UNIVERSIDADE FEDERAL DE SANTA CATARINA
CENTRO TECNOLÓGICO - CTC
PROGRAMA DE PÓS-GRADUAÇÃO EM ENGENHARIA MECÂNICA

Cláudio Ernesto Ponce Saldias

Protótipo de Sistema de Instrução Virtual para Treinamento de Pilotos de Aviões BOEING 737-800.

Florianópolis
2019

Claudio Ernesto Ponce Saldias

Protótipo de Sistema de Instrução Virtual para Treinamento de Pilotos de Aviões BOEING 737-800.

Tese de Doutorado submetida ao Programa de Pós-Graduação em Engenharia Mecânica da Universidade Federal de Santa Catarina para a obtenção do Grau de Doutor em Engenharia Mecânica.
Orientador: Prof. Carlos Alberto Martin, Dr.
Coorientador: Prof. Ricardo Azambuja Silveira, Dr.

Florianópolis
2019

Ficha de identificação da obra

PONCE SALDIAS, CLAUDIO ERNESTO
Protótipo de Sistema de Instrução Virtual para Treinamento de Pilotos de Aviões BOEING 737-800. / CLAUDIO ERNESTO PONCE SALDIAS ; orientador, CARLOS ALBERTO MARTIN, coorientador, RICARDO AZAMBUJA SILVEIRA, 2019.
162 p.

Tese (doutorado) - Universidade Federal de Santa Catarina, Centro Tecnológico, Programa de Pós-Graduação em Engenharia Mecânica, Florianópolis, 2019.

Inclui referências.

1. Engenharia Mecânica. 2. Simulador de Vôo. 3. Treinamento virtual inteligente. 4. Sistemas tutores inteligentes. I. MARTIN, CARLOS ALBERTO. II. AZAMBUJA SILVEIRA, RICARDO. III. Universidade Federal de Santa Catarina. Programa de Pós-Graduação em Engenharia Mecânica. IV. Título.

Claudio Ernesto Ponce Saldias

Protótipo de Sistema de Instrução Virtual para Treinamento de Pilotos de Aviões BOEING 737-800.

O presente trabalho, em nível de doutorado, foi avaliado e aprovado por banca examinadora composta pelos seguintes membros:

Profa. Rosa Maria Vicari, Dra.(Relatora da banca)
Universidade Federal de Rio Grande do Sul

Profa. Patrícia Augustin Jaques Maillard, Dra.
UNISINOS

Prof. Klaus Janschek, Dr.techn.
Technische Universität Dresden, Germany

Profa. Eliane Pozzebon, Dra.Eng.
Universidade Federal de Santa Catarina

Prof. Henrique Simas, Dr. Eng.
Universidade Federal de Santa Catarina

Certificamos que esta é a **versão original e final** do trabalho de conclusão que foi julgado adequado para obtenção do título de doutor em Engenharia Mecânica.

Assinado de forma digital por Jonny Carlos da Silva:51451506449
Dados: 2019.11.04 13:40:08 -03'00'

Prof. Jonny Carlos da Silva, Dr.
Coordenador do Programa

Documento assinado digitalmente
Carlos Alberto Martin
Data: 04/11/2019 14:06:37-0200
CPF: 153.056.940-00

Prof. Carlos Alberto Martin Dr.-Ing.
Orientador

Florianópolis, 2019

Este trabalho é dedicado a Deus e a minha família.

AGRADECIMENTOS

Quero agradecer, em primeiro lugar, a minha família, pelo apoio incondicional. Quero agradecer, também, ao meu professor orientador e aos professores envolvidos. Agradeço o importante apoio financeiro da CAPES e do CNPQ.

Todo sistema físico procura o estado de mínima energia e todo sistema inteligente natural ou artificial procura o estado de máximo conforto.

Claudio Ernesto Ponce Saldias, 2014

RESUMO

Na área da aviação civil e militar, a baixa disponibilidade de instrutores qualificados em escolas homologadas, e o problema da imparcialidade no julgamento entre instrutor e aluno faz com que seja atrativa a ideia de desenvolver um sistema de treinamento inteligente, que possa auxiliar ou, parcialmente, substituir o instrutor humano. O objetivo deste trabalho é apresentar uma proposta de sistema instrutor virtual inteligente para treinamento com simuladores para atender as necessidades de disponibilidade de instrutores humanos e de melhora na imparcialidade de julgamento na avaliação. Para abordar este objetivo foi definido um modelo de sistema de instrutor virtual, operado por agentes virtuais, com raciocínio baseado em conhecimento, desenvolvido para permitir projetar tutores virtuais para treinamento com veículos usando simuladores, podendo ser estendido para treinamento de operação máquinas e processos. O protótipo foi desenvolvido seguindo o modelo de sistema de instrução virtual. A primeira fase do desenvolvimento do protótipo foi a aquisição de conhecimento por meio de pesquisa bibliográfica e entrevistas com especialistas envolvidos com a operação do avião. A segunda fase foi a representação de conhecimento em regras e ontologias. As bases de conhecimento representam o modelado qualitativo e quantitativo (dinâmico) do avião e o ambiente (cenário); o conhecimento que envolve a operação do avião (navegação, regulamento aeronáutico, meteorologia, segurança no voo); o conhecimento que envolve o que e como ensinar e avaliar; o conhecimento de gerenciamento do sistema (protocolos de comportamento, comunicação e resolução de conflito dos agentes, modelo emocional, entre outros); e uma representação de modelo de aluno (um registro do perfil de aprendizado e do emocional do aluno). As funcionalidades principais do protótipo são quatro, a primeira é comum entre os sistemas de treinamento existentes, as outras são produto deste trabalho: Treinar (ensinar, assistir e avaliar) um piloto; detectar falhas no avião e agir quando acontecer; simular múltiplas situações para apoiar as decisões dos agentes; e controlar o avião automaticamente quando for necessário. Atualmente, tem-se um protótipo funcional. Os testes de verificação foram feitos em ciclos repetitivos desde o primeiro até a versão atual. Em cada ciclo foram feitas atualizações e evoluções no sistema e nas bases de conhecimento. Outra das principais contribuições deste trabalho é uma estrutura de ontologia que permite aos agentes interpretar a dinâmica da aeronave e detectar falhas.

Palavras-chave: Tutor inteligente, simulação, modelagem de veículos, bases de conhecimento, sistemas multiagentes.

ABSTRACT

On the civil and military aviation area, the low availability of qualified instructors in homologated schools, and the issue of impartiality on the judgment between instructor and student makes attractive the idea of developing an intelligent training system, which can assist or partially substitute the human instructor. This work aims to present a proposal of an intelligent virtual instructor system, to training with simulators, to supply the unavailability of human instructors and the need of improvement in judgment impartiality on the evaluation. To approach this objective, a virtual instructor system model was defined, commanded by virtual agents with its reasoning based on knowledgement, developed to allow the projection of virtual tutors for training with vehicles using simulators, which could be extended for machines and process operation training. The prototype was developed according to the system of virtual instruction. The first part of the prototype development was the knowledgement acquisition by bibliographical research and interviews with specialists involved with the plane operation. The second part was the representation of the knowledge in rules and ontologisms. The knowledge basis represents the qualitative and quantitative model (dynamic) of the plane and the environment (scenery), the knowledgement that involves the plane operation (Navigation, aeronautical regulation, meteorology, flight safety), knowledge that relates what and how to teach and evaluate, the knowledge of the system management (conduct protocols, communication and resolution of the agents conflict, emotional model, etc.), and the representation of the student model (a register of the learning and emotional student profile). It is four the main prototype functionalities: Train (teach, assist and evaluate) the pilot, detect failures on the plane and act when it happens, simulate multiple situations at the same time to support the agent's decisions, and control the plane automatically when necessary (e.g. if the student and the agents cannot handle an emergency, a dynamic model takes on the automatic plane control). Currently there is a functional prototype. The verification tests were performed in repetitive circles since the first prototype (beginning of the project) until the present version, which cycle actualizations and evolutions were made on the system and on the knowledge basis.

Keywords: Intelligent Tutoring, simulation, vehicle modeling, knowledge bases, multi agent systems.

LISTA DE FIGURAS

LISTA DE QUADROS

LISTA DE ABREVIATURAS E SIGLAS

A&A - Agents and Artifacs

ACT - Adaptive Control of Thought

ACT-r - Adaptive Control of Thought—Rational

AIS-IFT - Adaptative Instructional Sysytem - Intelligent Flight Trainers

AFA - Academia da Força Aérea - Brasil

ANAC - Agência Nacional de Aviação Civil - Brasil

ARI - Army Research Institute

IFT - Intelligent Flight Trainers

ITS - Intelligent Tutoring System

FAA - Federal Aviation Administration - EUA

FIPA - Foundation for Intelligent Physical Agents

OCC - Ortony, Clore and Collins Model

OWL - Web Ontology Language

SL - Semantic Language

TCP/IP - Transmission Control Protocol/Internet Protocol

UDP - User Datagram Protocol

W3C - World Wide Web Consortium

XML - EXtensible Markup Language

LISTA DE SÍMBOLOS

a_a Coeficiente de servo aileron (1/seg).

a_e Coeficiente de servo elevador (1/seg).

a_f Coeficiente de servo flap (1/seg).

a_L Coeficiente de servo leme (1/seg).

a_a Coeficiente de servo spoiler (1/seg).

a_t Coeficiente de servo Throttle (1/seg).

b Envergadura (ft)

b_a Coeficiente de altímetro (1/seg).

b_v Coeficiente de velocímetro (1/seg).

$b_{g\theta}$ Coeficiente de giroscópio de arfagem (1/seg).

$b_{g\varphi}$ Coeficiente de giroscópio de rolamento (1/seg).

$b_{g\psi}$ Coeficiente de giroscópio de guinada (1/seg).

b_α Coeficiente de sensor de ângulo de ataque (1/seg).

b_β Coeficiente de sensor de derrapagem (1/seg).

b_G Coeficiente de acelerômetro (1/seg).

$\bar{c}$ Corda média (ft)

C_D Coeficiente de arrasto total.

C_{Do} Coeficiente de arrasto em estado estacionário.

$C_{D\alpha}$ Variação no arrasto devido à ângulo de ataque (1/rad).

C_{Dih} Variação do arrasto devido à variação do ângulo de deflexão do leme (1/rad)

$C_{D\delta e}$ Variação do arrasto devido à variação do ângulo de deflexão do elevador (1/rad)

C_L Coeficiente de sustentação total.

C_{Lo} Coeficiente de sustentação em estado estacionário.

$C_{L\alpha}$ Variação na sustentação devido à variação do ângulo de ataque (1/rad).

C_{Lih} Variação da sustentação devido à variação do ângulo de deflexão do leme (1/rad)

$C_{L\delta e}$ Variação da sustentação devido à variação do ângulo de deflexão do elevador (1/rad)

C_l Coeficiente de rolamento total.

C_{lo} Coeficiente de rolamento em estado estacionário.

$C_{l\beta}$ Variação no momento devido à variação do ângulo de derrapagem (1/rad)

$C_{l\delta r}$ Variação no momento devido à variação do ângulo de deflexão do leme (1/rad)

$C_{l\delta e}$ Variação no momento devido à variação do ângulo de deflexão do elevador (1/rad)

C_M Coeficiente de arfagem total.

C_{Mo} Coeficiente de arfagem em estado estacionário.

$C_{M\alpha}$ Variação na arfagem devido à variação do ângulo de ataque (1/rad).

C_{Mih} Variação na arfagem devido à variação do ângulo de deflexão do leme (1/rad)

$C_{M\delta e}$ Variação na arfagem devido à variação do ângulo de deflexão do elevador (1/rad)

C_n Coeficiente de guinada total.

C_{no} Coeficiente de guinada em estado estacionário.

$C_{n\beta}$ Variação no momento devido à variação do ângulo de derrapagem (1/rad)

$C_{n\delta a}$ Variação no momento devido à variação do ângulo de deflexão do ailerão (1/rad)

$C_{n\delta r}$ Variação no momento devido à variação do ângulo de deflexão do leme (1/rad)

C_Y Coeficiente da força aerodinâmica lateral total.

C_{Yo} Coeficiente da força aerodinâmica lateral em estado estacionário.

$C_{Y\beta}$ Variação da força aerodinâmica lateral devido à variação do ângulo de derrapagem (1/rad).

$C_{Y\delta a}$ Variação do arrasto devido à variação do ângulo de deflexão do ailerão (1/rad)

$C_{Y\delta r}$ Variação do arrasto devido à variação do ângulo de deflexão do leme (1/rad)

F_A Vetor de força aerodinâmica (lbs)

F_{AY} Força aerodinâmica lateral (lbs)

F_T Vetor de força de tração (lbs)

F_{TX}, F_{TY}, F_{TZ}

Forças de tração nos respectivos eixos x,y,z (lbs)

g Vetor aceleração gravitacional (ft/s2)

gx Componente X do vetor g (ft/s2)

gy Componente Y do vetor g (ft/s2)

gz Componente Z do vetor g (ft/s2)

ih Deflexão do leme (rad)

Ixx,Iyy,Izz

Momentos de inércia (slug-ft2)

Ixy,Iyz,Ixz

Produtos de inércia (slug-ft2)

Iyx,Izy,Izx

Produtos de inércia (slug-ft2)

Ky Ganho de manche

Kr Ganho de pedais

Kt Ganho de throttle

Kf Ganho de alavanca de flapes

Ks Ganho de alavanca de spoilers

L_{Ai} Momento de rolamento aerodinâico (ft-lbs)

L_{Ti} Momento de rolamento de tração (ft-lbs)

L_T,M_T,N_T

Momento de rolamento, momento de arfagem e momento de guinada (ft-lbs)

m Massa (slugs)

M_A Vetor momento aerodinâmico (ft-lbs)

M_T Vetor momento de tração (ft-lbs)

M_{Aj} Momento de arfagem aerodinâmico (ft-lbs)

M_{Tj} Momento de arfagem de tração (ft-lbs)

N_{Ak} Momento de guinada aerodinâmica (ft-lbs)

N_{Tk} Momento de guinada de tração (ft-lbs)

P_i Taxa de rolamento. Componente do eixo X do vetor ω (rad/sec)

$\bar{q}$ Pressão dinâmica (lbs/ft2)

Q_j Taxa de arfagem. Componente do eixo Y do vetor ω (rad/sec)

R_K Taxa de guinada. Componente do eixo Z do vetor ω (rad/sec)

r' Vetor posição ao sistema X'Y'Z'

S Superfície aerodinâmica efetiva (ft2)

Ui Componente do eixo X do vetor Vp (Knots)

Vj Componente do eixo X do vetor Vp (Knots)

Vp Vetor velocidade linear (Knots)

ω Vetor velocidade angular (rad/sec)

Wj Componente do eixo X do vetor Vp (Knots)

XYZ Sistema de coordenadas de eixos ortogonais.

X'Y'Z' Sistema Inercial de eixos.

$X_sY_sZ_s$ Sistema de coordenada de estabilidade.

Símbolos Gregos

α Ângulo de ataque alpha (radianos)

β Ângulo de derrapagem ou Sideslip (radianos)

δe Deflexão do elevador (radianos)

δa Deflexão do ailerão (radianos)

δf Deflexão dos flappes (radianos)

δr Deflexão do leme (radianos)

δs Deflexão dos spoilers (radianos)

δt Deflexão do throttle (radianos)

ρ_A Densidade de massa (slugs/ft3)

θ, ψ, φ Ângulos de Euler; arfagem, guinada e rolamento (radianos)

ϒ Ângulo de trajetória (radianos)

SUMÁRIO

1 INTRODUÇÃO

Na aviação civil, a natureza da operação das aeronaves é altamente arriscada. Por isso é de vital importância o uso de simuladores para o treinamento de pilotos, além de o custo ser inferior ao uso de uma aeronave real. Tal cenário é bastante comum em uma grande diversidade de situações que envolvem a aprendizagem e a capacitação de pessoas em tarefas e atividades de alto risco ou de alta complexidade. Por essa razão, o treinamento por simulação é usualmente utilizado, a fim de reduzir riscos de acidentes na operação das máquinas, equipamentos ou veículos durante a aprendizagem.

Os simuladores, de um modo geral, são dispositivos ou sistemas virtuais utilizados para o apoio à aprendizagem, instrução ou treinamento de operação, em dispositivos ou equipamentos reais, criando ambientes artificiais que mimetizam, da forma mais próxima possível, os objetos de estudo. Neste trabalho é abordado o caso dos simuladores de voo, cujo propósito é simular o comportamento de uma aeronave envolvendo um baixo nível de abstração e um alto nível de envolvimento humano.

A essência de um simulador de voo é a criação de um modelo dinâmico do comportamento de uma aeronave, de modo a permitir que o usuário humano interaja com o simulador como parte da simulação de uma situação real das operações que envolvem a pilotagem do avião. Suas principais vantagens são (ROLFE, 1986) e (ALLBERTON, 2010):

a) A redução do custo de formação e treinamento de pessoal: o preço de aquisição de um simulador de voo varia de 30 a 65% do preço da aeronave e o custo de operação gira em torno de 8% do custo de operação da aeronave real;
b) A redução do tempo de formação e treinamento de pessoal: o treinamento pode ser centrado em uma manobra ou procedimento específico, não sendo sempre necessário ter que se repetir todo o voo;
c) Maior segurança: As situações potencialmente perigosas podem ser experimentadas sem risco de vida ou de perda de equipamento. Antes da adoção dos simuladores para esse tipo de treinamento, em certos casos, mais acidentes aconteciam durante os treinamentos de emergências do que em suas ocorrências reais. Um desses casos foi o da prática de pilotagem assimétrica após a falha simulada de um dos motores, o que resultou na perda da aeronave;

d) Oportunidade: para um treinamento em voo é preciso que a aeronave e o espaço aéreo estejam disponíveis, além de ser necessária a colaboração das condições atmosféricas.

O grande problema do treinamento com simuladores é a baixa disponibilidade de instrutores qualificados em escolas homologadas. Desse modo, é atrativa a ideia de desenvolver um sistema de treinamento inteligente que possa auxiliar ou substituir parcialmente o instrutor humano. Toda a capacitação de pilotos com simuladores exige pelo menos um mentor (ANAC, 2010), visto que não existe algum software suficientemente capaz de substituir ou assumir totalmente o papel de um tutor no treinamento.

O treinamento virtual totalmente instruído por software não é utilizado num nível profissional, já que os disponíveis são pouco flexíveis, muito previsíveis e repetitivos, geralmente são baseados numa sequência pré-definida de instruções e atividades. Isto encoraja a troca de informações entre alunos, podendo comprometer o sigilo e influir negativamente no resultado de aprendizagem (CRANE, 2006) e (DAHLSTORM, 2006).

Neste trabalho é apresentada uma proposta de instrutor virtual inteligente para treinamento de pilotos comerciais usando simuladores para aviões modelo Boeing 737-800. Foi testada e, parcialmente, validada por um piloto através de um protótipo simplificado. Esta tese de doutorado dá seguimento à linha de pesquisa do laboratório de hardware (LHW) da UFSC, cujos trabalhos anteriores, e em andamento, permitem antecipar limitações, não somente nos simuladores, mas também nos sistemas de treinamento e supervisão.

1.1 MOTIVAÇÃO

Na aviação civil, as rigorosas exigências impostas nos regulamentos (ICAO no mundo e ANAC no Brasil) para treinamento com simuladores (instalações e equipamentos aprovados pelos organismos reguladores, exigências para a qualificação dos instrutores, etc.) podem trazer algumas desvantagens:

a) Custos: Demanda onerosa por instalações especializadas, instrutores qualificados, que se refletem no preço que o aluno deve pagar por cada hora de voo no simulador;

b) Disponibilidade: A baixa disponibilidade de instrutores qualificados limita a proliferação de centros de treinamento, forçando o deslocamento dos alunos. Além disso, o treinamento básico exige um instrutor por aluno, o que não torna viável o treinamento coletivo e a distância de muitos pilotos;
c) Imparcialidade no julgamento de desempenho: Questões pessoais, como afinidade ou fatores externos, podem influir no resultado das avaliações, de forma que um aluno não devidamente qualificado possa vir a ser aprovado em função do relacionamento com seu instrutor. Um sistema virtual inteligente poderia ser mais independente em relação à avaliação dos alunos;

Nas primeiras fases de treinamento de pilotos comerciais a instrução é normalmente feita com simuladores de voo rodando em computadores pessoais (possivelmente, com algum equipamento periférico adicional, tais como manches, pedais, painéis, etc.) e guiados por instrutores humanos. A grande disponibilidade de equipamento (devido ao baixo custo) existente nas escolas de aviação, na maioria das cidades, constitui uma vantagem para os alunos, já que não precisam se deslocar a grandes distâncias. No entanto, a disponibilidade de instrutores humanos ainda é, relativamente, limitada.

Em fases avançadas de treinamento de pilotos a instrução é feita com simuladores mecânicos de cabine móvel, de alto custo, existindo ainda poucos em centros de treinamento especializados e em algumas escolas de aviação homologadas. Neste cenário há maior disponibilidade de instrutores humanos em relação a pouca quantidade destes simuladores, mas os alunos devem se deslocar aos locais de treinamento.

A ideia de projetar um sistema de treinamento virtual inteligente é atrativa, pois haveria maior disponibilidade de instrutores virtuais locais e remotos. Seria mais atraente, principalmente, nas primeiras fases de treinamento dos pilotos comerciais, cujo problema da pouca disponibilidade de instrutores humanos é maior. Um instrutor humano não pode ser totalmente substituído por um sistema inteligente, pois as técnicas existentes de inteligência artificial não abrangem toda a capacidade da perícia humana. Entretanto, grande parte das tarefas dos instrutores pode ser feita por sistemas de instrução artificial, deixando às pessoas a responsabilidade de tomar as decisões mais complexas.

1.2 OBJETIVOS

A pesquisa foi delimitada com o objetivo de melhorar os sistemas atuais de treinamento virtual de pilotos abordando algumas de suas limitações que são:

a) A inteligência artificial (agentes) não consegue entender a dinâmica do avião, pois não reconhece o significado da variação das variáveis.
b) Os sistemas não conseguem pilotar a aeronave em forma autônoma sem o apoio do piloto automático da aeronave.
c) Os sistemas não conseguem detectar falhas na aeronave nem entender o significado do contexto dos eventos.

A hipótese deste trabalho formula que é possível abordar as limitações dos IFT existentes apresentados anteriormente desenvolvendo uma estrutura adequada de bases de conhecimento para que os agentes consigam interpretar a dinâmica de veículos permitindo eles pilotar em forma autônoma, entender a dinâmica do avião e detectar falhas.

1.2.1 Objetivo Geral

O objetivo deste trabalho é propor e testar um protótipo de sistema instrutor virtual inteligente para treinamento com simuladores, que desempenha algumas atividades normalmente realizadas por instrutores humanos.

1.2.2 Objetivos Específicos

Para atender o objetivo principal foram estabelecidos os seguintes objetivos específicos:

a) Propor um sistema de instrução virtual para treinamento com simuladores de voo (para o avião comercial Boeing 737-800) a fim realizar as seguintes tarefas: pilotar o avião em forma autônoma, ensinar ao aluno o conteúdo de cada sessão de treinamento, avaliar o aluno no transcurso da sessão, conhecer a dinâmica do avião e detectar falhas no avião;
b) Desenvolver um *framework* ou biblioteca de classes para a implementação do protótipo;
c) Desenvolver as bases de conhecimento do instrutor virtual;

d) Desenvolver um protótipo simplificado para testar, verificar e validar o sistema de instrução proposto.

1.3 METODOLOGIA

O desenvolvimento deste trabalho foi feito em quatro grandes etapas:

a) Pesquisa bibliográfica: A revisão bibliográfica delimitou-se em sistemas de tutores inteligentes e modelos aplicados ao treinamento de pilotos;
b) Proposta do sistema de treinamento: Esta etapa foi iniciada com um levantamento de especificações de funcionalidade do protótipo, chegando à concepção da solução. Foi proposta uma estrutura de ontologia de diferentes níveis funcionais para atender as limitações expostas anteriormente. Finalmente, foi desenvolvido um *framework* (bibliotecas de classes) que permite estruturar o protótipo para testar a proposta;
c) Prototipagem: Nesta etapa foi feita a aquisição de conhecimentos iniciais para representação em regras e ontologias por meio de: entrevista com pilotos e especialistas, capacitação por meio de cursos online, capacitação num curso presencial de piloto privado, entre outras formas de aquisição de conhecimento. De forma paralela, foi desenvolvido o protótipo de sistema virtual de treinamento para pilotos de aeronaves Boeing 737-800. Este protótipo roda num PC com os seguintes periféricos: Manche de avião, unidade de manetes (potência e flaps) e pedais de leme. As bases de conhecimento iniciais foram integradas ao protótipo. Finalmente, foram feitas várias sessões de voo manual para o sistema observar e apreender as manobras.
d) Testes, verificação e validação do sistema: Nesta etapa foram testadas as funcionalidades do sistema (ignorando a inteligência artificial). Em seguida, foram verificadas as bases de conhecimento para detectar erros de sintaxe e semântica e, finalmente, foi feita a validação da inteligência artificial do sistema. Esta validação foi realizada por um piloto comercial que operou o simulador e observou o desempenho do sistema em várias fases de voo.

1.4 TRABALHOS RELACIONADOS

Não foram encontrados trabalhos correlatos que abordem todos os objetivos específicos desta tese, porém existem pesquisas que serviram de base para o desenvolvimento do sistema proposto.

Na área de treinamento virtual com simuladores para veículos destacam-se cinco trabalhos. O primeiro trabalho trata-se de um sistema IFT (*Intelligent flight Trainer*) baseado em raciocínio com regras, usando a ferramenta CLIPS (CLIPS, 1995), desenvolvido por Mulgund, Asdigha e outros, para treinamento de pilotos de helicópteros militares (MULGUND et.al, 1995). O Segundo trata-se do sistema AIS-IFT (*Aideid Interface System- Intelligent flight Trainer*) desenvolvido por Remolina e Howse, apresentando uma evolução aos sistemas IFT ao introduzir o uso de multiagentes, bases de conhecimento procedurais editáveis e a representação do aluno numa rede Bayesiana (considerando aspectos pedagógicos e emocionais dele).

O instrutor virtual também foi usado para treinamento de pilotos de helicópteros militares. O AIS-IFT é um projeto do exército dos Estados Unidos da América em conjunto com a ARI (*Army Research Institute*). Existem duas publicações referentes a este projeto, (LULWIG, 2002), (REMOLINA 2004). O terceiro trata-se do sistema ASIMIL (*Aero user-friendly SIMulation-based distance Learning*) desenvolvido na França e projetado com um sistema de raciocínio baseado em casos (CBR) para treinamento de voos por instrumentos para pilotos de aviões comerciais (ASIMIL, 2001). Outros trabalhos relacionados, na maioria dos casos, desenvolvem tutores virtuais, porém sem uso de técnicas de inteligência artificial, deixando-os sequenciais e previsíveis.

Na área de tutores virtuais para ensino existem diversos trabalhos interessantes que utilizam agentes pedagógicos baseados na arquitetura de sistemas tutores inteligentes (STI), especialmente, aqueles que se adaptam ao aluno, como por exemplo, no trabalho "Tutor inteligente adaptável conforme as preferências do aprendiz" (Pozzebon, E. 2003), e aqueles que usam modelos afetivos, como por exemplo, no livro "*Agent-Based Tutoring Systems by Cognitive and Affective Modeling*", (Viccari R.; Jackes P., 2008), permitindo que o tutor possa expressar emoções próprias e perceber as emoções do aluno. Diferentes técnicas de aprendizado têm sido usadas nos STI, entre as mais vanguardistas se destaca um sistema de ensino virtual baseado em resolução de problemas (*Problem-Based Learning - PBL*) (MABEL, 2012).

Na área das emoções em agentes, alguns sistemas tutores inteligentes (STI) que consideram modelagem emocional, estão baseados no trabalho de Ortony, Clore e Collins, o

modelo OCC [OCC, 1988]. Um desses trabalhos aborda um modelo evoluído de OCC: usado para tutores inteligentes, com base na mineração de dados e estados afetivos dos estudantes (BOFF, 2012).

1.5 ORGANIZAÇÃO DO TEXTO

É muito difícil sintetizar a grande quantidade de informação envolvida neste projeto em um livro só (procedimentos, entrevistas, etapas do projeto, marco teórico, etc.), pois abrange uma quantidade de tópicos e disciplinas, tais como: inteligência artificial, pedagogia, psicologia, aviação, sistemas lineares, sistemas de controle, entre outros. Também não foi possível incluir detalhes do acervo de documentação que foi gerado no desenvolvimento das etapas do trabalho, tais como: o projeto informacional (determinação das especificações), projeto conceitual (determinação das concepções do protótipo), aquisição e representação de conhecimento, relatórios de verificação, registro de atualizações das bases de conhecimento, entre outros. Não foi considerado necessário mostrar o desenvolvimento das interfaces entre o sistema tutor virtual e o simulador, porque é irrelevante para o objetivo do projeto.

Pelos motivos mencionados anteriormente o foco deste livro de tese foi apresentar a proposta do sistema de instrução virtual, o protótipo desenvolvido, e as considerações mais importantes na aquisição e representação de conhecimento, verificação e validação do sistema.

O Capítulo 1 apresenta a introdução, motivação e objetivo do trabalho, metodologia de desenvolvimento e trabalhos relacionados.

O Capítulo 2 apresenta um resumo do referencial teórico usado para o desenvolvimento da proposta.

O Capítulo 3 apresenta a proposta do sistema de treinamento baseado nos objetivos e especificações do sistema.

O Capítulo 4 apresenta a prototipagem do sistema de treinamento virtual, abordando a aquisição e representação de conhecimentos, estruturação do sistema e testes iniciais.

O Capítulo 5 apresenta como foram feitos os testes de verificação e validação do sistema, mostrando alguns casos de uso e outros representativos dos testes. A grande quantidade de documentação gerada nesta fase impossibilita a sintetização em um capítulo, por isso o foco deste é mostrar os métodos de testes e os resultados mais importantes.

O Capítulo 6 apresenta as conclusões do trabalho. São abordados a análise do atendimento das especificações do projeto, são discutidos os resultados dos testes de verificação e validação, são mostradas as contribuições do trabalho e indicados os possíveis trabalhos futuros.

2 MARCO TEÓRICO

Neste capítulo é apresentado um resumo da revisão bibliográfica focada em tutores virtuais inteligentes. Os sistemas IST (*Intelligent Tutor System*) ou sistemas de tutoria inteligentes foram desenvolvidos com o propósito de acompanhar o aprendizado do aluno a todo o momento, adaptando a estratégia de ensino e o plano de estudos em função do desempenho e evolução do aprendizado do aluno. Os STI são sistemas baseados em conhecimento (um dos métodos fortes de resolução de problemas na área da inteligência artificial).

Na área da inteligência artificial existem os métodos de resolução de problemas chamados fracos e fortes. Os métodos fracos são aqueles que resolvem um subproblema bem específico, mas que, por si só, não resolve um problema completo. Por exemplo, uma rede neural pode, por exemplo, reconhecer um pedestre na rua, mas não tem capacidade de raciocínio para que um veículo autônomo possa decidir se vai desviar o pedestre ou parar. Neste caso, a rede neural é um método fraco de resolução de problemas. Um método forte seria, por exemplo, um sistema especialista que utilizasse a rede neural para perceber padrões visuais e pudesse raciocinar em função a outros fatos externos e internos.

Os sistemas baseados em conhecimento são métodos fortes de resolução de problemas, trabalham com bases de conhecimento, motores de inferência (de regras, modelos e ontologias), unidades de resolução de conflitos e diversas interfaces. A maioria dos sistemas baseado em problemas operam com agentes (entidades virtuais autônomas) que assumem diversas tarefas locais ou distribuídas.

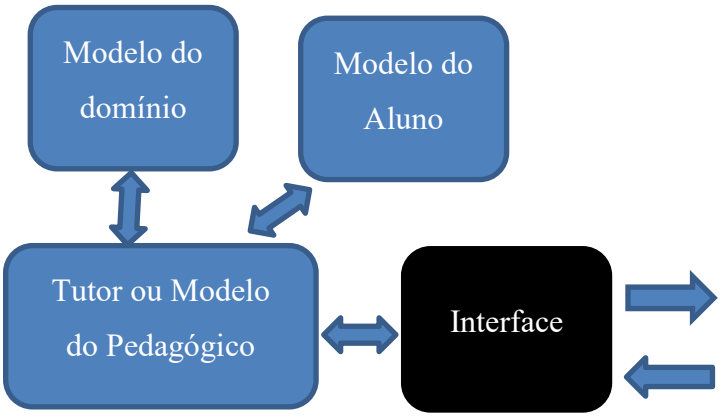
Na área da educação sempre existiu a necessidade de otimizar o aprendizado do aluno, tentando, na medida do possível, ser o mais verossímil, um nível que o tutor consiga traçar um perfil do aluno considerando quatro aspectos (PARK, 2009): desempenho, evolução do aprendizado, preferências de aprendizado (por exemplo se aprende melhor com slides ou com conteúdo no quadro) e perfil emocional. Com este perfil, o tutor pode acompanhar o aprendizado do aluno, podendo mudar o plano de ensino e a estratégia em todo momento. Muitos professores não contam com a capacitação ou com o tempo de dedicação para atender estas necessidades. A partir dos anos 1970, começaram a abordar este problema, criando os primeiros sistemas de tutoria inteligente, a partir das ideias do trabalho de Carbonell (CARBONELL, 1970).

2.1 TUTORES INTELIGENTES

Os sistemas STI foram idealizados por Sleerman e Brown no ano de 1982 (PARK, 2009). Estes sistemas são uma evolução dos sistemas de ensino assistidos por computador CAI (*Computer Aided Instruction*) que apresentavam apenas o conteúdo numa sequência finita e pré-definida. A concepção dos STI é baseada no controle do material de estudo em função das respostas dos alunos. Isto permite que os sistemas representem em um modelo a forma que o aluno resolve os problemas, a rapidez com que aprende, entre outros aspectos.

A arquitetura dos STI foi variando até a atualidade através de inúmeras pesquisas acadêmicas, porém todos se baseiam em uma estruturação comum, como mostrado na figura 1.

Figura 1 - Modelo base de STI



Fonte: Autor (2014).

O modelo do domínio são as bases de conhecimento do domínio a serem ensinadas, estas podem ser representadas em regras, em ontologias e modelos lógicos. O modelo do aluno é a representação que o tutor faz sobre o aprendizado dele, incluindo o perfil de desempenho, evolução de aprendizado, históricos de estudos, perfil emocional e perfil de estilos de aprendizado. O tutor, ou também chamado modelo do pedagógico, é o responsável de ensinar e avaliar o aluno, acompanhando a evolução do aprendizado e adaptando o estilo e conteúdo de ensino. Na maioria dos casos, o tutor é um agente virtual. A interface comunica o tutor com o ambiente de ensino do aluno (ensino local, ensino a distância EAD, ambientes virtuais de treinamento, etc.).

2.1.1 Modelo de domínio

O modelo de domínio é uma representação do conhecimento que o tutor deve ensinar e o aluno deve aprender. Normalmente, vem separado do material pedagógico apresentado ao aluno (imagens, atividades, slides, etc.) já que o domínio é o conhecimento sobre o material pedagógico a ser mostrado. O material pedagógico é armazenado na forma de objetos pedagógicos num container virtual (SILVEIRA et.al., 2005).

O domínio pode ser representado usando formalismos orientados a predicados, por meio de regras de produção e programação em lógica. Também, pode ser representado usando formalismos orientados a classes usando redes semânticas, quadros e ontologias.

A grande maioria dos sistemas STI pesquisados usam representação de domínio por meio de ontologias, já que facilita o reaproveitamento destes para consulta online em aplicações web-semânticas (W3C, 2014).

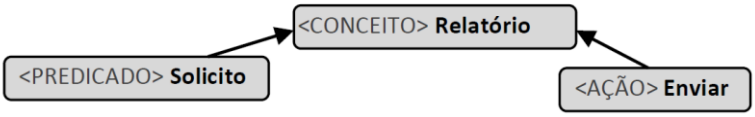
A ontologia é uma representação de conhecimento reusável e entendível por diversos sistemas. Define um domínio ou uma conceptualização acerca do conhecimento. Uma ontologia é organizada em hierarquias de conceitos (ou taxonomias). Os formatos de ontologia mais usados são: RDF, OWL e FIPA-SL. O formato RDF foi projetado para fornecer a interoperabilidade e semântica para meta-dados, de modo a facilitar busca por recursos na Web (GEROIMENKO, 2003). Foram propostos em fevereiro de 1999, pelo consórcio W3C. O formato OWL (*Ontology Web Language*) representa relações (também chamadas propriedades) entre dois elementos e estas relações são definidas como predicados. Os predicados relacionam dois tipos de conceitos: os conceitos sujeito e os conceitos objeto. Um sujeito é um elemento de um subconjunto domínio de conceitos e um objeto é um elemento de um subconjunto rango de conceitos. No OWL os conceitos são descritos como referências a recursos (IRI) das bases de conhecimento locais ou remotas. A figura 2 mostra um exemplo de relação de instância para representar a expressão <*gosta_De(JOÂO,MARIA*)>.

Figura 2 - Exemplo de relação de instância do formato OWL

Fonte: Autor (2014).

Às vezes, junto ao modelo de domínio de um STI vem uma base de conhecimentos que pode conter protocolos de comportamento do agente, por exemplo, para enviar um relatório a outro agente que solicitar (AIS-IFT, 2004). Nestes casos, a maioria dos sistemas utiliza o formato FIPA-SL. O formato FIPA-SL é uma especificação de linguagem de conteúdo entre agentes virtuais (o paradigma de agentes será explicado no corpo do capítulo). Assim, as ontologias neste formato definem protocolos de comunicação e as ações dos agentes. As ontologias são representadas com três elementos principais: o conceito, o predicado e a ação. O conceito é o objeto de interesse (tema de conversação nas frases); o predicado é a descrição do que está acontecendo com o conceito ou objeto de interesse; e a ação é o que o agente faz ao receber uma mensagem de outro agente, baseado do que acontece (predicado) com o objeto de interesse (conceito). Por exemplo, se o conceito for "relatório", na expressão <*solicito(relatório)*> o predicado é "solicito" e indica que o agente solicita a outro agente um relatório. Por outra parte, a expressão <*enviar(relatório*)> indica que o agente que recebe a mensagem deve enviar o relatório quando outro agente o solicitar (o conceito é "relatório" e a ação é "enviar"). A figura 3 mostra um exemplo de relação de ontologia de protocolo.

Figura 3 - Exemplo de instância de protocolo em uma ontologia de domínio



Fonte: Autor (2014).

2.1.2 Modelo de aluno

O modelo de aluno (em algumas publicações, também chamado modelo do aluno) é uma representação que o tutor vai construindo ao observar a evolução, desempenho, preferências e estados emocionais do aluno. Na maioria das vezes, o modelo de aluno é representado com ontologias e redes bayesianas.

As formas de representar o modelo de aluno são, na maioria dos casos, baseados em três modelos mencionados na dissertação de Pozzebon (POZZEBON, 2003):

a) Modelo diferencial: Neste tipo de modelo o conhecimento do aluno é considerado como um subconjunto do conhecimento representado no modelo de domínio. A forma do tutor avaliar o desempenho do aprendizado é comparando o conhecimento do aluno com o conhecimento do domínio;

b) Modelo Overlay: Neste tipo de modelo a representação do aluno deve evoluir até chegar a ser a mesma representação que do domínio;
c) Modelo BDI: O aluno é percebido pelo agente tutor como outro agente do sistema, com crenças, desejos e intenções.

2.1.3 Tutor ou modelo do pedagógico

O modelo Tutor é a unidade que contém as estratégias e planos de ensino. Na maioria dos casos, vistos nos trabalhos acadêmicos, um agente assume o papel do tutor. De fato, foram desenvolvidos sistemas STI com mais de um agente, por exemplo, no projeto *White Rabbit* desenvolvido pela universidade de Montreal, Canadá (THIBODEAU, 2000) e no sistema de treinamento de pilotos do exército dos EUA (AIS-IFT, 2004) foram usados diferentes agentes para diferentes tarefas: um para ensinar, outro para avaliar e outro para conciliar conflitos.

Existem duas linhas principais de ensino que os sistemas tutores usam (DAVIS, 1994). A primeira linha é a interacionista, onde o aprendizado é entendido como um processo de construção contínuo e recíproco do ser humano em relação ao meio onde está. Uma estratégia usada pelo tutor é interagir com o aluno, mas não para ensinar, senão para apenas interagir e acompanha-lo na exploração do ambiente virtual, na espera que o aluno peça ajuda. Outra estratégia usada pelo tutor é não ensinar, mas apenas mostrar o material pedagógico em função do avanço no aprendizado.

A segunda linha de ensino é a empirista, em que a experiência é a fonte de aprendizado, considerando o aluno como um ser adaptável, que se desenvolve em função do meio. Uma estratégia usada pelo tutor é reforço e punição, assim apresenta um comportamento no qual pode elogiar, questionar e exigir mais do aluno. Outra estratégia usada pelo tutor é o condicionamento, em que o tutor indica diferentes caminhos ou tarefas cada vez que o aluno tiver dificuldade.

As estratégias mais usadas pelos tutores em sistemas STI podem considerar as duas linhas de ensino, sendo estas mencionadas no livro de Beverly Park (PARK, 2009):

a) Treinamento ou Coaching: Normalmente, feito em ambientes de simulação: o tutor adapta as condições do treinamento em função do desempenho do aluno (assim, como no caso dos sistemas STI para treinamento de pilotos, também conhecidos como IFT, *instruction flight tutor*);

b) Socrático: esta estratégia busca apresentar material ao estudante, para que este identifique enganos e interpretações errôneas acerca do conteúdo;
c) Reativos: O tutor reage às perguntas do estudante adaptando o roteiro de ensino e focando em resolver as dúvidas do aluno;
d) *Troublemaker*: Esta estratégia consiste em: o tutor dá uma resposta errada a um problema e o aluno é forçado a reagir e propor uma solução correta;
e) Assistente: O tutor ao invés de ensinar, da assistência em tempo real ao aluno durante o desenvolvimento das atividades pedagógicas.

2.2 SISTEMAS MULTI AGENTES NOS STI

Os sistemas multiagentes são uma abordagem da inteligência artificial distribuída, (IAD) que oferecem uma solução cooperativa através de agentes virtuais distribuídos quando outros métodos não podem resolver o problema. Por exemplo, nas redes de computadores certos servidores contam com agentes distribuídos para observar possíveis ameaças de ataques de hackers, neste ambiente outro tipo de solução que não seja distribuída não seria tecnicamente viável.

Nos STI, os sistemas multiagentes oferecem soluções, quando as diversas tarefas devem ser feitas em forma autônoma e convergente. No caso em que as bases de conhecimento são grandes, estas podem se dividir em bases menores, organizadas em vários domínios, assim cada agente é especialista em um domínio diferente. Enquanto diferentes agentes executam suas respectivas tarefas no processo de ensino num STI, o tutor pode se dedicar, exclusivamente ao ensino, garantindo uma otimização da tarefa.

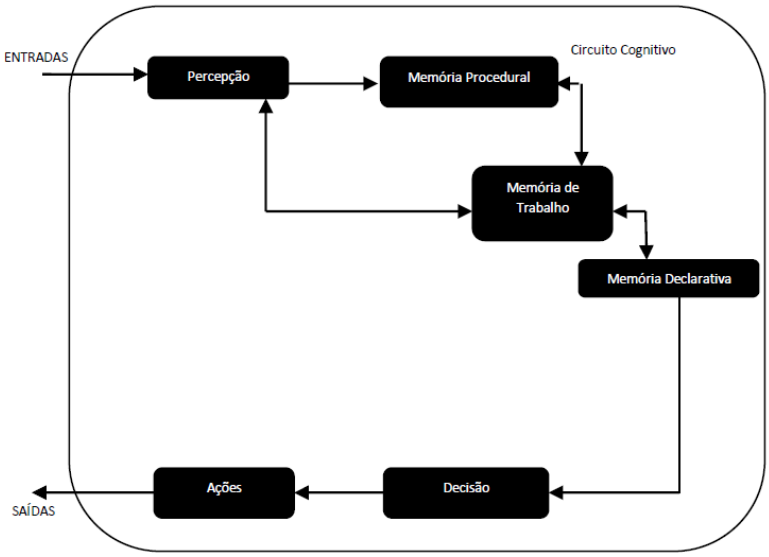
A interação dos agentes com os elementos do entorno ou ambiente é modelado no paradigma de agentes e artefatos A&A - *Agent and Artiffacts* (CARTAGO, 2010). Os artefatos são elementos passivos ou ativos do ambiente de trabalho, os quais podem ser manuseados e usados pelos agentes. Cada agente pode acessar remotamente qualquer artefato. Para o acesso, o agente deve se subscrever e logar (*login*). Os artefatos devem ser reusáveis de alto nível (primeiro nível), robustos e adaptáveis às condições do entorno de trabalho.

Na maioria dos casos dos STI modernos, os agentes são do tipo cognitivo ou deliberativo, ou seja, conhecem seu estado mental, apresentam comportamento orientado a objetivos, tomam iniciativas quando julgarem necessário, podem se comunicar com outros agentes por meio de mensagens em formato de linguagem que qualquer agente conectado online

no mundo possa entender (p.ex. FIPA-SL), alguns deles mantêm estados emocionais e podem inferir certas emoções do aluno.

Um dos agentes mais completos é apresentado na tese “Integração de emoção e raciocínio em agentes inteligentes” (GRAÇA, 2006) mostrado na figura 4. Pode usar o modelo emocional OCC e atender todas as caraterísticas mencionadas no parágrafo anterior. A vantagem desta estrutura de agente é que pode usar diferentes bases de conhecimento que podem modificar as emoções, comportamentos, responsabilidades e domínios. Pode ser construído na plataforma JADE para desenvolvimento de sistemas multiagentes, software livre (JADE, 2014), por exemplo.

Figura 4 - Arquitetura de agente proposto



Fonte: Adaptado de Graça (2006).

2.2.1 FIPA E FIPA ACL

A FIPA (Foundation for Intelligent Physical Agents) é uma fundação sem fins lucrativos, direcionada à produção de padrões para a interoperabilidade de agentes heterogêneos e interativos, e sistemas baseados em agentes. FIPA é uma organização membro da IEEE (Institute of Electrical and Electronics Engineers). A sua missão básica é facilitar a interligação de agentes e sistemas multiagentes entre múltiplos fornecedores de ambientes.

A (FIPA-ACL) é uma linguagem, utilizada no padrão FIPA de comunicação entre agentes, baseada em ações de fala. Sua especificação consiste em um conjunto de tipos de

mensagens e descrições dos efeitos das mensagens sobre os agentes que as enviam e sobre os que a recebem. Possui uma semântica definida precisamente, com uma linguagem de descrição de semântica. De acordo com (MENESES 2001), o Knowledge Query and Manipulation Language (KQML) tem sido muito criticada por usar o termo performativo para se referir às primitivas de comunicação. Em FIPA-ACL, essas primitivas são chamadas de ações ou atos comunicativos (communicative acts).

2.2.2 Emoções nos agentes

A computação afetiva envolve as emoções dos agentes, para simular estados emocionais ou para detectar o estado emocional de outros agentes ou das pessoas.

As emoções são importantes no desempenho de tarefas e tomadas de decisões, pois reduzem o "leque" de opções e ações, sem o qual os humanos ficariam presos em uma infinita gama de possibilidades diante a mais simples escolha (WIDMARK, 2003).

Quando há interação direta e prolongada entre máquinas (computadores) e humanos, o fator emocional resulta de vital importância, já que se a pessoa não desenvolver afinidade com sistema, ela vai ficar entediada ou chateada.

Desde o ano 1976, surgiram várias linhas de investigação, tendo como objetivo principal a integração de aspectos emocionais em modelos e arquiteturas de agentes inteligentes. As motivações para esses trabalhos são diversas, desde o facilitar o desenvolvimento de personagens mais realísticas para animação, entretenimento ou interação com humanos, até aquilo que constitui o nosso objetivo central, que é a utilização de mecanismos emocionais para a implementação de agentes capazes de lidar com os problemas que resultam da ação em ambientes reais. Estão desenvolvidos três tipos de modelos de agentes com emoções:

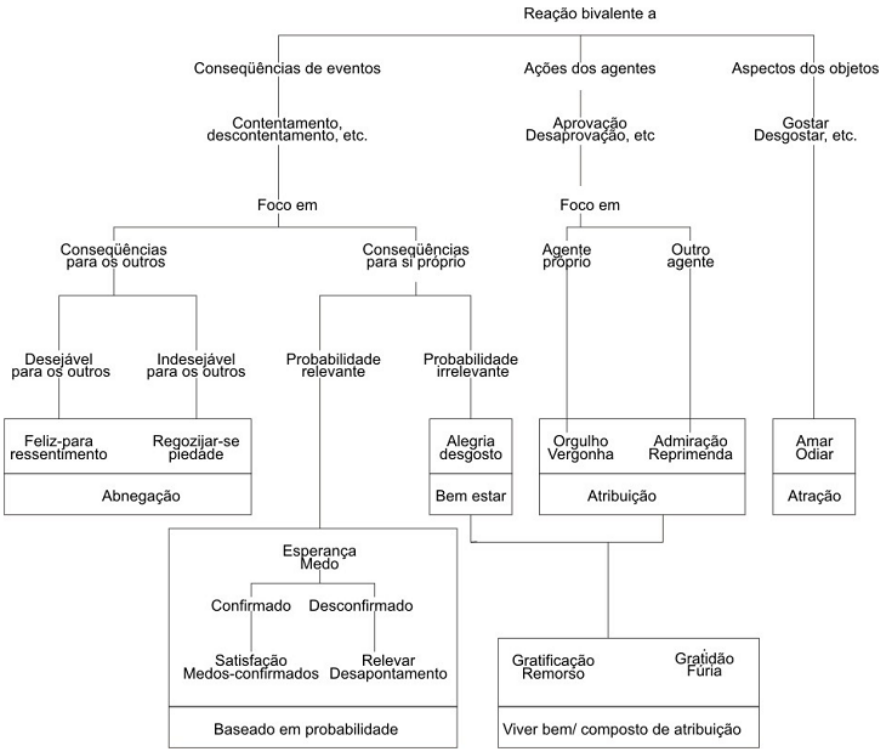
a) Modelos de base fisiológica: Estes relacionam os fenômenos emocionais com os fenômenos bioquímicos e homeostáticos que caracterizam os organismos biológicos (SCHULLIN, 2003);
b) Modelos de base cognitiva: Estes modelos estão relacionados ao comportamento da pessoa, a afinidade dela e os eventos que estão acontecendo, onde as emoções são inferidas (Exemplo, o modelo OCC);

c) O modelos Circumplex: Estes modelos descrevem as emoções em um arranjo de dimensões bipolares, e tem sido pesquisados desde 1941 por Schorberg (REMINGTON, 2000).

2.2.2.1 Modelo OCC

O modelo OCC é de base cognitiva e considera três fatores emocionais que são: a desejabilidade (emoções baseadas em eventos); a laudabilidade (emoções baseadas em atribuição); e a atratividade (emoções baseadas em objetos ou ideias abstratas), como mostrado na figura 5. A desejabilidade é avaliada em função dos objetivos e a interpretação dos eventos. A laudabilidade é avaliada em função das ações executadas pelos agentes ou usuários humanos baseado em fatores culturais, crenças, moral, etc. A atratividade é avaliada em função ao sentimento ou atitudes do agente quando vê ou pensa em objetos ou ideias.

Figura 5 - Modelo OCC



Fonte: Traduzido do (OCC, 1988)

Considerar a função desejabilidade $D(p,e,t)$ de um evento (e) para uma pessoa (p) num instante (t). Esta função possui valência positiva para eventos desejáveis, e valência negativa

para eventos indesejáveis. Considerar também a função intensidade global $I_g(p,e,t)$, a função de estado potencial de felicidade $P_j(p,e,t)$ e a função de disparo $T_j(p,t)$.

O estado emocional do próprio do agente ou de outro agente (ou pessoa) pode ser inferida a partir do algoritmo mostrado no quadro 1.

Quadro 1 – Algoritmo para inferência de emoções do modelo OCC

IF $D(p,e,t) > 0$
THEN set $P_j(p,e,t) = f_j(D(p,e,t), I_g(p,e,t))$
IF $P_j(p,e,t) > T_j(p,t)$
THEN set $I_j(p,e,t) = P_j(p,e,t) - T_j(p,t)$
ELSE set $I_i(p,e,t) = 0$

Fonte: (OCC, 1988, p.182)

2.3 INSTRUTORES VIRTUAIS INTELIGENTES DE VOO (INTELLIGENT FLIGHT TUTORS, IFT)

Desde o início da aviação os simuladores de voo vêm sendo empregados no treinamento de pilotos e tripulações completas. A sessão de treinamento tem sempre precisado da presença de um instrutor acompanhando o aluno. Até hoje, não é possível substituir totalmente as tarefas de um instrutor humano com os instrutores virtuais inteligentes existentes.

Os primeiros sistemas inteligentes para treinamento de pilotos foram desenvolvidos pela ARI (*Army Research Institute*) do exército dos EUA. O primeiro sistema bem-sucedido de treinamento de pilotos de aviões de combate foi desenvolvido por J. Regian (REGIAN, 1989), projetado para instruir procedimentos por instrumentos. Os dois trabalhos desenvolvidos mais representativos foram instrutores virtuais inteligentes para helicópteros militares. O primeiro trabalho trata-se de um sistema IFT baseado em raciocínio com regras usando a ferramenta CLIPS (CLIPS, 1995), desenvolvido por Mulgund, Asdigha e outros (MULGUND et.al., 1995). O segundo trabalho trata-se do desenvolvimento do sistema AIS-IFT, apresentando uma evolução aos sistemas IFT ao introduzir o uso de multiagentes, bases de conhecimento procedurais editáveis e a representação do aluno numa rede bayesiana (considerando aspectos pedagógicos e emocionais dele). Existem duas publicações referentes a este projeto, (LULWIG, 2002), (REMOLINA 2004).

Atualmente, a empresa Stottler Henke é proprietária do sistema AIS-IFT e tem disponível um framework de desenvolvimento multi-propósito para uso acadêmico e comercial, em uma versão de demonstração e uma versão paga (Stottler Henke, 2013).

2.4 SISTEMA IFT-AIS

O sistema AIS-IFT (*Adaptative Instructional Sysytem- Intelligent Flight Trainers*) é um instrutor virtual inteligente integrado com um simulador de voo de helicópteros (MULGAND, 1995). A operação do sistema centra-se num módulo chamado "Planejador Pedagógico", que coordena um grupo de agentes virtuais. Cada agente manuseia um roteiro de ações (modificável pelo projetista) para atender diferentes objetivos. O "Planejador Pedagógico" seleciona a estratégia de ensino baseado no perfil do aluno (confeccionado por um psicopedagogo após de uma entrevista com o aluno). Na planificação do treinamento são considerados fatores emocionais representados no modelo do aluno, mas o sistema não processa, nem simula, nem detecta emoções. A documentação original do projeto não especifica a função de cada agente nem a quantidade deles (ARI, 2004). O AIS-IFT é um dos primeiros sistemas de treinamento que contém um modelo dinâmico do aluno. O modelo é representado por uma rede bayesiana. O modelo especialista contém um conjunto de roteiros (ações e comportamentos) definidos para cada um dos objetivos, e um conjunto de critérios de avaliação para cada objetivo (critérios usados nos processos de diagnóstico). O módulo Facilitador regula o modelo da aeronave dependendo do nível de experiência do aluno. Para um aluno novo, o facilitador regula o modelo em uma configuração básica e simples de controlar (não realístico), e na medida em que o aluno adquire experiência, o facilitador configura o modelo para que seja mais complexo (mais realístico). O planejamento do treino é feito por um dos agentes (sem papel definido) em base nos objetivos do treino. O módulo Comunicador corresponde à interface do sistema. A comunicação do sistema IFT com o aluno é feita por meio de fones, usando uma interface de texto-fala. O treinamento do aluno é acompanhado em todo momento, sendo que os planos de ensino (roteiro de ações) podem ser mudados na hora dependendo do desempenho do aluno. A figura 6 mostra um treinamento no simulador com o sistema AIS-IFT.

Figura 6 - Sistema AIS - IFT

Fonte: U.S Army e Army Research Institute (2002)

Até o momento, o AIS-IFT é o sistema de treinamento de pilotos mais completo, ainda se os artigos que apresentam o sistema vão do ano 1995 até o ano 2004, o sistema é atualmente o mais usado para treinamento militar de aeronaves. As principais caraterísticas do sistema AIS-IFT são:

a) Contém um modelo especialista e um modelo de aluno;
b) Os agentes executam comportamentos de planejador, avaliador e tutor;
c) Mantém no modelo do aluno uma representação do perfil considerando aspectos emocionais dele;
d) Acompanha o treinamento do aluno em todo momento, mudando o plano de ensino quando for preciso (dependendo do desempenho do aluno);
e) O sistema tem uma representação do veículo.

3 PROPOSTA DE SISTEMA DE TREINAMENTO VIRTUAL

Entre os sistemas de treinamento virtual para pilotos estudados, destaca-se o AIS-IFT, por ser mais completo e ter melhores resultados. Porém, este sistema tem algumas limitações que serão indicadas mais para frente.

O desenvolvimento do sistema de treinamento virtual proposto neste trabalho foi focado em abordar as limitações do sistema AIS-IFT. Algumas soluções foram encontradas em sistemas de instrução virtual inteligente aplicado à educação, e outras soluções tiveram que ser criadas devido a não existência de trabalhos correlatos durante o período da pesquisa.

O sistema de treinamento virtual proposto foi batizado com o nome "JEMO" em apologia a uma canção proveniente de uma tribo da Oceania, cuja moral é que a tribo toda é responsável pela educação das crianças.

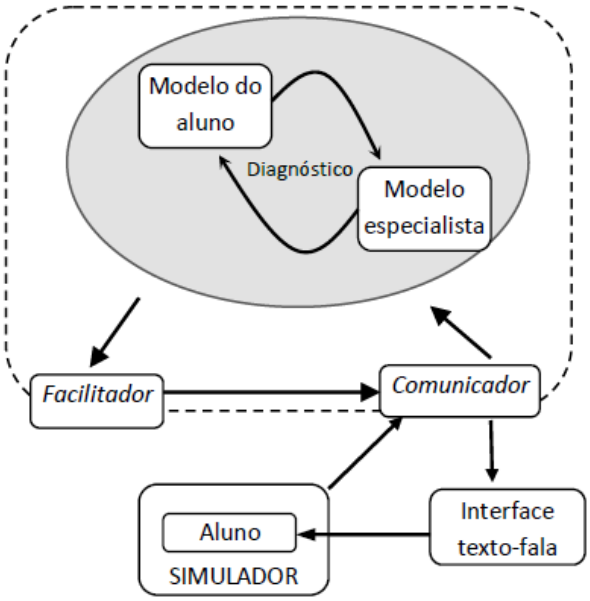
3.1 FUNDAMENTO DA PROPOSTA

O estudo começou identificando as caraterísticas essenciais do AIS-IFT para usá-las como base da proposta. Depois, foram identificadas as limitações do sistema e foram estudadas diferentes soluções possíveis.

O AIS-IFT, ou também conhecida como "*Adaptative Instructional Sysytem*", foi desenvolvida pela ARI (Army Research Institute) do exército dos Estados Unidos, (REMOLINA, 2004). O AIS-IFT é uma evolução do IFT (*Intelligent Flight Trainers*) (MULGAND, 1995), desenvolvido pela mesma instituição (ARI) e com o mesmo propósito de treinamento de pilotos de helicópteros. O sistema treinador está integrado com o simulador (permitindo compartilhar o modelo da aeronave entre o simulador e o treinador). O sistema conta com uma sociedade de multiagentes que não assume nenhum papel nos módulos do sistema, mas apenas executam ações e comportamentos parcialmente pré-definidos. Isso quer dizer, que qualquer um dos agentes poderia ter um comportamento de tutor, ou comportamento de avaliador. Os comportamentos dos agentes baseiam-se em máquinas de estado finitos.

A figura 7 mostra a estrutura do AIS-IFT. No círculo cinza é onde os agentes agem e executam os comportamentos (planos de ações).

Figura 7- Estrutura do AIS - IFT



Fonte: Traduzido do Remolina (2004)

O AIS-IFT é um dos primeiros sistemas de treinamento que incluem um modelo dinâmico do aluno (representado por uma rede bayesiana sob uma ontologia de termos de representação). O sistema seleciona a estratégia de ensino baseado no perfil e desempenho do aluno.

O módulo Facilitador regula o modelo da aeronave dependendo do nível de experiência do aluno. Para um aluno novo, o facilitador regula o modelo em uma configuração básica e simples de controlar (não realístico), e na medida em que o aluno adquire experiência, o facilitador configura o modelo para que seja mais complexo (mais realístico). O planejamento do treino é feito por um dos agentes (sem papel definido) em base nos objetivos do treino. O módulo Comunicador corresponde à interface do sistema. A comunicação do sistema IFT com o aluno é feita por meio de fones usando uma interface de texto-fala. O treinamento do aluno é acompanhado em todo momento, sendo que os planos de ensino (roteiro de ações) podem ser mudados na hora dependendo do desempenho do aluno.

Do AIS-IFT surgiu um framework (precisa comprar licença) de propósito geral chamado SimBionic [SIMBIONIC, 2014], baseado no manuseio de conhecimento procedural, que pode ser usado para simular comportamentos de agentes em vídeo-jogos, treinamento com simuladores e para processamento de eventos complexos (monitorar ambientes de negócios, controle de sistemas, etc.).Os aspectos mais fundamentais do sistema AIS-IFT são:

a) Contém um modelo especialista e um modelo de aluno;
b) Os agentes executam comportamentos de planejador, avaliador e tutor;
c) Mantém no modelo do aluno uma representação do perfil considerando aspectos emocionais dele;
d) Acompanha o treinamento do aluno em todo momento, mudando o plano de ensino quando for preciso (dependendo do desempenho do aluno);
e) O sistema tem uma representação do veículo.

As limitações encontradas no sistema AIS-IFT são:

a) O agente tutor carrega todas as bases de conhecimento (como acontece com muitos sistemas tutores inteligentes virtuais). Isto dificulta adicionar novas bases de conhecimentos devido ao aumento de carga de trabalho (mais ontologias e regras para processar) aumenta as chances de erros semânticos, erros de sintaxe e aumenta o tempo de processamento;
b) Todo o domínio (e subdomínios envolvidos no treinamento) é representado numa única base de conhecimento. Isto dificulta o acompanhamento de raciocínio (debug), dificulta muito a busca de erros semânticos e de sintaxe, e dificulta a adição de novos subdomínios, à medida que a base de conhecimento vai aumentando;
c) O sistema tem uma representação conceitual do veículo que pode ser processado pelos agentes, porém eles não conseguem interpretar a dinâmica do veículo processada no simulador (métodos numéricos). De fato, durante o período da pesquisa não foi encontrado nenhum trabalho acadêmico que implemente alguma forma de que os agentes entendam ou interpretem a dinâmica de um veículo;
d) O grau de dificuldade dos treinamentos é feito da seguinte forma: O sistema especialista manda um comando para o simulador modificar a dinâmica do veículo. Isto faz que a simulação se afaste da realidade nos níveis mais básicos;
e) O sistema de treinamento não tem todas as habilidades de um piloto, de fato não consegue pilotar a aeronave em todas as etapas, nem menos resolver situações de emergência. O sistema também não é capaz de detectar automaticamente falhas na aeronave, ainda forçando falhas no simulador.

Foi feito um projeto informal e um projeto conceitual usando a metodologia proposta no livro “Projeto Integrado de Produtos” (BACK et. al. 2008), que resultaram numa ideia inicial da proposta, as especificações para o sistema de treinamento virtual de pilotos e no conceito da solução. Os documentos são extensos e não é possível anexá-los, por isso são apresentados apenas os resultados dos estudos.

A ideia inicial da proposta de sistema de treinamento virtual para pilotos é um conjunto de caraterísticas e funcionalidades (levantadas a partir dos aspectos fundamentais do sistema AIS-IFT e a abordagem das limitações do mesmo) as quais são:

a) Um agente Tutor assume o modelo especialista, sendo o responsável de planificar o treinamento, de ensinar, e de acompanhar a aprendizagem do aluno;
b) Um grupo de agentes especialistas, em diferentes domínios específicos, assume o papel de diagnóstico (avaliação do desempenho do aluno). Cada agente carrega uma base de conhecimento diferente, referente ao seu domínio específico. A avaliação conjunta do desempenho do aluno é repassada ao agente Tutor;
c) O agente Tutor também tem a capacidade de pilotar a aeronave e sua habilidade é parecida com a de um humano. O agente tutor tem também a habilidade de entender a dinâmica da aeronave, permitindo detectar falhas ou prever estados do veículo;
d) O sistema tem um modelo do aluno. Neste modelo é representado, em forma de base de conhecimento, a evolução do aprendizado do aluno e o perfil emocional do aluno;
e) O agente Tutor usa o modelo do aluno para acompanhar o treinamento do piloto aprendiz a todo o momento, mudando o plano de ensino e a estratégia de ensino em função das preferências do aluno, o seu estilo de aprendizagem e seu estado emocional;
f) A dinâmica da aeronave no simulador não é modificada em nenhum momento, isto é importante para garantir realismo no treinamento virtual. Ao invés disso, o agente tutor pode dar assistência em diferentes momentos do treinamento, por exemplo, assumindo parcialmente a pilotagem da aeronave quando o aluno tiver dificuldade em algumas manobras.

As especificações da proposta resultante do projeto informacional foram a base do desenvolvimento da solução ou proposta do sistema de treinamento virtual. Estas especificações são:

a) O sistema deve conter uma representação do modelo do veículo, máquina ou processo (e do ambiente virtual envolvido), conter uma base de conhecimentos que representem os domínios envolvidos, um mecanismo de raciocínio contextual dos subsistemas, e um mecanismo de processamento numérico que represente os estados das variáveis envolvidas;
b) O sistema deve controlar o veículo, máquina ou processo quando for preciso, que seja para recuperar o avião nas situações de risco iminente (e assim que o perigo passar, devolver o controle ao aluno), ou para dar assistência a alunos novos em algumas manobras;
c) O sistema deve representar um modelo do aluno, considerando o seu histórico dos treinamentos, atitudes e comportamento, nível de experiência e preferências;
d) O sistema deve simular algumas emoções e representar algumas características humanas. Ao mesmo tempo, deve observar o comportamento do aluno e determinar dentro do possível o estado emocional dele;
e) O sistema deve planejar o treinamento de cada sessão, ensinar os conteúdos, acompanhar a aprendizagem do aluno e se adaptar ao ritmo dele. Cada vez que for necessário, o sistema pode mudar o plano de treinamento dependendo do desempenho dele;
f) Para poder acompanhar o aprendizado do aluno, o sistema deve avaliar o desempenho dele, considerando todos os domínios envolvidos no treinamento;
g) O sistema deve ser extensível e expansível, podendo suportar um desenvolvimento incremental;
h) O sistema deve atender, na medida possível, as normas, especificações e recomendações de organismos e fundações internacionais tais como. FIPA, OWL, W3C, etc;
i) O sistema deve ser independente na interoperabilidade com outros sistemas;

j) O sistema deve ter um alto grau de confiabilidade, considerando fatores como robustez (independência e autonomia, suportar cargas de trabalho, resolver conflitos, etc.) e de tolerante a falhas;
k) O sistema deve ser autônomo, sendo que o operador humano externo não pode participar nem influenciar nas decisões, planificação, ensino, nem avaliações do tutor virtual.

3.2 CONCEPÇÃO DA PROPOSTA

O projeto conceitual foi desenvolvido com o propósito de avaliar diferentes soluções que atendam as especificações obtidas no projeto informacional. Para conceituar a proposta, foram considerados dois aspectos; o loop interno (*inner loop*), ou como o aluno interage com cada atividade, e o loop externo (*outer loop*), que foca na sequência das atividades, ou seja, qual será a próxima atividade a partir das informações obtidas no momento (PELANEK, 2014). No loop interno, o sistema interage com o aluno proporcionando ajuda ou explicações em tempo real. No loop externo, o sistema planeja o treinamento em função do desempenho do aluno durante o treino, considerando diversos aspectos como: estado emocional e evolução do aprendizado.

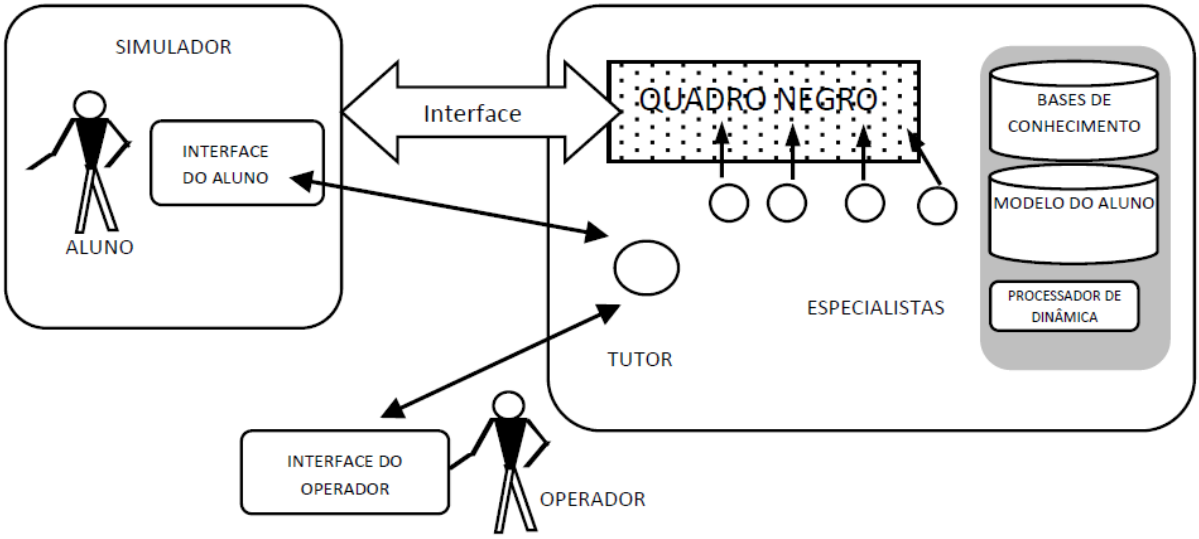
O resultado do projeto conceitual foi um sistema operado por agentes com raciocínio baseado em conhecimento. A sociedade multiagentes assume o papel de um grupo de especialistas com domínios de conhecimentos em diferentes tópicos. Um dos especialistas age como um tutor que entrega os conteúdos pedagógicos, o resto dos especialistas age como avaliadores. Os agentes se comunicam com mensagens e compartilham informação da interface por intermédio de um quadro negro. O agente Tutor se comunica com o aluno em uma interface montada no simulador.

O sistema é supervisionado por um operador externo (humano), que seleciona os objetivos do treinamento. O operador pode estar na mesma sala, ou em qualquer parte do mundo, já que a interface do operador se conecta com o sistema remotamente.

A interface entre o simulador e o sistema pode ser uma conexão TCP/IP ou UDP entre ambas as partes. As variáveis e comandos do simulador são dispostos em um quadro-negro acessível por qualquer um dos agentes. O agente tutor pode ler e escrever no quadro (p.ex. ao escrever no quadro negro, o tutor pode comandar um avião remotamente, como se fosse o

piloto). Os outros agentes especialistas apenas podem ler as informações do quadro. A figura 8 mostra a arquitetura resultante do projeto conceitual.

Figura 8 - Arquitetura do sistema de treinamento virtual proposto



Fonte: Autor (2004).

O sistema proposto pode ser descrito com quatro tipos de elementos: atores, bases de conhecimento, agentes e artefatos. O paradigma de sistemas agentes e artefatos A&A surgiu no ano 2003, definindo os artefatos dentro de um sistema multiagente como todos os objetos, módulos, interfaces ou abstrações que são utilizados pelos agentes, podendo ser real ou virtual, como por exemplo, braço robótico, labirintos, tabuleiros, cenários, quadro negro, agendas, simuladores, processos, etc. (RICCI, 2006).

Os atores são os protagonistas externos que interagem como o sistema, sendo eles o aluno e o operador (podendo ser humanos ou outros agentes externos). O operador é quem supervisiona externamente o treinamento e quem configura os objetivos e as condições do treinamento. O operador pode estar na mesma sala, ou em qualquer parte do mundo, já que a interface do operador se conecta com o sistema remotamente. O aluno é o usuário do sistema, ele opera o simulador, e interage com o sistema por meio de uma interface.

Os agentes assumem o papel de um grupo de especialistas com domínios de conhecimentos em diferentes tópicos. Um deles é o Tutor que entrega os conteúdos pedagógicos, o restante dos especialistas são os avaliadores. Os agentes se comunicam com mensagens e compartilham informação da interface por intermédio de um quadro negro. O

agente tutor se comunica com o aluno em uma interface montada no simulador e com o operador por meio de uma interface remota. Cada agente utiliza diferentes bases de conhecimento dependendo do domínio de cada um; o agente Tutor, em alguns casos, pode consultar qualquer base de conhecimento, dependendo das tarefas que esteja executando.

Os artefatos do sistema são a interface de aluno, interface do operador, interface do simulador, o quadro negro, o processador de dinâmica. O simulador é considerado um artefato externo ao sistema inteligente0. O núcleo da funcionalidade do sistema depende das bases de conhecimento, pois nelas está a funcionalidade efetiva pela qual foi projetado o sistema. As bases de conhecimento, que são processadas pelos agentes, estão organizadas em três categorias:

a) Gerenciamento, que representam as responsabilidades, protocolos, modelos emocionais e comportamentos dos agentes;
b) Treinamento, que representam o conhecimento envolvido com o veículo do simulador externo;
c) Modelo do aluno, que é uma base de conhecimento que representa os alunos em treinamento, contendo o perfil de desempenho, modelo emocional, perfil comportamental e histórico dos treinamentos deles.

No sistema opera um agente Tutor e vários agentes avaliadores. A quantidade de agentes avaliadores no sistema depende da quantidade de domínios de conhecimento que envolve o veículo, máquina ou processo representado no simulador externo. O agente Tutor é responsável das seguintes tarefas:

a) Gerenciar sistema completo, incluindo a organização dos agentes avaliadores. Resolver conflitos e emergências;
b) Se comunicar diretamente com o operador e com o aluno;
c) Dominar o conhecimento de comandos do veículo podendo controlá-lo quando for necessário;
d) Ensinar, acompanhar o treinamento e dar assistência ao aluno quando for preciso;
e) Avaliar o desempenho do aluno no comando e controle do veículo, máquina ou processo, de acordo aos objetivos do treinamento;
f) Receber as configurações do operador externo, planejar um roteiro de ensino e entregar o conteúdo pedagógico. Dependendo do desempenho do aluno, o tutor

pode mudar o roteiro de ensino e repetir o treinamento de diferentes formas, até considerar que o conjunto de conhecimento foi bem apreendido pelo aluno.

Cada agente especialista manipula um subconjunto de domínio de conhecimento para avaliar o desempenho do aluno. Os agentes avaliadores estão ativos apenas no treinamento do aluno. Os avaliadores são responsáveis pelas seguintes tarefas:

a) Dar apoio ao agente tutor na tarefa de avaliação, assim a carga de trabalho é aliviada, pois processando menos regras a chance de ter erros de sintaxe e de semântica diminuem. Cada avaliador se especializa num domínio específico e processa seu próprio grupo de regras; e
b) Reportar alguma anomalia do sistema ou do simulador quando eles detectarem.

A arquitetura principal do sistema é constituída por seis artefatos interconectados: a carcaça ou módulo principal, o Quadro Negro, o Processador de Dinâmica, a Interface do simulador, a Interface do Aluno e a Interface do Operador.

3.3 ARQUITETURA DO SISTEMA DE TREINAMENTO VIRTUAL PROPOSTO

Nesta seção são descritos os elementos da arquitetura proposta na figura 8, resultantes do projeto conceitual feito no início do projeto.

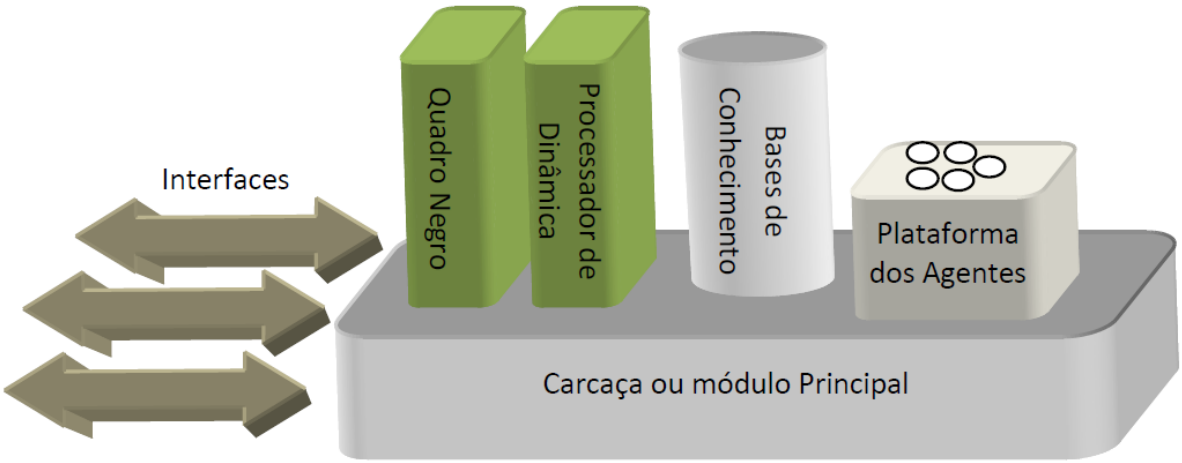
3.3.1Artefatos

Os artefatos seguem as especificações A&A (*Agent and Artefact*), sendo as quatro mais importantes: a autonomia, transparência de linguagem de programação, a capacidade de iteração remota, e a interação por meio de subscrição (RICCI, 2006). Esta recomendação permite uma normalização do sistema que facilita a interoperabilidade, robustez e autonomia dos elementos. O sistema de instrução virtual é suportado por um conjunto de artefatos interconectados. Um dos artefatos (carcaça) serve de plataforma que suporta o sistema. A funcionalidade do conjunto não é definida no sistema mesmo, senão nas bases de conhecimento, incluindo o gerenciamento do sistema.

A carcaça (na literatura usam os nomes "*shell*" ou "*skull*") é a base do sistema. Sua função é gerenciar a interação entre o quadro negro, processador de dinâmica, a interface do

simulador, a interface do aluno, a interface do operador, como mostrado na figura 9. Suporta as bases de conhecimento e a plataforma multiagente ou container, onde os agentes se alojam e operam.

Figura 9 - Carcaça ou "Shell" do sistema completo



Fonte: Autor (2004).

O quadro negro é um recurso compartilhado que mantêm atualizados os estados (variáveis e parâmetros) do simulador externo. É usado pelos agentes para consultar estados ou sobre escrever comandos. Assim, como um quadro real, os agentes podem ler o conteúdo e escrever nele. O que os agentes escrevem no quadro é reflexo no simulador, por meio da interface deste. Apenas o agente tutor pode escrever no quadro negro, pois assim se evita confusão na hora de tentar controlar o sistema do simulador externo. O Quadro negro permite aos agentes acessar a mesma informação do ambiente e, assim, facilitar o processo de inferência, decisão e solução ao problema. A entrada é um buffer circular que compartilha dados com a interface do simulador em forma local ou remota (por exemplo, usando TCP/IP).

A interface do simulador é um artefato que interage diretamente com o simulador externo, podendo acessar ou modificar recursos, variáveis e parâmetros. Os dados que fluem na interface convergem em um quadro negro, pois os mesmos dados são processados por vários agentes. O desenvolvimento desta interface depende totalmente da arquitetura e plataforma do simulador externo, sendo feita em formato de "*plug-in*".

A Interface do Operador é um artefato que permite a interação direta entre o operador e o agente tutor. O operador configura na interface os objetivos e condições do treino, e indica

qual aluno vai estar no simulador (assim, o agente tutor acessa o modelo (perfil) do aluno indicado). O operador também pode se comunicar com o aluno por meio das interfaces.

A Interface do Aluno é um artefato que permite a iteração direta entre o agente tutor e o aluno. Também permite a comunicação entre o aluno e o operador. Testes iniciais mostraram que é melhor alocar a interface no canto da tela e evitar mensagens sem fundo transparente, já que do contrário há perda de foco e distração no treinamento.

3.3.2 Agentes

A sociedade de agentes baseia-se em uma organização em hierarquia horizontal, ninguém manda e ninguém recebe ordens, apenas informam ou pedem informação. A sociedade está formada por um agente principal, o agente Tutor, e um grupo de agentes avaliadores que são especialistas em diferentes domínios. Hierarquia horizontal não quer dizer que todos têm o mesmo grau de importância e de responsabilidade, neste caso, o agente Tutor é o protagonista principal da sociedade e leva a maior carga de trabalho.

A coordenação da sociedade baseia-se em informar os estados e agir de acordo a eles, por exemplo, se um dos agentes detectar um risco iminente de acidente, informa da situação a todos os agentes, então os agentes avaliadores suspendem as atividades e o tutor tenta resolver a emergência. No treinamento do aluno, o Tutor coordena e sincroniza as ações dos outros agentes informando os estágios e estados do treinamento. Dependendo dos estágios, o Tutor pode pedir avaliações parciais do desempenho do aluno. A linguagem usada pelos agentes segue as normas da FIPA (FIPA, 2013) e está representada em uma ontologia baseada em conceitos, predicados e ações. O agente Tutor pode resolver eventuais conflitos entre os outros agentes. O agente Tutor opera em todos os estados do sistema, interage com o operador externo e com o aluno. Pode ensinar e assistir ao treinamento e, também, assumir o controle dos mandos, quando for necessário. Eventualmente, pode resolver algumas emergências com ajuda do especialista em falhas/emergências. O agente Tutor pode acessar e modificar todas as bases de conhecimento, processador de modelo, o modelo do aluno e tem controle total no quadro negro podendo ler e escrever nele. Os agentes avaliadores interagem apenas entre eles e o agente Tutor. Eles operam apenas nos estados de treinamento do aluno e na verificação de sistema. Podem acessar apenas as bases de conhecimento que envolve o domínio de cada um, apenas ler o modelo do aluno, observar o roteiro de ensino, acessar o processador de modelo e apenas ler o quadro negro.

A arquitetura de cada agente é, essencialmente, a mesma, tendo todos a mesma estrutura, entretanto a funcionalidade deles se diferencia nas responsabilidades e privilégios definidos nas bases de conhecimento. Os agentes seguem as especificações da fundação FIPA, incluindo a linguagem ACL entre os agentes e o uso de ontologias com formato FIPA-SL (FIPA, 2013).

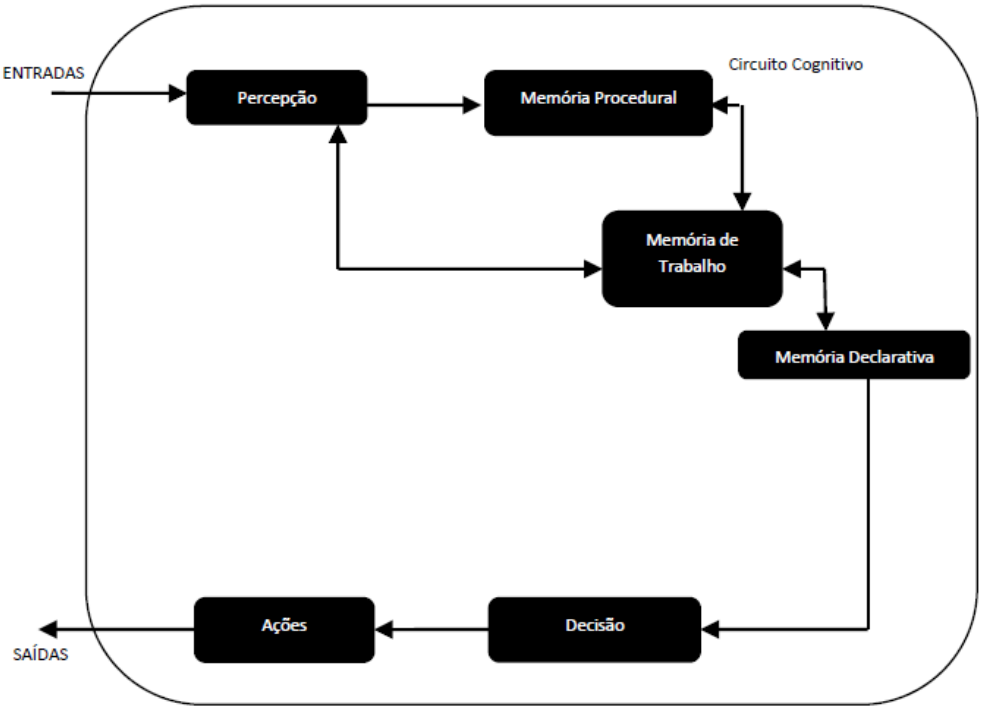
Cada agente acessa uma ontologia de domínio de acordo a sua especialidade, liberando a carga de trabalho do tutor em até 50% (Testes de ping deram até 1ms no máximo para um computador com velocidade de 1GHZ).

Entre as arquiteturas de agentes estudadas, a que atende da melhor forma as especificações é o modelo apresentado na tese "Integração de emoções em agentes inteligentes" (GRAÇA, 2006), já que trabalha diretamente com ontologias. O modelo emocional também é representado numa ontologia, e o aspecto mais importante e decisivo na escolha de arquiteturas é o fato que o comportamento do agente é descrito também por uma ontologia. Este aspecto é importante, porque modificações no comportamento do agente são feitas na base de conhecimento e não na programação do agente. Isto permite criar vários agentes idênticos e definir o comportamento deles nas ontologias.

A arquitetura interna dos agentes é mostrada na figura 10. Cada agente contém uma ontologia própria que define seu comportamento. Nela tem um conjunto de regras que gerencia o módulo de percepção, cuja função é transformar dados externos em informações ou fatos internos, que são armazenados na memória de trabalho. Dentro da mesma ontologia tem um conjunto de relações e regras que descrevem as emoções do agente e que definem o seu comportamento. Estas são processadas na memória declarativa.

A memória procedural processa a ontologia que representa o domínio de conhecimento do agente. Tanto a memória declarativa como a procedural compartilham a memória de trabalho, onde são registrados os fatos resultantes do processo de raciocínio do agente.

Figura 10 - Arquitetura interna dos agentes



Fonte: Graça, 2006.

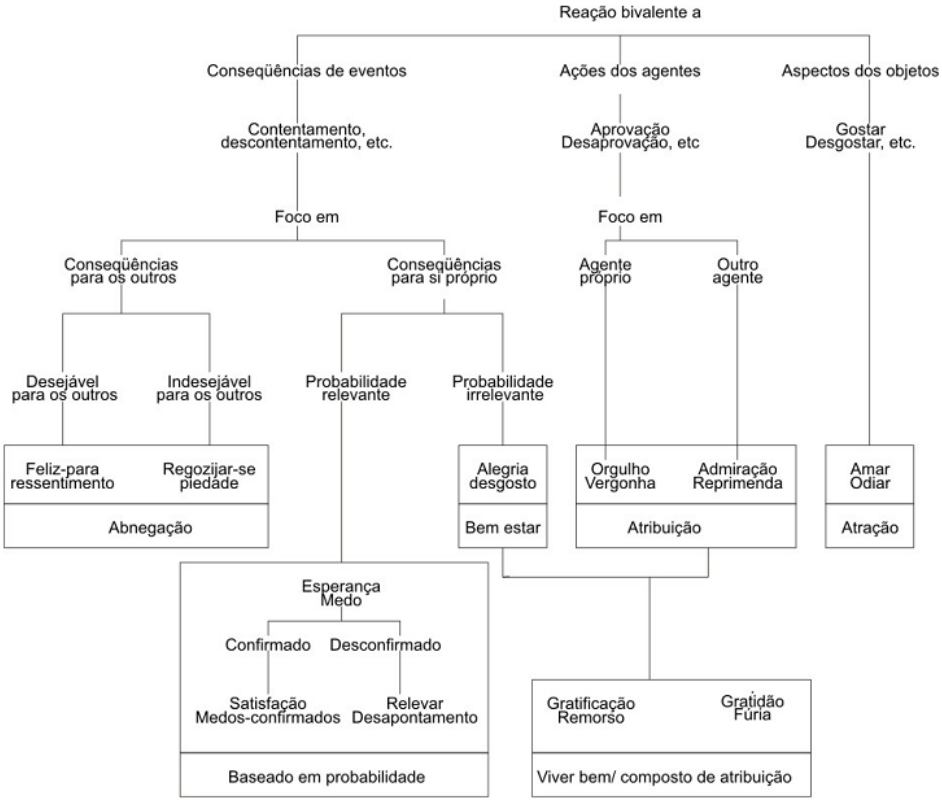
O módulo de Decisão é um filtro inteligente que avalia os planos de ações (lista de ações atômicas, ou seja, cada ação é feita em apenas um passo) a serem executados pelo agente. Cada plano de ações é avaliado sob critérios como: disponibilidade de recursos, disponibilidade de tempo, falibilidade, ações conflitantes, possíveis ações que poderiam bloquear outros processos, fatores externos, entre outros.

O módulo de Ações é um conjunto de interfaces que permitem executar diversos tipos de ações, entre elas: manusear artefatos, se comunicar com outros agentes, interação remota com outros sistemas, livrarias de protocolos, linguagens de ontologias, middlewares, etc.

O processo cognitivo no agente acontece em processos cíclicos, em que as informações externas são convertidas em fatos no módulo de Percepção. Os fatos de nível médio são processados diretamente na memória procedural e os fatos de alto nível são enviados para a memória de trabalho. Esses fatos são lidos e processados pela memória declarativa, onde são mapeados e processados. Ao final do processamento, os novos fatos gerados são enviados para a memória de trabalho, e os planos de ações são enviados para o módulo de decisão. Os fatos registrados na memória de trabalho são lidos pelo módulo de Percepção (para processamento reativo) e pela memória procedural. O módulo de Decisão ao selecionar o plano de ações, envia o plano para executar as ações no módulo de ações. O modelo emocional dos

agentes é descrito na base do conhecimento de cada um, e se baseia no modelo OCC, cujas siglas correspondem aos sobrenomes dos autores Ortony, Clore e Collins (ORTONY et.al., 1988). Este modelo descreve como seria a emoção do mesmo agente e dos outros diante certa situação. Por exemplo, se o aluno reprovar um treinamento o agente tutor sentiria frustração e o aluno vergonha. A figura 11 apresenta o modelo OCC traduzido do artigo original (ORTONY et.al, 1988). O modelo OCC considera três fatores emocionais que são: a desejabilidade (emoções baseadas em eventos), a laudabilidade (emoções baseadas em atribuição) e a atratividade (emoções baseadas em objetos ou ideias abstratas). A desejabilidade é avaliada em função dos objetivos e a interpretação dos eventos. A laudabilidade é avaliada em função das ações executadas pelos agentes ou usuários humanos baseado em fatores culturais, crenças, moral, etc. A atratividade é avaliada em função ao sentimento ou atitudes do agente quando vê ou pensa em objetos ou ideias.

Figura 11 – Modelo OCC



Fonte: Traduzido do artigo (ORTONY et.al, 1988).

3.3.3 Bases de Conhecimento

As bases de conhecimento são o núcleo que determinam a funcionalidade e gerenciamento do sistema. Representa o gerenciamento dos agentes (protocolos de comunicação, conflitos, comportamentos, responsabilidades e privilégios, etc), o conhecimento a ser ensinado e avaliado, o registro de avanço de aprendizado dos alunos e seus perfis emocionais (modelo do aluno).

Nos sistemas baseados em conhecimento existem três tipos principais de raciocínio:

a) Raciocínio baseado em regras (sistemas de produção, sistemas especialistas, etc.);
b) Raciocínio baseado em modelos (interpretador de ontologias, grafos, etc.);
c) Raciocínio baseados em casos (CBR).

No caso do sistema de treinamento virtual proposto, é usado raciocínio baseado em regras e raciocínio baseado em modelos. As bases de conhecimento são organizadas em três categorias.

A primeira categoria corresponde às bases de conhecimento de treinamento, que representam os domínios relacionados ao ensino e à linguagem entre os agentes. Estas são organizadas da seguinte forma:

a) Ontologias de Modelo: As ontologias de modelo descrevem as partes e subpartes de um sistema (aeronave, processo e ambiente) e as relações entre elas. Também descrevem em forma quantitativa (com sistemas de equações) as variáveis do sistema, malhas de controle e algoritmos de resolução (referências de chamados a métodos da classe do processador de dinâmica);
b) Ontologias de Domínio: As ontologias de domínio representam o conhecimento que envolve a operação do sistema representado na ontologia de modelo;
c) Ontologias de Conteúdo: As ontologias de conteúdo representam o material pedagógico que o agente tutor apresenta para o aluno no treinamento;
d) Ontologias de Linguagem: As ontologias de linguagem descrevem a sintaxe dos fatos processados pelos agentes e das mensagens que eles enviam e recebem. Não requer aquisição de conhecimento.

A segunda categoria corresponde às bases de conhecimento de Gerenciamento, que representam as responsabilidades, protocolos, modelos emocionais e comportamentos dos agentes.

A terceira categoria corresponde ao Modelo do aluno, que é uma base de conhecimento que representa os alunos em treinamento, contendo o perfil de desempenho, perfil emocional observado, perfil comportamental e histórico dos treinamentos.

3.3.3.1 Ontologia de modelo

A ontologia de modelo representa o avião do simulador, incluindo uma descrição estrutural, descrição de dinâmica e descrição de controle. Também, inclui uma representação do ambiente (geografia do cenário e condições meteorológicas). Esta ontologia é usada pelos agentes para as seguintes tarefas:

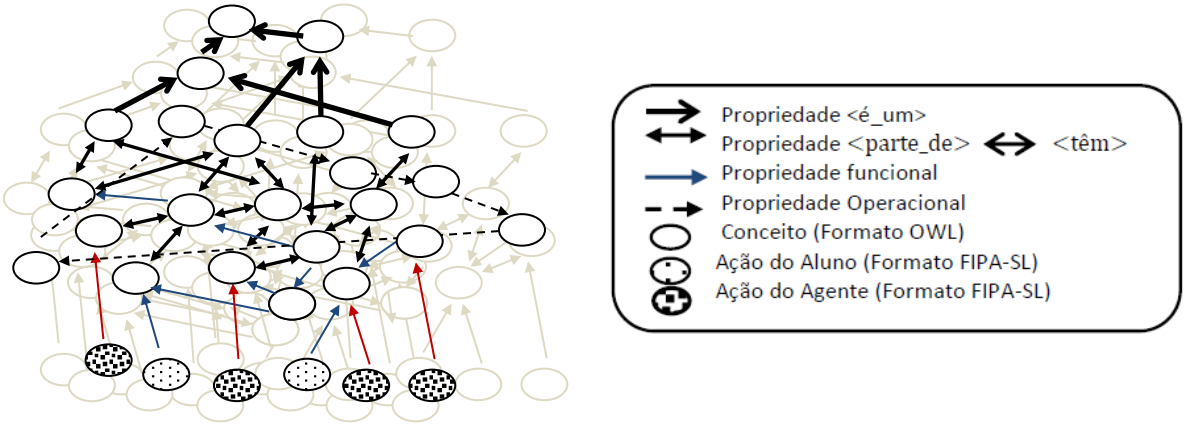
a) Verificar a consistência qualitativa do funcionamento do sistema, assim é possível procurar falhas no veículo, máquina ou processo, e falhas no simulador externo;
b) Analisar as condições do ambiente, tanto meteorológicas como geográficas, para planejar roteiros de navegação para veículos;
c) Permitir ao agente colher informação de equações, malhas de controle, parâmetros, vetores e matrizes que representam da dinâmica do sistema e acionar o processador de dinâmica para executar simulações. No caso que precisar, o agente pode configurar o processador de dinâmica para que controle o veículo do simulador externo.

Cada agente processa internamente uma instância (cópia) da ontologia de modelo, já que esta ontologia é a representação mental que cada agente tem do veículo do simulador. A ontologia de modelo tem três camadas: a camada estrutural, camada operacional e camada funcional.

A camada estrutural descreve os sistemas e subsistemas da aeronave num nível conceitual. A figura 12 mostra uma visão global da camada estrutural. Os círculos brancos representam os conceitos do modelo (podem ser partes, subpartes, sistemas, subsistemas, variáveis, operadores, etc.). Os círculos com recheio representam as ações do aluno feitas no simulador externo e as ações do agente que processa a ontologia. As ações do aluno e as ações

do agente devem ser definidas no formato FIPA-SL. O predicado <*é_um*> indica que um conceito sujeito é um tipo de conceito objeto (p.ex. <*é_um(planador,aeronave*)> indica que um planador é um tipo de aeronave). As setas de dois sentidos indicam em um sentido que um conceito objeto é parte (<*parte_de*>) de um conceito sujeito, e no sentido contrário indicam que um conceito sujeito tem (<tem>) um conceito objeto (p.ex. a expressão <*parte_de(asas,avião)*> indica que as asas são parte de um avião, e a expressão inversa <*tem(avião,asas)*> indica que um avião tem asas).

Figura 12 – Camada estrutural de uma ontologia de modelo



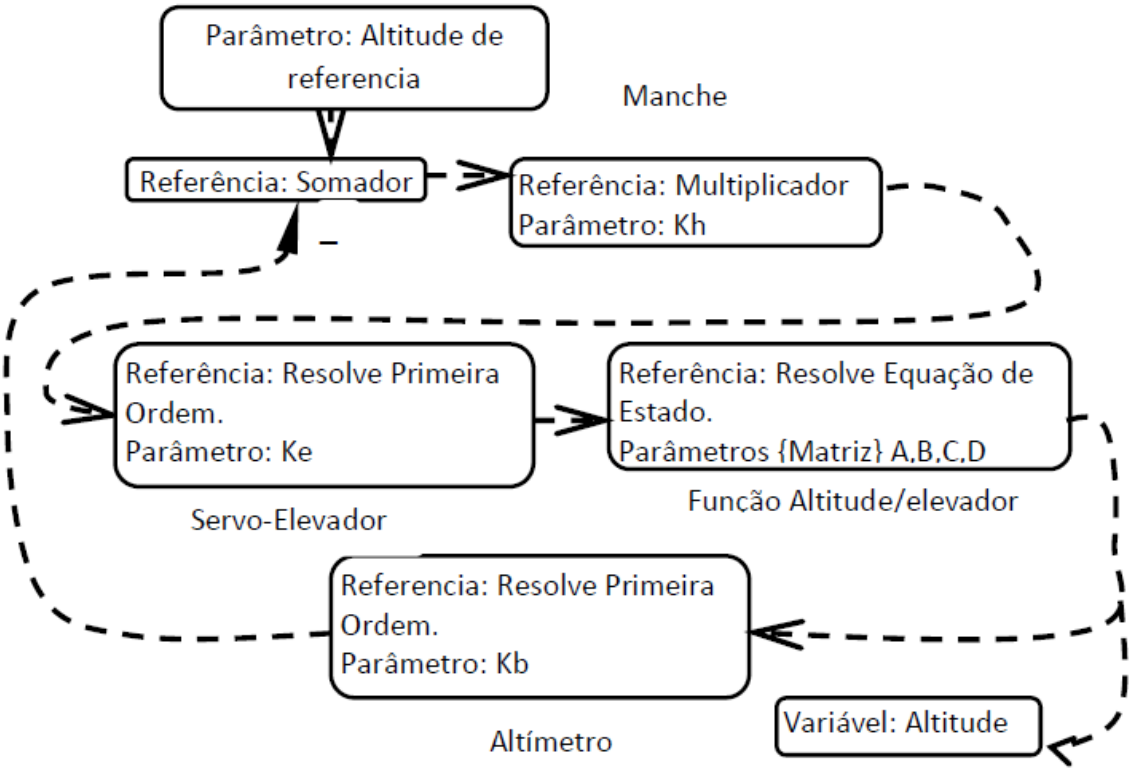
Fonte: Autor (2014).

A camada operacional descreve a dinâmica das malhas de controle do sistema. As setas tracejadas indicam propriedades operacionais ou predicados de controle. Os predicados de controle encadeiam conceitos que, no conjunto, formam malhas de controle. Os conceitos envolvidos contêm parâmetros tais como: constantes, vetores e matrizes. Também eles dão referências de chamados a métodos do processador de dinâmica, que executam operações matemáticas tais como: integrar, diferenciar, somar, multiplicar, resolver equações de estado, algoritmos de resolução, etc.

O padrão de mapeamento de uma malha de controle está definido por um grupo de variáveis de referência, um grupo de mandos, um grupo de servo acionadores, um grupo de funções de transferência e um grupo de sensores. O sinal que entra no grupo de mandos é a subtração entre o sinal do grupo de variáveis de referência e o sinal do grupo de sensores. O sinal de saída do grupo de mandos entra no grupo de servo acionamentos. A saída do grupo de servo acionamentos corresponde à posição angular da superfície de controle ou da servo válvula em questão. A entrada do grupo de funções de transferência corresponde à saída do grupo de

servo acionadores. A saída do grupo de funções de transferência corresponde à variável que deseja ser controlada. A realimentação do sistema de controle corresponde ao grupo de sensores. Por exemplo, para representar um sistema de controle de altitude numa aeronave, pode usar a seguinte malha de controle mostrada na figura 13.

Figura 13 – Exemplo de encadeamento operacional no controle de altitude



Fonte: Autor (2014).

A camada funcional da ontologia de modelo é utilizada pelo processador de dinâmica. Os agentes podem acionar este artefato em duas situações. Primeiro, quando algum agente precisa fazer uma (s) simulação (ões) no processador de dinâmica para apoiar alguma decisão; e, segundo, quando o agente tutor entra em um estado mental emocional de pânico ele aciona o processador de dinâmica para tomar controle do avião do simulador externo. Em qualquer um dos casos o processador de dinâmica mapeia, na ontologia de modelo, os predicados operacionais e os encadeamentos entre sujeitos e objetos.

Os sujeitos e objetos contêm, nos atributos (propriedades), referências aos chamados métodos do processador de dinâmica, os quais são executados na sequência mapeada no

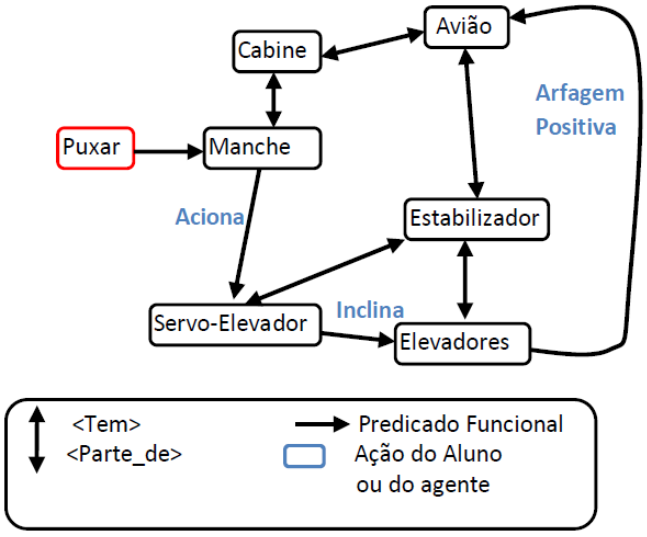
encadeamento operacional dos fatos. As propriedades funcionais indicam que, quando uma ação do aluno ou ação do agente é feita, é gerado um encadeamento funcional envolvendo os elementos (conceitos e predicados funcionais) diretamente relacionados.

Por exemplo, a ativação do predicado <*puxa*> na expressão <*puxa(piloto,manche)*> gera uma cadeia funcional com as seguintes expressões:

puxa(piloto,manche)->aciona(manche,servo_elevador)=>
inclina(servo_elevador,elevadores) => arfagem_positiva(elevadores,avião).

Ou seja, se um piloto (aluno ou agente) puxar o manche da cabine de um avião, este aciona um servo mecanismo hidráulico do avião, que faz inclinar os elevadores. A inclinação dos elevadores faz que o avião suba o nariz (arfagem positiva). A figura 14 ilustra o exemplo anterior. A ação <*Puxa*> pode ser executada pelo aluno ou pelo agente, gerando um encadeamento funcional, representado pela sequência de setas uni-direcionais.

Figura 14 – Exemplo de encadeamento funcional



Fonte: Autor (2014).

As setas de um sentido indicam generalização (subclasses) e as setas de dois sentidos indicam composição (composição significa que um elemento pode apenas ser parte de um

conjunto e não pode pertencer a outro, p.ex. o motor esquerdo é parte da asa esquerda e não pode pertencer a outro conjunto).

O conceito (classe) Operador contém, nos adjetivos (atributos), referências chamadas de métodos da classe do módulo processador de dinâmica. As operações mais importantes são a integração e a soma. Qualquer representação dinâmica de veículo, máquina ou processo pode ser conformada apenas com integradores e somadores após uma manipulação matemática.

As variáveis são representadas como conceitos (classes) ao invés de propriedades, porque é necessário relacioná-los com predicados simbólicos do tipo *aumenta, diminui, esquerda, direita, para cima, para baixo*, etc.; e os atributos numéricos das variáveis são representados como predicados de operação numérica (p.ex.*, =100, >100, <100*, etc.), que podem ser interpretados pelas máquinas de inferência dos módulos do agente.

O encadeamento funcional é usado pelos agentes para verificar qualitativamente o funcionamento do modelo. O processo de verificação qualitativa do modelo é iniciado quando o agente ou aluno executa alguma ação, e o mesmo agente (ou outro) gera o encadeamento funcional, e compara o último fato da cadeia com o que acontece no simulador externo.

Se houver um fato equivalente gerado pelo módulo de percepção do agente, quer dizer que o avião do simulador está se comportando como esperado. Se não for assim, aquele fato é marcado como conflitante e o agente inicia uma busca de falha.

No processo de busca de falha no avião do simulador, o agente gera uma lista de cadeias funcionais, que envolvem o fato conflitante. Para cada cadeia, o agente inicia a busca no penúltimo fato da cadeia. Em cada fato revisado, o agente liga e desliga no processador de dinâmica o elemento do avião (conceito objeto) e observa o comportamento deste. Se o comportamento for equivalente ao comportamento do avião no simulador externo, o agente conclui que o conceito do objeto associado é a parte do avião que está falhando. Se ao final da busca não houver comportamentos equivalentes entre o processador de modelo e o simulador externo, o agente conclui que a falha é do simulador externo.

O algoritmo de verificação qualitativa e de busca de falhas é mostrado no quadro 2.

Quadro 2 – Algoritmo de verificação qualitativa para busca de falhas

```
  Verificação qualitativa:{
    na ocorrência de (ação) {gerar (cadeia funcional);}
procura fatos(gerado pelo modulo percepção);
procurar pelos fatos com predicado equivalente ao (ultimo fato) da (cadeia funcional);
Se existir (algum fato equivalente)  { comportamento do avião é o esperado;}
           senão  {Marcar (ultimo fato) da (cadeia funcional) como (fato conflitante);
                  iniciar(busca de falhas);}
  }

  Busca de falhas:{
        Gerar todas as cadeias funcionais que envolvam o (fato conflitante);
        Para cada cadeia funcional fazer {iniciar no fato (n-1)  finalizar no fato (1)
                                          {Identificar (objeto) do (fato);
      Se (acionado no processador de dinâmica){desligar(objeto);}
          senão {desligar (objeto);}
            Observa comportamento (processador de dinâmica)
                           Observa comportamento (simulador externo)
                           Se comportamentos fossem equivalentes
                                 {falha está no (objeto) do avião;
                                  finaliza (Busca de falhas)}
                                              senão Se (acionado processador de dinamica)
                                               {desligar(objeto);}
                                      senão {ligar (objeto);}
                                                                                    }
                                         }
        Se (não existir (fato conflitante equivalente){A falha está no simulador externo}
```

Fonte: Autor (2014).

3.3.3.2 Ontologia e Regras de Domínio

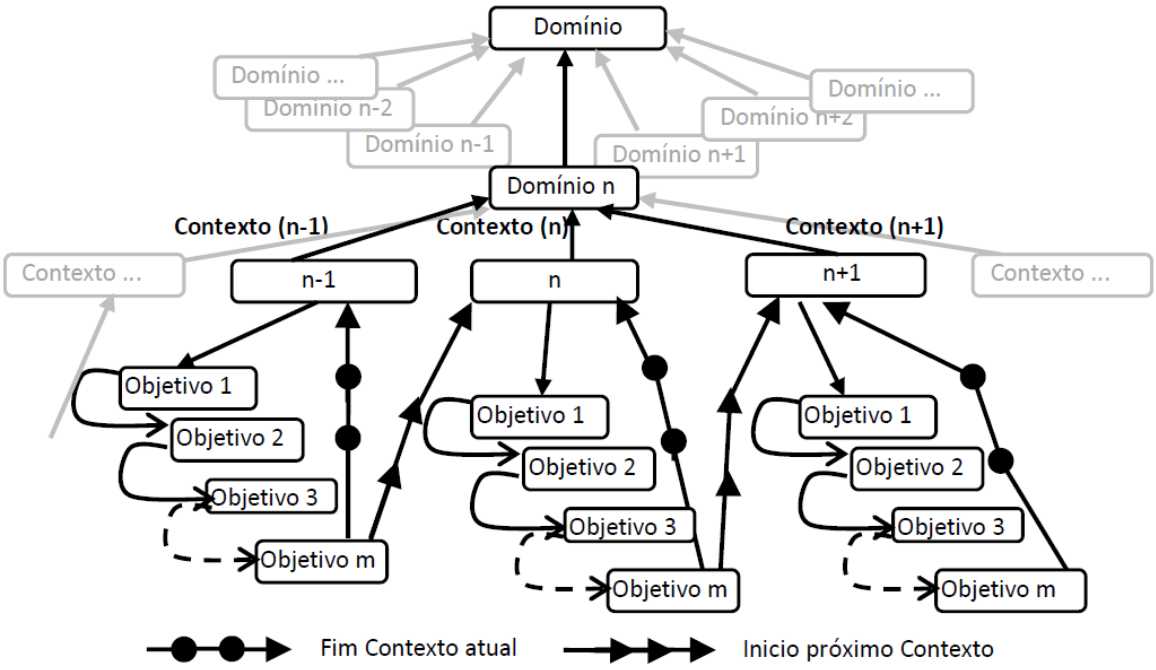
As ontologias de domínio representam todo o conhecimento que envolve a operação do veículo, máquina ou processo, incluindo controle, comando, regulamento, normas, medidas de segurança, iteração com o ambiente (condições ambientais), orientação, navegação, ergonomia, etc. Estas estruturas foram idealizadas criadas pelo autor em função da não existência de trabalhos correlatos que permitam que um agente possa interpretar e processar a dinâmica de um veículo.

Cada agente processa apenas um domínio. Para cada domínio devem ser definidos os contextos de domínio (estados, estágios, etapas, etc.). Para cada contexto deve ser definido um conjunto de regras de domínio e uma cadeia de conceitos de estado. Os conceitos de estado são elementos equivalentes aos estados de uma máquina de estados finitos. Os conceitos de estado descrevem “o quê” deve ser feito e as regras de domínio descrevem “como” deve ser feito.

O lado esquerdo das regras de domínio define o objetivo (a ação que se espera que seja executada) e o lado direito da regra define o próximo objetivo (a próxima ação que se espera que seja executada) especificando o conjunto de ações que deveriam ser executados para

conseguir o objetivo. A figura 15 ilustra uma ontologia de domínio. Um contexto inicia (seta com círculos) quando o contexto anterior finaliza (seta com triângulos). Cada objetivo é ativado, consecutivamente, até chegar ao último objetivo, terminando o contexto atual e iniciando o próximo contexto.

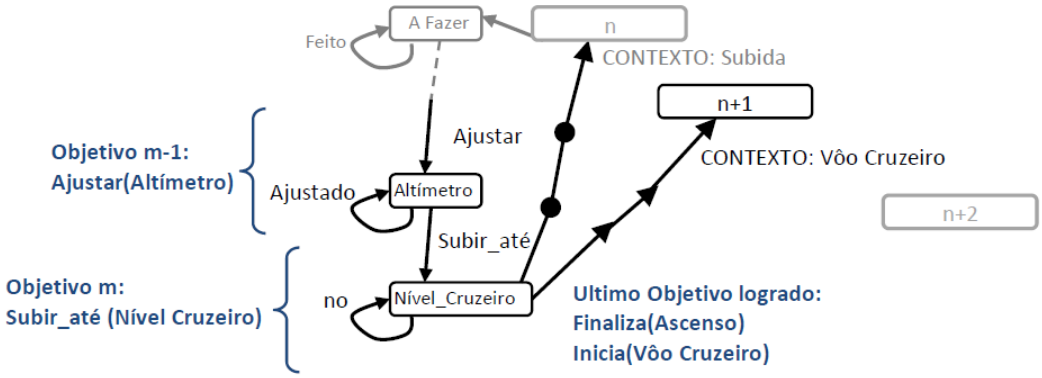
Figura 15 – Estrutura de uma ontologia de Domínio



Fonte: Autor (2014).

Os contextos são estados que podem ser ativados um por vez (não pode existir mais de um contexto ativado), representados por predicados que podem ser verdadeiros ou falsos (booleanos). Podem existir contextos com o mesmo nome em diferentes domínios e operam em forma equivalente a um chamado morfológico de método (equivalente, mas funcionalmente não é o mesmo). Se a memória de trabalho de cada agente conter um fato com predicado contextual, automaticamente, é ativado aquele contexto em todos os domínios. A figura 16 ilustra um exemplo em que o contexto atual é da rampa de subida de um avião e o objetivo atual (*m*) é de subir até o nível cruzeiro. A regra do objetivo atual foi disparada quando o objetivo anterior foi logrado. No lado direito da regra indica o próximo objetivo, que é subir até o nível cruzeiro, e também define uma lista de expectativas de ações a serem executadas. Quando todas as ações estiverem feitas é adicionado um fato que indica que o avião está no nível cruzeiro. Com o fato anterior, a regra de fim de contexto é acionada, finalizando o contexto de subida e iniciando o contexto de voo cruzeiro.

Figura 16 – Exemplo de conceitos de estado e regras de domínio




```
Regra (Objetivo m
        ?x<- Ajustado(Altímetro)
=>
        retract ?x
        assert<objetivo>: Subir_até(Nível_Cruzeiro)
        assert<espectativa>{<ação1>,<ação2>...<ação n>
)                                          ->assert: no(Nível_Cruzeiro)}

Regra (Fim Contexto
         ?x <- no(Nível_Cruzeiro)
=>
         retract  ?x
         assert <ação>: Finaliza(Subida)
         assert <ação>: Inicia(Vôo Cruzeiro)
    )
```

Fonte: Autor (2014).

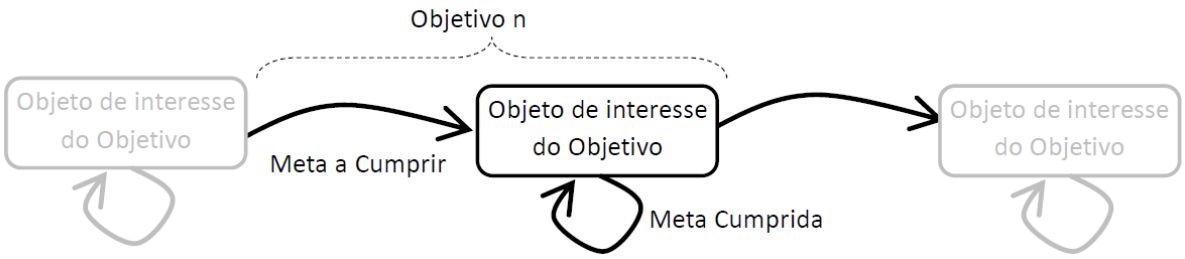
Todos os agentes podem avaliar o desempenho do aluno. O mecanismo de raciocínio para avaliação vem em duas etapas ou processos paralelos: a primeira avalia eventos nas variáveis e a segunda avalia os procedimentos do aluno.

Na avaliação de eventos as regras de domínio de nível médio são carregadas na memória procedural. Os eventos avaliados são assíncronos e, normalmente, podem indicar pontos de referência de navegação (na pista, passando por uma cidade, passando por uma referência fixa, etc.), condições de voo (nivelado, planado, estol, etc.), ou algum erro do aluno (não se manter na aerovia, não se manter no eixo da pista no pouso ou decolagem, manter os flapes sobre a velocidade limite, navegação errada, etc.). Na ocorrência de eventos são gerados fatos que são adicionados na memória de trabalho. Na verificação deve se observar qual desses fatos são usados na memória declarativa para poder acompanhar o encadeamento lógico.

Na avaliação de procedimentos, as regras de alto nível são carregadas na máquina de inferência da memória declarativa, enquanto o interpretador de ontologias acessa a ontologia de domínio específico para cada agente. O mecanismo de raciocínio é radicado na estrutura da

ontologia de domínio, como se fosse um roteiro de regras sequenciais de raciocínio. Na camada de objetivos são definidos predicados de metas a serem cumpridas, predicados de metas cumpridas e conceitos que representam o "objeto de interesse" do objetivo. A figura 17 mostra esta camada.

Figura 17 – Estrutura da camada de objetivos da ontologia de domínio



Fonte: Autor (2014).

O objetivo é definido como a expressão ou fato composto pela meta a cumprir e pelo objeto de interesse (p.ex. <*esperar(transição)*>, <*ajustar(altímetro)*>, etc.). A regra de objetivo é uma regra de alto nível (processado na memória declarativa) acionada pelo objetivo (fato). O lado direito da regra define um fato condicionado composto pela meta cumprida e o objeto de interesse (p.ex. <*na(transição)*>, <ajustado(altímetro)>, etc.) e é ativado se um grupo de ações for executado. O quadro 3 mostra um exemplo deste processo.

Quadro 3 – Sintaxe e exemplo das regras de objetivos

Sintaxe da regra de objetivo	Exemplo
(Regra de objetivo (?x<- (meta_a_cumprir(objeto_de_interesse)) => => ((retract (?x)) ((retract(?x)) (assert(meta_cumprida(objeto_de_interesse))))conditioned_to)conditioned_to {<action> a_1 : <action> a_n}))	(Regra ajuste_altimetro (?x<- (ajustar(altimetro)) (assert(ajustado(altímetro))) {<action> reduzir=200(Veolcidade) <action> esperar=200(Veolcidade) <action> =1010(Bar_Altimetro)}

Fonte: Autor (2014).

O objeto de interesse refere-se ao elemento "objeto" definido na especificação de ontologia OWL. O fato condicionado é usado para acionar as regras de ontologia de domínio na ativação de objetivos. Pode ser que na estrutura da ontologia os objetivos não sejam

sequenciais, mas eles são considerados logicamente sequenciais. O processo de avaliação é cíclico e consiste em três passos:

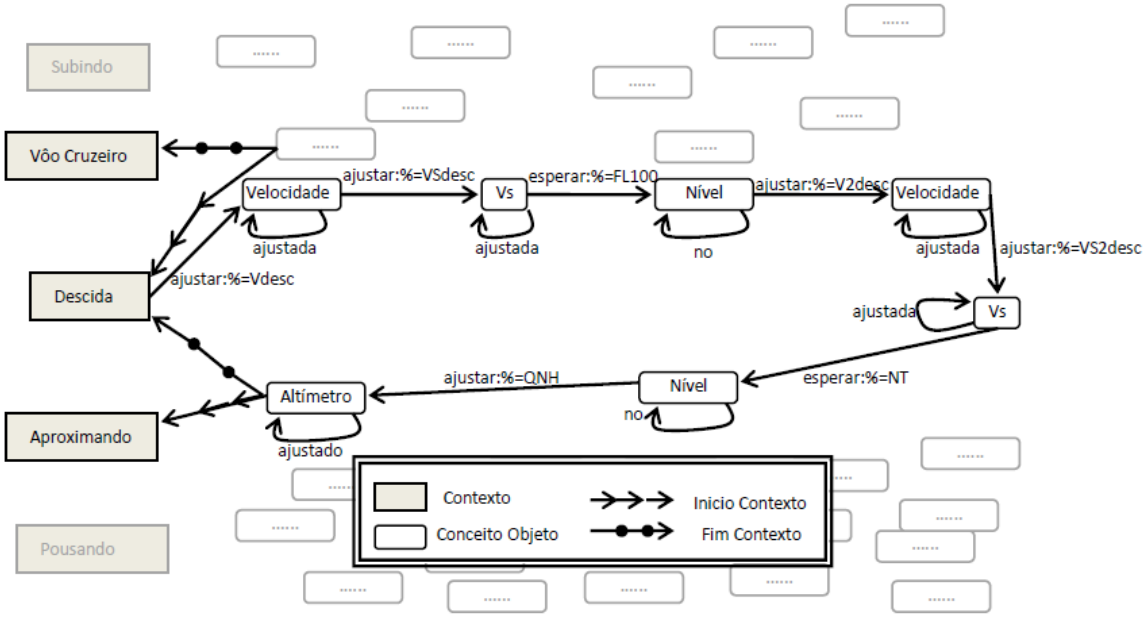
a) Quando o aluno executa alguma ação o módulo de percepção o interpreta como fato de baixo, médio ou alto nível (fatos de nível baixo e médio são processados na memória procedural, enquanto que os fatos de alto nível são processados na memória declarativa);
b) O agente observa os fatos de alto nível e os compara com a lista das ações definidas no fato condicional do lado direito da regra de objetivo;
c) Se todas as ações fossem feitas o fato condicional é acionado, ativando a regra do próximo objetivo. O agente tutor avalia o aluno positivamente, mas desconta pontuação dependendo do tempo que demorou em executar as ações e das tentativas erradas que fez antes de fazer as ações certas.

Para exemplificar um caso, a considerar uma estrutura simplificada de ontologia de domínio no contexto de “Descida”, começando com o fim do contexto “voo cruzeiro” e finalizando com o início do contexto de “Aproximação”. As camadas de “Objetivos” contêm oito objetivos, completando um ciclo de objetivos. O ciclo de objetivos termina quando finaliza o contexto atual e inicia o próximo contexto.

A figura 18 mostra a estrutura do exemplo. O símbolo %= indica uma faixa, onde um valor de referência considera-se pertencente ao conjunto de valores que satisfazem a expressão (p.ex. %=100(Velocidade) define uma faixa de valores de velocidade em torno a 100 Kias que satisfazem a expressão, tornando o fato verdadeiro).

Cada vez que as ações do aluno encadeiam fatos, e esses fatos completam a lista de ações do fato condicional do lado direito da regra de objetivo, um novo objetivo é gerado. As conclusões do agente acontecem na memória declarativa e representam estados mentais.

Figura 18 – Exemplo de um ciclo de objetivos



Fonte: Autor (2014).

O agente tutor pode controlar autonomamente o sistema (veículo, máquina ou processo), acessando a mesma estrutura de ontologia de domínio que usa para avaliar o aluno. O mecanismo de raciocínio é equivalente ao mecanismo de avaliação, exceto que as regras de objetivos são processadas usando encadeamento reverso (*backward chaining*). Ao invés de esperar que o fato condicional seja verdadeiro ao atender a lista de ações, o agente executa as ações em forma sequencial e logo valida o lado esquerdo da regra. Em outras palavras, o agente tutor segue a sequência de objetivos, executando a lista de ações contidas nas regras de objetivo. O controle é feito quando o agente tutor executa as ações escrevendo os comandos no quadro negro. Os comandos escritos no quadro negro passam ao simulador externo por meio da interface de simulador.

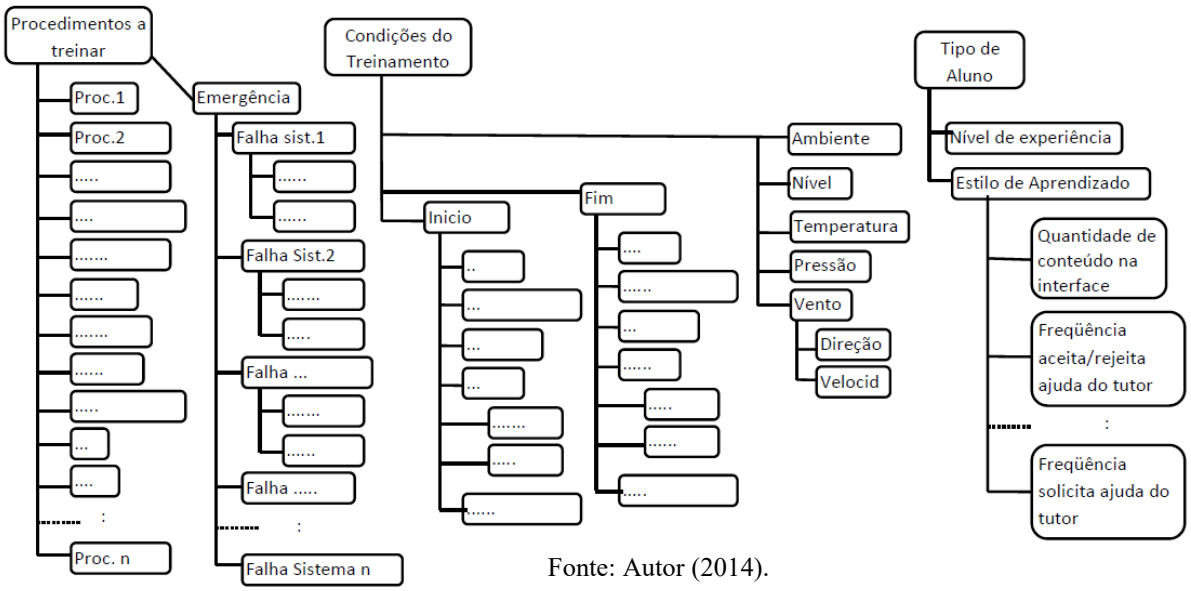
Existem duas formas de assistência ao aluno: o aluno pode pedir ajuda se não souber como proceder, ou o agente tutor pode oferecer ajuda ao aluno se ele errar certo limite de procedimentos, ou se demorar certo tempo em executar alguma ação. Em ambos os casos, o mecanismo que o agente tutor utiliza para assistir o aluno consiste em dar sugestões na tela com base nas ações listadas na regra do objetivo atual. Uma boa forma de conferir as regras é implementando um mecanismo de explicação, permitindo que seja impresso na tela o

encadeamento lógico de cada regra disparada em cada ciclo e um comentário que explique como se chegou naquela conclusão parcial. Assim, facilitaria muito o processo de verificação.

3.3.4 Ontologia de Conteúdo

A ontologia de conteúdo representa o material pedagógico que o agente tutor apresenta ao aluno na interface. A ontologia de contexto é estruturada em duas camadas. A primeira camada é um grupo de conceitos que representam os objetivos, condições (restrições, ambiente, etc.) e qualidades esperadas no nível aluno. A figura 19 mostra a estrutura da hierarquia na primeira camada.

Figura 19 – Primeira camada da ontologia de conteúdo

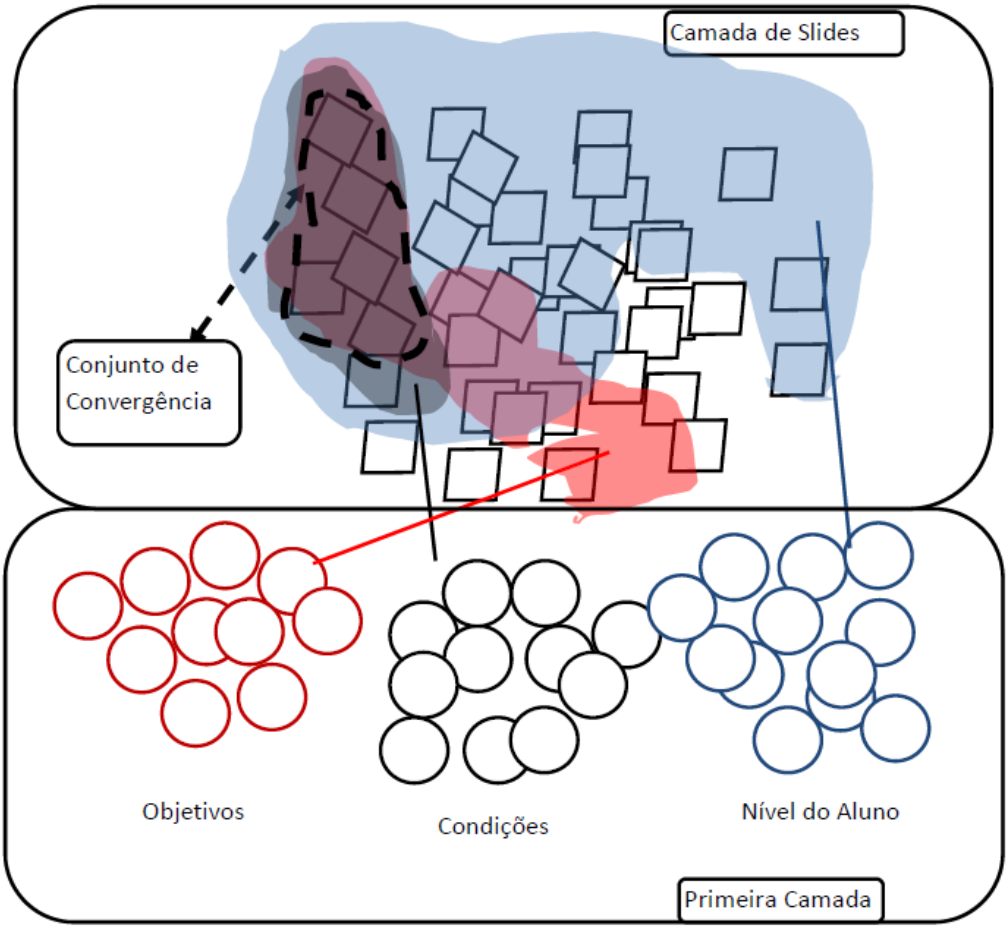


Fonte: Autor (2014).

A segunda camada é uma base de slides (conceitos com identificadores que representam um slide). Quando o agente tutor acessa esta ontologia inicia um processo de busca na primeira camada da ontologia, até convergir em um grupo de slides da segunda camada. O conjunto de convergência de slides é o roteiro de treinamento. Assim, o agente mostra ao aluno os slides selecionados em sequência até finalizar o treinamento.

A figura 20 mostra uma visão global das duas camadas de uma ontologia de conteúdo.

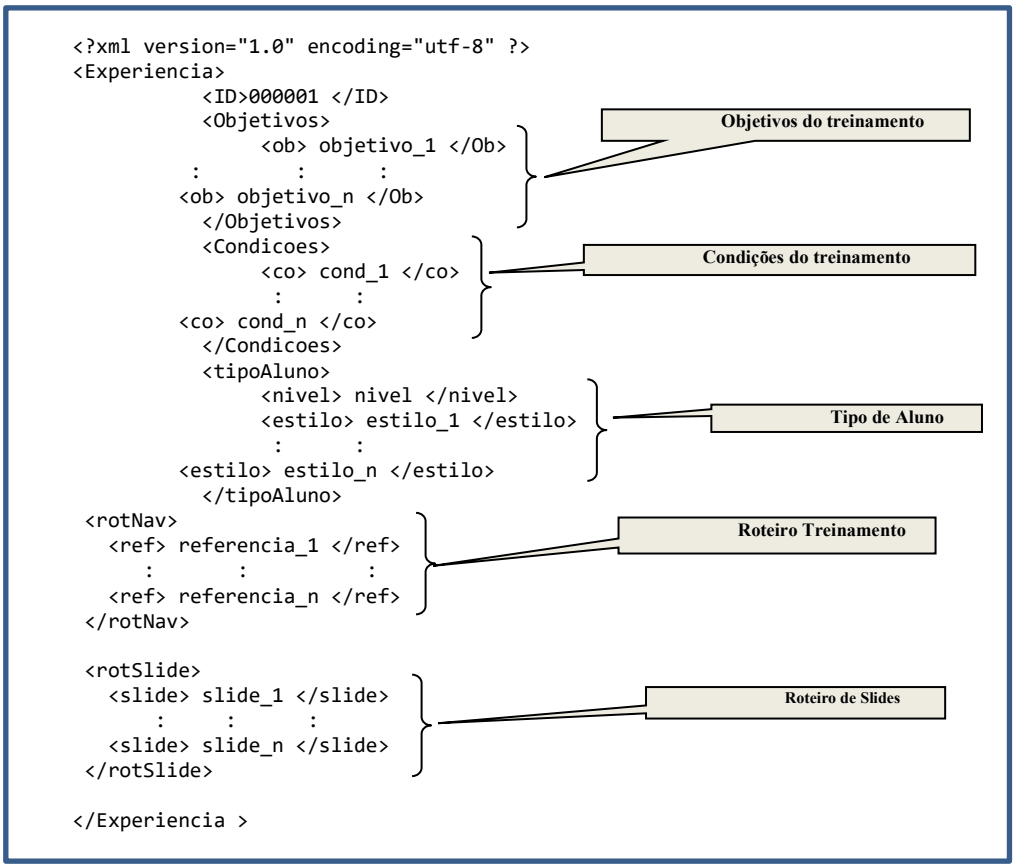
Figura 20 – Visão global das duas camadas de uma ontologia de domínio



Fonte: Autor (2014).

O agente tutor deve conservar um container de experiências pedagógicas, que é um registro dos roteiros de treinamentos. Desse modo, em um novo treinamento, se os objetivos, condições e capacidades do aluno encaixar em algum roteiro anterior, o agente recolhe a experiência guardada e executa o roteiro. Se no meio do treinamento o agente perceber que o aluno não encaixa no roteiro, ele pode fazer uma nova busca, modificando algum objetivo ou condição e executar um novo roteiro. As experiências são encapsuladas no formato XML dentro de objetos "*Experiência*". Estas experiências são armazenadas em um container de experiências (objetos) na memória procedural do agente tutor. A estrutura das experiências consta de cinco partes principais: os objetivos do treinamento, as condições do treinamento, tipo de aluno, roteiro de navegação e roteiro de slides. O molde ou *template* de uma experiência é mostrado na figura 21.

Figura 21 –*Template* XML de uma experiência



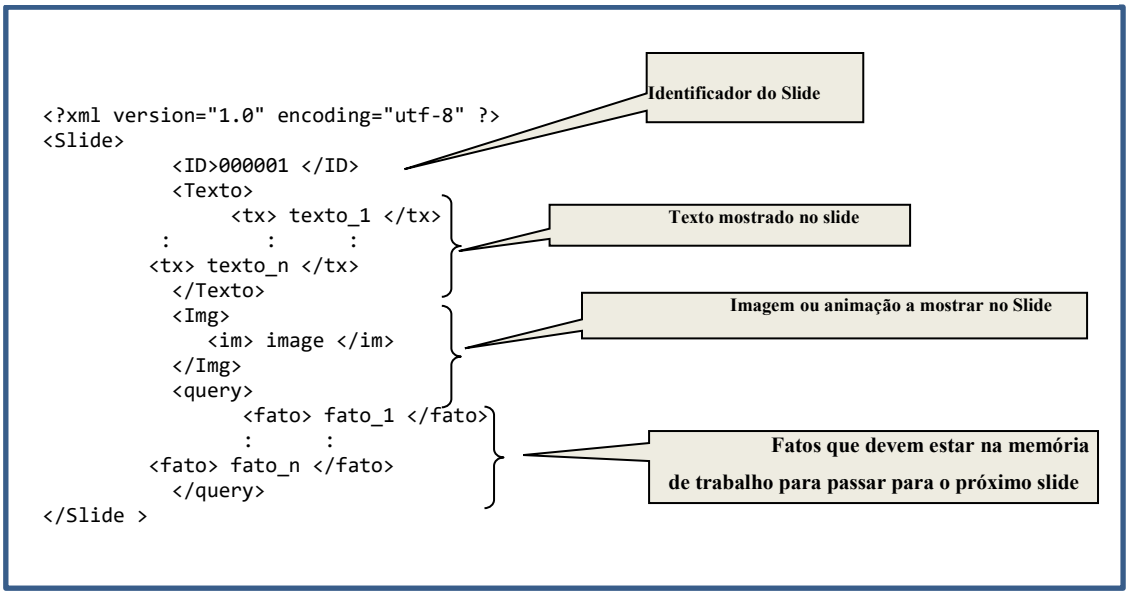
Fonte: Autor (2014).

No *template*, os objetivos são os objetivos do treinamento. As condições são as especificações do treinamento desejado. O tipo de aluno especifica o seu nível de experiência (pontuação numérica) e o estilo de aprendizado ou preferências dele, podendo ser, por exemplo, que não gosta de usar assistente, se distrai facilmente se houver muitas dicas na tela, etc. O formato das experiências usado neste protótipo não cumprem todas as especificações de objetos de aprendizagem (objetos pedagógicos), mas é altamente recomendado aos projetistas, para quando desenvolverem este tipo de sistemas de instrução virtual, para treinamento com veículos, utilizem a abordagem de objetos pedagógicos (SILVEIRA et.al, 2005).

Os slides são encapsulados em formato XML dentro dos conceitos slides (conceitos são representados com objetos).

A figura 22 mostra um *template* ou molde para apresentação de slides na interface do aluno.

Figura 22 – *Template* de um slide



Fonte: Autor (2014).

Cada slide tem um identificador para ser mostrado na sequência correta, uma lista de frases que podem ser apresentadas como texto ou como voz na interface do aluno, uma referência a imagem ou animação a ser mostrada na interface do aluno, e, finalmente, deve ser definida uma lista de fatos que devem existir na memória de trabalho para finalizar o slide atual e passar para o próximo.

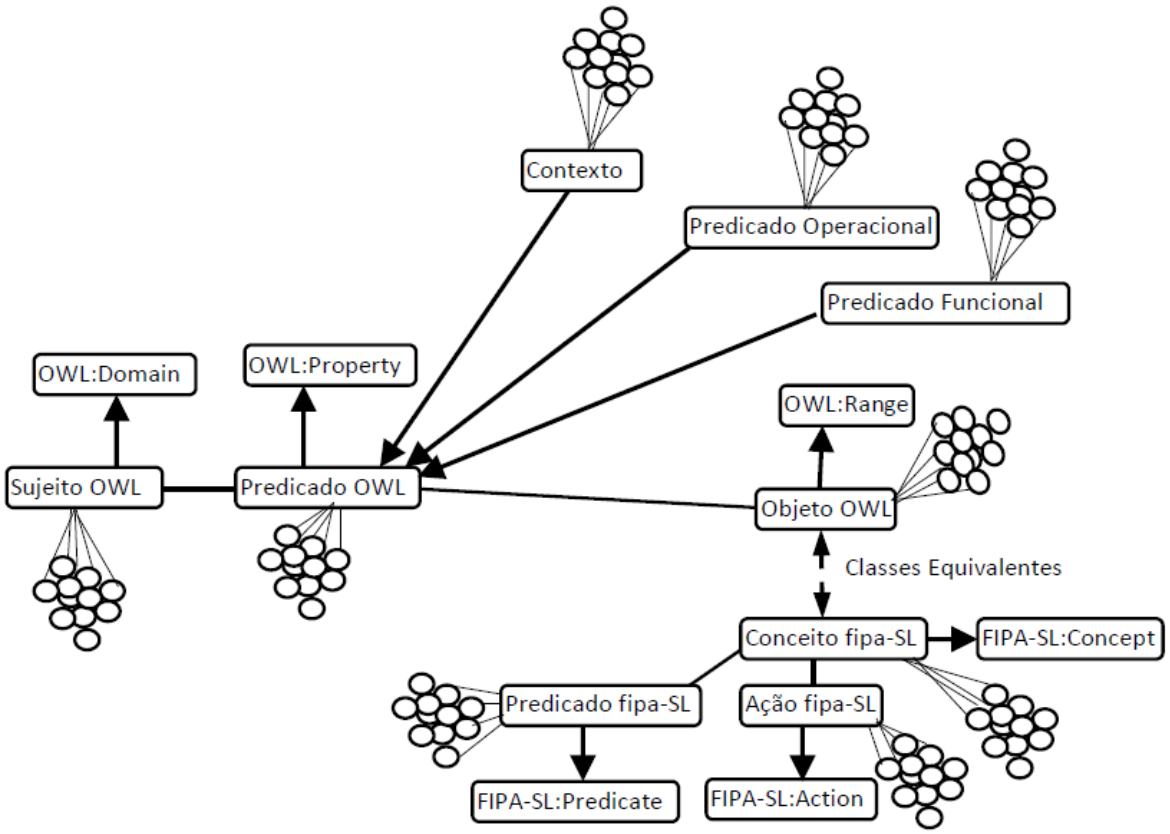
3.3.5 Ontologia de Linguagem

A ontologia de linguagem é um dicionário que pode ser acessado pelos agentes para procurar o a sintaxe das expressões usadas nas outras ontologias. Deve se definir as palavras declaradas como: conceito, sujeito, predicado, contexto, objeto, propriedade e ação. Os conceitos foram criados com o editor de ontologias *protegé* (protegé, 2011) já que nele é gerado, automaticamente, um dicionário da sintaxe das expressões usadas nas ontologias e pode ser aproveitado como ontologia de linguagem (também são geradas automaticamente as classes JAVA). As classes em formato FIPA-SL no *protegé* podem ser transformadas nas classes JAVA e gerar o dicionário, automaticamente, usando o *plugin BeanGenerator*.

Qualquer erro nesta ontologia é facilmente encontrado como erro de sintaxe conceitual (em tempo de execução). Assim, por exemplo, se tentar usar um predicado como se fosse um conceito em alguma mensagem ou expressão (fato), os agentes capturam (catch) o erro, com a

exceção “erro de sintaxe”. A figura 23 mostra uma visão global da ontologia de linguagem usada no protótipo do sistema de treinamento virtual proposto.

Figura 23 – Ontologia de linguagem do protótipo



Fonte: Autor (2014).

3.3.6 Modelo de Aluno

O modelo de Aluno representa um conjunto de perfis de todos os alunos que usam o sistema. Cada perfil está representado por instancias dinâmicas (ou seja, que o agente tutor pode modificar). Contém um perfil de desempenho do aluno, um perfil emocional e um histórico dos treinamentos anteriores.

O perfil emocional do aluno é baseado no modelo emocional OCC descrito anteriormente. As valências positivas e negativas são focadas nos outros (neste caso o aluno, já que as próprias emoções correspondem ao agente). As linhas tracejadas representam um eixo imaginário em que um estado emocional muda de um extremo para outro (p.ex. de vergonha para orgulho e vice-versa). As linhas com seta representam predicados de fluxo de informação

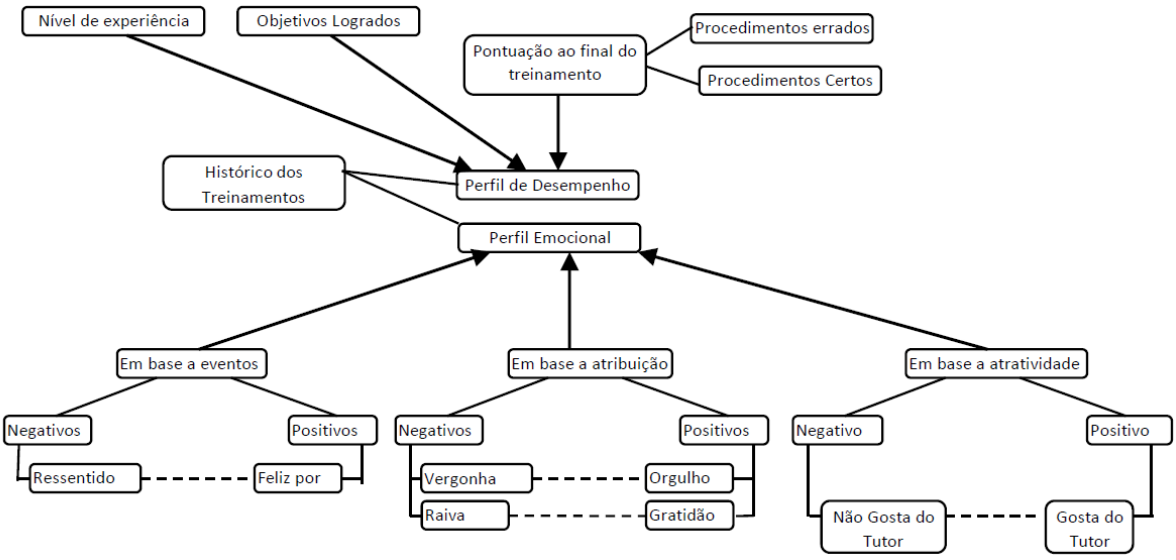
e indicam como os dados do perfil de desempenho e do perfil emocional são recolhidas quando o agente acessa o modelo do aluno.

No final de cada treinamento o agente modifica o perfil do aluno atualizando o perfil emocional e perfil de desempenho. A modificação é registrada no histórico dos treinamentos.

Quando um agente acessa o modelo de aluno em um novo treinamento, as informações do perfil emocional e do perfil de desempenho serão correspondentes ao último treinamento realizado pelo aluno. Se o agente precisar de maiores informações pode consultar o histórico dos treinamentos.

Os históricos dos treinamentos é um repositório de objetos que encapsulam um formulário XML com as informações de todos os treinamentos feitos pelo aluno. A figura 24 mostra a estrutura do modelo de aluno.

Figura 24 – Ontologia de modelo do aluno



Fonte: Autor (2014).

3.3.7 Ontologia de Gerenciamento

A base de conhecimento de Gerenciamento representa os protocolos dos agentes de (comunicação, resolução de conflitos e comportamento), gerenciamento do sistema e modelo emocional dos agentes.

O gerenciamento do sistema define a hierarquia e privilégios dos agentes na interação com os artefatos e bases de conhecimento do sistema. O agente tutor toma as decisões finais e resolve os conflitos.

Os privilégios do agente tutor são os seguintes:

a) Pode modificar o quadro negro para controlar o veículo do simulador externo;
b) Pode modificar o modelo do aluno;
c) Pode interagir com o operador e com o aluno por meio das interfaces;
d) Tem acesso a todas as bases de conhecimento;
e) Pode mandar sobre os avaliadores.

Os agentes avaliadores estão ativos apenas na fase de treinamento e seus privilégios são bem limitados:

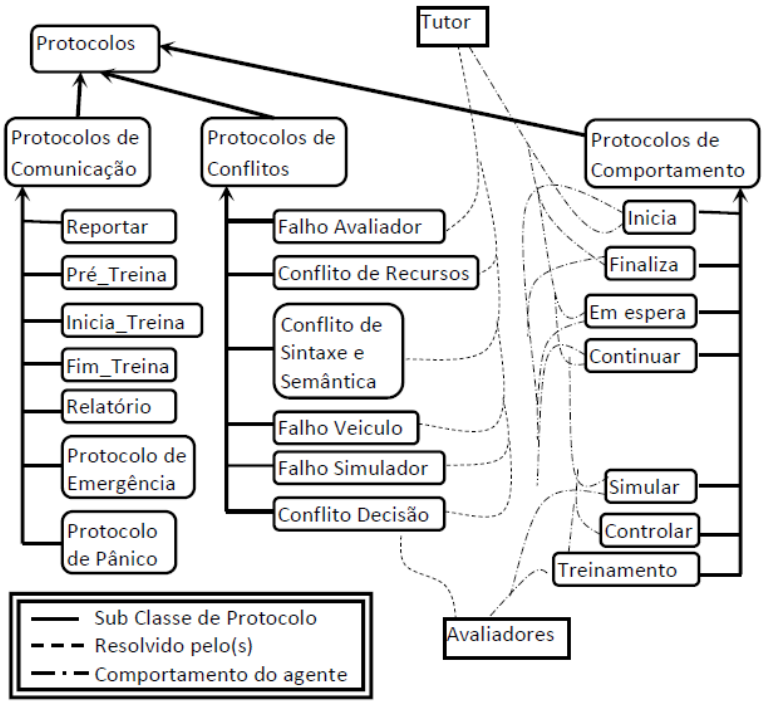
a) Apenas podem ler a base de conhecimento do seu domínio correspondente;
b) Apenas podem ler o quadro negro;
c) Apenas podem acionar o processador de modelo para executar simulações.

A ontologia de gerenciamento está constituída em três camadas ou níveis, sendo elas: a camada de protocolos, a camada de privilégios e a camada emocional. A camada de protocolos representa três tipos de protocolos, sendo eles: os protocolos de comunicação, os protocolos de conflitos e os protocolos de comportamento.

A figura 25 ilustra uma visão global da camada de protocolos. A camada de privilégios representa os privilégios de acessibilidade e permissões.

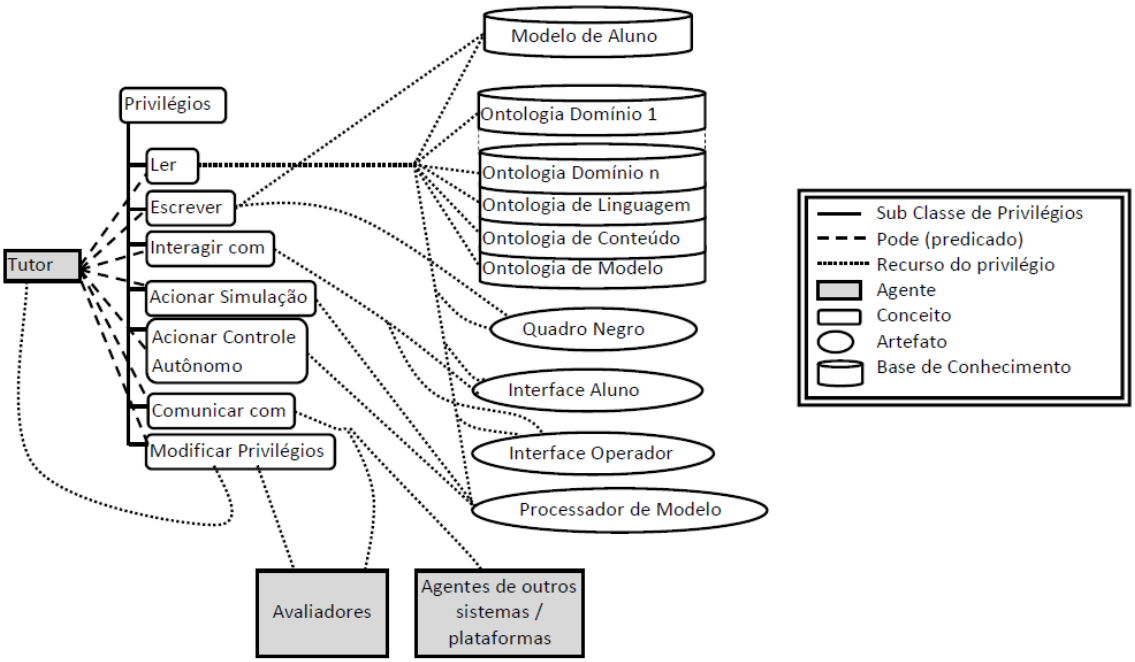
A figura 26 ilustra uma visão global da camada de privilégios do agente tutor. A figura 27 ilustra a camada de privilégios de cada agente avaliador (especialista no domínio $\boldsymbol{m}$).

Figura 25 – Camada de Protocolos da ontologia de gerenciamento



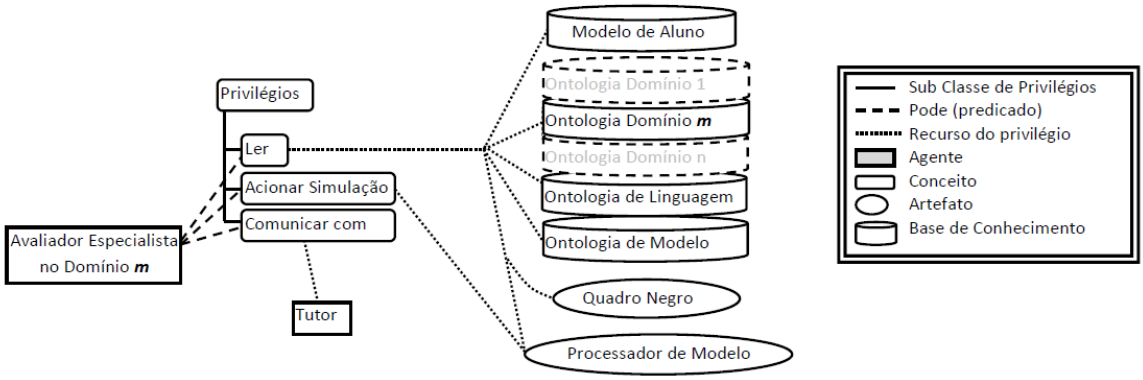
Fonte: Autor (2014).

Figura 26 – Camada de privilégios da ontologia de gerenciamento



Fonte: Autor (2014).

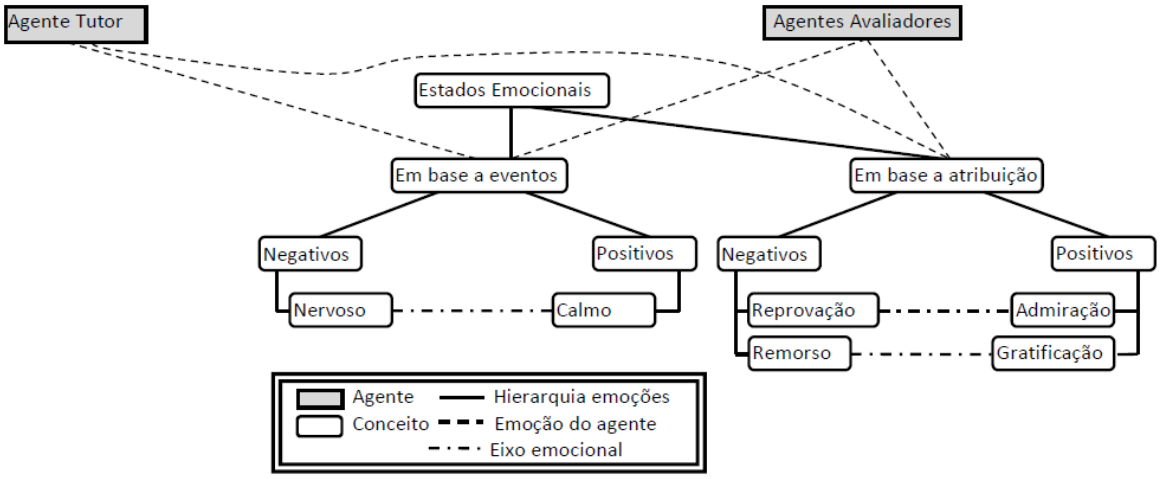
Figura 27 – Camada de privilégios de cada agente



Fonte: Autor (2014).

Os elementos artefatos e bases de conhecimento não pertencem à ontologia de gerenciamento, mas são representados na mesma ontologia como referências aos recursos. Finalmente, uma visão global da camada de estados emocionais é mostrada na figura 28.

Figura 28 – Estados emocionais dos agentes na ontologia de gerenciamento



Fonte: Autor (2014).

Considerar a função desejabilidade $D(p,e,t)$ de um evento (e) para uma pessoa (p) num instante (t). Esta função possui valência positiva para eventos desejáveis, e valência negativa para eventos indesejáveis. Considerar também a função intensidade global $I_g(p,e,t)$, a função de estado potencial de felicidade $P_j(p,e,t)$ e a função de disparo $T_j(p,t)$.

As emoções dos agentes são estimadas com o seguinte algoritmo:

IF $D(p,e,t) > 0$
THEN set $P_j(p,e,t) = f_j(D(p,e,t), I_g(p,e,t))$
IF $P_j(p,e,t) > T_j(p,t)$
THEN set $I_j(p,e,t) = P_j(p,e,t) - T_j(p,t)$
ELSE set $I_j(p,e,t) = 0$

Os protocolos dos agentes são apresentados no APÊNDICE A.

4 PROTORIPAGEM

Neste capítulo descreve-se o processo de desenvolvimento do protótipo. O domínio escolhido para o protótipo foi o treinamento para pilotos de aviões Boeing 737-800, usando o simulador de voo XPLANE9. O sistema de instrução virtual opera independente do simulador e assume o papel de um instrutor e avaliador de voo humano. O desenvolvimento do protótipo foi feito em duas fases.

A primeira fase no desenvolvimento do protótipo consistiu em desenvolver um framework em Java, baseado no sistema proposto anteriormente e foi nomeado “JEMO”.

Na segunda fase foi construído o protótipo aplicando a partir das bibliotecas do framework JEMO.

4.1 FRAMEWORK JEMO

O *framework* JEMO é um conjunto de classes em Java que incluem *templates* (moldes para preencher com códigos) das funcionalidades do sistema proposto e foi construído com o propósito de facilitar a programação e estruturação do protótipo e, também, de disponibilizar uma ferramenta aos projetistas no desenvolvimento de sistemas similares. Assim, um projetista que deseje desenvolver um sistema de instrução inteligente virtual que envolva veículos, máquinas ou processos pode utilizar o framework. As principais considerações que deve ter na hora de usar o framework JEMO são:

a) O framework trabalha em conjunto com a plataforma de multiagentes JADE (ambiente onde os agentes trabalham). JADE é um conjunto de bibliotecas (livres) que permite gerenciar os agentes e segue todos os protocolos e normas da fundação FIPA;
b) O framework JEMO permite comunicação local e externa por meio dos protocolos disponíveis para TCP/IP e UDP;
c) O framework foi projetado em inglês para que qualquer usuário possa utilizá-lo em outros projetos.

As classes do framework JEMO estão organizadas em três bibliotecas sem raiz comum. A primeira é “*Jemo.jar”* que contém as classes principais da estrutura do sistema de

treinamento. Inclui uma plataforma de comandos para configurar e fazer testes de verificação no projeto da aplicação e nas bases de conhecimento.

A segunda biblioteca é "*JemoBroker.jar"*, que contém as classes que implementam os artefatos do sistema e os slots que suportam as bases de conhecimento. Os artefatos que implementa são: o quadro negro, o processador de dinâmica, a interface do aluno e a interface do operador (inclui classes para definir novos artefatos). Os slots de suporte de bases de conhecimento são: modelo de aluno, ontologia de conteúdo, ontologia de domínio, ontologia de modelo e ontologia de linguagem.

A terceira biblioteca é "*JemoAgent.jar"*, que contém as classes que representam a estrutura e as funcionalidades internas dos agentes (Percepção, Memória Declarativa, Memória Procedural, Memória de Trabalho, Decisão, Ações). Inclui uma classe "*JemoJADE.jar"* para criar e gerenciar um agente na plataforma JADE mantendo a estrutura interna do agente apresentada na proposta de sistema virtual.

4.2 ESPECIFICAÇÕES DO PROTÓTIPO

As especificações de protótipo foram obtidas usando a metodologia proposta no livro "Projeto Integrado de Produtos" (BACK et. al. 2008). Esta metodologia permite organizar as especificações por prioridades (na ordem que apresentam maior risco ao sucesso do projeto, caso as especificações não sejam atendidas).

A primeira prioridade é a pedagogia adaptativa do sistema. O sistema deve acompanhar em todo momento a aprendizagem do aluno (se adaptando ao comportamento, desempenho e perfil do aluno). A forma de verificar este requisito é observando o comportamento, estratégias de ensino e de avaliação do sistema em função do desempenho e comportamento do aluno. Se este requisito não for atendido, perde-se o paradigma de instrutor virtual e o cliente pode rejeitar o produto.

A segunda prioridade é a iteração natural entre sistema e aluno considerando a linguagem natural nas interfaces, simulação de emoções no sistema e detecção de emoções do aluno (pelo sistema). A forma de verificar este requisito é observando o comportamento do sistema em função ao comportamento (e emoções) do aluno, e observando a linguagem entre agentes e a linguagem nas interfaces (devem ser naturalmente entendidos pelos usuários humanos). Se o requisito não for atendido, perde-se o paradigma de vínculo afetivo inconsciente

entre o aluno e o sistema, e a qualidade dos treinamentos ao longo prazo podem se vir negativamente afetados.

A terceira prioridade é a usabilidade do sistema para os usuários. As interfaces devem ser intuitivas e fáceis de usar. A comunicação com o sistema deve ser naturalmente entendida pelos usuários. A forma de verificar este requisito é por meio de validações feitas por alunos de escolas de aviação. Se o requisito não for atendido o sistema pode ser rejeitado pelo cliente.

A quarta prioridade é a confiabilidade do sistema. O sistema deve ser robusto, com baixa taxa de falhas e conflitos (se acontecerem o sistema deve suportar esses problemas). Este requisito deve ser comprovado por meio de verificações e validações. Se o requisito não for atendido o sistema pode ser rejeitado pelo cliente.

A quinta prioridade é construir o protótipo usando ferramentas de desenvolvimento independentes de hardware, sistema operacional e linguagem (lidar com diferentes linguagens entre diferentes sistemas), bem desenvolvidas e evoluídas, bem documentadas, facilmente acessíveis e frequentemente atualizadas, e, no possível, ser normalizadas por organismos internacionais. A forma de verificar este requisito é observar a operação transparente entre diferentes sistemas (simulador, interfaces, instrutor virtual, servers de serviços remotos, linguagens e hardware, computadores, celulares, tablets, etc.). Se o requisito não for atendido, o projeto pode se ver retrasado por um tempo considerável (tendo em consequência uma perda considerável de recursos) e terminar, sendo limitado na interoperabilidade e o protótipo pode terminar sendo obsoleto em um curto prazo. A evolução do protótipo na etapa de desenvolvimento pode ser mais demorada e difícil e, possivelmente, não distribuir o sistema remotamente.

A sexta prioridade é contar com a disponibilidade e disposição dos especialistas e participantes das validações. A forma de verificar este requisito é observar se as agendas das entrevistas e validações são cumpridas, e se o trato e entendimento com os participantes é bom ou ruim. Se o requisito não for atendido, o desenvolvimento do protótipo pode ser retrasado consideravelmente.

A sexta prioridade é a reusabilidade do modelo e do framework, permitindo aos projetistas desenvolver novos sistemas de instrução virtual com simuladores (com veículos, máquinas ou processos). A forma de verificar o requisito é observar ao longo prazo novos projetos baseados no modelo e/ou framework. Se o requisito não for atendido, o modelo e o framework podem terminar sendo obsoletos.

4.3 ESTRUTURA DO PROTÓTIPO

O sistema de treinamento virtual foi projetado para operar em conjunto com o simulador de voo “Xplane9” por meio de conexão TCP/IP local. O protótipo consiste em dois computadores: no primeiro roda o simulador de voo “Xpane9” em conjunto com a interface do aluno. No segundo, computador roda o sistema virtual de treinamento (JEMO). O computador que executa o simulador XPlane-9 conta com periféricos que reproduzem alguns dos mandos de um Boeing (manche, pedais e trhottle), como mostra a figura 29.

No computador da esquerda se executa o simulador e a interface de aluno. Na esquerda se executa o sistema tutor virtual e na direita a interface de operador.

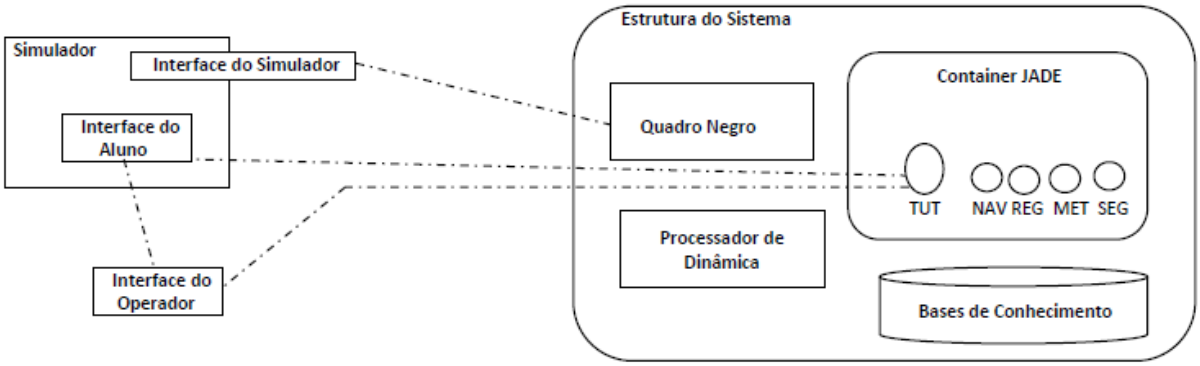
.

Figura 29- Protótipo

Fonte: Autor (2014).

A interface do operador foi implementada em um computador remoto (laptop de testes) e interage com a interface de aluno no computador onde roda o simulador e, ao mesmo tempo, interage com o agente Tutor que opera no sistema de treinamento virtual, como mostrado na figura 30. As linhas tracejadas representam a comunicação remota entre computadores.

Figura 30- Estrutura do protótipo



Fonte: Autor (2014).

O sistema conta com uma plataforma que suporta o container dos agentes (plataforma JADE, que atende as especificações da FIPA). Conta também com slots que suportam as bases de conhecimento, o Quadro Negro, o Processador de Dinâmica, e suporte às comunicações TCP/IP, com as interfaces de Aluno, de Operador e de Simulador. O protótipo conta com cinco agentes: um agente Tutor (TUT) e quatro agentes Especialistas:

a) NAV (Agente especialista em navegação aeronáutica);
b) MET (Agente especialista em meteorologia);
c) REG (Agente especialista no regulamento aeronáutico);
d) SEG (Agente especialista em segurança no voo e procedimentos de emergência).

O quadro negro foi construído diretamente do framework “JEMO” e foi configurado para ler e escrever as variáveis do simulador. O processador de dinâmica também foi construído diretamente do framework “JEMO” e opera com a ontologia de modelo.

4.4 INTERFACE DO SIMULADOR

A interface do simulador foi feita em C++, como um plugin que se executa dentro do simulador X-Plane. O objetivo do plugin é ler e escrever as variáveis e comandos de controle do simulador, depois tabular cada item em uma planilha em formato XML. Cada variável é marcada com uma cabeceira correspondente a posição da tabela da planilha. O fluxo de dados (variáveis marcadas com uma cabeceira) é conectado remotamente com o protótipo via socket (TCP/IP). No protótipo o fluxo de dados é conectado via socket (TCP/IP), com o Quadro Negro do sistema. No Quadro Negro as variáveis são ordenadas em uma tabela na posição indicada na

cabeceira (depois as cabeceiras são removidas das variáveis). Quando o agente Tutor ou o Processador de Modelo escreve no Quadro negro, a variável é posicionada em um formulário, marcada com uma cabeceira de posição e enviada para interface do simulador via socket. A variável ou comando é alocado no formulário da interface (e a cabeceira é removida), e, finalmente, é escrita dentro do simulador. A interface do simulador foi configurada como artefato com as funcionalidades de login, privilégios e atributos de uso.

4.5 INTERFACE DE ALUNO

A interface de aluno foi implementada em JAVA e se comunica com a interface de operador e com o agente Tutor por meio de sockets. A interface é mostrada na mesma tela do simulador (no canto superior), como mostrado na figura 31. A comunicação com o aluno é por chat (o aluno pode conversar com o agente Tutor utilizando frases e palavras predefinidas). No mesmo diálogo por chat, o aluno pode conversar com o operador externo. O agente Tutor apresenta o conteúdo pedagógico por meio de imagens, texto e animações de alerta. Na tela do simulador tem uma janela transparente de assistência, onde o aluno pode pedir ajuda ao agente Tutor, ou o Tutor pode oferecer ajuda ao aluno se perceber que há dificuldades na sessão. Em projetos futuros, a comunicação entre o agente tutor e o aluno pode ser feita por uma interface de voz (usando fones e microfones).

Figura 31- Interface do aluno executada no canto esquerdo do simulador.

Fonte: Autor (2014).

Na parte superior da figura mostra uma imagem ou animação do slide. Na parte central mostra o conteúdo do slide (principalmente orientações). Na parte inferior da interface tem um chat para o aluno poder se comunicar com o operador humano. No lado direito, na tela do

simulador, existe a interface de assistência do agente Tutor, onde é oferecida ajuda ao aluno em caso de dificuldades. O aluno também pode pedir ajuda clicando no botão "AJUDA".

4.6 INTERFACE DE OPERADOR

A interface foi desenvolvida em JAVA e se comunica com a interface de aluno e com o agente Tutor por meio de sockets. A interface contém um grupo de objetivos que podem ser selecionados; um formulário, onde pode preencher as condições do treino, e uma janela de chat para conversar com o aluno. Na interface, o operador pode visualizar e modificar o estado emocional dos agentes e pode, também, visualizar o estado emocional (detectado) do aluno mapeado no modelo de aluno. O operador pode utilizar comandos definidos na janela de chat para diferentes funções, como visto no quadro 4:

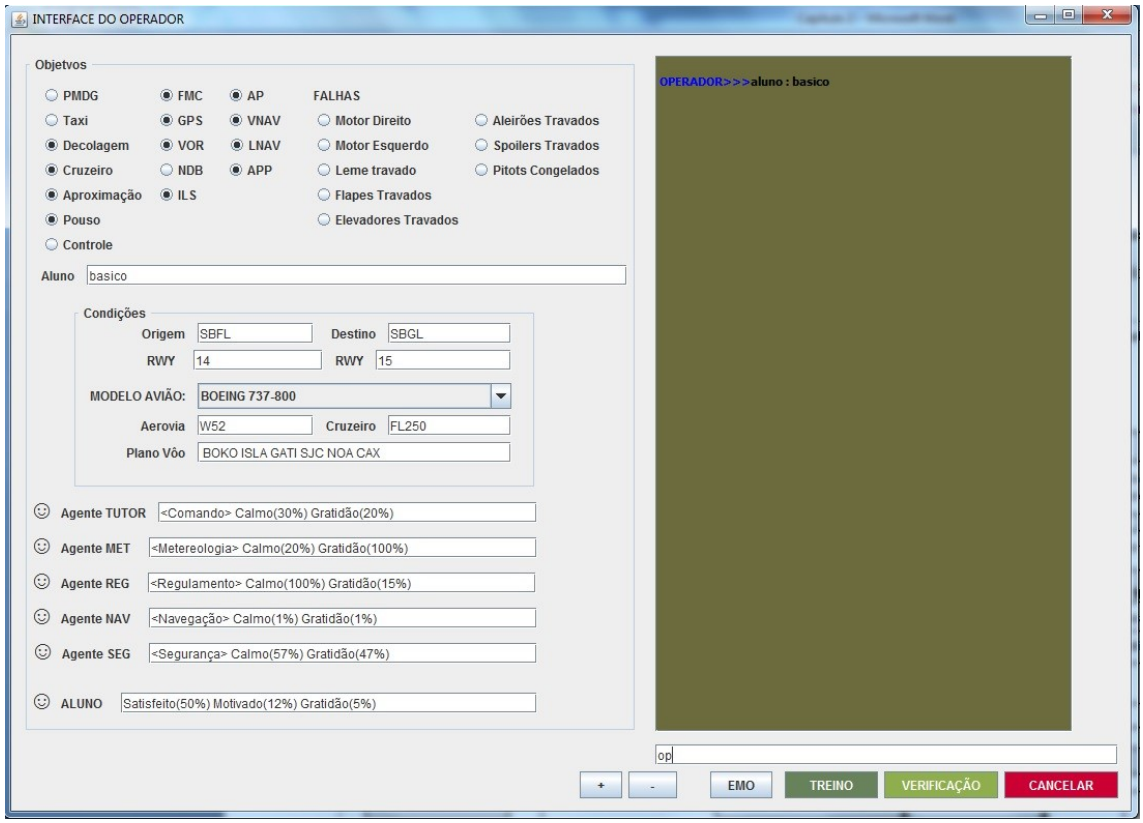
Quadro 4 – Comandos da interface de operador

/perfil<nome do aluno> para referenciar o aluno que vai ser treinado. */novo perfil<nome do aluno>* para criar um novo perfil de aluno. */aborta* para abortar treino/verificação. */relatório* para abrir uma janela e selecionar o relatório que quer ler. */<variável>=<valor>* sobrescreve uma variável no quadro negro (pode configurar condições meteorológicas e estados no avião). */ST<nome agente>,<valor>* sobrescreve o ST(tolerância ao estresse) do agente. ST pode ter valores reais entre 0.00 e 10.00 (ST=0 para um agente muito calmo, ST=10 para um agente facilmente irritável).

Fonte: Autor (2014).

Na interface de operador mostrada na figura 32 podem se configurar os objetivos, as condições do treinamento, os estados emocionais iniciais dos agentes. O operador pode usar o chat para conversar com o aluno no simulador ou para escrever comandos de controle de sistema.

Figura 32 - Interface de operador



Fonte: Autor (2014).

4.7 AQUISIÇÃO DE CONHECIMENTO

Os sistemas baseados em conhecimento têm como filosofia principal captar o conhecimento e a experiência do humano especializado para resolver problemas em domínios específicos. Por isso, é importante insistir em evitar representar conhecimentos baseados unicamente em pesquisas bibliográficas. Consistiu em recopilar conhecimento de diferentes fontes, principalmente, dos especialistas. Outras fontes de conhecimento foram cursos online e cursos presenciais extracurriculares (entre eles um curso de piloto privado). A aquisição de conhecimento foi feita por meio de uma série de entrevistas, com diferentes especialistas (pilotos comerciais, técnicos de manutenção de aviões, professores de escolas de aviação, etc.). Foi necessário que, a cada certo tempo, os especialistas avaliem (com testes, questionários, etc.) se está havendo um bom entendimento dos conteúdos dos domínios. Depois de todas as entrevistas, todos os conhecimentos adquiridos, foram organizados e separados por domínios e subdomínios. Cada subdomínio foi marcado (referenciado) com um contexto. O método de coleta de dados foi totalmente adquirido com dados qualitativos cardinais e ordinais.

O domínio de interesse é o avião de passageiros BOEING 737-800 (BOEING, 1997) e todos os tópicos que envolvem a operação do avião, sendo eles:

a) Operação da aeronave;
b) Procedimentos de voo;
c) Teoria de voo;
d) Navegação;
e) Regulamento aeronáutico;
f) Meteorologia;
g) Segurança no voo;
h) Comportamento dinâmico, sistemas e subsistemas do avião.

4.8 REPRESENTAÇÃO DE CONHECIMENTO

Esta foi a fase mais demorada no desenvolvimento do protótipo. O objetivo desta etapa foi construir as bases de conhecimento do sistema (ontologias e regras). No início, foi muito frequente não ter considerado alguns aspectos dos domínios de conhecimento nas entrevistas com os especialistas ou errar no modo de organizar as regras e ontologias, gerando erros frequentes de sintaxe e semântica, obrigando a revisar a aquisição de conhecimento (novas entrevistas) e reorganizar o conhecimento.

Foi necessário verificar periodicamente e ciclicamente as bases de conhecimento, conforme elas evoluíam em forma incremental durante todo o tempo de desenvolvimento do sistema. De um ponto de vista da engenharia de software, foi aplicado uma representação incremental.

As bases de conhecimento são o núcleo da funcionalidade e efetividade do sistema, pois definem o gerenciamento do sistema e dos agentes, e definem o conhecimento a ser manuseado para ensinar e avaliar as sessões de treinamento, e definem o conhecimento para controlar remotamente o avião do simulador. As bases de conhecimento são classificadas em três categorias.

a) Bases de conhecimento para Treinamento: Estas bases representam o conhecimento envolvido com o veículo do simulador externo. Estas ontologias são classificadas em quatro subcategorias:
    i. Modelo: As ontologias de modelo descrevem a dinâmica do avião e representam também uma descrição qualitativa das

partes e subpartes do sistema (maquina, veículo, processo, ambiente) e as relações entre elas. Também descrevem em forma quantitativa (com sistemas de equações) as variáveis do sistema, malhas de controle e algoritmos de resolução (referências de chamados a métodos da classe do processador de dinâmica);

ii. Domínio: As ontologias de domínio representam o conhecimento que envolve a operação do sistema representado na ontologia de modelo;

iii. Conteúdo: As ontologias de conteúdo representam o material pedagógico que o agente tutor apresenta para o aluno no treinamento;

iv. Linguagem: As ontologias de linguagem descrevem a sintaxe dos fatos processados pelos agentes e das mensagens que eles enviam e recebem. Não requer aquisição de conhecimento.

b) Bases de conhecimento para gerenciamento: elas representam as responsabilidades, protocolos, modelos emocionais e comportamentos dos agentes;

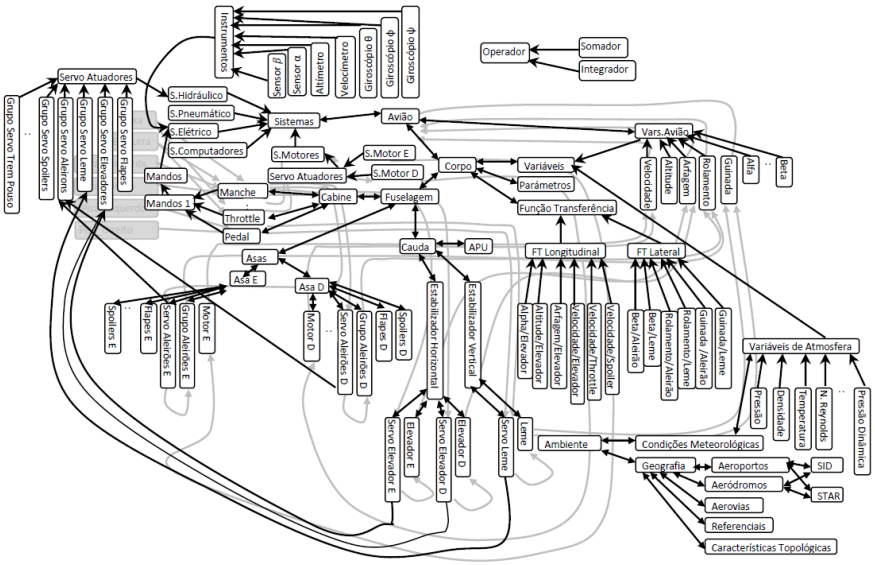
c) Base de conhecimento do Modelo do aluno: esta contém uma representação do aprendizado do aluno em treinamento; também um perfil de desempenho, perfil comportamental e histórico dos treinamentos.

4.8.1 Ontologia de Modelo

A ontologia de modelo representa o avião do simulador incluindo uma descrição estrutural, descrição de dinâmica e descrição de controle. Também, inclui uma representação do ambiente (geografia do cenário e condições meteorológicas). Esta base de conhecimento é usada pelo sistema para pilotar o avião ou para avaliar o estado do avião, ou também para procurar falhas. A ontologia de modelo do protótipo consta de 457 conceitos (entre conceitos de linguagem, sujeitos, objetos e ações) e de 1.732 predicados. Estão representados o modelo do avião e do ambiente (incluindo referências geográficas e atmosfera). A representação dinâmica foi feita com funções de transferência e malhas de controle, as quais são operadas no

processador de dinâmica por resolução de equações de estado. O grande tamanho da ontologia dificulta a representação gráfica dela, por isso são mostradas apenas as classes mais importantes. A figura 33 mostra a representação global da estrutura do modelo conformado pelas classes relacionadas pelos predicados do tipo subclasse (*é_um*) e do tipo composição (*têm<->parte_de*). As setas de um sentido indicam generalização (subclasses) e as setas de dois sentidos indicam composição (composição significa que um elemento pode apenas ser parte de um conjunto e não pode pertencer a outro, por exemplo, o motor esquerdo é parte da asa esquerda e não pode pertencer a outro conjunto).

Figura 33- Visão global das relações de generalização e composição



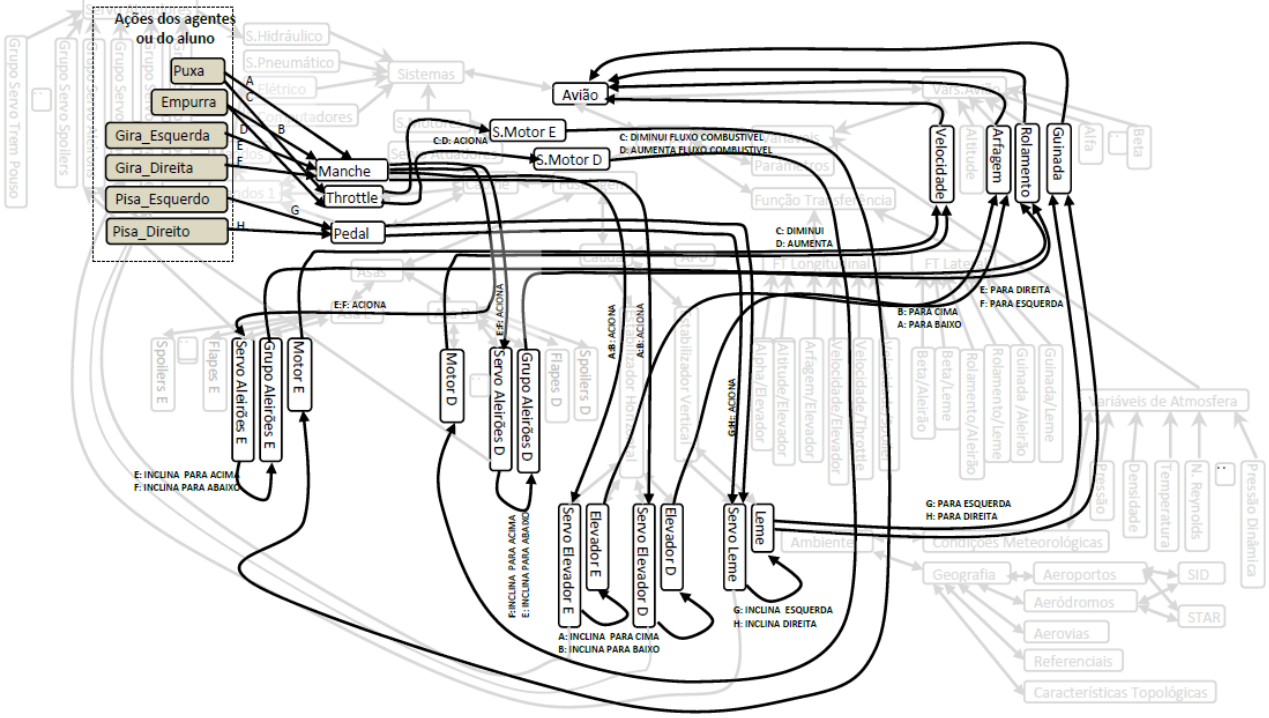
Fonte: Autor (2014).

O conceito (classe) Operador contém nos adjetivos (atributos) referências chamadas métodos da classe do módulo processador de dinâmica. As operações mais importantes são a integração e soma. Qualquer representação dinâmica de veículo, máquina ou processo pode ser conformada apenas com integradores e somadores após uma manipulação matemática.

As variáveis são representadas como conceitos (classes) ao invés de propriedades, porque é necessário relacioná-los com predicados simbólicos do tipo *aumenta, diminui, esquerda, direita, para cima, para baixo*, etc., e os atributos numéricos das variáveis são representados como predicados de operação numérica (p.ex, =100, >100, <100, etc), que podem ser interpretados pelas máquinas de inferência dos módulos do agente.

A figura 34 apresenta uma visão global dos predicados ou propriedades funcionais entre os elementos mais importantes da ontologia de modelo do protótipo. Os conceitos com preenchimento representam as ações que os agentes ou o aluno podem executar. Cada ação é representada como conceito <ação> no formato FIPA-SL. Os conceitos sem preenchimento representam os sujeitos e objetos da ontologia de modelo no formato OWL.

Figura 34- Visão global das propriedades funcionais da ontologia de Modelo.



Fonte: Autor (2014).

Cada ação é associada a uma TAG ou etiqueta que se mantém em todos os termos do encadeamento funcional. Assim, por exemplo, a relação entre a ação <pisa_esquerdo> e o objeto <Pedal> é associada á TAG "G". Quando a ação for executada, o encadeamento funcional é gerado:

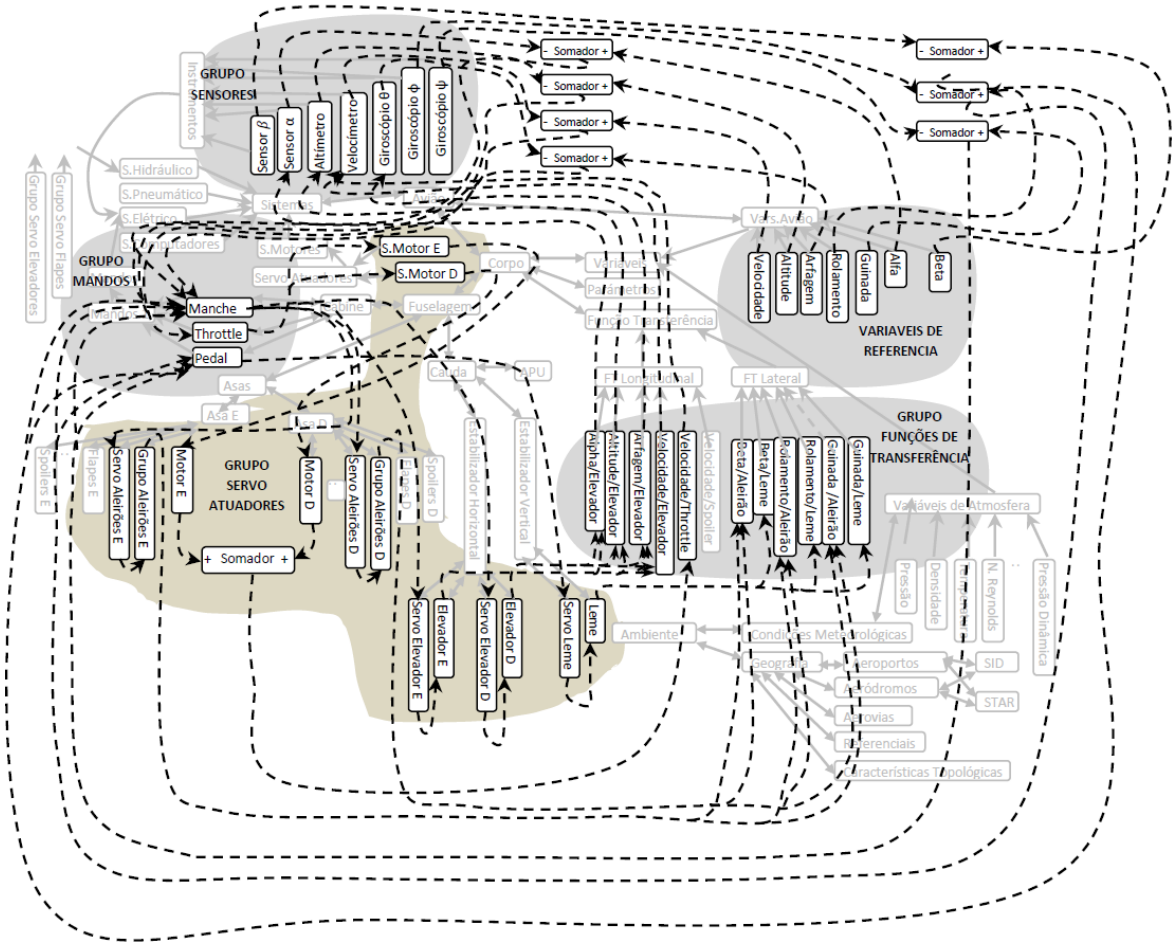
<pisa_esquerdo(Pedal)> =>

<G:Aciona(Servo_Leme)> -> <G:Inclina_esquerda(Leme)>

-> <G:para_Esquerda(Avião,Guinada)

A figura 35 mostra as relações operacionais de dinâmica do modelo (predicados de controle).

Figura 35- Visão global das relações operacionais da ontologia de modelo

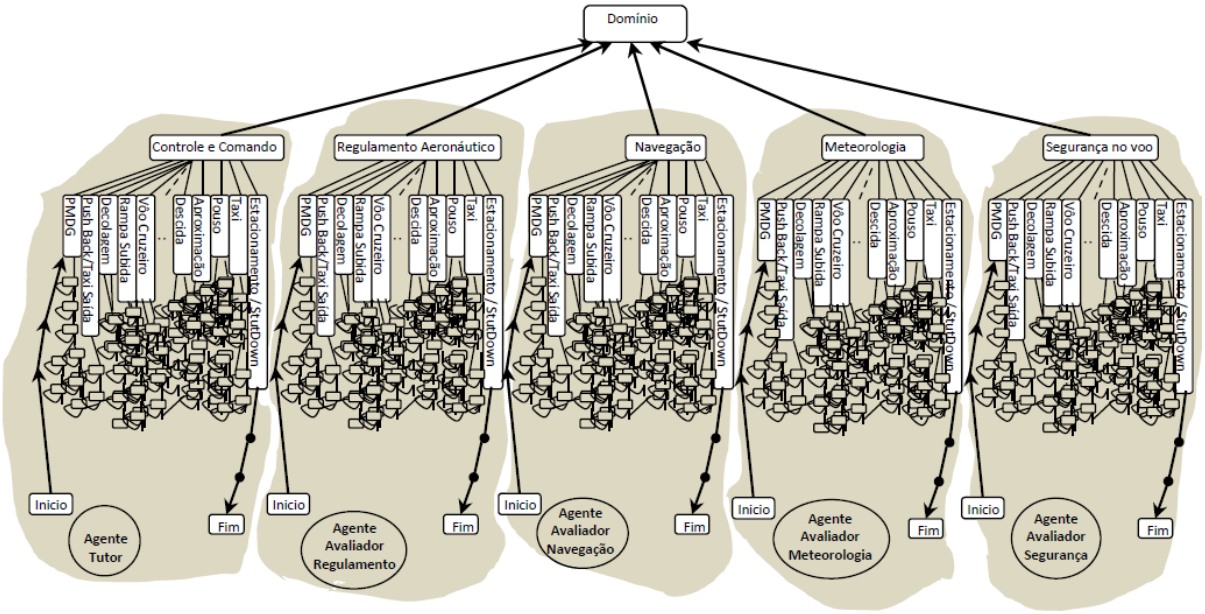


Fonte: Autor (2014).

4.8.2 Ontologia de Domínio

A ontologia de Domínio representa o conteúdo pedagógico que o aluno deve aprender; esta foi representada com cinco domínios (comando e controle, regulamento aeronáutico, navegação, meteorologia e segurança no voo) e 14 contextos em cada domínio (PMDG, push back/taxi saída, decolagem, subida, voo cruzeiro, descida, aproximação, circuito de espera, go around, emergência técnica e emergência de comunicação, pouso, táxi, estacionamento/ShutDown). A figura 36 apresenta uma visão global da ontologia.

Figura 36- Visão global da ontologia de Domínio.



Fonte: Autor (2014).

O domínio de comando e controle é processado pelo agente Tutor e representa o conhecimento adquirido nos seguintes tópicos:

a) Painéis da cabine do avião incluindo mandos, instrumentação, navegação e comunicação;
b) Procedimentos em condições normais em emergências;
c) Teoria de voo.

O domínio de regulamento é processado pelo agente especialista em Regulamento aeronáutico e representa o conhecimento adquirido nos seguintes tópicos:

a) Regulamento aeronáutico; e
b) Procedimentos (condições normais).

O domínio de navegação é processado pelo agente especialista em Navegação e representa o conhecimento adquirido nos seguintes tópicos:

a) Navegação; e
b) Procedimentos (condições normais).

O domínio de meteorologia é processado pelo agente especialista em Meteorologia e representa o conhecimento adquirido nos seguintes tópicos:

a) Meteorologia. e

b) Procedimentos (condições normais).

O domínio de segurança no voo é processado pelo agente especialista em Segurança e representa o conhecimento adquirido nos seguintes tópicos:

a) Procedimentos de segurança; e
b) Procedimentos de emergência.

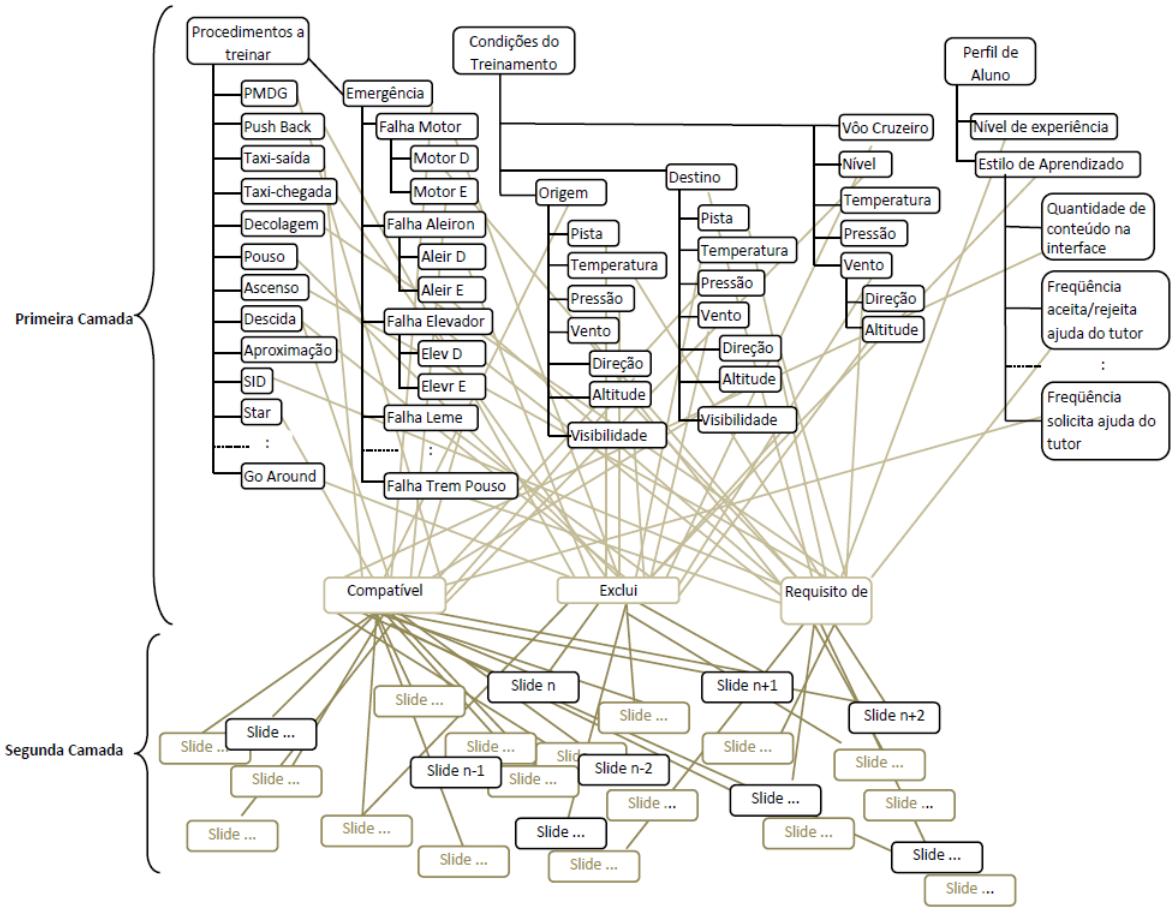
4.8.3 Ontologia de Conteúdo

A ontologia de conteúdo representa o material pedagógico que o agente tutor apresenta ao aluno na interface. A figura 37 apresenta uma visão global da ontologia de conteúdo. A primeira camada é um grupo de conceitos que representam os objetivos, condições (rotas, ambiente, etc.) e qualidades esperadas no nível aluno. A primeira camada mapeia a compatibilidade, exclusividade e dependência entre os objetivos e condições do treinamento, e o perfil do aluno.

O mapeamento da primeira camada converge em um conjunto de elementos da segunda camada (camada de slides) numerados em sequência. A saída da segunda camada é uma lista de slides ordenados em sequência e um roteiro de navegação (resultante das cidades, aeroportos, referencias, etc.), mapeados no modelo de ambiente da Ontologia de Modelo.

Cada roteiro ou lista de slides e roteiro de navegação são registrados em formulários XML, e logo serializados e encapsulados em objetos. Estes objetos representam experiências de treinamento e são guardados num container. As experiências são reusadas cada vez que em um novo treinamento os objetivos, condições e perfil de aluno são compatíveis.

Figura 37- Visão global da ontologia de conteúdo.



Fonte: Autor (2014).

4.8.4 Modelo de Aluno e Bases de Conhecimento de Gerenciamento

O Modelo de Aluno e as ontologias de gerenciamento usadas no protótipo têm a mesma estrutura apresentada na seção “Proposta de sistema de treinamento virtual”. Estas bases de conhecimento podem se considerar uma espécie de "*shell*”, já que podem ser reusadas em qualquer outro projeto de desenvolvimento de sistemas de instrução virtual de qualquer veículo, máquina ou processo usando qualquer tipo de simulador.

4.8.5 Sessão de Treinamento

Toda sessão começa quando o operador humano configura a Interface de Operador. O operador pode conversar com o aluno antes e durante do treinamento por meio do chat (evitando

usar o símbolo "/", reservado para comandos de configuração e teste). A configuração na Interface de Operador é feita em quatro passos:

Configurar os objetivos do treinamento, selecionando na secção de Objetivos ou escrevendo na linha de comando do chat "/objetivo: +" seguido do objetivo.

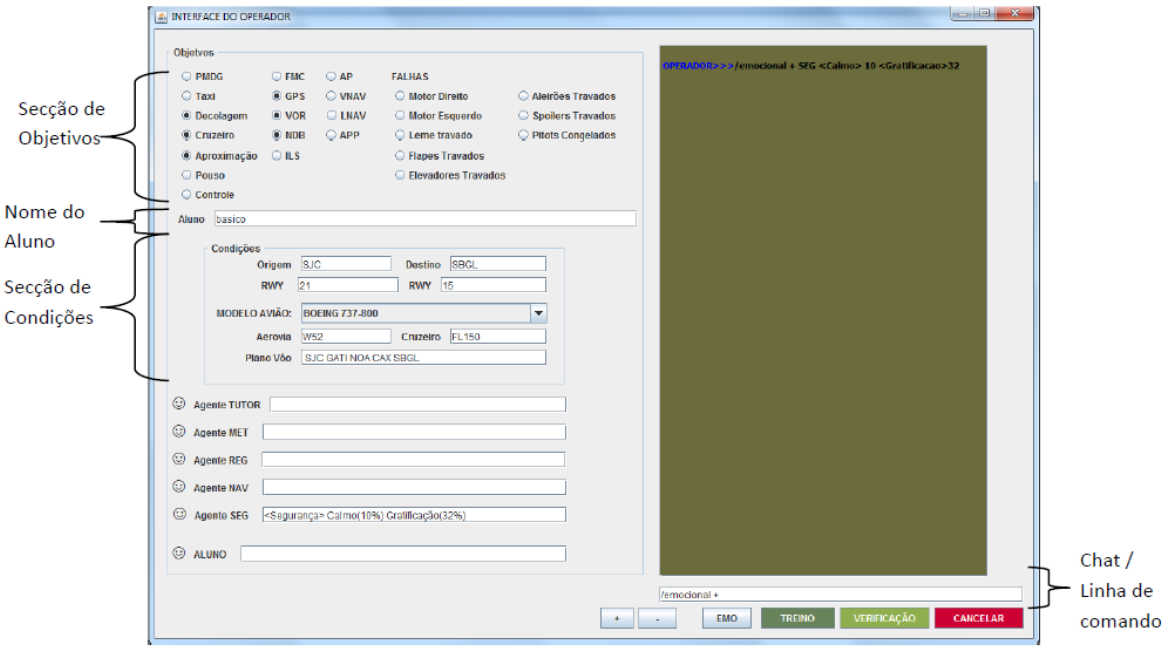
Configurar as condições do treinamento, preenchendo as caixas da seção das Condições ou escrevendo na linha de comando do chat "/cond: +" seguido da condição.

Escrever o nome do aluno que irá ser treinado. Se o aluno for novo, pode ser cadastrado por meio da linha de comando no chat (por exemplo, *"/cadastra: + Claudio <nivel> basico <experiencia> sem*").

Configurar os parâmetros e estados emocionais iniciais dos agentes por meio da linha de comando no chat (por exemplo, "*/emocional: + NAV <Normal> 60*").

A figura 38- mostra uma tela da interface do operador. O treinamento é iniciado clicando no botão "TREINO". Uma consola de testes pode ser invocada clicando no botão "VERIFICAÇÃO". Clicando no botão "EMO" podem se visualizar os parâmetros emocionais e comportamentais dos agentes e quais bases de conhecimento eles vão manusear. Clicando no botão "CANCELAR" pode interromper uma sessão de treinamento, podendo reiniciá-la ou cancelá-la definitivamente na linha de comando.

Figura 38- Interface de Operador



Fonte: Autor (2014).

Uma vez iniciado o treinamento, o aluno visualiza na tela uma mensagem de início de treinamento, mostrando os objetivos e condições da sessão. Se houver qualquer dúvida, o aluno pode conversar com o operador por meio do chat na Interface de Aluno. Na figura 39 pode se ver a mensagem de início de treinamento.

Figura 39 – Mensagem de início de treinamento

Fonte: Autor (2014).

Uma vez que o aluno termina de ler a mensagem e clica no botão "OK" é mostrada na Interface de aluno uma sequência de slides, que avança à medida que o aluno vai executando os procedimentos solicitados, como mostrado na figura 40. Na parte direita da tela, tem se uma janela transparente, onde o aluno pode pedir ajuda ao Tutor em caso de dificuldades, clicando no botão "AJUDA".

Figura 40- Interface do Aluno na tela do simulador.

Fonte: Autor (2014).

Quando o agente Tutor percebe que o aluno erra vários procedimentos, ele oferece ajuda ao aluno na janela de assistência. O aluno pode aceitar ou rejeitar a ajuda, como mostrado na figura 41. Se aceitar, o agente orienta passo a passo o que fazer e como fazer, até o aluno completar o procedimento.

Figura 41- Assistência ao aluno.

Fonte: Autor (2014).

O aluno pode optar também por desligar o serviço de assistência. Neste caso, o Tutor dá apenas as orientações sobre o treinamento.

Além de orientar o aluno por meio de mensagens no chat, o agente Tutor mostra sinais visuais na cabine para indicar qual instrumento visualizar, qual *mando* mexer, e qual botão girar ou apertar, como mostrado na figura 42.

Figura 42- Assistência visual

Fonte: Autor (2014).

Normalmente, se o avião se acidentar, o aluno tem que reiniciar o treinamento, o que significa uma grande perda de foco no objetivo do treino e de tempo. Por isso, o agente Tutor fica atento nas situações críticas e assume o controle do avião quando a possibilidade de catástrofe é iminente, como mostrado na figura 43. O que o aluno vê na interface é um aviso para deixar os mandos. Se o agente não conseguir resolver a situação, ele entra em um estado de pânico e aciona o Processador de Dinâmica para tomar o controle do avião e tentar recuperá-lo.

Figura 43 - Tutor assumindo o controle da aeronave

Fonte: Autor (2014).

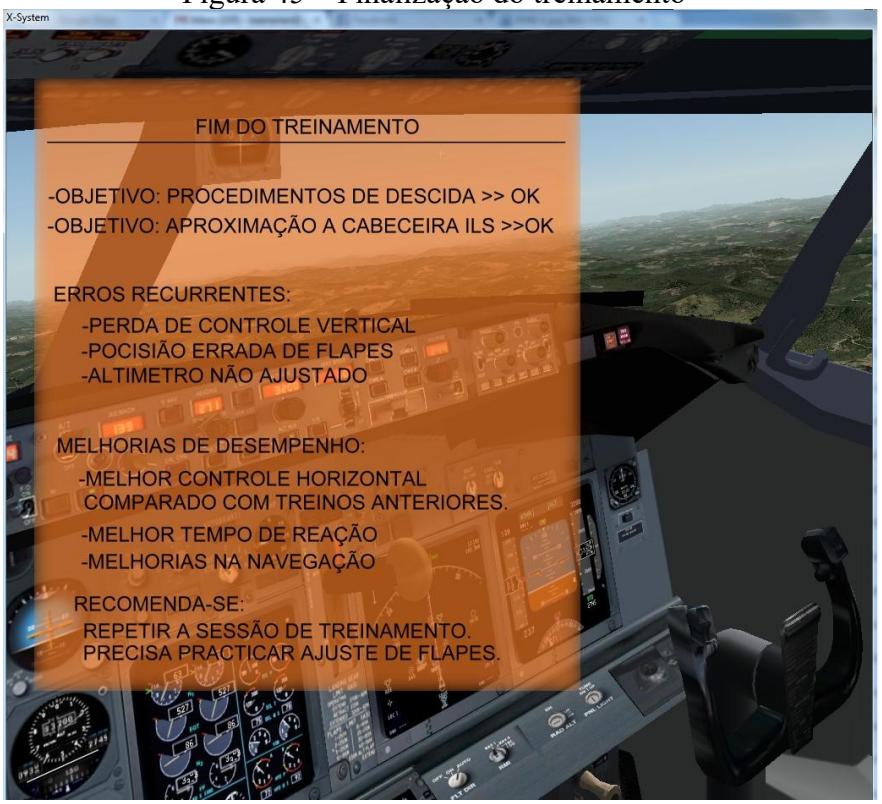
Se nem o agente Tutor e nem o Processador de Dinâmica resolver a emergência, a sessão de treinamento é reiniciada. Caso contrário, se a situação for resolvida, o agente Tutor devolve os mandos ao aluno, como mostrado na figura 44.

Figura 44 - Tutor devolvendo o controle do avião.

Fonte: Autor (2014).

Ao finalizar um treinamento, é mostrado ao aluno um relatório resumido indicando os objetivos logrados, os erros recorrentes no treino, as melhorias de desempenho e a recomendação final, como mostrado na figura 45.

Figura 45 – Finalização do treinamento



Fonte: Autor (2014).

O operador humano recebe, na Interface de Operador, dois relatórios (extensos) em formato de texto. O primeiro é o Relatório Técnico: mostra a lista completa de processos dos artefatos (simulador, interfaces, componentes do sistema), encadeamentos de raciocínio dos agentes, e as atividades dos agentes.

O segundo relatório é de avaliação do aluno. Indica as avaliações dos agentes e o parecer do agente Tutor. Mostra um Log ou lista sequencial dos erros, acertos e os estados emocionais (mapeados) do aluno.

5 TESTES DE VERIFICAÇÃO E VALIDAÇÃO

A verificação é um processo de revisão da funcionalidade do sistema com foco nas especificações de protótipo. O propósito da verificação das bases de conhecimento é procurar erros semânticos e de sintaxe. Devido à grande quantidade de testes feitos não foi possível detalhar todos os resultados, por isso é apresentada como foi feita a verificação, mostrando alguns casos representativos. A forma em que foi verificada a ontologia de modelo segue o princípio de que uma falha nas funcionalidades do sistema envolve um erro na ontologia, isto é, porque o sistema é baseado em manuseio de conhecimento. A expressão inversa é falsa, ou seja, ausência de erros nas funcionalidades não quer dizer que as bases de conhecimento estejam livres de falhas.

A validação, por outro lado, é um processo em que um profissional da área e usuários finais avaliam a funcionalidade do sistema.

5.1 VERIFICAÇÃO DA ONTOLOGIA DE MODELO

A ontologia de modelo representa o conhecimento da estrutura e a dinâmica do avião. Esta base de conhecimento foi verificada em três etapas:

a) Verificação da estrutura do modelo (hierarquias de classes e agregações) por meio de perguntas (querys) na plataforma;
b) Verificação dos encadeamentos funcionais gerados pelos agentes para observar o comportamento do veículo e detectar falhas;
c) Verificação da dinâmica do veículo modelado usando o processador de dinâmica.

5.1.1 Verificação da estrutura do modelo

O objetivo deste teste é verificar as hierarquias de classes dos conceitos e a suas relações de agregação. Pode ser feito diretamente no editor de ontologias protegé na aba "*DLQuery*" ou na plataforma JEMO da biblioteca “JEMO.jar” (ver Capítulo 4). As *querys* seguem a especificação *Manchester Syntax* para ontologias OWL [W3C, 2012].

Os predicados estruturais são as relações de agregação (parte_de, têm) e de generalização (é_um, sub_classe_de). O procedimento de verificação dos predicados estruturais é formulando perguntas (querys) na plataforma JEMO.

A pergunta <onde> é usada para mapear o domínio do conjunto de agregação (parte_de) de algum objeto. Por exemplo, a pergunta para saber onde está o motor direito é <onde(motorD)>.

A pergunta <partes> é usada para mapear os elementos pertencentes ao rango de agregação de algum sujeito. Por exemplo, a pergunta para saber de que partes é composto o sistema hidráulico é <partes(Sistema_Hidraulico)>.

A pergunta <tipo> é usada para mapear superclasses na hierarquia de conceitos. Por exemplo, a pergunta para saber quais superclasses contêm a classe avião é <tipo(Aviao)>, e algumas das respostas na plataforma serão:

<é_um(Aeronave)> (subclasse de Aeronave)
<é_um(Veiculo)> (subclasse de Veículo)

A pergunta <que_é> é usada para mapear superclasses na hierarquia de conceitos e mapear o conjunto domínio das agregações. Por exemplo, a pergunta para saber o que é um giroscópio phi é <que_é(Giroscopio_phi)>, e algumas das respostas na plataforma serão:

<é_um(Instrumento_inercial)> (subclasse de Instrumento_inercial)
<é_um(Intrumento_arfagem)> (subclasse de Instrumento_arfagem)
<é_um(Intrumento)> (subclasse de Instrumento)
<parte_de(Cabine)> (agregação ao elemento Cabine pertencente ao domínio)
<parte_de(Aviao) (agregação ao elemento Avião pertencente ao domínio)

As perguntas genéricas com busca de padrões são feitas diretamente com a sintaxe da especificação Manchester syntax. Seguem alguns exemplos de perguntas com este formato:

(Quais superfícies de controle fazem o avião guinar?)
(?x:Superfície_controle) some guinar(aviao)
(Quais instrumentos são exclusivamente para navegação?)
(?x:Mandos) only (navegacao)
(Que instrumentos e rádios têm a cabine?)

parte_de(cabine)and (?x:Instrumento) and (?y:Radio)

(Quais partes do avião são fabricadas pela GE?)

(?x:parte) and fabricado value "GE"

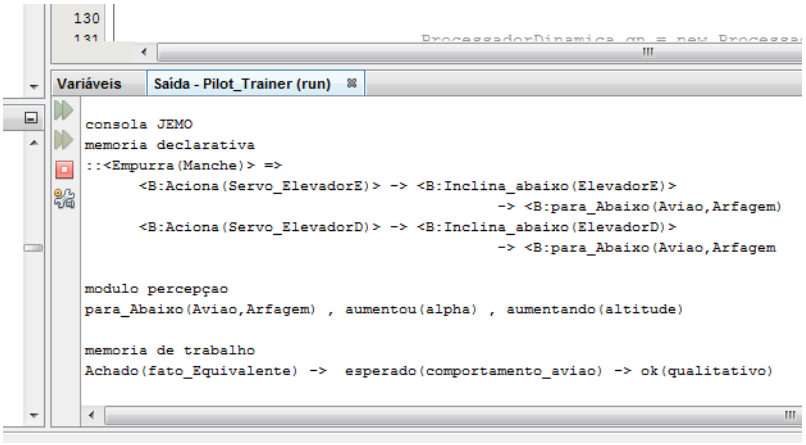
(Quais mandos reduzem a velocidade sem afetar alpha?)

(?x:Mandos) and reduz some(Velocidade) and not (diminui some(Altitude))

5.1.2 Verificação dos Encadeamentos Funcionais

Na verificação foi usado um agente de teste com a estrutura standard FIPA (*Dummy Agent*). Este agente tem a missão de observar, em um processo cíclico, os fatos gerados no módulo de percepção e os fatos gerados na memória declarativa. Para visualizar os fatos gerados nos módulos do agente foi usada a plataforma JEMO, como pode se verificar na figura 46.

Figura 46 - Plataforma de comandos JEMO



Fonte: Autor (2014).

Os Quadros 5 e 6 mostram alguns testes de verificação dos predicados funcionais

Quadro 5- Testes de predicados funcionais.

Ação na cabine: Empurrar manche
Fatos gerados no encadeamento funcional: <Empurra(Manche)> => <B:Aciona(Servo_ElevadorE)> -> <B:Inclina_abaixo(ElevadorE)> -> <B:para_Abaixo(Aviao,Arfagem) <B:Aciona(Servo_ElevadorD)> -> <B:Inclina_abaixo(ElevadorD)> -> <B:para_Abaixo(Aviao,Arfagem)
Fatos gerados no módulo de percepção: para_Abaixo(Aviao,Arfagem) , aumentou(alpha) , aumentando(Altitude), diminuindo(Velocidade)
Conclusão do agente de teste: Achado(fato_Equivalente) -> esperado(comportamento_aviao) -> ok(qualitativo)
Ação na cabine: Puxar manche
Fatos gerados no encadeamento funcional: <Puxa(Manche)> => <A:Aciona(Servo_ElevadorE)> -> <A:Inclina_acima(ElevadorE)> -> <A:para_Acima(Aviao,Arfagem) <A:Aciona(Servo_ElevadorD)> -> <A:Inclina_acima(ElevadorD)> -> <A:para_Acima(Aviao,Arfagem)
Fatos gerados no módulo de percepção: para_Acima(Avião,Arfagem) , aumentando(alpha) , diminuindo(Altitude), aumentando(Velocidade)
Conclusão do agente de teste: Achado(fato_Equivalente) -> esperado(comportamento_aviao) -> ok(qualitativo)
Ação na cabine: Girar manche esquerda
Fatos gerados no encadeamento funcional: <Gira_esquerda(Manche)> => <E:Aciona(Servo_AleiraoE)> -> <E:Inclina_acima(AleiraoE)> <E:para_Esquerda(Aviao,Guinada)-> <E:para_Esquerda(Aviao,Rolamento) <E:Aciona(Servo_AleirãoD)> -> <E:Inclina_abaixo(AleiraoD)> <E:para_Esquerda(Aviao,Guinada)-> <E:para_Esquerda(Aviao,Rolamento)
Fatos gerados no módulo de percepção: para_Esquerda(Avião,Rolamento) , diminuindo(alpha) , diminuindo(altitude)
Conclusão do agente de teste: Achado(fato_Equivalente) -> esperado(comportamento_aviao) -> ok(qualitativo)
Ação na cabine: Girar manche Direita
Fatos gerados no encadeamento funcional: <Gira_direita(Manche)> => <F:Aciona(Servo_AleiraoE)> -> <F:Inclina_abaixo(AleiraoE)> <F:para_Direita(Aviao,Guinada)-> <F:para_Direita(Aviao,Rolamento)> <F:Aciona(Servo_AleirãoD)> -> <F:Inclina_acima(AleiraoD)> <F:para_Direita(Aviao,Guinada)-> <F:para_Direita(Aviao,Rolamento)>
Fatos gerados no módulo de percepção: para_Direita(Avião,Rolamento) , diminuindo(alpha) , diminuindo(altitude)
Conclusão do agente de teste: Achado(fato_Equivalente) -> esperado(comportamento_aviao) -> ok(qualitativo)
Ação na cabine: Pisar pedal Esquerdo
Fatos gerados no encadeamento funcional: <Pisa_esquerdo(Pedal)> => <G:Aciona(Servo_Leme)> -> <G:Inclina_esquerda(Leme)> -> <G:para_Esquerda(Aviao,Guinada)>
Fatos gerados no módulo de percepção: para_Esquerda (Avião,Guinada) , diminuindo(alpha) , diminuindo(altitude)
Conclusão do agente de teste: Achado(fato_Equivalente) -> esperado(comportamento_aviao) -> ok(qualitativo)

Fonte: Autor (2014)

.

Quadro 6-Exemplos de verificação de predicados funcionais

Ação na cabine: Pisar pedal Direito
Fatos gerados no encadeamento funcional: <Pisa_direito(Pedal)> => <H:Aciona(Servo_Leme)> -> <H:Inclina_direita(Leme)> -> <H:para_Direita(Aviao,Guinada) >
Fatos gerados no módulo de percepção: para_Direita(Avião,Guinada) , diminuindo(alpha) , diminuindo(altitude)
Conclusão do agente de teste: Achado(fato_Equivalente) -> esperado(comportamento_aviao) -> ok(qualitativo)
Ação na cabine: Empurrar Throttle Direito e esquerdo
Fatos gerados no encadeamento funcional: <Empurra(ThrottleD)> => <D:Aciona(Servo_ValvulaD)> -> <D:Aumenta_potencia(motorD)> -> <D:para_Esquerda(Aviao,Guinada) -> <D:aumenta(Aviao,Velocidade)> <Empurra(ThrottleE)> => <C:Aciona(Servo_ValvulaD)> -> <C:Aumenta_potencia(motorE)> -> <C:para_Direita(Aviao,Guinada) -> <C:aumenta(Aviao,Velocidade)>
Fatos gerados no módulo de percepção: Aumenta(Avião,Velocidade) , compensada(Guinada)
Conclusão do agente de teste: Achado(fato_Equivalente) -> esperado(comportamento_aviao) -> ok(qualitativo)
Ação na cabine: Empurrar Spoilers
Fatos gerados no encadeamento funcional: <Empurra(Spoilers)> => <I:Aciona(Servo_SpoilersD)> -> <I:inclina_acima(SpoilersD)> -> <I:diminui(Aviao,Velocidade)> <Empurra(Spoilers)> => <j:Aciona(Servo_SpoilersE)> -> <J:inclina_acima(SpoliersE)> -> <J:diminui(Aviao,Velocidade)>
Fatos gerados no módulo de percepção: diminui(Avião,Velocidade) , aumentando(alpha)
Conclusão do agente de teste: Achado(fato_Equivalente) -> esperado(comportamento_aviao) -> ok(qualitativo)

Fonte: Autor(2014).

O procedimento de verificação dos encadeamentos funcionais consiste em quatro passos:

a. Comandar o painel da cabine do avião do simulador externo (um comando por vez, p.ex. puxar manche, pressionar pedal, etc.);
b. Observar os fatos gerados no encadeamento funcional do agente de teste na plataforma JEMO;
c. Observar o fato gerado no módulo de percepção do agente de teste na plataforma JEMO;
d. Observar a conclusão do agente de teste na plataforma JEMO.

5.1.3 Verificação dos encadeamentos funcionais em condições de falha no simulador

Este teste de verificação tem o propósito de conferir a funcionalidade de detecção de falhas no veículo ou no simulador. Ante um comportamento anômalo do avião, os agentes devem revisar os sistemas e subsistemas. O procedimento de verificação consiste em oito passos:

a) Forçar uma falha no veículo do simulador (pode ser feito na interface do operador, clicando em alguma falha);
b) Comandar o painel da cabine do avião do simulador externo (um comando por vez, p.ex. puxar manche, pressionar pedal, etc.);
c) Observar os fatos gerados no encadeamento funcional do agente de teste na plataforma JEMO;
d) Observar o fato gerado no módulo de percepção do agente de teste na plataforma JEMO;
e) Observar a conclusão do agente de teste na plataforma JEMO, que deveria ser uma possível falha no veículo. O agente marca o fato conflitante com o tag "CONFL:";
f) Observar os encadeamentos funcionais gerados que envolvem os conceitos da expressão do fato conflitante;
g) Observar um por um os componentes que o agente revisa;
h) Observar a conclusão final do agente. Ele procura uma falha por vez, ou seja, se achar uma falha, finaliza o processo de busca. Se houver mais falhas, perceberá no próximo ciclo.

O quadro 7 apresenta um exemplo de falhas no motor esquerdo do avião e o quadro 8 apresenta um exemplo de falha no aileron da asa esquerda.

Quadro 7 – Exemplo de falha no motor esquerdo

Ação na cabine: Pisar pedal Esquerdo; **Falha:** Perda total de potência no motor direito
Fatos gerados no encadeamento funcional: <Pisa_esquerdo(Pedal)> => <**G**:Aciona(Servo_Leme)> -> <**G**:Inclina_esquerda(Leme)> -> <**G**:para_Direita(Aviao,Guinada)>
Fatos gerados no módulo de percepção: para_Esquerda (Avião,Guinada) , diminuindo(alpha) , diminuindo(altitude)
Conclusão do agente de teste: falhou(qualitativo) -> **CONFL**:para_Direita(**Aviao,Guinada**) -> achado(conflitante) -> possivel(falha)
Encadeamentos envolvendo fato conflitante: #1) <Pisa_esquerdo(Pedal)> => <**G**:Aciona(Servo_Leme)> -> <**G**:Inclina_esquerda(Leme)> -> <**CONFL**:para_Esquerda(**Aviao,Guinada**)> -> <**G**:para_Esquerda(Aviao,Rolamento) #2) <Pisa_direito(Pedal)> => <**H**:Aciona(Servo_Leme)> -> <**H**:Inclina_direita(Leme)> -> <**H**:para_Direita(**Aviao,Guinada**)> -> <**H**:para_Direita(Aviao,Rolamento) #3) <Empurra(ThrottleD)> => <**D**:Aciona(Servo_ValvulaD)> -> <**D**:Aumenta_potencia(motorD)> -> <**D**:para_Esquerda(**Aviao,Guinada**) -> <**D**:aumenta(Aviao,Velocidade)> #4) <Empurra(ThrottleE)> => <**C**:Aciona(Servo_ValvulaD)> -> <**C**:Aumenta_potencia(motorE)> -> <**C**:para_Direita(**Aviao,Guinada**) -> <**C**:aumenta(Aviao,Velocidade)> #5) <Gira_esquerda(Manche)> => <**E**:Aciona(Servo_AleiraoE)> -> <**E**:Inclina_acima(AleiraoE)> <**E**:para_esquerda(**Aviao,Guinada**)-> <**E**:para_Esquerda(Aviao,Rolamento) <**E**:Aciona(Servo_AleirãoD)> -> <**E**:Inclina_abaixo(AleiraoD)> <**E**:para_esquerda(**Aviao,Guinada**)-> <**E**:para_Esquerda(Aviao,Rolamento) #6) <Gira_direita(Manche)> => <**F**:Aciona(Servo_AleiraoD)> -> <**F**:Inclina_acima(AleiraoD)> <**F**:para_Direita(**Aviao,Guinada**)-> <**F**:para_Esquerda(Aviao,Rolamento) <**F**:Aciona(Servo_AleirãoE)> -> <**F**:Inclina_abaixo(AleiraoE)> <**F**:para_Direita(**Aviao,Guinada**)-> <**F**:para_Esquerda(Aviao,Rolamento)
Componentes revisados pelo agente e resposta do módulo percepção: **Encadeamento #1)** revisando(Leme) ->esperado(comportamento) revisando(servo_lemeE) -> esperado(comportamento) revisando(pedal) -> esperado(comportamento) **Encadeamento #2)** revisando(servo_lemeE) -> esperado(comportamento) revisando(pedal) -> esperado(comportamento) **Encadeamento #3)** revisando(MotorD) -> anomalo(comportamento)->**CONFL**:para_Direita(**Aviao,Guinada**)
Conclusão final do agente de teste: falha(MotorD)

Fonte: Autor (2014)

Quadro 8 - exemplo de falha de aileron

Ação na cabine: Girar Manche Esquerda; **Falha:** Aileron esquerdo totalmente travado na posição neutral
Fatos gerados no encadeamento funcional: <Gira_esquerda(Manche)> => <**E**:Aciona(Servo_AleiraoE)> -> <**E**:Inclina_acima(AleiraoE)> <**E**:para_Esquerda(Aviao,Guinada)-> <**E**:para_Esquerda(Aviao,Rolamento)
Fatos gerados no módulo de percepção: para_Direita (Avião,Rolamento) , diminuindo(alpha)
Conclusão do agente de teste: falhou(qualitativo) -> **CONFL**:para_Direita(**Aviao,Rolamento**) -> achado(conflitante) -> possivel(falha)
Encadeamentos envolvendo fato conflitante: #1) <Pisa_esquerdo(Pedal)> => <**F**:Aciona(Servo_Leme)> -> <**F**:Inclina_esquerda(Leme)> -> <**F**:para_Esquerda(Aviao,Guinada) -> <**CONFL**:para_Esquerda(**Aviao,Rolamento**)> #2) <Pisa_direito(Pedal)> => <**G**:Aciona(Servo_Leme)> -> <**G**:Inclina_direita(Leme)> -> <**G**:para_Direita(Aviao,Guinada) -> <**G**:para_Direita(**Aviao,Rolamento**) > #3) <Gira_direita(Manche)> => <**F**:Aciona(Servo_AleirãoD)> -> <**F**:Inclina_abaixo(AleiraoD)> <**F**:para_Direita(Aviao,Guinada)-> <**F**:para_Esquerda(**Aviao,Rolamento**) <**F**:Aciona(Servo_AleiraoE)> -> <**F**:Inclina_acima(AleiraoE)> <**F**:para_Direita(Aviao,Guinada)-> <E:para_Esquerda(**Aviao,Rolamento**) #4) <Gira_esquerda(Manche)> => <**E**:Aciona(Servo_AleiraoE)> -> <**E**:Inclina_acima(AleiraoE)> <**E**:para_Esquerda(Aviao,Guinada)-> <E:para_Esquerda(**Aviao,Rolamento**) <**F**:Aciona(Servo_AleirãoD)> -> <**F**:Inclina_abaixo(AleiraoD)> <**F**:para_Direita(Aviao,Guinada)-> <**F**:para_Esquerda(**Aviao,Rolamento**)
Componentes revisados pelo agente e resposta do módulo percepção: **Encadeamento #1)** revisando(Leme) ->esperado(comportamento) revisando(servo_lemeE) -> esperado(comportamento) revisando(pedal) -> esperado(comportamento) **Encadeamento #2)** revisando(servo_lemeE) -> esperado(comportamento) revisando(pedal) -> esperado(comportamento) **Encadeamento #3)** revisando(AleiraoD) ->esperado(comportamento) revisando(servo_AleiraoD) -> esperado(comportamento) revisando(Manche) -> esperado(comportamento) revisando(AleiraoE) -> anomalo(comportamento)->**CONFL**:para_Direita(**Aviao,Rolamento**)
Conclusão final do agente de teste: falha(AleiraoE)

Fonte: Autor (2014)

5.2 VERIFICAÇÃO DA ONTOLOGIA DE DOMÍNIO

As ontologias de domínio representam todo o conhecimento didático que o piloto aprendiz (aluno) deve aprender e envolve a operação do veículo, máquina ou processo, incluindo controle, comando, regulamento, normas, medidas de segurança, iteração com o ambiente (condições ambientais), orientação, navegação, ergonomia, entre outros.

Para verificar as ontologias e regras de domínio é necessário conferir três funcionalidades. Um erro nestas funcionalidades envolve diretamente um erro na base de conhecimento no domínio específico, seja nas ontologias ou nas regras. Como já foi dito anteriormente, a expressão contrária desta afirmação é falsa, pois não encontrar erros nas

funcionalidades não, necessariamente, garante que a base de conhecimento no domínio específico esteja livre de erros. Isto se deve porque a combinação possível de casos de uso é infinita e nunca poderão todos serem testados. Em consequência, a verificação nunca será completa, mas pelo menos poderia se garantir um bom funcionamento do sistema enquanto for operado com os casos de uso já testados (ainda assim, podem surgir erros devido a inúmeros fatores não controláveis como, por exemplo, alguma operação do aluno fora do roteiro, alguma situação inesperada no simulador externo, etc.).

As três funcionalidades a serem conferidas para verificar as ontologias de domínio são:

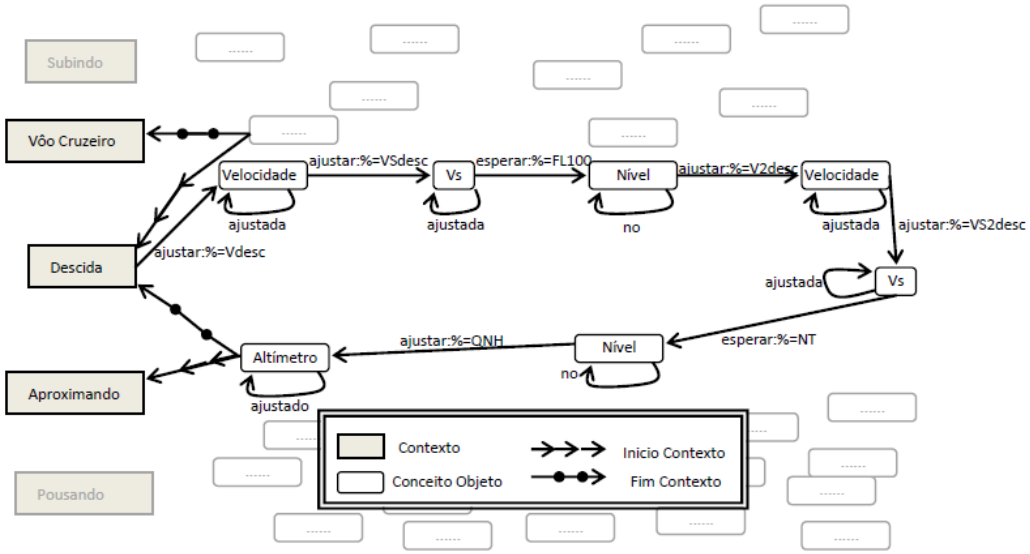
a) O mecanismo de raciocínio usado pelos agentes para avaliar o aluno;
b) O mecanismo de raciocínio do agente tutor para controlar o avião nos voos autônomos;
c) O mecanismo de raciocínio do agente tutor para dar assistência ao aluno (dar assistência não é o mesmo que treinar - será explicado mais adiante).

Um caso representativo para exemplificar um teste de verificação é uma estrutura simplificada de ontologia de domínio no contexto de "Descida", começando com o fim do contexto "voo cruzeiro" e finalizando com o início do contexto de "Aproximação".

As camadas de "Objetivos" contêm 8 objetivos completando um ciclo de objetivos. O ciclo de objetivos termina quando finaliza o contexto atual e inicia o próximo. A figura 47 mostra a estrutura do exemplo. O símbolo *%=* indica uma faixa onde um valor de referência considera-se pertencente ao conjunto de valores que satisfazem a expressão (p.ex. %=100 (Velocidade) define uma faixa de valores de velocidade em torno a 100 *Kias* que satisfazem a expressão, tornando o fato verdadeiro).

O quadro 9 acompanha os fatos mais importantes gerados nos diferentes módulos do agente (fatos mais importantes referem-se aos fatos diretamente envolvidos com o processo de avaliação). Cada vez que as ações do aluno encadeiam fatos, e esses fatos completam a lista de ações do fato condicional do lado direito da regra de objetivo, um novo objetivo é gerado. As conclusões do agente acontecem na memória declarativa.

Figura 47 – Exemplo representativo para teste de domínio



Fonte: Autor (2014).

Quadro 9 – Teste de avaliação do desempenho do aluno

Ação do Aluno	Fatos gerados pelo modulo de percepção	Fatos gerados pela memória procedural	Fatos gerados pela memória declarativa	Ações esperadas (na regra de objetivo)	Conclusões do agente de teste (dummy)
			Fim_Contexto:Cruzeiro Novo_Contexto:Descida		novo(Contexto)
Mantendo controle no avião		nivelado(altitude) mantendo(velocidade)	Objetivo:<ajustar_1(Velocidade)>	acionar(Vnav-Hold) %=Vdesc(APspeed)	fim_contexto(Voo_cruzeiro) inicio_contexto(Descida) novo(Objetivo)
-Pressiona o botão Vnav-Hold -Ajusta APspeed em Vdesc=230	acionado(Vnav-Hold) %=230(APspeed)	nivelado(arfagem) nivelado(altitude) diminuindo(velocidade)	Ajustada_1(Velocidade)		esperadas (Ações_Aluno) positiva(avaliação) +20(Pontos)
Mantendo controle no avião			Objetivo:<ajustar_2(Vs)>	desligar(Vnav-Hold) acionar(v/s) %=VSdesc(Vs)	novo(Objetivo)
-Pressiona botão desligando Vnav-Hold -Pressiona botão Vs -Ajusta Vs em VSdesc=-2000	desligado(Vnav-Hold) acionado(v/s) %=-2000(Vs)	diminuindo(altitude) mantendo(velocidade)	Ajustada_2(Vs)		esperadas (Ações_Aluno) positiva(avaliação) +30(Pontos)
Mantendo controle no avião		diminuindo(altitude) mantendo(velocidade) Esperando(FL100)	Objetivo:<esperaFL100(nivel)>		novo(Objetivo)
Mantendo controle no avião		mantendo(velocidade) %=FL100(Nivel_voo)	no_FL100(nivel_voo)		esperado(Evento)
Mantendo controle no avião			Objetivo:<ajustar_3(Velocidade)>	%=V2desc(APspeed)	novo(Objetivo)
-Ajusta APspeed em Vdesc=200	%=200(APspeed)	nivelado(arfagem) diminuindo(altitude) diminuindo(velocidade)	Ajustada_3(Velocidade)		esperadas (Ações_Aluno) positiva(avaliação) +10(Pontos)
Mantendo controle no avião			Objetivo:<ajustar_4(Vs)>	%=VS2desc(Vs)	novo(Objetivo)
-Ajusta Vs em VSdesc=-1500	%=-1500(Vs)	aumentando(arfagem) diminuindo(altitude) diminuindo(velocidade)	Ajustada_4(Vs)		esperadas (Ações_Aluno) positiva(avaliação) +10(Pontos)
Mantendo controle no avião		diminuindo(altitude) mantendo(velocidade)	Objetivo:<espera_NT(nivel)>		novo(Objetivo)
Mantendo controle no avião		mantendo(velocidade) %=NT(Nivel_voo)	no_NT(nivel_voo)		esperado(Evento)
Mantendo controle no avião		diminuindo(altitude) mantendo(velocidade)	Objetivo:<ajustaQNH(Altimetro)>	%=QNH(Altimetro)	novo(Objetivo)
-Ajusta altímetro no QNH do aeroporto de destino	%=QNH(Altimetro)	mantendo(arfagem) diminuindo(altitude) diminuindo(velocidade)	ajustadoQNH(Altimetro)		esperadas (Ações_Aluno) positiva(avaliação) +10(Pontos).
			Fim_Contexto:Descida Novo_Contexto:Aproximando		novo(Contexto)

Fonte: Autor (2014).

No processo de avaliação os agentes utilizam um raciocínio de encadeamento direto de regras (*forward chaining*) em todos os módulos de raciocínio.

O agente tutor pode controlar autonomamente o avião acessando a mesma estrutura de ontologia de domínio que usa para avaliar o aluno. O mecanismo de raciocínio é equivalente ao mecanismo de avaliação, exceto que as regras de objetivos são processadas usando encadeamento reverso (*backward chaining*). Ao invés de esperar que o fato condicional seja verdadeiro ao atender a lista de ações, o agente executa as ações em forma sequencial e logo valida o lado esquerdo da regra. Em outras palavras, o agente tutor segue a sequência de objetivos executando a lista de ações contidas nas regras de objetivo. O quadro 10 mostra o processo de controle do avião considerando o exemplo do ponto anterior; oito objetivos no contexto de "descida". O controle é feito quando o agente tutor executa as ações escrevendo os comandos no quadro negro. Os comandos escritos no quadro negro passam ao simulador externo por meio da interface de simulador.

Quadro 10 – Teste de avaliação do agente tutor pilotando o avião

Ação do Agente	Fatos gerados pelo modulo de percepção	Fatos gerados pela memória procedural	Fatos gerados pela memória declarativa	Ações esperadas (na regra de objetivo)	Conclusões do agente
			Fim_Contexto:Cruzeiro Novo_Contexto:Descida		novo(Contexto)
Mantendo controle vertical e horizontal do avião		nivelado(altitude) mantendo(velocidade)	Objetivo:<ajustar_1(Velocidade)>	acionar(Vnav-Hold) %=Vdesc(APspeed)	fim_contexto(Voo_cruzeiro) inicio_contexto(Descida) novo(Objetivo)
-aciona Vnav-Hold -ajusta APspeed em Vdesc=230	acionado(Vnav-Hold) %=230(APspeed)	nivelado(arfagem) nivelado(altitude) diminuindo(velocidade)	Ajustada_1(Velocidade)		completadas (Ações)
Mantendo controle vertical e horizontal do avião			Objetivo:<ajustar_2(Vs)>	desligar(Vnav-Hold) acionar(v/s) %=VSdesc(Vs)	novo(Objetivo)
-desliga Vnav-Hold -aciona Vs -ajusta Vs em VSdesc=-2000	desligado(Vnav-Hold) acionado(v/s) %=-2000(Vs)	diminuindo(altitude) mantendo(velocidade)	Ajustada_2(Vs)		completadas (Ações)
Mantendo controle vertical e horizontal do avião		diminuindo(altitude) mantendo(velocidade) Esperando(FL100)	Objetivo:<esperaFL100(nivel)>		novo(Objetivo)
Mantendo controle vertical e horizontal do avião		mantendo(velocidade) %=FL100(Nivel_voo)	no_FL100(nivel_voo)		esperado(Evento)
Mantendo controle vertical e horizontal do avião			Objetivo:<ajustar_3(Velocidade)>	%=V2desc(APspeed)	novo(Objetivo)
-ajusta APspeed em Vdesc=200	%=200(APspeed)	nivelado(arfagem) diminuindo(altitude) diminuindo(velocidade)	Ajustada_3(Velocidade)		completadas (Ações)
Mantendo controle no avião			Objetivo:<ajustar_4(Vs)>	%=VS2desc(Vs)	novo(Objetivo)
-ajusta Vs em VSdesc=-1500	%=-1500(Vs)	aumentando(arfagem) diminuindo(altitude) diminuindo(velocidade)	Ajustada_4(Vs)		completadas (Ações)
Mantendo controle vertical e horizontal do avião		diminuindo(altitude) mantendo(velocidade)	Objetivo:<espera_NT(nivel)>		novo(Objetivo)
Mantendo controle vertical e horizontal do avião		mantendo(velocidade) %=NT(Nivel_voo)	no_NT(nivel_voo)		esperado(Evento)
Mantendo controle vertical e horizontal do avião		diminuindo(altitude) mantendo(velocidade)	Objetivo:<ajustaQNH(Altimetro)>	%=QNH(Altimetro)	novo(Objetivo)
-ajusta altímetro no QNH do aeroporto de destino	%=QNH(Altimetro)	mantendo(arfagem) diminuindo(altitude) diminuindo(velocidade)	ajustadoQNH(Altimetro)		completadas (Ações)
			Fim_Contexto:Descida Novo_Contexto:Aproximando		novo(Contexto)

Fonte: Autor (2014).

Existem duas formas de assistência ao aluno: o aluno pode pedir ajuda se não souber como proceder, ou o agente tutor pode oferecer ajuda ao aluno se ele errar certo limite de

procedimentos ou se demorar certo tempo em executar alguma ação. Em ambos os casos o mecanismo que o agente tutor utiliza para assistir o aluno consiste em dar sugestões na tela com base nas ações listadas na regra do objetivo atual.

Considerar o seguinte exemplo: o avião está se aproximando da pista. Para um pouso guiado vai ser acionado a captura de ILS (grupo de antenas instaladas na pista para pouso assistido), e o objetivo atual é capturar a localização (LOC) do ILS. A regra do objetivo é:

```
(Regra LOC_ILS (
          ?x<- capturar(LOC)
     =>
    (     (retract ?x)
        (assert(capturado(LOC))
    ) conditioned_to
              { <action>ajusta=RW_ILS(NAV1)
                <action>ajusta=NAV1(Source)
                <action>aciona(LOC)
                <action>espera:faixa(ILS)
                <action>espera:capture(LOC) }
)
```

Supondo que o aluno executa corretamente as duas primeiras ações, mas não sabe como proceder depois. Ele pode pedir ajuda clicando no botão “Ajuda” na interface. Se não pedir ajuda e errar as ações ou se passar certo tempo sem fazer nada, o agente pode oferecer ajuda na interface. O aluno pode aceitar ou rejeitar a ajuda.

Se ele pedir ajuda ou aceitar a ajuda do agente, o Tutor procede do seguinte modo:

a) Lê a próxima ação que o aluno deveria executar;
b) A ação está associada a um comentário explicando como proceder (o mecanismo de explicação dentro das regras de alto nível são as que associam um texto a uma ação definida). O agente mostra esta informação na interface;
c) As ações etiquetadas como “espera” são associadas às regras de eventos de nível médio. O agente acessa os comentários ligados a essas regras e mostra na interface. Assim, por exemplo, a ação *espera:faixa(ILS*) tem um grupo de regras de nível médio associadas. Se o avião estiver acima, embaixo, esquerda ou direita da faixa ILS, as regras de nível meio ativadas contêm comentários como: “Está indo muito alto, baixa altitude com VS -1500”, “Vira para esquerda com rumo 50 graus”, etc. O agente mostra os comentários na

interface, à medida que as regras vão sendo ativadas, guiando ao aluno o tempo todo.

O quadro 11 ilustra a situação do exemplo, mostrando uma sequência de interações entre o aluno e o tutor por meio da interface.

Quadro 11- Exemplo de interação entre o aluno e o agente tutor

Ação do Aluno	Ação do Agente	Mensagem na interface
Ajusta NAV1 em 110.30 MHZ	Confere a regra de objetivo e conclui que a ação <action>ajusta=RW_ILS(NAV1) foi feita corretamente.	
Ajusta Source na posição NAV1	Confere a regra de objetivo e conclui que a ação <action>ajusta=NAV1(Source) foi feita corretamente.	
Aluno não sabe como proceder. Fica sem fazer nada por mais de 10 segundos.	Conclui que o avião está perto da pista e que o aluno precisa agir rápido. Oferece ajuda ao aluno.	Parece que você está tendo problemas. Precisa ajuda?
Aluno aceita a ajuda.	Agente procura mensagem associada à ação <action>aciona(LOC) (nas regras de explicação).	No painel do piloto automático tem o botão "LOC". Pressione-o. (e mostra uma seta na tela indicando o botão)
Aluno pressiona botão LOC.	Agente procura as regras de nível médio associadas à ação <action>espera:faixa(ILS). Agente mostra as mensagens das regras ativadas (p.ex. avião voando muito alto).	Você precisa entrar na faixa da rampa ILS para poder capturar a localização. Está voando muito alto, precisa descer com VS=-2000.
Aluno baixa altitude do avião, mas se desvia para a esquerda.	Agente procura a mensagem de texto da regra de nível médio ativada, associada ao desvio do avião para a esquerda.	Você precisa virar para a direita com rumo de 20 graus.
Aluno entra na faixa ILS e o avião captura a localização LOC.	Confere a regra de objetivo e conclui que as ações <action>espera:faixa(ILS) <action>espera:capture(LOC) foram feitas corretamente. Agente desliga funcionalidade de assistência.	Você conseguiu capturar a localização do ILS.

Fonte: Autor (2014).

5.3 VERIFICAÇÃO DO CONJUNTO DAS BASES DE CONHECIMENTO

As bases de conhecimento que não foram mencionadas anteriormente neste capítulo não foram testadas individualmente, porque não faz sentido fazer esta tarefa. É preciso fazer testes em conjunto, incluindo as bases de conhecimento, os agentes e os artefatos.

Os métodos de verificação descritos anteriormente nos diferentes tipos de regras e ontologias foram úteis para conferir funcionalidades individuais, ajudando, assim, detectar certos erros de sintaxe e de semântica. Mas na hora de verificar a operação conjunta do sistema com as bases de conhecimento surgiram mais erros, produto da iteração entre agentes, artefatos e bases de conhecimento. A tarefa de rastreio e correção destes erros foi o maior gargalho no andamento do projeto. Estes erros foram na maioria de procedimentos produto de erros de semântica (fatos que faltam ou não deveriam estar na memória de trabalho no momento esperado, regras ativadas erradamente ou sem ativar quando eram esperadas, conclusões (ações) conflitantes, etc.). Os procedimentos considerados são aqueles declarados nas ontologias de domínio (procedimentos de assistência, avaliação e controle), ontologias de gerenciamento (procedimentos de treinamento e de avaliação em função do modelo emocional, protocolos e resolução de conflitos), e modelo de aluno (procedimento de atualização do perfil, de avaliação emocional e de desempenho).

A taxa percentual de erros foi medida comparando as ações erradas com as ações esperadas. Para estes testes de verificação as ações esperadas foram definidas como:

a) Procedimentos de operação do agente Tutor ao controlar o avião do simulador externo em missões - função de diferentes objetivos e condições, incluindo situações críticas;
b) Planejamento do roteiro de navegação e roteiro de slides feito pelo agente Tutor;
c) Orientações, dicas, advertências e qualquer outra forma de assistência dada pelo tutor ao aluno;
d) Avaliação feita pelos especialistas durante a sessão de treinamento.

A princípio, a taxa percentual de erros era em média de 30% dos procedimentos avaliados. Encontrar a fonte desses enganos era muito difícil e, por consequência, impossível consertar, já que se desconhecia a origem deles. Para encontrar a fonte dos erros foi necessário idealizar outra forma de verificação. Foram feitas grandes quantidades de ciclos de testes até

conseguir diminuir a taxa percentual de erros a 1%, acumulando, aproximadamente, 300 horas-simulador.

O 1% de taxa percentual de erros é muito difícil de reduzir, pois essas falhas acontecem por efeitos de escolha de critérios de conjuntos numéricos (limite para determinar se estão dentro ou fora da aerovia, critérios para determinar se os parâmetros são críticos ou não, etc.). Outra fonte de lapsos são os elementos aleatórios associados à execução da simulação externa, pois as condições podem variar ainda reiniciando com as mesmas situações.

Os passos de cada ciclo de verificação conjunta são:

Primeira parte do ciclo:

a) O agente Tutor assume o papel do aluno;
b) O agente Tutor assume todas as bases de conhecimento, exceto o modelo do aluno;
c) O operador configura missões com certos objetivos e condições. O agente Tutor controla o avião do simulador externo em um voo autônomo tentando lograr todos os objetivos;
d) Os agentes avaliadores observam o desempenho do agente Tutor como se ele fosse o aluno. Um dos avaliadores mantém atualizado um perfil do aluno correspondente ao agente Tutor;
e) No final das missões os agentes avaliadores geram relatórios de desempenho (eventos da missão) e relatórios semânticos (mensagens entre agentes, regras ativadas e fatos na memória procedural, memória declarativa, módulo percepção e memória de trabalho);
f) Observar no relatório de desempenho os erros de procedimento do agente Tutor e acompanhar o encadeamento de regras e fatos no relatório semântico dos agentes Avaliadores;
g) Observar no relatório semântico os erros de procedimentos dos agentes Avaliadores.

Segunda parte do ciclo:

a) O usuário humano assume o papel de aluno cadastrando no sistema certo perfil de aluno;
b) O operador humano configura os objetivos e condições da sessão;

c) Fazer uma sessão em condições normais. Registrar os planos de roteiros e de slides do agente tutor. Registrar passo a passo as ações do agente tutor, dando prioridade às ações de assistência ao aluno (orientações, dicas, oferecer ajuda, etc.). Registrar passo a passo as avaliações feitas pelos agentes especialistas;
d) Observar todos os aspectos dos relatórios finais.

Terceira parte do ciclo:

a) Após de recolher todos os erros, ordená-los numa tabela.
b) Se o erro for do Tutor ao controlar o avião do simulador externo, a fonte do erro está numa das ontologias de domínio. Por isso, é importante sempre colocar um marcador nas regras e ontologias para fazer um seguimento dos encadeamentos e mapeamentos;
c) Se o erro for do Tutor ao planejar um roteiro de navegação, a fonte de erro está na ontologia de modelo de ambiente, provavelmente, na definição de heurísticas de critérios de tempo de navegação, consumo de combustível, aspectos geográficos, etc.;
d) Se o erro for do Tutor no planejamento do roteiro de slides, a fonte do erro está na ontologia de conteúdo. Se o agente fez uma decisão intuitiva, a fonte do erro encontra-se no algoritmo de seleção de experiências ou no conteúdo das experiências;
e) Se o erro for do Tutor em algum conteúdo na interface do aluno, a fonte de erro está na ontologia de domínio de comando. Se o Tutor oferecer ajuda ou deixar de oferecer quando não corresponde, ou se mostrar conteúdo a mais ou a menos, ou se não atender o aluno quando pede ajuda, a fonte de erro está no modelo emocional do agente localizada na ontologia de gerenciamento;
f) Se o erro for dos agentes especialistas na avaliação, a fonte de erro está nas ontologias de domínio;
g) Se algum dos agentes não seguir o comportamento esperado a fonte de erro está na ontologia de protocolos (bases de conhecimento de gerenciamento).

Quarta parte do ciclo:

a) Uma vez identificada a fonte de erro, fazer uma revisão na documentação de aquisição de conhecimento, no procedimento de representação de conhecimento;
b) Depois das revisões, atualizar e/ou evoluir as bases de conhecimento;
c) Repetir o ciclo completo.

5.4 VALIDAÇÃO COM USUÁRIOS DO SISTEMA

Os usuários finais do sistema de instrução virtual são alunos de escolas de aviação comercial. O objetivo destes testes é validar a efetividade do sistema, principalmente, as especificações de protótipo. Os pontos a avaliar são principalmente:

a) Usabilidade do Sistema ou a percepção do usuário sobre a facilidade de uso, e em que grau é intuitivo e amigável. O método de validação é por meio de questionário;
b) Atratividade do Sistema ou em que grau o usuário gosta ou não do sistema. O método de validação é por meio de questionário;
c) Os estados emocionais do aluno. O modelo emocional dos agentes encontra-se na ontologia de gerenciamento, sendo que na mesma ontologia se define a relação entre a forma de ensinar do agente tutor e a forma de avaliar dos agentes especialistas, depende de seus estados emocionais. Por outro lado, o modelo emocional do aluno encontra-se na ontologia de modelo, onde é mapeado o que os agentes percebem do aluno e, dependendo do comportamento dele, o estado emocional é mapeado nesta ontologia. As duas ontologias estão indiretamente relacionadas (seguindo o modelo OCC), ou seja, algumas emoções dos agentes são alteradas pelo estado emocional do aluno. Assim, se o aluno está chateado por ir mal num treinamento, os agentes vão sentir remorso, como os professores fariam se vissem que seu aluno não foi bem. Os testes foram qualitativos, comparando os estados emocionais mapeados na ontologia de Modelo de Aluno, e o estado emocional que os estes declararam ter (os alunos

confirmavam verbalmente seu estado emocional a cada certo intervalo entre eventos ou de tempo);

d) A Pedagogia adaptativa. Os planos de treinamento feitos pelo agente Tutor são feitos em certo grau em base ao perfil emocional do aluno e das preferências pedagógicas. As estratégias de ensino e avaliação dependem dos estados emocionais mapeados no Modelo do Aluno. Observando os planos e ações dos agentes, pode ser acompanhado como eles mudam suas estratégias e planos para se adaptar ao aluno. Mas o problema é que sem testes com usuários não há como saber se aquela adaptabilidade é compatível com alunos humanos. A forma de avaliar esta questão é perguntando (por meio de questionário) ao aluno em que grau percebe que o sistema acompanha ou não o aprendizado dele, e se percebe que o sistema reage ao seu desempenho e comportamento;
e) A interação natural entre sistema e aluno. O objetivo de avaliar se o aluno percebe a interface do sistema como um meio de comunicação com outro humano, sendo que na realidade está se comunicando com um agente virtual. Um extremo positivo seria se o aluno achasse que está conversando com um tutor humano por meio da interface. Este ponto deve ser incluído no questionário de validação. Outra questão importante de ser perguntada ao aluno, é se consegue perceber algum estado emocional do Tutor (por exemplo, um aluno que faz tudo errado pode perceber que o Tutor está ficando chateado, baseado na forma em que se expressa na interface).

Foram convocados 12 voluntários, para fazer duas sessões, de uns 20 minutos cada. Durante as sessões foram feitas, com uma frequência de 5 minutos, perguntas sobre o estado emocional deles (em referência ao modelo OCC). A primeira sessão de treinamento foi fácil e a segunda foi muito difícil (para observar como reagem quando erram em quase todo no treinamento).

O estudo foi totalmente qualitativo, dando os seguintes resultados:

a) Os voluntários na maioria interagiram com o sistema sem problemas, no questionário declararam que a interface é fácil de usar, amigável e intuitiva. A maioria se manifestou positivamente na questão da atratividade (em uma escala de 0 a 10, gostou (10) o não gostou (0) do sistema; a média das respostas foi 8);

b) Há diferencias significativos entre os estados emocionais negativos mapeados no Modelo de Aluno, e os estados emocionais dos alunos humanos, especialmente, na raiva e na frustração. O modelo tende a ser bem mais negativo. Em projetos futuros deve se ajustar as escalas emocionais negativas no Modelo do Aluno;
c) Todos os voluntários estavam duvidosos de quem conversava com eles por meio da interface, isto é, se era humano ou virtual. Ninguém conseguiu perceber o estado emocional do Tutor. Provavelmente, em 20 minutos não é possível perceber. Nos próximos estudos seria bom realizar uma quantidade mais significativa de testes, com sessões de maior duração;
d) A maioria dos voluntários achou que o sistema acompanhou em certo grau o aprendizado deles, e todos perceberam que ele reage, principalmente, ao mau desempenho.

5.5 VALIDAÇÃO COM UM ESPECIALISTA

O teste de validação com um especialista, neste caso, um piloto de linha aérea, tem como objetivo avaliar a efetividade do conhecimento representado no sistema e no manuseio dele. Os testes foram feitos em duas sessões de duas horas cada, executados em duas etapas; testes de controle autônomo e testes no instrutor virtual. No final das sessões o piloto respondeu dois questionários (ver no Anexo B), um para cada sessão, incluindo avaliações e recomendações para melhorar o sistema.

5.5.1 Validação de pilotagem

Na primeira etapa, o piloto apenas observou o sistema controlando (pilotando) o avião do simulador externo em diferentes missões, com diferentes objetivos e condições. O objetivo desta etapa foi avaliar de forma prática os conhecimentos do sistema no comando do avião, nos procedimentos de voo, na navegação, no regulamento aeronáutico, na operação em função das condições meteorológicas e nos procedimentos de emergência. Os passos do teste são mostrados no quadro 12.

Quadro 12- Validação da inteligência artificial (agente tutor) pilotando o avião

Objetivos	Condições	Pontos a serem avaliados	Resumo da avaliação do profissional	Recomendações do Professional
Decolagem Ascenso Voo Cruzeiro Aproximação Pouso Rota e Aerovia de livre escolha.	Origem: Guarulhos 09R Pressão na pista: 997 hpA Destino: Galeão 033 Pressão na pista: 1004 hpA Vento 012 11kt no solo. Vento 100 20kt FL100.	Planejamento de voo. Escolha de aerovia e nível de voo. Execução de procedimentos. Execução de manobras.	Bom controle lateral e longitudinal. Plano de voo, escolha de aerovia e nível de voo dentro do esperado e respeita o regulamento. Procedimentos seguem o regulamento aeronáutico. Alguns problemas na execução de manobras; Tendência de levantar voo antes do recomendado. Execução de curvas muito fechadas (provocaria incômodo aos passageiros).	Revisar a execução das manobras, pois são muito bruscas. Revisar a execução da decolagem.
Recuperar o avião de uma perda de sustentação.	Altitude 25000 pés, velocidade 340 Kias. Vento 011 10Kt.	Procedimentos e Manobras de emergência. Decisões feitas.	Procedimentos e manobras certas, mas existem leves demoras nas decisões que comprometem a segurança no voo.	Existem manobras de emergência que devem ser atualizadas, de acordo com os novos protocolos da ANAC.
Manobrar o avião após uma falha do motor direito na decolagem.	Decolando e subindo a 2000 ft/min e 180 kias. Aeroporto Galeão 033.	Procedimentos e manobras na: -Estabilização do avião. -Pouso em segurança.	Procedimentos e manobras certas, mas devem ser melhor sincronizadas.	Existem manobras de emergência que devem ser atualizadas, de acordo com os novos protocolos da ANAC.

Fonte: Autor (2014).

5.5.2 Validação do instrutor virtual

Na segunda etapa, o piloto avaliou o desempenho do instrutor virtual, observando o conteúdo dos slides apresentados na interface e os conteúdos das orientações e recomendações. O piloto assumiu o papel de aluno e fez uma mesma sessão de treinamento duas vezes, na primeira fazendo tudo certo, e na segunda vez, cometendo erros de propósito. O quadro 13 mostra o resumo das duas sessões. No final das sessões o piloto observou e avaliou o relatório de erros gerado pelo protótipo.

Quadro 13- Validação do instrutor virtual

Objetivos	Condições	Pontos a serem avaliados	Resumo da avaliação do profissional
Decolagem Ascenso Voo Cruzeiro FMS,GPS Aproximação VOR Pouso ILS	Origem: Guarulhos 09R Pressão na pista: 990 hpA Destino: Galeão 033 Pressão na pista: 1009 hpA Vento 020 01kt no solo. Vento 111 01kt FL100. Aerovia: W52 Nível: FL250 Agentes configurados para serem rigorosos (*ST*=10).	Planejamento dos Slides. Conteúdo das recomendações e orientações do instrutor virtual. Pedagogia adaptativa do instrutor virtual. Relatório de avaliação gerado pelo instrutor de voo.	A sequência de slides é coerente, mas o conteúdo às vezes entrega informação redundante. Muito texto nos slides, às vezes tira o foco de atenção. Quando o instrutor oferece ajuda está acima da hora do problema, deveria se antecipar mais um pouco. O conteúdo da ajuda do instrutor orienta em forma oportuna e certeira. Ao pedir ajuda, às vezes o instrutor faz perguntas demais, exigindo ter que clicar nos botões “sim” ou “não”, isso distrai muito. Pode se observar claramente como o instrutor virtual reage aos erros recorrentes. Foi interessante quando mudou o plano de voo e o conteúdo dos slides mudaram para um nível mais básico. Os relatórios do instrutor virtual mostram uma avaliação coerente, mas, em geral não foram considerados alguns erros de sincronia nos procedimentos.

Fonte: Autor (2014).

6 CONCLUSÕES

Neste capítulo serão analisados aspectos do trabalho de doutorado; começando com o atendimento das especificações de projeto e as especificações de protótipo. Na sequência, serão discutidos os aspectos mais relevantes dos testes de verificação feitos. Por último, serão analisados futuros possíveis projetos ou novas ideias que possam ser geradas a partir deste trabalho.

6.1 ANÁLISE NO ATENDIMENTO DAS ESPECIFICAÇÕES DE PROJETO

No capítulo 3 foram definidas as especificações do modelo de sistema de treinamento que foi usado como base para o desenvolvimento do protótipo. A primeira e mais importante especificação exige o uso de agentes no sistema, a qual foi atendida.

A Ontologia de Modelo permite ao sistema atender a especificação de manter uma representação do veículo e do ambiente em que opera. A Ontologia de Domínio atende a especificação de manter uma representação dos domínios de conhecimento que envolve a operação do veículo. A estrutura dos agentes permite ao sistema atender a especificação de contar com um mecanismo de raciocínio contextual no manuseio do modelo do veículo. O Processador de Modelo permite que sejam atendidas as especificações de contar com um mecanismo de processamento numérico, que pode controlar o veículo remotamente nas situações de emergência e no apoio aos alunos novos em algumas manobras.

A base de conhecimento Modelo de Aluno permite ao sistema atender a especificação de representar o modelo do aluno, considerando o histórico dos treinamentos, um perfil de desempenho, um perfil emocional e as preferências de treinamento. Também permite atender a especificação de mapeamento de emoções do aluno com base nas suas respostas e no comportamento.

A camada de estados emocionais dos agentes na Ontologia de Gerenciamento permite ao sistema atender a especificação de simulação de emoções nos agentes, sendo que, em função das suas emoções, eles avaliam e ensinam (apenas o Tutor ensina).

A Ontologia de Conteúdo permite ao sistema atender a especificação de planejamento das sessões dos treinamentos, sendo possível fazer novos planejamentos de roteiros de slides e planos de navegação quando o agente Tutor considerar necessário. O atendimento desta especificação foi testado na verificação e na validação.

O modelo de sistema é extensível para uso com qualquer veículo, máquina ou processo, e expansível a quantas novas funcionalidades for preciso (dependendo das especificações de cada projeto), já que depende da estrutura das representações nas bases de conhecimento de Treinamento e Gerenciamento, e não do sistema em si mesmo. As bases de conhecimento permitem desenvolver um protótipo inicial e evoluí-lo de forma incremental quantas vezes forem necessárias, até chegar a uma versão final. O modelo de sistema de instrução virtual recomenda usar normas e especificações da FIPA, A&A e da W3C, sendo eles organismos internacionais bem conhecidos, que mantêm uma grande coleção de documentação, canais de atendimento; e os conteúdos são periodicamente atualizados na medida em que a tecnologia evolui. O modelo permite a independência de operação com qualquer tipo de simulador e, também, a interoperabilidade com outros sistemas (herdado das especificações da FIPA e A&A).

A confiabilidade do sistema se apoia na autonomia dos elementos: se um artefato falhar, um agente pode acionar uma nova instância do artefato; se um agente falhar, outro agente pode assumir o papel dele; e conflitos comuns podem ser resolvidos por meio dos protocolos estabelecidos no modelo de instrução virtual.

A autonomia do sistema não permite ao Operador humano participar nas decisões no treinamento, ou seja, o Operador apenas pode configurar os objetivos e condições da sessão, observar o treinamento e ler os relatórios no final da sessão.

6.2 DISCUSSÃO DAS ESPECIFICAÇÕES DE PROTÓTIPO

No capítulo 3 foram definidas as especificações de protótipo. Em base, as especificações de protótipo foram definidas, a concepção e desenvolvimento do protótipo.

A especificação de protótipo mais importante é a pedagogia adaptativa do sistema. O mapeamento na Ontologia de Conteúdo permite ao agente Tutor planejar o treinamento em função do perfil do aluno, considerando o seu perfil de desempenho, emocional e as preferências pedagógicas (estilo de aprendizado, por exemplo, se prefere descobrir sozinho como fazer um novo procedimento ao invés de pedir ajuda ao Tutor). Observando os planos e ações dos agentes, pode se verificar que seguem as estratégias de ensino e avaliação estabelecidos na Ontologia de Gerenciamento. O aspecto da percepção humana desta funcionalidade foi validado por voluntários estudantes de uma escola de aviação e por um profissional (piloto de linha aérea). Eles percebem que o sistema acompanha o aprendizado e

que reage ao desempenho, especialmente quando é negativo. Em trabalhos futuros seria interessante trabalhar nas reações ao desempenho positivo, e fazer o agente demonstrar que está feliz de que o aluno esteja indo bem.

A segunda especificação de protótipo é conseguir melhorar a interação natural entre o agente Tutor e o Aluno. Nos testes de validação com voluntários, a maioria não sabia distinguir se o tutor por trás da interface era humano ou um agente virtual. Este fato pode se considerar suficientemente bom para atender esta especificação.

A terceira especificação estabelece que a interface deva ser intuitiva, amigável e fácil de usar. Nos testes de validação, os voluntários efetivamente acharam o sistema intuitivo, amigável e fácil de usar, em média com um peso 7 em escala de 0 a 10, suficientemente bom para considerar esta especificação atendida.

A quarta especificação estabelece que o sistema deva ser confiável. A taxa de falhas e conflitos resultante dos testes de verificação é, consideravelmente, baixa, já que os protocolos estabelecidos permitem que os agentes resolvam a maior parte dos conflitos.

A quinta especificação estabelece que as ferramentas de desenvolvimento usadas permitam independência de hardware; sistema operacional bem desenvolvido e evoluído, documentado; facilmente acessíveis; frequentemente atualizadas; normalizadas e que sigam especificações de organismos internacionais. O uso da linguagem JAVA permite independência e interoperabilidade entre hardware e sistemas operacionais. O uso do formato FIPA-SL e do framework JADE permite que o sistema esteja normalizado com as especificações da FIPA. O uso do formato OWL e do framework A&A permite que o sistema esteja normalizado pela W3C.

A sexta especificação estabelece disponibilidade de especialistas. Neste quesito foi difícil atender esta especificação, já que os pilotos de linhas aéreas ficam pouco tempo na cidade, assim complicado agendar alguma reunião.

A última especificação estabelece a usabilidade do sistema. O framework desenvolvido para construir o protótipo pode ser reusado por qualquer projetista, já que será liberado como software livre. O framework permitirá desenvolver sistemas de instrução virtual para qualquer veículo, máquina ou processo que possa ser utilizado com qualquer tipo de simulador.

6.3 DISCUSSÃO DA VERIFICAÇÃO DO SISTEMA

De forma geral, a quantidade de testes de verificação feitas no período de desenvolvimento gerou uma grande quantidade de documentação, sendo impossível poder sintetizá-la. Por isso, serão discutidos alguns resultados significativos.

A verificação de estrutura de modelo é complexa porque é necessário formular tantas perguntas quantas relações diretas e indiretas existam no modelo de veículo e do ambiente (mais de 500). Não existe mecanismo autônomo para fazer isso. Os resultados dos testes foram positivos para as relações que foram testadas. Ainda existe possibilidade de existir algum erro ao testar alguma relação pela primeira vez. Faz parte dos erros inevitáveis.

Na verificação dos encadeamentos funcionais para detectar falhas, foram testadas todas as combinações possíveis (67), pelo que pode se garantir que não vai ter erros de encadeamento funcional.

A verificação da dinâmica do avião mostra resultados suficientemente bons para a funcionalidade requerida, já que as diferenças entre o modelo próprio do sistema e o modelo do simulador externo são muito baixas (faixa de um 1%). A diferença entre os modelos é devido a que o simulador externo (XPLANE9) baseia seus cálculos na análise de elementos finitos da estrutura distribuída do avião, e o Processador de Modelo baseia os cálculos nas equações de estado com parâmetros concentrados.

Quando o Processador de Modelo é acionado para recuperar o avião de uma emergência, a taxa de recuperação do avião é proporcional à altitude em que acontece a emergência. Quanto mais alto, a taxa de recuperação se aproxima de 100%, mas quando a emergência acontece a menos de 3000 pés, a taxa de recuperação diminui a 20%; a menos de 1000 pés a taxa de recuperação cai a 5%; a menos 500 pés é, praticamente, impossível recuperar o avião se houver instabilidade lateral. Perda de sustentação por instabilidade lateral é muito mais difícil de recuperar a qualquer altitude. Isso se deve aos efeitos cruzados entre o ângulo de arfagem e o ângulo de rolamento em resposta a deflexão dos ailerons e do leme.

O mecanismo de raciocínio dos agentes na avaliação é de encadeamento reverso na memória declarativa, baseado na expectativa. Foram avaliados todos os encadeamentos de domínio (59), pelo que a possibilidade de erros de domínio é mínima (e não totalmente zero, já que os encadeamentos foram testados com as ontologias de domínio isoladas, e não com todas as ontologias interconectadas).

O mecanismo de raciocínio do agente tutor para controlar o avião nos voos autônomos é de encadeamento direto. Foram testados todos os encadeamentos possíveis (67), pelo que a possibilidade de erros de domínio direto é mínima (e não totalmente zero, já que os encadeamentos foram testados com as ontologias de domínio isoladas, e não com todas as ontologias interconectadas).

O mecanismo de raciocínio do agente tutor para dar assistência ao aluno também é baseado no encadeamento direto, com a diferença de que o raciocínio é orientado em objetivos e não em expectativas. Todos os encadeamentos foram testados, mas a possibilidade de erro ainda não é zero, já que os testes foram feitos com a ontologia isolada e não em conjunto. O problema de fazer testes de encadeamento com as ontologias em conjunto é que a combinatória de relações cresce exponencialmente, fazendo inviável este tipo de teste, já que não existe um mecanismo que faça tudo automaticamente.

O mecanismo e critérios para uso de experiências foram testados apenas para bater 100% das condições. Assim, não é garantido não ter erros se mudar os parâmetros de busca.

Os testes de protocolo foram feitos em condições forçadas, pois a probabilidade de ocorrência de conflitos é muito baixa. Os resultados dos testes garantem um bom grau de robustez do sistema.

6.4 DISCUSSÃO DA VALIDAÇÃO DO SISTEMA

A opinião geral do profissional (piloto de linha aérea) foi que o protótipo tem um grande potencial de evoluir o conhecimento, cobrindo totalmente todos os domínios envolvidos, mas requer um trabalho dedicado de uma equipe de especialistas focados totalmente no projeto. Por agora, o protótipo precisa de revisão no conhecimento em alguns procedimentos e uma revisão nos mecanismos de avaliação na sincronia de manobras. Também, insistiu que a interface precisa ter menos texto e menos elementos que distraem (como a necessidade de clicar em botões na tela enquanto executa alguma manobra nos mandos).

A opinião geral dos voluntários (estudantes de escolas de aviação) foi que a interface é amigável e intuitiva, e que gostaram da forma que o instrutor virtual ensina. Quando foram consultados dos seus estados emocionais, indicando o grau de escala (0-10), as emoções negativas dos voluntários foram diferentes dos estados emocionais mapeados no Modelo de Aluno (muito mais negativos). Em projetos futuros devem-se ajustar as escalas emocionais negativas no Modelo do Aluno. Em todos os casos, foram observadas claramente reações do

sistema de instrução virtual aos procedimentos errados, mas é difícil perceber mudanças ou reações no sistema quando os procedimentos são feitos corretamente. Em projetos futuros deve-se melhorar o mecanismo de recompensa (parabenizar ao aluno) nos procedimentos bem feitos.

6.5 CONTRIBUIÇÕES DO TRABALHO

Neste trabalho foi apresentada uma proposta de sistema de treinamento autônomo que pode ser vantajosa onde se tem pouca disponibilidade de instrutores humanos. Foi apresentada uma proposta de sistema de instrução virtual para veículos usando simuladores.

A contribuição principal deste trabalho foi uma melhoria que aborda as limitações dos sistemas existentes IFT-AIS usados pelo exército dos EUA. Quanto à contribuições à área da educação, foram apresentadas estruturas de melhorias de treinamento com simuladores (ensino coaching) para que os agentes consigam compreender o significado dos processos dinâmicos do simulador.

Foram propostos avanços em software de dinâmica de veículos, máquinas ou processos utilizando processamento numérico e simbólico ao nível conceitual, diferentemente dos softwares convencionais, apresentando uma nova estrutura de base de conhecimento que permite um fácil processamento simbólico e numérico integrado. Esta questão responde a necessidade de processar de forma simples um modelo de veículo, máquina ou sistema em forma determinista e ao mesmo tempo em forma lógica.

Assim, é mais simples manter uma coerência conceitual e numérica do sistema. Em consequência, foram agregadas as melhores práticas em ambientes inteligentes de aprendizagem aos simuladores, considerando a arquitetura do sistema, estrutura dos agentes, interação entre agentes, artefatos e bases de conhecimento, e, considerando também, a forma da estrutura e organização do conhecimento, agregando grandes avanços ao estado da arte nos sistemas multiagentes aplicados a treinamento e supervisão virtual. Proximamente, será liberado um framework em JAVA para desenvolver sistemas de treinamento virtual (baseado no framework do modelo JEMO usado no desenvolvimento do protótipo). Este permitirá aos projetistas desenvolver sistemas de instrução virtual para treinamento com qualquer veículo, máquina ou processo usando qualquer tipo de simulador (sempre que seja feita a interface certa).

A grande contribuição para a área da pedagogia é a estrutura da ontologia de conteúdo, que gerencia o material didático a ser mostrado ao aluno, oferecendo aos programadores uma

plataforma que facilita a implementação de sistemas pedagógicos por meio do framework desenvolvido neste trabalho (JEMO.jar).

6.6 POSSÍVEIS TRABALHOS FUTUROS E NOVAS IDEIAS

A modelagem de veículos, máquinas ou processos num formato que pode ser, ao mesmo tempo, manuseado por agentes inteligentes (ou qualquer sistema com inteligência artificial) e processado por simuladores e softwares de modelado e simulação, abre infinitas possibilidades de aplicações, principalmente, nos campos da engenharia e ciências. Deste ponto de vista, seria possível:

a) Melhorar os modelos de agentes inteligentes, integrando um raciocínio simbólico e numérico;
b) Poderia ser criada uma biblioteca distribuída e compartilhada, contendo diversos modelos de veículos, máquinas e processos em um formato normalizado, acessível a qualquer pessoa, seja por algum meio pago ou não.

A capacidade de detecção de falhas graças ao manuseio das ontologias de modelo (capítulo 3) abre um amplo campo de aplicações, desde detecção de falhas físicas nos sistemas, falhas de códigos em softwares, até detecção de uso errado de ferramentas de software como Simulink, Modelica, Solid Works, etc. Todo isto permitiria:

a) Desenvolver sistemas detectores de falhas em sistemas físicos e sistemas de software;
b) Desenvolver ferramentas de apoio e interpretação para diversos softwares de desenvolvimento; por exemplo: para um aluno de engenharia seria bem útil contar uma ferramenta de apoio ao modelar um sistema no Simulink, pois a ferramenta o guiaria na configuração certa de parâmetros de modelo e dos métodos de resolução, e ajudaria na interpretação das curvas de saída das simulações, podendo explicar os fenômenos que acontecem e por que acontecem.

A capacidade de controle autônomo de veículos, máquinas e processos, abre grandes possibilidades no campo de sistemas que operam sem intervenção humana e veículos não tripulados. Algumas possibilidades de aplicações podem ser:

Melhoramento na tecnologia de drones autônomos para missões de vigilância, busca, resgate, exploração, etc.;

Melhoramento dos sistemas autônomos de sensoriamento remoto.

6.7 ASPECTOS GERAIS

Este trabalho de doutorado foi desenvolvido com alguns contratempos devido a problemas de erros semânticos e de sintaxe nas bases de conhecimento. Devido à falta de bibliografia e trabalhos afins sobre bases de conhecimento de grande porte aplicado aos veículos, foi investida uma grande quantidade de tempo em resolver, principalmente, o problema dos erros semânticos. No final foram atendidos os objetivos que era abordar as limitações dos IFT-AIS. Os objetivos propostos nesta tese foram atendidos.

Este trabalho pode se tornar muito melhor com uma interface de voz (sintetizador e reconhecedor), e se as ontologias e regras pudessem ser representadas em um editor integrado, se o framework contasse com um ambiente gráfico, e se o Processador de Modelo incluísse algoritmos de resolução de sistemas não lineares e de análise de elementos finitos.

Outra melhora que poderia ser feita em trabalhos futuros é a implementação de um mecanismo de aprendizado não assistido, pois assim poderia ser representado qualquer sistema sem ter que fazer representação de conhecimento.

Uma primeira versão foi desenvolvida a partir de adaptações no modelo OCC e nos modelos da estrutura dos STI e dos Agentes Cognitivos. Esta versão foi verificada pelo piloto comercial da linha aérea GOL (documentação no APÊNDICE B). Uma nova versão foi feita mantendo os modelos originais sem alterar, fazendo as mudanças nas bases de conhecimento e protocolos do sistema. Os testes de verificação foram feitos apenas na plataforma de debug, mostrando um leve aumento no tempo de ciclo de consulta nas bases de conhecimento (10% aproximadamente) e mantendo a funcionalidade.

REFERÊNCIAS

ANAC, [2010]: Regulamento para certificação e normalização de simuladores de Voo. Disponível em:<http://www2.anac.gov.br/simulador/qualificacao.asp> Acesso em: Maio de 2013.

[ALLERTON, D. **The impact of flight simulation in aerospace**. The Aeronautical Journal, 114(1162), 747-756, 2010. *[S.l].[S.n]*.doi:10.1017/S0001924000004231, 2010.

ANDERSON, J.: **The Adaptive Character of Thought**. Hillsdale, NJ: Erlbaum Associates, 1990. *[S.l].[S.n]*.

ACT-r, 2005. **Framework para implementar raciocínio baseado no modelo ACT.** Disponível em: < http://act-r.psy.cmu.edu/software/> Acesso em: Janeiro de 2012.

BACK N., OGLIARI A., ET.al : **Projeto Integrado de Produtos** - Planejamento, Concepção e Modelagem. Ed. Manole, 2008.

BEHAR, P. **Modelos pedagógicos em educação a distância**. In: BEHAR, Patrícia Alejandra. (Org.).Modelos pedagógicos em Educação a distância. Porto Alegre: Artmed, 2009, p. 15-32.

BECKER, F. : **Ensino e construção do conhecimento**: o processo de abstração reflexionante. Educação e Realidade, Porto Alegre, RS, v. 18, n. 1, p. 43-52, 1993.

BOEING 737-600,700,800,900; **Operation Manual**, THE BOEING CORPORATION,1997. *[S.l].[S.n]*.

BOEING 737-800 **Series Technical Sheets**, THE BOEING CORPORATION, 1999. *[S.l].[S.n]*.

CARBONELL, J.AI in CAI: An **Artificial-Intelligence Approach to Computer -** Assisted Instruction.Man-Machine Systems, IEEE Transactions, 1970. *[S.l].[S.n]*.

CATALDI Cataldi, Z., & Lage, F. **Sistemas tutores inteligentes orientados a la enseñanza para la comprensión**, 2009. *[S.l].[S.n]*.

CLIPS: **Framework para desenvolvimento de Sistemas Especialistas**. Disponível em: <http://clipsrules.sourceforge.net/> Acesso em: Janeiro de 2014.

CRANE, P. et. al.: **Advancing Fighter Employment Tactics in the Swedish and US Air Forces** Using Simulation Environments, In Transforming Training and Experimentation through Modelling and Simulation (pp. 19-1 – 19-12). Meeting Proceedings RTO-MPMSG-045, Paper 19. Neuilly-sur-Seine, France, 2006.

DAVIS C., OLIVEIRA Z. **Psicologia na educação**. 2 ed. Cortez, São Paulo, 1994.

DAHLSTROM, Dahlstrom, S. Dekker, R. van Winsen & J. Nyce. **Fidelity and validity of simulator training, Theoretical Issues in Ergonomics Science**. 10:4, 305-314, DOI: 10.1080/14639220802368864, 2009. *[S.l].[S.n]*.

FIPA: **Especificações para sistemas multiagentes**. Disponível em:< http://www.fipa.org/specs/> Acesso em: Fevereiro, 2014

FIPA-SL: **Especificações de linguagem de conteúdo para agentes:** Disponível em:<http://www.fipa.org/specs/fipa00008/XC00008G.pdf> Acesso em: Fevereiro, 2014

GRAÇA L.: **Integração de Emoção e Raciocínio em Agentes Inteligentes**. Tese de Doutorado em Informática, Universidade de Lisboa, 2006.

GIRAFFA L.: **Uma Arquitetura de Tutor Utilizando Estados Mentais.** Tese de Doutorado, PPGCC-UFRGRS, 1999. Porto Alegre, RS.

JADE, 2014. **Plataforma aberta para desenvolvimento de sistemas multi agentes**. Disponível em:<http://Jade.tilab.com> Acesso em: Fevereiro, 2014

JASON, L. **The Study of Emotions.** Stanford University,2005.

KING, M. **Process Control**: A Practical Approach. Chichester, UK: John Wiley & Sons Ltd. ISBN 978-0-470-97587-9, 2010.

LACKLAVIK, M., BABIK, M., et.al. **AgentOWL**: Semantic Knowledge Model and Agent Architecture, In Computing and Informatics. Vol. 25, no. 5 (2006), p. 419-437. ISSN 1335-9150, Chapters 1, 4, 5. *[S.l].[S.n]*.

LAMINAR RESEARCH, [2013]: **Simulador de voo X-PLANE.** Disponível em:<http://www.laminarresearch.com/> Acesso em: Fevereiro, 2014
LUDWIG J., RAMACHANDRAN S. **Developing an Adaptative Intelligent Flight Trainer**, U.S. Army Research Institute, Ft. Rucker, AL, 2002.

MACHADO, A. **Neurologia funcional**. São Paulo, Ed. Ateneu, 1998.

MILLS, T., Kleinman S.: **Emotions, Reflexivity and Action:** An Interactionist Analysis. Social Forces,Ed.Stor, Vol.66 No.4 pag.1009-1027, jun.1998. *[S.l]*.

KRISHNAKUMAR, K., SAWAL D., ET.al: **A simulator-based automated helicopter hover trainer**: Synthesis and verification. In IEEE Transactions on Systems, Man, and Cybernetics, 21, 961-970. 1991. *[S.l]*.

MULGAND S., ASDIGHA M., ET.al: **An intelligent tutoring system for simulator-based helicopter flight training**. In Proceedings of the 1995 Flight Simulation Technologies Conference. 1995. *[S.l]*.

ONTORY, A., CLORE G., COLLINS A. **The cognitive structure of emotions**. Cambridge: Cambridge University Press,1998.

OWL, 2013, **Formato OWL**, Normas e Especificações. Disponível em:< http://www.w3.org/2004/OWL/> Acesso em: Fevereiro, 2014

PARK, Beverly Park Woolf. **Building Intelligent Interactive Tutors**. Ed. Morgan Kaufmann, 2009.

PELANEK, R. **Bayesian knowledge tracing, logistic models, and beyond**: an overview of learner modeling techniques, "User Modeling and User-Adapted Interaction", 2014, pp. 313-350. *[S.l].[S.n]*.

POZZEBON, E.; "**Tutor inteligente adaptável conforme as preferências do aprendiz"**. Dissertação (mestrado) - Universidade Federal de Santa Catarina, Centro Tecnológico, Programa de Pós-Graduação em Ciência da Computação, Araranguá, 2003.

PROTEGE: **Editor de ontologias**. Disponível em:< http://protege.stanford.edu/> Acesso em: Fevereiro, 2014

PRESSMAN, S.: **Software Engineering**: A Practitioner's Approach. 6th McGraw-Hill Higher Education. 2006, ISBN:0072496681

PSOTKA, J., MASSEY, L. D., & MUTTER, S. A. (Eds.). (1988). **Intelligent tutoring systems: Lessons learned.** Hillsdale, NJ, US: Lawrence Erlbaum Associates, Inc.

REMOLINA, E., RAMACHANDRAN, S. ET.al.ç **Intelligent Simulation-Based Tutor for Flight Training**, Ft. Belvoir Defense Technical Information Center,2004. *[S.l]*.

RICCI A.,VIROLI M., OMICINI A.: **Give Agents their Artifacts**: The A&A Approach for Engineering Working Environments in MAS. 6th International Joint Conference "Autonomous Agents & Multi-Agent Systems" (AAMAS 2007), 14-18 May 2007 *[S.l]*.

ROMANO, L., BACK N. , OGLIARI A. **Sistemática de avaliação do processo de desenvolvimento de máquinas agrícolas**. 5 Congresso Brasileiro de Gestão de Desenvolvimento de Produto, 2005, Curitiba, PR. 5 CBGDP, 2005.

ROSKAM, J. **Airplane Flight Dynamics & Automatic Flight Controls**: Part I & II. DAR Corporation, 2008.

ROSEMAN, I., WIEST C., SWARTZ T. **Phenomenology, behaviors, and goals differentiate discrete emotions**. Journal of Personality and Social Psychology, 67, 206-221, 1994. *[S.l]*.

RUSSELL, S., NORVIG P.: **Artificial Intelligence**: A Modern Approach., New Jersey: Prentice Hall, 2005, 932 p.

RUSSEL, J., FERNANDEZ-DOLS J. **The psychology of facial expression**. New York: Cambridge University Press, 1997.

SCHAFER, F. **Generalized feedback control and application to vehicle path following control**, 200 pp. Tese: Fakultat IV - Elektrotechnik und Informatik,TU Berlin, Alemanha, 2004.

SELF, J. **The Defining Characteristics of Intelligent Tutoring System Research.** International Journal of Artificial Intelligence in Education, 1999. *[S.l]*.

SILVEIRA, R., GOMES E., VICARI R.: **Intelligent Learning Objects: An Agent-Based Approach of Learning Objects**. In: WEERT, Tom J. Van; TATNALL, Arthur. (Org.). Information and Communication Technologies and Real-Life Learning : New Education for the Knowledge Society. Kluwer, Boston, 2005, p. 103-110. ISBN 0-387-25996-1.

SIMBIONIC: **Framework open source para desenvolvimento de inteligência artificial nas áreas de treinamento, videojogos e processos complexos**.
Disponível em:<http://www.simbionic.com/indexjs.htm> Acesso em: Fevereiro, 2014

SCHERER, K.: **What are emotions**? And how can they beWhat are emotions? And how can they bemeasured?. Measured Social Science Information & 2005 SAGE Publications (London, Thousand Oaks, CA and NewDelhi), 0539-018

THIBODEAU, Marc-André; BÉLANGER, Simon; FRASSON, Claude.WHITE RABBIT – **Matchmaking of User Profiles Based on discussion analysis Using Intelligent Agents.** In Lectures Notes in Computer Science. Inteligent Tutoring Systems. Procedings of 5th International Conference, ITS 2000, Montreal, Canadá, June 2000. P.113-122.

[VANLEHN, , Kurt. **The Behavior of Tutoring Systems**. LRDC, University of Pittsburg, Pittsburg, PA, USA, 2006.

VICCARI R., JAQUES P., VERDIN R.: **Agent-Based Tutoring Systems by Cognitive and Affective Modeling**. 1. ed. Hershey, PA: IGI Global, 2008. 300p.

WIDMARK, J: **Social Agent**: Facial Expression Driverfor an e-Nose. 2003. *[S.l].[S.n]*.

WOOLDRIDGE, M., JENNINGS N.: **Agent Theories**, **Arquitechtures and Languages**; A Asurvey. Intelligent Agents - Springter-Verlag, Berlin, 1995.

APÊNDICE A – PROTOCOLOS DOS AGENTES

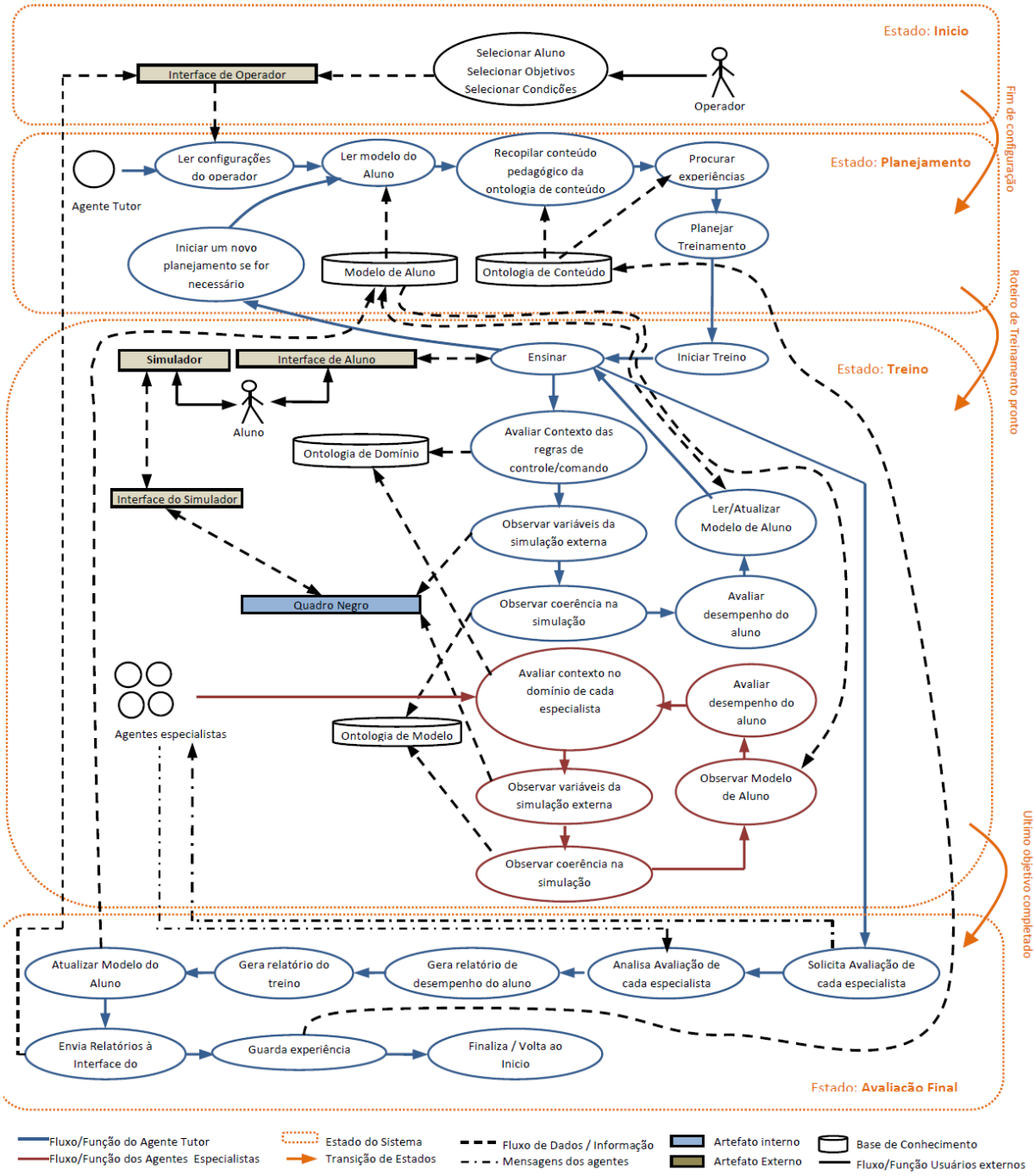
A.1. PROTOCOLOS DE COMPORTAMENTO DOS AGENTES

Os protocolos de comportamento dos agentes são sete;

i. **Inicia.** Neste protocolo o agente zera os estados mentais, os registros de avaliação, roteiros, etc., preparando as interfaces, artefatos e estados internos dos agentes para um novo treinamento.
ii. **Finaliza**. Neste protocolo o agente fecha os ciclos de treinamento e avaliação, concluindo os protocolos com os artefatos e interfaces.
iii. **Em espera**. Neste protocolo o agente interrompe, temporalmente, todos os processos internos e interações com artefatos e interfaces.
iv. **Continuar**. Neste protocolo o agente retoma o processo e interações que interrompeu no protocolo de espera.
v. **Treinamento**. Este protocolo é composto por quatro estados sequenciais, onde o agente Tutor e os agentes avaliadores participam. Na figura 2.39 pode se visualizar um ciclo de treino.
vi. **Controlar.** Este protocolo é executado apenas pelo agente Tutor. Na assistência ao aluno ou num protocolo de emergência o agente tutor assume o controle do sistema do simulador externo, comandando-o ao escrever no quadro negro.
vii. **Simular**. Este protocolo é executado dentro do protocolo de conflito de decisão. Se o conflito de decisão envolve diretamente o domínio do sistema alvo; o agente envolvido no conflito solicita uma simulação interna do sistema alvo no *Processador de Dinâmica*.

A figura 48 mostra o protocolo de treinamento dos agentes.

Figura 48- Protocolo de treinamento.



Fonte: Autor (2014).

No protocolo de treinamento todos os agentes participam em uma sequência de quatro estados:

- Estado "Início": Quando o sistema está no estado de "início" o operador externo usa a interface do operador para selecionar o aluno (carregar perfil/criar perfil) a ser treinado,

depois seleciona os objetivos e condições do treinamento. A transição para o próximo estado acontece quando o operador externo conclui a configuração do treino.

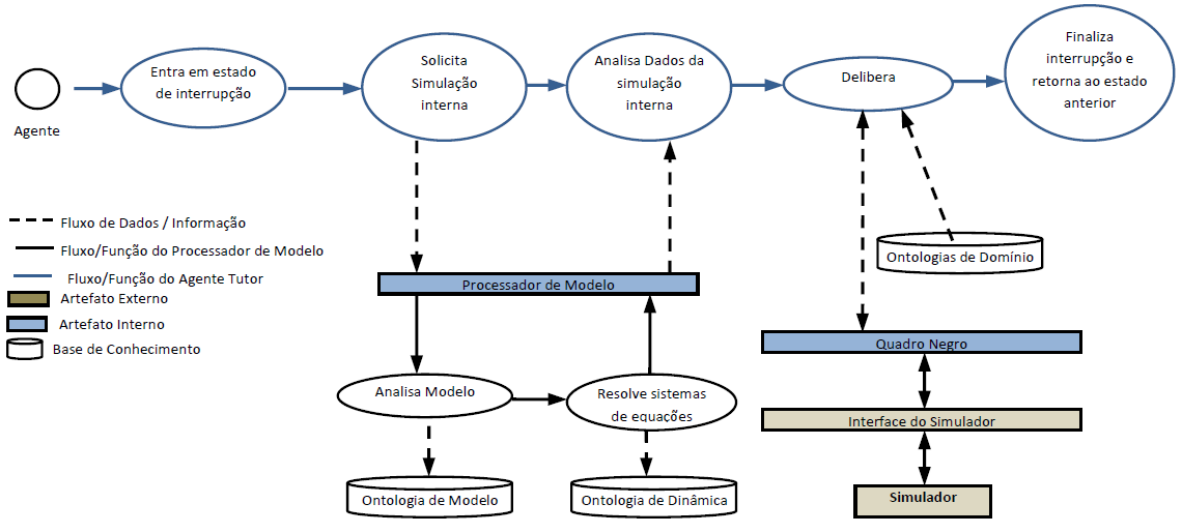
- Estado "Planejamento": No estado de planejamento, o agente Tutor lê os objetivos e condições do treinamento desde a interface do operador, coleta informações do aluno do *Modelo do Aluno*, recopila material pedagógico da *Ontologia de Conteúdo* (de acordo com as necessidades dos objetivos e condições do treinamento) e logo planeja um roteiro de ações (para o treinamento). A transição para o próximo estado acontece quando o plano de ações ou roteiro de treinamento for gerado.
- Estado "Treino": No estado de treino o agente Tutor inicia a comunicação com o aluno por meio da *Interface de aluno* e depois inicia um subciclo repetitivo, que termina quando o último objetivo for concluído. No subciclo, o agente Tutor executa seis tarefas consecutivas (Se o projetista precisar pode configurar as tarefas para serem executadas em paralelo ou em subciclos combinados). Em primeiro lugar, o agente Tutor expõe o conteúdo pedagógico ao aluno por meio da *Interface do aluno* seguindo o roteiro de treinamento estabelecido no estado anterior. Em segundo lugar, o Tutor avalia o contexto das regras de comando e controle (verifica se as regras conferem com o contexto da situação atual) encapsuladas na *Ontologia de Domínio*. Em terceiro lugar, o Tutor observa as variáveis da simulação externa (do ambiente simulado e do veículo, máquina ou processo) por meio do quadro negro, assim pode observar as ações do aluno no simulador. Em quarto lugar, o agente Tutor verifica a consistência da simulação externa (detecção de erros) analisando a ontologia de modelo. Em quinto lugar, o tutor avalia as ações do aluno em base as regras encapsuladas na *Ontologia de Domínio* e que foram selecionadas no contexto da situação, e em base as variáveis observadas no quadro negro (ao observar as variáveis o agente pode analisar o desempenho do aluno, o comportamento do aluno, e estimar o estado emocional dele). Em sexto lugar, o agente completa o subciclo atualizando o modelo do aluno quando for necessário. Se o agente Tutor considerar necessário (p.ex. em base ao comportamento e desempenho do aluno) pode interromper o estado de "Treino" e voltar ao estado de "Planejamento" para planejar um novo roteiro de treinamento. Em forma independente, cada um dos agentes especialistas avalia (de acordo ao domínio de especialidade de cada um) o desempenho do aluno. Cada agente especialista realiza um subciclo repetitivo de cinco tarefas (o subciclo termina quando o agente Tutor solicita aos agentes especialistas uma avaliação final do desempenho do aluno). Em primeiro lugar, cada agente verifica o contexto das

regras de acordo ao domínio de cada um e situação global do sistema. Em segundo lugar, cada agente especialista observa as variáveis da simulação externa (do ambiente simulado e do veículo, máquina ou processo) por meio do quadro negro, assim pode observar as ações do aluno no simulador. Em terceiro lugar, cada agente especialista verifica a consistência da simulação externa (detecção de erros) analisando a ontologia de modelo. Em quarto lugar, cada agente observa o *modelo do aluno* para verificar o desempenho em cada domínio, mudanças de comportamento e de estado emocional do aluno. Em quinto lugar (por último), cada agente avalia o desempenho do aluno, registrando os eventos dos subciclos em um relatório (pode ser encapsulado em um objeto). A transição para o próximo estado acontece no momento que o agente Tutor completar o último objetivo do treinamento, interrompendo os ciclos de tarefas de todos os agentes.

- Estado “Avaliação final”: No estado de "Avaliação final" o agente tutor solicita aos agentes especialistas os relatórios finais de cada um. Os agentes especialistas respondem com seu respectivo relatório, e assim o agente tutor analisa os relatórios em conjunto. Com base nisto, o agente Tutor gera um relatório final do desempenho do aluno, incluindo avanços, erros, comportamentos e aspectos emocionais de importância. Depois, o agente Tutor gera um relatório do treino, incluindo aspectos da simulação e eventos do treino. Uma vez gerados os relatórios, o agente Tutor atualiza o modelo do aluno, depois envia os relatórios para a interface do operador. Assim, o operador externo pode ler e analisar cada relatório. O modelo retorna ao estado de “Início” em espera de uma nova operação.

A figura 49 mostra o protocolo de acionamento do módulo de dinâmica.

Figura 49- Protocolo de simulação.



Fonte: Autor (2014).

A2. PROTOCOLOS DE COMUNICAÇÃO DOS AGENTES

Os protocolos de comunicação dos agentes são baseados em duas performativas básicas da especificação FIPA-SL:

- INFORM : A performativa INFORM consiste em que um agente notifica um evento para um ou todos os agentes, e o(s) agente(s) receptor(es) responde(m) com a performativa UNDERSTOOD se estão cientes do evento ou com a performativa NOT_UNDERSTOOD, se não entenderam o conteúdo da mensagem (isso acontece normalmente porque alguma palavra não está no dicionário da ontologia de linguagem que o agente está usando, outro motivo pode ser um erro de sintaxe na expressão por parte do agente que enviou a mensagem).
- QUERY_IF: A performativa QUERY_IF consiste em uma solicitação que um agente faz a outro(s) agente(s) por meio de uma mensagem. O(s) agente(s) receptor(es) respondem com a performativa ACCEPT se aceitar a solicitação ou responde(m) com a performativa REJECT se rejeitar a solicitação, enviando no conteúdo da mensagem de resposta o motivo porque a solicitação for rejeitada.

Os protocolos de comunicação entre os agentes são oito:

- **Reportar:** A cada certo intervalo de tempo, os agentes avaliadores mandam uma mensagem com performativa INFORM para o agente tutor informando que estão ativos. Assim, se algum agente não se reportar, o agente tutor pode tentar se comunicar usando performativa

QUERY_IF, solicitando ao agente avaliador se reportar. Se não houver resposta após de certa quantidade de tempo e de tentativas, o agente pode acionar o protocolo de falha de agente avaliador (será explicado mais para frente).

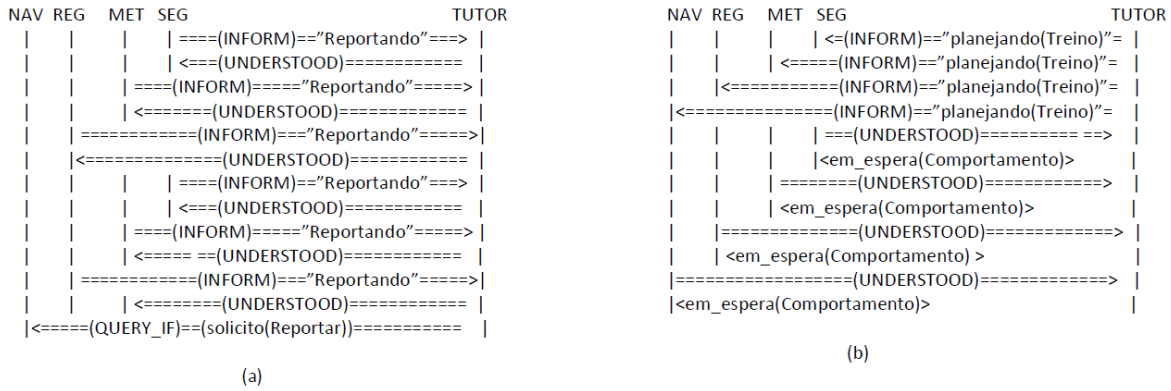
- **Pré-Treinamento:** Antes de iniciar um treinamento, o agente tutor deve planejar o roteiro de treinamento e a rota de navegação. Antes disso, o agente tutor manda uma mensagem a todos os agentes avaliadores com a performativa INFORM comunicando que está planejando o treinamento. Os agentes avaliadores respondem a mensagem com a performativa UNDERSTOOD e executam a ação de ficar na espera.
- **Início_Treinamento:** Quando o agente tutor está pronto para iniciar o treinamento manda uma mensagem aos agentes avaliadores com performativa QUERY IF solicitando iniciar a avaliação. Em resposta, os agentes mandam uma mensagem com a performativa ACCEPT e executam a ação de iniciar a avaliação o aluno.
- **Fim_treinamento:** Ao finalizar um treinamento o agente tutor envia uma mensagem aos agentes avaliadores com a performativa QUERY_IF solicitando finalizar a avaliação. Os agentes avaliadores respondem com a performativa ACCEPT e executam a ação de finalizar a avaliação do aluno.
- **Relatório:** Ao finalizar um treinamento, o agente tutor envia uma mensagem aos agentes avaliadores com a performativa QUERY_IF solicitando o relatório de avaliação. Os agentes avaliadores respondem com a performativa ACCEPT e executam a ação enviar o relatório de avaliação do aluno (objeto serializado na mensagem).
- **Emergência:** Se qualquer um dos agentes perceberem alguma probabilidade de evento catastrófico no veículo do simulador externo, ele notifica ao agente tutor com uma mensagem com performativa INFORM. Quando o agente tutor recebe a notificação ele envia uma mensagem aos outros agentes com performativa QUERY_IF solicitando interromper a avaliação do aluno e ficar em estado de espera. Quando o agente tutor der conta do problema notifica aos agentes avaliadores enviando uma mensagem com a performativa INFORM de que o problema foi resolvido. Os agentes ao receber a mensagem reiniciam a avaliação do aluno.
- **Pânico:** Se em um evento de emergência o agente tutor não consegue dar conta do problema, ele entra em um estado mental emocional de pânico. Automaticamente, envia uma notificação aos outros agentes com performativa QUERY_IF solicitando interromper a avaliação do aluno e ficar em estado de espera. Se a emergência for resolvida, o agente

tutor notifica aos agentes avaliadores enviando uma mensagem com a performativa INFORM de que o problema foi resolvido. Os agentes ao receber a mensagem reiniciam a avaliação do aluno.

- **Conflito:** Se algum dos agentes perceberem algum conflito de qualquer natureza, ele notifica ao agente tutor com uma mensagem com performativa INFORM.

A figura 50 mostra testes de protocolo para "Reportar" e "Pré-Treino". No exemplo (a) é testado o protocolo de comunicação "*Reportar*" no seguinte cenário: um dos agentes avaliadores falha (agente NAV). Ciclicamente, os agentes avaliadores enviam uma mensagem para o agente tutor com conteúdo "*Reportando*". Se após de dois ciclos um dos agentes não se reportar, o tutor tenta comunicação com ele (Se não houver resposta, o agente tutor aciona o protocolo de conflito de falho de agente). No exemplo (b) é testado o protocolo de "*Pré-Treino*". O agente tutor manda um broadcast com performativa "INFORM" e conteúdo "planejando (Treino)". Os agentes avaliadores, em resposta, mudam seus comportamentos ficando em estado de espera.

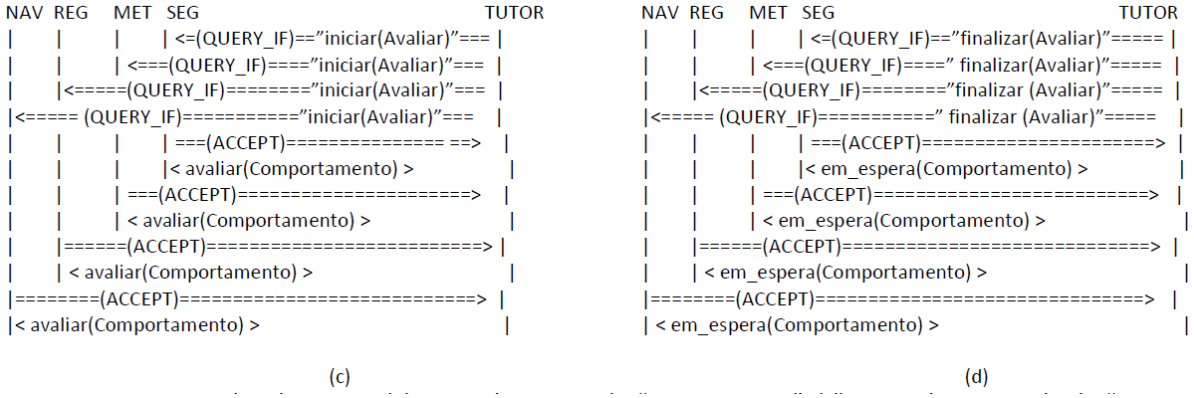
Figura 50- Testes de protocolo para "Reportar"(a) e "Pré-Treino" (b).



Fonte: Autor (2014).

A figura 51 mostra o teste de início de treino e fim de treino. No exemplo (c) é testado o protocolo de "Inicio_Treino". O agente tutor envia um broadcast com a performativa "QUERY_IF" solicitando aos agentes iniciarem a avaliação do aluno. Em resposta, os agentes mudam seus comportamentos (comportamento avaliar). No exemplo (d) é testado o protocolo de "Fim_Treino". O tutor envia um broadcast com a performativa "QUERY_IF" solicitando aos agentes finalizarem a avaliação do aluno.

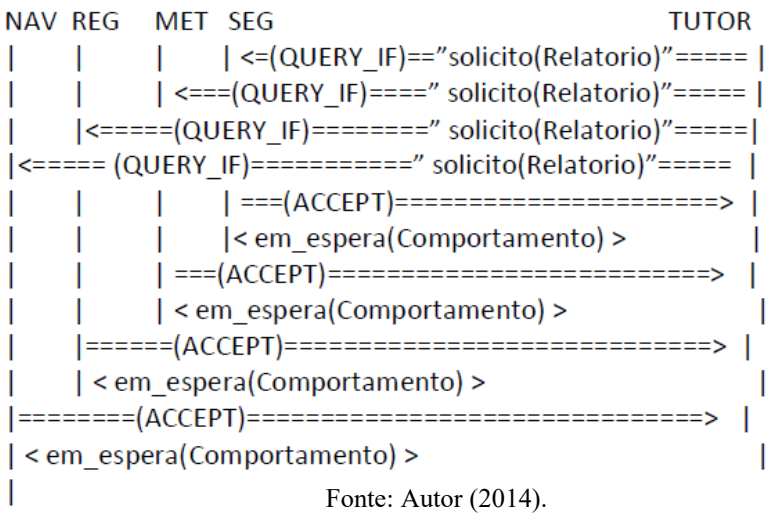
Figura 51- Teste de início de treino (c) e fim de treino (d).



Fonte: Autor (2014).

Na figura 52 é testado o protocolo de “Relatório”. O agente tutor envia um broadcast com a performativa “QUERY_IF”, solicitando aos agentes enviarem o relatório de avaliação.

Figura 52- Teste de solicitação de relatório



Fonte: Autor (2014).

Na figura 53 são testados os protocolos de “Emergência” e de “Pânico” em um cenário que o agente tutor não consegue recuperar o avião e aciona o processador de dinâmica para tomar o controle. Primeiro, um dos agentes percebe uma possibilidade próxima de perda de sustentação (Stall), então ele envia uma mensagem ao agente tutor informando um possível stall. Depois, o agente toma o controle do avião e solicita aos avaliadores interromper a avaliação. Ao perceber que não consegue resolver a emergência, o agente entra em um estado de pânico e aciona o processador de dinâmica (ProcDin). Quando o problema é resolvido o

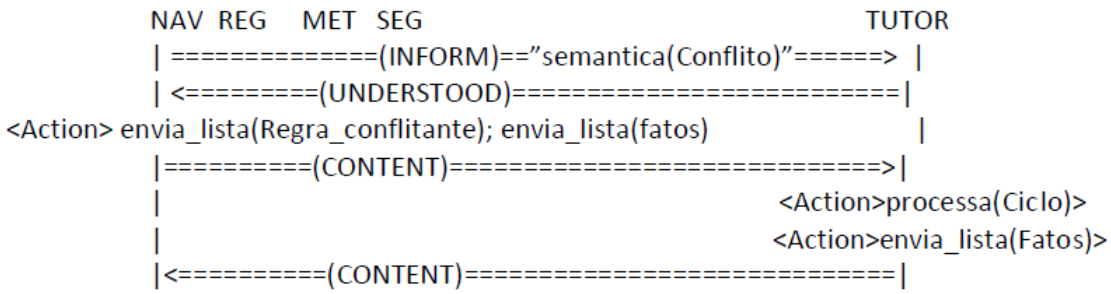
agente tutor informa aos avaliadores que o problema foi resolvido. Em resposta, os avaliadores retornam à tarefa de avaliação.

Figura 53- Teste de emergência e pânico

Fonte: Autor (2014).

A figura 54 mostra o teste do protocolo de “Conflito”. Qualquer um dos agentes que perceberem algum conflito de qualquer tipo, ele reporta o problema ao Tutor. O agente Tutor é quem decide como resolver os conflitos. O exemplo ilustra um cenário em que o agente de navegação NAV não consegue chegar a uma conclusão de raciocínio (erro semântico), porque a base de conhecimento de domínio que ele manipula não abrange certa situação, então, ele informa ao agente tutor o problema, enviando a lista das referências das regras ativadas conflitantes e a lista de fatos do último ciclo de raciocínio. O tutor acessa todas as bases de conhecimento de domínio disponíveis e repete o ciclo de raciocínio. Em seguida, o agente tutor envia a conclusão do raciocínio (lista de fatos) para o agente NAV.

Figura 54- Teste de protocolo para um conflito



Fonte: Autor (2014).

Os protocolos de resolução de conflitos no protótipo são seis:

- **Falho_avaliador:** O falho de algum agente avaliador aciona um protocolo de ajuste de carga de trabalho. O agente tutor procura o agente ativo com menor carga de trabalho (carga de trabalho é proporcional ao tamanho da base de conhecimento que têm que processar) e solicita a ele (QUERY_IF) assumir a ontologia de domínio que correspondia ser processada pelo agente que falhou. Se todos os agentes avaliadores falharem, o agente tutor pode assumir todas as ontologias de domínio, mas as chances de erros semânticos e de sintaxe aumentarão notoriamente.
- **Conflito de recurso:** Se dois ou mais agentes concorrerem aos recursos do mesmo artefato ao mesmo tempo, o agente tutor dará prioridade de acesso ao agente com mais carga de trabalho e deixará os outros na fila de espera (ciclo de até 0.1 segundo para o agente ativo num processador de 1GHZ).
- **Conflito de sintaxe ou de semântica:** Quando um agente percebe um erro de semântica ou de sintaxe, ele notifica ao agente tutor. Sendo que o tutor tem acesso a todas as bases de conhecimento, ele repete o último encadeamento conflitante. Se o problema for resolvido, o tutor notifica a solução ao agente afetado, caso contrário, o tutor notifica do problema ao operador.
- **Conflito de falha no veículo:** Se os objetivos do operador não envolvessem treinar com falhas no veículo, e algum dos agentes perceberem uma falha no veículo, ele notifica ao agente tutor. O agente tutor cede privilégios ao agente de segurança permitindo a comunicação direta com a interface do aluno, assim o agente pode dar orientações ao aluno de como lidar com a emergência.

- **Conflito de falha no simulador externo:** Se algum dos agentes concluírem que o simulador externo está falhando, eles comunicam o evento ao agente tutor. O tutor notifica o evento ao operador humano por meio da interface. O operador pode decidir se cancela ou continua o treinamento.
- **Resolução de Conflitos:** Caso o agente avaliador parar de funcionar (-Falho de avaliador) o agente com menor carga de trabalho assume a tarefa, podendo ser um dos agentes avaliadores que não esteja com objetivos pendentes na fila ou que a fila de ações dele esteja vazia. A carga do tutor aumenta ao assumir as tarefas que saturam os outros agentes, chegando a ter um tempo de ciclo de consulta equivalente a 100ms num computador de 1GHZ.
- **Conflito de decisão:** Se algum dos agentes não conseguir decidir qual plano de ação executar na avaliação, ensino ou controle do veículo (apenas o tutor pode ensinar e controlar o veículo), ele pode acionar o processador de dinâmica para simular as diferentes opções de planos de ações. Ao observar o resultado das simulações poderá tomar uma decisão solida.

APÊNDICE B – VALIDAÇÃO DO SISTEMA FEITA POR UM PILOTO DA COMPANHIA AEREA GOL

B.1- INTRODUÇÃO

O sistema de instrução virtual para pilotos de aviões comerciais Boeing 737-800, usando o simulador XPLANE9 foi desenvolvido para assumir a maior parte das responsabilidades de um instrutor humano.

O protótipo é baseado em conhecimento, ou seja, a funcionalidade e efetividade dependem do conhecimento transmitido pelo projetista (ou engenheiro de conhecimento). Em outras palavras, a qualidade do conhecimento manuseado pelo sistema depende do conhecimento da pessoa que transmite a informação (quem representa o conhecimento).

Neste caso em particular, o conhecimento foi transmitido pelo doutorando para o sistema. O conhecimento transmitido é limitado, já que o doutorando não é especialista no domínio da aviação.

B.2- OBJETIVO PRINCIPAL DA AVALIAÇÃO

Pede-se ao profissional julgar o protótipo, não apenas pela qualidade do conhecimento representado, se não pela potencialidade do protótipo de poder evoluir, se o conhecimento fosse incrementado e evoluído por uma equipe de especialistas e profissionais dos domínios envolvidos.

Pede-se também ao profissional comentar se percebe o instrutor virtual acompanhando o treinamento (se o instrutor virtual reage ao aluno).

B.3-ROTEIRO DE TESTES DE AVALIAÇÃO

Primeira sessão:

Duração aproximada 90 minutos.

Na primeira sessão o profissional observará uma série de voos autônomos feitos pelo sistema. Os pontos a serem avaliados são:

-Procedimentos e manobras da inteligência artificial.

-Decisões feitas pela inteligência artificial

Segunda sessão:

Na segunda sessão o profissional assumirá o papel de aluno treinando no simulador. O profissional fará dois treinamentos, no primeiro terá que executar os procedimentos e manobras sem errar (dentro do possível). No segundo treinamento, o profissional cometerá erros de propósito. Após os treinamentos, deverão ser avaliados os seguintes pontos:

-Planejamento, coerência e sequência de slides;

-Planejamento do roteiro de navegação;

-Conteúdo e coerência do conteúdo da assistência dada pelo instrutor virtual (orientações e recomendações);

-Consistência no roteiro da avaliação final feita pelo sistema.

Questionário da Primeira Sessão de Validação

1. Considere o desempenho do sistema executando as missões (voo autônomo). Como foram feitas as manobras e procedimentos? Responda de acordo com uma escala de 0 a 10 (0: Totalmente erradas, 10:Todo feito à perfeição):

Resposta: 7

Comente sua resposta:

As manobras em geral foram realizadas de acordo com os preceitos e conhecimentos que se esperam de um piloto em fase inicial de treinamento. Alguns erros foram observados, contudo, os conceitos teóricos das manobras foram assimilados, principalmente, daqueles referentes à física e teoria de voo. O piloto automático foi acionado manualmente e logo desabilitado em uma manobra brusca, o sistema conseguiu manter o avião estável.

2. Quais foram os maiores erros de desempenho do sistema?

R: No primeiro momento houve a aplicação equivocada de alguns dispositivos de voo na fase de voo que a aeronave se encontrava, como por exemplo, a utilização do dispositivo hiper-sustentator (*flap*) para baixas velocidades sem o auxílio da potência dos motores.

No retorno para aproximação, o sistema não encontrou corretamente o aeroporto para pouso, assim como não efetuou a aproximação. As técnicas que seriam utilizadas para pouso não puderam ser avaliadas, visto que não houve tal procedimento.

3. Quais procedimentos e manobras foram mais bem executados?

R: O sistema efetuou a decolagem utilizando os recursos reais necessários, com aplicação correta de potência, *pitch* assim como suavidade nos movimentos. Durante a execução das curvas houve a coordenação da aplicação dos controles de voo (direcional, longitudinal e vertical) fazendo com que a aeronave, corretamente, não perdesse altitude na realização da manobra.

4. Considere as decisões feitas pelo sistema nas missões. Como foram as decisões feitas pelo sistema em cada situação? Responda de acordo com uma escala de 0 a 10 (0: Totalmente erradas, 10:Todo feito à perfeição):

Resposta: 8

Comente sua resposta

As decisões feitas pelo sistema correspondem as de um aluno que estivesse no início de seu processo de instrução para piloto de aeronaves de asas fixas, havendo necessidade de um maior conhecimento em relação à teoria das forças que agem sobre a aeronave durante o voo. Esse conhecimento pode ser adquirido estudando e ensinando ao sistema teoria de voo, no qual teremos todas as forças que agem na aeronave, e quais são os meios de melhor se aproveitar tais forças ou em outros casos, como contrariá-las a fim de mantermos o voo estabilizado.

6. Quais são suas recomendações e sugestões para melhorar o protótipo?

R: Para a aplicação proposta inicialmente, de ser um treinador utilizando a aeronave Boeing B737-800, é necessário fortalecer os conceitos teóricos de voo, para que o sistema compreenda e consiga de forma autônoma melhor gerenciar todas as fases de voo. Com a dominância destes conceitos, poderemos iniciar o treinamento de manobras, de como executá-las e, principalmente, de como corrigi-las.

Questionário da Segunda Sessão de Validação

1. Considere os slides apresentados na interface. Qual a opinião da qualidade de conteúdo exposto? Responda de acordo com uma escala de 0 a 10 (0: Insuficiente para o aprendizado, 5:Conteúdo suficiente para um bom treinamento, 10: Conteúdo excessivo ou redundante)

R:4

Comente sua resposta:

Os slides apresentados ainda estão em uma fase inicial de desenvolvimento. O tempo de apresentação deveria ser um pouco maior, e de forma mais clara sobre o que o aluno deve fazer para corrigir a manobra.

Uma melhora no layout da apresentação é necessária, talvez uma antecipação no tempo da mensagem, não deixando que a mesma ocorra num momento crítico do voo. O sistema tem que perceber o seu limite de controle, e esse, teoricamente, deve ser menor que o limite de erro do aluno, ou seja, quando a aeronave se aproximar de uma determinada situação o sistema deve intervir antes que a situação se agrave, informando ao aluno e, nos casos mais extremos, assumindo o controle e evitando a situação indesejável.

2. Considere novamente slides apresentados na interface. Qual a opinião sobre o planejamento da sequência dos slides? Responda de acordo com uma escala de 0 a 10 (0: Inconsistente e confuso, 10: Perfeitamente consistente)

R: 7

Comente sua resposta:

Com o decorrer da missão, as mensagens e o planejamento foram mais temporais, ocorrendo em momentos mais próximos ao desejado num voo de instrução.

3. Considere a assistência (ajuda) do tutor. O sistema oferece ajuda oportunamente? Responda de acordo com uma escala de 0 a 10 (0: Definitivamente não, 10: Ajuda totalmente oportuna)

R: 7

Comente sua resposta:

As ajudas oferecidas foram oportunas. Faltando melhorar a interface, posteriormente, para uma sintetização de voz, pois voar e ter que ler as instruções em uma segunda tela pode comprometer o voo e as manobras.

4. Quando o aluno pede ajuda, considera que o sistema responde adequadamente? Responda de acordo com uma escala de 0 a 10 (0: Definitivamente não, 10: Orientações e recomendações totalmente certeiras e oportunas)

Comente sua resposta:

R: Não testamos este item durante os voos realizados.

5. De acordo aos erros feitos de propósito no segundo treinamento, considera que o sistema os detectou e avaliou corretamente? Responda de acordo com uma escala de 0 a 10 (0: Definitivamente não, 10: Sim, em sua totalidade como esperado).

R:10

Comente sua resposta:

Durante os voos nos quais, voluntariamente, expusemos a aeronave a situações de não normalidade, como atitudes anormais, o sistema corretamente as identificou, reagindo imediatamente ao evento proposto.

6. Percebeu alguma ação do instrutor virtual em resposta ao desempenho ou conduta de você? Responda de acordo com uma escala de 0 a 10 (0: Definitivamente não, 10: Sim, resulta muito evidente).

R: 10

Comente sua resposta :

Quando realizamos experimentalmente erros, o sistema os identificou e buscou corrigi-los de forma rápida.

7. Imagine que o instrutor virtual fosse humano. Você poderia saber ou dizer qual era o estado emocional dele?

Comente sua resposta:

R: Nas sessões realizadas não foi possível avaliar o estado emocional. A realização de mais voos talvez demonstrasse esse sentimento com o acúmulo de erros ou acertos.

7. Quais são as suas sugestões e recomendações para melhorar o sistema?

R: Realização de mais voos para melhor avaliarmos as reações do sistema; com esses resultados aplicarmos melhorias na aplicação de comandos e avaliação do sentimento do instrutor virtual.

8. Achou a interface amigável e fácil de usar? Responda de acordo com uma escala de 0 a 10 (0: Definitivamente não, 10: Sim, muito intuitiva e fácil de usar).

R: 10

Comente a sua resposta:

A interface apresentada é muito real, de fácil compreensão, faltando alguns ajustes como mencionados anteriormente.

8. Em sua opinião, gostou do protótipo? Responda de acordo com uma escala de 0 a 10(0:Definitivamente não, 10: Sim, muito).

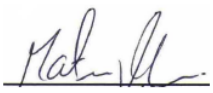
R: 9

Comente a sua resposta:

O protótipo tem um potencial e uma aplicabilidade em diferentes modais de transporte para o treinamento de pessoas, com o apoio e um maior desenvolvimento das tecnologias aplicadas pode trazer grandes benefícios à sociedade.

9. Em sua opinião, e baseado em todas as sessões feitas, você está de acordo que, evoluindo o sistema com a representação de conhecimento adquirida de expertos e profissionais trabalhando em uma equipe dedicada, o instrutor virtual teria um nível de qualidade de conhecimento suficiente para assumir parcialmente as tarefas de um instrutor humano? Acha que seria possível obter uma versão profissional e de alta qualidade do sistema?

R: Com certeza, a evolução deste sistema poderá auxiliar nas tarefas do processo ensino aprendizagem dos pilotos, ou de outras pessoas dependendo do modal de transporte que for adaptado futuramente. O desenvolvimento desse sistema acarretaria em mais uma ferramenta de treinamento, com custos menores para o treinamento de pessoas. Como sugestão para o aperfeiçoamento, seria a criação de um grupo de trabalho formado por pilotos (instrutores de voo), pedagogos e psicólogos que poderiam ajudar a aperfeiçoar a parte técnica do treinamento, assim como ensinar ao sistema a melhor forma de transmitir os conhecimentos e como desenvolver o "emocional" do instrutor virtual, posteriormente, pilotos-alunos, ainda em fase de treinamento em escolas de formação, poderiam fazer voos com seus erros normais à fase que se encontram, para então avaliarmos corretamente o instrutor virtual.



Mateus Rodrigues Ghisleni
Piloto de Boeing 737
Instrutor de Voo
Bacharel em Ciências Aeronáuticas – Habilitação Piloto Comercial – PUCRS
mateus.ghisleni@gmail.com

APÊNDICE C- AQUICISÃO DE CONHECIMENTO

O domínio de interesse é o avião de passageiros BOEING 737-800 [BOEING, 1997] e todos os tópicos que envolvem a operação do avião, sendo eles operação da aeronave, procedimentos de voo, teoria de voo, navegação, regulamento aeronáutico, meteorologia, segurança no voo, comportamento dinâmico, modelo qualitativo, sistemas do avião e malhas de controle.

Nesta etapa foi coletado um grande volume de conhecimento por meio de entrevistas com profissionais e pesquisa de fontes bibliográficas. Neste capitulo é mostrado apenas um resumo dos tópicos abordados nas entrevistas e uma síntese do conteúdo mais relevante pesquisado nas fontes bibliográficas.

Os sistemas baseados em conhecimento têm como filosofia principal captar o conhecimento e a experiência do humano especializado para resolver problemas em domínios específicos. Por isso é importante insistir em evitar representar conhecimentos baseados unicamente em pesquisas bibliográficas.

A aquisição de conhecimento para representar a ontologia de conteúdo e ontologia de domínio consistiu em recopilar conhecimento principalmente de especialistas humanos por meio de entrevistas.

Grande parte da aquisição de conhecimento para a ontologia de modelo foi feita com base em pesquisa bibliográfica devido ao denso conteúdo matemático bem entendido, bem estabelecido, normalizado e aceito.

C.1. AQUISIÇÃO DE CONHECIMENTO BASEADO EM ENTREVISTAS

Neste apartado será apresentada apenas uma síntese dos conhecimentos recopilados mencionando os tópicos abordados em cada entrevista, devido à grande quantidade de informação que seria impossível mostrar em um capitulo só. O foco das entrevistas foi recopilar conhecimentos (priorizados no avião Boeing 737-800) nos domínios de controle do avião, comandos, procedimentos, navegação, meteorologia, regulamento aeronáutico e procedimentos de emergência.

Tópico principal: Avião Boeing 737-800
Entrevistado: Técnico em manutenção de Boeing 737-800.
Sessões: 3. Resumo dos conhecimentos recolhidos
Tópico abordado: Características do Avião.
Dimensões, peso, variáveis de desempenho (autonomia, velocidade máxima, etc.)
Tópico abordado: Painéis, instrumentos, rádios e mandos da cabine.
Painel superior, painel do piloto automático, painel central e painel de navegação.
Tópico abordado: Sistemas.
Computadores, sistema elétrico, sistema hidráulico, sistema pneumático, motores.

Tópico principal: Teoria de voo

Descrição: A teoria de voo é a base com que os pilotos iniciam qualquer curso de aviação privada, comercial ou militar.

Entrevistado: Instrutor de voo.

Sessões: 4. Resumo dos conhecimentos recolhidos

Tópico abordado: Superfícies de controle.

Ailerons, leme, elevadores, compensadores, flapes, slats, spoilers, speedbrakes.

Tópico abordado: Ângulo de ataque

Influência no Arrasto e na Sustentação, angulo de sustentação, estol.

Tópico abordado: Sustentação

Dependência da velocidade, altitude e densidade do ar. Ângulo de sustentação

Tópico abordado: Estol.

Estol no voo nivelado e nas curvas, recuperação ao estol.

Tópico abordado: Voo Horizontal.

Ângulo de ataque crítica, potência disponível, potência necessária, velocidade máxima, velocidade de máximo alcance, velocidade de máxima autonomia, velocidade mínima, velocidade de estol, velocidade de teto, velocidade verdadeira, velocidade indicada.

Influência das variações da velocidade, do ângulo de ataque, do peso, da altitude e da potência necessária.

Potência/Velocidade

Procedimentos de ganho/perda de altitude e de velocidade, procedimentos de recuperação.

Tópico abordado: Descida e Voo Planado.

Ângulo de planeio, efeito dos flapes no ângulo de planeio, ângulo de ataque/ângulo de planeio, ângulo de ataque/velocidade de máximo alcance, ângulo ataque/ velocidade de máxima autonomia.

Velocidade final e velocidade limite, influência do peso do avião, influência do vento de proa e vento de cauda, influência da altitude.

Procedimentos de descida, rampas de descida, transição de descida.

Tópico abordado: Voo em Subida.

Velocidade de máxima razão de subida, velocidade de máximo ângulo de subida, teto prático, teto absoluto.

Relação entre altitude, potência máxima e potência necessária.

Rampa de Ascenso, transição de subida.

Tópico abordado: Voo em curva

Efeito da força centrípeta, compensação da sustentação, Efeito na virada com a variação de velocidade e do raio da curva, efeito do peso do avião, glissada e derrapagem.

Evitar Estol, recuperar do estol.

Tópico principal: Navegação

Descrição: A teoria da navegação aeronáutica tem como propósito a orientação espacial (rumo, proa, rota) e temporal (fuso horário) no voo visual e no voo por instrumentos, considerando aspectos como desvios por causa do vento, rotas alternativas em caso de emergências, planificação do voo (plano de voo), etc.

Entrevistado: Professor escola aviação.

Sessões: 5. Resumo dos conhecimentos recolhidos

Tópico abordado: Conceitos

Sistemas de Coordenadas, Paralelos, Meridianos, Rota Ortodromia, Rota Ortodrómica, Proa, Rumo, Rota, Linha Agônica, Dmg, Inclinação Magnética, Linhas Isoclínicas, Linhas Isopóricas, Rumo Verdadeiro, Rumo Magnético, Proa Verdadeira, Proa Magnética, Proa Bússola. Fuso Horário, Hora UTC, Hora local HLO, Fuso O, Fuso P, Fuso Q, Fuso R.

Tópico abordado: Instrumentos Primários

Altímetro, Altitude de Pressão, Altitude Indicada, Altitude Densidade, Altitude Verdadeira, Altitude Absoluta, Ajuste QNE, Ajuste QNH.

Indicador de Subida ou Descida (Climb Rate), Velocímetro, Velocidade Indicada, Velocidade Calibrada, Velocidade Equivalente, Velocidade Verdadeira, Velocidade no Solo.

Glydescopio, Atitude (Pitch) e Rolagem (ROLL), Horizonte Artificial.

Tópico abordado: Navegação por Instrumentos

Navegação VOR, sintonização da VOR, leitura do DME, captura de Radial, curvas DME

Navegação NDB, sintonização no ADF, orientação e curvas.

Navegação GPS, programação no FMS

Aproximação ILS, entrada na rampa, procedimentos.

Tópico abordado: Leitura e Interpretação de cartas

SID, STAR, Rota (ERC), Área (ARC)

Tópico principal: Regulamento Aeronáutico

Descrição: A teoria do regulamento aeronáutico tem como propósito dar as diretrizes das convenções internacionais para trafego aéreo para normalizar as comunicações entre todas as partes envolvidas (pilotos, controladores, etc), e estabelecer regras de trafego e procedimentos para prevenir incidentes e acidentes no voo e no solo.

Entrevistado: Controlador aéreo.

Sessões: 4. Resumo dos conhecimentos recolhidos

Tópicos

Configuração das pistas de pouso, área de manobra e lousa dos aeródromos/Aeroportos.

Regras de Voo Visual e Voo por Instrumentos

Regras no Solo, Regras no Ar.

Níveis de Voo

Prevenção de Colisões: Proximidade, Direito de Passagem, Ultrapassagem, Direito de Pouso e Direito de Decolagem.

Situações de Emergência: Socorro, Urgência

Aerovias: Regras nas aerovias, dimensões.

Estrutura do Espaço Aéreo: Divisão Vertical, Designação das Áreas, Controle Aéreo, Classificação dos Espaços Aéreos.

Serviços de Tráfego Aéreo (ATS)

Controle de Aeródromo/Aeroporto, Luzes, sinalizações, Faróis, Sistemas Visuais de Aproximação.

Esteiras de Turbulência entre Aviões.

Controle de Aproximação, Controle de Área, Controle em Rota.

Plano de Voo (Para inserir no FMS).

Fraseologia Padrão.

Tópico principal: Meteorologia

Descrição: A teoria do regulamento aeronáutico tem como propósito dar as diretrizes das convenções internacionais para trafego aéreo para normalizar as comunicações entre todas as partes envolvidas (pilotos, controladores, etc), e estabelecer regras de trafego e procedimentos para prevenir incidentes e acidentes no voo e no solo.

Entrevistado: Meteorologista.

Sessões: 4. Resumo dos conhecimentos recolhidos

Conceitos

Latitude tropical, equatorial, temperada e polar, camadas da atmosfera, ar seco, húmido, saturado.

Temperatura do AR: na superfície, em altitude, em voo (IAT:Temperatura Indicada, CAT: Temperatura calibrada, TAT: Temperatura verdadeira).

Inversão Térmica.

Pressão Atmosférica, Sistemas de Pressão, Cartas Sinóticas.

Atmosfera Padrão (ISA), Condição ISA, Altitudes:Altura(QFE), Altitude Indicada(QHN), Altitude-Pressão(QFE).

Medida de Humidade, Umidade Relativa e Absoluta.

Tópicos

Altimetria: Altímetro, ajustes altímetricos, altitude de transição, nível de transição, erros altimétricos, altitude-densidade.

Atmosfera padrão.

Leitura do Radar Meteorológico.

Tipos de nuvens e nevoeiros, turbulências, vento de tesoura. Formação de gelo, trovoadas. Estabilidade Meteorológica.

Circulação dos ventos. Massas de Ar e Frentes.

Interpretação do METAR, SPECI, SIGMET, AIRMET. Código TAF. Cartas SIG-WX-PROG

Tópico principal: Procedimentos

Descrição: Procedimentos padrões dão orientações de seqüências de ações e manobras que o piloto deve realizar em cada etapa do voo.

Entrevistado: Piloto de linha aérea.

Sessões: 5. Resumo dos conhecimentos recolhidos

Inspeção de painéis: É um procedimento de checagem na cabine do avião. Checam-se os itens essenciais antes de conectar a APU ou External Power.

Preparação de cabine: São procedimentos executados na sua totalidade em cada início de voo ou troca de tripulação, ou após trabalho da manutenção.

Preflight check list: Verificação que deve ser feita antes do push-back .

Pre-Start (Preparando para ligar motores): Procedimentos que devem ser feitos antes de ligar motores.

Pre-Start check list. Verificação que deve ser feita antes de ligar motores.

Procedimento de ligação de motores.

After start checklist

Antes do taxi: Procedimentos que devem serem executados antes de levar o avião até a pista.

Before taxi checklist

Procedimentos para taxi (levar o avião da louça até a cabeceira da pista)

Before Takeoff check list: Verificação que deve ser feita antes de decolar.

Procedimentos de decolagem

After Takeoff checklist: Verificação que deve ser feita logo após decolar

Procedimentos de subida (Climb)

Procedimentos de voo cruzeiro.

Procedimentos de descida.

Approach check list: Verificação que deve ser feita antes da aproximação ao aeroporto.

Procedimentos de aproximação.

Landing check list: Verificação que deve ser feita antes de pousar.

Procedimentos de pouso ou arremetida.

Procedimentos de Taxi (levar o avião da pista até a lousa e estacionar).

Procedimentos de desligamento.

Shutdown check list: Verificação que deve ser feita no desligamento do avião.

Procedimentos em caso de turbulência.

Procedimentos em caso de falha em um dos motores (ou em ambos).

Procedimentos em caso de trem de pouso travado.

Procedimentos em caso de superfície de controle travada.

C.2. AQUISIÇÃO DE CONHECIMENTO COM FONTES BIBLIOGRÁFICAS

A aquisição de conhecimento desde fontes bibliográficas foi feita com o propósito de representar a ontologia de modelo do avião Boeing 737-800. Os agentes utilizam a ontologia de modelo para verificar a coerência da simulação e procurar falhas no avião. O processador de modelo utiliza a ontologia de modelo e a ontologia de dinâmica para fazer simulações do comportamento do avião e eventualmente controlar o avião nas situações de assistência ao aluno e na emergência.

C.2.1 Equações da Dinâmica do Avião

Devido a grande extensão da formulação matemática para obter as funções de transferência do avião, neste capitulo será apresentado uma síntese dos pontos mais importantes do analise. Mais detalhes podem ser consultados nos livros "*Airplane Flight Dynamics & Automatic Flight Controls: Part I & II*" de Roskam [ROSKAM,1998]. Baseado no modelo dedutivo de Roskam, presume-se que o avião é um contínuo de elementos de massa *dm*. Cada elemento é referenciado á origem do sistema de eixos X'Y'Z' (considerado inercial) por meio do vetor posição r'. Cada elemento de massa é sujeito á gravidade g, orientada ao longo do eixo Z'.

Define-se vetor $\omega = (P_{\hat{\imath}}, Q_{\hat{\jmath}}, R_{\hat{k}})$ composto pelas componentes da velocidade angular, taxa de rolamento *(P)*, taxa de arfagem(Q) e taxa de guinada *(R)*. Define-se também o vetor $\vec{V}_P = (U_{\hat{\imath}}, V_{\hat{\jmath}}, W_{\hat{k}})$ cujas componentes são de velocidade linear. A equação (1) pode se desmembrar em componentes vetoriais como segue:

$$\left.\begin{array}{l} \text{Força ao longo de X: } \mathrm{m}\left(\dot{\mathrm{U}} - \mathrm{VR} + \mathrm{WQ}\right) = \mathrm{mg_x} + \mathrm{F_{Ax}} + \mathrm{F_{Tx}} \\ \text{Força ao longo de Y: } \mathrm{m}\left(\dot{\mathrm{V}} + \mathrm{UR} - \mathrm{WP}\right) = \mathrm{mg_y} + \mathrm{F_{Ay}} + \mathrm{F_{Ty}} \\ \text{Força ao longo de Z: } \mathrm{m}\left(\dot{\mathrm{W}} - \mathrm{UQ} + \mathrm{VP}\right) = \mathrm{mg_z} + \mathrm{F_{Az}} + \mathrm{F_{Tz}} \end{array}\right\} \quad (1)$$

Onde F_A é o vetor da força aerodinâmica resultante e F_T o vetor da tração resultante. Define-se o vetor momento aerodinâmico $\overrightarrow{M_A} = (L_{A\hat{\imath}}, M_{A\hat{\jmath}}, N_{A\hat{k}})$ composto pelas componentes: momento aerodinâmico da rolagem (L), momento aerodinâmico da arfagem (M) e momento aerodinâmico da guinada (N). Define-se também o vetor tração $\overrightarrow{M_T} = (L_{T\hat{\imath}}, M_{T\hat{\jmath}}, N_{T\hat{k}})$. A equação (2) que representa pode se desmembrar como segue:

$$\left.\begin{array}{l} Momento\ de\ rolamento\ ao\ redor\ de\ X: I_{XX}\dot{P} - I_{XZ}\left(\dot{R} + PQ\right) + (I_{ZZ} - I_{YY})RQ = L_{A\hat{\imath}} + L_{T\hat{\imath}} \\ Momento\ de\ arfagem\ ao\ redor\ de\ Y: I_{YY}\dot{Q} - I_{XZ}(R^2 - P^2) + (I_{XX} - I_{ZZ})PR = M_{A\hat{\jmath}} + M_{T\hat{\jmath}} \\ Momento\ de\ guinada\ ao\ redor\ de\ Z: I_{ZZ}\dot{R} - I_{XZ}\left(\dot{P} - QR\right) + (I_{YY} - I_{XX})PQ = N_{A\hat{\imath}} + N_{T\hat{\imath}} \end{array}\right\} \quad (2)$$

Se o vetor $\vec{r} = (x_{\hat{\imath}}, y_{\hat{\jmath}}, z_{\hat{k}})$ representa as distancias que definem a localização dos elementos se massa da aeronave, as inércias estão determinadas como segue:

$$\left.\begin{aligned} I_{XY} = I_{YX} = 0;\ (\text{simetrías}) \\ I_{XX} = \iiint (y^2 + z^2)\rho_A dV \\ I_{YY} = \iiint (x^2 + z^2)\rho_A dV \\ I_{ZZ} = \iiint (x^2 + y^2)\rho_A dV \\ I_{XZ} = I_{ZX} = \iiint xz\rho_A dV \end{aligned}\right\} \quad (3)$$

Os ângulos de Euler definem o ângulo $\psi \in [0, 2\pi]$ de guinada (yaw), o ângulo $\theta \in [-\pi/2, \pi/2]$ de arfagem ou atitude (pitch) e o ângulo $\Phi \in [-\pi, \pi]$ de rolamento (roll).

A trajetória de voo referente à terra $\vec{r}' = (x', y', z')$ pode se determinar com a seguinte relação matricial:

$$\begin{bmatrix} \dot{x}' \\ \dot{y}' \\ \dot{z}' \end{bmatrix} = \begin{bmatrix} cos\,\psi & -sen\,\psi & 0 \\ sen\,\psi & cos\,\psi & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} cos\,\theta & 0 & sen\,\theta \\ 0 & 1 & 0 \\ -sen\,\theta & 0 & cos\,\theta \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 \\ 0 & cos\,\phi & -sen\phi \\ 0 & sen\,\phi & cos\,\phi \end{bmatrix} \begin{bmatrix} U \\ V \\ W \end{bmatrix} \quad (4)$$

As relações entre a velocidade angular e a taxa de variação dos ângulos de Euler estão determinadas como segue (equações cinemáticas):

$$\left.\begin{aligned} Taxa\ de\ rolamento\ em\ relação\ a\ X: P = \dot{\phi} - \dot{\psi} sen\theta \\ Taxa\ de\ arfagem\ em\ relação\ a\ Y: Q = \dot{\theta} cos\phi + \dot{\psi} cos\theta sen\phi \\ Taxa\ de\ guinada\ em\ relação\ a\ Z: R = \dot{\psi} cos\theta cos\phi - \dot{\theta} sen\phi \end{aligned}\right\} \quad (5)$$

Os componentes da força gravitacional podem se expressar em função dos ângulos de Euler:

$$\left.\begin{aligned} mgx = -mgsen\theta \\ mgy = mgsen\phi cos\theta \\ mgz = mgcos\phi cos\theta \end{aligned}\right\} \quad (6)$$

Reescrevendo a equação (1) em termos de (6) tem se que:

$$\left.\begin{aligned} m(\dot{U} - VR + WQ) = -mgsen\theta + F_{Ax} + F_{Tx} \\ m(\dot{V} + UR - WP) = mgsen\phi cos\theta + F_{Ay} + F_{Ty} \\ m(\dot{W} - UQ + VP) = mgcos\phi cos\theta + F_{Az} + F_{Tz} \end{aligned}\right\} \quad (7)$$

Há duas condições de voo que são de particular interesse:

- Voo em regime, quando todas as variáveis de movimento permanecem constantes ($\vec{V}_P = 0\ e\ \vec{\omega} = 0$) em referencia ao sistema de coordenadas XYZ.
- Voo perturbado, quando todas as variáveis de movimento são relativamente definidas a uma condição definida de voo em estado estacionário (somatória de estados estacionários e perturbações).

Nas equações de movimento é adicionado o subscrito “1” a todas as variáveis de movimento para indicar que a incógnita é uma variável de estado estacionário. Logo as equações de movimento em voo estacionário se descrevem como segue:

$$\left.\begin{array}{l}\text{Força ao longo de X: } \mathrm{m}(-\mathrm{V_1R_1} + \mathrm{W_1Q_1}) = -mgsen\theta_1 + \mathrm{F_{Ax_1}} + \mathrm{F_{Tx_1}} \\ \text{Força ao longo de Y: } \mathrm{m}(\mathrm{U_1R_1} - \mathrm{W_1P_1}) = mgsen\phi_1 cos\theta_1 + \mathrm{F_{Ay_1}} + \mathrm{F_{Ty_1}} \\ \text{Força ao longo de Z: } \mathrm{m}(-\mathrm{U_1Q_1} + \mathrm{V_1P_1}) = mgcos\phi_1 cos\theta_1 + \mathrm{F_{Az_1}} + \mathrm{F_{Tz_1}}\end{array}\right\} \quad (8)$$

$$\left.\begin{array}{c}Momento\ de\ rolamento\ ao\ redor\ de\ X\text{:}\ \ -I_{XZ}(P_1Q_1) + (I_{ZZ} - I_{YY})R_1Q_1 = L_{A\hat{\imath}} + L_{T\hat{\imath}} \\ Momento\ de\ arfagem\ ao\ redor\ de\ Y\text{:}\ -I_{XZ}({P^2}_1 - {R^2}_1) + (I_{XX} - I_{ZZ})P_1R_1 = M_{A\hat{\jmath}} + M_{T\hat{\jmath}} \\ Momento\ de\ guinada\ ao\ redor\ de\ Z\text{:}\ -I_{XZ}(R_1Q_1) + (I_{YY} - I_{XX})P_1Q_1 = N_{A\hat{\imath}} + N_{T\hat{\imath}}\end{array}\right\} \quad (9)$$

$$\left.\begin{array}{c}Taxa\ de\ rolamento\ em\ rela\tilde{c}ao\ a\ X\text{:}\ P_1 = \dot{\phi}_1 - \dot{\psi}_1 sen\theta_1 \\ Taxa\ de\ arfagem\ em\ rela\tilde{c}ao\ a\ Y\text{:}\ Q_1 = \dot{\theta}_1 cos\phi_1 + \dot{\psi}_1 cos\theta_1 sen\phi_1 \\ Taxa\ de\ guinada\ em\ rela\tilde{c}ao\ a\ Z\text{:}\ R_1 = \dot{\psi}_1 cos\theta_1 cos\phi_1 - \dot{\theta}_1 sen\phi_1\end{array}\right\} \quad (10)$$

C.2.2 Forças e momento aerodinâmicos para pequenas perturbações

Supondo que todas as forças e momento perturbados são funções dos valores instantâneos das variáveis de movimento perturbadas, tem que para forças e momento longitudinais as equações são:

$$\left.\begin{array}{l}f_{ax} = \left(\frac{u}{U_1}\right)\frac{\partial F_{ax}}{\partial\left(\frac{u}{U_1}\right)} + \alpha\frac{\partial F_{ax}}{\partial\alpha} + \left(\frac{\dot{\alpha}\bar{c}}{2U_1}\right)\frac{\partial F_{ax}}{\partial\left(\frac{\dot{\alpha}\bar{c}}{2U_1}\right)} + \left(\frac{q\bar{c}}{2U_1}\right)\frac{\partial F_{ax}}{\partial\left(\frac{q\bar{c}}{2U_1}\right)} + \delta e\frac{\partial F_{ax}}{\delta e} + \delta f\frac{\partial F_{ax}}{\delta f} \\ f_{az} = \left(\frac{u}{U_1}\right)\frac{\partial F_{az}}{\partial\left(\frac{u}{U_1}\right)} + \alpha\frac{\partial F_{az}}{\partial\alpha} + \left(\frac{\dot{\alpha}\bar{c}}{2U_1}\right)\frac{\partial F_{az}}{\partial\left(\frac{\dot{\alpha}\bar{c}}{2U_1}\right)} + \left(\frac{q\bar{c}}{2U_1}\right)\frac{\partial F_{az}}{\partial\left(\frac{q\bar{c}}{2U_1}\right)} + \delta e\frac{\partial F_{az}}{\delta e} + \delta f\frac{\partial F_{az}}{\delta f} \\ m_a = \left(\frac{u}{U_1}\right)\frac{\partial M_a}{\partial\left(\frac{u}{U_1}\right)} + \alpha\frac{\partial M_a}{\partial\alpha} + \left(\frac{\dot{\alpha}\bar{c}}{2U_1}\right)\frac{\partial M_a}{\partial\left(\frac{\dot{\alpha}\bar{c}}{2U_1}\right)} + \left(\frac{q\bar{c}}{2U_1}\right)\frac{\partial M_a}{\partial\left(\frac{q\bar{c}}{2U_1}\right)} + \delta e\frac{\partial M_a}{\delta e} + \delta f\frac{\partial M_a}{\delta f}\end{array}\right\} \quad (11)$$

Para forças e momento lateral-direcionais as equações são:

$$\left.\begin{array}{l}f_{aY} = \beta\frac{\partial F_{ay}}{\partial\beta} + \left(\frac{pb}{2U_1}\right)\frac{\partial F_{ay}}{\partial\left(\frac{pb}{2U_1}\right)} + \left(\frac{\dot{\beta}\bar{c}}{2U_1}\right)\frac{\partial F_{ay}}{\partial\left(\frac{\dot{\beta}\bar{c}}{2U_1}\right)} + \left(\frac{rb}{2U_1}\right)\frac{\partial F_{ay}}{\partial\left(\frac{rb}{2U_1}\right)} + \delta a\frac{\partial F_{ay}}{\delta a} + \delta r\frac{\partial F_{ay}}{\delta r} \\ l_a = \beta\frac{\partial L_a}{\partial\beta} + \left(\frac{pb}{2U_1}\right)\frac{\partial L_a}{\partial\left(\frac{pb}{2U_1}\right)} + \left(\frac{\dot{\beta}\bar{c}}{2U_1}\right)\frac{\partial L_a}{\partial\left(\frac{\dot{\beta}\bar{c}}{2U_1}\right)} + \left(\frac{rb}{2U_1}\right)\frac{\partial L_a}{\partial\left(\frac{rb}{2U_1}\right)} + \delta a\frac{\partial L_a}{\delta a} + \delta r\frac{\partial L_a}{\delta r} \\ n_a = \beta\frac{\partial N_a}{\partial\beta} + \left(\frac{pb}{2U_1}\right)\frac{\partial N_a}{\partial\left(\frac{pb}{2U_1}\right)} + \left(\frac{\dot{\beta}\bar{c}}{2U_1}\right)\frac{\partial N_a}{\partial\left(\frac{\dot{\beta}\bar{c}}{2U_1}\right)} + \left(\frac{rb}{2U_1}\right)\frac{\partial N_a}{\partial\left(\frac{rb}{2U_1}\right)} + \delta a\frac{\partial N_a}{\delta a} + \delta r\frac{\partial N_a}{\delta r}\end{array}\right\} \quad (12)$$

C.2.3 Funções de Transferência Longitudinais

Para as equações de voo perturbado longitudinal tem se:

$$\left.\begin{array}{c} \dot{u} = -g\theta cos\Theta_1 + X_u u + X_{T_u} u + X_\alpha \alpha + X_{\delta_e}\delta e \\ U_1\dot{\alpha} - U_1\dot{\theta} = -g\theta sin\Theta_1 + Z_u u + Z_\alpha \alpha + Z_{\dot{\alpha}}\dot{\alpha} + Z_q\dot{\theta} + Z_{\delta_e}\delta e \\ q = M_u u + M_{T_u} u + M_\alpha \alpha + M_{\dot{\alpha}}\dot{\alpha} + M_{T_\alpha}\alpha + M_{\delta_e}\delta e + M_q\dot{\theta} \end{array}\right\} (13)$$

Aplicando a transformada de *Laplace*, obtém se:

$$\left.\begin{array}{c} u(s)\big(s - X_{u-}X_{T_u}\big) - X_\alpha \alpha(s) + gcos\Theta_1\theta(s) = X_{\delta_e}\delta e(s) \\ -Z_u u(s) + [s(U_1\dot{\alpha} - Z_{\dot{\alpha}}) - Z_\alpha]\alpha(s) + [-(Z_q + U_1)s + g\theta sin\Theta_1]\theta(s) = Z_{\delta_e}\delta e(s) \\ -\big(M_u + M_{T_u}\big)u(s) - \big(M_{\dot{\alpha}}s + M_\alpha + M_{T_\alpha}\big)\alpha(s) + \big(s^2 + M_q s\big)\theta(s) = M_{\delta_e}\delta e(s) \end{array}\right\} (14)$$

Representando (14) em forma matricial:

$$\begin{pmatrix} \big(s - X_{u-}X_{T_u}\big) & -X_\alpha & gcos\Theta_1 \\ -Z_u & [s(U_1\dot{\alpha} - Z_{\dot{\alpha}}) - Z_\alpha] & [-(Z_q + U_1)s + g\theta sin\Theta_1] \\ -(M_u + M_{T_u}) & -(M_\alpha + M_{\dot{\alpha}}s + M_{T_\alpha}) & s^2 - M_q s \end{pmatrix} \begin{pmatrix} \frac{u(s)}{\delta e(s)} \\ \frac{\alpha(s)}{\delta e(s)} \\ \frac{\theta(s)}{\delta e(s)} \end{pmatrix}$$

$$= \left.\begin{pmatrix} X_{\delta_e} \\ Z_{\delta_e} \\ M_{\delta_e} \end{pmatrix}\right\} (15)$$

Sendo este sistema a representação longitudinal do avião em malha aberta. Deste sistema é obtido o grupo de funções de transferência longitudinais (usando método de Gauss). Essas funções são a velocidade em função ao elevador ($u/\delta e$), arfagem em função ao elevador ($\theta/\delta e$), e o ângulo de ataque alpha em função do elevador ($\alpha/\delta e$) onde elas têm a forma:

$$\frac{u}{\delta e} = \frac{A_u S^3 + B_u S^2 + C_u S + D_u}{A_1 S^4 + B_1 S^3 + C_1 S^2 + D_1 S + E_1} \quad (16)$$

$$\frac{\theta}{\delta e} = \frac{A_\theta S^2 + B_\theta S + C_\theta}{A_1 S^4 + B_1 S^3 + C_1 S^2 + D_1 S + E_1} \quad (17)$$

$$\frac{\alpha}{\delta e} = \frac{A_\alpha S^3 + B_\alpha S^2 + C_\alpha S + D_\alpha}{A_1 S^4 + B_1 S^3 + C_1 S^2 + D_1 S + E_1} \quad (18)$$

E de forma indireta, obtém se mais três funções de transferência; essas funções são a altitude em função ao elevador ($h/\delta e$), velocidade em função do throttle ($u/\delta t$), e a velocidade função da deflexão dos spoilers($u/\delta s$), onde elas tem a forma:

$$\frac{h}{\delta e} = \frac{A_h S^3 + B_h S^2 + C_h S + D_h}{S(A_1 S^4 + B_1 S^3 + C_1 S^2 + D_1 S + E_1)} \quad (19)$$

$$\frac{u}{\delta t} = \frac{A_t S^3 + B_t S^2 + C_t S + D_t}{A_1 S^4 + B_1 S^3 + C_1 S^2 + D_1 S + E_1} \quad (20)$$

$$\frac{u}{\delta s} = \frac{A_s S^3 + B_s S^2 + C_s S + D_s}{A_1 S^4 + B_1 S^3 + C_1 S^2 + D_1 S + E_1} \quad (12)$$

C.2.4 Funções de transferência laterais-direcionais (perturbadas)

Para um voo perturbado lateral-direcional tem-se:

$$\left.\begin{aligned} m(\dot{v} + U_1 r) &= mg\Phi cos\Theta_1 + \bar{q}_1 S(C_{y\beta}\beta + C_{yp}\frac{pb}{2U1} + C_{yr}\frac{rb}{2U1} + C_{y\delta a}\delta_a + C_{y\delta r}\delta_r) \\ I_{xx}\dot{p} - I_{xz}\dot{r} &= \bar{q}_1 Sb\left(C_{l\beta}\beta + C_{lp}\frac{pb}{2U1} + C_{lr}\frac{rb}{2U1} + C_{l\delta a}\delta_a + C_{l\delta r}\delta_r\right) \\ I_{zz}\dot{r} - I_{xz}\dot{p} &= \bar{q}_1 Sb\left(C_{n\beta}\beta + C_{np}\frac{pb}{2U1} + C_{nr}\frac{rb}{2U1} + C_{n\delta a}\delta_a + C_{n\delta r}\delta_r + C_{nT\beta}\beta\right) \end{aligned}\right\} (13)$$

Aplicando transformada de *Laplace*:

$$\left.\begin{aligned} (sU_1 - Y_\beta\beta)\beta(s) - (sY_p + gcos\Theta_1)\Phi(s) + (U_1 - Y_r)\Psi(s) &= Y_{\delta r}\delta_r \\ -L_\beta\beta(s) + (s^2 - L_p s)\Phi(s) - (s^2\bar{A}_1 - sL_r)\Psi(s) &= L_{\delta r}\delta_r \\ -(N_\beta N_{T\beta})\beta(s) + (s^2\bar{B}_1 - sN_p)\Phi(s) + (s^2 - N_r s)\Psi(s) &= N_{\delta r}\delta_r \end{aligned}\right\} (14)$$

Podendo se representar em forma matricial:

$$\begin{pmatrix} (sU_1 - Y_\beta\beta) & (sY_p + gcos\Theta_1) & (U_1 - Y_r) \\ -L_\beta & (s^2 - L_p s) & -(s^2\bar{A}_1 - sL_r) \\ -(N_\beta N_{T\beta}) & (s^2\bar{B}_1 - sN_p) & (s^2 - N_r s) \end{pmatrix} \begin{pmatrix} \frac{\beta(s)}{\delta_r(s)} \\ \frac{\Phi(s)}{\delta_r(s)} \\ \frac{\Psi(s)}{\delta_r(s)} \end{pmatrix} = \begin{pmatrix} Y_{\delta r} \\ L_{\delta r} \\ N_{\delta r} \end{pmatrix} (15)$$

Do sistema matricial pode se obter três funções de transferência; a derrapagem em função do leme ($\beta/\delta r$), guinada em função do leme ($\psi/\delta r$), e o rolamento em função do leme ($\Phi/\delta r$), onde:

$$\frac{\beta}{\delta r} = \frac{A_{\beta r} S^3 + B_{\beta r} S^2 + C_{\beta r} S + D_{\beta r}}{A_2 S^4 + B_2 S^3 + C_2 S^2 + D_2 S + E_2} \quad (16)$$

$$\frac{\psi}{\delta r} = \frac{A_{\psi r} S^3 + B_{\psi r} S^2 + C_{\psi r} S + D_{\psi r}}{A_2 S^4 + B_2 S^3 + C_2 S^2 + D_2 S + E_2} \quad (17)$$

$$\frac{\phi}{\delta r} = \frac{A_{\phi r} S^3 + B_{\phi r} S^2 + C_{\phi r} S + D_{\phi r}}{A_2 S^4 + B_2 S^3 + C_2 S^2 + D_2 S + E_2} \quad (18)$$

Podem se obter também em forma indireta a derrapagem em função do aileron ($\beta/\delta a$), guinada em função do aileron ($\psi/\delta a$), e o rolamento em função do aileron ($\phi/\delta a$), onde:

$$\frac{\beta}{\delta a} = \frac{A_\beta S^3 + B_\beta S^2 + C_\beta S + D_\beta}{A_2 S^4 + B_2 S^3 + C_2 S^2 + D_2 S + E_2} \quad (19)$$

$$\frac{\psi}{\delta a} = \frac{A_\psi S^3 + B_\psi S^2 + C_\psi S + D_\psi}{A_2 S^4 + B_2 S^3 + C_2 S^2 + D_2 S + E_2} \quad (20)$$

$$\frac{\phi}{\delta a} = \frac{A_\phi S^3 + B_\phi S^2 + C_\phi S + D_\phi}{A_2 S^4 + B_2 S^3 + C_2 S^2 + D_2 S + E_2} \quad (21)$$

Os diferentes termos representados por A, B, C e D são termos variáveis obtidos por meio da interface do simulador X-PLANE9.