

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

**NOME DO AUTOR**

**TÍTULO EM PORTUGUÊS**

**DISSERTAÇÃO**

**CURITIBA**

**2019**

**NOME DO AUTOR**

**TÍTULO EM PORTUGUÊS**

Dissertação apresentada ao Programa de Pós-graduação em Engenharia Elétrica e Informática Industrial da Universidade Tecnológica Federal do Paraná como requisito parcial para obtenção do grau de “Mestre em Ciências” – Área de Concentração: Informática Industrial.

Orientador:      Nome do Orientador

**CURITIBA**

**2019**

---

Dados Internacionais de Catalogação na Publicação

---

T137    Sobrenome, Nome

          Título em português/ Nome do Autor. – 2019.

          39 f. : il. ; 30 cm

          Orientador: Nome do Orientador.

          Dissertação (Mestrado) – Universidade Tecnológica Federal do Paraná. Programa de Pós-graduação em Engenharia Elétrica e Informática Industrial. Curitiba, 2019.

          Bibliografia: f. 35-37.

          1. Teoria do controle. 2. Redes de comutação. 3. TCP/IP (Protocolo de rede de computação), ...

CDD (22. ed.) 621.3

---

Biblioteca xxxxxx

Título da Dissertação Nº 596:

**“Esquema de Controle de Congestionamento para  
TCP Baseado na Banda Disponível”.**

por

**Marcos Talau**

Esta dissertação foi apresentada como requisito parcial à obtenção do grau de MESTRE EM CIÊNCIAS – Área de Concentração: Telemática, pelo Programa de Pós-Graduação em Engenharia Elétrica e Informática Industrial – CPGEI – da Universidade Tecnológica Federal do Paraná – UTFPR – Câmpus Curitiba, às 09h30min. do dia 04 de maio de 2012. O trabalho foi aprovado pela Banca Examinadora, composta pelos professores:

Visto da coordenação:

Texto da dedicatória.

## **AGRADECIMENTOS**

Texto dos agradecimentos.

Texto da epígrafe.

## **RESUMO**

SOBRENOME, Nome. TÍTULO EM PORTUGUÊS. 39 f. Dissertação – Programa de Pós-graduação em Engenharia Elétrica e Informática Industrial, Universidade Tecnológica Federal do Paraná. Curitiba, 2019.

Texto do resumo (máximo de 500 palavras).

**Palavras-chave:** Palavra-chave 1, Palavra-chave 2, ...



## **ABSTRACT**

SOBRENOME, Nome. TITLE IN ENGLISH. 39 f. Dissertação – Programa de Pós-graduação em Engenharia Elétrica e Informática Industrial, Universidade Tecnológica Federal do Paraná. Curitiba, 2019.

Abstract text (maximum of 500 words).

**Keywords:** Keyword 1, Keyword 2, ...

## LISTA DE FIGURAS

FIGURA 1	– Arvore de objetivos definido pelo modelo Moise (HÜBNER et al., 2002)	22
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, $\langle ident \rangle$ é usado para denotar uma <i>string</i> e $\langle int \rangle$ inteiros. Os termos $\langle b-prop \rangle$ e $\langle i-prop \rangle$ são usados para designar dois tipos de conjuntos de proposições que são disjuntos entre si	24
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).	25
FIGURA 4	– 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).	29
FIGURA 5	– Ontologia que descreve <i>Domain Model</i> no model V3S (BAROT et al., 2013)	30

## LISTA DE TABELAS

TABELA 1	– Construtores da linguagem <i>ACTIVITY-DL</i> (BAROT et al., 2013) .....	31
TABELA 2	– As pré-condições possíveis para as atividades (BAROT et al., 2013) .....	32

## **LISTA DE QUADROS**

## **LISTA DE SIGLAS**

## LISTA DE SÍMBOLOS

## SUMÁRIO

<b>1 INTRODUÇÃO</b>	<b>13</b>
1.1 MOTIVAÇÃO	13
1.2 RELEVÂNCIA	13
1.3 OBJETIVOS	13
1.3.1 Objetivo Geral	13
1.3.2 Objetivos Específicos	13
<b>2 FUNDAMENTAÇÃO TEÓRICA</b>	<b>14</b>
2.1 AGENTES	14
2.2 ARTEFATOS	16
2.3 SISTEMA MULTIAGENTE	18
2.3.1 Conceitos Gerais de uma Organização de Sistemas Multiagentes	18
2.3.2 Formalização de Conceitos Específicos para SMA	20
2.4 NORMAS	23
2.5 RISCOS	25
2.6 POSSIBILIDADES	25
<b>3 METODOLOGIA</b>	<b>27</b>
3.1 ANÁLISE DOS MODELOS	27
3.2 ACOMPANHAMENTO DE PROFISSIONAIS EM ATIVIDADE DE RISCO	27
3.3 CONSTRUÇÃO DO MODELO	27
3.4 IMPLEMENTAÇÃO	27
<b>4 RESULTADOS</b>	<b>28</b>
4.1 ESTRUTURA CONCEITUAL	28
4.1.1 Módulos	28
4.1.2 Conjuntos	28
4.1.3 Predicados	28
4.1.4 Regras	28
4.2 UML	28
4.2.1 Diagrama de Classes	28
4.2.2 Diagrama de Atividades	28
4.3 CASO DE ESTUDO	28
4.4 RACIOCÍNIO	28
4.5 VALIDAÇÃO	28
<b>5 ANÁLISE COMPARATIVA</b>	<b>29</b>
5.1 MOISE+	29
5.1.1 Estrutura	29
5.1.2 Análise comparativa	29
5.2 DASTANI	29
5.2.1 Estrutura	29
5.2.2 Análise comparativa	30
5.3 V3S	30
5.3.1 Estrutura	30

5.3.2 Análise comparativa .....	32
5.4 NORMMAS .....	32
5.4.1 Estrutura .....	32
5.4.2 Análise comparativa .....	33
5.5 PERSPECTIVA GENÉRICA .....	33
<b>6 CONCLUSÃO .....</b>	<b>34</b>
6.1 AVALIAÇÃO DOS OBJETIVOS .....	34
6.2 TRABALHOS FUTUROS .....	34
<b>REFERÊNCIAS .....</b>	<b>35</b>
<b>Apêndice A – NOME DO APÊNDICE .....</b>	<b>38</b>
<b>Anexo A – NOME DO ANEXO .....</b>	<b>39</b>



## **1 INTRODUÇÃO**

### **1.1 MOTIVAÇÃO**

### **1.2 RELEVÂNCIA**

### **1.3 OBJETIVOS**

#### **1.3.1 OBJETIVO GERAL**

#### **1.3.2 OBJETIVOS ESPECÍFICOS**

## 2 FUNDAMENTAÇÃO TEÓRICA

### 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 *autonomo* é o cerne de noção que se tem por agente (JENNINGS; LESPÉRANCE, 2000). Apesar disso, a construção de um modelo computacional não pode ser feito sem uma definição. Assim sendo, nesse texto um agente é um sistema computacional que está situado em um dado ambiente e que apresenta comportamento autonomo (JENNINGS; LESPÉRANCE, 2000) (JENNINGS; LESPÉRANCE, 2000). Não apenas isso, mas um agente faz uso do seu comportamento autonomo com o propósito de atingir objetivos que a ele é designado (JENNINGS; LESPÉRANCE, 2000) (JENNINGS; LESPÉRANCE, 2000).

Como a definição de agente faz uso do conceito de ambiente, não é possível tratar esse tópico de forma consistente sem considerar a semântica deste termo. Assim sendo, ambiente é aquilo que apresenta as propriedades listadas a seguir (RUSSELL; NORVIG, 2003) (JENNINGS; LESPÉRANCE, 2000);

- *Acessibilidade vs Inacessibilidade*; Um ambiente acessível é aquele onde 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 onde 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 quando 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 existe um número finito de ações e percepções.

Outro aspecto que está contido na definição consiste no conceito expresso pelo termo *autônomo*. Assim sendo, a pesquisa retratada neste texto trabalha esse termo da seguinte forma; Uma entidade que possui essa natureza tem a capacidade agir por si mesmo. Essa entidade não precisa de nenhuma outra entidade externa (ex. ser humano) para realizar decisões (JENNINGS; LESPÉRANCE, 2000) (JENNINGS; LESPÉRANCE, 2000).

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

- Se a temperatura estiver abaixo (ou acima) 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 dado evento no ambiente, esses programas realizam uma dada atuação (JENNINGS; LESPÉRANCE, 2000).

Os exemplos presentes neste texto são apropriados dentro do conceito de agentes. Contudo, esses exemplos não se enquadram dentro da definição de agentes inteligentes (JENNINGS; LESPÉRANCE, 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), pro-atividade (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 (e possivelmente humanos) a fim de poder satisfazer os próprios objetivos) (JENNINGS; LESPÉRANCE, 2000) (RUSSELL; NORVIG, 2003).

Os agentes podem ser definidos em categorias. Um desses são agentes 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; LESPÉRANCE, 2000).

Outra categoria de agentes são aqueles que possuem estados. Essas entidades possuem uma dada estrutura de dados interna que são considerados quando agente toma uma certa decisão (JENNINGS; LESPÉRANCE, 2000).

Uma outra maneira de analisar os agentes se dá por meio das arquiteturas e modelos disponíveis para representar os agentes (tomada de decisão, estado interno). 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 onde os agentes realizam deduções lógicas para tomar uma decisão (GENESERETH; NILSSON, 1987), a segunda arquitetura consiste nos agentes reativos que tomam decisões com base em um dado mapeamento de uma certa situação em uma dada ação (BONASSO et al., 1995), a terceira arquitetura é *BDI* cuja decisão 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 diversas camadas abstração a cerca do ambiente (WOOLDRIDGE; JENNINGS, 1995) (JENNINGS; LESPÉRANCE, 2000).

## 2.2 ARTEFATOS

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

Um agente que é caracterizado como *goal-governed* são aqueles que apresentam capacidades cognitivas, que podem representar os seus respectivos objetivos e, portanto, são capazes de definir seus objetivos em interesse (CASTELFRANCHI et al., 2018) (RICCI et al., 2006). Em contraste com isso, *goal-oriented* consiste nos agentes que são programados com a finalidade de alcançar um determinado objetivo (CASTELFRANCHI et al., 2018) (RICCI et al., 2006). Em um sistema multiagente muitas vezes uma dada entidade não é adequadamente

representada por nenhuma dessas duas categorias.

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

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

Uma outra distinção entre agentes e artefatos se torna claramente explícita quando verificar sobre o ponto de vista conceitual e filosófico. Isso, pois agentes apresentam a capacidade de se comunicar com outros agentes, em contraste com isso artefatos não apresentam essa condição (RICCI et al., 2006).

Existem quatro elementos que podem ser usados para caracterizar um artefato (RICCI et al., 2006), que são; *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 et al., 2006). A *instruções de operação* consiste na descrição de como fazer o uso das funcionalidades do artefato. Isso implica protocolos de uso das operações que devem ser invocadas pelo agente (RICCI et al., 2006). A *Funcionalidade* do artefato consiste no propósito definido pelo programador do sistema (RICCI et al., 2006). A *estrutura-comportamento* consiste nos aspectos internos do artefato, que é como o artefato é implementado a fim de providenciar as suas funcionalidades (RICCI et al., 2006).

Cartago (*Common "Artefacts for Agents" Open Framework*) é um framework usado para especificar as relações entre agentes e artefatos. Tendo em vista o fato de ser um dos principais *frameworks* na descrição das 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*, *artifact* e *workspace* (RICCI et al., 2007).

*Agent bodies* é o que possibilita a inteligência do agente se relacionar com o ambiente. 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 (captam sinais do ambiente em sua volta). A relação completa entre agente-*agent*

*bodies*-artefato, no Cartago, é dada da seguinte forma; eventos observáveis são gerados pelos artefatos, sensores (componentes presentes no *agent bodies*) são sensibilizados, essa informação é transmitida para a inteligência do agente (esta, 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 dada ação é produzida no ambiente de trabalho.

*Artifacts* (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 *artefat 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 dada ação é executada. Os artefatos também apresentam nome completo que inclui o nome do(s) *workspace(s)* onde está logicamente localizados (RICCI et al., 2007).

*Workspaces* é a localização lógica dos artefatos ocorre dentro de *workspaces*, que pode ser usado para definir a topologia do ambiente de trabalho. Neste âmbito, o *workspaces* são feitos com a finalidade de especificar nome lógico e id. Está dentro do escopo deste item definir uma topologia de ambiente possibilitando que 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 et al., 2002) (ABBAS et al., 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 et al., 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 et al., 2015). Ainda sobre isso, uma organização SMA 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 et al., 2015).

O subtópico *Conceitos Gerais de uma Organização de Sistemas Multiagentes* tem como finalidade detalhar melhor os elementos presentes da Teoria da Organização dentro do contexto de SMA em relação a esse estudo. Já o Subtópico *Formalização de Conceitos Específicos para SMA* tem como por objetivo realizar uma verificação analítica dos elementos presentes no modelo de SMA denominando por *MOISE+*. Os conceitos que serão analisados são; objetivos, planos e papéis *organisation of multiagents system*.

### 2.3.1 CONCEITOS GERAIS DE UMA ORGANIZAÇÃO DE SISTEMAS MULTIAGENTES

A finalidade desta subseção consiste trabalhar com uma maior riqueza de detalhes todos os conceitos que constituem a ideia de uma organização de um sistema multiagente.

**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 são sujeitas a classificação e ocorrem com diferentes frequências e em diferentes regiões dentro das definições espaciais da organização (ABBAS et al., 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 et al., 2015).

**Composição** Uma organização é composta por elementos menores. No caso dos multiagentes, os elementos atômicos que estruturam a organização são os agentes (ABBAS et al., 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 et al., 2015) ?.

**Coordenação:** Todo sistema é dependente de algum dado recurso. Assim sendo, se faz necessário que esse recurso seja utilizado de forma inteligente a fim de que possa se manter ao longo do tempo. Para que 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 relacionando aos efeitos combinados resultam influenciam negativamente nas eficiencias. Portanto, para manter a eficiência organizacional se faz necessário a existência de elementos otimizadores sobre os padrões de atividades (ABBAS et al., 2015).

**Recursividade:** Uma organização é constituída por sub-organizações. Isso ocorre em multiplos níveis de estrutura e se dá por intermédio de um padrão recursivo (ABBAS et al., 2015).

**Representação Multi-Nível e Causalidade:** Uma organização é estruturada por suborganizações em diferentes níveis estruturas. Isso, por sua vez, resultam em atividades ocorrendo em diferentes escalas espaciais, temporais e estruturais. Como consequencia disto, as cadeias causais presentes em estruturas organizacionais são processos multi-níveis (ABBAS et al., 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, existe modelos abstratos de sistemas auto-organizaveis 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 a respeito disto consiste no conceito de *Poder* o qual é entendido como a capacidade de influenciar uma dada organização (ABBAS et al., 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 um organização apresentam três interpretações, essas são; como estruturas (especificações procedurais do que deve ser feito), como coação as ações definindo o que pode e não pode ser feito e como um compilado das experiências (ABBAS et al., 2015).

**Incerteza:** Não é possível conceber o conceito de uma organização sem ao menos entende-la como uma estrutura que distribui informação em si mesma. Sobre essa 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 *framework*, o *MOISE+* trata a rigor acadêmico na ótica da computação clássica a tratativa dada para os conceitos em interesse a esse estudo. Assim sendo, um estudo aprofundado do modelo, bem como dos textos em referência, satisfaz com exatidão os fundamentos teóricos para as análises em interesse.

A constituição do *MOISE+* é estruturada em três categorias de especificação, essas são; estrutural, funcional e deontica. O texto a seguir exibe com maior requie de detalhes cada uma dessas especificações.

A especificação estrutural acontece em três níveis, individual, social e coletivo. O nível individual trata de definir os papéis  $\rho$  dos agentes. Uma possível entre os papéis acontece por intermédio da hereditariedade em que se  $\rho'$  é filho de  $\rho$ . Isso implica afirmar que  $\rho'$  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 et al., 2002) (FERBER; GUTKNECHT, 1998) (FOX et al., 1996) (CARRON; BOISSIER, 2001). Em termos formais, essa relação é dada por;

$$\rho_a \sqsubset \rho_b$$

O nível social estabelece relações de ligação dado pelo predicado  $link(\rho_s, \rho_d, t)$ . Existe 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 et al., 2002) (HÜBNER et al., 2002) (CARRON; BOISSIER, 2001). O *MOISE+* define as seguintes relações de implicabilidade

$$\begin{aligned} link(\rho_s, \rho_d, auth) &\rightarrow link(\rho_s, \rho_d, com) \\ link(\rho_s, \rho_d, com) &\rightarrow link(\rho_s, \rho_d, acq) \end{aligned} \tag{1}$$

O modelo também determina como se dá as relações de hereditariedade para o predicado de *link*, é dado por (HÜBNER et al., 2002) (CARRON; BOISSIER, 2001);

$$\begin{aligned}
& link(\rho_s, \rho_d, t) \wedge \rho'_s \sqsubset \rho'_s \rightarrow link(\rho'_s, \rho_d, t) \\
& link(\rho_s, \rho_d, t) \wedge \rho'_d \sqsubset \rho'_d \rightarrow link(\rho_s, \rho'_d, t)
\end{aligned} \tag{2}$$

O nível coletivo determina a existência de compatibilidade entre os papeis (HÜBNER et al., 2002). Essa é uma relação reflexiva e transitiva de determina que se um 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 et al., 2002) (CASTELFRANCHI, 1995).;

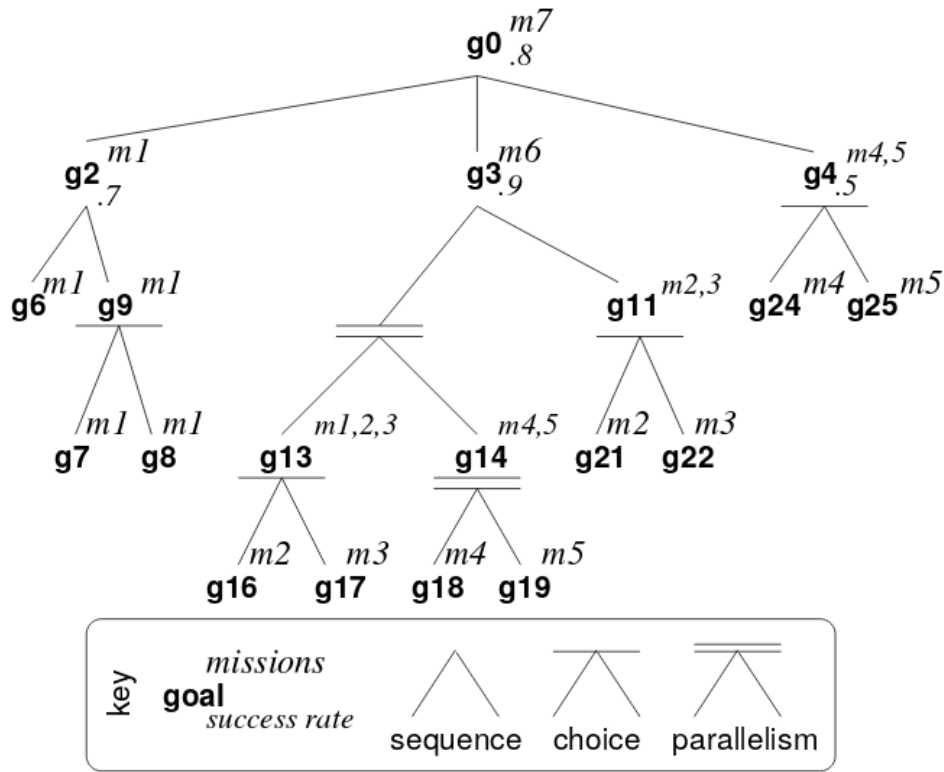
$$\rho_a \bowtie \rho_b \wedge \rho_a \neq \rho_b \wedge \rho_a \sqsubset \rho' \rightarrow \rho' \bowtie \rho_b$$

O nível coletivo também apresenta o conceito de grupo dado por  $gt$  e constituído por;

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

Em que  $R$  é o conjunto dos papeis 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 et al., 2002).

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



**Figura 1: Arvore de objetivos definido pelo modelo Moise (HÜBNER et al., 2002)**

A figura 1 define três tipos de relação de subobjetivos; *sequence* onde todos os subobjetivos 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 *framework* 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;

$$m_k = \{g_n, \dots, g_m\}$$

A Especificação Deontica define predicados para estabelecer permissões e obrigações entre os papéis e as missões. Toda obrigação implica necessariamente em uma permissão. A relação a seguir estabelece isso (HÜBNER et al., 2002) (CASTELFRANCHI, 1995);

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

$$obl(\rho, m, tc) \wedge \rho \sqsubset \rho' \rightarrow obl(\rho', m, tc) \quad (3)$$

$$per(\rho, m, tc) \wedge \rho \sqsubset \rho' \rightarrow per(\rho', m, tc) \quad (4)$$

$$(5)$$

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, pois há diversos estudos que tratam o conceito de norma sobre perspectivas diferentes. Por exemplo, os estudos (ESTEVA et al., 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 sobre a ótica da lógica deontica e é usada para especificar aos agentes com dado papel as suas obrigações em relação as missões (HÜBNER et al., 2002) (HÜBNER et al., 2002).

O estudo (DASTANI et al., 2009) apresenta uma linguagem formal para especificar sistemas multiagentes normativos. Essa 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>  $\Rightarrow$  <i-literals> )+;
<sanctions>  := ( <i-literals>  $\Rightarrow$  <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,  $\langle ident \rangle$  é usado para denotar uma *string* e  $\langle int \rangle$  inteiros. Os termos  $\langle b-prop \rangle$  e  $\langle i-prop \rangle$  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 pro meio de *effects* define como se dá a transição de estados do sistema. A tag  $\langle actionName \rangle$  representa os eventos que geram transição dos estados. Uma norma, portanto, é definida em termos de *Counts-as rules*. O termo  $\langle counts-as \rangle$  aponta para transição entre  $\langle literals \rangle$  e  $\langle i-literals \rangle$ . Isso representa os fatos que resultam em violações. Portanto, 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  $\langle sanctions \rangle$  e esses, por sua vez, para a transição entre  $\langle i-literals \rangle$  e  $\langle b-literals \rangle$  sendo que esses  $\langle i-literals \rangle$  advem 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 (DASTANI et al., 2009).

A figura 4 apresenta um programa escrito na linguagem proposta em

```

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} ⇒ {viol1},
                 {in_train, -ticket} ⇒ {viol1}
Sanction rules: {viol1} ⇒ {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).**

Esse programa apresenta contém um agente chamado de *passenger* (especificações desse agente são detalhadas com maior rigor em *PassProg*). Os *Facts* deste programa são - *at\_plataform* (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\_plataform* e *at\_plataform*, 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 se um *ticket*. A segunda define que uma violação 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.

## 2.5 RISCOS

(RASMUSSEN, 1997) (FADIER et al., 2003)

- Comentar as observacoes de Rasmussen
- Definir o risco dentro da estrutura do framework (FADIER et al., 2003)

## 2.6 POSSIBILIDADES

(DUBOIS et al., 1994)

- Comentar a ocorrência das possibilidades dentro do contexto da lógica fuzzy
- Descrever a ocorrência da lógica no contexto modal (encontrar paper?).

### **3 METODOLOGIA**

#### **3.1 ANÁLISE DOS MODELOS**

#### **3.2 ACOMPANHAMENTO DE PROFISSIONAIS EM ATIVIDADE DE RISCO**

#### **3.3 CONSTRUÇÃO DO MODELO**

#### **3.4 IMPLEMENTAÇÃO**



## **4 RESULTADOS**

### **4.1 ESTRUTURA CONCEITUAL**

#### **4.1.1 MÓDULOS**

#### **4.1.2 CONJUNTOS**

#### **4.1.3 PREDICADOS**

#### **4.1.4 REGRAS**

### **4.2 UML**

#### **4.2.1 DIAGRAMA DE CLASSES**

#### **4.2.2 DIAGRAMA DE ATIVIDADES**

### **4.3 CASO DE ESTUDO**

### **4.4 RACIOCÍNIO**

### **4.5 VALIDAÇÃO**

## 5 ANÁLISE COMPARATIVA

### 5.1 MOISE+

O Moise+ é usado para realizar a especificação de sistemas multiagentes. Para cumprir com essa finalidade, existe três tipos de especificação que são; Estrutural, Funcional, e Deontica.

#### 5.1.1 ESTRUTURA

#### 5.1.2 ANÁLISE COMPARATIVA

### 5.2 DASTANI

#### 5.2.1 ESTRUTURA

```

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} ⇒ {viol1},
                 {in_train, -ticket} ⇒ {viol⊥}
Sanction rules: {viol1} ⇒ {fined10}

```

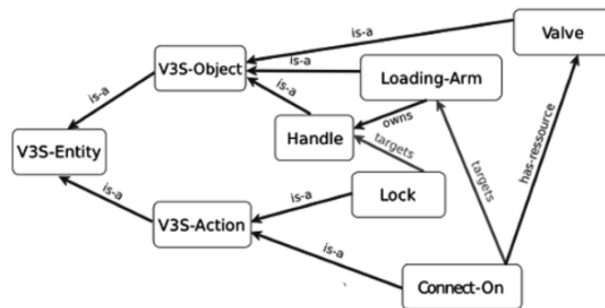
**Figura 4:** Um programa descrito na linguagem proposta neste estudo onde um agente representa um passageiro em uma estação de trem que pode entrar com ou sem um *ticket* na plataforma e no trem (DASTANI et al., 2009).

A figura 4 apresenta um programa que contém um agente com nome *passenger*. O agente pode estar ou não na plataforma e no trem, sem ou com *ticket*. Se o agente entrar na plataforma ou no trem sem o *ticket*, então esse agente cometeu uma violação. Para este programa, a sanção da violação que ocorre por entrar na plataforma sem o *ticket* resulta em uma punição onde o agente deve pagar 10 Euros pelo ocorrido (DASTANI et al., 2009).

## 5.2.2 ANÁLISE COMPARATIVA

## 5.3 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 5 exibe a estrutura de classe desta ontologia.



**Figura 5: Ontologia que descreve *Domain Model* no model V3S (BAROT et al., 2013)**

### 5.3.1 ESTRUTURA

*Activity Model* é estruturado sobre uma linguagem de descrição conhecido por *ACTIVITY-DL*. Essa linguagem usa 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  $Y$

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)**

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

No que tange a questões referentes a segurança e violação, a linguagem *ACTIVITY-DL* define *tags* no estudo (FADIER et al., 2003). Essas *tags* são BCTUs, BATUs. BCTUs representam situações informais entre os atores de um determinado campo, por exemplo; trabalhar com produtos químicos sem usar equipamentos de proteção individual. BATUs corresponde a práticas de risco tolerado. Existe dois tipos de BATUs, primeiro - quando realizado por obrigação ou interesse de conforto (quando os operadores acham impossível realizar a operação respeitando as instruções), segundo - ocorre durante a operação do sistema com o propósito de evitar que o sistema deixe de funcionar.

*Activity Model* apresenta uma categorização de pré-condições. A tabela 2 detalhe essa categorização.

*Risk Models* é a parte do modelo que define a análise de risco. Existe duas categorias; risco de análise clássico 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 et al., 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.

<b>Categoria</b>	<b>Pré-condição</b>	<b>Descrição</b>
Condições para perceber	Nomológico	Descreve o estado do 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 do mundo necessário para uma boa realização da atividade de acordo com prescrito em procedimento. Exemplo: Para desconectar o tubo, a proteções devem ser desgastado.
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)**

### 5.3.2 ANÁLISE COMPARATIVA

### 5.4 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).

#### 5.4.1 ESTRUTURA

**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 defição 1 pode ser compreendida sobre o seguinte exemplo;

*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;

$$\langle obligation, action, valid(Passport), accept(Passport), loss(5) \rangle \quad (6)$$

*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.

#### 5.4.2 ANÁLISE COMPARATIVA

#### 5.5 PERSPECTIVA GENÉRICA

## **6 CONCLUSÃO**

### **6.1 AVALIAÇÃO DOS OBJETIVOS**

### **6.2 TRABALHOS FUTUROS**

## REFERÊNCIAS

- ABBAS, H.; SHAHEEN, S.; AMIN, M. H. Organization of multi-agent systems: An overview. **International Journal of Intelligent Information Systems**, 06 2015.
- 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>>.
- 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.
- 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)>.
- BONASSO, R. P. et al. Experiences with an architecture for intelligent, reactive agents. In: . [S.l.: s.n.], 1995. v. 9, p. 187–202.
- 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>>.
- CARRON, T.; BOISSIER, O. Towards a temporal organizational structure language for dynamic multi-agent systems. 01 2001.
- CASTELFRANCHI, C. Commitments: From individual intentions to groups and organizations. In: **ICMAS**. [S.l.: s.n.], 1995.
- CASTELFRANCHI, C. et al. Social trust: A cognitive approach. 12 2018.
- 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.
- 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.
- DUBOIS, D.; LANG, J.; PRADE, H. Possibilistic logic. In: \_\_\_\_\_. [S.l.: s.n.], 1994. p. 439–513.
- 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.



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

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>>.

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>>.

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

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>>.

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>>.

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.

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.

HübNER, J.; SICHMAN, J.; BOISSIER, O. Moise+: Towards a structural, functional, and deontic model for mas organization. In: . [S.l.: s.n.], 2002. p. 501–502.

JENNINGS, N. R.; LESPÉRANCE, 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.

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

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

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.

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

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

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

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.

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>>.

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>>.

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.

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.

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>>.

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

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.

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.

**APÊNDICE A – NOME DO APÊNDICE**

**ANEXO A – NOME DO ANEXO**