto da máquina (relacionada ao *BCTU*) não levam em consideração todos os aspectos relacionados a dinâmica profissional-máquina, fazendo o que o profissional envolvido tenha que atuar dentro de um certo intervalo de tolerância no que diz respeito a segurança profissional *BATU*.

## 2.6 Lógica Modal

A lógica modal consiste em uma linguagem para tratar proposições que necessariamente ocorrem e proposições que possivelmente ocorrem. As proposições dadas como necessárias são aquelas que necessariamente são verdade. Por exemplo: a água sobre 1 atm e entre 0,1 °C - 99 °C se apresenta no estado líquido. O conceito de possibilidade é totalmente dependente do conceito de necessidade. Isso, pois uma proposição possível é aquela que necessariamente não é falsa (GARSON, 2018).

A lógica modal é do tipo **K** e isso significa que nela está contida símbolos  $\sim$  para não,  $\rightarrow$  para “se ... então” e  $\Box$  para “Isto é necessário”.

De **K** e  $\Box$ , tem-se as seguintes regras:

Sendo que  $isTheorem(A, \mathbf{K})$  representa “Se A é teorema de **K**”.

$$isTheorem(A, \mathbf{K}) \rightarrow \Box A \quad (2.10)$$

$$\Box(A \rightarrow B) \rightarrow (\Box A \rightarrow \Box B) \quad (2.11)$$

O operador  $\Diamond$  apresenta o seguinte correspondente semântico: “Isto é possível”. A relação entre  $\Box$  e  $\Diamond$  é dada pela regra que se segue.

$$\Diamond A = \sim \Box \sim A \quad (2.12)$$

As relações a seguir apresentam outras regras válidas para essa lógica:

$$\Box(A \wedge B) \rightarrow \Box A \wedge \Box B \quad (2.13)$$

$$\Box A \vee \Box B \rightarrow \Box(A \vee B) \quad (2.14)$$

$$\Box A \rightarrow A \quad (2.15)$$

$$\Box A \rightarrow \Box \Box A \quad (2.16)$$

$$\Diamond A \rightarrow \Box \Diamond A \quad (2.17)$$

$$\Box \Box \dots \Box = \Box \quad (2.18)$$

$$\Diamond \Diamond \dots \Diamond = \Diamond \quad (2.19)$$

$$A \rightarrow \Box \Diamond A \quad (2.20)$$

## 2.7 Demais Modelos

Essa seção apresenta outros arcabouços que foram usados na estrutura do modelo conceitual em análise. Esses modelos possuem em sua respectiva estrutura conceitos que foram apresentados nas seções anteriores.

### 2.7.1 Modelo V3S

V3S é um modelo com a finalidade de gerar ambientes para desenvolver treinamentos complexos em ambiente de realidade virtual visando atividades de risco e de emergência. O modelo é composto por três submodelos: *Domain Model*, *Activity Model* e *Risk Model* (BAROT et al., 2013). O *Domain Model* é o núcleo do sistema. Todos os objetos, ações e relações são descritos por uma ontologia. A Figura 4 exibe a estrutura de classe desta ontologia.

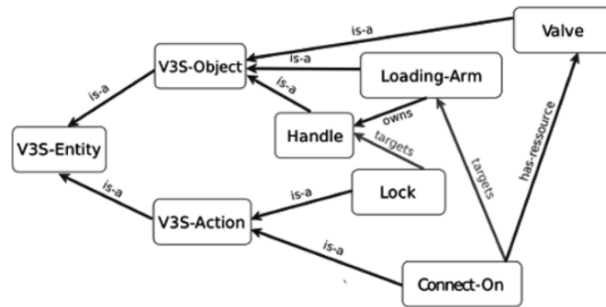


Figura 4 – Ontologia que descreve *Domain Model* no modelo V3S (BAROT et al., 2013)

*Activity Model* é estruturado sobre uma linguagem de descrição conhecido por *ACTIVITY-DL*. Um dos elementos dessa linguagem é baseado na álgebra de Allen's que tem como finalidade definir raciocínios temporais (ALLEN, 1983). As relações definidas por essa álgebra é dada por:

1.  $X < Y$  onde  $X$ : ocorre antes de  $Y$
2.  $XmY, YmiX$ :  $X$  encontra  $Y$
3.  $XoY, XoiY$ :  $X$  sobrepõem a  $Y$
4.  $XsY, YsiX$ :  $X$  começa  $Y$
5.  $XdY, YdiX$ :  $X$  ocorre durante  $Y$
6.  $XfY, YfiX$ :  $X$  termina junto com  $Y$
7.  $X = Y$   $X$  é igual a  $Y$

A *Activity Model* define construtores que são semanticamente equivalente a certos operadores da álgebra de Allen's. Esses construtores (atuantes sobre atividades) são definidos pela Tabela 1

Construtor	Nome	Relações de Allen
IND	Independent	$A\{<, >, m, mi, o, oi, s, si, d, di, f, fi, =\}B$
SEQ	Sequential	$A\{<, >, m, mi\}B$
SEQ-ORD	Ordered	$A\{<, >, m\}B$
PAR	Parallel	$A\{o, oi, s, si, d, di, f, fi, =\}B$
PAR-SIM	Simultaneous	$A\{=\}B$
PAR-START	Start	$A\{s, si, =\}B$
PAR-END	End	$A\{f, fi, =\}B$

Tabela 1 – Construtores da linguagem *ACTIVITY-DL* (BAROT et al., 2013)

As relações temporais entre as subatividades são especificadas por intermédio de construtores que são formalmente definidos no estudo (ALLEN, 1983). Essas relações são intermediadas pelo vocábulo *Pré-condição* que tem como propósito apresentar o contexto sobre qual uma atividade deve ser executada. A Tabela 2 apresenta esses contextos.

<b>Categoria</b>	<b>Pré-condição</b>	<b>Descrição</b>
Condições para perceber	Nomológico	Descreve o estado de mundo necessário para que a tarefa seja fisicamente realizável. Condições dependem diretamente das regras de ação definidas no modelo de domínio. Exemplo: abre a porta se estiver fechada.
Condições para perceber	Regulamentar	Descreve o estado de mundo necessário para uma boa realização da atividade de acordo com prescrito em procedimento. Exemplo: para desconectar o tubo, as proteções devem ser desgastadas.
Condições para examinar	Contextual	Descreve o estado de mundo em que a atividade é relevante. Quando essa condição é falsa, então a atividade deve ser ignorada. Exemplo: limpar o tubo é relevante apenas se o tubo estiver sujo.
Condições para examinar	Favorável	Descreve o estado de mundo onde a tarefa é preferencial sobre as demais. Essas condições ajudam a escolher entre várias tarefas quando existe uma alternativa para a realização de uma tarefa decomposta. Exemplo: se o parafuso estiver enferrujado, desarmar.

Tabela 2 – As pré-condições possíveis para as atividades (BAROT et al., 2013)

No que tange a questões referentes a segurança e violação, a linguagem *ACTIVITY-DL* deve lidar com atividades em estados de alta degradação bem como com compromissos cognitivos que são um grande potencial para a geração de risco. Essa condição possibilita a verificação de erros nos seguintes aspectos: atividades de aprendizagem e demonstração de comportamentos similares tendo como base personagens virtuais (BAROT et al., 2013). Por conta disso, a linguagem *ACTIVITY-DL* incorpora os conceitos de BCTUs e BATUs. Ambas tags trabalham com o fato de que, ao menos em partes, o profissional decide por cometer uma violação tendo em vista a inviabilidade (ou por não ser prático) efetuar a ação com base no que é definido pelos manuais.

*Risk Models* é a parte do modelo que define a análise de risco. Existem duas categorias: risco de análise clássica e método de análise de confiabilidade humana. A primeira categoria permite definir uma análise quantitativa de risco, contudo falha ao definir a complexidade dos resultados frente a fatores humanos. Em contrapartida, a segunda categoria considera fatores humanos, contudo falha em definir medidas objetivas sobre questões de segurança (BAROT et al., 2013). O V3S combina ambas situações usando a abordagem MELISSA (CAMUS; LENNE; PLOT, 2012; BAROT et al., 2013). Essa abordagem é baseada em três pontos (1) atividades relacionadas em cenários de acidentes, (2) descrição das tarefas de representação e (3) fatores influentes em potencial nas atividades. MELISSA representa o cenário de acidente por meio do gráfico *Bowtie*, o qual consiste na identificação de todos os cenários de acidentes bem como no provisionamento e uma listagem de barreiras para os mesmos. O risco aceitável consiste em escolhas que verificam o número e desempenho dessas barreiras. O ponto central do gráfico de *Bowtie* consiste em eventos críticos, a parte da esquerda desse gráfico implica as causas do evento e a parte direita do mesmo corresponde as consequências do evento (BAROT et al., 2013; CAMUS; LENNE; PLOT, 2012). Essa descrição pode ser analisada na Figura 5.

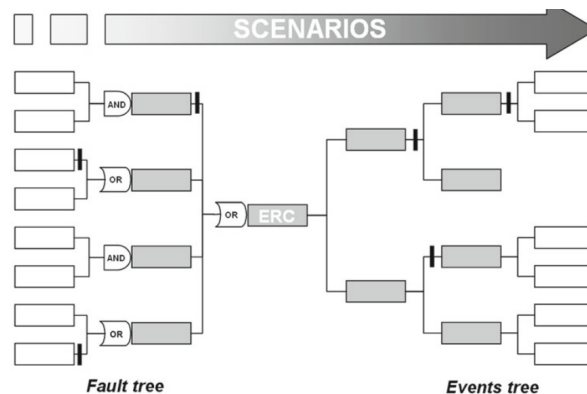


Figura 5 – Gráfico de Bowtie do texto (CAMUS; LENNE; PLOT, 2012)

Com o propósito de gerar reflexões no que tange aos riscos de dada atividade, o V3S trabalha com o conceito de personagens virtuais em um ambiente. Os raciocínios a cerca destes personagens são feitos usando um formalismo matemático denominado por redes de Petri, ou - mais especificamente Máquinas de Estado (BAROT et al., 2013) tendo como base simular a complexidade, flexibilidade e variabilidade de comportamentos que podem ser verificados em um ser humano. Por conta disso, os cientistas desse estudo decidiram modelar esse comportamento usando sistemas multi-agentes, mas especificamente um *framework* conhecido como MASVERP (tratado na seção 2.1).

O V3S tem como finalidade providenciar um modelo que seja coerente, relevante, variado e eficiente em termos de cenário de treinamento com a finalidade de proporcionar atividades de aprendizagem. Esse modelo também apresenta um módulo que monitora cenários adaptativos conhecidos por *HERA*. Em vez de interromper o usuário de forma sistemática a fim de explicar os seus erros, o *framework* possibilita que o agente cometa erros e observe suas res-

pectivas consequências no mundo virtual. Portanto, a dinâmica do cenário adaptativo permite trazer situações de treinamento relevantes.

*HERA*, por intermédio de regras baseadas em modelos pedagógicos, fornece o respaldo ao instrutor. Esse retorno é feito por intermédio dos seguintes critérios pedagógicos: escala de modificação - ampliar determinadas partes de um objeto com a finalidade de melhorar a visualização; reificação - verificar como o aprendiz lida com determinados conceitos e abstrações em termos concretos, restrições nos limites das ações do aprendiz que consistem em envio de mensagem ao agente quando ele comete sérios erros e superposição de informações se o aprendiz cometer e argumentar sobre as consequências das ações. *HERA* é integrado ao módulo de reconhecimento que tem como finalidade usar técnicas que permitem redefinir as relações de atividades usando a linguagem *ACTIVITY-DL* parametrizando-se nas ações, erros e violações. Essa parte do *V3S* é capaz de distinguir entre os tipos de erros, que são: 1 - erros relacionados a atividades, 2 - erros relacionados ao ato de cumprir com o objetivo, 3 - erros de *BATU*, 4 - erros de função e 5 - erros de ponto de vista.

### 2.7.2 Modelo NormMAS

*NormMAS* é um modelo usado para definir comportamento normativo de sistemas multiagentes (CHANG; MENEGUZZI, 2016). No que tange questões referentes ao comportamento normativo, o modelo trabalha com duas definições (CHANG; MENEGUZZI, 2016).

**Definição 1.** *Um norma é definida por meio de uma tupla  $N = \langle \mu, \kappa, \chi, \tau, \rho \rangle$*

- $\mu \in \{obligation, prohibition\}$  representa as modalidades de norma;
- $\kappa \in \{action, state\}$  representa o tipo de *trigger* da condição;
- $\chi$  representa o conjunto de estados em que uma norma se aplica;
- $\tau$  representa a norma da condição de *trigger*;
- $\rho$  representa a sanção aplicada pela violação do agente.

A definição 1 pode ser compreendida sobre o seguinte exemplo:

**Exemplo da Definição 1.** *Todos os imigrantes que possuem passaporte válido, devem ser aceitos. A falha resulta na perda de 5 créditos.*

Dentro da definição 1, o exemplo fica:

$$\langle obligation, action, valid(Passport), accept(Passport), loss(5) \rangle \quad (2.21)$$

*NormMAS* define um *Registro de ação* que é dado pela definição 2.

**Definição 2.** *Um Registro de Ação é definido por meio de uma tupla  $R = \langle \gamma, \alpha, \beta \rangle$*



- $\gamma$  representa o agente executando uma ação;
- $\alpha$  representa a ação sendo executada pelo agente  $\gamma$ ;
- $\beta$  representa os estados internos do agente  $\gamma$  no momento da execução.

Para demonstrar como se dá o uso dessa definição pode-se considerar a seguinte sentença:

*O oficial John aprovou passport 3225. O passaporte 3225 é definido como validado.*

Nessa sentença, *John* é o agente dado por  $\gamma$ , o ato de aprovar o passaporte é o  $\gamma$  que pode ser definido pelo predicado por  $approve(3225)$  e o estado de ser validado por ser dado pelo predicado  $valid(3225)$ .

Usando a **Definição 2**, isso poder ser especificado da seguinte maneira:

$$\langle John, approve(3225), valid(3225) \rangle \quad (2.22)$$

## 3 Metodologia

Esse capítulo apresenta os recursos, técnicas e estratégias adotados com a finalidade de atingir o objetivo dessa pesquisa. As etapas adotadas são as seguintes: revisão bibliográfica em conjunto com atuação em campo atrelada a uma dada atividade envolvendo acidentes, formalizações em um modelo conceitual, realização de inferências para avaliar os cenários e explorar os arcabouços possíveis.

### 3.1 Revisão Exploratória e Análise de Campo

Para realizar um levantamento dos conceitos que estão atrelados a atividades envolvendo acidentes, se faz necessário analisar uma atividade onde esse tipo de situação acontece. Para isso, estudou-se manutenção em linha viva onde eletricitas executam atividades preventivas e corretivas em equipamentos elétricos energizados. Esses profissionais são submetidos a riscos de serem eletrocutados e, conseqüentemente, mortos.

A análise da atividade se deu por meio do levantamento dos seguintes pontos:

1. Estudo dos manuais técnicos privativos de uma companhia de energia, análise (teórica e prática) das ferramentas usadas na execução da atividade.
2. Conversas com profissionais que atuam diretamente na área de manutenção.
3. Entrevista com engenheiro de manutenção em linha viva (sobre os procedimentos de manutenção) de uma companhia de energia (A).
4. Acompanhamento de um procedimento de manutenção em linha viva (de uma subestação, verificando-se a real execução dos procedimentos).
5. Criação de textos descrevendo cenários de manutenção.
6. Participação em *workshops* para treinamentos sobre técnicas e procedimentos em linha viva.
7. Revisão bibliográfica exploratória em busca de textos e pesquisas sobre arcabouços que apresentam o potencial para representar esse tipo de situação.

A etapa referente ao item 2 se deu em diversos encontros com os profissionais da área de linha viva, tanto informalmente como em reuniões, com propósito de realizar troca de informações sobre os procedimentos de manutenção.

A etapa referente ao item 3 se deu por meio de uma entrevista que durou mais de duas horas com perguntas voltadas para o entendimento de diversos tipos de procedimentos de manutenção (questionário da entrevista no apêndice A). O engenheiro respondeu as perguntas de forma clara e didática diante de equipamentos de uma subestação. Toda entrevista foi gravada. A escuta do áudio da mesma foi realizada diversas vezes para a construção do modelo.

A etapa referente ao item 4 se deu através de observações em campo do autor em relação a um dado procedimento em linha viva (troca de um isolador de pedestal) que se deu em uma subestação na cidade de Maringá - Paraná. Durante o ato da manutenção, o autor acompanhou o procedimento de dentro da subestação (em uma região de segurança) a uma distância de aproximadamente 15 metros (adotando procedimentos e equipamentos de proteção individual).

A etapa referente ao item 7 se deu através de pesquisas em bases de dados especializadas em publicação acadêmica nos temas atrelados aos aspectos que foram identificados nas etapas anteriores sendo essas correlatas à computação. Esses aspectos são concepções relevantes para o entendimento do cenário em análise como, por exemplo, o fato de que os profissionais se orientam por meio de uma organização ou que a ocorrência de acidentes acontece por descuidos e entre muitos outros aspectos que serão melhor trabalhados ao longo do texto. O termo 'correlatas a computação' faz referência a modelos computacionais que são apropriados para esse aspecto. Uma maneira de compreender esse contexto consiste em refletir sobre o aspecto que os trabalhadores atuam de forma organizada por meio de papéis, hierarquias a fim de cumprir missões e objetivos. Tendo em vista todos esses pontos, um arcabouço razoável para resolver essa situação é o MOISE+.

Ainda sobre a etapa 7 é importante frisar que os textos foram analisados em função da relevância, sendo verificadas as referências a fim de encontrar outros textos com potencial para contribuir a causa desse estudo.

### 3.2 Formalização de um Modelo Conceitual

Uma vez identificados quais são esses modelos (esses arcabouços são apresentados na Seção 4.1), e uma vez realizada as observações em campo, foram formulados os conceitos e as relações para o estudo em análise. Feito isso, ocorreu a etapa da adaptação, onde essas categorias foram redefinidas em estruturas conceituais mais específicas com objetivo de construir um vocabulário especializado para as condições de interesse desse estudo. Como resultado desse processo, obteve-se uma lista de conceitos e suas relações específicas para representar cenários delineados pelos resultados das observações.

A próxima etapa consistiu em escrever esses conceitos em algum tipo de formalismo. Nesse estudo, optou-se por usar a teoria de conjuntos e lógica de predicados. Isso possibilitou estruturar o modelo em voga numa linguagem formal onde cada parte (conjunto, elemento e predicados) é um vocábulo com uma semântica clara.

Os conjuntos são usados para representar um certo conceito. Os elementos de um conjunto representam os objetos atrelados ao conceito. Por exemplo, supondo que uma atividade de manutenção utilize os seguintes equipamentos de proteção individual: *óculos*, *bota*, *capacete*. Nesta situação, o conceito de equipamentos de proteção individual é representado pelo conjunto  $C$  e os modelos são elementos do mesmo. Portanto, numa linguagem matemática formal tem-se a seguinte situação  $C = \{oculos, bota, capacete\}$ .

Dentro do conceito matemático, uma relação é uma correspondência entre elementos de conjuntos não vazio, sendo dada por  $R \subseteq A \times B = \{(a, b) | a \in A \wedge b \in B\}$ . A Lógica de Predicados foi usada para representar essas relações.

A *UML* também é uma ferramenta que foi usada para criar representações do modelo, com o propósito de: apresentar a perspectiva global do modelo, definir com maior precisão os critérios existenciais (agregação, composição), tornar o processo de apresentação mais didático e aproxima-lo de mecanismos de implementação (ex. linguagens de programação).

Ainda nessa parte da pesquisa construíram-se as regras que definem como se dá a transição de estados do sistema. Os raciocínios portanto, são definidos com base nessas regras que, em linguagem natural, correspondem a *Se ..., Então*. Em termos formais, foi utilizado o formalismo lógico de implicabilidade dado por  $\rightarrow$ . Também foram adotados os seguintes critérios para construir essas regras: elaboração de raciocínios práticos, análise das regras dos outros modelos e verificação da semântica dos vocábulos.

### 3.3 Explorar os Arcabouços Possíveis

Com a formulação do modelo conceito e com o desenvolvimento de inferências que correspondem em partes a realidade, obteve-se um modelo conceitual que permitiu estabelecer correspondências com os arcabouços disponíveis. Isso foi feito por intermédio de uma análise da estrutura conceitual desses modelos, a fim de averiguar a correspondência dos mesmos com o modelo resultante.

Essa análise da estrutura conceitual consiste em entender a linguagem na qual o modelo é formulado, compreender as regras de sintaxe e averiguar a semântica de todos os elementos. Tendo em vista o fato de que o modelo conceitual possui as suas estruturas semânticas alicerçadas em uma literatura acadêmica, que é um ponto em comum com esses arcabouços, a equivalência semântica se torna algo difícil de ser feito (existem alguns modelos onde averiguar essa correspondência semântica é algo relativamente complicado de ser feito, contudo essas questões são tratadas e justificadas na discussão).

Como o modelo conceitual permitiu a formulação de regras e raciocínios, a análise comparativa também verificou a correspondência que o modelo conceitual tem com os arcabouços no que tange a conclusões de riscos e acidentes.

### 3.4 Aplicação em Estudo de Caso

Nessa parte da pesquisa foi utilizado o modelo conceitual para representar um dado cenário de manutenção. Optou-se por usar a manutenção em linha viva (em específico a troca do isolador de pedestal) tendo em vista o fato de que foi um caso bastante estudado e investigado - Projeto RV2-Institutos Lactec-Copel ([SAMARA et al., 2018](#); [ROSENDO et al., 2011](#); [GRUBER et al., 2018](#)). Esse processo se deu por definir esses procedimentos de manutenção na estrutura do modelo conceitual e realizar as inferências com a finalidade de verificar se os cenários que podem ser derivados dessas regras correspondem com à realidade dos fatos.

## 4 Resultados

Esse capítulo tem como finalidade exibir os resultados obtidos ao longo do estudo dessa pesquisa. Os resultados presentes na Seção 4.1 são referentes a etapa metodológica descrita na Seção 3.1, já os resultados presentes em 4.2 são consequências da descrição metodológica apresentada em 3.2. A Seção 4.3 mostra a aplicação deste modelo para um caso simples com a finalidade de introduzir didaticamente o leitor à estrutura do modelo.

### 4.1 Revisão Bibliográfica Exploratória e Análise de Campo

As observações de campo, análise dos manuais, entrevista com Engenheiro da área e conversas com diversas categorias de profissionais que atuam no segmento da manutenção em linha viva, apontam que o modelo conceitual deve conter, em sua estrutura, elementos com a capacidade de representar os itens da lista que se segue:

1. Os trabalhadores que executam os procedimentos.
2. Os diferentes papéis (ou funções) desses trabalhadores.
3. As ferramentas usadas pelos profissionais.
4. Os equipamentos que são submetidos a manutenção.
5. As interações entre todo tipo de entidade tais como trabalhadores, ferramentas e equipamentos.
6. As etapas das tarefas que devem ser finalizadas.
7. As relações entre todas as entidades e interações com as tarefas (ex. Quais são as entidades e interações que definem uma dada atividade como alcançada?)
8. Averiguar a incerteza presente em certos processos e equipamentos que podem contribuir como causas para a ocorrência de um acidente.
9. Medidas que são tomadas pelos profissionais para lidar com essas incertezas.
10. Verificar cenários onde o trabalhador execute uma dada atividade que não corresponde com o esperado pelas práticas seguras .
11. Na ocorrência de 10 verificar os riscos gerados a todos os envolvidos na situação.
12. Avaliar a ocorrência de incertezas e possibilidades do risco de um determinado acidente ocorrer a(s) um (alguns) profissional(is).

O resultado da revisão exploratória dos arcabouços que apresentam potencial para representar os elementos contidos na lista anterior são apresentados na lista que se segue:

1. MOISE+ (HÜBNER; SICHTMAN; BOISSIER, 2002a)
2. Modelo de Agentes Normativos de Dastani (DASTANI et al., 2009)
3. V3S (BAROT et al., 2013)
4. NormMAS (CHANG; MENEGUZZI, 2016)

Com os estudos realizados conclui-se que as atividades podem ser expressas em termos de objetivos com seus pré-requisitos e efeitos. Como a preocupação do modelo é representar falhas nas atividades, os pré-requisitos que não estão presentes quando se inicia a perseguição de um objetivo levam a violações que podem causar consequências no objetivo atual ou em subsequentes.

## 4.2 Estrutura Conceitual

Essa parte do texto apresenta a estrutura final do modelo conceitual que é definida em termos de módulos (exibe informações sobre os módulos e os conceitos neles contidos), Predicados (apresenta os predicados e justifica a sua existência) e regras (exibe as regras e explica porque sua existência dentro do modelo é necessária). Cada componente desses será exibido em termos de uma subseção.

### 4.2.1 Módulos

Os conjuntos que representam os conceitos são organizados em módulos. Isso ocorre porque muitos conceitos apresentam similaridades, sendo razoável conceber uma taxonomia desses conjuntos organizando-os em conjuntos maiores. A Figura 6 apresenta a estrutura de módulos (a fim de evitar poluição visual, as relações serão apresentadas em outra Figura).

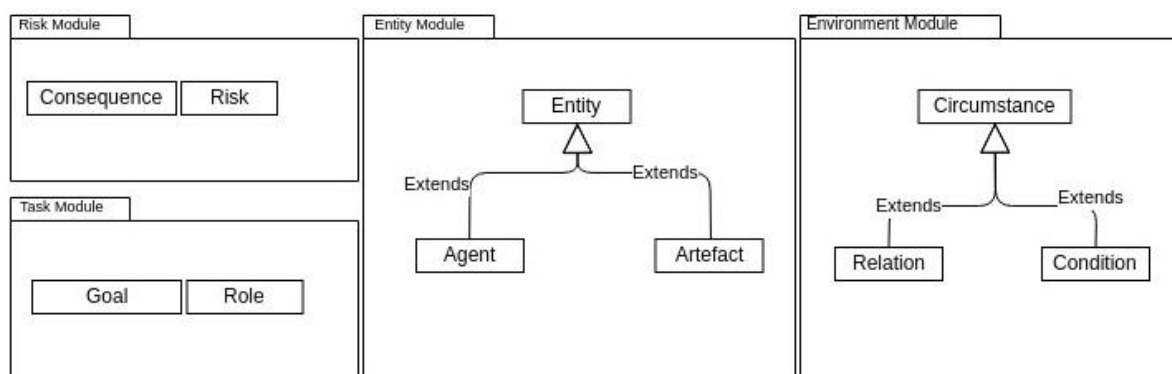


Figura 6 – A estrutura geral das classes do modelo

Assim sendo, assumindo que existe  $\Omega_{Model}$  (um conjunto global onde todos os outros conjuntos do modelo estão contidos nele), os módulos são representados da seguinte maneira:

$$\Omega_{Model} = \{M_{Risk}, M_{Task}, M_{Entity}, M_{Environment}\} \quad (4.1)$$

O mundo que este modelo pretende representar trabalha com o fato de que tanto agentes como artefatos possuem algumas propriedades em comum, que é: existem, ocupam lugar no espaço, estão sujeitos ao tempo, apresentam estados e participam de processos. Essa premissa possui os seus fundamentos alicerçados na 2.1 e 2.2 e será demonstrado com maior rigor no texto que se segue. Tendo em vista a ocorrência de certos conceitos necessários para lidar com essas questões, se fez necessário definir um módulo de entidades para agrupá-los em uma estrutura única. Esse módulo é composto pelos seguintes conceitos:

$$M_{Entity} = \{Entity\} \quad (4.2)$$

**Entity** - o termo entidade é sujeito a profundos debates filosóficos, porém neste texto o termo é usado para referenciar uma “coisa” que pode ser identificada, como uma pessoa, companhia ou um evento (CHEN, 1976). É de conhecimento que as propriedades anteriormente mencionadas caracterizam as “coisas” que podem ser identificadas, logo são entidades. É digno de nota a existência de entidades que não se adequam a todas essas propriedades. Contudo, essas propriedades fazem referência ao que se caracteriza por entidade, respeitando o conceito padrão (CHEN, 1976) e restringindo para o escopo deste modelo. Isso contempla tanto os agentes como os artefatos, como fica claro na Relação 4.3. O texto a seguir demonstra como essas propriedades se aplicam a agentes e a artefatos (sendo isso demonstrado, logo demonstrado-se que são entidades).

*Ocupa lugar no espaço, estão sujeitos ao tempo* - como definido em 2.1 e 2.2, ambos são situados em ambientes. Isso possibilita inferir que se faça necessário a presença de um conceito que se apresente como uma propriedade de estado para agentes e artefatos. Um ambiente, no contexto onde os agentes e artefatos são usados para representar atividades das pessoas, condiz com a relação de espaço e tempo.

*Participa de Processos* - processos podem constituir entidades, bem como entidades necessariamente constituem processos por intermédio da ação que aquelas manifestam nesses. No que se verifica ao primeiro caso, é possível usar o ser humano como exemplo - onde a entidade ser humano é formulada por uma série de processos bio-químicos. Sobre o segundo caso, relações climáticas exemplificam isso, onde a água é uma entidade presente em processos termodinâmicos.



*Apresenta estados* - o fato de que artefatos bem como agentes apresentam atributos (que podem mudar e podem assumir diferentes valores no que tange aos eventos externos e internos), então ambos também apresentam a concepção de estados (sendo esse termo usado diretamente em certos pontos dos textos presentes tanto em 2.1 2.2).

$$\{Agent \cup Artefact\} \subset Entity \quad (4.3)$$

**Agent** - esse estudo adota a definição de agentes presentes no primeiro parágrafo da Seção 2.1. Isso implica em entidades autônomas, ou seja, que apresenta a capacidade de agir por si mesma quando diante de condições onde isso é necessário. A seção 2.1, apresenta o conceito de agentes inteligentes e esse mesmo conceito é adotado neste modelo.

Não é preocupação deste estudo, delimitar as representações do agente, bem como definir algoritmos para verificar como se dão às relações de tomada de decisão dos mesmos. Assim sendo, fica em aberto para o modelador definir como acontece os processos de tomada de decisão, estados internos e modelos de representação que serão usados para definir o comportamento do agente.

**Artefact** - são entidades que existem para que os agentes possam cumprir com os seus objetivos e que apresentam interface de uso, instruções de operação, funcionalidade e estrutura-comportamento. Essas entidades não são orientados à objetivos e não apresentam capacidade de comunicação como definido na Seção 2.2. Predicados que contemplam esses aspectos do artefato serão apresentados mais adiante ao decorrer do texto.

Os agentes são autônomos e orientados à objetivos sendo esses dois elementos descaracterizantes do que se define por artefato. Logo, apesar de agentes e artefatos serem entidades, não é possível existir um agente que seja artefato ou um artefato que seja agente, o que é dado pela relação presente na Expressão 4.4.

$$Agent \cap Artefact = \{\emptyset\} \quad (4.4)$$

Para tornar a apresentação de certos conceitos mais didática, o autor desenvolveu um exemplo intitulado por: **Exemplo da Redação**, que é o seguinte cenário: “O Professor Aristóteles definiu uma atividade: escrever uma redação sobre o livro Metafísica. Para isso, o aluno Alexandre o Grande deve escrever um texto, deve ler o livro sobre o tópico em definido, deve pegar uma folha, deve pegar um lápis e escrever a redação”.

**O Módulo de Atividades** - *Task Module* representado por  $M_{Task}$  condiz com os conceitos relacionados aos objetivos que devem ser atingidos, bem como aos papéis que são assumidos pelos agentes.

$$M_{Task} = \{Goal, Role\} \quad (4.5)$$

**Goal** - faz referência aos objetivos que devem ser atingidos pelos agentes. Os fundamentos semânticos deste conjunto estão presentes na Seção 2.3 mais especificamente na subseção 2.3.2. Neste modelo, um objetivo é descrito em termos de entidades relações e condições necessárias para que uma dada atividade possa ser dada como concluída.

**Role** - apresenta o papel que um agente pode adotar dentro de um SMA. Esse conceito também é importado no MOISE+ 2.3.2 e define as relações deonticas entre os agentes e os objetivos. Para exemplificar, pode-se considerar o **Exemplo da Redação** onde existem dois agentes  $Agent = \{aristoteles, alexandre\}$ , existem dois papéis  $Role = \{professor, aluno\}$ . Neste caso, o agente *aristoteles* é o *professor* e o agente *alexandre* é o *aluno*.

**O Módulo de Ambiente** - *Environment Module* - consiste em conjuntos que representam relações e condições ambientes, que são:

$$M_{Environment} = \{Circumstance\} \quad (4.6)$$

**Relation** - uma entidade estabelece relações com outras entidades ao seu redor (CHEN, 1976). No modelo proposto neste texto. O autor optou por um conjunto para representar os relacionamentos entre as entidades que possibilite identifica-los. Isso facilita o desenvolvimento de raciocínios. O uso dos relacionamentos podem ser exemplificado por meio do **Exemplo da Redação**. Ao definir uma atividade, o professor Aristóteles, que é uma entidade, estabeleceu uma relação com o seu aluno Alexandre, representado aqui por *relAristotelesAlexandre*. Para cumprir essa tarefa o aluno precisou ler o livro - *relAlexandreLivro*, pegar uma folha - *relAlexandreFolha*, pegar um lápis *relAlexandreLapis* e escrever a redação, o que implica em uma relação entre lápis e folha *relLapisFolha*. Portanto, o conjunto de relacionamentos se dá da seguinte maneira:

$$M_{Environment} = \{relAristotelesAlexandre, relAlexandreFolha, relAlexandreLivro, relAlexandreLapis, relLapisFolha\}$$

Obviamente, cada entidade do grupo **Relation** tem um vínculo com elementos do grupo *Entity*, como por exemplo *relAlexandreFolha* apresenta um vínculo com as entidades *Alexandre* e *Folha*. Há um predicado que trata disto e será exibido posteriormente. **Condition** - Esse conjunto representa as condições que devem ser mantidas para que um objetivo possa ser alcançado. Tendo em vista certas relações de predicado, que serão apresentados com maior riqueza de detalhes mais adiante, se faz necessário definir abstração de alto nível entre **Condition** e **Relation**. Assim sendo, o autor assume a existência de um conjunto **Circumstance** que é dado pela seguinte relação:

$$Circumstance = \{Relation, Condition\} | Relation \cap Condition = \{\emptyset\} \quad (4.7)$$

**Módulo de Risco** - *Risk Module* contém conjuntos que correspondem a conceitos relacionados a temática da segurança. O módulo de risco é dado pela relação que se segue:

$$M_{Risk} = \{Risk, Consequence\} \quad (4.8)$$

**Risk** - Na seção 2.5 o termo risco é usado para referenciar a um evento que apresenta um potencial de ocorrer, e que gera consequências negativas às pessoas associadas quando acontece. Para exemplificar, pode-se considerar uma condição onde um eletricista está trocando um disjuntor de um quadro elétrico. Nesse processo, o eletricista está sujeito ao risco de ser eletrocutado. Essas consequências negativas também são representadas por um conjunto, o qual é denominado **Consequence**. O uso deste conjunto pode ser apresentado utilizando esse mesmo exemplo do eletricista, pois a consequência de se submeter a um evento desses implica morte (nem sempre é assim, mas para efeitos didáticos pode-se considerar que o quadro elétrico é de certa potência que a morte é certa para o profissional que for eletrocutado).

#### 4.2.2 Predicados

O predicado  $possEntityRel(r_l, e_i, e_k) | r_l \in Relation \wedge e_i, e_k \in Entity$  é usado para, identificar a relação entre duas entidades. Esse predicado se lê da seguinte forma: O relacionamento  $r_l$  possui a entidade  $e_i$  e a entidade  $e_k$ . Para demonstrar como se dá o uso desse predicado pode-se considerar o **Exemplo da Redação**. A entidade *alexandre* apresenta uma relação com *folha* que é identificada como *relAlexandreFolha*. Portanto, com o predicado, a representação fica:  $possEntityRel(relAlexandreFolha, alexandre, folha)$

O predicado  $adoptsRole(ag_n, \rho_m) | ag_n \in Agent \wedge \rho_m \in Role$  que tem sua origem nos estudos do *MOISE+* onde cada agente tem uma função dentro do contexto do *SMA*. Esse predicado se lê da seguinte forma: o agente  $ag_n$  tem um papel  $\rho_m$ . Usando o **Exemplo da Redação** tem-se o seguinte:  $adoptsRole(artistoteles, professor)$ .

O predicado  $hasObligation(\rho_m, g_j) | \rho_m \in Role \wedge g_j \in Goal$  tem suas origens nos estudos da lógica deôntica também presentes no modelo *MOISE+*. Esse predicado pode ser lido da seguinte maneira: o agente que assumir o papel  $\rho_m$  tem a obrigação de concluir o objetivo  $g_j$ . No exemplo padrão deste texto (**Exemplo da Redação**), o primeiro objetivo é uma obrigação do professor, portanto:  $hasObligation(professor, g_0)$

O predicado  $hasPermission(\rho_m, g_j) | \rho_m \in Role \wedge g_j \in Goal$  também está associado aos estudos da lógica deôntica relacionada ao *MOISE+*. A leitura se dá desta maneira: o agente  $\rho_m$  que tem a permissão de concluir o objetivo  $g_j$ . Usando o exemplo padrão como base onde *professor* tem a obrigação de executar  $g_0$ , então também tem permissão para isso (será melhor explicado em uma regra que será apresentada mais tarde). Portanto:  $hasPermission(professor, g_0)$ .

O predicado  $instanceOfCond(circ_k) | circ_k \in Condition$  informa que  $circ_k$  é uma condição, ou seja, pertence ao conjunto *Conditions*. A existência desse predicado é necessária porque

alguns raciocínios de violação necessitam verificar se o elemento em análise é uma condição. A mesma situação acontece para as relações. Assim sendo o predicado  $instanceOfRel(r_k)|r_k \in Relation$  também deve fazer parte da estrutura do modelo.

O predicado  $reached(g_k)|g_k \in Goal$  define que todos os agentes, que eram obrigados a alcançar o objetivo  $g_k$ , concluíram. A existência desse predicado se dá devido ao fato de que em certos raciocínios é necessário identificar que um certo objetivo foi atingido.

O predicado  $stopped(g_n, ag_m)|g_n \in Goal \wedge ag_m \in Agent$  apresenta a seguinte leitura: toda a atividade foi encerrada com o objetivo  $g_n$  por uma ação associada ao agente  $ag_m$ . Para certos raciocínios se faz necessário identificar o encerramento das atividades como um todo e por quem isso aconteceu. Em certos casos não há a necessidade de identificar o agente  $ag_m$ . Para isso há uma outra versão deste mesmo predicado escrito da seguinte forma:  $stopped(g_n)|g_n \in Goal$  e sua semântica indica que o objetivo  $g_n$  teve sua execução encerrada.

O predicado  $nextGoal(g_i, g_j)|g_i, g_j \in Goal$  possui a seguinte semântica: O objetivo  $g_i$  tem um próximo objetivo que é  $g_j$ . Sua necessidade advém do fato de que este modelo trabalha os mesmos conceitos presentes em *MOISE+* porém sob uma abordagem diferente. Em vez de usar uma estrutura de super-sub objetivos e definindo operadores de série e paralelo, os objetivos não possuem estruturas, e suas relações se dão por um objetivo apontando para o próximo. Então, para exemplificar pode-se considerar uma atividade descrita por quatro objetivos, sendo que  $g_0$  é pré-requisito para  $g_1$  e  $g_2$ . Em contrapartida  $g_3$  só começa a ser atingido depois da finalização de  $g_1$  e  $g_2$ . Assim sendo, na linguagem proposta neste estudo, esse problema é escrito da seguinte forma:  $nextGoal(g_0, g_1), nextGoal(g_0, g_2), nextGoal(g_1, g_3), nextGoal(g_2, g_3)$ .

O predicado  $requiresCirc(goal_i, circ_j)|goal \in Goal \wedge circ_j \in Circumstance \wedge Relation \subset Circumstance \wedge Condition \subset Circumstance$  é lido da seguinte forma: para que o objetivo  $goal_i$  possa ser alcançado é necessário a circunstância  $circ_j$ . Essa circunstância pode ser tanto relações  $r_j \in Relation$  ou pode ser condições  $c_k \in Condition$ .

O predicado  $requiresEntity(goal_i, e_j)|g_i \in Goal \wedge e_j \in Entity$  é lido da seguinte forma: para que o objetivo  $g_i$  seja atingido, as entidades  $e_j$  devem estar presentes no instante em que  $g_i$  estiver sendo alcançado. O propósito deste predicado reside na necessidade de identificar quais são as entidades que devem estar presentes para que um dado objetivo possa ser executado. No **Exemplo da Redação**, um objetivo  $g_2$  só pode ser atingido se as entidades  $\{aluno, folha, lapis\}$  estiverem presentes no momento em que  $g_2$  estiver sendo alcançado. Esse cenário é representado da seguinte forma:  $requiresEntity(g_2, aluno), requiresEntity(g_2, folha), requiresEnt$

O predicado  $isPresent(circ_i)|circ_i \in Circumstance$  retorna como verdade de  $circ_i$  está presente no momento em que este predicado é invocado (retorna falso para o contrário). Em alguns raciocínios é de importância verificar se um elemento está presente durante a tentativa de um dado agente executar algum objetivo.

O predicado  $starts(ag_i, g_j)|ag_i \in Agent \wedge g_j \in Goal$  é lido da seguinte forma: um agente

$ag_i$  está tentando atingir o objetivo  $g_j$ . Para algumas situações é de crucial importância identificar quando um agente está tentando atingir um objetivo, sendo necessário a existência de um predicado apenas para esse propósito. Para exemplificar, pode-se considerar o exemplo da redação. Para que o professor *Aristoteles* possa alcançar o objetivo  $g_0$ , é necessário a existência de uma tentativa. Neste modelo, essa situação é representada da seguinte maneira:  $starts(aristoteles, g_0)$ . Seguindo a linha desse predicado, há também o  $ableReach(ag_i, g_j) | ag_i \in Agent \wedge g_j \in Goal$  cuja semântica expressa que o agente  $ag_i$  está habilitado a tentar buscar o objetivo  $g_j$ . Contudo, não significa que o agente fará o correspondente de  $starts(ag_i, g_j)$ . O que definirá a transição de  $ableReach(ag_i, g_j)$  para  $starts(ag_i, g_j)$  são os estados internos do agente. Contudo, não é do interesse deste estudo aprofundar na dinâmica do agente em si, deixando esse processo em aberto para o programador decidir como resolverá essa questão.

O predicado  $conditionViol(ag_i, g_j, c_k) | ag_i \in Agent \wedge g_j \in Goal \wedge c_k \in Condition$  deve ser lido da seguinte maneira: o agente  $ag_i$  comete uma violação de condição no objetivo  $g_j$  por tentar realizar uma determinada atividade em que a condição  $c_k$  era essencial porém não estava presente. Os fundamentos deste predicado estão vinculados com a Seção 2.4. No modelo em 2.4 para representar aspectos normativos dos agentes, a regra do tipo *Count-as* apresenta quais são as circunstâncias que ocasionam uma violação. Esse predicado cumpre esse propósito para o conjunto **Conditions**. Para exemplificar, no **Exemplo da Redação**, se o professor Aristóteles lecionar sem que haja luz suficiente para isso, então ele cometeu uma violação de condição caracterizada da seguinte maneira:  $conditionViol(aristoteles, g_0, luz)$

O predicado  $relationViol(ag_i, g_j, r_k) | ag_i \in Agent \wedge g_j \in Goal \wedge r_k \in Relation$  possui a mesma situação presente em  $conditionViol(ag_i, g_j, c_k)$  contudo o foco diz respeito aos relacionamentos. A leitura se dá desta forma: o agente  $ag_i$  pratica uma violação de relação no objetivo  $g_j$  por executar a atividade sem que a relação  $r_k$  esteja presente. O uso deste predicado por ser feito considerando o exemplo em análise com a adição de uma breve descrição de uma situação que possa ocorrer: a ponta do lápis que Alexandre tenta usar para escrever a redação está quebrada. Portanto, a relação *relLapisFolha* não pode ser feita. Se o agente *Alexandre* alcançar o objetivo  $g_2$  sem ter as circunstâncias necessárias para isso, então comete uma violação de relação sendo escrito da seguinte forma:  $relationViol(alexandre, g_2, r_2)$ .

O predicado  $entityViol(ag_i, g_j, e_k) | ag_i \in Agent \wedge g_j \in Goal \wedge e_k \in Entity$  advém das mesmas situações dos dois predicados acima. A leitura deste predicado se dá da seguinte forma: O agente  $ag_i$  cometeu uma violação por tentar alcançar o objetivo  $g_j$  sem que a entidade  $e_k$  esteja presente. No exemplo padrão o uso deste predicado consiste em Alexandre tentar executar  $g_2$  sem ter a entidade lápis o que resulta no seguinte:  $entityViol(alexandre, g_2, lapis)$ .

O predicado  $hasRisk(crts_i, risk_j, cs_k) | crts_i \in Circumstance \wedge risk_j \in Risk \wedge cs_k \in Consequence$  é baseado nos estudos presentes na seção 2.5 e tem a finalidade de definir os riscos associados ao tentar executar alguma atividade sem que  $c_k$  ou  $r_k$  esteja presente. Portanto, a leitura deste predicado é dada da seguinte forma: a ocorrência de uma violação onde  $crts$  ocasiona em um evento

associado ao  $risk_j$  com a consequência  $cs_k$ . Ao explicar sobre o conjunto **Consequence**, foi dado um exemplo sobre um eletricitista que tem o potencial de ser eletrocutado. Para esse exemplo, o uso deste predicado apresenta o seguinte formato: *hasRisk(relFerramentaIsolanteBarramento, eletro*

O predicado *possOfNegConseqFor( $r_l$ )* |  $r_l \in Relation$  tem seus fundamentos associados ao estudo da lógica modal, presente na Seção 2.6, no operador  $\Diamond$  cuja semântica denota possibilidade. Contudo, neste estudo esse termo apresenta o seguinte conceito semântico: há a possibilidade de acontecer um evento ruim associado ao risco  $r_l$  vinculado ao agente que está associado a essa relação, mesmo que esse agente não tenha cometido nenhum erro durante o procedimento. Esse predicado tem por finalidade representar situações onde um evento ruim acontece, não pelo erro do profissional diretamente associado a situação, mas sim por outras cadeias causais complexas de serem identificadas e justamente por isso são abstraídas por conceitos de aleatoriedades, idem possibilidades. Para exemplificar pode-se considerar a situação onde um eletricitista de linha viva usa um bastão isolante para acessar um barramento altamente energizado. Contudo, esse bastão pode estar com isolamento comprometido. Testes que devem ser feitos antes de fazer uso de uma ferramenta podem eliminar qualquer possibilidade de que os riscos venham a se tornar eventos reais, pois se o bastão em questão estiver em bom estado, então o eletricitista não se envolverá em um acidente por esse fator. Se o bastão estiver em mal estado - isso será identificado e a ferramenta será adequadamente substituída. Contudo, considerando um cenário onde por negligência de profissionais a medição não é feita, surge uma possibilidade do isolamento estar comprometido. Essa situação torna verdadeiro o seguinte predicado *possOfNegConseqFor(relBastaoBarramento)*.

O predicado *affectsRels( $r_k, r_n$ )* |  $r_k, r_n \in Relation$  trabalha em conjunto *possOfNegConseqFor( $r_l$ )*. Esse predicado se lê da seguinte forma: se  $r_k$  não foi realizado, ou se for realizado de forma inapropriada, isso afeta  $r_n$  tornando verdade *possOfNegConseqFor( $r_n$ )*. Ambos são importantes em raciocínios onde se deseja mostrar que a não execução de uma relação não gera consequências negativas imediatas a ninguém, mas resulta em consequências futuras inclusive sobre pessoas que não compartilham da mesma situação. No exemplo do eletricitista, a relação *relBastaoBarramento* é afetada pela não execução da relação *relBastaoMedidor* (que define a relação entre o aparelho medidor de corrente de fuga e o bastão isolante). Esse exemplo é escrito da seguinte maneira: *affectsRels(relBastaoMedidor, relBastaoBarramento)*

O predicado *negConseqFor( $g_k, ag_i, risk_k, cs_m$ )* |  $g_k \in Goal \wedge ag_i \in Agent \wedge cs_m \in Condition$  pode ser lido da seguinte maneira: ocorreu um evento ruim no objetivo  $g_k$  associado ao agente  $ag_i$  e associado ao risco  $risk_k$  que atribuiu ao agente  $ag_i$  as consequências  $cs_m$ . Esse predicado tem por finalidade traduzir semanticamente as consequências ruins sobre alguém quando ocorre o pior caso. Sobre o exemplo do eletricitista que acessa um barramento de alta tensão, esse predicado é escrito da seguinte forma: *negConseqFor(gacessoBarramento, eletricitista, eletrocutado, morte)*. Associado a isso há o predicado *happensNegConseqFor( $r_m$ )* |  $r_m \in Relation$  cujo propósito define que um evento ruim aconteceu no relacionamento  $r_m$ .

O predicado  $lastGoal(g_i, \rho_m) | g_i \in Goal \wedge \rho_m \in Role$  apresenta um último objetivo  $g_i$  que deve ser alcançado por agentes com determinado papel  $\rho_m$ . Esse predicado tem sua existência justificada em certos raciocínios, que serão demonstrados mais adiante, onde esse tipo de informação é relevante.

### 4.2.3 Diagrama de Classes

Diferente da Figura 6, o foco da Figura 7 consiste em apresentar como se dá a estrutura de classes e dos relacionamentos. Ambas situações poderiam ser representadas em uma única Figura, contudo decidiu-se por seccionar em duas, a fim de tornar o processo mais didático. Por esse mesmo motivo não está apresentado neste *UML* todas as classes e propriedades.

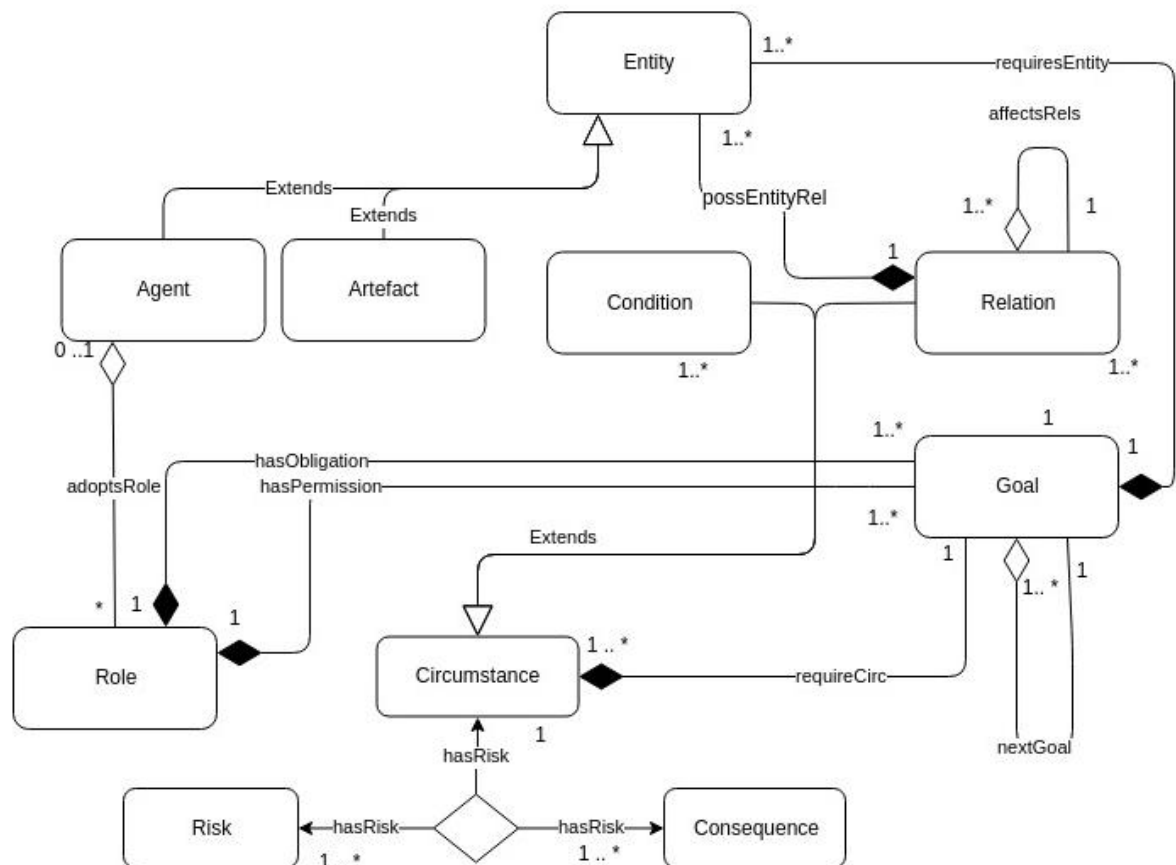


Figura 7 – Diagrama de classes do Modelo

Uma vantagem deste tipo de diagrama em relação a representação por conjuntos, consiste na ocorrência de uma sintaxe específica para tratar dois pontos relevantes dentro do contexto computacional que são: cardinalidade e associações existenciais. Um dos predicados interessantes de serem analisados, neste contexto, é *adoptsRole* que define um relacionamento fracamente agregada entre *Agent* e *Role*. Assim, dentro do escopo deste modelo, um agente pode existir sem ter um papel, portanto este não é um critério necessário para definir aquele. A cardinalidade se justifica tendo como base o fato de que um agente pode ter um ou mais papéis.



As relações *hasObligation*, *hasPermission* se dão por meio de agregações fortes tendo em vista que não há sentido para um papel  $\rho$  existir sem que esteja vinculado a menos um objetivo. Como um papel se relaciona com diversos objetivos, os engenheiros adotaram a cardinalidade de 1 para 1..\*.

Em *UML* a questão de conjunto-subconjunto entre *Circumstance* com *Relation* e com *Condition* é definida por meio de classes que possuem esses mesmos nomes. Na *UML*, as classes *Relation* e *Condition* são extensões da classe *Circumstance*. Dada essa situação, é possível representar a relação *hasRisk* que ocorre entre *Circumstance*, *Consequence* e *Risk* e isso é por meio de uma associação ternária entre essas classes.

Um objetivo não pode ser definido sem saber quais são as entidades *Entity*, relações *Relation* e consequências *Consequence* necessárias para que tal objetivo seja alcançado. Por isso os predicados *requiresCirc* e *requiresEntity* estabelecem composição forte de suas respectivas classes com *Goal*. Como um objetivo pode apontar para diversas instâncias dessas classes, o autor optou - para cada uma das relações - trabalhar com a cardinalidade 1 – 1..\*.

No modelo proposto *Relation* deve estar relacionada com duas entidades. Tendo em vista o fato de que não é possível definir uma relação sem pelo menos uma entidade, então associação *possEntityRel* entre ambas as classes se caracteriza por uma composição forte de cardinalidade 1 – 1..\*.

A classe *Goal* possui uma relação consigo mesma dada por *nextGoal*. Essa é uma agregação fraca, pois do contrário seria impossível haver uma única instância desta classe. Isso se deve ao fato de que a primeira instância necessitaria de uma instância de *Goal* para existir. Contudo, como não há um elemento de *Goal* antes do primeiro elemento de *Goal*, logo esse primeiro elemento não pode existir. Um objetivo poder ter como próximo um ou mais objetivos, justificando a ocorrência da cardinalidade 1..n.

O predicado *affectsRels*, por motivos similares a *nextGoal* deve ter agregação fraca. Como uma relação pode afetar uma ou mais, a cardinalidade adequada para essa circunstância é dada por 1 - 1..\*

#### 4.2.4 Regras

A Regra 4.9 tem os fundamentos teóricos na lógica deôntica e em modelos como *MOISE+*. Assim sendo, todas as relações de obrigação implicam relações de permissão. O que essa regra determina consiste no fato de que se um agente  $a_j$  é obrigado a trabalhar sob o objetivo  $g_j$ , então esse agente também tem a permissão de trabalhar sobre o objetivo  $g_j$

$$\begin{aligned} hasObligation(\rho_m, g_j) \rightarrow hasPermission(\rho_m, g_j), \\ \rho_m \in Role \wedge g_j \in Goal \end{aligned} \quad (4.9)$$



As regras 4.10, 4.11, e 4.12 são fundamentadas em Seção 2.4 onde o conceito do que pode ser feito é definido em termos das regras *Count-as*. Essas regras determinam quais são os elementos que resultam em violação. Inspirando-se nesse tipo de estrutura são trabalhadas as referidas regras.

A complexidade de estudo é extramente ampla, e com certeza existe mais tipos de violações do que as três consideradas a seguir, contudo optou-se por estudar essas violações porque são essenciais para os objetivos deste estudo. Outro questionamento que pode surgir consiste no por que definir três tipos de violações? Isso reside no fato de que essas violações resultam em consequências diferentes, por conta disto em um primeiro momento os engenheiros do modelo decidiram tratá-las em estruturas diferentes.

A explicação das regras será feita sempre analisando-se a semântica do predicado que é implicado em relação aos estudos pelos quais elas se fundamentam. Partindo desta premissa, o entendimento da relação 4.10 somente pode ser feito na ocorrência de uma investigação sobre quais são os elementos que semanticamente correspondem ao predicado  $conditionViol(ag_m, g_i, c_k)$ . O primeiro ponto reside em verificar quais são as condições necessárias de  $g_i$ . Quem tem essa finalidade é o predicado  $requiresCirc(g_i, c_k)$ . Contudo, saber todas as condições não são o suficientes, pois a violação acontece na ausência de uma condição  $c_k$  e isso deve ser verificado nesta relação de implicabilidade. Então se faz necessário considerar um predicado que analisa se  $c_k$  está presente no ato da manutenção, e é com esse propósito que  $isPresent(c_k)$  faz parte da relação. Contudo, as informações ficam desconstruídas se  $c_k$  não for uma instância de *Condition*, por isso é importante fazer essa análise também através do predicado  $instanceOfCond(c_k)$ . Esses são componentes essenciais, porém não são suficientes porque não consideram a condição do agente. Assim, afirmar sobre a ocorrência de uma violação de um agente sem considerar se ele está efetivamente tentando alcançar um objetivo consiste em desconsiderar a semântica daquilo que está sendo implicado. Isso é resolvido por considerar o termo  $starts(ag_m, g_i)$ .

$$\begin{aligned}
 &requiresCirc(g_i, c_k) \wedge \neg isPresent(c_k) \wedge instanceOfCond(c_k) \wedge starts(ag_m, g_i) \rightarrow \\
 &\hspace{15em} conditionViol(ag_m, g_i, c_k) \\
 &g_i \in Goal, c_k \in Condition, ag_m \in Agent \quad (4.10)
 \end{aligned}$$

O propósito da regra 4.10, quando definido em termos de linguagem natural tem a finalidade de exprimir o seguinte: se um agente tentar executar um determinado objetivo sem que existam todas as condições ambientes necessárias para isso, então esse agente comete uma violação de condição neste respectivo objetivo.

A regra 4.11 define as condições que resultam em uma violação de relação. O predicado  $relationViol(ag_m, g_i, r_k)$  considera que a violação se dá por um agente  $ag_m$  em um objetivo  $g_i$  na relação  $r_k$ . Portanto, para respeitar a semântica deste predicado se faz necessário considerar ao menos um termo que vincule o objetivo  $g_i$  com a relação  $r_k$ . Para esse propósito é que se

considera o termo  $requiresCirc(g_i, circ_k)$  pois define quais são as circunstâncias que devem estar presentes para que o objetivo  $g_i$  possa ser alcançado. Contudo, só isso não é o suficiente, pois se faz necessário analisar se  $r_k$  está contido em  $Relation$ . Isso se deve ao fato de que o predicado  $requiresCirc(g_i, circ_k)$  não permite saber se  $circ_k$  está contido em  $Relation$  ou se está contido em  $Condition$ . O predicado  $instanceOfRel(r_k)$  resolve essa situação. Outro fator atrelado e importante para que o predicado  $relationViol(ag_m, g_i, r_k)$  retorne verdade, reside em saber se o agente em sua tentativa de atingir  $g_i$  não executa  $r_k$  de forma apropriada. Por conta disso se faz necessário considerar o  $isPresent(r_k)$ . A semântica de  $relationViol$  só é conservada em sua inteireza se a presença do agente também for analisada. Para esse propósito é que se verifica a necessidade do uso de  $starts(ag_m, g_i)$  que deverá retornar se o agente está tentando alcançar o objetivo  $g_i$ .

$$\begin{aligned}
&requiresCirc(g_i, r_k) \wedge \neg isPresent(r_k) \wedge instanceOfRel(r_k) \wedge starts(ag_m, g_i) \rightarrow \\
&\quad relationViol(ag_m, g_i, r_k) \\
&g_i \in Goal, r_k \in Relation, ag_m \in Agent \quad (4.11)
\end{aligned}$$

Traduzindo a regra 4.11 para linguagem natural obtêm-se a seguinte expressão: se um agente tentar alcançar um certo objetivo sem que todas as relações necessárias para isso estejam presentes (considerando as relações do domínio dele, tal como manuseio de uma ferramenta específica, e considerando as relações que são independentes dele), então esse agente comete uma violação de relação.

A regra 4.12 tem o propósito de definir quais são as condições que resultam em uma violação de entidade. Como em outras situações, para cumprir com esse propósito é necessário que os fatores implicantes sejam correspondentes com  $entityViol(ag_m, g_i, e_k)$ . Para cumprir com essa finalidade, se faz necessário considerar o predicado  $requiresEntity(g_i, eg_n)$  (para avaliar as entidades  $e_k \in Entity$  que devem estar presentes a fim de cumprir com o objetivo  $g_i$ ),  $isPresent(e_k)$  (para verificar se a entidade  $e_k$  está ou não, presente no momento da execução) e  $starts(ag_m, g_i)$  (para avaliar se  $ag_m$  começou a tentar alcançar o objetivo  $g_i$ ). A semântica do predicado também considera o momento em que o agente está atuando sobre o objetivo  $g_i$ , por isso o predicado  $starts(ag_m, g_i)$  também é posto na relação de implicabilidade.

$$\begin{aligned}
&requiresEntity(g_i, eg_n) \wedge \neg isPresent(e_k) \wedge starts(ag_m, g_i) \rightarrow \\
&\quad entityViol(ag_m, g_i, e_k) \\
&g_i \in Goal, e_k \in Entity, ag_m \in Agent \quad (4.12)
\end{aligned}$$

Em termos de linguagem natural, a regra 4.12 se apresenta da seguinte forma: se um agente tentar alcançar um certo objetivo sem ter todas as entidades presentes para isso, então esse agente cometeu uma violação de entidade.

As regras 4.13 e 4.14 são inspiradas nos estudos presentes na Seção 2.4 onde as consequências de uma violação são definidas como sanções no que é denominado por *SanctionRule*. A estrutura dessas regras, em 2.4 e em (DASTANI et al., 2009) é dada como  $violation \rightarrow \dots$ . Contudo, este estudo leva em consideração não apenas o termo que se refere a violação, mas também as circunstâncias que são consideradas juntas, que neste caso advêm do predicado *hasRisk*. Assim como em 2.4, o modelo deste estudo define que uma sanção corresponde a uma penalidade que o agente deve pagar. Na estrutura da problemática em análise, a penalidade ocorre pelo fato do agente sofrer fisicamente os efeitos dos seus erros. Esse comportamento é dado pelo predicado  $negConseqFor(g_i, ag_m, risk_j, cs_m)$  cujo correspondente semântico define que o agente  $ag_m$  sofre o evento associado em  $risk_j$ , no objetivo  $g_i$  com a consequência  $cs_m$ . Se os engenheiros deste modelo considerarem apenas  $conditionViol(ag_m, g_i, c_k)$  para a relação 4.13 e  $relationViol(ag_m, g_i, r_k)$ , o correspondente semântico de  $negConseqFor(g_i, ag_m, risk_j, cs_m)$  é desrespeitado, não especificando  $risk_j, cs_m$ . Contudo, isso é resolvido por levar em consideração o predicado  $hasRisk(c_k, risk_j, cs_m)$  para 4.13 e o predicado  $hasRisk(r_k, risk_j, cs_m)$  para 4.14.

$$\begin{aligned}
& conditionViol(ag_m, g_i, c_k) \wedge hasRisk(c_k, risk_j, cs_m) \rightarrow \\
& \quad negConseqFor(g_i, ag_m, risk_j, cs_m) \\
& ag_m \in Agent, g_i \in Goal, c_k \in Condition, risk_k \in Risk, cs_m \in Consequence \quad (4.13)
\end{aligned}$$

Em termos de linguagem natural, a relação em 4.13 é definida da seguinte maneira: “uma violação de condição de um determinado agente, em um dado objetivo ocasiona em uma consequência ruim a ele. Essa consequência ruim está associada ao risco da condição violada”.

$$\begin{aligned}
& relationViol(ag_m, g_i, r_k) \wedge hasRisk(r_k, risk_j, cs_m) \rightarrow \\
& \quad negConseqFor(g_i, ag_m, risk_j, cs_m) \\
& ag_m \in Agent, g_i \in Goal, r_k \in Relation, risk_k \in Risk, cs_m \in Consequence \quad (4.14)
\end{aligned}$$

A regra 4.14, quando posta em linguagem natural é definida desta forma: “uma violação de relação de um determinado agente, em um dado objetivo resulta em uma consequência ruim a ele. Essa consequência está atrelada ao risco da relação violada”.

Neste estudo o termo *risco* deve ser analisado com muito cuidado, pois, dependendo do contexto, a complexidade deste termo é praticamente infinita e neste estudo a concepção deste termo se reduz a dois dos muitos possíveis usos. Neste modelo, risco é analisado como um evento que tem potencial de acontecer. Contudo, nas relações de implicação um dos usos do termo risco advêm de considerá-lo como evento que acontece apenas na ausência de uma dada condição ou de uma dada relação. Optou-se por essa tratativa ao estudar os conceitos presentes no referencial teórico na Seção 2.5 e ao analisar o estudo de caso (que será apresentado mais

tarde). Com base nestes estudos, verificou-se que acidentes acontecem porque profissionais tentam executar uma certa atividade sem ter as condições apropriadas para isso e é à essa circunstância sobre a qual o risco está associado (em (FADIER; GARZA; DIDELOT, 2003), isso é explicado visando a melhoria da eficiência e da produção). Por exemplo, para poder navegar em alto mar a fim de poder pescar, um barco pesqueiro deve ter a sua disposição uma determinada condição climática. Se a tripulação decidir por navegar sem a presença da condição climática apropriada, então o barco está submetido ao risco de naufragar sob as consequências de morte da tripulação inteira. Portanto é com essa semântica que as relações de implicação 4.13 e 4.14 empregam o conceito de risco.

Obviamente, existe a possibilidade do barco poder desbravar um mar sem as apropriadas condições e voltar para a terra a salvo. Contudo, considerar situações assim, apesar de serem interessantes, levam a um aprofundamento da complexidade deste modelo. Não que isso seja uma justificativa coerente para não se fazer isso, contudo - neste estudo o interesse reside em uma primeira versão que torne possível a modelagem de condições assim por meio de um vocabulário mais específico. Nesse sentido, decidiu-se por simplificar essa situação e considerar que toda a ação tomada por um agente sem que as condições necessárias estejam presentes, ou as relações apropriadas sejam feitas, resultam em penalidades associadas ao risco da ausência desses elementos.

Dentro do que condiz ao conceito de sanção que é tratado neste estudo, apenas as regras 4.13 e 4.14 são sanções. Isso se deve ao fato de que essas regras consideram que o equívoco do agente, gerou penalidades a ele mesmo. Apesar de levar em consideração predicados associados à violação, as demais regras não são consideradas como regras de sanção porque elas apresentam uma condição onde o comportamento inapropriado de um agente A, resulta em consequências ruins a outros agentes. Como o erro do agente A não recai sobre si, é um inequívoco, dentro do escopo deste estudo, afirmar que ele sofreu uma sanção por conta disto.

A regra 4.15 é usada com o propósito de demonstrar que uma dada violação em uma certa relação afeta outras relações. Muitas vezes o ato de não executar uma determinada relação não gera consequências imediatas no instante a ser considerado, contudo essas consequências se manifestam em relações futuras. Não somente isso, mas a regra 4.15 também considera um dado componente de aleatoriedade que está atrelado com este tipo de raciocínio. O predicado *possOfNegConseqFor*( $r_n$ ) semanticamente corresponde que existe a possibilidade de acontecer algo errado associado ao relacionamento  $r_n$ . O sentido deste termo é correspondido quando se verificam os elementos que causam este tipo de condição - que no caso desta regra envolve a ocorrência de uma violação em  $r_k$ , sendo que esse relacionamento afeta  $r_n$ .

$$\begin{aligned}
& relationViol(ag_m, g_i, r_k) \wedge affectsRels(r_k, r_n) \\
& \rightarrow possOfNegConseqFor(r_n) \\
& ag_m \in Agent, g_i \in Goal, r_k, r_n \in Relation,
\end{aligned} \tag{4.15}$$

O entendimento desta regra pode ser feito ao considerar um exemplo que já foi mencionado neste texto ao apresentar o correspondente do predicado *possOfNegConseqFor* e o predicado *affectsRels*, onde um eletricista usa um bastão isolante para acessar um dado barramento. Naquela parte do texto o problema é modelado por meio de duas relações: *relBastaoMedidor* (que define a relação que deve ser feita entre o bastão isolante com um aparelho medidor de corrente de fuga) e *relBastaoBarramento* (que consiste na relação entre o bastão com o barramento elétrico do quadro de energia). Tendo em vista que a ausência de uma medida em *gmedida* afeta a possibilidade de ocorrer algum evento grave em *relBastaoBarramento*, é dado - para esse caso - como verdade o seguinte predicado *affectsRels(relBastaoMedidor, relBastaoBarramento)*. Assim sendo, em um cenário em que ocorre a violação de relação em *gmedida*, o seguinte raciocínio pode ser feito: *relationViol(eletricista<sub>medidor</sub>, gmedida, relBastaoMedidor) ∧ affect(relBastaoMedidor, relBastaoBarramento) → possOfNegConseqFor(relBastaoBarramento)*.

A regra 4.15 demonstra como um agente pode ser submetido a consequências ruins sem necessariamente ser culpado por isso. Contudo, essa regra denota apenas possibilidade, não demonstrando o que acontece efetivamente quando o agente é submetido ao lado não favorável da possibilidade. Essa situação está atrelada à regra 4.25. Para lidar com as situações onde um agente é submetido a condições ruins, fez-se o uso do predicado *possOfNegConseqFor(g<sub>i</sub>, ag<sub>m</sub>, risk<sub>j</sub>, cs<sub>m</sub>)*. Entretanto, diferente das regras 4.13 e 4.14, essas consequências negativas tem seus correspondentes semânticos em outros predicados. O predicado *possOfNegConseqFor(r<sub>k</sub>)* é invocado com o propósito demonstrar que *r<sub>k</sub>* apresenta a possibilidade da ocorrência de um evento ruim mesmo que o agente que esteja executando essa relação não faça nada de errado. Contudo, esse predicado só denota a possibilidade. Para que o sentido semântico de que a possibilidade de um evento ruim realmente acontece foi considerado o uso do predicado *happensNegConseqFor(r<sub>k</sub>)*. Para o contexto desta regra, a semântica deste predicado exhibe o seguinte significado: o evento ruim associado a essa relação realmente aconteceu. Nesta situação se faz necessário adotar a outra concepção associada ao termo risco que é adotado neste modelo. Nesta regra, esse termo é adotado como um evento em potencial devido a incerteza associada ao evento.

Para compreender melhor essa situação é possível voltar ao exemplo do eletricista-bastão isolante-quadro de energia. Como já citado anteriormente o fato do agente medidor não executar sua atividade gera uma incerteza sobre a condição do isolamento do bastão. Se a medida for executada com sucesso (partido do pressuposto de que o medidor está em condições apropriadas de funcionamento), a condição do bastão é revelada eliminando qualquer incerteza a respeito disto. Contudo, como está sendo considerado um cenário em que isso não

foi feito, a não execução de *relBastaoMedidor* resultou no surgimento do risco *eletrocutado* com uma consequência de morte. Esse risco é definido como um potencial evento até que o eletricista de acesso ao barramento faça uso da ferramenta. Por conta disto, se usa o predicado  $hasRisk(r_k, risk_j, cs_m)$ . Tendo em vista que isso se dá por uma relação que está atrelada a um objetivo, se faz necessário considerar  $requiresCirc(g_i, r_k) \wedge (r_k \in rg_n)$  e  $instanceOfRel(r_k)$ . Para verificar a ação do agente nesta situação, o predicado  $starts(ag_m, g_i)$  também deve compor a regra.

$$\begin{aligned}
& possOfNegConseqFor(r_k) \wedge happensNegConseqFor(r_k) \wedge requiresCirc(g_i, r_k) \\
& \wedge instanceOfRel(r_k) \wedge hasRisk(r_k, risk_j, cs_m) \wedge starts(ag_m, g_i) \\
& \rightarrow negConseqFor(g_i, ag_m, risk_j, cs_m) \\
& r_k \in Relation, g_i \in Goal, risk_k \in Risk, cs_m \in Consequence \quad (4.16)
\end{aligned}$$

O exemplo em voga pode ser implementado nesta regra da seguinte forma:

$$\begin{aligned}
& possOfNegConseqFor(relBastaoBarramento) \\
& \wedge happensNegConseqFor(relBastaoBarramento) \\
& \wedge requiresCirc(g_{acessoBarramento}, relBastaoBarramento) \\
& \wedge instanceOfRel(relBastaoBarramento) \\
& \wedge hasRisk(relBastaoBarramento, eletrocutado, morte) \\
& \wedge starts(eletricista_{executor}, g_{acessoBarramento}) \\
& \rightarrow negConseqFor(g_{acessoBarramento}, eletricistaExecutor, eletrocutado, morte) \quad (4.17)
\end{aligned}$$

O exemplo se traduz na situação em que um bastão apresenta uma possibilidade de estar com o seu isolamento comprometido e isso resulta em um risco de eletrocutar o profissional que o usa, resultando na morte dele. Portanto, no momento em que a ferramenta é usada o eletricista morre eletrocutado, porque esse bastão pertencia as ferramentas cujo isolamento estava deteriorado.

A violação de entidade, dada pela regra 4.18, diferente das demais, resulta apenas no encerramento da atividade referente ao objetivo onde o método foi invocado. Os engenheiros desse modelo definiram essa regra partindo do pressuposto que a ausência de uma ferramenta, profissional, peça de substituição ou máquina simplesmente gera o impedimento do prosseguimento das atividades. Voltando ao exemplo do eletricista, se o profissional não tiver o bastão isolante para executar a ação, ele simplesmente não consegue dar prosseguimento ao objetivo fazendo com que o procedimento seja encerrado naquele exato instante.

$$\begin{aligned}
& entityViol(ag_m, g_i, e_k) \rightarrow stopped(g_i) \\
& ag_m \in Agent, g_i \in Goal, e_k \in Entity
\end{aligned} \tag{4.18}$$

Em termos de linguagem natural, a regra 4.18 é definida da seguinte forma: se acontecer uma violação de entidades, então o procedimento é encerrado no objetivo onde aconteceu.

A regra 4.26 advém do pressuposto de que na ocorrência de uma calamidade onde um profissional sai extremamente ferido ou morto (ocorrência do acidente), os demais envolvidos na manutenção não continuam por executar os procedimentos.

$$\begin{aligned}
& negConseqFor(g_k, ag_m, risk_j, cs_m) \rightarrow stopped(g_k) \\
& g_k \in Goal, risk_j \in Risk, cs_m \in Consequence, ag_m \in Agent
\end{aligned} \tag{4.19}$$

Essa regra, no escopo da linguagem natural, pode ser lida desta forma: Se acontecer um evento ruim em que um profissional sai morto ou gravemente ferido, então a manutenção é encerrada no objetivo onde a fatalidade aconteceu. Não há como afirmar que as regras 4.18 e 4.26 se aplicam para todo tipo de situação em qualquer procedimento. Operações militares, por exemplo, não se enquadram em situações assim, pois a morte de um soldado ferido não impede que o resto do batalhão continue em conflito. Contudo, neste estudo entendeu-se que o pressuposto dessas duas regras englobam diversos cenários, tais como: cenário industrial, subestação, usinas de produção de energia, certas atividades hospitalares, e entre outras da mesma natureza.

A regra estabelecida pela Figura 8 define o critério para quando um dado objetivo é considerado como atingido. Isso ocorre quando todos os agentes  $ag_n | n = i \dots j$  que são obrigados a atingir um certo objetivo  $g_k$  fazem isso sem a ocorrência e uma interrupção  $stopped(g_k, ag_n)$ . A expressão 8 retrata isso. Diferente das outras regras, especificar-se essa expressão como um algoritmo que avalia se um determinado objetivo foi interrompido agente por agente através de um (foreach) sobre *agentArray* (um *array* que trás todos os agentes que tentaram alcançar o objetivo *goal*).



```

function ifNotStopped(agentArray,goal)
    foreach(agentArray is agent)
        if(stopped(goal,agent)
            return false;
        endif
    endforeach
    if(!allAgenteObliged(agentArray,goal))
        return false;
    endif
    return true;
endFunction

if(ifNotStopped(agentArray,goal))
    return reached(goal);

```

---

Figura 8 – Condição para definir se um dado objetivo foi atingido ou não

Se o teste dado por  $if(stopped(goal, agent))$  é verdadeiro para pelo menos um dos agentes, então a função  $ifNotStopped(agentArray, goal)$  retorna falso. Se essa situação não acontecer para todos os agentes carregados em  $agentArray$ , então o algoritmo determina um segundo teste, que é dado pela função  $allAgenteObliged(agentArray, goal)$ . Essa função verifica se todos os agentes que são obrigados a alcançar o objetivo  $goal$  estão contidos em  $agentArray$ . No contexto desse algoritmo, se a função  $allAgenteObliged(agentArray, goal)$  retorna falso, então  $ifNotStopped$  também retorna falso, caso contrário  $ifNotStopped$  retorna verdade. Se a função  $ifNotStopped(agentArray, goal)$  retorna verdade, então o predicado  $reached(goal)$  é verdade também. O algoritmo apresenta a função  $isTrue$  onde o argumento é o predicado  $reached(goal)$ . A função  $isTrue(arg)$  sempre retorna verdade e tem como por objetivo informar que o seu respectivo argumento é verdade. Isso é necessário tendo em vista a diferença do formalismo adotado pelas demais regras em relação a 8.

Na linguagem natural essa expressão fica da seguinte forma: se todos os agentes que têm permissão para alcançar um dado objetivo o fizeram sem que esse tenha sido interrompido e considerando que um subgrupo deles é constituído por agentes que são obrigados a isso, então o objetivo é dado como alcançado.

A regra 4.20 apresenta a condição adequada para quando um agente está habilitado para atingir novos objetivos. Para isso, ele deve possuir um papel onde existe uma permissão para que ele possa atingir o próximo objetivo. Isso é traduzido por  $adoptsRole(ag_n, \rho_m) \wedge hasPermission(\rho_m, g_j)$ . Não apenas isso, mas o objetivo atual do agente deve ter sido atingido  $reached(g_i)$  e o objetivo em interesse deve estar associado como predicado  $nextGoal(g_i, g_j)$ . O termo  $enabledToStart(ag_i, g_j)$  corresponde semanticamente apenas que o agente está habilitado a buscar novos objetivos mas não significa que isso implicará em  $starts(ag_i, g_j)$  pois o que decide esses processos de transição consiste em aspectos que não correspondem a esse modelo. Essa dinâmica é discutida mais tarde na seção de *Predicados de Controle*.



$$\begin{aligned}
&adoptsRole(ag_n, \rho_m) \wedge hasPermission(\rho_m, g_j) \wedge nextGoal(g_i, g_j) \wedge reached(g_i) \\
&\quad \rightarrow enabledToStart(ag_i, g_j) \\
&ag_i, ag_n \in Agent, \rho_m \in Role, g_j \in Goal, g_i \in Goal \quad (4.20)
\end{aligned}$$

Em linguagem natural, a regra 4.20 exibe o seguinte: se um agente que alcançou um objetivo atual tem um papel que lhe dá permissão para buscar o próximo objetivo, então esse agente está habilitado para fazer isso.

A regra 4.21 apresenta a condição de parada do agente em relação ao seu papel, pois se o agente, que tem um determinado papel, cumpriu com todos os objetivos designados a ele, então ele deve encerrar sua operação. A verificação do papel é dado por  $adoptsRole(ag_n, \rho_m) \wedge hasPermission(\rho_m, g_i)$ , a análise semântica sobre o último objetivo associado a um certo papel é dado por  $lastGoal(g_i, \rho_m)$  e a verificação se aquele último objetivo foi atingido é dado por  $reached(g_i)$ .

$$\begin{aligned}
&adoptsRole(ag_n, \rho_m) \wedge hasPermission(\rho_m, g_i) \wedge lastGoal(g_i, \rho_m) \wedge reached(g_i) \\
&\quad \rightarrow stopped(g_i) \\
&ag_n \in Agent, \rho_m \in Role, g_i \in Goal \quad (4.21)
\end{aligned}$$

Portanto, a regra 4.21 em linguagem natural é definida da seguinte maneira: Se um agente cumpriu com todos os objetivos associados a permissão do papel dele, então esse agente deve encerrar suas atividades (em relação a esse papel).

#### 4.2.5 Diagrama de Atividades

As Figuras 9, 13 e 11 (o diagrama foi quebrado em três Figuras distintas com o propósito de melhorar a qualidade da resolução) apresentam a aplicação das regras em termos de diagrama de atividades. Essas Figuras devem ser entendidas como uma proposta de orientação das regras registradas na Subseção 4.2.4. Há outras maneiras de organizar essas regras em diferentes diagramas de atividades, sendo que a apresentada neste texto é apenas uma dessas. Isso se deve ao fato de que esse estudo pretende fornecer um modelo e não um algoritmo.

A primeira atividade do Diagrama da Figura 9 corresponde a “carregar todos os agentes”. Esse elemento é indiferente à estrutura das regras do modelo. A existência desta atividade no diagrama se dá por finalidades de implementação, uma vez que para uma máquina poder processar todos as atividades, primeiramente se faz necessário que informações sobre os agentes sejam carregadas na memória. As atividades “selecione um dos agentes”, “carregar os objetivos”, “há objetivos que não foram alcançados” e “o agente escolhe por tentar alcançar o objetivo” fazem referência às regras 4.20 e 4.9. Aquela analisa qual é a próxima regra que esta

em condições de ser atingida pelo agente e esta verifica a permissão do agente no que tange a possibilidade de poder adotar o objetivo.

Na figura 13, o ponto de decisão “todas as condições necessárias para esse objetivo estão presentes?” e a atividade “violação de condição” fazem referência a regra 4.10, que define uma violação de condição para o caso do agente tentar executar alguma atividade sem que todas as condições estejam presentes naquele instante.

Na figura 11, o ponto de decisão “todas as entidades necessárias para esse objetivo estão presentes?” e a atividade “violação de relação” fazem referência à regra 4.12, pois definem o ocorrido no que diz respeito a ausência de uma entidade ao verificar um dado objetivo em análise.

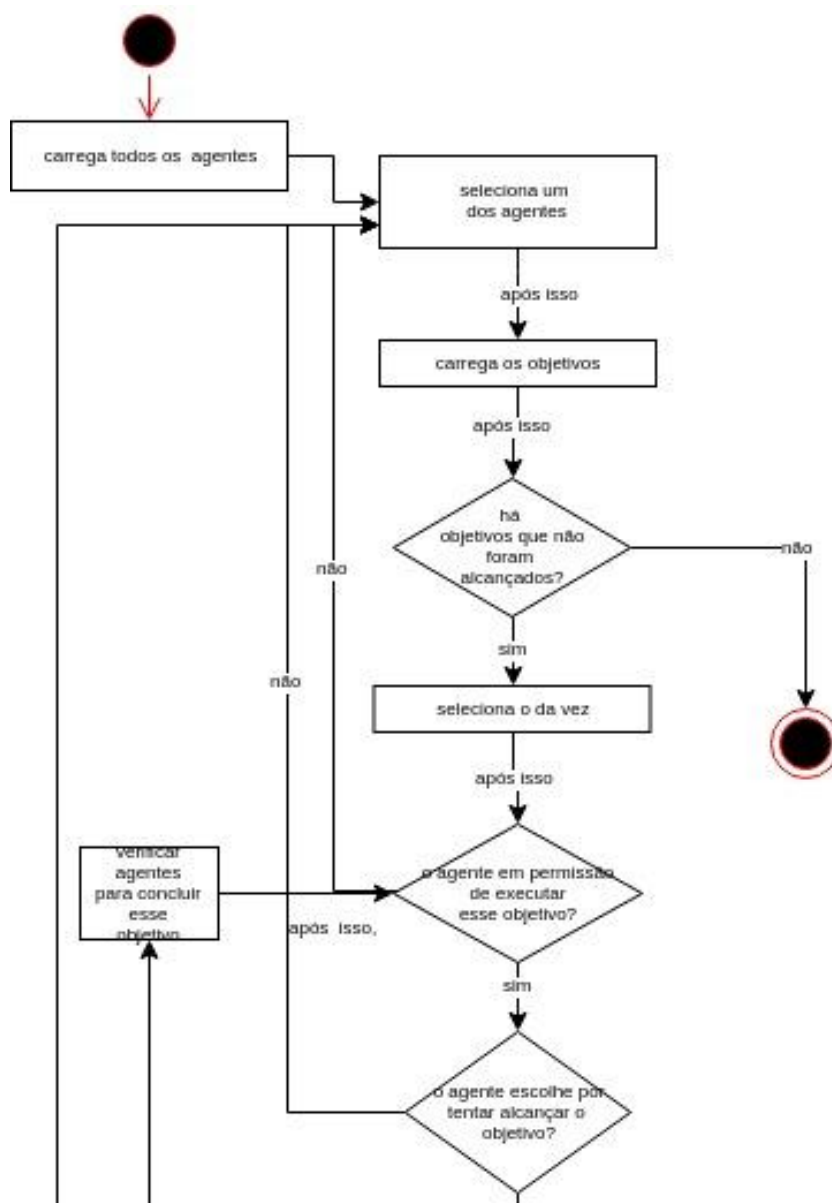


Figura 9 – Diagrama de atividades do modelo (1)

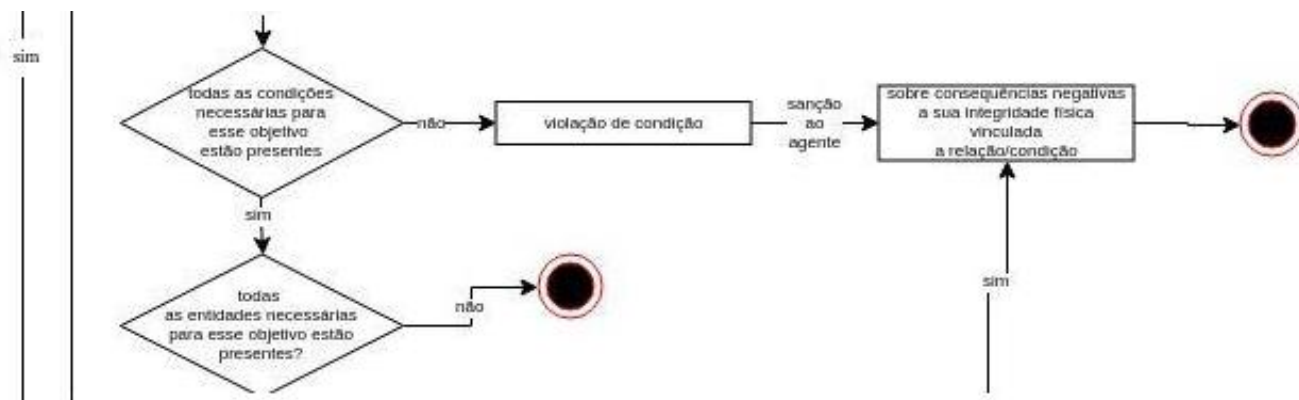


Figura 10 – Diagrama de atividades do modelo (2)

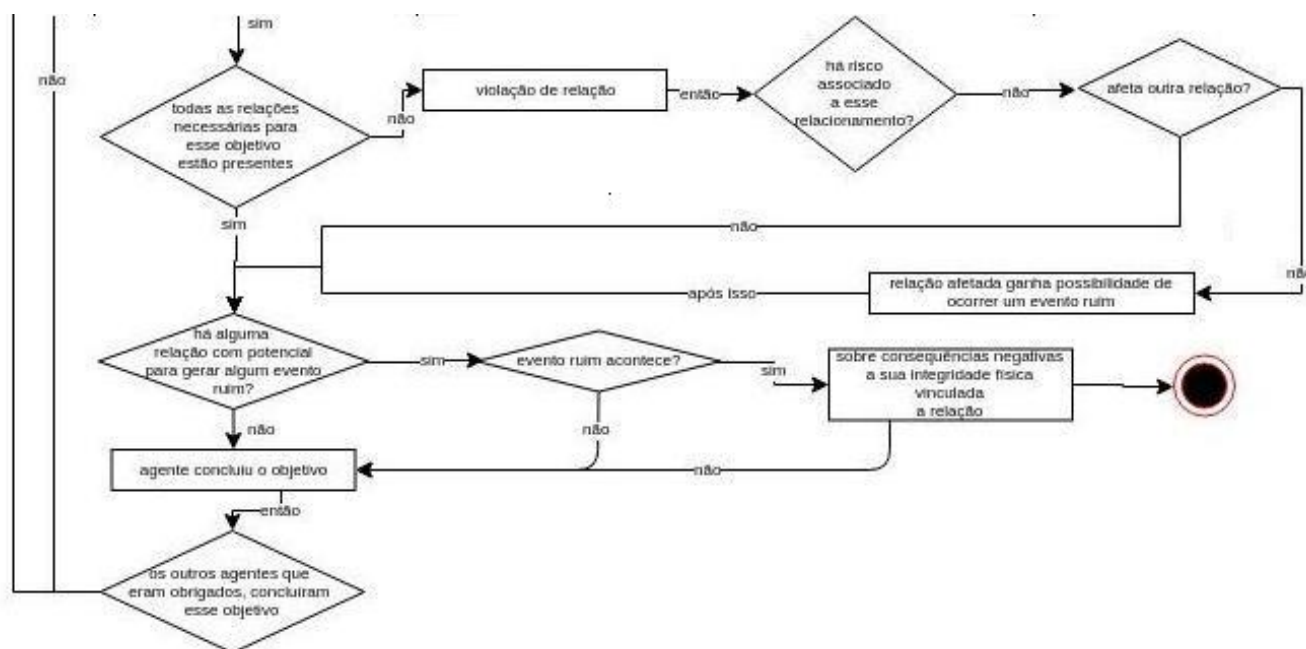


Figura 11 – Diagrama de atividades do modelo (3)

O ponto de decisão “todas as relações para esse objetivo estão presentes” e “violação de relação” representam a regra 4.11. Isso se deve ao fato de que essas atividades avaliam se uma das relações necessárias para cumprir com o objetivo não está presente, resultando em uma violação de relacionamento.

As atividades “violação de condição”, e “sobre consequências negativas a sua integridade física vinculada a relação/condição” condizem com a regra 4.13 que define as consequências de uma violação de condição. A atividade “violação de relação” em conjunto com a segunda atividade das presentes na sentença anterior fazem referência a regra 4.14, pois apresenta as sanções relacionadas a uma violação de relacionamento.

O ponto de decisão “há risco associado a esse relacionamento?”, “afeta outra relação” e “relação afetada ganha possibilidade de ocorrer um evento ruim”, apontam para a regra 4.15

pois ambas situações representam como a ocorrência de uma violação de relacionamento afeta um outro relacionamento. A regra 4.25 corresponde aos seguintes aspectos do diagrama, “há alguma relação com potencial para gerar algum evento ruim?”, “evento ruim acontece?”, “sobre consequências negativas a sua integridade física vinculadas a relação” tendo em vista a equivalência semântica entre esses elementos sendo que ambas as representações se preocupam com a análise de relações que possuem possibilidades de algum evento ruim surgir sobre o agente do objetivo.

A regra 4.18 é representada por “violação de entidade” e pelo elemento que indica o fim do programa. Isso se deve ao fato de que a regra 4.18 determina o encerramento do processo na ocorrência de uma violação de entidade.

Os eventos “agente concluiu o objetivo”, “os outros agentes que eram obrigados, concluíram esse objetivo” e “verificar agentes para concluir esse objetivo” apontam para as regras 4.20, 4.21. Esse conjunto de atividades e pontos de decisões apresentam os critérios para definir quando um objetivo foi totalmente alcançado (que é a mesma finalidade dessas regras justificando a equivalência entre ambos formalismos). A regra 8 é explicitada no diagrama toda vez que a atividade “sobre consequências negativas a integridade física vinculada a relação/condição” aponta para o fim do programa, tendo em vista que ambas representações definem o encerramento das atividades na ocorrência de feridos.

O diagrama presente nas Figuras 9, 13 e 11 apresenta 4.10 como a primeira regra de violação a ser executada. O motivo disto reside no fato de que essa regra verifica se o agente respeitou todas as condições do ambiente. Se o agente não fizer isso, ele está sujeito a penalidades físicas encerrando o programa. Ou seja, não abre margens para verificação de outras violações, porque em um caso real, alguém que executa uma atividade sem que todas as condições estejam presentes, então esse alguém está fadado a encerrar qualquer ação em curso. Mesmo que esse alguém estivesse na condição de cometer outras violações, não seria possível fazê-las pois esta primeira violação cometida por ele foi o suficiente para interromper os procedimentos como um todo. Esse mesmo princípio fundamenta o sequenciamento das demais regras, sendo que logo em seguida é a 4.12, pois se alguma entidade necessária (ferramenta), não estiver presente no cenário, então não existe possibilidade da continuidade dos procedimentos inviabilizando a realização das relações não fazendo sentido verificar 4.11. Contudo, os engenheiros definem essa estrutura apenas como uma proposta que deve ser modificada em função dos interesses da aplicação. Por exemplo, supondo que uma equipe tenha o interesse de usar este modelo para desenvolver jogos sérios com a intenção de analisar todas as violações que podem ser cometidas por um jogador um dado cenário, então para esse caso não há sentido usar esse fluxo de atividades. Em uma condição assim, os engenheiros do jogo devem - usando as mesmas regras - mudar o fluxo do diagrama de atividade para verificar todas as regras de violação antes de analisar se o programa deve ou não ser interrompido.

#### 4.2.6 Predicados de Controle e de Estrutura

Esse estudo apresenta duas categorias de predicados: *controle* e *estrutura*. Os predicados de estrutura são aqueles, cujo usuário do modelo não possui a liberdade de definir sua estrutura interna por intermédio de outras regras lógicas ou por valores. Isso se deve ao fato de que esses vocábulos têm sua estrutura alicerçada nas concepções deste modelo sendo que são essenciais para que o modelo funcione como foi concebido. Assim sendo, o modelador deve fazer uso delas, apenas com o propósito de especificar os objetos de interesse. Em termos práticos não existe dificuldade em identificar esses predicados, pois sua própria natureza não abre margem para que o modelador consiga escrever novos predicados e novas regras para determinar o seu respectivo valor. Esses predicados são:

- $possEntityRel(r_l, e_i, e_k)$
- $adoptsRole(ag_n, \rho_m)$
- $hasObligation(\rho_m, g_j)$
- $hasPermission(\rho_m, g_j)$
- $reached(g_k)$
- $stopped(g_n, ag_m)$
- $nextGoal(g_i, g_j)$
- $requiresCirc(g_i, circ_k)$
- $requiresEntity(g_i, e_k)$
- $conditionViol(ag_i, g_j, c_k)$
- $relationViol(ag_i, g_j, r_k)$
- $entityViol(ag_i, g_j, e_k)$
- $hasRisk(circ_i, risk_j, cs_k)$
- $possOfNegConseqFor(r_l)$
- $affectsRels(r_k, r_n)$
- $negConseqFor(g_k, ag_i, risk_k, cs_m)$
- $lastGoal(g_i, \rho_m)$
- $instanceOfRel(circ_i)$
- $instanceOfCond(circ_i)$

- $enabledToStart(ag_i, g_j)$

Para exemplificar, pode-se considerar o predicado  $possEntityRel(r_l, e_i, e_k)$ . Se existir uma entidade  $A$ , uma entidade  $B$  e uma relação entre  $relAB$ , então esse termo é escrito desta forma:  $possEntityRel(relAB, A, B)$ . O valor verdade deste predicado, não pode ser modificado para a criação de algum cenário e nem pode ser determinado por outras regras. Se modelador fizer isso então estará modificando a estrutura do modelo. Ou seja, esse é um predicado de estrutura no que tange aos aspectos fundamentais e semânticos desta representação.

Por outro lado os predicados *de controle* possuem um correspondente sintático e semântico no modelo, mas os seus valores devem ser forçados conforme o cenário que se deseja criar ou conforme outras regras de implicabilidade que não fazem parte da estrutura deste modelo. A não determinação dos valores destes predicados inviabilizam que o modelo seja analisado de forma procedural. Portanto, a aplicabilidade deste modelo em um dado contexto a fim de realizar uma averiguação procedural dos fatos só pode ser feita por determinar os valores Verdade para esses predicados. Fazem parte deste conjunto os seguintes termos:

- $isPresent(X)$
- $starts(ag_i, g_j)$
- $happensNegConseqFor(r_k)$

Pode-se considerar o seguinte exemplo: Um agente  $ag_a$  deve executar o objetivo  $g_1$  e  $g_2$ , os predicados a seguir implementam este modelo para o exemplo:

- $nextGoal(g_1, g_2)$
- $possEntityRel(rAB, entA, entB)$
- $possEntityRel(rCE, entC, entE)$
- $instanceOfRel(rAB),$
- $instanceOfRel(rCE)$
- $instanceOfCond(cond_1)$
- $requiresCirc(g_1, rAB)$
- $requiresCirc(g_2, rCE)$
- $requiresCirc(g_1, cond_1)$
- $requiresCirc(g_2, cond_1)$

- $requiresEntity(g_1, entA)$
- $requiresEntity(g_1, entB)$
- $requiresEntity(g_2, entC)$
- $requiresEntity(g_2, entE)$
- $affectsRels(rAB, rCE)$
- $hasRisk(cg_1, risk_1, cs_1)$
- $hasRisk(rCE, risk_2, cs_2)$
- $adoptsRole(ag_a, \rho_1)$
- $hasObligation(\rho_1, g_1)$
- $hasObligation(\rho_1, g_2)$

Apesar de todos os predicados denotarem uma certa condição, e de serem suficientes, para definir uma certa representação de mundo, não é possível fazer raciocínio algum. Isso, pois não se sabem quais são as ações dos agentes, tão pouco quais condições e cenários se desejam representar.

Para isso, se faz necessário definir um cenário de mundo. Por exemplo, pode-se definir o seguinte cenário:  $starts(ag_a, g_1)$ ,  $\neg isPresent(rAB)$ ,  $starts(ag_b, g_2)$  e  $possOfNegConseqFor(rCE) \rightarrow happensNegConseqFor(rCE)$ .

Para esse caso é possível obter as seguintes relações de inferência:

$$requiresCirc(g_1, rAB) \wedge \neg isPresent(rAB) \wedge instanceOfRel(rAB) \wedge starts(ag_a, g_1) \rightarrow relationViol(ag_a, g_1, rAB) \quad (4.22)$$

$$relationViol(ag_a, g_1, rAB) \wedge affectsRels(rAB, rCE) \rightarrow possOfNegConseqFor(rCE) \quad (4.23)$$

$$possOfNegConseqFor(rCE) \rightarrow happensNegConseqFor(rCE) \quad (4.24)$$

$$\begin{aligned}
& possOfNegConseqFor(rCE) \\
& \wedge happensNegConseqFor(rCE) \\
& \wedge requiresCirc(g_2, rCE) \\
& \wedge instanceOfRel(rCE) \\
& \wedge hasRisk(rCE, eletrocutado, cs_2) \\
& \wedge starts(ag_a, g_2) \\
& \rightarrow negConseqFor(g_2, ag_a, risk_2, cs_2)
\end{aligned} \tag{4.25}$$

$$negConseqFor(g_2, ag_a, risk_2, cs_2) \rightarrow stopped(g_2) \tag{4.26}$$

Esses raciocínios e conclusões só foram possíveis porque o modelador forçou o valor de três predicados e definiu uma relação de implicação. Isso acontece por conta de três motivos: 1 - Esse é um modelo de *SMA*, 2 - esse modelo apresenta grau de liberdade para escolher a disposição das entidades, condições e relações e 3 - não há como definir a solução de uma possibilidade.

Para o primeiro caso o predicado *starts(ag<sub>i</sub>, g<sub>j</sub>)* é resultado de estados internos do agente. Por exemplo, o desenvolvedor pode programar um agente que possui o estado de medo, então sob certas condições ele resolve não tentar alcançar o objetivo gerando valor falso para esse predicado, ou pode definir um agente que pondera pouco ao decidir se deve ou não tentar alcançar o objetivo. Isso pode ser feito por meio de modelos de agentes tais como: agentes lógicos, arquitetura BDI, agentes reativos e agentes em camada. Se for do interesse do modelador, o mesmo pode simplesmente definir o valor verdade para o predicado em certas condições.

O mesmo se aplica para o *isPresent(X)* em que o desenvolvedor pode definir um cenário por meio dos estados internos, por exemplo, os agentes esqueceram uma determinada ferramenta em um certo local ou, o agente apresenta um algoritmo para determinar qual ferramenta é a mais apropriada para uma condição específica. O modelador também é livre para gerar diferentes cenários simplesmente por definir valores diferentes para *isPresent(X)*. Por exemplo, supondo que uma equipe está desenvolvendo um jogo sério para avaliar profissionais de uma certa indústria. Para avaliar a competência dos trabalhadores, o modelador poderá usar este predicado por adicionar ou remover entidades e condições com base nas necessidades de avaliação.

O terceiro motivo reside no fato de que o predicado *possOfNegConseqFor(X)* denota apenas que existe uma possibilidade de ocorrer algum evento ruim *happensNegConseqFor(X)*.



Contudo, se esse evento ocorrerá ou não, não é possível definir pois isso depende de questões estatísticas do objeto de estudo. Assim sendo, o usuário deste modelo possui algumas possibilidades de ação, tais como: quando *possOfNegConseqFor(X)* for verdade, então definir *happensNegConseqFor(X)* por meio de um número aleatório, para algumas situações onde ocorre *possOfNegConseqFor(X)*, tratar *happensNegConseqFor(X)* como verdade e para das situações tratar *happensNegConseqFor(X)* como falso, ou definir verdade para *happensNegConseqFor* como base em algum estudo probabilístico. Isso dependerá da finalidade dos modeladores.

### 4.3 Caso Introdutório

A clareza ao leitor na tarefa de mostrar o uso do modelo conceitual para especificar um estudo de caso real é extremamente prejudicada tendo em vista a quantidade de elementos presentes no modelo. Com base nisso, o autor desse texto entendeu que antes de apresentar a aplicação do modelo em um estudo de caso, se faz necessário avaliar como se dá a aplicação deste modelo para um caso fictício mais simples.

Para isso pode-se considerar o seguinte cenário: dois funcionários (Fernando e Bruno) são escalados para realizar a troca de uma lâmpada. Então, esses profissionais necessitam dos seguintes objetos: escada, lâmpada e bocal. Essa troca deve ser feita sob as seguintes condições: piso seco, ambiente iluminado e o disjuntor da respectiva lâmpada deve estar devidamente desligado. Os riscos associado a esse trabalho são os seguintes: queda da escada (com a consequência de fratura), queda da lâmpada (com a consequência de ferir algum profissionais que esteja debaixo deste objeto) e ser eletrocutado (com a consequência de ferimentos sérios). A lista a seguir trás o sequenciamento das atividades nas quais esse processo ocorre:

1. Tanto Fernando como Bruno, devem posicionar a escada logo a baixo da lâmpada.
2. Fernando deve subir na escada em direção da lâmpada enquanto Bruno a segura.
3. Quando Fernando estiver posicionado sob a lâmpada, então ele deve removê-la e entregá-la a Bruno.
4. Bruno deve posicionar a lâmpada antiga em uma caixa. Após isso, deve entregar a lâmpada nova a Fernando.
5. Fernando, então, deve colocar a lâmpada nova no respectivo local.
6. Fernando, enquanto Bruno segura as escadas, deve retornar ao chão.

A Subseção a seguir exhibe esse estudo de caso aplicado a esse modelo.

#### 4.3.1 Aplicação do Modelo

Tanto Fernando como Bruno podem ser considerados como Agentes e portanto são representados da seguinte forma:

$$Agents = \{fernando, bruno\}$$

Os demais elementos são artefatos e dentro do contexto deste estudo se enquadram da seguinte forma:

$$Artefact = \{escada, lampadaAntiga, lampadaNova, bocal\}$$

As condições necessárias para que um objetivo possa ser alcançado são representadas desta maneira:

$$Condition = \{ambIlumin, disjDesl, chaoSeco\}$$

Em que *ambIlumin* = Ambiente Iluminado, *disjDesl* = disjuntor da lâmpada desligado e *chaoSeco* = chão seco. No que tange ao papel dos funcionários, eles serão considerados *trocLamp* (trocaador de lâmpada) e *aux* (auxiliar).

$$Roles = \{trocLamp, aux\} \quad (4.27)$$

Os Riscos são representados da seguinte forma:

$$Risk = \{quedaEscada, quedaLampada, eletrocutado\} \quad (4.28)$$

Onde *quedaEscada* = queda da Escada, *quedaLampada* = queda da Lâmpada e *eletrocutado* = eletrocutado. As consequências sob as quais esses profissionais são submetidos caso aconteça um acidente associado a esses riscos são:

$$Consequence = \{fratura, ferQL, ferEletr\} \quad (4.29)$$

Em que *fratura* = fratura, *ferQL* = ferimento causado pela Queda da Lâmpada e *ferEletr* = ferimento causado por eletricidade.

Os objetivos podem ser representados da seguinte maneira:

1.  $gBL \rightarrow$  Posicionar a escada embaixo da lâmpada.
2.  $gSubir \rightarrow$  Subir a escada embaixo da lâmpada:
3.  $gSegura \rightarrow$  Segurar a escada.
4.  $gRemover \rightarrow$  Remover a lâmpada antiga.
5.  $gCaixa \rightarrow$  Guardar a lâmpada antiga dentro da caixa.
6.  $gNova \rightarrow$  Entregar lâmpada nova
7.  $gNovaBocal \rightarrow$  Colocar a lâmpada no respectivo bocal.
8.  $gRetornarChao \rightarrow$  Retornar ao chão.

No que tange as relações entre as entidades, pode-se considerar o seguinte:

1.  $rFerEsc \rightarrow$  relação do Fernando, ter de tocar na escada para movimentá-la.
2.  $rBrunEsc \rightarrow$  relação do Bruno, ter de tocar na escada para movimentá-la ou para mantê-la em sua respectiva posição.
3.  $rSubirFerEsc \rightarrow$  relação do Fernando, subir pela escada.
4.  $rFerLampAnt \rightarrow$  relação do Fernando tocar na lâmpada antiga.
5.  $rBrunLampAnt \rightarrow$  relação do Bruno tocar na lâmpada antiga.
6.  $rBrunLampNova \rightarrow$  relação do Bruno tocar na lâmpada nova.
7.  $rFerLampNova \rightarrow$  relação do Fernando tocar na lâmpada nova.
8.  $rLampNovaBocal \rightarrow$  relação de posicionar a lâmpada nova no bocal.
9.  $rDescFerEsc \rightarrow$  relação entre Fernando e Escada onde aquele está descendo por esta.

Uma vez definidos as entidades das classes, é possível especificar os predicados. A lista a seguir especifica o predicado  $possEntityRel(r_l, e_i, e_k)$ :

1.  $possEntityRel(rFerEsc, fernando, escada)$
2.  $possEntityRel(rBrunEsc, bruno, escada)$
3.  $possEntityRel(rSubirFerEsc, fernando, escada)$
4.  $possEntityRel(rFerLampAnt, fernando, lampadaAntiga)$

5.  $possEntityRel(rBrunLampAnt, fernando, lampadaAntiga)$
6.  $possEntityRel(rBrunLampNova, bruno, lampadaNova)$
7.  $possEntityRel(rFerLampNova, fernando, lampadaNova)$
8.  $possEntityRel(rLampNovaBocal, lampadaNova, bocal)$

A lista a seguir define o predicado  $adoptsRole(ag_n, \rho_m)$ :

1.  $adoptsRole(fernando, trocL)$
2.  $adoptsRole(bruno, aux)$

A lista a seguir define o predicado  $hasObligation(\rho_m, g_n)$ .

1.  $hasObligation(trocLamp, gBL)$
2.  $hasObligation(aux, gBL)$
3.  $hasObligation(trocLamp, gSubir)$
4.  $hasObligation(aux, gSegurar)$
5.  $hasObligation(trocLamp, gRemover)$
6.  $hasObligation(aux, gCaixa)$
7.  $hasObligation(aux, gNova)$
8.  $hasObligation(trocLamp, gNovaBocal)$
9.  $hasObligation(trocLamp, gRetornarChao)$

A lista a seguir exibe a especificação do predicado  $nextGoal(g_i, g_j)$ .

1.  $nextGoal(gBL, gSubir)$
2.  $nextGoal(gSubir, gRemover)$
3.  $nextGoal(gSegura, gCaixa)$
4.  $nextGoal(gRemover, gCaixa)$
5.  $nextGoal(gCaixa, gNova)$
6.  $nextGoal(gNova, gNovaBocal)$

7. *nextGoal(gNovaBocal, gRetornarChao)*

A lista a seguir exibe a especificação do predicado *requiresCirc(goal, circ)* para cada objetivo.

1. *requiresCirc(gBL, rFerEsc), requiresCirc(gBL, rBrunEsc),  
requiresCirc(gBL, ambIlumin), requiresCirc(gBL, chaoSeco)*
2. *requiresCirc(gSubir, rSubirFernEsc), requiresCirc(gSubir, ambIlumin),  
requiresCirc(gSubir, chaoSeco)*
3. *requiresCirc(gRemover, rFerLampAnt), requiresCirc(gRemover, ambIlumin),  
requiresCirc(gRemover, disjDesl)*
4. *requiresCirc(gCaixa, rBrunoLampAnt), requiresCirc(gCaixa, ambIlumin),  
requiresCirc(gCaixa, disjDesl)*
5. *requiresCirc(gNova, rBrunLampNova), requiresCirc(gNova, ambIlumin),  
requiresCirc(gNovachaoSeco)*
6. *requiresCirc(gNovaBocal, rFerLampNova), requiresCirc(gNovaBocal, rLampNovaBocal),  
requiresCirc(gNovaBocal, ambIlumin), requiresCirc(gNovaBocalchaoSeco)*
7. *requiresCirc(gRetornar, rDescFerEsc), requiresCirc(gBL, rBrunEsc),  
requiresCirc(gBL, ambIlumin), requiresCirc(gBL, chaoSeco)*

A lista a seguir exibe a especificação do predicado *requiresEntity(goal<sub>i</sub>, e<sub>j</sub>)* para cada objetivo.

1. *requiresEntity(gBL, fernando), requiresEntity(gBL, escada), requiresEntity(gBL, bruno)*
2. *requiresEntity(gSubir, fernando), requiresEntity(gSubir, escada)*
3. *requiresEntity(gRemover, fernando), requiresEntity(gRemover, lampadaAntiga)*
4. *requiresEntity(gCaixa, bruno), requiresEntity(gCaixa, lampadaAntiga)*
5. *requiresEntity(gNova, bruno), requiresEntity(gCaixa, lampadaNova)*
6. *requiresEntity(gNovaBocal, fernando), requiresEntity(gNovaBocal, lampadaNova),  
requiresEntity(gNovaBocal, bocal)*
7. *requiresEntity(gRetornar, fernando), requiresEntity(gRetornar, escada),  
requiresEntity(gRetornar, bruno)*

A lista a seguir exhibe a especificação do predicado  $hasRisk(crts, risk_j, cs_k)$

1.  $hasRisk(ambIlumin, quedaEscada, fratura)$  - o profissional que executa alguma atividade, como por exemplo subir numa escada em um ambiente mal iluminado, pode cair da escada por não ter uma clara visão de como se mover.
2.  $hasRisk(ambIlumin, quedaLampada, ferQL)$  - o profissional que executa alguma atividade, como colocar uma lâmpada, em um ambiente mal iluminado pode errar a adequada posição da mesma fazendo com que esse objeto entre em queda livre ferindo os profissionais embaixo.
3.  $hasRisk(disjDesl, eletrocutado, ferEletr)$  - o profissional que executa a troca de lâmpada sem que o disjuntor esteja desligado, está se submentendo ao risco de ser eletrocutado tendo ferimentos por eletricidade.
4.  $hasRisk(chaoSeco, quedaEscada, fratura)$  - o profissional que sobe em uma escada com chão molhado acaba se submentendo ao risco de escorregar e fraturar-se por conta de uma queda.
5.  $hasRisk(rSubirFerEsc, quedaEscada, fratura)$  - se o Fernando subir de maneira errada pela escada, ele pode cair e se lesionar.
6.  $hasRisk(rDescFerEsc, quedaEscada, fratura)$  - se o Fernando descer de maneira errada pela escada, ele pode cair e se lesionar.

A lista a seguir exhibe a especificação do predicado  $affectsRels(r_k, r_n)$ :

1.  $affectsRels(rFerEsc, rSubirFerEsc)$  - se Fernando posicionar a escada de forma inapropriada, por mais que isso não gere consequências imediatas naquele instante, o agente que estiver atrelado a  $rSubirFerEsc$  pode sofrer uma queda vindo a se lesionar por conta disso.
2.  $affectsRels(rBrunEsc, rSubirFerEsc)$  - se Bruno posicionar a escada de forma inapropriada, por mais que isso não gere consequências imediatas naquele instante, o agente que estiver atrelado a  $rSubirFerEsc$  pode sofrer uma queda vindo a se lesionar por conta disso.
3.  $affectsRels(rFerEsc, rDescFerEsc)$  - se Fernando posicionar a escada de forma inapropriada, por mais que isso não gere consequências imediatas naquele instante, o agente que estiver atrelado a  $rDescFerEsc$  pode sofrer uma queda vindo a se lesionar por conta disso.

4. *affectsRels(rBrunEsc, rDescFerEsc)* - se Bruno posicionar a escada de forma inapropriada, por mais que isso não gere consequências imediatas naquele instante, o agente que estiver atrelado a *rDescFerEsc* pode sofrer uma queda vindo a se lesionar por conta disso.

#### 4.3.2 Raciocínios

Uma vez especificado os elementos básicos desse cenário nos moldes do modelo em análise nesse estudo, é possível aplicar as regras para avaliar o que acontece em cenários específicos.

Um cenário específico consiste em avaliar o que acontece, segundo esse modelo, quando Fernando realiza a substituição da lâmpada antiga sem que o disjuntor esteja devidamente desligado. Para esse tipo de situação, os predicados a serem considerados são esses:

1. *requiresCirc(gRemover, disjDesl)* - esse predicado tem que ser considerado porque está associado ao objetivo *gRemover* a condição de *disjDesl*.
2.  $\neg isPresent(disjDesl)$  - esse predicado tem que ser considerado porque mostra que a condição *disjDesl* não está presente quando o respectivo agente tenta alcançar o objetivo.
3. *instanceOfCond(disjDesl)* - esse predicado é relevante porque se faz necessário avaliar se *disjDesl* é de fato uma condição, tendo em vista que a natureza desta violação consiste em uma violação de condição.
4. *starts(fernando, gRemover)* - esse predicado é necessário porque denota que o agente fernando realmente começou a executar o objetivo *gRemove*.
5. *hasRisk(disjDesl, eletrocuado, ferEletr)* - esse predicado é necessário porque relaciona a condição *disjDesl* com o risco do profissional ser *eletrocutado* e com as consequências *ferEletr*.

Uma vez feito o levantamento dos predicados necessários, pode-se considerar as regras do modelo. Neste caso, as regras que avaliam esse tipo de situação são 4.10 e 4.13:

$$\begin{aligned}
 &requiresCirc(gRemover, disjDesl) \wedge \\
 &\quad \neg isPresent(disjDesl) \wedge \\
 &\quad instanceOfCond(disjDesl) \wedge \\
 &\quad starts(fernando, gRemover) \rightarrow \\
 &conditionViol(fernando, gRemover, disjDesl)
 \end{aligned}$$

(4.30)

$$\begin{aligned}
& conditionViol(fernando, gRemover, disjDesl) \wedge \\
& hasRisk(disjDesl, eletrocutado, ferEletr) \rightarrow \\
& negConseqFor(gRemover, fernando, eletrocutado, ferEletr)
\end{aligned}
\tag{4.31}$$

Por esse raciocínio, é possível observar que se o agente executar o procedimento de manutenção sem que a condição, desligar disjuntor, esteja presente, então esse agente (Fernando) será eletrocutado e terá ferimentos por conta disso.



## 5 Estudo de Caso

Esse capítulo apresenta os resultados atrelados à etapa metodológica apresentada na Seção 3.4 que tem por finalidade validar o modelo conceitual no que tange a um cenário real de manutenção com alta potencialidade para a ocorrência de acidentes.

### 5.1 Introdução ao Problema

O estudo de caso desta pesquisa consiste em sete profissionais de linha viva (profissionais que realizam manutenção em equipamentos elétricos energizados) designados com o propósito de realizar a substituição de um isolador de pedestal. Os papéis desses profissionais são: um supervisor e seis executores. A manutenção deve ser executada apenas sob as seguintes condições: céu ensolarado e umidade relativa do ar menor que 70 por cento. Todos os profissionais devem possuir os EPI's necessários: capacete, óculos de sol, roupa isolante e antichamas, luvas isolantes e botas isolantes. Os profissionais que entram no potencial devem estar vestidos com roupa condutiva e cabo guarda. As ferramentas necessárias para resolver esse problema são: bastão garra de diâmetro 64 x 3600 mm, sela de diâmetro 65, colar, corda de fibra sintética, carretilha, chave com catraca, bastão universal, soquete adequado, locador de pino e bastão com soquete multiangular. O método selecionado para esse tipo de manutenção é a distância onde o eletricista não acessa diretamente o potencial, mas faz isso por intermédio de um bastão isolante. A substituição do isolador de pedestal pode ser escrita nos seguintes objetivos:

1. Limpar, secar e testar corda.
2. Instalar Bastão Garra na estrutura com o pedestal a ser substituído.
3. Instalar sela com colar na estrutura
4. Amarrar o bastão na parte superior da estrutura com a corda.
5. Amarrar o olhal do bastão ao cavalo da sela atrás de uma corda.
6. Instalar um segundo conjunto bastão e sela no lado oposto da estrutura.
7. Enforcar um estropo de Náilon no corpo do isolador.
8. Colocar a extremidade do estropo no gancho da corda de serviço.
9. Afrouxar os parafusos do conector que prendem a barra ao isolador.
10. Terminar de retirar os parafusos com o bastão com o soquete multiangular.
11. Elevar a barra através da corda que une a sela ao bastão.

12. Apertar o colar através da porca borboleta.
13. Segurar firmemente a corda de serviço.
14. Sacar parafusos da base da coluna.
15. Baixar o isolador ao solo
16. Içar o Isolador
17. Colocar Parafusos na base da coluna.
18. Baixar a barra para que a mesma apoie no novo isolador.
19. Colocar os parafusos do conector que prende a barra ao novo isolador.
20. Retirar Equipamentos

#### 5.1.1 Especificação dos Agentes e seus Papéis

A tabela 3 apresenta todos os agentes que fazem parte da manutenção.

<b>símbolo</b>	<b>significado</b>
agente1	Um dos agentes participantes da manutenção
agente2	Um dos agentes participantes da manutenção
agente3	Um dos agentes participantes da manutenção
agente4	Um dos agentes participantes da manutenção
agente5	Um dos agentes participantes da manutenção
agente6	Um dos agentes participantes da manutenção
agente7	Um dos agentes participantes da manutenção

Tabela 3 – Os agentes que constituem uma manutenção

A tabela 4 apresenta todas as funções que deverão ser exercidas pelos agentes.

<b>papel</b>	<b>descrição</b>
supervisor	Atribui papel a outros profissionais
executor1	Tem como por finalidade executar certas atividades manuais vinculadas a manutenção
executor2	Tem como por finalidade executar certas atividades manuais vinculadas a manutenção
executor3	Tem como por finalidade executar certas atividades manuais vinculadas a manutenção
executor4	Tem como por finalidade executar certas atividades manuais vinculadas a manutenção
executor5	Tem como por finalidade executar certas atividades manuais vinculadas a manutenção

Tabela 4 – Os papéis relevantes para a ocorrência da manutenção

A tabela 5 define o predicado  $adoptsRole(ag_n, \rho_m)$  onde  $ag_n$  é representado pela coluna agente e  $\rho_m$  é representado pela coluna papel.

agente	papel
agente1	supervisor
agente2	executor1
agente3	executor1
agente4	executor2
agente5	executor3
agente6	executor4
agente7	executor5

Tabela 5 – Relação  $adoptsRole(ag_n, \rho_m)$

### 5.1.2 Especificação das Ferramentas e Objetivos

A tabela 6 apresenta todos artefatos que fazem parte da descrição deste estudo de caso.

Artefato	Descrição
capacete	EPI usado pelo profissional para proteger a cabeça
óculos	Óculos usado para evitar dificuldades de enxergar presentes em dias claros
roupagem	Consiste em roupas isolantes e anti-chamas
luva	Luvas Isolantes
bota	Botas Isolantes para evitar que o profissional seja eletrocutado
bastaoGarra	Bastão isolante que possui uma ferramenta em estrutura de garra. 64 X 3600 mm
sela	Possui diâmetro 65 mm, é fixada na torre para sustentar o bastão.
colar	Estrutura que fica fixa na sela, bastão isolante é travado no colar.
corda	Corda Isolante.
carretilha	Carretilha que, em conjunto com a corda, é usada para mover material na vertical.
bastaoUniversal	Bastão isolante que permite o acoplamento de múltiplas ferramentas.
soquete	Usado na manipulação de parafusos.
locador	Usado como pino direcional em alinhamento de furo de parafusos, auxiliado na inserção de pinos e parafusos.
bastaoGarra	Bastão universal que possui uma garra.
isoladorVelho	Isolador de pedestal danificado a ser substituído
isoladorNovo	Isolador de pedestal novo que será posicionado no local do isolador velho.
torre	Estrutura metálica onde fica fixo o isolador
condutor	Em formato de cabo, fica fixo sobre o topo do isolador.e é por onde passa grandes quantidades de energia elétrica.
estropo	Pano firme usado para segurar Isolador quando estiver suspenso
pano	Pano usado para limpar ferramentas
glicerina	Substância usada para limpar as ferramentas adequadamente
condutímetro	Medidor de corrente de fuga sobre o bastão universal.
parafuso	Parafusos prendem o conector condutor-Isolador e também prendem o Isolador a base
conector	Estrutura que tem como por finalidade manter condutor,cabeçote do isolador em conjunto.

Tabela 6 – Definição de todos os artefatos presentes na manutenção

As etapas da atividade anteriormente posta foram analisadas em conjunto com engenheiros da área e foram estruturadas com base nos objetivos expostos na tabela 7. Essa tabela também apresenta as especificações para o predicado  $nextGoal(g_i, g_j)$  onde *Objetivo* representa  $g_i$ , *Próximo*  $g_j$  e *Descrição* é referente a  $g_i$ .

Objetivo	Próximo	Descrição
gSupervisor	g1,g6	Atribui objetivos aos demais agentes.
g0	gSupervisor	Vestir os EP'Is
g1	g2	Limpar, secar e testar ferramentas com material isolante.
g2	g3	Medir a corrente de fuga de ferramentas isolantes
g3	g4	Instalar sela com colar na estrutura
g4	g5	Passar o bastão garra por dentro do olhal do colar.
g5	g12	Amarrar o bastão garra na parte superior da estrutura com a corda, fixar no condutor
g6	g7	Amarrar o olhal do bastão garra ao cavalo da sela atrás de uma corda.
g7	g8	Instalar sela com colar no outro lado da estrutura estrutura
g8	g9	Passar o bastão universal por dentro do olhal do colar
g10	g11	Pender carretilha no bastão Universal.
g11	g12	Amarrar o bastão universal na parte superior da estrutura com a corda:
g12	g13	Rotacionar estrutura olhal garra em 45 graus.
g13	g14	Enforçar um estropo de Náilon no corpo do isolador velho.
g14	g15	Colocar a extremidade do estropo no gancho da corda de serviço.
g15	g16	Afrouxar os parafusos do conector que prendem a barra ao isolador.
g16	g17	Terminar de retirar os parafusos com o bastão com o soquete multian-
		gular.
g17	g18	Elevar o condutor através da corda que une a sela ao bastão.
g18	g19	Apertar o colar através da porca borboleta.
g19	g20	Sacar parafusos da base da coluna.
g20	g21	Segurar firmemente a corda de serviço,baixar o isolador ao solo
g21	g22	Passar Estropo no Isolador Novo
g22	g23	Colocar a extremidade do estropo no gancho da corda de serviço.
g23	g24	Içar o Isolador
g24	g25	Colocar Parafusos na base da coluna.
g25	g26	Baixar o condutor para que a mesma se sustente no novo isolador.
g26	g27	Colocar os parafusos do conector que prende a barra ao novo isolador.
g27		Retirar Equipamentos

Tabela 7 – Definição e descrição dos objetivos bem como dos respectivos pré-requisitos

### 5.1.3 Especificação das Condições e Relações

A tabela 8 apresenta  $c_k$  dado pela coluna condição e pela coluna descrição. Essa tabela define  $hasRisk(c_k, risk_j, cs_m)$  onde  $risk_j$  é descrito pela coluna risco e  $cs_m$  é descrito como consequência.

condição	descrição	risco	consequência
umidade70	Umidade Relativa do Ar deve ser inferior a setenta por cento.	eletrocutado	morte
noVento	Não deve haver vento durante os procedimentos de manutenção.	eletrocutado	morte
noChuva	Não deve haver chuva durante o ato da manutenção	eletrocutado	morte
sol	O dia deve estar ensolarado	eletrocutado	morte

Tabela 8 – Define as condições necessárias para que a manutenção tenha possibilidade de acontecer

A tabela 9 apresenta a especificação para dois predicados onde um deles é  $possEntityRel(r_l, e_i, e_k)$  tal que  $r_l$  é definido pela coluna *relacionamento*,  $e_i$  e  $e_k$  pelas *entidades envolvidas*. O outro predicado é dado por  $hasRisk(r_k, risk_j, cs_m)$  onde  $risk_j$  é dado pela coluna risco e  $cs_m$  é dado pela coluna consequência. Algumas relações (instâncias do conjunto *Relation*) serão apresentadas

usando o termo  $X$ . O objetivo disto consiste em tornar as tabelas mais enxutas por intermédio de uma regra a qual é:

A variável  $X$  deve ser substituída pelo agente que tem a permissão de executar alguma ação em certo objetivo em prol a sua função. Essa regra pode ser sintetizada na seguinte expressão para um agente  $ag_n$  que é referenciado por  $AGENT$ :

$$\begin{aligned} &adoptsRole(AGENT, \rho_m) \wedge hasPermission(\rho_m, g_i) \wedge requiresEntity(g_i, otherEntity) \\ &\rightarrow requiresCirc(g_1, relAGENTotherEntity) \end{aligned} \quad (5.1)$$

Para mostrar como se dá o uso desta regra pode-se considerar um relacionamento  $relXCapacete$  entre  $X$  e  $capacete$  (essa relação será melhor descrita na tabela 9). Essa relação acontece no objetivo  $g_0$  onde o trabalhador deve colocar o capacete em sua cabeça, por conta disto se aplica a todos os agentes que representam esses profissionais. Nesta implementação, esses agentes são: *agente1*, *agente2*, *agente3*, *agente4*, *agente5*, *agente6* e *agente7*. Aplicando a regra 5.1 para esse caso, obtêm-se as seguintes relações: *relAgente1Capacete*, *relAgente2Capacete*, *relAgente3Capacete*, *relAgente4Capacete*, *relAgente5Capacete*, *relAgente6Capacete* e *relAgente7Capacete*. Portanto, ao ler a tabela 9:

$$relXCapacete|X, capacete| \quad (5.2)$$

Para o objetivo  $g_0$ , essa linha é equivalente a:

$$\begin{aligned} &relAgente1Capacete|Agente1, capacete| \\ &relAgente2Capacete|Agente2, capacete| \\ &relAgente3Capacete|Agente3, capacete| \\ &relAgente4Capacete|Agente4, capacete| \\ &relAgente5Capacete|Agente5, capacete| \\ &relAgente6Capacete|Agente6, capacete| \\ &relAgente7Capacete|Agente7, capacete| \end{aligned} \quad (5.3)$$

É digno de nota observar que a regra 5.1 não faz parte da estrutura do modelo. A sua existência se deve única e exclusivamente neste estudo de caso a fim de simplificar e otimizar o grande volume de informações repetitivas.

<b>relacionamento</b>	<b>entidades envolvidas</b>	<b>risco</b>	<b>consequência</b>
relXCapacete	X,capacete	nenhum	nenhum
relXOculos	X,oculos	nenhum	nenhum
relXRoupagem	X,roupagem	nenhum	nenhum
relXLuva	X,luva	nenhum	nenhum
relXBotas	X,bota	nenhum	nenhum
relXPano	X,pano	nenhum	nenhum
relPanoGlicerina	pano,glicerina	nenhum	nenhum
relPanoCorda	pano,corda	nenhum	nenhum
relPanoBastaoUniversal	pano,bastaoUniversal	nenhum	nenhum
relPanoSoquete	pano,soquete	nenhum	nenhum
relPanoBastaoUniversal	pano,bastaoGarra	nenhum	nenhum
relXSela	X,sela	nenhum	nenhum
relXColar	X,colar	nenhum	nenhum
relXBastaoGarra	X,bastaoGarra	nenhum	nenhum
relTorreSela	torre,sela	nenhum	nenhum
relSelaColar	sela,colar	nenhum	nenhum
relColarBastaoGarra	colar,bastaoGarra	nenhum	nenhum
relBastaoGarraCondutor	bastaoGarra,condutor	eletrocutado	morte
relXBastaoUniversal	X,bastaoUniversal	nenhum	nenhum
relCordaBastaoUniversal	corda,bastaoUniversal	nenhum	nenhum
relCordaCarretilha	corda,carretilha	nenhum	nenhum
relBastaoUniversalCarretilha	bastaoUniversal,carretilha	nenhum	nenhum
relBastaoUniversalColar	bastaoUniversal,colar	nenhum	nenhum
relBastaoUniversalEstopo	bastaoUniversal,estopo	nenhum	nenhum
relCordaEstopo	corda,estopo	eletrocutado	morte
relEstopoIsoladorVelho	estopo,isoladorVelho	nenhum	nenhum
relXChaveCatraca	X,chaveCatraca	nenhum	nenhum
relChaveCatracaBastaoUniversal	chaveCatraca,bastaoUniversal	nenhum	nenhum
relChaveCatracaParafuso	chaveCatraca,parafuso	eletrocutado	morte
relParafusoConector	parafuso,conector	eletrocutado	morte
relXBastaoSoquete	X,bastaoSoquete	nenhum	nenhum
relSoqueteParafuso	soquete,parafuso	eletrocutado	morte
relXCorda	X,corda	eletrocutado	morte
relXIsoladorVelho	X,isoladorVelho	nenhum	nenhum
relXIsoladorNovo	X,isoladorNovo	nenhum	nenhum
relCordaBastaoGarra	corda,bastaoGarra	nenhum	nenhum

relBastaoGarraSela	bastaoGarra, sela	nenhum	nenhum
relXCarretilha	X,carretilha	nenhum	nenhum
relBastaoUniversalCorda	bastaoUniversal,corda	nenhum	nenhum
relBastaoUniversalTorre	bastaoUniversal,torre	nenhum	nenhum
relEstropoCorda	estropo,corda	eletrocutado	morte
relEstropoIsoladorNovo	estropo,isoladorNovo	nenhum	nenhum
relBastaoUniversalSela	universal,sela	nenhum	nenhum
relBastaoGarraTorre	bastaoGarra,torre	nenhum	nenhum
relBastaoUniversalEstropo	bastaoUniversal,estropo	nenhum	nenhum
relXColar	X,colar	nenhum	nenhum
relParafusoTorre	parafuso,torre	eletrocutado	morte
relCondutivimetroCorda	condutímetro,corda	nenhum	nenhum
relCondutivimetroBastaoUniversal	condutímetro,bastaoUniversal	nenhum	nenhum
relCondutivimetroBastaoGarra	condutímetro,bastaoGarra	nenhum	nenhum
relCondutivimetroSoquete	condutímetro,soquete	nenhum	nenhum

Tabela 9 – Descrição das entidades em função das relações

Tendo em vista o texto presente em 4.2.2, todos os elementos de *Relation* bem como de *Condition* também são elementos de *Circumstance*.

A Tabela 10 apresenta a relação  $affectsRels(r_k, r_n)$  onde  $r_k$  é representado pela coluna relacionamento-errado e  $r_n$  é representado pela coluna relacionamento-afetado.

<b>Relacionamento-errado</b>	<b>Relacionamento-afetado</b>
relXCapacete	relBastaoGarraCondutor
relXCapacete	relCordaEstropo
relXCapacete	relChaveCatracaParafuso
relXCapacete	relParafusoConector
relXCapacete	relSoqueteParafuso
relXCapacete	relXCorda
relXCapacete	relEstropoCorda
relXOculos	relBastaoGarraCondutor
relXOculos	relCordaEstropo
relXOculos	relChaveCatracaParafuso
relXOculos	relParafusoConector
relXOculos	relSoqueteParafuso

relXOculos	relXCorda
relXOculos	relEstropoCorda
relXLuva	relBastaoGarraCondutor
relXLuva	relCordaEstropo
relXLuva	relChaveCatracaParafuso
relXLuva	relParafusoConector
relXLuva	relSoqueteParafuso
relXLuva	relXCorda
relXLuva	relEstropoCorda
relXBotas	relBastaoGarraCondutor
relXBotas	relCordaEstropo
relXBotas	relChaveCatracaParafuso
relXBotas	relParafusoConector
relXBotas	relSoqueteParafuso
relXBotas	relXCorda
relXBotas	relEstropoCorda
relXPano	relBastaoGarraCondutor
relXPano	relCordaEstropo
relXPano	relChaveCatracaParafuso
relXPano	relParafusoConector
relXPano	relSoqueteParafuso
relXPano	relXCorda
relXPano	relEstropoCorda
relPanoGlicerina	relBastaoGarraCondutor
relPanoGlicerina	relCordaEstropo
relPanoGlicerina	relChaveCatracaParafuso
relPanoGlicerina	relParafusoConector
relPanoGlicerina	relSoqueteParafuso
relPanoGlicerina	relXCorda
relPanoGlicerina	relEstropoCorda
relPanoCorda	relCordaEstropo
relPanoCorda	relXCorda
relPanoCorda	relEstropoCorda
relPanoBastaoUniversal	relBastaoGarraCondutor
relPanoBastaoUniversal	relChaveCatracaParafuso
relPanoBastaoUniversal	relParafusoConector
relPanoBastaoUniversal	relBastaoGarraCondutor



relPanoSoquete	relBastaoGarraCondutor
relPanoSoquete	relCordaEstropo
relPanoSoquete	relChaveCatracaParafuso
relPanoSoquete	relParafusoConector
relPanoSoquete	relSoqueteParafuso
relPanoSoquete	relXCorda
relPanoSoquete	relEstropoCorda
relCondutivimetroCorda	relBastaoGarraCondutor
relCondutivimetroCorda	relCordaEstropo
relCondutivimetroCorda	relChaveCatracaParafuso
relCondutivimetroCorda	relParafusoConector
relCondutivimetroCorda	relSoqueteParafuso
relCondutivimetroCorda	relXCorda
relCondutivimetroCorda	relEstropoCorda
relCondutivimetroCorda	relParafusoTorre
relPanoBastaoUniversal	relBastaoGarraCondutor
relPanoBastaoUniversal	relChaveCatracaParafuso
relPanoBastaoUniversal	relParafusoConector
relPanoBastaoUniversal	relParafusoTorre
relPanoBastaoUniversal	relBastaoGarraCondutor
relPanoSoquete	relBastaoGarraCondutor
relPanoSoquete	relCordaEstropo
relPanoSoquete	relChaveCatracaParafuso
relPanoSoquete	relParafusoConector
relPanoSoquete	relSoqueteParafuso
relPanoSoquete	relXCorda
relPanoSoquete	relEstropoCorda
relPanoSoquete	relParafusoTorre
relCondutivimetroCorda	relBastaoGarraCondutor
relCondutivimetroCorda	relCordaEstropo
relCondutivimetroCorda	relChaveCatracaParafuso
relCondutivimetroCorda	relParafusoConector
relCondutivimetroCorda	relSoqueteParafuso
relCondutivimetroCorda	relXCorda
relCondutivimetroCorda	relEstropoCorda
relCondutivimetroCorda	relParafusoTorre

---

Tabela 10 – Define o impacto que o erro em um relacionamento gera em outro relacionamento por mudar a possibilidade de algo errado acontecer.

#### 5.1.4 Relação entre Objetivos e Papéis

A tabela 11 apresentam a relação  $hasObligation(\rho_m, g_i)$  onde  $\rho_m$  é representado pela coluna papel e  $g_i$  é representado pela coluna objetivo.

Papel	Objetivo
executor1	g0
executor2	g0
executor3	g0
executor4	g0
executor5	g0
supervisor	g0
supervisor	gSupervisor
executor1	g1
executor2	g1
executor1	g2
executor2	g2
executor1	g3
executor2	g2
executor1	g4
executor2	g4
executor1	g5
executor2	g5
executor3	g6
executor4	g6
executor5	g6
executor3	g7
executor4	g7
executor5	g7
executor3	g8
executor4	g8
executor5	g8
executor3	g9

executor4	g9
executor5	g9
executor3	g10
executor4	g10
executor5	g10
executor3	g11
executor4	g11
executor5	g11
executor1	g12
executor2	g12
executor3	g12
executor4	g12
executor1	g13
executor2	g13
executor3	g13
executor4	g13
executor1	g14
executor2	g14
executor3	g14
executor4	g14
executor2	g15
executor3	g15
executor4	g15
executor5	g15
executor2	g16
executor3	g16
executor4	g16
executor5	g16
executor1	g17
executor3	g17
executor4	g17
executor5	g17
executor1	g18
executor3	g18
executor4	g18
executor5	g18
executor1	g19

executor3	g19
executor4	g19
executor5	g19
executor1	g20
executor3	g20
executor4	g20
executor5	g20
executor1	g21
executor3	g21
executor4	g21
executor5	g21
executor1	g22
executor2	g22
executor3	g22
executor5	g22
executor1	g23
executor2	g23
executor3	g23
executor5	g23
executor1	g24
executor2	g24
executor3	g24
executor5	g24
executor1	g25
executor2	g25
executor3	g25
executor4	g25
executor1	g26
executor2	g26
executor3	g26
executor4	g26
executor1	g27
executor2	g27
executor3	g27
executor4	g27
executor5	g27

Tabela 11 – Objetivos que devem ser atingidos pelo agente que assumir um dada função

### 5.1.5 Relacionamentos das Entidades, Relações e Condições com Objetivos

As próximas tabelas têm como finalidade apresentar como se dá a especificação para os seguintes predicados:  $requiresCirc(goal_n, circ_m), requiresEntity(goal_n, ent_m)$ . Contudo, existe uma série de circunstâncias  $circ \in Circumstance$  que apontam para o mesmo objetivo  $g \in Goal$ . Para tornar a apresentação desses resultados mais simples, o autor optou por agrupar todas as circunstâncias que se relacionam na mesma célula na coluna esquerda. A coluna direita contém o respectivo objetivo com o qual essas circunstâncias se relacionam.

Por exemplo, suponha que em uma determinada situação, um objetivo  $g1$  necessita a presença das circunstâncias  $circ_a, circ_b$  e  $circ_c$ . De acordo com o modelo, essa situação é resolvida da seguinte maneira:  $requiresCirc(g1, circ_a), requiresCirc(g1, circ_b)$  e  $requiresCirc(g1, circ_c)$ . Porém, para economizar linha em tabelas, o autor optou por representar esse cenário como na tabela 12:

Circunstância	Tipo de Instância	Objetivo
$circ_a, circ_b, circ_c$	instanceOfRel	$g1$

Tabela 12 – Exemplificação dos dados atrelados ao predicado  $requiresCirc(goal_n, circ_m), requiresEntity(goal_n, ent_m)$  serão organizados nas tabelas a seguir.

A coluna *Circunstância* apresenta todas as circunstâncias atreladas ao objetivo  $g1$  descritas pelos predicados  $requiresCirc$  expostos um pouco antes apresentar a tabela 12. A coluna *Tipo de Instância* define se a circunstância é uma instância de *Relation* ou se é uma instância de *Condition*. Assim sendo, ao analisar a tabela 12 pode-se concluir que  $circ_a, circ_b$  e  $circ_c$  são todos *Relations*. A coluna *Objetivo* apresenta o objetivo no qual essas circunstâncias estão relacionadas. A Tabela 13 apresenta essa mesma estrutura, porém aplicadas ao estudo de caso em interesse.

Relacionamentos	Tipo de Instância	Objetivo
relXcapacete relXoculos relXroupa- gem relXluva relXbotas	instanceOfRel	$g0$
relXPano, relPanoGlicerina, relPano- Corda, relPanoBastaoUniversal, rel- PanoBastaoGarra, relPanoSoquete	instanceOfRel	$g1$
umidade70,noVento,noChuva,sol	instanceOfCond	$g1$
relCondutivimetroCorda, relConduti- vimetroBastaoUniversal, relConduti- vimetroBastaoGarra, relCondutivime- troSoquete	instanceOfRel	$g2$

relCondutivimetro	instanceOfRel	g3
relXBastaoGarra, relColarBastao-Garra	instanceOfRel	g4
relXBastaoGarra, relXCordarelCordaBastaoGarra, relBastaoGarraTorre, relBastaoGarraCondutor	instanceOfRel	g5
relBastaoGarraSela, relXBastao-Garra, relXSela	instanceOfRel	g6
relXSela, relXColar, relTorreSela	instanceOfRel	g7
relBastaoUniversalColar, relXBastaoUniversal	instanceOfRel	g8
relXBastaoUniversal, relXCarretilha, relBastaoUniversalCarretilha	instanceOfRel	g9
relXCorda, relXBastaoUniversal, relBastaoUniversalCorda, relBastaoUniversalTorre	instanceOfRel	g10
relXCorda, relXBastaoUniversal, relXColar, relBastaoUniversalColar, relBastaoUniversalSela	instanceOfRel	g11
relXColar	instanceOfRel	g12
relXBastaoUniversal, relBastaoUniversalEstropo, relEstropoIsoladorVelho	instanceOfRel	g13
relXBastaoUniversal, relBastaoUniversalCordarelCordaEstropo, relEstropoCorda	instanceOfRel	g14
relChaveCatracaBastaoUniversal, relXChaveCatraca, relXBastaoUniversal, relChaveCatracaParafuso	instanceOfRel	g15
relXBastaoSoquete relSoqueteParafuso	instanceOfRel	g16
relXCorda relCordaBastaoGarra, relBastaoGarraCondutor	instanceOfRel	g17
relXColar	instanceOfRel	g18

relChaveCatracaBastaoUniversal, relXChaveCatraca, relXBastaoU- niversal, relChaveCatracaParafuso, relParafusoTorre, relXBastaoSoque- terelSoqueteParafuso	instanceOfRel	g19
relXCorda	instanceOfRel	g20
relXEstropo, relEstropoIsoladorNovo	instanceOfRel	g21
relXBastaoUniversal, relBastaoU- niversalCorda, relCordaEstropo, relEstropoCorda	instanceOfRel	g22
relXCorda	instanceOfRel	g23
relChaveCatracaBastaoUniversal, relXChaveCatraca, relXBastaoU- niversal, relChaveCatracaParafuso, relParafusoTorrerelXBastaoSoquete, relSoqueteParafuso	instanceOfRel	g24
relXCorda, relCordaBastaoGarra, rel- BastaoGarraCondutor	instanceOfRel	g25
relChaveCatracaBastaoUniversal, relXChaveCatraca, relXBastaoUni- versal, relChaveCatracaParafuso	instanceOfRel	g26
relXSela, relXColarrelXBastaoGar- rarelXBastaoUniversal, relXBastao- Soquete, relXCorda, relXCarretilha, relXChaveCatraca, relColarBas- taoGarra, relCordaBastaoGarra, relBastaoGarraTorre, relBastao- GarraCondutor, relBastaoUniver- salCarretilha, relBastaoGarraSela, relBastaoUniversalSela, relSelaColar, relTorreSela, relBastaoUniversal- Corda, relBastaoGarraCorda	instanceOfRel	g27

Tabela 13 – Especificação do predicado  $requiresCirc(goal_i, circ_j)$ , do predicado  $instanceOfRel(circ_n)$  e do predicado  $instanceOfCond(circ_n)$

A Tabela 14 apresenta os dados em relação ao predicado  $requiresEntity(goal_i, e_j)$  e

segue o mesmo padrão apresentado pelo exemplo na tabela 12. Contudo, tendo em vista o fato de que está implícito no conceito do predicado  $requiresEntity(goal_i, e_j)$  a natureza da instância atrelada ao segundo argumento (ou seja, não há dúvida que  $e_j$  é uma entidade e está contida em  $Entity$ ), a Tabela 14 não apresenta coluna *Tipo de Instância*.

Entidades	Objetivo
capacete,óculos,roupagem,luvas,botas X = {agentes em relação aos objetivos}	g0
pano,glicerina,carretilha,bastaoUniversal,corda,bastaoGarra,X = {agentes em relação aos objetivos}	g1
pano,glicerina,carretilha,bastaoUniversal,corda,bastaoGarra,condutímetro,X = {agentes em relação aos objetivos}	g2
sela,colarX = {agentes em relação aos objetivos}	g3
colar,bastaoGarraX = {agentes em relação aos objetivos}	g4
corda,bastaoGarra,bastaoGarraTorre,condutorX = {agentes em relação aos objetivos}	g5
bastaoGarra,selaX = {agentes em relação aos objetivos}	g6
sela,colarX = {agentes em relação aos objetivos}	g7
sela,bastaoUniversal,Colar,X = {agentes em relação aos objetivos}	g8
bastaoUniversal,carretilha,X = {agentes em relação aos objetivos}	g9
corda,bastaoUniversal,corda,torre,X = {agentes em relação aos objetivos}	g10
bastaoUniversal,corda,colar,selaX = {agentes em relação aos objetivos}	g11
colar,X = {agentes em relação aos objetivos}	g12
bastaoUniversal,estropo,isoladorVelhoX = {agentes em relação aos objetivos}	g13
bastaoUniversal,corda,estropoX = {agentes em relação aos objetivos}	g14
chaveCatraca,bastaoUniversal,parafusoX = {agentes em relação aos objetivos}	g15
bastaoSoquete,parafuso,X = {agentes em relação aos objetivos}	g16
bastaoGarra,condutorcordaX = {agentes em relação aos objetivos},	g17
colar,X = {agentes em relação aos objetivos},	g18
chaveCatraca,bastaoUniversal,prafusobastaoSoquete,parafuso,torreX = {agentes em relação aos objetivos}	g19
cordaX = {agentes em relação aos objetivos}	g20
estropo, isoladorNovo,X = {agentes em relação aos objetivos}	g21
bastaoUniversal,corda,estropoX = {agentes em relação aos objetivos}	g22
cordaX = {agentes em relação aos objetivos}	g23
chaveCatraca,bastaoUniversal,prafusobastaoSoquete,parafuso,torreX = {agentes em relação aos objetivos}	g24



bastaoGarra,condutorcordaX = {agentes em relação aos objetivos},	g25
chaveCatraca,bastaoUniversal,prafusoX = {agentes em relação aos objetivos}	g26
sela, colar, bastaoGarra, bastaoUniversal, bastaoSoquete, corda, carretilha, chaveCatraca, torre, condutor	g27

Tabela 14 – Especificação do estudo de caso atrelado ao predicado  $requiresEntity(goal_i, e_j)$

## 5.2 Raciocínio

Uma vez que o modelo foi definido e que foi implementado em um estudo de caso, é possível avaliar as conclusões possíveis em certa condição de mundo. Essa seção demonstra como esse modelo cumpre o proposto por demonstrar certos raciocínios tendo em vista o estudo de caso em análise.

### 5.2.1 Raciocínio - 1

O raciocínio a seguir mostra o que acontece se o *agente4* esquecer de passar glicerina no pano, especificado pela relação *relPanoGlicerina*, designados a ele no objetivo g1. Todos os predicados vinculados a essa situação são:

1.  $adoptsRole(agente4, executor2)$
2.  $hasObligation(executor2, g1)$
3.  $requiresCirc(g1, relPanoGlicerina)$
4.  $instanceOfRel(relPanoGlicerina)$
5.  $starts(agente4, g1)$
6.  $affectsRels(relPanoGlicerina, relBastaoGarraCondutor)$
7.  $affectsRels(relPanoGlicerina, relCordaEstropo)$
8.  $affectsRels(relPanoGlicerina, relChaveCatracaPara fuso)$
9.  $affectsRels(relPanoGlicerina, relPara fusoConector)$
10.  $affectsRels(relPanoGlicerina, relSoquetePara fuso)$
11.  $affectsRels(relPanoGlicerina, relAgente4Corda)$

12. *affectsRels*(*relPanoGlicerina*, *relEstropoCorda*)

Com base nisso, as relações de implicabilidade resultantes são:

$$\begin{aligned}
 &requiresCirc(g1, relPanoGlicerina) \wedge \\
 &\neg isPresent(relPanoGlicerina) \wedge \\
 &instanceOfRel(relPanoGlicerina) \wedge \\
 &\quad starts(agente4, g1) \rightarrow \\
 &relationViol(agente4, g1, relPanoGlicerina)
 \end{aligned} \tag{5.4}$$

$$\begin{aligned}
 &relationViol(agente4, g1, relPanoGlicerina) \\
 &\wedge affectsRels(relPanoGlicerina, relBastaoGarraCondutor) \\
 &\rightarrow possOfNegConseqFor(relBastaoGarraCondutor)
 \end{aligned} \tag{5.5}$$

$$\begin{aligned}
 &relationViol(agente4, g1, relPanoGlicerina) \\
 &\wedge affectsRels(relPanoGlicerina, relCordaEstropo) \\
 &\rightarrow possOfNegConseqFor(relCordaEstropo)
 \end{aligned} \tag{5.6}$$

$$\begin{aligned}
 &relationViol(agente4, g1, relPanoGlicerina) \\
 &\wedge affectsRels(relPanoGlicerina, relParafusoConector) \\
 &\rightarrow possOfNegConseqFor(relParafusoConector)
 \end{aligned} \tag{5.7}$$

$$\begin{aligned}
 &relationViol(agente4, g1, relPanoGlicerina) \\
 &\wedge affectsRels(relPanoGlicerina, relSoqueteParafuso) \\
 &\rightarrow possOfNegConseqFor(relSoqueteParafuso)
 \end{aligned} \tag{5.8}$$

$$\begin{aligned}
 &relationViol(agente4, g1, relPanoGlicerina) \\
 &\wedge affectsRels(relPanoGlicerina, relAgente4Corda) \\
 &\rightarrow possOfNegConseqFor(relAgente4Corda)
 \end{aligned} \tag{5.9}$$

$$\begin{aligned}
& relationViol(agent4, g1, relPanoGlicerina) \\
& \wedge affectsRels(relPanoGlicerina, relEstropoCorda) \\
& \rightarrow possOfNegConseqFor(relEstropoCorda)
\end{aligned} \tag{5.10}$$

### 5.2.2 Raciocínio - 2

O raciocínio a seguir mostra o que acontece se o pano não estiver presente no local da manutenção quando os eletricitas alcançarem o g1. A lista a seguir exhibe todos os predicados necessários para averiguar essa condição de mundo.

1.  $adoptsRole(agent2, executor1)$
2.  $adoptsRole(agent3, executor1)$
3.  $adoptsRole(agent4, executor2)$
4.  $hasObligation(executor1, g1)$
5.  $hasObligation(executor2, g1)$
6.  $starts(agent2, g1)$
7.  $starts(agent3, g1)$
8.  $starts(agent4, g1)$
9.  $requiresEntity(g1, pano)$
10.  $\neg isPresent(pano)$

$$\begin{aligned}
& requiresEntity(g1, pano) \wedge \\
& \neg isPresent(pano) \wedge \\
& starts(agent2, g1) \rightarrow \\
& entityViol(agent2, g1, pano)
\end{aligned} \tag{5.11}$$

$$\begin{aligned}
& requiresEntity(g1, pano) \wedge \\
& \neg isPresent(pano) \wedge \\
& starts(agente3, g1) \rightarrow \\
& entityViol(agente3, g1, pano)
\end{aligned}
\tag{5.12}$$

$$\begin{aligned}
& requiresEntity(g1, pano) \wedge \\
& \neg isPresent(pano) \wedge \\
& starts(agente4, g1) \rightarrow \\
& entityViol(agente4, g1, pano)
\end{aligned}
\tag{5.13}$$

$$entityViol(agente4, g1, pano) \rightarrow stopped(g1) \tag{5.14}$$

### 5.2.3 Raciocínio - 3

O raciocínio a seguir mostra o que acontece se o *agente5* tentar alcançar o objetivo *g11* com a umidade relativa do ar superior a setenta por cento. A lista a seguir exhibe todos os predicados necessários para averiguar essa condição de mundo.

1. *adoptsRole(agente5, executor3)*
2. *hasObligation(executor3, g11)*
3. *starts(agente5, g11)*
4. *requiresCirc(g11, umidade70)*
5. *isInstanceOfCond(umidade70)*
6. *¬isPresent(umidade70)*
7. *hasRisk(umidade70, eletrocutado, morte)*

$$\text{requiresCirc}(g11, \text{umidade70}) \quad (5.15)$$

$$\begin{aligned} & \wedge \neg \text{isPresent}(\text{umidade70}) \\ & \wedge \text{instanceOfCond}(\text{umidade70}) \\ & \wedge \text{starts}(\text{agente5}, g11) \rightarrow \\ & \text{conditionViol}(\text{agente5}, g11, \text{umidade70}) \end{aligned} \quad (5.16)$$

$$\begin{aligned} & \text{conditionViol}(\text{agente5}, g11, \text{umidade70}) \\ & \wedge \text{hasRisk}(\text{umidade70}, \text{eletrocutado}, \text{morte}) \rightarrow \\ & \text{negConseqFor}(g11, \text{agente5}, \text{eletrocutado}, \text{morte}) \end{aligned} \quad (5.17)$$

$$\text{negConseqFor}(g11, \text{agente5}, \text{eletrocutado}, \text{morte}) \rightarrow \text{stopped}(g11) \quad (5.18)$$

#### 5.2.4 Raciocínio - 4

O raciocínio a seguir mostra o que acontece se o *agente3* errar a forma adequada de realizar o relacionamento *relChaveCatracaPara fuso* no objetivo *g15*. Os predicados envolvidos são:

1. *adoptsRole*(*agente4*, *executor2*)
2. *hasObligation*(*executor4*, *g15*)
3. *starts*(*agente4*, *g15*)
4. *requiresCirc*(*g15*, *relChaveCatracaPara fuso*)
5. *isInstanceOfRel*(*relChaveCatracaPara fuso*)
6.  $\neg \text{isPresent}(\text{relChaveCatracaPara fuso})$
7. *hasRisk*(*relChaveCatracaPara fuso*, *eletrocutado*, *morte*)

$$\begin{aligned}
& requiresCirc(g15, relChaveCatracaParafuso) \wedge \\
& \neg isPresent(relChaveCatracaParafuso) \wedge \\
& instanceOfRel(relChaveCatracaParafuso) \wedge \\
& starts(agente4, g15) \rightarrow \\
& relationViol(agente4, g15, relChaveCatracaParafuso)
\end{aligned}$$

$$\begin{aligned}
& relationViol(agente4, g15, relChaveCatracaParafuso) \\
& \wedge hasRisk(relChaveCatracaParafuso, eletrocutado, morte) \\
& \rightarrow \\
& negConseqFor(g15, agente4, eletrocutado, morte)
\end{aligned} \tag{5.19}$$

$$negConseqFor(g15, agente4, eletrocutado, morte) \rightarrow stopped(g15) \tag{5.20}$$

### 5.2.5 Raciocínio - 5

A finalidade dessa demonstração consiste em mostrar como um agente pode ser submetido a consequências ruins tendo em vista os erros cometidos por outros profissionais. O raciocínio 1 mostra que o fato do *agente4* não conseguir realizar o relacionamento *relPanoGlicerina* resulta na violação *relationViol(agente4, g1, relPanoGlicerina)*. Essa violação, por sua vez, impacta diversas outras relações, em que *possOfNegConseqFor(relParafusoConector)* é uma delas. Assim sendo, antes do *agente4* cometer o erro, a possibilidade da ocorrência de um evento ruim acontecer era 0, se o agente realizar a relação *relParafusoConector* sem cometer violação alguma. Contudo, após a ocorrência do erro cometido pelo *agente4*, existe uma possibilidade de um evento ruim acontecer na relação *relParafusoConector* mesmo que tudo seja feito de acordo com os conformes. Assim sendo, a lista de predicados e o raciocínio mostra o que acontece dado a seguinte situação: o possível evento ruim presente em *relParafusoConector* se torna uma realidade:

1. *requiresCirc(g19, relParafusoConector)*
2. *hasObligation(executor3, g19)*
3. *hasObligation(executor4, g19)*
4. *hasObligation(executor5, g19)*

5.  $starts(agente5, g19)$
6.  $starts(agente6, g19)$
7.  $starts(agente7, g19)$
8.  $adoptsRole(agente5, executor3)$
9.  $adoptsRole(agente6, executor4)$
10.  $adoptsRole(agente7, executor5)$
11.  $hasRisk(relParafusoConector, eletrocutado, morte)$
12.  $possOfNegConseqFor(relParafusoConector)$
13.  $happensNegConseqFor(relParafusoConector)$

$$\begin{aligned}
& possOfNegConseqFor(relParafusoConector) \\
& \wedge happensNegConseqFor(relParafusoConector) \\
& \wedge requiresCirc(g19, relParafusoConector) \\
& \wedge instanceOfRel(relParafusoConector) \\
& \wedge hasRisk(relParafusoConector, eletrocutado, morte) \\
& \wedge starts(agente5, g19) \\
& \rightarrow negConseqFor(g19, agente5, eletrocutado, morte)
\end{aligned} \tag{5.21}$$

$$negConseqFor(g19, agente5, eletrocutado, morte) \rightarrow stopped(g19) \tag{5.22}$$

$$\begin{aligned}
& possOfNegConseqFor(relParafusoConector) \\
& \wedge happensNegConseqFor(relParafusoConector) \\
& \wedge requiresCirc(g19, relParafusoConector) \\
& \wedge instanceOfRel(relParafusoConector) \\
& \wedge hasRisk(relParafusoConector, eletrocutado, morte) \\
& \wedge starts(agente6, g19) \\
& \rightarrow negConseqFor(g19, agente6, eletrocutado, morte)
\end{aligned} \tag{5.23}$$

$$\text{negConseqFor}(g19, \text{agente6}, \text{eletrocutado}, \text{morte}) \rightarrow \text{stopped}(g19) \quad (5.24)$$

$$\begin{aligned} & \text{possOfNegConseqFor}(\text{relParaFusoConector}) \\ & \wedge \text{happensNegConseqFor}(\text{relParaFusoConector}) \\ & \wedge \text{requiresCirc}(g19, \text{relParaFusoConector}) \\ & \wedge \text{instanceOfRel}(\text{relParaFusoConector}) \\ & \wedge \text{hasRisk}(\text{relParaFusoConector}, \text{eletrocutado}, \text{morte}) \\ & \wedge \text{starts}(\text{agente7}, g19) \\ & \rightarrow \text{negConseqFor}(g19, \text{agente7}, \text{eletrocutado}, \text{morte}) \end{aligned} \quad (5.25)$$

$$\text{negConseqFor}(g19, \text{agente7}, \text{eletrocutado}, \text{morte}) \rightarrow \text{stopped}(g19) \quad (5.26)$$

#### 5.2.6 Raciocínio - 6

Esse raciocínio tem a finalidade de mostrar como se dá os raciocínios para quando se faz necessário verificar se um objetivo foi atingido. O objetivo *g23* deve ser atingido pelos agentes com as funções de *executor1*, *executor2*, *executor3* e *executor5*.

1.  $\text{stopped}(\text{agente2}, g23) \rightarrow F$
2.  $\text{stopped}(\text{agente3}, g23) \rightarrow F$
3.  $\text{stopped}(\text{agente4}, g23) \rightarrow F$
4.  $\text{stopped}(\text{agente5}, g23) \rightarrow F$
5.  $\text{stopped}(\text{agente7}, g23) \rightarrow F$
6.  $\text{hasObligation}(\text{executor1}, g23)$
7.  $\text{hasObligation}(\text{executor2}, g23)$
8.  $\text{hasObligation}(\text{executor3}, g23)$
9.  $\text{adoptsRole}(\text{agente2}, \text{executor1})$
10.  $\text{adoptsRole}(\text{agente3}, \text{executor1})$



11. *adoptsRole*(*agente4*,*executor2*)
12. *adoptsRole*(*agente5*,*executor3*)
13. *adoptsRole*(*agente7*,*executor5*)

Essa situação é resolvida pelo algoritmo presente em 8. A primeira etapa do algoritmo se dá por executar a função *ifNotStopped*(*agentArray*,*goal*). Um dos argumentos, portanto, é um vetor de *Agents*. Para essa situação em específico, esse vetor é descrito da seguinte forma  $ag_{array} = \{agente2, agente3, agente4, agente5, agente7\}$ . O argumento *goal* é carregado com *g23*. A primeira etapa da desta função reside avaliar todos os agentes (um por um) em um (*forEach*). Nessa avaliação é feito um teste sobre o predicado *stopped*(*ag*) que se retornar verdade faz com que *ifNotStopped*(*agentArray*,*goal*) retorne falso. Contudo, para o caso em análise, verificamos que o predicado *stopped*(*ag<sub>n</sub>*) retorna falso para todos os agentes. O segundo passo consiste na avaliação da função *allAgentObligate*(*agentArray*,*goal*) cujo propósito consiste verificar se todos os agentes que são obrigados a alcançar o objetivo em análise estão presentes. Nesse estudo de caso é possível verificar que os agentes 2,3,4,5 e 7 adotaram as funções de executor 1,2,3 e estas são obrigadas a executar o objetivo *g23*. Logo, para esse problema a função *allAgentObligate*(*agentArray*,*goal*) retorna verdade e, por consequência, a função *ifNotStopped*(*agentArray*,*goal*) retorna que *reached*(*g23*) é verdade.

#### 5.2.7 Raciocínio - 7

O raciocínio para o caso onde *agente1* tente alcançar o objetivo *g23*.

1. *adoptsRole*(*agente1*,*supervisor*)
2. *hasObligation*(*agente1*,*g23*)  $\rightarrow F$

Isso implica uma afirmação falsa, então esse mundo não é possível segundo o modelo implementado para este estudo de caso.

### 5.3 Implementação

O estudo de caso especificado pelo modelo conceitual proposto nesse estudo foi implementado em Prolog. A seguir segue um exemplo de como a regra 4.9 é escrita em Prolog.

*hasPermission*(*RHO*,*GOAL*) :- *hasObligation*(*RHO*,*GOAL*).

A seguir há um exemplo da especificação *adoptsRole*(*agente1*,*supervisor*) implementada em Prolog.

*adoptsRole*(*agente1*,*supervisor*).

As consultas, em *Prolog*, são feitas por escrever a especificação do implicador (*implicador*  $\rightarrow$  *implicado*). Usando o algoritmo *Backtracking*, é possível encontrar todos os predicados que são verdade. Por exemplo, para avaliar o Raciocínio 3 se fazem necessárias fazer as seguintes consultas:  $? - stopped(g1)$ ,  $? - entityViol(agente4, g1, pano)$ ,  $? - entityViol(agente3, g1, pano)$ ,  $? - entityViol(agent2, g1, pano)$ .

O apêndice B apresenta a implementação das regras e dos raciocínios em Prolog e, no caso do algoritmo, em JavaScript.

## 6 Discussão

Esse capítulo tem dois propósitos, sendo que o primeiro consiste discutir o modelo conceitual proposto nesse estudo em relação aos demais modelos e linguagens, feito na Seção 6.1. O segundo objetivo consiste em discutir a metodologia e os resultados e isso é feito na Seção 6.3.

### 6.1 Análise Comparativa

Uma vez estruturada a problemática da representação de cenários de acidentes em termos de um modelo computacional, é possível discutir semelhanças e diferenças de arcabouços existentes na literatura. Isso possibilita inferir implicações que ocorrem ao fazer uso de um arcabouço em específico (tendo o modelo conceitual desse estudo como parâmetro). Os arcabouços selecionados para comparação são: *MOISE+* (HÜBNER; SICHMAN; BOISSIER, 2002b), Modelo presente no texto do Dastani (DASTANI et al., 2009), o *V3S* (BAROT et al., 2013) e o modelo *NormMAS Framework* (CHANG; MENEGUZZI, 2016). Portanto, as próximas seções apresentam os seguintes aspectos: análise da estrutura do modelo para aqueles onde isso não foi feito na fundamentação, Tabelas que realizam uma análise comparativa entre os arcabouços selecionados e o modelo conceitual proposto (os atributos foram escolhidos caso a caso com base nas características de cada modelo) e uma análise única de todos os modelos em uma Tabela para avaliar a expressividade em certos atributos.

### 6.1.1 Modelo MOISE+

A Tabela 15 apresenta uma comparação entre o *MOISE+* e o modelo proposto neste estudo.

Atributos	MOISE+	Modelo deste Estudo
Finalidade	Representar a estrutura organizacional de um SMA: estrutural, funcional e deôntica	Representar cenários de Acidentes com base em uma estrutura organizacional similar ao Moise+
Agentes	Não representar os processos mentais do agente	Não representar os processos mentais do agente
Representação de objetivos	Baseado em uma teoria para administração de tarefas, possui uma sintaxe baseada em decomposição, sequência e paralelismo para representar objetivos	Possui uma estrutura simples para representar objetivos que avalia apenas questões de pré-requisitos. Permite representar sequências e paralelismos

Atributos	MOISE+	Modelo deste Estudo
Raciocínios	Possui relações atreladas a questões deônticas e a tipos de comunicações que os agentes podem fazer	Presença de regras que possibilitam reproduzir cenários de acidentes entre os agentes. Não apenas isso, mas faz parte da estrutura dessas regras questões atreladas a normas, violações e sanções. Esses raciocínios também representam cenários onde agentes sofrem consequências negativas mesmo não sendo responsáveis pelas causas dos acidentes e fazem isso por levar em consideração questões possibilísticas e influências entre relações
Normas	Faz uso de lógica deôntica para definir o papel do agente em relação ao objetivo	Faz uso de lógica deôntica para definir o papel do agente em relação ao objetivo, contudo também explora outras estruturas lógicas para determinar violações (quando o agente descumpra com o seu respectivo dever) e sanções (penalização por conta de uma violação)

<b>Atributos</b>	<b>MOISE+</b>	<b>Modelo deste Estudo</b>
SMA	Presença de conceitos como agentes, papéis, objetivos, missões e grupos bem como presença de operadores (compatibilidade, herança)	Presença de conceitos como papéis, agentes e objetivos
Estrutura da Linguagem	Lógica de Predicado de Primeira Ordem	Teoria de Conjuntos e Lógica de Predicados de Primeira Ordem

Tabela 15 – Comparação entre o *MOISE+* e o modelo proposto neste estudo

#### 6.1.2 Modelo DASTANI

A Tabela 16 apresenta uma comparação entre o *DASTANI+* e o modelo proposto neste estudo.

<b>Atributos</b>	<b>DASTANI</b>	<b>Modelo deste Estudo</b>
Finalidade	Modelo de <i>SMA</i> que leva em consideração questões atreladas à normas, sanções e violações	Representar cenários de acidentes
Agentes	Não define os estados mentais do agente	Não define os estados mentais do agente
Generalização	Trata de cenários onde se tem o interesse de representar agentes, normas, violações e sanções	Trata de cenários específicos para acidentes com a finalidade de identificar as causas destes

<b>Atributos</b>	<b>DASTANI</b>	<b>Modelo deste Estudo</b>
Normas	Presença de regras que permitem expressar violações e sanções para os mais diversos casos possíveis	Presença de regras que permitem expressar violações e sanções para cenários de acidentes
Estrutura da Linguagem	Linguagem de programação baseada em Lógica de Predicados de Primeira Ordem	Teoria de Conjuntos e Lógica de Predicados de Primeira Ordem
Raciocínios	Permite representar raciocínios que considera agentes cometendo ou não violações e tendo que lidar com as respectivas sanções. Esses raciocínios levam em consideração conceitos como eventos e ações (estados que geram a transição de eventos)	Presença de regras que possibilitam reproduzir cenários de acidentes entre os agentes. Não apenas isso, mas faz parte da estrutura dessas regras questões atreladas às normas, violações e sanções. Esses raciocínios também representam cenários onde agentes sofrem consequências negativas mesmo não sendo responsáveis pelas causas dos acidentes e fazem isso por levar em consideração questões possibilísticas e influências entre relações.

Tabela 16 – Comparação entre o *DASTANI+* e o modelo proposto neste estudo

## 6.2 Comparação entre VS3 e o modelo proposto neste estudo

A Tabela 17 apresentam uma comparação entre o VS3 e o modelo proposto neste estudo.

<b>Atributos</b>	<b>VS3</b>	<b>Modelo deste Estudo</b>
------------------	------------	----------------------------

Finalidade	Representar cenários de acidentes a fim de simulá-los com propósito de treinamento profissional	Representar cenários de acidentes
Artefatos-Objetos	Representa objetos através da classe (da ontologia atrelada ao módulo <i>Domain Model</i> ) <i>V3S-Object</i>	Representa objetos através do conjunto <i>Artefact</i>
Relações entre entidades	Representa as relações entre objetos através da classe (da ontologia atrelada ao módulo <i>Domain Model</i> ) <i>V3S-Action</i> através de relacionamentos com a classe <i>V3S-Object</i>	Representa as relações entre as entidades através do predicado $possEntityRel(r_l, e_i, e_k)$
Riscos e Acidentes	Modelo <i>MELISSA</i> (BATU E BCTU)	Uso de normas, violações, sanções e lógica possibilística aplicadas a contextos específicos por meio das regras <a href="#">4.10</a> , <a href="#">4.11</a> , <a href="#">4.12</a> , <a href="#">4.13</a> e <a href="#">4.14</a>
Agentes	Uso do <i>Framework</i> MAS-VERP	Não representar os processos mentais do agente
Estrutura da Linguagem	Diferentes formalismos da computação tais como: ontologias, lógica de primeira ordem, algoritmos e entre outros	Teoria de Conjuntos e Lógica de Predicados



Atributos	VS3	Modelo deste Estudo
Estruturas para Avaliação e Treinamento	HERA	Não consta

Tabela 17 – Comparação entre o VS3 e o modelo proposto neste estudo

A lista a seguir apresenta uma comparação entre os construtores do *ACTIVITY-DL* e a parte deste modelo conceitual focada em representar as relações entre os objetivos. Observar Tabela 1

1.
  - *ACTIVITY-DL*: A *IND* B.
  - *Modelo*:  $nextGoal(A, B), nextGoal(B, A), nextGoal(A, B) \rightarrow F, nextGoal(B, A) \rightarrow F, nextGoal(X, A) \wedge nextGoal(X, B) \wedge nextGoal(A, Y) \wedge nextGoal(B, Y), nextGoal(X, A) \wedge nextGoal(X, B) \wedge nextGoal(A, Y) \wedge nextGoal(B, Y) \rightarrow F$  e outras situações que não podem ser expressadas nessa representação.
2.
  - *ACTIVITY-DL*: A *SEQ* B.
  - $nextGoal(A, B), nextGoal(B, A)$  e outras situações que não podem ser expressadas nessa representação.
3.
  - *ACTIVITY-DL*: A *SEQ – ORDER* B.
  - $nextGoal(A, B)$  e outras situações que não podem ser expressadas nessa representação.
4.
  - *ACTIVITY-DL*: A *PAR – SIM* B, A *PAR – SIM* B, A *PAR – START* B, A *PAR – END* B
  - $nextGoal(X, A) \wedge nextGoal(X, B) \wedge nextGoal(A, Y) \wedge nextGoal(B, Y)$  e outras situações que não podem ser expressadas nessa representação.

#### 6.2.1 NORMMAS

A Tabela a seguir exibem uma comparação entre o *NORMMAS* e o modelo proposto neste estudo.

Atributos	NORMMAS	Modelo deste Estudo
Finalidade	Representar agentes considerando questões atreladas a normas, sanções e violações	Representar cenários de acidentes
Generalização	Trata de cenários onde se tem o interesse de representar agentes, normas, violações e sanções	Trata de cenários específicos para acidentes onde se deseja conhecer as causas do acidente
Agentes	Não define os estados mentais do agente	Não define os estados mentais do agente
Representação de Objetivos	Não possui estrutura para representar objetivos	Possui uma estrutura simples para representar objetivos. Essa estrutura considera apenas questões de pré-requisitos
Normas	Presença de tuplas que permitem expressar proibições e sanções para os mais diversos casos possíveis	Presença de regras que permitem expressar violações e sanções para cenários de acidentes
Generalização	Trata de cenários onde se tem o interesse de representar agentes, normas, violações e sanções	Trata de cenários específicos para acidentes onde se deseja verificar as causas do acidente
Estrutura da Linguagem	Lógica de Primeira Ordem (definições dos conceitos em termos de tuplas)	Teoria de Conjuntos e Lógica de Predicados

### 6.2.2 Comparação Genérica entre os Modelos

Tendo como base as análises feitas nas seções anteriores, é possível elaborar na Tabela 18 que apresenta uma análise comparativa dos arcabouços no que tange a expressividade do modelo computacional proposto nesse texto. Por expressividade, se entende capacidade de expressar, representar o objeto de interesse. Para essa análise foi feita a seguinte escala: nenhuma expressividade  $\prec$  pouco expressivo  $\prec$  expressivo  $\prec$  muito expressivo  $\prec$  altamente expressivo. O termo nenhuma expressividade não indica que é impossível definir a estrutura em observação dentro do modelo em voga, mas sim que o engenheiro de modelagem terá que criar uma estrutura conceitual *ad hoc*. Sobre o mesmo aspecto reside pouco expressivo, contudo o modelo - neste caso - possui algumas estruturas pré-definidas que diminuem o esforço da especificação. O termo expressivo deixa claro que o modelo permite especificar o objeto de interesse sem que o engenheiro tenha de criar muitos atributos para o domínio de interesse. O termo muito expressivo define que o modelo apresenta diversos conceitos específicos para representar o objeto em interesse, contudo ainda há margem para que o modelador tenha que criar um ou mais atributos. O termo altamente expressivo define o caso onde o modelo especifica o objetivo de interesse muito bem fazendo com que o modelador não precise definir nenhum critério conceitual a mais

(ou terá que montar poucas definições).

Crítérios	MOISE+	DASTANI	V3S	NORMMAS
<b>Agente</b>	pouco	pouco	muito	pouco
<b>SMA</b>	altamente	pouco	expressivo	pouco
<b>Artefato</b>	nenhuma	pouco	expressivo	pouco
<b>Norma</b>	nenhuma	altamente	pouco	altamente
<b>Violação</b>	nenhuma	altamente	pouco	altamente
<b>Sanção</b>	nenhuma	altamente	pouco	altamente
<b>Risco</b>	nenhuma	pouco	altamente	pouco
<b>P.O.A.E</b>	nenhuma	pouco	pouco	pouco
<b>Objetivos</b>	muito	pouco	muito	pouco
<b>C.A</b>	nenhuma	pouco	pouco	pouco
<b>I.AG.AR</b>	nenhuma	pouco	pouco	pouco
<b>D.C.A</b>	nenhuma	pouco	altamente	pouco

Tabela 18 – Análise comparativa sobre a expressividade desses modelos no que tange aos objetivos deste estudo.

O critério **Agente** condiz com representação dos estados internos que um agente pode ter. O critério **SMA** condiz com presença de elementos que são necessários para especificar um *Sistema Multiagente*. O critério **Artefato** condiz com elementos que correspondem a definição presente na Seção 2.2. O critérios de **Norma** corresponde a Regras que devem ser acatadas pelos agentes. **Violação** define o que corresponde o não cumprimento de uma dada regra. **Sanção** implica penalidade que está sobre o agente. **Risco** consiste no evento ruim que tem um potencial de ocorrer sobre o agente. **P.O.A.E** significa Possibilidade de Ocorrer algo Errado e corresponde a expressar condições onde existe potencial de acontecer algo inapropriado sobre o agente mesmo que esse realize sua função com excelência. **Objetivos** implica alvos que devem ser atingidos pelos agentes. **C.A** consiste em Condições Ambientais que interagem com a atividade executada pelos agentes. **I.AG.AR** representa as Interações entre Agentes e Artefatos. **D.C.A** - Descrição de Cenários de Acidentes, consiste na capacidade de desenvolver raciocínios a fim de representar cenários de acidentes.

As subseções anteriores em conjunto com a Tabela 18 permitem concluir que esse trabalho é inovador no que tange a ter um vocabulário específico para representar cenários de risco e de acidentes (tanto sobre o responsável pelo acidente bem como a vítima) de atividades manuais usando para isso, o conceito de sistema multiagente normativo. Esse vocabulário apresenta limitações as quais serão debatidas na próxima Seção.

### 6.3 Discussão do Modelo Conceitual

A discussão dos resultados que estão expostos nas sub-seções 4.2.1, 4.2.2, 4.2.4 foi feita na própria apresentação dos mesmos. Isso se deve a natureza desses resultados, não é possível realizar a exposição deles sem discutir os fundamentos conceituais que justifiquem a existência dos mesmos. Essa situação não se aplica no texto presente em 5 e em 5.2 onde os resultados estão apenas expostos mas não foram discutidos. O mesmo acontece com a Metodologia usada para chegar nesses resultados, foi exposta mas não foi discutida. O texto a seguir fará uma

discussão desses elementos apresentando as principais dificuldades que foram encontradas na realização desse estudo.

### 6.3.1 Considerações sobre Critérios Metodológicos ao estudo de caso

A primeira fase consistiu em descrever a manutenção em termos de objetivos que as vezes são organizados em série e as vezes em paralelo. Não houve grandes dificuldades para fazer isso, pois essa atividade é claramente composta de subatividades. O que foi um ponto relativamente complicado de se verificar nesse estudo, é que os profissionais não precisam executar os objetivos na estrutura em que o modelo foi apresentado. Inclusive, muitas vezes os técnicos planejam a manutenção de um jeito e ao chegar no ambiente de execução eles mudam o encadeamento dos objetivos. Há um número finito e relativamente pequeno (é difícil definir um número, mas as observações permitem concluir que este número está na ordem de 10 formas diferentes) sobre como esses objetivos podem ser organizados e isso ameniza a falta de previsibilidade de como a manutenção será realizada.

O problema da organização dos objetivos pode ser resolvida de três maneiras diferentes. Em uma delas o engenheiro de manutenção modela o problema para todos os cenários possíveis. Portanto, se houver 10 formas diferentes de organizar esses objetivos, o engenheiro deverá refletir acerca dessas 10 formas. Outra forma consiste em definir todas as relações possíveis que o predicado  $nextGoal(g_i, g_j)$  permite o uso de uma única estrutura. Nessa implementação do modelo, o agente por intermédio dos seus estados internos, escolhe a qual objetivo ele deverá tentar alcançar. Essa questão não foi levada em consideração no estudo de caso em análise porque o autor estava interessado em realizar a análise necessária sobre a possibilidade de usar este modelo em um problema real. É possível argumentar que aquele arranjo de objetivos não é o único possível, contudo não deixa de ser um arranjo real e que pode servir de referência aos profissionais. A terceira solução a ser considerada consiste no estudo e uso de algoritmos de *Partial Order Planning* aplicados a essa problemática (BARRETT; WELD, 1994).

Com a realização estudo não é possível sabe afirmar se o arranjo dos objetivos interfere no predicado  $affectsRels(r_k, r_n)$ . Para que isso seja analisado se faz necessário aplicar esse modelo para diversas situações diferentes onde todas devem apresentar a problemática do arranjo de objetivos. Se em uma dessas situações a especificação do predicado  $affectsRels(r_k, r_n)$  mudar, então a proposição - o arranjo de objetivos que afeta o predicado  $affectsRels(r_k, r_n)$  - deve ser tida como verdadeira, contudo se não mudar, não é possível afirmar que essa proposição é falsa.

A utilização do conceito de papel e da relação entre o agente e o seu papel foram muito adequados para as análises desse modelo. Isso se deu por observar como ocorre a distribuição dos objetivos aos agentes, pois na manutenção em linha viva todos os profissionais são tidos como executores, contudo existe uma distribuição de tarefas tendo em vista o conhecimento e a experiência de cada profissional ali envolvido. Portanto, para enquadrar essa questão nos

moldes do modelo em análise se fez necessário encontrar um padrão de como os objetivos são distribuídos em função das atividades dos agentes. Com base nisso o autor concluiu o que está exposto na Tabela 5.

Os conceitos de artefato e de relação foram adequados para esse estudo de caso não tendo a necessidade de definir nenhuma outra abordagem para tal. Todo o rol de ferramentas e de equipamentos foram definidos como artefatos. A fim de tornar a modelagem mais expressiva, criou-se subconjuntos de artefatos definindo um apenas para ferramentas e outro apenas para equipamentos. Isso não foi feito para não induzir os leitores desse estudo ao erro por entender que essa divisão pertence a estrutura conceitual do modelo propriamente dito. Uma taxonomia dessas seria adequada apenas para esse caso em estudo, caso contrário diminuiria o poder de generalização do modelo.

Os conceitos de condição, risco e consequência bem como o predicado  $hasRisk(c_k, risk_k, cs_m)$  foram muito apropriados tanto no contexto do modelo em si como também em aplicação ao caso de estudo. Isso se deve ao fato de que uma condição não é um agente e não é um artefato mas é algo que está presente no meio da atividade e interfere com grande intensidade no andamento dos processos, portanto desconsiderar esse conceito ou compactá-lo como parte de outras estruturas implicaria uma representação míope da realidade. Nesse estudo entendeu-se que essas condições são o suficiente para poder realizar a representação desse modelo. As relações entre condição, risco e consequência foram apropriadas para representar diversos cenários dentro do estudo de caso.

Uma consideração que deve ser feita sobre os conceitos de risco e consequência (cujas instâncias para o estudo de caso estão apresentadas nas Tabelas 8, 9) em relação ao estudo de caso é a de que foi considerado apenas um único risco, que é o de ser eletrocutado e uma única consequência, que é a morte. Contudo, há considerações que devem ser feitas no que tange a realidade, pois essa atividade exibe outros riscos tais como: queda, animais peçonhentos, queimadura, entre outros, que assim como eletrocutado, podem apresentar outras consequências além da morte. Esses riscos a mais não foram considerados no caso em estudo porque os raciocínios exibidos na Subseção 5.2 puderam ser feitos sem a necessidade deles. Outro ponto que corroborou com isso consiste no fato de que neste estudo o interesse era obter primeiramente uma versão mais simples do modelo para então, se necessário, torná-lo mais complexo. Isso implica em realizar algumas escolhas pragmáticas e uma delas consiste na verificação de qual risco é o mais importante e o mais temeroso na atividade. A análise com os profissionais mostram que o risco de ser eletrocutado é o principal e é mais preocupante ao executar uma atividade de manutenção em linha viva.

Outro ponto reside na verificação das consequências desse risco o que remete a uma pergunta: um profissional de linha viva ao executar manutenção em uma subestação de energia pode se envolver em um acidente em que ele é eletrocutado, e ainda sobreviver? A resposta a essa pergunta é sim, porém muito improvável. Descargas de equipamentos que operam a 69 kV

- 1500 kVA (o que é relativamente baixo) costumam matar o profissional eletrocutado mesmo que os disjuntores atuem na ordem de milissegundos. Portanto existem, outras consequências além da morte tal como: queimaduras e perda de membros, contudo na grande maioria dos casos o profissional recairá no óbito.

O predicado  $possEntityRel(r_l, e_i, e_k)$ , em que as instâncias para o estudo de caso são apresentada na Tabela 9, define como se dá a relação entre duas entidades. Essa estrutura se tornou muito útil para fazer diversos raciocínios interessantes que estão presentes nas regras. Portanto conclui-se que ela foi adequada, necessária e importante para essa representação e para esse estudo de caso, contudo, ela tornou a especificação da modelagem um processo muito custoso, porque o modelador teve de refletir em todas as relações possíveis que são executadas na atividade e depois disso, teve que ver quais relações se enquadravam em cada objetivo. Esse custo também está presente nos raciocínios que devem ser feitos pois dependendo da situação há uma série de relações que devem ser avaliadas. Essa questão nos permite refletir sobre a viabilidade de um modelo assim, para situações onde o número de artefatos bem como o número de relações entre esses artefatos tendem ao infinito. Contudo, o fato do modelador ter que refletir sobre todas as relações bem como seus respectivos riscos, permite a realização de uma análise muito mais profunda da atividade e de como a segurança dos profissionais pode ser afetada de situação para situação.

O predicado  $affectsRels(r_k, r_n)$  (cujas instâncias dos conceitos, bem como a relação estão presentes nas Tabela 10) agregou dificuldades tanto na concepção como na aplicação ao estudo de caso. Houveram muitas tentativas de resolver essa questão sem ter que abstrair tanto quanto esse predicado faz. Contudo, realizar um mapeamento minucioso de como se dão as atividades, resulta em uma carga de trabalho relativamente custosa e que pode apresentar diversas fragilidades no que tange à consistência lógica (ou seja, um sistema que se contradiz). Portanto, em uma primeira abordagem, admitir que a não execução (ou a má execução) de uma relação afeta negativamente outra relação implica uma visão pragmática e simples para resolver o problema onde um electricista se envolve em um acidente sobre o qual ele não tem responsabilidade alguma. Assim verificou-se que esse é um ponto do modelo a ser melhorado a fim de se obter representações consistentes, expressivas e com relativo baixo custo de modelagem.

Os predicados  $requiresCirc(circ_n, g_m)$ ,  $requiresEntity(goal_i, e_j)$ ,  $instanceOfRel(circ_n)$  e  $instanceOfCond(circ_n)$  (em que as instâncias para esses predicados estão presentes nas Tabelas 13 e 14) apresentam a especificação dos relacionamentos entre entidades, relações e condições com os objetivos. O lado positivo dessa abordagem é a possibilidade de ter uma descrição lógico-formal da manutenção altamente detalhada. Isso permite a equipe responsável pela manutenção avaliar o problema com mais profundidade e, portanto, tomar decisões mais eficientes. O problema dessa abordagem reside na mesma situação atrelada à Tabela 9 que consiste em um processo altamente custoso em termos de tempo e de trabalho a fim de especificar as relações, entidades e condições com os objetivos.

### 6.3.2 Considerações sobre Critérios Metodológicos ao Raciocínios

Os raciocínios feitos sobre o modelo são de crucial importância para definir a eficácia desse projeto pois é com base nisso que se torna possível avaliar o quão efetivo vem a ser essa representação. Os Raciocínios 1 e 5, dados respectivamente pelas Subseções 5.2.1 e 5.2.5 apresentam o problema com bastante expressividade. Tendo em vista o fato de que a Glicerina é um composto químico relevante para manter o isolamento da parte não condutiva do bastão universal, esquecer de passar esse composto na ferramenta gera um potencial acidente de ser eletrocutado em todas as outras situações onde o bastão será usado (a não ser nas situações onde o bastão universal não será usado em condutores energizados). Tanto os predicados como as regras que estão atreladas a violação de relação e suas respectivas consequências representaram essa condição com sucesso. Nesse caso não aconteceu nenhuma sanção sobre o agente 4, portanto nem toda violação de relação gera necessariamente uma sanção. O predicado *possOfNegConseqFor* conseguiu trazer com êxito a sensação de possibilidade que existe em fenômeno desse gênero.

Quando concebeu-se esse modelo, considerou-se a possibilidade de trabalhar problemas dos Raciocínios 1 e 5 por meio do conceito de *Probabilidade*. Isso é interessante porque um predicado que consegue expressar fenômenos estatísticos válidos com excelência, apresenta possibilidades de aplicações extremamente elevadas. Contudo, trabalhar com probabilidade resulta em diversas complicações de modelagem. Uma dessas complicações consiste na coleta de dados de uma amostragem significativa de uma população de acidentes e na análise estatística adequada para descrever a probabilidade de um acidente acontecer. Entretanto, isso não é o suficiente pois essa probabilidade é condicionada a ocorrência de uma determinada relação.

O Raciocínio 1, por exemplo, demonstra que a não execução de *relPanoGlicerina* resulta na possibilidade de ocorrer um acidente em *relBastaoGarraGondutor*. Se estivesse sendo trabalhado com o conceito de probabilidade, então seria necessário desenvolver técnicas que verificam a probabilidade de acontecer algo ruim na relação *relBastaoGarraGondutor* para o caso da relação *relPanoGlicerina* não ser efetivada com sucesso. Contudo, se a não execução de uma outra relação também afetar *relBastaoGarraGondutor*, então também se faz necessário encontrar essa outra probabilidade. Além de aumentar a complexidade desse modelo, abre diversas indagações no que tange a como fazer isso, o que pode ser um potencial campo de investigação científica. Com a finalidade de viabilizar uma primeira versão do modelo, optou-se por usar o conceito de possibilidade em vez de probabilidade. Apesar de diminuir a expressividade do modelo no que tange a questão de que existe um componente sobre aleatoriedade, isso simplifica o processo de especificação, facilita o desenvolvimento de raciocínios e evita que o modelo seja estruturado sob proposições falsas (por exemplo, definir uma probabilidade para uma condição de mundo que não é precisamente valorada).

O vocabulário definido neste modelo foi apropriado para representar a condição de mundo presente no Raciocínio 2 que está exposto na Subseção 5.2.2. Em uma situação onde



não há um pano para poder limpar todas as ferramentas, a manutenção é interrompida e essa situação ficou claramente representada por esse raciocínio onde a geração da violação de entidades corresponde a finalização da manutenção. Há a possibilidade de existir um cenário onde os profissionais criam algum tipo de técnica alternativa para poder transpassar a falta de algum artefato, inclusive se esse não apresentar grande complexidade estrutural como é o caso de um pano. Contudo, decidiu-se por não incorporar esse tipo de situação no modelo por conta de complexidades que isso pode trazer a estrutura da representação. Manter o modelo assim permite representar os cenários mais prováveis, tendo vista que a ausência de diversos tipos de artefatos muitas vezes não permite a continuidade da atividade.

A execução de uma manutenção em linha viva deve seguir a risca as condições ambientais adequadas para essa finalidade. Uma dessas condições é a umidade relativa do ar, que deve estar necessariamente inferior a setenta por cento. O Raciocínio 3 na Subseção 5.2.3 demonstra esse tipo de situação onde um agente tenta executar uma atividade com a umidade relativa do ar em níveis inapropriados para isso, ocasionando o surgimento de uma violação de condição gerando uma sanção no agente que corresponde a ser eletrocutado e, conseqüentemente, morto. É interessante observar que nem toda violação de condição, no mundo real, resulta necessariamente em uma sanção ao violador. A umidade relativa do ar recai nessa situação, pois pode ser que o profissional cometa essa violação sem se envolver em um acidente. Isso pode ser resolvido por construir regras tratando condições em relação ao predicado *possOfNegConseqFor*. Contudo, a desobediência de condições ambientes normalmente resultam em acidentes. Portanto, essa condição - apesar de não tratar todos os cenários possíveis, trata um bom número dos mesmos.

A chave catraca é usada pelo profissional de linha viva para remover um parafuso que está preso ao conector. Uma execução inapropriada dessa relação resulta na ocorrência do electricista ser eletrocutado e morto. Há diversas formas de como isso pode acontecer, sendo que uma delas consiste no profissional se posicionar de forma inapropriada para realizar essa relação e, por consequência, esbarrar tanto com o corpo quanto com a ferramenta em algum condutor de forma inapropriada. Portanto é de crucial importância que o profissional realize a execução com excelência. Esse comportamento é descrito pelo raciocínio dado na Subseção 5.2.4. Assim como na situação relacionada condição, a realidade dos fatos pode produzir cenários possíveis “nesse caso” que não são adequadamente representados por esse modelo. Um possível cenário para essa situação consiste no fato do profissional simplesmente não conseguir executar a relação, sem que isso resulte em algum acidente. Contudo, a situação descrita pelo modelo apresenta o pior cenário possível.

Em um acidente, pessoas que não são responsáveis por atos cometidos podem sofrer duras consequências desses atos. Essa situação está demonstrada no Raciocínio 5 presente na Subseção 5.2.5. Nessa situação, não passar glicerina no pano pode gerar um acidente ao montar a relação *relPanoGlicerina*, porque o bastão isolante a ser usado nesse processo, não estará em



condições operacionais seguras, uma vez que a superfície dessa ferramenta pode conter algum tipo de impureza que corrobore com aumento de corrente de fuga em níveis suficientemente altos para matar alguém.

O Raciocínio 1 apresentou com excelência essa influência que a falta do uso de glicerina tem sobre a possibilidade de ocorrer algo errado no momento em que um certo profissional remover o parafuso usando o bastão. O autor entende, portanto que todos os predicados usados para representar essa situação foram necessários sendo que a ausência de um ou de outro, descaracteriza completamente essa representação. Nessa condição, se faz necessário saber com qual objetivo *relParafusoConector* está atrelado, e isso é feito por intermédio do  $\in$  e de *requiresCirc(goal<sub>i</sub>, circ<sub>j</sub>)*. Além disso, não é possível efetuar nenhum tipo de raciocínio sobre essa condição sem levar em consideração se os agentes tentam alcançar esse objetivo, e isso é feito por meio do predicado *starts(agent<sub>n</sub>, g<sub>m</sub>)*. O fato de ocorrer um acidente, ser independente do agente que está executando o objetivo, é muito bem representado pelos predicados *possOfNegConseqFor* e *happensNegConseqFor*. Pela regra 4.25, essa reunião de fatores em conjunto com os riscos associados ao evento dão como verdade para o predicado *possOfNegConseqForEven* gerando a morte do profissional e a interrupção da manutenção.

A presença do Raciocínio 6 dado na Subseção 5.2.6 demonstrou que o modelo é capaz de interpretar quando o objetivo g23 for atingido. Em conjunto com o predicado *enabledToStart(ag<sub>i</sub>, g<sub>j</sub>)*, com a programação dos estados internos do agente e com a Regra 4.20 é possível verificar uma relação de continuidade para a representação (desde que o modelador defina os critérios que levam o agente a tentar atingir um dado objetivo).

O Raciocínio 7 dado por 5.2.7 apresenta como o modelo se comporta quando é feito uma consulta que não corresponde a realidade, ou seja quando *agente1* tenta alcançar o objetivo g23 sendo que este não faz parte dos objetivos que devem ser alcançados por aquele. O resultado obtido foi o que se esperava.

Conclui-se que o estudo de caso foi representado de forma apropriada pelos predicados e regras presentes nesse texto. Contudo, muitos raciocínios não foram capazes de apresentar todos os cenários possíveis a uma dada circunstância. Porém, todos os cenários resultantes do modelo correspondem a circunstâncias reais da manutenção. Não apenas isso, mas são tanto as circunstâncias mais prováveis como as mais sérias. Assim sendo, escapa ao poder desse modelo representar muitos cenários, contudo os cenários mais importantes foram muito bem computados por essa representação.

## 7 Conclusão

Esse capítulo tem a finalidade de realizar uma consideração sobre todo estudo, verificar quais são as conclusões que podem ser feitas e discernir sobre perspectivas futuras para essa pesquisa.

### 7.1 Avaliação do Objetivo

Essa Seção têm o propósito de gerar uma reflexão sobre se o estudo presente nesta dissertação atingiu o objetivo geral bem como os objetivos específicos a que se propõem. A Subseção 7.1.1 apresenta essa reflexão para o objetivo geral e a subseção 7.1.2 apresenta as reflexões a respeito dos objetivos específicos.

#### 7.1.1 Reflexão sobre Objetivo Geral

Essa seção apresenta uma análise detalhada do correspondente do objetivo geral proposto ao texto do estudo. A seguinte parte do objetivo geral: “*Sintetizar, construir e avaliar, por intermédio de observações, de análises de documentos técnicos, de análises de modelos computacionais e de entrevista com profissionais da área*”, foi abordada no estudo dos documentos técnicos, conversas com profissionais da área, entrevistas com engenheiro de manutenção em linha viva, acompanhamento de um procedimento de manutenção de linha viva, realização da descrição de cenários, participação de *workshops* na área e realização de uma revisão exploratória de textos sobre arcabouços que podem ser aplicados ao contexto desse estudo.

A seguinte parte do objetivo geral: “*um modelo conceitual que define os conceitos e as relações para representar os cenários de ambientes de atividades, bem como os respectivos acidentes que podem acontecer*”, foi trabalhada na construção de um modelo conceitual capaz de abordar cenários de manutenção.

A seguinte parte do objetivo geral: “*em que a validação ocorre por verificar se os raciocínios (para um dado estudo de caso do setor de energia elétrica)*”, foi tratada aplicação do modelo obtido em um estudo de caso cujo qual consiste em procedimentos de manutenção em linha viva. Esse estudo de caso apresenta tanto a modelagem como o desenvolvimento de raciocínios de determinados cenários que são previstos pelo modelo. Por intermédio de observações anteriores, foi possível avaliar se a depuração desses raciocínios está de acordo com a realidade.

A seguinte parte do objetivo geral: “*a fim de levantar um entendimento formal do problema para a comunidade acadêmica no que tange a que tipo de representação computacional é mais apropriada para determinado contexto*” foi abordada na avaliação do modelo conceitual em relação a arcabouços computacionais consolidados pela comunidade acadêmica e verificação do estado do problema no que tange a determinada conjuntura.

### 7.1.2 Reflexão sobre Objetivos Específicos

O objetivo específico “*Identificar os pontos essenciais que devem fazer parte da estrutura do modelo em relação aos riscos e consequências (acidentes) para os atores e atividades (continuidade), que sejam relevantes na prática da atividade de manutenção, em caso de falha na operação*” foi tratado nos estudos de manuais técnicos, entrevista com um engenheiro da área, participação em *workshops*, levantamento de manuais técnicos e acompanhamento da execução de um certo procedimento de manutenção em linha viva.

O objetivo específico “*Construir um modelo conceitual que seja implementável computacionalmente e que produza as inferências que respondam às questões definidas como essenciais*” foi atingido no momento da obtenção de um modelo conceitual com a capacidade de produzir raciocínios dentro do formalismo Lógica de Predicados.

O objetivo específico “*Validar o modelo por aplicá-lo a um dado estudo de caso a fim de averiguar se os raciocínios produzidos nessa situação estão de acordo com a realidade*” foi verificado na aplicação do modelo a um dado estudo de caso como produção de raciocínios sobre certos cenários a fim de confronta-los contra a realidade para sintetizar uma avaliação se ambos são correspondentes e em quais proporções isso ocorre.

O objetivo específico “*Analisar modelos computacionais em relação ao modelo conceitual desse estudo a fim de ter um levantamento formal do estado do problema*” foi atingido como na comparação do modelo conceitual com os arcabouços computacionais propostos pela comunidade acadêmica. Nessa comparação foi possível observar em quais aspectos o modelo se assemelha e se distancia de cada arcabouço permitindo mapear em termos formais o estado desse problema.

Assim sendo, é possível concluir que esse estudo concebeu um modelo conceitual com a capacidade de representar e definir raciocínios sobre acidentes e que, por conta disso, possibilita esclarecer o contexto do problema a ser analisado a luz da computação.

## 7.2 Conclusão Geral

Em síntese é possível concluir que o modelo conceitual concebido nesse estudo contém os conceitos e relações relevantes para representar ambientes de acidentes averiguando causas e consequências. No que tange ao estudo de caso e a validação é possível concluir que o modelo foi capaz de abranger uma dada quantidade de cenários (sendo esses os mais relevantes para os contextos em análise) possíveis dentro de uma certa situação. Sobre averiguação de um entendimento formal do problema é possível concluir que uma análise computacional (seja por meio de modelos computacionais, ou seja por meio de algoritmos), que tenha como por meta trabalhar com acidentes de trabalho, deve considerar pelo menos alguns dos seguintes conceitos: Agente, SMA, Artefato, Norma, Violação, Sanção, Riscos, Objetivos, Condições Ambientais, Interações entre Entidades (Agentes, Artefatos) e Eventos Probabilísticos. Sobre os tipos de

representação computacional que se enquadram no domínio em estudo, conclui-se o que se encontra na lista que se segue:

- *MOISE+* é mais apropriado para representar os seguintes conceitos: *Agente, SMA, objetivos*.
- *Dastani* é mais apropriado para representar os seguintes conceitos *Normas, Violações, Sanções*.
- *V3S* é mais apropriado para representar *SMA, Artefato, Riscos e contém estruturas otimizadas para descrever dinamicamente os cenários de acidentes*.
- *NORMMAS* é mais apropriado para representar *Normas, Violações, Sanções*.

No que tange ao contraste do modelo conceitual proposto nesse estudo em relação aos arcabouços verificados nesse texto, conclui-se que aquele unifica em uma única estrutura concepções que são tratadas de formas isoladas nestes, no que diz respeito a tratativa de acidentes em ambientes de trabalho. Então, a estrutura proposta nesse texto apresenta mecanismos para tratar de Agente, SMA, Artefato, Norma, Violação, Sanção, Risco, Possibilidades de ocorrer algum evento ruim, Objetivos, Integração entre Agentes e Artefatos e Descrição dos Cenários de Acidentes. No *MOISE+*, por exemplo, os mecanismos deste arcabouço tratam apenas de Agente, SMA e Objetivos.

### 7.3 Trabalhos Futuros

Esse estudo abre margem para muitos trabalhos futuros. Alguns desses residem no fato de que este texto apresenta análises de certos arcabouços na representação do modelo conceitual aqui posto. Para cada arcabouço é possível derivar um estudo futuro a fim de usar o modelo conceitual aqui posto para conceber a formulação de requisitos e especificações com o propósito de estudar aspectos inerentes ao arcabouço em si (sendo o levantamento de requisitos e especificações deste modelo conceitual um fim em si mesmo de estudo acadêmico). Por exemplo, supondo que em dada circunstância há o interesse de usar o *MOISE+* para especificar um determinado problema, como o modelo conceitual aqui posto pode ser usado para levantar os requisitos no *MOISE+*? Quais são os pontos positivos e negativos dessa situação? Qual cenário é o mais apropriado para essa situação? Qual cenário é o mais desapropriado? Existem adaptações que devem ser feitas no modelo a fim de melhorar a formalização de requisitos? Se sim, quais são, como e por quais motivos isso deve ocorrer?

O texto presente na Subseção 6.3.2 discute, para cada um dos cinco primeiros raciocínios, elementos que foram representados assim como elementos que não foram representados adequadamente. Verificar como cada um desses problemas podem ser resolvidos também resultam em estudos futuros. A listagem a seguir sintetiza esses problemas em sentenças objetivas:

1. Em vez de trabalhar conceitos de possibilidade, sintetizar um modelo estatístico probabilístico na estrutura conceitual proposta neste estudo.
2. Investigar novas estruturas conceituais para tratar cenários onde os agentes buscam técnicas alternativas para resolver um determinado problema.
3. Investigar novas estruturas conceituais onde uma violação pode ou não gerar uma sanção para as condições ambientes (ou seja, em vez de tratar a possibilidade sobre um relacionamento futuro, tratar a possibilidade sobre um relacionamento presente).
4. Investigar novas estruturas conceituais que considerem a violação de relacionamento em termos de possibilidades e não de um efeito direto no que tange a uma dada causa.

Uma outra abordagem futura de estudo consiste em produzir um modelo conceitual mais enxuto com estruturas conceituais e de relacionamento mais genéricas. Por exemplo, criar formas genéricas de violações.

Além desses, desenvolver uma ontologia para definir o conceito de Objetivo (em relação a estrutura lógico-descritiva do modelo proposto neste estudo) é um outro trabalho futuro

## Referências

- ABBAS, H.; SHAHEEN, S.; AMIN, M. H. Organization of multi-agent systems: An overview. *International Journal of Intelligent Information Systems*, 06 2015. Citado 3 vezes nas páginas 16, 17 e 18.
- ALLEN, J. F. Maintaining knowledge about temporal intervals. *Commun. ACM*, ACM, New York, NY, USA, v. 26, n. 11, p. 832–843, nov. 1983. ISSN 0001-0782. Disponível em: <<http://doi.acm.org/10.1145/182.358434>>. Citado 2 vezes nas páginas 27 e 28.
- ASHBY, W. R. Principles of the self-organizing system. In: FOERSTER, H. v.; ZOPF, G. W. (Ed.). *Principles of Self-Organization: Transactions of the University of Illinois Symposium*. London: Pergamon, 1962. p. 255–278. Citado na página 17.
- BAROT, C. et al. V3S: A virtual environment for risk-management training based on human-activity models. *Presence*, v. 22, n. 1, p. 1–19, 2013. Disponível em: <[http://www.mitpressjournals.org/doi/abs/10.1162/PRES\\_a\\_00134](http://www.mitpressjournals.org/doi/abs/10.1162/PRES_a_00134)>. Citado 8 vezes nas páginas 7, 8, 27, 28, 29, 30, 38 e 98.
- BARRETT, A.; WELD, D. S. Partial-order planning: Evaluating possible efficiency gains. *Artificial Intelligence*, v. 67, n. 1, p. 71 – 112, 1994. ISSN 0004-3702. Disponível em: <<http://www.sciencedirect.com/science/article/pii/0004370294900124>>. Citado na página 107.
- BONASSO, R. P. et al. Experiences with an architecture for intelligent, reactive agents. In: . [S.l.: s.n.], 1995. v. 9, p. 187–202. Citado na página 14.
- CAMUS, F.; LENNE, D.; PLOT, E. Designing virtual environments for risk prevention: the melissa approach. *International Journal on Interactive Design and Manufacturing (IJIDeM)*, v. 6, n. 1, p. 55–63, Feb 2012. ISSN 1955-2505. Disponível em: <<https://doi.org/10.1007/s12008-011-0138-4>>. Citado 2 vezes nas páginas 7 e 30.
- CARRON, T.; BOISSIER, O. Towards a temporal organizational structure language for dynamic multi-agent systems. 01 2001. Citado 3 vezes nas páginas 18, 19 e 20.
- CASTELFRANCHI, C. Commitments: From individual intentions to groups and organizations. In: *ICMAS*. [S.l.: s.n.], 1995. Citado 2 vezes nas páginas 19 e 21.
- CASTELFRANCHI, C. et al. Social trust: A cognitive approach. 12 2018. Citado 2 vezes nas páginas 14 e 15.
- CHANG, S.; MENEGUZZI, F. Simulating normative behaviour in multi-agent environments using monitoring artefacts. In: DIGNUM, V. et al. (Ed.). *Coordination, Organizations, Institutions, and Norms in Agent Systems XI*. Cham: Springer International Publishing, 2016. p. 59–77. ISBN 978-3-319-42691-4. Citado 3 vezes nas páginas 31, 38 e 98.
- CHEN, P. P. shan. The entity-relationship model: Toward a unified view of data. *ACM Transactions on Database Systems*, v. 1, p. 9–36, 1976. Citado 2 vezes nas páginas 39 e 41.

DASTANI, M. et al. Normative multi-agent programs and their logics. In: MEYER, J.-J. C.; BROERSEN, J. (Ed.). *Knowledge Representation for Agents and Multi-Agent Systems*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2009. p. 16–31. ISBN 978-3-642-05301-6. Citado 7 vezes nas páginas 7, 21, 22, 23, 38, 50 e 98.

EDWARD, L. et al. Modelling autonomous virtual agent behaviours in a virtual environment for risk. *IJVR*, v. 7, p. 13–22, 01 2008. Citado na página 14.

ESTEVA, M.; PADGET, J.; SIERRA, C. Formalizing a languages for institutions and norms. In: MEYER, J.; TAMBE, M. (Ed.). *Lecture Notes Artificial Intelligence*. [S.l.]: Springer-Verlag, 2002. v. 2333, p. 348–366. Citado na página 21.

EXPLORING Organizational Designs with TAEMS: A Case Study of Distributed Data Processing. 09 1996. Citado na página 20.

FADIER, E.; GARZA, C. D. L.; DIDELOT, A. Safe design and human activity: construction of a theoretical framework from an analysis of a printing sector. *Safety Science*, v. 41, n. 9, p. 759 – 789, 2003. ISSN 0925-7535. Disponível em: <<http://www.sciencedirect.com/science/article/pii/S092575350200022X>>. Citado 4 vezes nas páginas 23, 24, 25 e 51.

FERBER, J.; GUTKNECHT, O. A meta-model for the analysis and design of organizations in multi-agent systems. In: *Proceedings of the 3rd International Conference on Multi Agent Systems*. Washington, DC, USA: IEEE Computer Society, 1998. (ICMAS '98), p. 128–. ISBN 0-8186-8500-X. Disponível em: <<http://dl.acm.org/citation.cfm?id=551984.852257>>. Citado 2 vezes nas páginas 18 e 19.

FOERSTER, H. von. On self-organizing systems and their environments. *Self-organizing systems*, p. 31–50, 01 2003. Citado na página 17.

FOX, M. S.; BARBUCEANU, M.; GRUNINGER, M. An organisation ontology for enterprise modeling: Preliminary concepts for linking structure and behaviour. *Computers in Industry*, v. 29, n. 1, p. 123 – 134, 1996. ISSN 0166-3615. WET ICE '95. Disponível em: <<http://www.sciencedirect.com/science/article/pii/0166361595000798>>. Citado 2 vezes nas páginas 18 e 19.

GARSON, J. Modal logic. In: ZALTA, E. N. (Ed.). *The Stanford Encyclopedia of Philosophy*. Fall 2018. [S.l.]: Metaphysics Research Lab, Stanford University, 2018. Citado na página 25.

GARVEY, A.; LESSER, V. Design-to-time scheduling and anytime algorithms. *SIGART Bull.*, ACM, New York, NY, USA, v. 7, n. 2, p. 16–19, abr. 1996. ISSN 0163-5719. Disponível em: <<http://doi.acm.org/10.1145/242587.242591>>. Citado na página 20.

GENESERETH, M. R.; NILSSON, N. J. *Logical Foundations of Artificial Intelligence*. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 1987. ISBN 0-934613-31-1. Citado na página 14.

GRUBER, Y. A. et al. Using game engines in lightning shielding: The application of the rolling spheres method on virtual as-built power substations. In: . [S.l.: s.n.], 2018. Citado na página 36.

HÜBNER, J.; SICHTMAN, J.; BOISSIER, O. Moise+: Towards a structural, functional, and deontic model for mas organization. In: . [S.l.: s.n.], 2002. p. 501–502. Citado 2 vezes nas páginas 21 e 38.



HÜBNER, J. F.; SICHMAN, J. S.; BOISSIER, O. A model for the structural, functional, and deontic specification of organizations in multiagent systems. In: BITTENCOURT, G.; RAMALHO, G. L. (Ed.). *Advances in Artificial Intelligence*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2002. p. 118–128. ISBN 978-3-540-36127-5. Citado 7 vezes nas páginas 7, 16, 18, 19, 20, 21 e 98.

JENNINGS, N. R.; LÉSPRANCE, Y. *Intelligent Agents VI. Agent Theories, Architectures, and Languages: 6th International Workshop, ATAL'99, Orlando, Florida, USA, July 15-17, 1999. Proceedings*. [S.l.: s.n.], 2000. ISBN 978-3-540-67200-5. Citado 3 vezes nas páginas 12, 13 e 14.

LENDARIS, G. On the definition of self-organizing systems. *Proceedings of the IEEE*, v. 52, p. 324– 325, 04 1964. Citado na página 17.

LOPEZ, F.; LUCK, M. Modelling norms for autonomous agents. In: . [S.l.: s.n.], 2003. p. 238 – 245. ISBN 0-7695-1915-6. Citado na página 21.

LÓPEZ, F. López y; LUCK, M. A model of normative multi-agent systems and dynamic relationships. In: LINDEMANN, G.; MOLDT, D.; PAOLUCCI, M. (Ed.). *Regulated Agent-Based Social Systems*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2004. p. 259–280. ISBN 978-3-540-25867-4. Citado na página 21.

MOSES, Y.; TENNENHOLTZ, M. Artificial social systems. *Computers and Artificial Intelligence*, v. 14, 12 1995. Citado na página 21.

PENTLAND, B.; RUETER, H. H. Organizational routines as grammars of action. *Administrative Science Quarterly*, v. 39, p. 484, 09 1994. Citado na página 18.

PENTLAND, B. T. Grammatical models of organizational processes. *Organization Science*, v. 6, n. 5, p. 541–556, 1995. Citado na página 18.

PRIGOGINE, I.; NICOLIS, G. Self-organisation in nonequilibrium systems: Towards a dynamics of complexity. In: \_\_\_\_\_. *Bifurcation Analysis: Principles, Applications and Synthesis*. Dordrecht: Springer Netherlands, 1985. p. 3–12. ISBN 978-94-009-6239-2. Citado na página 18.

RAO, A. S.; GEORGEFF, M. P. Modeling rational agents within a bdi-architecture. In: *Proceedings of the Second International Conference on Principles of Knowledge Representation and Reasoning*. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 1991. (KR'91), p. 473–484. ISBN 1-55860-165-1. Disponível em: <<http://dl.acm.org/citation.cfm?id=3087158.3087205>>. Citado na página 14.

RASMUSSEN, J. Risk management in a dynamic society: a modelling problem. *Safety Science*, v. 27, n. 2, p. 183 – 213, 1997. ISSN 0925-7535. Disponível em: <<http://www.sciencedirect.com/science/article/pii/S0925753597000520>>. Citado na página 23.

RICCI, A.; VIROLI, M.; OMICINI, A. Programming mas with artifacts. In: BORDINI, R. H. et al. (Ed.). *Programming Multi-Agent Systems*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2006. p. 206–221. ISBN 978-3-540-32617-5. Citado 2 vezes nas páginas 14 e 15.



RICCI, A.; VIROLI, M.; OMICINI, A. Cartago: A framework for prototyping artifact-based environments in mas. In: WEYNS, D.; PARUNAK, H. V. D.; MICHEL, F. (Ed.). *Environments for Multi-Agent Systems III*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2007. p. 67–86. ISBN 978-3-540-71103-2. Citado 2 vezes nas páginas 15 e 16.

ROSENDO, M. et al. Towards the development of a 3d serious game for training in power network maintenance. In: . [S.l.: s.n.], 2011. p. 16 – 23. Citado na página 36.

RUSSELL, S. J.; NORVIG, P. *Artificial intelligence - a modern approach, 2nd Edition*. Prentice Hall, 2003. (Prentice Hall series in artificial intelligence). ISBN 0130803022. Disponível em: <<http://www.worldcat.org/oclc/314283679>>. Citado 2 vezes nas páginas 12 e 13.

SAMARA, J. M. et al. Um modelo de multiagentes normativos para descrição de personagens de jogos sérios. In: *Knowledge Representation for Agents and Multi-Agent Systems*. [S.l.]: XII Workshop-Escola de Sistemas de Agentes, seus Ambientes e Aplicações, 2018. p. 49–60. Citado na página 36.

SO, Y.-p.; DURFEE, E. An organizational self-design model for organizational change. 03 1996. Citado na página 20.

WOOLDRIDGE, M.; JENNINGS, N. R. Intelligent agents: theory and practice. *The Knowledge Engineering Review*, Cambridge University Press, v. 10, n. 2, p. 115–152, 1995. Citado na página 14.

WRIGHT, G. H. von. On the logic and ontology of norms. In: \_\_\_\_\_. *Philosophical Logic*. Dordrecht: Springer Netherlands, 1969. p. 89–107. ISBN 978-94-010-9614-0. Citado na página 22.

## APÊNDICE A – Perguntas da Entrevista

### A.1 Perguntas Gerais

- Dentro do contexto da manutenção em linha viva, quais são todos os papéis envolvidos?
- Quais são as funções de cada papel? Tentar especificar em uma lista todas as funções relacionadas a cada papel.
- Entre os eletricitas, existe alguma divisão de tipos de eletricitas, por exemplo, eletricista sênior, eletricista pleno ou eletricista júnior?
- Entre os Engenheiros e os Gerentes existe algum tipo de subdivisão?
- Quais são as relações de Link de autoridades (ou seja, quem tem autoridade sobre quem) existente entre os papéis?
- Esses papéis se mantêm para diferentes tipos de manutenção?
- Quais são as relações de compatibilidade entre os papéis, ou seja, se não houver nenhuma pessoa de um papel, qual é a pessoa de outro papel que pode substituir ela?
- Se fosse necessário realizar uma divisão em grupo de diferentes papéis, você acha que os grupos: Grupo dos Engenheiros, Grupo dos Gerentes, Grupo dos Eletricitas são suficientes para descrever todos os elementos da manutenção?
- O número de pessoas necessárias para cada papel varia com relação ao tipo de manutenção? Isso ocorre em todos os papéis?
- Se não varia (ou seja, temos um número mínimo e um número máximo para qualquer tipo de manutenção em linha viva) então qual é o número de pessoas mínimo e o número de pessoas máximo para cada um dos papéis?
- Se se varia, então qual é o número mínimo e o número máximo de pessoas para cada papel em relação a esses tipos de manutenção: substituição do isolador de pedestal (método a distância), substituição de pingo aéreo (mét dist e mét potencial), substituição do conector (métd a distância e métd. ao potencial)?
- Quais são os critérios que influenciam nesses números?

- Levando em consideração esses papéis, quais são as relações mínimas e máximas entre cada papel? Ou seja, quantos eletricitas eu preciso para cada gerente? Quantos eletricitas eu preciso para cada Engenheiro? Quantos gerentes eu preciso para cada Engenheiro? Esses números são fixos ou variam para cada manutenção? Se variam, quais são os critérios que influenciam nessas relações?
- Durante o planejamento da manutenção, quais são as variáveis ambientais que afetam as tomadas de decisão? Como essas variáveis influenciam nessa tomada de decisão dos agentes?
- Durante o desenvolvimento da manutenção, quais são as situações imprevisíveis ou difíceis de prever que influenciam no decorrer da manutenção (ou seja, o que pode mudar o comportamento da manutenção durante a execução) - essas situações mudam muito conforme o tipo de manutenção?
- O que influencia no método da manutenção (distância ou ao potencial)?
- O planejamento da manutenção acontece com quantos dias de antecedência da execução da manutenção em si?
- Como é feita a análise das distâncias seguras? Quais são os critérios usados?
- A simulação da manutenção é feita? O que impacta nessa necessidade?
- Como é feita a previsão do tempo de manutenção?
- Como é feita a análise dos fatores inerentes de intervenção?
- As etapas como Sinalização e Isolamento da Área de Trabalho, Afastamento de Fases, Teste de Corrente de Fuga em Andaime e Escada isolante e Instalação de By-Pass Provisórios podem ser entendidas como uma pré-manutenção ou já estão dentro do escopo da manutenção?
- Como é feita essa inspeção visual? Quais são os critérios que influenciam nessa inspeção?
- O que são as condições necessárias de segurança?
- No que diz respeito a ferramenta há um eletricista responsável de cuidar somente elas? O trabalho dele é focado apenas nisso ou ele revessa de trabalho com outro eletricista?
- Existe um eletricista responsável apenas para fazer a medida do isolamento e das estruturas ou cada eletricista deve tomar cuidado para fazer a medida dos seus equipamentos em específico?
- Confere que o trabalho é iniciado apenas depois que a operação libera o circuito para a intervenção? Quando essa liberação acontece? Antes da manutenção, quando a equipe chega no local?

- O gerente ou o responsável pela equipe fica o tempo inteiro coordenando a manutenção? Dizendo a todo momento a todos o que cada um deve fazer?
- O responsável pela equipe também pode trabalhar como um eletricista? Quando isso acontece, porque isso acontece?
- Quais são os critérios que influenciam no revezamento dos eletricistas?
- O que define, no método ao potencial, se a abordagem será feita através da escada ou através do andaime?
- As distâncias no manual Linha Vida de Subestação Método a Distância e ao Potencial são informações atualizada? Onde consigo referências mais atuais?
- Os procedimentos Gerais para Manutenção valem em qualquer situação?

## A.2 Perguntas sobre Substituição do Isolador de Pedestal

A lista a seguir descreve os passos desse procedimento.

1. Instalar a sela e o colocar na coluna do lado 1
2. Fixar o bastão garra no colar e sela lado 1
3. Fixar o topo do bastão garra do lado 1 no Condutor
4. Instalar a sela e o colar na coluna base do lado 2 (Esquerdo)
5. Fixar o bastão garra invertido no colar e na sela do lado 2 (Esquerdo)
6. Fixar a carretilha com a corda de fibra sintética na ponta do bastão
7. Girar a cela de forma a erguer o bastão com a carretilha e corda na ponta
8. Instalar e enforçar com um estopo o alto do isolador usando um bastão universal
9. Conectar a corda no estopo
10. Usar um bastão universal com a chave catraca para soltar os parafusos de cima do conector
11. Retirar a parte de cima do conector com a chave catraca para soltar os parafusos de cima do conector
12. Suspender o cabo condutor através do bastão garra do lado 1 (Direito)
13. Tencionar a corda presa ao estopo

14. Tirar os parafusos de baixo do isolador com a chave catraca
15. Passar corda na base do isolador para auxiliar na descida
16. Puxar a corda para erguer o isolador
17. Fazer a retirada completa do isolador até o chão
18. Enforcar estopo no novo isolador
19. Suspender novo isolador
20. Engatar corda guia na base do novo isolador
21. Içar novo isolador até a posição correta com o auxílio da corda guia
22. Parafusar base do isolador na coluna usando a chave catraca
23. Soltar o colar do bastão garra do lado 1 (Direito) para baixar o cabo condutor no conector do isolador
24. Fechar os parafusos do conector usando o bastão universal com a catraca
25. Desconectar o bastão garra do conector
26. Retirar o bastão garra do lado 1 (Direto)
27. Retirar cela e colar garra do lado 1 (Direito)
28. Retirar estopo do isolador
29. Retirar cela e colar garra do lado 2 (Esquerdo)
30. Recolher equipamentos.

As perguntas são:

- Por que a atividade é feita apenas ao método a distância?
- Existe alguma situação em que pode ser feita através do potencial?
- Esses passos estão certos?
- Para um número razoável de eletricitistas, quais passos estariam associados a quais eletricitistas?

### A.3 Perguntas sobre Substituição do Pingo Aéreo pelo acesso a Dinstância

A lista a seguir descreve os passos desse procedimento.

1. Proceder uma análise preliminar do espaço físico, das condições da conexão e condutores
2. Medir com uma corda limpa, seca e previamente testada, o comprimento do novo condutor
3. Escovar os pontos de fixação do by-pass provisório
4. Instalar o by-pass provisório através de bastões de manobras nas duas extremidades do pingo a ser substituído
5. Soltar os parafusos que prendem o pingo ao condutor nas duas extremidades. Caso haja algum parafuso fundido ou espanado, cortá-lo com a serra em um bastão universal.
6. Mover as conexões para liberar o espaço para os novos conectores
7. Escovar os pontos das novas conexões
8. Prender novo pingo com os conectores no torque correto deixando somente as partes que serão colocadas no barramento com apenas dois parafusos provisórios em cada conector. De preferência que esses parafusos sejam ligeiramente maiores que os originais, a fim de facilitar a instalação.
9. Colocar no condutor uma carretilha com uma corda através de um bastão universal com gancho espiral. A corda deverá estar limpa, seca e testada através do testador de bastão. Este teste deverá ser feito no próprio local
10. Colocar na parte superior do novo pingo, próximo ao conector, um grampo manipulador de condutor instalado em um bastão universal
11. Amarrar uma das pontas da corda no bastão junto ao grampo manipulador de condutor (no caso de condutor de peso e/ou comprimento elevado).
12. Içar o pingo tomando todos os cuidados necessários para que o mesmo não toque ou se aproxime de potenciais diferentes.
13. Com auxílio de uma chave de fenda presa em um bastão universal, abrir a conexão da parte superior colocando a mesma no condutor
14. Encostar parcialmente as porcas, através do bastão com soquete multiangular.
15. Com um sacador e colocador de pingo, colocar restante dos parafusos

16. Prender as porcas e arruelas com fita crepe em um soquete, colocando-as nos parafusos com o bastão com soquete multiangular.
17. Através deste mesmo processo, substituir os parafusos provisórios.
18. Utilizar a chave com catraca para apertar aos poucos, um a um, para então dar o torque final.
19. Repetir a operação na conexão inferior
20. Fazer uma última inspeção para prevenir-se de eventuais rachaduras durante a operação de torque.
21. Transferir o bastão com grampo manipulador de condutor amarrado à corda para o pingo antigo
22. Segurar firmemente a corda.
23. Retirar totalmente os parafusos necessários para liberar o pingo antigo.
24. Retirar os jumpers provisórios.
25. Com auxílio de um outro grampo manipulador de condutor instalado em um bastão universal, sacar o condutor da conexão na parte inferior.
26. Baixar o pingo, tomando todos os cuidados necessários para que o mesmo não toque nem se aproxime de potenciais diferentes. O mesmo só poderá ser tocado com as mãos depois que estiver totalmente estendido no solo.
27. Retirar a carretilha com a corda do condutor.
28. Recolher o material.

As perguntas sobre esse procedimento.

- Esses passos estão corretos?
- O que é um pingo aéreo?
- Como um pingo aéreo é usado na subestação?
- Como é feita a escovação dos condutores?
- Nos passos, dá a entender que o pingo novo é colocando antes da remoção do pingo antigo, por quê isso acontece? Como é feito, em maiores detalhes, esse procedimento?
- Como é feita a instalação da carretilha? Há sub-passos?

- Quais são as ferramentas usadas em cada passo?
- Dado um certo número de eletricitas, quais passos estariam relacionados com quais eletricitas?

#### A.4 Perguntas sobre Substituição do Pingo Aéreo pelo acesso ao Potencial

A lista a seguir descreve os passos desse procedimento.

1. Proceder uma análise preliminar do espaço físico, das condições da conexão e condutores
2. Medir com uma corda limpa, seca e previamente testada, o comprimento do novo condutor
3. Escovar os pontos de fixação do by-pass provisório
4. Instalar o by-pass provisório através de bastões de manobras nas duas extremidades do pingo a ser substituído
5. Soltar os parafusos que prendem o pingo ao condutor nas duas extremidades. Caso haja algum parafuso fundido ou espanado, cortá-lo com a serra em um bastão universal.
6. Mover as conexões para liberar o espaço para os novos conectores
7. Escovar os pontos das novas conexões
8. Prender novo pingo com os conectores no torque correto deixando somente as partes que serão colocadas no barramento com apenas dois parafusos provisórios em cada conector. De preferência que esses parafusos sejam ligeiramente maiores que os originais, a fim de facilitar a instalação.
9. Colocar no condutor uma carretilha com uma corda através de um bastão universal com gancho espiral. A corda deverá estar limpa, seca e testada através do testador de bastão. Este teste deverá ser feito no próprio local

As perguntas são:

- Quais são as ferramentas usadas em cada passo?
- Dado um certo número de eletricitas, quais passos estariam relacionados com quais eletricitas?



## A.5 Substituição do Conector - Método a Distância

Descrição dos passos:

1. Colocar no condutor, uma carretilha com uma corda através de um bastão universal com gancho espiral.
2. Instalar próximo ao conector no jumper a ser liberado, um grampo manipulador de condutor instalado em um bastão universal
3. Amarrar uma das pontas da corda no bastão, junto ao grampo manipulador de condutor.
4. Segurar firmemente a corda e o bastão com grampo manipulador de condutor
5. Soltar os parafusos do conector com uma chave catraca e soquete adequado instalados em um bastão universal.
6. Havendo parafusos espanados ou fundidos, serrá-los com o arco de serra em um bastão universal
7. Sacar a conexão
8. Soltar parcialmente a corda que suporta o condutor, cuidado para não transferir o esforço para o by-pass provisório.
9. Amarrar a corda em um ponto firme no solo
10. Com um grampo manipulador de condutor em um bastão universal, segurar o condutor e escovar os pontos da nova conexão
11. Instalar o novo conector
12. Fazer última inspeção para prevenir-se de eventuais rachaduras durante a operação de torque.
13. Retirar o by-pass provisório
14. Retirar o bastão com a corda de segurança
15. Retirar a carretilha com a corda.
16. Recolher o material.

As perguntas são:

- O que é um Conector?
- Em quais condições um conector é usado?

- Nos passos, dá a entender que o conector antigo é removido antes da instalação do conector antigo, como se dá esse processo?
- Quais são as ferramentas usadas em cada passo?
- Como é feita a instalação da carretilha? Há sub-passos?
- Dado um certo número de eletricitistas, quais passos estariam relacionados com quais eletricitistas?

## A.6 Substituição do Conector - Método ao Potencial

Descrição dos passos:

1. Determinar maneira mais adequada para o acesso ao potencial.
2. Verificar distâncias entre fase terra e fase fase, e se estas permitem a execução segura do trabalho.
3. Determinar o melhor ponto para montar o equipamento de acesso, levando sempre em consideração as distâncias de segurança do eletricitista na escalada e durante o trabalho ao potencial bem como prever possíveis amarrações do condutor à escada ou andaime, após a desconexão do mesmo.
4. Montar o equipamento.
5. Escalar o andaime ou escada, sempre mantendo as distâncias seguras.
6. Entrar no potencial.
7. Fazer uma análise rigorosa das condições da conexão e do condutor.
8. Escovar os pontos de fixação do by-pass provisório.
9. Instalar o by-pass provisório.
10. Colocar no condutor a ser liberado, um grampo de torção com um pedaço de corda amarrada ao mesmo.
11. Amarrar esta corda no andaime ou escada para evitar o balanço ou queda do mesmo.
12. Sacar a conexão. Caso haja algum parafuso fundido ou espanado, cortá-lo com serra.
13. Retirar o conector com defeito.
14. Ajustar a corda que prende o condutor à escada ou andaime.
15. Escovar os pontos de fixação da nova conexão.

16. Soltar parcialmente a corda que prende o condutor à escada ou andaime.
17. Instalar o novo conector.
18. Fazer uma última inspeção para prevenir-se de eventuais rachaduras durante a operação de torque.
19. Retirar o by-pass provisório.
20. Retirar a corda de segurança.
21. Descer ferramentas e materiais.
22. Sair do potencial.
23. Desfazer ferramentas e materiais
24. Enrolar as fitas condutivas colocando-as no bolso da jaqueta e fechando-o e seguida.
25. Desmontar os equipamentos.
26. Recolher o material

Perguntas:

- Quais são as ferramentas usadas em cda passo?
- Dado um certo número de eletricitas, quais passos estariam relacionados com quais eletricitas?

## APÊNDICE B – Programa

### B.1 Regras

hasPermission(RHO,GOAL) :- hasObligation(RHO,GOAL).  
conditionViol(AGENT,GOAL,CONDITION):- requiresCirc(GOAL,CONDITION),notIsPresent(CO  
relationViol(AGENT,GOAL,RELATION):- requiresCirc(GOAL,RELATION),notIsPresent(RELAT  
entityViol(AGENT,GOAL,ENTITY) :- requiresEntity(GOAL,ENTITY),notIsPresent(ENTITY),sta  
negConseqFor(GOAL,AGENT,RISK,CONSEQUENCE) :- conditionViol(AGENT,GOAL,CONDIT  
negConseqFor(GOAL,AGENT,RISK,CONSEQUENCE) :- relationViol(AGENT,GOAL,RELATIO  
possOfNegConseqFor(OTHERRELATION) :- relationViol(AGENT,GOAL,RISK),affectsRels(REL  
negConseqFor(GOAL,AGENT,RISK,CONSEQUENCE) :- possOfNegConseqFor(OTHERRELATI  
stopped(GOAL):-entityViol(AGENT,GOAL,ENTITY).  
stopped(GOAL) :- negConseqFor(GOAL,AGENT,RISK,CONSEQUENCE).  
enabledToStart(AGENT,GOAL) :- adoptsRole(AGENT,RHO),hasPermission(RHO, GOAL),nextG  
stopped(GOAL) :- adoptsRole(AGENT,RHO),hasPermission(RHO, GOAL),lastGoal(GOAL,NEXT

### B.2 Raciocínio 1

adoptsRole(agente4,executor2).  
hasObligation(executor2,g1).  
requiresCirc(g1,relPanoGlicerina).  
instanceOfRel(relPanoGlicerina).  
starts(agente4,g1).  
notIsPresent(relPanoGlicerina).  
affectsRels(relPanoGlicerina,relBastaoGarraCondutor).  
affectsRels(relPanoGlicerina,relCordaEstropo).  
affectsRels(relPanoGlicerina,relChaveCatracaParafuso).  
affectsRels(relPanoGlicerina,relParafusoConector).  
affectsRels(relPanoGlicerina,relSoqueteParafuso).

affectsRels(relPanoGlicerina,relAgente4Corda).  
affectsRels(relPanoGlicerina,relEstropoCorda).

### B.3 Raciocínio 2

adoptsRole(agente2,executor1).  
adoptsRole(agente3,executor1).  
adoptsRole(agente4,executor2).  
hasObligation(executor1,g1).  
hasObligation(executor2,g1).  
starts(agente2,g1).  
starts(agente3,g1).  
starts(agente4,g1).  
requiresEntity(g1,pano).  
notIsPresent(pano).

### B.4 Raciocínio 3

adoptsRole(agente5,executor3).  
hasObligation(executor3,g11).  
starts(agente5,g11).  
requiresCirc(g11,umidade70).  
instanceOfCond(umidade70).  
notIsPresent(umidade70).  
hasRisk(umidade70,eletrocutado,morte).

### B.5 Raciocínio 4

adoptsRole(agente4,executor2).  
hasObligation(executor4,g15).  
starts(agente4,g15).  
requiresCirc(g15,relChaveCatracaParafuso).  
instanceOfRel(relChaveCatracaParafuso).

notIsPresent(relChaveCatracaParafuso).

hasRisk(relChaveCatracaParafuso,eletrocutado,morte).

## B.6 Raciocínio 5

requiresCirc(g19,relParafusoConector).

hasObligation(executor3,g19).

hasObligation(executor4,g19).

hasObligation(executor5,g19).

starts(agente5,g19).

starts(agente6,g19).

starts(agente7,g19).

adoptsRole(agente5,executor3).

adoptsRole(agente6,executor4).

adoptsRole(agente7,executor5).

hasRisk(relParafusoConector,eletrocutado).

possOfNegConseqFor(relParafusoConector).

happensNegConseqFor(relParafusoConector).

instanceOfRel(relParafusoConector).

hasRisk(relParafusoConector,eletrocutado,morte).

## B.7 Raciocínio 6

```
1 function stopped(context)
2 {
3   context.forEach(espObj => {
4     if(espObj.stopped)
5       true;
6     else
7       false;
8   });
9 }
10
11 function allAgentObliged(context,goal,obligatedToThisGoal)
12 {
13   let whoIsObliged = [];
14   context.forEach(espObj => {
15     if(espObj.goal.includes(goal))
16     {
17       if(!espObj.role.includes(obligatedToThisGoal))
18         return false;
19     }
20   });
21
22   return true;
23 }
24
25 function ifNotStopped(context,goal,obligatedToThisGoal)
26 {
27   if(stopped(context))
28     return false;
29
30   if(!allAgentObliged(context,goal,obligatedToThisGoal))
31     return false;
32   return true;
33 }
34
```

Figura 12 – Raciocínio 6 parte 1

```

34
35 var context = [
36     {
37         agent:"agente2",
38         goal:"g23",
39         stopped:false,
40         role:"executor1",
41         obligationOfThisRole:["g23"]
42     },
43     {
44         agent:"agente3",
45         goal:"g23",
46         stopped:false,
47         role:"executor1",
48         obligationOfThisRole:["g23"]
49     },
50     {
51         agent:"agente4",
52         goal:"g23",
53         stopped:false,
54         role:"executor2",
55         obligationOfThisRole:["g23"]
56     },
57     {
58         agent:"agente5",
59         goal:"g23",
60         stopped:false,
61         role:"executor3",
62         obligationOfThisRole:["g23"]
63     },
64     {
65         agent:"agente7",
66         goal:"g23",
67         stopped:false,
68         role:"executor5",
69         obligationOfThisRole:[]
70     }
71 ];
72
73
74 var goal = 'g23';
75 if(ifNotStopped(context,goal,['executor1','executor2','executor3']))
76     console.log("reached(goal): " + goal + " is true");
77 else
78     console.log("reached(goal): " + goal + " is false");

```

Figura 13 – Raciocínio 6 parte 2