MINISTÉRIO DA CIÊNCIA E TECNOLOGIA
INSTITUTO NACIONAL DE PESQUISAS ESPACIAIS

INPE-14092-TDI/1075

APLICAÇÃO DA TECNOLOGIA DE AGENTES DE PLANEJAMENTO EM OPERAÇÕES DE SATÉLITES

Luciana Sêda Cardoso

Dissertação de Mestrado do Curso de Pós-Graduação em Computação Aplicada, orientada pelos Drs. Mauricio Gonçalves Vieira Ferreira e Valcir Orlando, aprovada em 14 de março de 2006.

INPE
São José dos Campos
2006

681.3.019

CARDOSO, L. S.
Aplicação da tecnologia de agentes de planejamento em operações de satélites / L. S. Cardoso.- São José dos Campos: INPE, 2006.
167p. ; (INPE-14092-TDI/1075).

1.Inteligência artificial. 2.Planejamento. 3.Planos de vôo. 4.Representação do conhecimento. 5. Projeto de sistemas computacionais. I.Título.

Aprovado (a) pela Banca Examinadora em cumprimento ao requisito exigido para obtenção do Título de **Mestre** em

Computação Aplicada

Dr. Nilson Sant'Anna

Presidente / INPE / SJCampos - SP

Dr. Valcir Orlando

Orientador(a) / INPE / SJCampos - SP

Dr. Mauricio Gonçalves Vieira Ferreira

Orientador(a) / INPE / SJCampos - SP

Dr. José Carlos Becceneri

Membro da Banca / INPE / SJCampos - SP

Dr. José Demisio Simões da Silva

Membro da Banca / INPE / SJCampos - SP

Dr. Henrique Jesus Quintino de Oliveira

Convidado(a) / UMC / Mogi das Cruzes - SP

Aluno (a): **Luciana Sêda Cardoso Gonçalves**

São José dos Campos, 14 de março de 2006

“A mente que se abre a uma nova idéia jamais volta ao seu tamanho original.”

ALBERT EINSTEIN

À minha família,
o maior tesouro que tenho.

AGRADECIMENTOS

É muito bom na conclusão de um trabalho, dar uma parada e olhar o caminho percorrido. Muitos foram os obstáculos vencidos e é com alegria que percebo que não poderia estar escrevendo estas palavras se não fosse a ajuda recebida de Deus e das várias pessoas que me rodearam durante estes anos dando uma idéia aqui, um incentivo ali e porque não, uns puxões de orelha. Por isso, agradeço a todas as pessoas que me incentivaram e ajudaram a realizar este trabalho.

À Divisão de Desenvolvimento de Sistemas de Solo – DSS, e em especial a Edenilse F. E. Orlandi, pela oportunidade de estudos e utilização de suas instalações.

Ao Centro de Rastreio e Controle - CRC, em especial aos colegas de trabalho, Odair Aparecido de Oliveira e Jun Tominaga, que não mediram esforços em passar a experiência e o conhecimento adquiridos nas operações dos satélites do INPE.

Aos meus orientadores, Prof. Dr. Mauricio G. V. Ferreira e Prof. Dr. Valcir Orlando, pela orientação e amizade.

A meu esposo, Paulo Eduardo Cardoso, pelo apoio e paciência nas minhas crises de ansiedade e pela revisão deste trabalho.

A meu filho, Leonardo Cardoso Gonçalves, que do jeitinho dele torceu para que tudo desse certo.

A meus pais que sempre acreditaram na importância do estudo e que nunca deixaram de me apoiar.

E a todos os meus colegas de trabalho, que estavam sempre dispostos a me escutar e trocar idéias.

RESUMO

A busca de solução de automatização aplicada às operações espaciais é uma necessidade mundial para diminuir os custos das missões. No Instituto Nacional de Pesquisas Espaciais (INPE) uma grande parte das atividades de operação dos satélites é ainda feita de forma manual. Por isso, encontrar caminhos de automatização das atividades de operação de satélites do INPE é de extrema importância para conseguir manter o bom desempenho nas operações de rastreio dos satélites mesmo com a escassez de recursos financeiros disponíveis. Com este enfoque voltado à automatização de operações espaciais, este trabalho propõe uma arquitetura de um sistema de Planejamento Inteligente de Planos de Operação de Vôo (PlanIPOV) que emprega a tecnologia de planejamento da Inteligência Artificial (IA) na geração automática dos Planos de Operação de Vôo (POVs) para a fase operacional de rotina de satélites artificiais. O sistema PlanIPOV tem como propostas o uso da Linguagem de Definição do Domínio de Planejamento (PDDL) para modelagem da base de conhecimento do domínio de rastreio de satélites e também a geração automática dos arquivos de problemas. Um protótipo foi desenvolvido baseado na arquitetura PlanIPOV usando a linguagem PDDL2.2 e o planejador Local Searching for Planning – Timed initial literal and Derived predicates (LPG-TD). Este protótipo foi testado para o domínio de rastreio dos satélites atualmente controlados pelo INPE: SCD1, SCD2 e CBERS2. Os POVs gerados pelo protótipo foram comparados com os POVs reais e os resultados obtidos foram satisfatórios. Durante o desenvolvimento da base de conhecimento e do protótipo, foram encontradas algumas limitações na aplicação da tecnologia de planejamento e soluções alternativas tiveram que ser adotadas para possibilitar a geração automática de POVs. Adicionalmente, esta pesquisa produziu um processo para a modelagem de uma base de conhecimento para um problema de planejamento genérico e um *framework* de classes que tem como objetivo gerar arquivos de problemas de forma automática.

APPLYING THE PLANNING AGENT TECNOLOGY IN SATELLITE OPERATIONS

ABSTRACT

Research for automatic solutions for space operations is a real need for all space agencies in order to reduce space mission costs. Nowadays, a significant parcel of satellite operation activities at the National Institute for Space Research (INPE) is still performed manually. Finding automated alternatives for the satellite operation activities at INPE is an important issue in order to maintain the currently satisfactory performance of these activities, despite the scarcity of financial resources. In this work an architecture of an Intelligent Planning System for the automatic generation of satellite flight operation plans (PlanIPOV) is proposed. This architecture employs the planning technology of Artificial Intelligence (AI) in the automatic Flight Operation Plans (FOPs) generation for the routine operational phase of a satellite. The PlanIPOV system proposes the use of Planning Domain Definition Language (PDDL) to model the knowledge base of satellite tracking domain and also the automatic generation of problem files. A prototype was developed based on the PlanIPOV architecture using the PDDL2.2 language and Local Searching for Planning – Timed initial literal and Derived predicates (LPG-TD) planner. It was tested for the tracking domain of the satellites currently being controlled by INPE (SCD1, SCD2 and CBERS2). The FOPs generated by the prototype were compared with the real INPE FOPs and the obtained results were satisfactory. During the knowledge base and prototype development, some limitations were observed in the utilization of the planning technology and solutions were adopted to overcome them to allow the automatic generation of FOP. Additionally, the research work leads to the development of a process to model a knowledge base for a generic planning problem and a framework of classes, which has the aim to generate automatic problem files.

SUMÁRIO

Pág.

LISTA DE FIGURAS

LISTA DE TABELAS

LISTA DE SIGLAS E ABREVIATURAS

CAPÍTULO 1 INTRODUÇÃO 25
1.1 Motivação do Trabalho de Pesquisa 25
1.2 Objetivos do Trabalho de Pesquisa 30
1.3 Metodologia de Desenvolvimento do Trabalho de Pesquisa 30
1.4 Esboço Geral 31

CAPÍTULO 2 AUTOMATIZAÇÃO NA OPERAÇÃO DE SISTEMAS ESPACIAIS 35
2.1 Definição de um Sistema Espacial com Ênfase no Segmento Solo 35
2.2 Conceitos de Operação de Missão 38
2.3 Automatização nas Operações de Missões Espaciais 43

CAPÍTULO 3 TECNOLOGIA DE AGENTE DE PLANEJAMENTO 47
3.1 Definição de Agente 47
3.2 Definição de Agente de Planejamento 51
3.2.1 Técnicas de Planejamento 55
3.2.1.1 Técnica - Planejamento em Grafo 57
3.2.2 Linguagens de Representação de Problemas de Planejamento 66
3.2.2.1 Linguagem PDDL 68
3.2.3 Planejador LPG-TD 79
3.3 Engenharia de Conhecimento 81

CAPÍTULO 4 SOLUÇÃO PARA GERAÇÃO DE PLANOS DE OPERAÇÕES DE VÔO PARA SATÉLITES USANDO AGENTE DE PLANEJAMENTO 85
4.1 Preparação e Execução da Operação de Satélites no INPE 85
4.2 Arquitetura do Sistema de Planejamento Inteligente de Planos de Operações de Vôo 90
4.3 Processo para Modelagem do Domínio 96
4.3.1 Especificação clara do ambiente de tarefa do agente 97
4.3.2 Aquisição de conhecimento do domínio de planejamento 98
4.3.3 Definição da versão da linguagem PDDL a ser usada e do agente de planejamento 99
4.3.4 Especificação da ontologia do domínio 100
4.3.5 Codificação do arquivo de domínio na linguagem PDDL 102
4.3.6 Validação do domínio 103
4.4 *Framework* para Geração de Planos Usando a Linguagem PDDL 105

CAPÍTULO 5 IMPLEMENTAÇÃO DE UM PROTÓTIPO DA ARQUITETURA 117
5.1 Opção de Configuração da Base de Dados 121
5.2 Opção de Geração de POVs 123

CAPÍTULO 6 APRESENTAÇÃO E ANÁLISE DOS RESULTADOS OBTIDOS 127
6.1 Teste e Resultados Obtidos 127
6.2 Limitações Encontradas na Utilização da Linguagem PDDL 2.2 e Soluções Adotadas 134

CAPÍTULO 7 CONCLUSÃO 139
7.1 Contribuições 139
7.1.1 Pontos Favoráveis 139
7.1.2 Pontos Desfavoráveis 140
7.1.3 Principais Contribuições 142
7.2 Trabalhos Futuros 143
7.3 Considerações Finais 144

REFERÊNCIAS BIBLIOGRÁFICAS 147

GLOSSÁRIO 155

APÊNDICE A
ARQUIVO DE DOMÍNIO CODIFICADO EM PDDL 2.2 PARA O DOMÍNIO DE RASTREIO DE SATÉLITE DO INPE 157

APÊNDICE B
EXEMPLO DE UM ARQUIVO DE PROBLEMAS CODIFICADO EM PDDL 2.2 GERADO AUTOMATICAMENTE PELO PROTÓTIPO 165

LISTA DE FIGURAS

2.1 - Relacionamento entre o Segmento Espacial, o Segmento Solo e o Cliente 36
2.2 - Estrutura do Segmento Solo 38
2.3 - Atividades Típicas de uma Passagem 40
3.1 - Agentes interagem com ambientes por meio de sensores e atuadores 49
3.2 - Agente baseado em objetivos 52
3.3 - Agente de Planejamento Clássico 53
3.4 - Representação de um grafo de planejamento 58
3.5 – Representação gráfica da definição ***mutex*** 60
3.6 - Pseudocódigo do algoritmo de Planejamento em Grafo 61
3.7 – Grafo de Planejamento para o problema do jantar-surpresa 63
3.8 – Resultado de um plano encontrado na extração da solução 66
3.9 - Base de conhecimento em PDDL 70
3.10 - Exemplo do arquivo de domínio de satélite 71
3.11 - Definição da ação *turn-to* 73
3.12 - Exemplo do arquivo de problema para o domínio de satélite. 74
3.13 – Ambiente de Geração automática de planos 76
3.14 - Plano gerado para o domínio de satélite 77
3.15 - Exemplo de definição de evento exógeno no arquivo de problema 78
3.16 – Plano para o domínio de satélite respeitando o evento exógeno 79
4.1 - Preparação e Execução do POV 86
4.2 – Diagrama de contexto do sistema PlanIPOV 92
4.3 - Arquitetura do sistema PlanIPOV 93
4.4 – Arquitetura de teste do domínio de rastreio de satélite 104
4.5 – Diagrama de Classes 106
4.6 – Definição da classe CDomain 107
4.7 – Definição da classe CProblem 108
4.8 – Diagrama de classes do domínio de rastreio de satélite 110
4.9 – Definição da classe CPOVPassConfiguration 112
4.10 – Diagrama de seqüência de geração do POV 115
5.1 – Diagrama de Componente de Execução do POVGenerator 118
5.2 – Diagrama de componente de construção do POVGenerator 119
5.3 – Tela principal do protótipo. 120
5.4 – Tela do Editor da base de dados de Configuração. 121
5.5 – Tela de Configuração de Parâmetros de Passagem. 122
5.6 – Tela atualizada com as informações extraídas do arquivo PVP fornecido 123
5.7 – Tela com o resultado dos POVs gerados para as passagens selecionadas 124
5.8 – Tela com o conteúdo do POV selecionado 125
6.1 – POV gerado para a passagem 60509-4 do satélite SCD1 133
6.2 – Prioridades entre os eventos exógenos. 136
6.3 – Prioridades nos sub-objetivos 138

LISTA DE TABELAS

TABELA 3.1 – Descrição do problema do jantar-surpresa. 62
TABELA 5.1 – Artefatos do protótipo x módulos do PlanIPOV. 117
TABELA 6.1 – Resultado da Comparação entre os POVs gerados e os POVs reais... 129

LISTA DE SIGLAS E ABREVIATURAS

ADL	-	*Action Description Language*
ALC	-	Alcântara
APSS	-	*Activity Planning and Sequencing Subsystem*
ASE	-	*Autonomous Science Experiment*
ASPEN	-	*Automated Scheduling and Planning Environment*
CBA	-	Cuiabá
CBERS	-	*China Brazil Earth Resource Satellites*
CCM	-	Centro de Controle de Missão
CCS	-	Centro de Controle de Satélites
CLEaR	-	*Closed Loop Execution and Recovery*
CM	-	Controle de Missão
CRC	-	Centro de Rastreio e Controle
CX	-	*Citizen eXplorer*
EM	-	Exploração da Missão
EO1	-	*Earth Orbiter* 1
EQUARS	-	*Equatorial Atmosphere Research Satellite*
ESOC	-	*European Space Operations Centre*
ET	-	Estação Terrena
GRAPHPLAN	-	Planejador em Grafo
HSP	-	Planejador de Pesquisa Heurística
IA	-	Inteligência Artificial

INPE	-	Instituto Nacional de Pesquisas Espaciais
INTELMOD	-	*INTELligent MODeller*
IPC	-	*International Planning Competition*
ISO	-	*International Organization for Standardization*
LEOP	-	Fase de Lançamento e Órbitas Preliminares
LPG	-	*Local Searching for Planning Graphs*
LPG-TD	-	*Local Searching for Planning Graphs – Timed initial literal and Derived predicates*
MAPGEN	-	*Mixed-initiative Activity Plan GENerator*
MECB	-	Missão Espacial Completa Brasileira
MER	-	*Mars Exploration Rover*
MIPS	-	*Model checking Integrated Planning System*
MIRAX	-	Monitor e Imageador de Raios-X
NASA	-	*National Aeronautics and Space Administration*
NET	-	Rede de Comunicação de Dados
PDDL	-	*Planning Domain Definition Language*
PlanIPOV	-	Planejamento Inteligente de Planos de Operação de Vôo
POP	-	Planejamento em Ordem Parcial
POS	-	Plano de Operação de Solo
POV	-	Plano de Operação de Vôo
PVP	-	Plano de Previsão de Passagem
OO	-	Orientada a Objeto
SATPLAN	-	Planejamento por Satisfatibilidade

SCCU	-	Sistema de Controle de Carga Útil
SCD	-	Satélite de Coleta de Dados
SCO	-	Sistema de Controle de Operações
SE	-	Segmento espacial
SEM	-	Sistema de Exploração da Missão
SGPlan	-	*Subgoal partitioning and Global search Planner*
SICS	-	Sistema de Controle de Satélites
SPACEOPS	-	*International Conference on Space Operations*
SRI	-	*Stanford Research Institute*
SSR	-	Satélite de Sensoriamento Remoto
STRIPS	-	*Stanford Research Institute Problem Solver*
TALPlanner	-	*Temporal Action Logics Planner*
TilSAPA	-	*Timed Initial Literals in Temporal Planning*
TLPlan	-	*Temporal Logical Plan*
UFO1	-	*Navy UHF Follow On 1*

CAPÍTULO 1

INTRODUÇÃO

Atualmente uma das maiores preocupações da comunidade que lida com atividades espaciais é o alto custo para manter as equipes de operação das missões espaciais. Cada vez mais as atenções estão voltadas a encontrar soluções de automatização de forma a reduzir o custo total das missões através da diminuição das equipes de operação e a buscar maior confiabilidade das operações. Esta tendência pôde ser observada no Congresso Internacional de Operações Espaciais (SPACEOPS/ 2004).

Uma tarefa estratégica na área de controle de solo de satélites artificiais é o planejamento das atividades operacionais envolvidas, de forma a usar o mínimo de recursos e gerar o máximo de produtos da missão espacial (dados científicos, dados tecnológicos, imagens e telecomunicação), sem comprometer a segurança do satélite. Essa tarefa é complexa, pois envolvem tomadas de decisões.

Uma solução para automatização é a tecnologia de planejamento da área de Inteligência Artificial (IA). Esta técnica permite a geração de planos (seqüências de ações) de forma automatizada para se alcançar um objetivo. Na visão de IA, um algoritmo de planejamento pode ser concebido como um agente de software que sente o ambiente e atua gerando uma seqüência de ações para se alcançar o objetivo desejado.

1.1 Motivação do Trabalho de Pesquisa

O Instituto Nacional de Pesquisas Espaciais (INPE) tem como um de seus principais objetivos o domínio de tecnologia espacial através do desenvolvimento e operação de satélites artificiais. Dada a extensão do território brasileiro, os satélites se constituem em uma importante ferramenta para várias aplicações de grande interesse para a população do país, como por exemplo: a monitoração de vegetação, plantações, rios e clima; o levantamento de recursos naturais; e telecomunicações.

No momento, o INPE tem três satélites em operação: SCD1, SCD2 (Satélites de Coleta de Dados) e CBERS-2 (China-Brazil Earth Observation Satellites).

Os satélites de coleta de dados SCD1 e SCD2 foram integralmente desenvolvidos pelo INPE, como parte da chamada Missão Espacial Completa Brasileira (MECB). Ambos têm como objetivo receber os dados transmitidos por estações automáticas de coleta de dados espalhadas pelo território brasileiro e retransmiti-los às estações de rastreio brasileiras (Cuiabá e Alcântara), onde são gravados. Após cada passagem de um dos satélites sobre uma estação de rastreio (uma passagem, em média, a cada período orbital, o que significa, aproximadamente, uma passagem a cada 100 minutos) os dados gravados na estação são transmitidos ao Centro de Controle da Missão (CCM), em Cachoeira Paulista, onde são processados e, a seguir, disponibilizados aos usuários da missão.

O satélite CBERS-2 é o segundo satélite de sensoriamento remoto desenvolvido em conjunto com a China dentro do programa *China Brazil Earth Resource Satellites*. Este satélite encontra-se em uma órbita polar heliosíncrona, cuja característica principal é passar por uma mesma latitude geográfica sempre no mesmo horário local, o que significa sob mesmas condições de iluminação solar. Esta característica é adequada a sua missão de observação da Terra (imageamento).

Para os próximos anos estão previstos os lançamentos dos satélites EQUARS (*Equatorial Atmosphere Research Satellite*) e MIRAX (Monitor e Imageador de RaiosX) que fazem parte do programa de micro-satélites científicos; os satélites SSR-1 e SSR-2 (Satélites de Sensoriamento Remoto) que utilizarão a plataforma multi-missão que está sendo projetada no INPE; e, além disso, o acordo com a China foi renovado para o desenvolvimento e lançamento de mais três satélites de sensoriamento remoto (CBERS-2B, CBERS-3 e CBERS-4).

Na estrutura organizacional formal do INPE, a entidade interna responsável pelas atividades de rastreio e controle em órbita de satélites é o Centro de Rastreio e Controle de Satélites (CRC), que é composto pelo Centro de Controle de Satélites (CCS), em São José dos Campos, SP e pelas estações de rastreio de Cuiabá (CBA), MT e de Alcântara

(ALC), MA. Estes três locais estão interligados por uma rede privada de comunicação de dados, que permite a troca de informações entre estas unidades. A localização geográfica da estação de Cuiabá, próximo ao centro geodésico da América do Sul, possibilita a cobertura de praticamente todo o seu território. A estação de Alcântara, por estar localizada nas vizinhanças do Centro de Lançamento de Alcântara, permite o rastreio de satélites lançados a partir deste centro desde o instante de injeção em órbita.

O contato do sistema de controle de solo com o satélite é estabelecido pelas estações de rastreio quando este passa sobre a região de visibilidade de suas antenas. Durante períodos de visibilidade o sinal transmitido pelo satélite é captado pela antena da estação, ficando estabelecido um enlace descendente de comunicação. O sinal recebido do satélite contém as informações de telemetria que revelam seu estado atual de funcionamento. Após o estabelecimento do enlace descendente, a estação estabelece também um enlace ascendente, que é utilizado para envio de telecomandos e execução de medidas de rastreio (medidas de distância e de velocidade).

Durante períodos de visibilidade de satélites, o CCS conecta-se à estação de rastreio, através da rede de comunicação de dados, ficando habilitado a receber dados de telemetria do satélite visível e a transmitir telecomandos a este, em tempo real. Dois grandes sistemas de software foram desenvolvidos no INPE para o controle de seus satélites: o **software aplicativo de tempo real** e o **software de dinâmica orbital**.

O software aplicativo de tempo real é dedicado às atividades que exigem a interação direta dos operadores com os satélites durante os sucessivos períodos de visibilidade. Implementa, basicamente, as seguintes funções:

- Receptação, decodificação, visualização e armazenamento da telemetria (dados transmitidos pelo satélite que informam o estado dos seus subsistemas) gerada pelos satélites;

- Geração e encaminhamento de telecomandos (comandos remotos enviados de solo para os subsistemas do satélite);

- Execução de medidas de rastreio (medidas de distância e de velocidade).

O software de dinâmica orbital opera em tempo não real, a partir de arquivos de dados de telemetria e medidas de rastreio gravados pelo software aplicativo de tempo real durante passagens de satélites. As medidas de rastreio são utilizadas no processo de determinação de órbita (Kuga e Kondapalli, 1993), através do qual o conhecimento da órbita do satélite é atualizado periodicamente e mantido dentro de requisitos de precisão preestabelecidos. Os resultados da determinação de órbita servem de entrada ao processo de propagação de órbita que, com auxílio de modelos dinâmicos do movimento orbital, gera previsões de órbita do satélite. Estas previsões permitem o cálculo de dados necessários ao rastreio de futuras passagens do satélite sobre as estações terrenas. Os valores previstos dos ângulos de apontamento de antena, calculados neste processo, são suficientemente precisos para garantir a pronta aquisição do sinal do satélite em passagens que só ocorrerão vinte dias, ou mais, após sua geração. Estas previsões de passagem são também utilizadas para a geração da programação de cada ação de controle a ser executada em cada passagem de satélite sobre estações de rastreio. Uma programação individual é gerada para cada satélite, sob a forma de um roteiro de operação em rotina denominado Plano de Operação em Vôo (POV), que abrange o período de uma semana.

As operações de controle dos satélites são realizadas por equipes de operação bem treinadas, responsáveis pelas atividades em cada unidade da infra-estrutura de solo do INPE. Essas equipes executam fielmente as ações listadas no correspondente POV para a operação de cada um dos satélites controlados. A execução dessas ações não é, presentemente, automatizada. Elas são executadas manualmente pelos operadores, a partir dos consoles de operação da sala de controle do CCS. A maior parte do trabalho desses operadores concentra-se na fase de operação em rotina dos satélites. Com a previsão do aumento do número de satélites a serem controlados nos próximos anos, e com a diminuição dos recursos financeiros destinados a operação de satélites, a busca de solução de automatização se faz necessária.

Um primeiro passo neste sentido foi o desenvolvimento, no CCS, de programas que geram automaticamente o POV de um dado satélite, a partir de um arquivo de predição de passagens gerado pelo software de dinâmica orbital. Deve-se salientar, porém, que se

trata de programas desenvolvidos particularmente para cada satélite. Isto significa que para cada novo satélite a ser controlado um novo programa de geração do POV deve ser desenvolvido ou, pelo menos um dos programas existentes pode ser adaptado, caso os satélites tenham características semelhantes. Esses programas não foram desenvolvidos a partir da aplicação de nenhuma metodologia especializada à geração de planejamento de atividades e nem empregam qualquer técnica de IA. Trata-se de programas, conforme mencionado acima, completamente dedicados à geração de planos de atividades do satélite específico para o qual foi desenvolvido. Outro inconveniente relativo ao tipo de solução adotada na criação desses programas diz respeito ao problema de seleção da estação terrena que será utilizada no rastreio de um dado satélite quando mais de um satélite são simultaneamente visíveis por uma dada estação, ou quando um satélite é visível simultaneamente por mais de uma estação. Devido à solução ora adotada se for necessária a inserção de uma nova estação terrena no ambiente de rastreio e controle, implicará na atualização de todos os programas de geração do POV para cada satélite controlado pelo CRC. A atualização destes programas implica na modificação de um número grande de linhas de código contendo uma complexa e entrelaçada estrutura de comandos condicionais.

Outro problema relativo aos programas atuais de geração de POV é que os programas foram desenvolvidos dentro do próprio CCS, por especialistas em controle de satélites, e por nenhum especialista em desenvolvimento de sistemas de software. Deste modo, não foi adotado e seguido nenhum paradigma de engenharia de software, o que se refletiu principalmente em não ter sido gerada uma documentação adequada sobre os programas gerados. Como a vida de um satélite, a exemplo do SCD1, pode ultrapassar 10 anos o pessoal que desenvolveu tais programas pode não se encontrar mais na instituição, quando uma adaptação, ou mesmo desenvolvimento de um novo programa para um novo satélite for necessário. Com a falta de documentação sobre os programas este problema tenderia a se tornar particularmente grave.

Pelos motivos acima expostos e comentados, a busca de um caminho que facilite o planejamento das operações dos satélites desvinculando o ambiente de rastreio e os objetivos a serem atingidos no plano do comportamento do domínio de rastreio de

satélite é de grande proveito para o CRC. Além disso, a automatização do processo de geração dos POVs, através de um processo genérico e com características de IA constitui-se de um importante passo para a automatização futura também do processo de execução dos POVs.

1.2 Objetivos do Trabalho de Pesquisa

Esta pesquisa visa buscar um caminho de automatização para as operações de satélite do INPE. Propõe-se a criação de um sistema que permita a geração automática de POVs para fase operacional de rotina dos satélites controlados, que é a fase mais longa e importante desse tipo de artefato, já que, praticamente, se confunde com a sua própria vida útil. Além disso, as atividades operacionais na fase de rotina são bastante repetitivas, o que favorece a adoção de soluções automatizadas de execução.

O objetivo deste trabalho de pesquisa é aplicar a tecnologia de planejamento da IA (Russell e Norvig, 2004), que é dedicada à resolução de problemas de planejamento, em problemas reais que envolvem restrições de tempo e recursos, como é o caso do planejamento de operação de satélites. A comunidade de planejamento deseja que mais aplicações de planejadores em problemas do mundo real sejam desenvolvidas de forma a colher resultados que auxiliem o avanço da tecnologia de planejamento na direção de oferecer soluções aos desafios que as aplicações reais apresentam (Fox e Long, 2003).

1.3 Metodologia de Desenvolvimento do Trabalho de Pesquisa

Na primeira etapa deste trabalho, a pesquisa se concentrou no levantamento da bibliografia existente sobre as soluções de automatização que foram ou estão sendo adotadas pela comunidade espacial, bem como o estado da arte na tecnologia de planejamento e também um levantamento da situação atual das atividades de operação de satélites no INPE.

Em seguida, foram implementados protótipos para avaliar a Linguagem de Definição do Domínio de Planejamento (PDDL). Na literatura pesquisada sobre agentes de

planejamento constatou-se que esta é a linguagem mais recentemente desenvolvida, que está sendo adotada como uma linguagem-padrão de planejamento e é capaz de modelar o problema de planejamento para a operação de satélite que envolva restrições de tempo e recurso.

Após o estudo aprofundado da linguagem PDDL, passou-se a fase de engenharia do conhecimento que teve como resultado a base de conhecimento modelada para o domínio de rastreio de satélite codificado na linguagem PDDL 2.2.

Uma vez definido o modelo do comportamento do domínio de rastreio de satélite e com isso, levantadas todas as vantagens e as limitações do uso da técnica de planejamento, foram elaborados o projeto e a implementação de um protótipo do sistema de Planejamento Inteligente de Plano de Operação de Vôo (PlanIPOV) de satélites artificiais para a fase de rotina. O PlanIPOV foi projetado para ser utilizado na geração dos POVs para os vários satélites atualmente controlados pelo INPE, considerando as estações terrenas envolvidas no processo. O projeto foi desenvolvido usando a metodologia orientada a objeto (OO).

Encerrada a fase de desenvolvimento, foram feitos testes, utilizando a base de dados reais dos satélites do INPE, com o objetivo de avaliar os planos gerados pelo PlanIPOV tendo como base de comparação os planos reais existentes no CRC.

Finalmente, o trabalho de pesquisa realizado foi expresso nesta Dissertação, que tem a sua estrutura detalhada a seguir.

1.4 Esboço Geral

Além do capítulo introdutório atual, este trabalho contém outros seis capítulos adicionais, descritos a seguir:

- *CAPÍTULO 2 – AUTOMATIZAÇÃO NA OPERAÇÃO DE SISTEMAS ESPACIAIS*: neste Capítulo é apresentada à definição de um sistema espacial, os

conceitos de operação de uma missão espacial e o caminho apontado pela comunidade espacial para o uso de automatização nas missões espaciais.

- *CAPÍTULO 3 – TECNOLOGIA DE AGENTES DE PLANEJAMENTO*: neste Capítulo são abordados os conceitos de agente inteligente, de agente de planejamento, as técnicas de planejamento e as linguagens de representação de problemas de planejamento. A técnica de planejamento em grafo e a linguagem PDDL são descritas com mais detalhe. É abordada também a engenharia do conhecimento que orienta a criação de uma base de conhecimento.

- *CAPÍTULO 4 – SOLUÇÃO PARA GERAÇÃO DE PLANOS DE OPERAÇÕES DE VÔO PARA SATÉLITES USANDO AGENTE DE PLANEJAMENTO*: neste Capítulo, primeiramente, é mostrado em linhas gerais como atualmente é preparada e executada a operação de satélites no CRC do INPE na fase de rotina. Em seguida, é apresentada a solução proposta para a geração automática de POVs para satélites usando a tecnologia de agente de planejamento, o sistema de Planejamento Inteligente de Planos de Operação de Vôo (PlanIPOV). Também é apresentado um processo para a modelagem de uma base de conhecimento para um problema de planejamento usando a linguagem PDDL e um *framework* de classes que tem como objetivo gerar arquivos de problemas de forma automática que faz uso da linguagem PDDL.

- *CAPÍTULO 5 – IMPLEMENTAÇÃO DE UM PROTÓTIPO DA ARQUITETURA:* este Capítulo inclui a descrição do protótipo que foi implementado para a geração automática de POVs para os satélites rastreados atualmente pelo INPE baseado na arquitetura do sistema PlanIPOV proposta.

- *CAPÍTULO 6 – APRESENTAÇÃO E ANÁLISE DOS RESULTADOS OBTIDOS:* neste Capítulo são apresentados os resultados obtidos durante os testes realizados para a validação do protótipo. Bem como, as limitações encontradas na aplicação da tecnologia de planejamento durante a implementação do protótipo e as soluções adotadas para resolver estas limitações.

- *CAPÍTULO 7 – CONCLUSÃO*: neste Capítulo são apresentadas as conclusões e os trabalhos futuros vislumbrados.

CAPÍTULO 2
AUTOMATIZAÇÃO NA OPERAÇÃO DE SISTEMAS ESPACIAIS

Neste Capítulo é apresentada a definição de um sistema espacial, os conceitos de operação de uma missão espacial e o caminho apontado pela comunidade espacial para o uso de automatização nas missões espaciais.

2.1 Definição de um Sistema Espacial com Ênfase no Segmento Solo

Um sistema espacial é desenvolvido para atender os requisitos de uma missão espacial específica. Por exemplo: missões espaciais com enfoque em telecomunicações, em sensoriamento remoto, em coleta de dados, ou na realização de experimentos científicos e tecnológicos. Um sistema espacial é composto de dois segmentos: segmento espacial e o segmento solo.

O segmento espacial é a espaçonave que compreende a plataforma de serviço (conjunto de subsistemas projetados para sustentar a operação da missão em órbita) e as cargas úteis (conjunto de equipamentos dedicados às aplicações da missão espacial). O Segmento Solo, por sua vez, consiste de todo *hardware*, *software*, pessoas e procedimentos envolvidos na preparação e execução das atividades de operação remota da missão (ECSS-E-70 , 2000).

O segmento solo é aquele que faz o elo entre o segmento espacial e o cliente da missão, ou seja, os usuários que têm interesse nos dados de carga útil transmitidos pelo satélite. A Figura 2.1 exemplifica o relacionamento entre o segmento espacial, o segmento solo e os usuários.

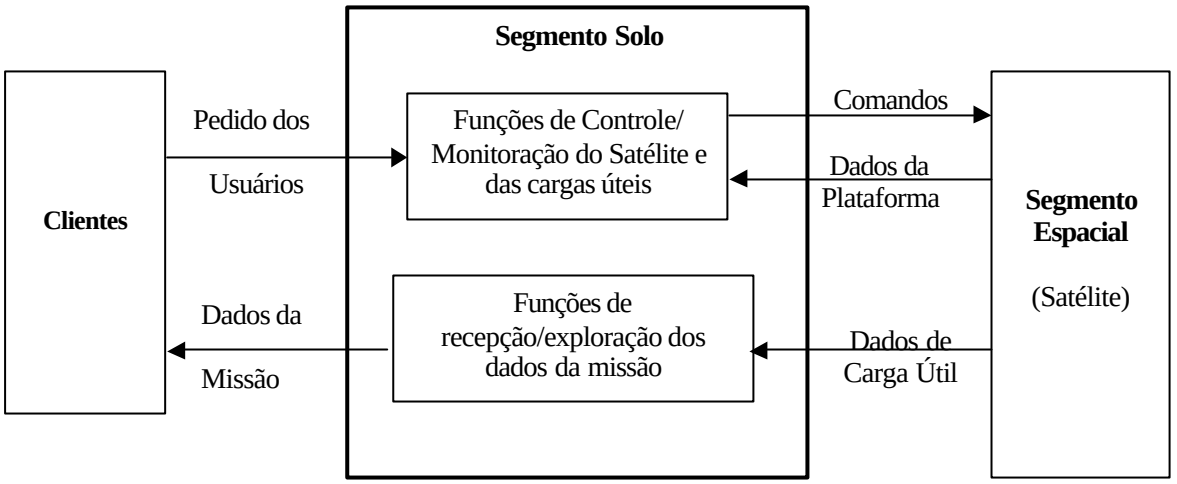

FIGURA 2.1 - Relacionamento entre o Segmento Espacial, o Segmento Solo e o Cliente.

FONTE: adaptada de Larson e Wertz (1996, pág. 581).

O segmento solo é composto de dois componentes principais: organização das operações de solo e os sistemas de solo. A organização das operações de solo compreende os recursos humanos que são responsáveis pela execução das várias tarefas de operações e pela preparação dos dados de operações da missão. Os sistemas de solo consistem-se dos elementos de solo (*hardware*, *software*, instalações) que suportam o controle e a monitoração do segmento espacial e a exploração dos dados da missão (ECSS-E-70 , 2000). Os principais elementos de solo são:

- **Sistema de Controle da Missão** (CM) - Este elemento é logicamente decomposto nos seguintes elementos funcionais: Sistema de Controle de Operações (SCO), Sistema de Controle de Carga Útil (SCCU) e Sistema de Exploração da Missão (SEM). Os dois primeiros são responsáveis por planejar, monitorar e controlar a plataforma e as cargas úteis respectivamente. O último é responsável por distribuir aos usuários os produtos da missão e todas as informações auxiliares requeridas para o planejamento e utilização dos dados da missão.

- **Sistema de Estação Terrena** (ET) – Este elemento constitui a interface direta com o segmento espacial. Cabe a ele estabelecer o elo de telecomunicação com

o satélite durante os períodos em que este passa sobre a estação. Durante estes períodos são transmitidos comandos ao segmento espacial (telecomandos) e recebidos dados da plataforma e das cargas úteis (telemetrias). O sistema ET pode ser classificado em duas instâncias lógicas:

- Estação Terrena de serviço que é utilizada para o controle do Segmento Espacial (ET-SE) provendo serviços de telemetria, telecomando e de rastreio tanto para a plataforma como para carga útil.

- Estação Terrena de aplicação que suporta a exploração da missão (ET-EM) provendo serviços de recepção dos dados de carga útil tais como o recebimento de sinais de telecomunicações, dados científicos e imagens da terra.

- **Rede de comunicação de dados** (NET) que é responsável pela conexão entre todos os sistemas de solo permitindo a comunicação entre eles e com os usuários da missão.

A Figura 2.2 mostra a estrutura do segmento solo e os seus principais elementos. A decomposição apresentada é de natureza lógica e, na prática, essas funções podem ser fisicamente agrupadas de acordo com o tipo da missão e da organização do segmento solo.

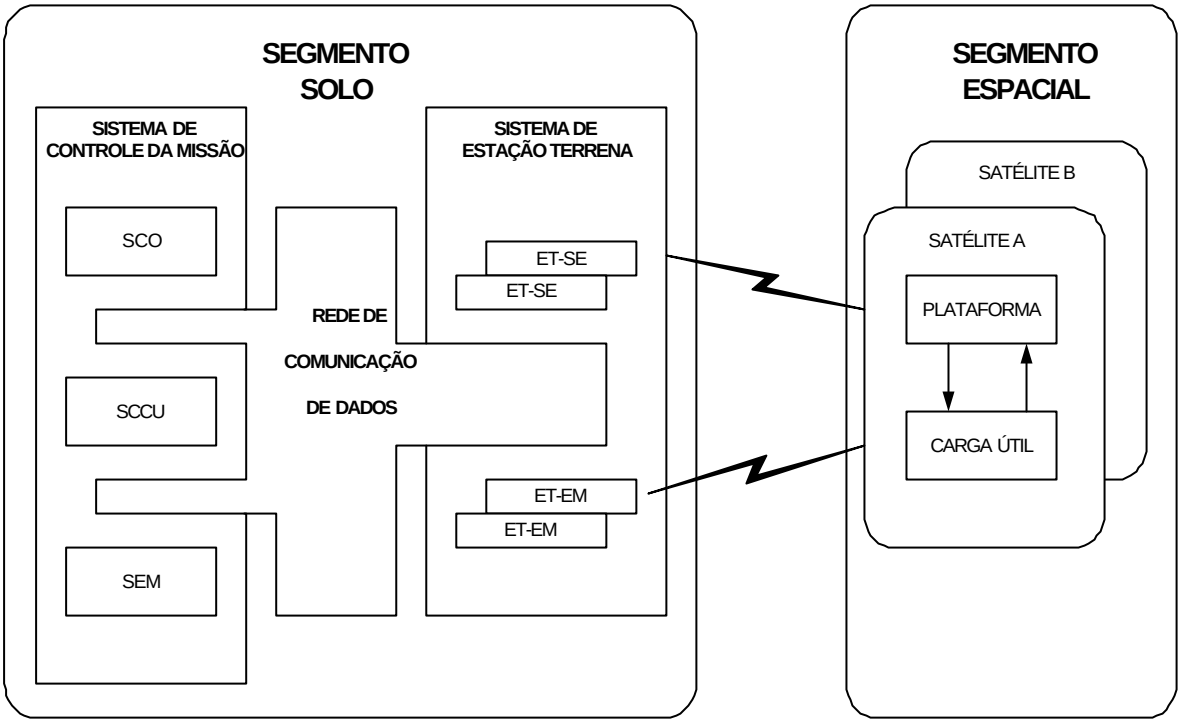

FIGURA 2.2 - Estrutura do Segmento Solo.

FONTE: adaptada da norma ECSS-E-70 (2000, pág. 21).

2.2 Conceitos de Operação de Missão

Para que uma missão espacial tenha sucesso, além da preparação física dos segmentos espacial e solo, é de fundamental importância o planejamento da operação da missão. Este planejamento deve definir de forma clara como a operação deve ser feita nas suas diferentes fases, para permitir que os objetivos da missão sejam alcançados da maneira mais eficiente possível.

A operação de uma missão espacial de acordo com o padrão ISO 14950:2004 (ISO, 2004) é uma atividade sob responsabilidade do segmento solo que tem como objetivo:

- Garantir a provisão da missão e dos serviços e produtos científicos;
- Executar as tarefas de operações de rotina;
- Recuperar as contingências de bordo;

- Gerenciar os recursos de bordo de forma a maximizar a vida útil da missão e as provisões de serviços e produtos.

A operação de uma missão pode ser classificada em dois grupos: operações de bordo e operações de solo. O primeiro grupo corresponde a todas as atividades relacionadas ao planejamento, execução e avaliação do controle do segmento espacial quando em órbita. Já as operações de solo correspondem a todas as atividades relacionadas ao planejamento, execução e avaliação do controle das facilidades de solo de suporte tais como as estações terrenas e rede de comunicação de dados de solo (ECSS-E-70 , 2000). Desta forma, normalmente as operações são executadas seguindo dois planos de operação principais: Plano de Operação de Vôo (POV) e o Plano de Operação de Solo (POS).

O POV é de extrema importância para o sucesso da missão, pois é por meio da sua execução que se mantém o segmento espacial funcionando convenientemente, de forma a atingir os objetivos da missão. Todas as atividades do POV têm como foco as passagens do satélite sobre estações terrenas, que são os períodos em que é estabelecido o contacto entre os segmentos solo e espacial. O intervalo de tempo em que um satélite permanece na região de visibilidade da antena de uma estação terrena em uma passagem restringe as operações de vôo a serem realizadas durante essa passagem. As atividades típicas de uma passagem, como mostrado na Figura 2.3, podem ser classificadas de acordo com três períodos distintos: pré-passagem, passagem e pós-passagem.

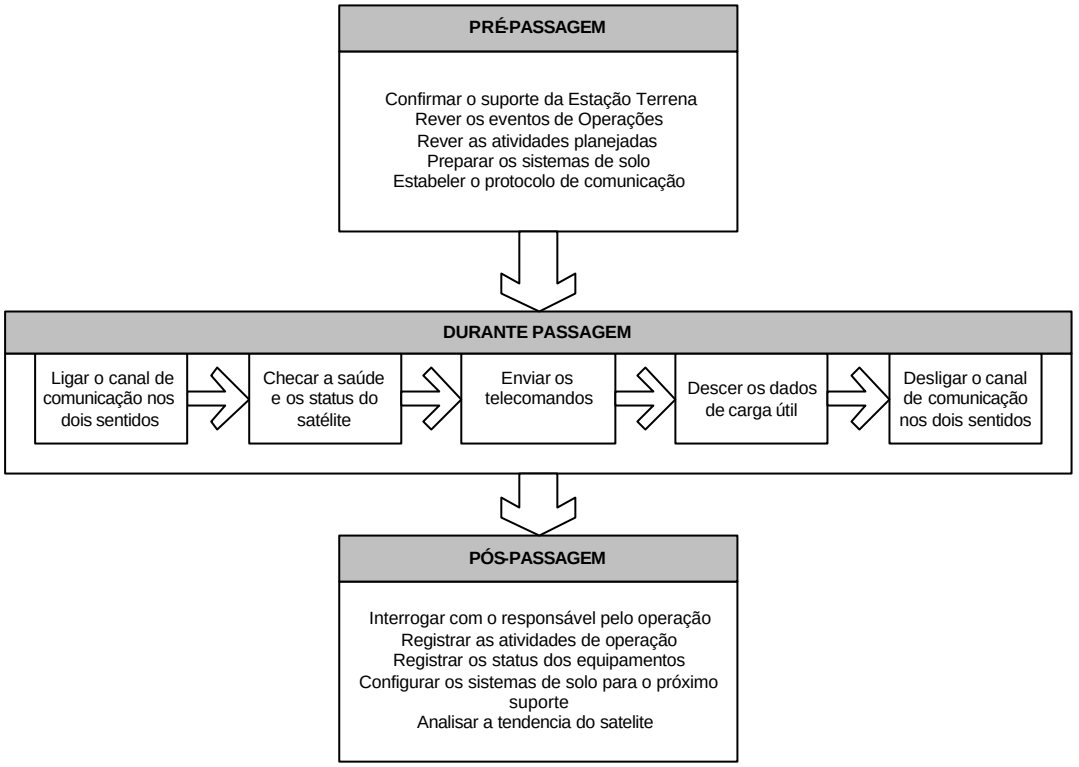

FIGURA 2.3 - Atividades Típicas de uma Passagem.

FONTE: adaptada Larson e Wertz (1996, pág. 559).

Cada passagem de um satélite é preparada configurando-se os sistemas de solo de forma a suportar as necessidades de rastreio, de transmissão de telecomandos e recepção de telemetria. Uma vez que o satélite torna-se visível para a estação terrena, o canal de comunicação é ligado. No final da passagem, o canal de comunicação é desligado.

Existem dois tipos básicos de POV normalmente confeccionados para uma missão espacial. O primeiro é desenvolvido para a fase crítica de Lançamento e Órbitas Preliminares (LEOP) em que a órbita do satélite deve ser determinada pela primeira vez e os subsistemas do satélite devem ter o funcionamento verificado em órbita. O segundo tipo de POV é desenvolvido para atender a fase de rotina, na qual a operação é constituída pela aplicação de procedimentos operacionais repetitivos, permitindo o uso de um procedimento automático para sua geração. Neste caso, o POV é gerado para

atender um determinado período que depende da missão e ao final de cada período um novo POV é gerado para atender o próximo.

Inicialmente a geração do POV é baseada nas informações providas pelo Manual do Usuário do Satélite preparado pelo segmento espacial, nos resultados das análises da missão e na base de dados da missão que contêm a definição de todos os parâmetros de telemetria e telecomando. O detalhamento do POV e o desenvolvimento de procedimentos operacionais padrões para atender tanto as situações nominais quanto a recuperação de algumas situações principais de contingência previstas, são feitos por especialistas da equipe de operação na fase de preparação da missão. Nesta fase os procedimentos operacionais escritos são validados normalmente com o emprego de simuladores em *software* de satélites. No POV se encontra o cronograma detalhado (por passagem do satélite sobre as estações terrenas envolvidas) das atividades a serem executadas onde são referenciados os procedimentos operacionais desenvolvidos, que devem se encontrar devidamente documentados e acessíveis aos operadores, tanto na fase de treinamento, quanto para consulta na fase de operação real.

Diferentes equipes de operação e suporte se fazem necessárias para a operação da missão e das facilidades de solo. As equipes são estruturadas da seguinte forma como encontrado na norma ECSS-E-70 (2000):

- Equipe de operação que é composta pelos seguintes grupos:
    - Controle de Vôo – responsável por todo o controle da missão e do segmento espacial. Este grupo pode ser dividido durante a fase de rotina da missão em equipes para operação da plataforma e para operação da carga útil;
    - Dinâmica de Vôo – responsável pela determinação de órbita, previsão de órbita e eventos orbitais, preparação de manobras de controle de atitude e órbita. Este grupo fornece suporte à equipe de Controle de Vôo;

- Operação de Solo – responsável por operar e manter as facilidades de suporte do solo tais como as estações terrenas, redes de comunicação de dados e centro de controle da missão.
- Exploração da Missão – responsável pela interface com o usuário da missão gerando os produtos, arquivando e distribuindo os dados de carga útil.

- Equipe de Suporte que é composta pelos grupos:
  - Suporte de Solo – composto pelos especialistas dos sistemas de solo que fornecem suporte para as equipes de operação de solo;
  - Suporte do Projeto – composto pelos especialistas do segmento espacial que fornecem suporte para a equipe de Controle de Vôo.

A operação da missão é dividida em duas fases operacionais principais. A fase considerada **crítica** que cobre todas as operações necessárias durante a fase de lançamento e aquisição do satélite, comissionamento em órbita e validação do segmento espacial. Após esta fase, se inicia a fase chamada de **rotina** ou **nominal** quando o segmento espacial está pronto para iniciar a exploração da missão. O procedimento de planejamento, objeto da presente pesquisa, é aplicável às atividades desta fase.

Durante a fase operacional de rotina, um processo interativo composto das tarefas de Preparação, Execução e Avaliação conduz a execução das operações que se fundamentam no POV e no POS. A tarefa de Preparação cobre todas as atividades relativas ao planejamento das operações futuras e das atividades de manutenção, e a produção dos cronogramas detalhados de operação baseados nos pedidos dos usuários e nos resultados da avaliação de desempenho dos segmentos espacial e solo. A tarefa de Execução cobre a implementação das operações contidas nos planos e a execução da monitoração do estado dos elementos dos segmentos espacial e solo, incluindo a execução de ações corretivas necessárias no caso de anomalias. E a tarefa de Avaliação

cobre a avaliação dos segmentos espacial e solo incluindo as análises de tendências que suportam a preparação de operações futuras e atividades de manutenção preventiva.

2.3 Automatização nas Operações de Missões Espaciais

Uma grande preocupação na comunidade espacial é com o alto custo para manter uma equipe de operação para as missões espaciais. Uma das tendências observadas para a busca de possibilidade de redução do custo é encontrar soluções de automatização com conseqüente diminuição das equipes de operação. Esta tendência pôde ser observada no último Congresso Internacional de Operações Espaciais (SPACEOPS/ 2004) como pode ser verificado por meio das seguintes citações:

- "Mais e mais gerentes de operações estão voltando suas atenções para a possibilidade de automatizar as operações de missões espaciais dentro do segmento solo, com a expectativa de diminuir custos através da redução das equipes de controles de satélites" (Monham e Rudolph, 2004).

- "O aumento do nível de automação tem sido identificado como um dos principais meios para reduzir os custos de operações" (Ferri, Pecchioli e Pena, 2004).

- "Durante os últimos anos, o objetivo do Centro de Operações Espacial Européia (ESOC) tem sido explorar caminhos possíveis para aumentar a automação das atividades de operações de satélites" (Ercolani et al, 2004).

Como escrito por Ferri (2004), a dificuldade está em definir o que deve ser automatizado para realmente reduzir os custos. Segundo ele, aplicar automação em qualquer processo que hoje em dia requer um esforço manual é uma idéia muito atrativa e em primeira vista um caminho fácil para reduzir custos de operação mesmo dentro do complexo mundo das operações espaciais. Entretanto, o custo da implementação, teste, depuração e validação das operações automatizadas são significantes e aumentam rapidamente com a complexidade das funções requeridas. As principais lições aprendidas por ele neste caminho da automatização de operações espaciais foram:

- A automatização não pode ser implementada como um todo, mas deve ser iniciada com uma simples função e crescer de acordo com as experiências adquiridas nas operações;

- O custo de desenvolvimento de sistemas automatizados, particularmente nas fases de teste e validação, cresce exponencialmente com a complexidade da função sendo automatizada. Mas por outro lado, uma grande economia já pode ser alcançada com a automatização de funções simples, porém repetitivas que normalmente são as que mais gastam esforço humano e mais sujeitas a erros.

Muito esforço tem sido feito no sentido de se encontrar soluções automatizadas para a operação de missão. Alguns estão investindo na autonomia do segmento espacial desenvolvendo sistemas de controle de bordo inteligentes com intuito de diminuir as tarefas de solo, enquanto que outros estão investindo na automatização do segmento solo ou em ambos. A seguir, alguns sistemas automatizados desenvolvidos com sucesso na área espacial são citados com uma breve descrição:

- APSS - *Activity Planning and Sequencing Subsystem* (Ko et al., 2004) – este subsistema faz parte do sistema de seqüência e planejamento da missão MER (*Mars Exploration Rover*). Foi desenvolvido baseado na tecnologia de planejamento da IA e é responsável pelo planejamento da estratégia da missão, da observação científica, das atividades de engenharia e pelo envio de seqüências de comando para a espaçonave.

- ASPEN -*Automated Scheduling and Planning Environment* (Chien et al., 2002) – é um sistema adaptável, reconfigurável e modular baseado em técnicas de IA que é capaz de suportar aplicações de planejamento e programação, incluindo planejamento das operações do satélite, planejamento para o projeto da missão, planejamento das atividades de veículos de superfície operados remotamente ("*Rover*"), e planejamento de utilização das antenas de solo. É utilizado tanto em aplicações de bordo como de solo. Algumas missões espaciais que estão utilizando este sistema são: *Citizen Explorer* (CX), U. S. *Navy UHF Follow On 1* (UFO1) e *Closed Loop Execution and Recovery* (CLEaR).

- ASE - *Autonomous Science Experiment* (Sherwood et al., 2004) – este experimento iniciou o seu teste operacional a bordo da missão EO1 (*Earth Orbiter* 1). O software do ASE usa execução a bordo baseada em objetivos, tarefas robustas e planejamento contínuo. Também utiliza máquina de aprendizado e reconhecimento de padrões a bordo para aumentar radicalmente o retorno de dados científicos pela capacidade de selecionar descidas de dados de forma inteligente e reprogramação dos objetivos de forma autônoma. Desta forma, as funções de análise e planejamento da missão em solo são simplificadas reduzindo assim custos operacionais.

- INTELMOD - *INTELligent MODeller* (Donati e Romani, 2000) – é um sistema inteligente baseado no conhecimento para dar suporte à equipe de operação de vôo nas suas tarefas de execução da missão. Consiste em um sistema de suporte à decisão para gerenciamento de falhas. Este sistema foi customizado para a missão CLUSTER-II.

- MAPGEN - *Mixed-initiative Activity Plan GENerator* (Ai-Ghang et al., 2004) – é um sistema de suporte à decisão baseado em solo e que foi desenvolvido para a missão MER. Ele fornece capacidade de planejamento e geração automática de cronogramas à equipe de operação os quais agilizam significativamente o processo de planejamento diário e o aumento do retorno de dados científicos da superfície de Marte. Ele faz uso do estado da arte nas técnicas de IA para auxiliar o usuário na geração de planos seguros, complexos e confiáveis. A sua utilização apontou um importante caminho no desenvolvimento de aplicações mais inteligentes capazes de gerar ferramentas de suporte de solo para as missões da NASA. Combina técnicas de computação com o conhecimento humano de forma a obter grandes benefícios nas operações de missão.

Dos sistemas citados acima pode ser notado que técnicas de planejamento de IA estão sendo usadas para poder dar soluções de automatização nas operações de missões espaciais. O próximo Capítulo apresenta uma visão geral da tecnologia de agentes e

uma descrição mais detalhada sobre agentes de planejamento, técnicas de planejamento e linguagens de representação de problemas de planejamento.

CAPÍTULO 3

TECNOLOGIA DE AGENTES DE PLANEJAMENTO

Neste Capítulo partiu-se do conceito mais explorado por Russel e Norvig (2004) de que um agente de planejamento é um agente inteligente e, sendo assim, primeiro é dada uma visão geral sobre agente (item 3.1). Em seguida, o conceito de agente de planejamento é descrito (item 3.2) destacando a técnica de planejamento em grafo e a linguagem de planejamento PDDL. Como para a implementação de um agente de planejamento, a criação de uma base de conhecimento se faz necessária, o item 3.3 apresenta um processo para geração desta base.

3.1 Definição de Agente

O que é um agente? O termo agente não possui uma definição unanimemente aceita. Várias definições vinculadas a pesquisadores empenhados no estudo e desenvolvimento de agentes são encontradas. A seguir, serão relacionadas algumas das definições encontradas para agentes.

Maes (1995) considera que agentes são sistemas computacionais que habitam algum ambiente dinâmico e complexo, percebem e agem de forma autônoma nesse ambiente, e fazendo isso, realizam um conjunto de metas ou tarefas para as quais foram projetados.

Wooldridge e Jennings (1995) sugerem considerar dois enfoques na conceituação de agentes: noção fraca e forte. Para a noção fraca, o termo agente é empregado em sua forma mais geral, ou seja, denotando uma entidade baseada em hardware e em software. A noção forte de agentes considera especialmente a área de IA onde o termo agente possui um significado mais específico e forte. O agente é entendido como sendo uma entidade que, além das propriedades apresentadas anteriormente, é implementado empregando conceitos mais usualmente aplicados a seres humanos e caracterizado por estados mentais, tais como, crença, intenção e compromisso.

Para Franklin e Graesser (1996), um agente é um sistema situado dentro de um ambiente e como parte deste ambiente, sente e age sobre o mesmo, ao longo do tempo, em busca de cumprir sua própria agenda e de maneira que afete o seu próprio sentido no futuro.

Para Feber (1999), um agente é uma entidade física ou virtual que tem as seguintes características:

- É capaz de agir em um ambiente;
- Pode comunicar diretamente com outros agentes;
- É dirigido por um conjunto de tendências;
- Possui recursos próprios;
- É capaz de perceber o ambiente, mas com uma extensão limitada;
- Tem somente uma representação parcial do seu ambiente;
- Possui habilidades e pode oferecer serviços;
- Pode ser capaz de se reproduzir e
- Seu comportamento tende a satisfazer seus objetivos, levando em conta os recursos e habilidades disponíveis para ele e depende de sua percepção, suas representações e da comunicação que ele recebe.

Por entidade física, entende-se alguma coisa que age no mundo real como um robô, uma espaçonave ou um carro. Entidade virtual é um componente de software e um modelo computacional que não tem existência física.

Para Russell e Norvig (2004), um agente é tudo que pode ser considerado capaz de perceber seu ambiente por meio de sensores e de agir sobre esse ambiente por intermédio de atuadores. Um agente robótico, por exemplo, poderia ter câmeras e

detectores da faixa de infravermelho funcionando como sensores e vários motores como atuadores. Um outro exemplo pode ser um agente de software que recebe seqüências de teclas digitadas, conteúdo de arquivos e pacotes de rede como entradas sensoriais e atua sobre o ambiente exibindo algo na tela, gravando arquivos e enviando pacotes de rede.

Para se projetar um agente inteligente é importante definir a natureza do ambiente de forma a poder determinar o projeto apropriado e qual a técnica de implementação de agentes a ser aplicada. Para isso, se faz necessário definir o **ambiente de tarefa** do agente a ser desenvolvido e classificá-lo.

A Figura 3.1 ilustra uma visão simplificada de um agente e será usada como base para entendimento do ambiente de tarefa.

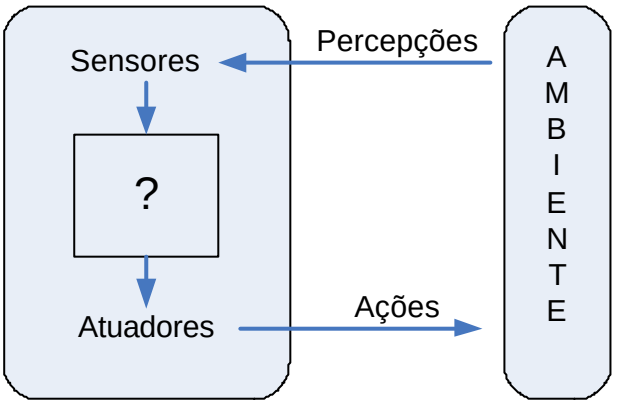

FIGURA 3.1 - Agentes interagem com ambientes por meio de sensores e atuadores.

FONTE: Russell e Norvig (2004, pág. 34).

A primeira etapa do projeto de agente deve consistir sempre em especificar o ambiente de tarefa que é essencialmente formado pelos "problemas" para os quais os agentes buscam as "soluções". O ambiente de tarefa é composto da especificação da medida de desempenho, do ambiente externo, dos atuadores e dos sensores do agente.

A medida de desempenho avalia o comportamento do agente em um ambiente. Quando um agente é inserido em um ambiente, ele gera uma seqüência de ações, de acordo com as percepções que recebe por meio dos sensores. Essa seqüência de ações realizadas pelos atuadores faz o ambiente passar por uma seqüência de estados. Se a seqüência é a

esperada, isso quer dizer que o agente funcionou bem. A medida de desempenho deve ser especificada de acordo com o resultado realmente desejado no ambiente.

Para exemplificar, o ambiente de tarefa de um agente para um sistema de análise de imagens de satélite seria: (i) definição correta da classe de imagem (medida de desempenho); (ii) *link* de transmissão do satélite com o solo (ambiente); (iii) exibir a categorização da imagem (atuadores); (iv) *arrays* de *pixels* multiespectrais (sensores).

Os ambientes de tarefas variam em função de diversas características significativas. A combinação dessas características em um ambiente de tarefa é o que determina o grau de dificuldade na implementação de um agente. Um ambiente de tarefa pode ser classificado de acordo com as características identificadas por Russell e Norvig (2004) que estão relacionadas a seguir:

- **Completamente observável versus parcialmente observável** – um ambiente de tarefa é completamente observável se os sensores do agente permitem acesso ao estado completo do ambiente em cada instante. Um ambiente é parcialmente observável devido a ruído e a sensores imprecisos ou porque partes do seu estado estão simplesmente ausentes nos dados do sensor.

- **Determinístico versus estocástico** – um ambiente de tarefa é considerado determinístico se seu próximo estado é completamente determinado pelo estado atual e pela ação executada pelo agente. Caso contrário, ele é estocástico. Se o ambiente é determinístico exceto pelas ações de outros agentes, dizemos que o ambiente é **estratégico**.

- **Episódico versus seqüencial** - em um ambiente de tarefa episódico, a escolha da ação em cada episódio só depende do próprio episódio e o agente não precisa pensar para frente. Cada episódio consiste na percepção do agente, e depois na execução de uma única ação. Em um ambiente seqüencial, a decisão atual pode afetar todas as decisões futuras.

- **Estático versus dinâmico** – o ambiente de tarefa é dinâmico se o ambiente puder se alterar enquanto o agente está deliberando, caso contrário, ele é estático. Em ambiente estático, o agente não precisa continuar a observar o mundo enquanto está decidindo sobre a realização de uma ação e não precisa se preocupar com a passagem do tempo. Já os ambientes dinâmicos estão sempre perguntando ao agente o que ele deseja fazer. Se o próprio ambiente não mudar com a passagem do tempo, mas o nível de desempenho do agente se alterar, o ambiente é classificado como **semi-dinâmico**.

- **Discreto versus contínuo** – a distinção entre discreto e contínuo pode se aplicar ao estado do ambiente, ao modo como o tempo é tratado e às percepções e ações do agente. Um jogo de xadrez é considerado como um ambiente de tarefas discreto. Já um jogo de futebol é considerado como um ambiente contínuo.

- **Agente único versus multi-agente** – a distinção entre esses dois tipos está relacionada à quantidade de agentes em um ambiente. Um agente que resolve um problema sozinho está claramente em um ambiente de agente único. Em um ambiente multi-agente existem vários agentes que se relacionam entre si para atingir um objetivo comum.

Um ambiente de tarefa que seja parcialmente observável, estocástico, seqüencial, dinâmico, contínuo e multi-agente é o caso mais difícil de se implementar. A direção de um táxi é um exemplo de um ambiente de tarefa com estas características. Já o ambiente de tarefa para análise de imagens de satélite é completamente observável, determinístico, episódico, semi-dinâmico, contínuo e composto de um único agente.

3.2 Definição de Agente de Planejamento

Como existem vários tipos de agentes, este item tem o objetivo de definir o agente de planejamento, por ser este tipo de interesse para o trabalho.

O agente de planejamento incorpora o tipo básico de programa de agente baseado em objetivos. A Figura 3.2 apresenta a estrutura do agente baseado em objetivos. Este tipo

de agente necessita da descrição do estado atual do ambiente e informações sobre objetivos que descrevem situações desejáveis. O programa de agente combina essas informações (estado atual e objetivo) com informações sobre os resultados das ações possíveis, a fim de escolher a seqüência de ações que permitam que o objetivo seja alcançado.

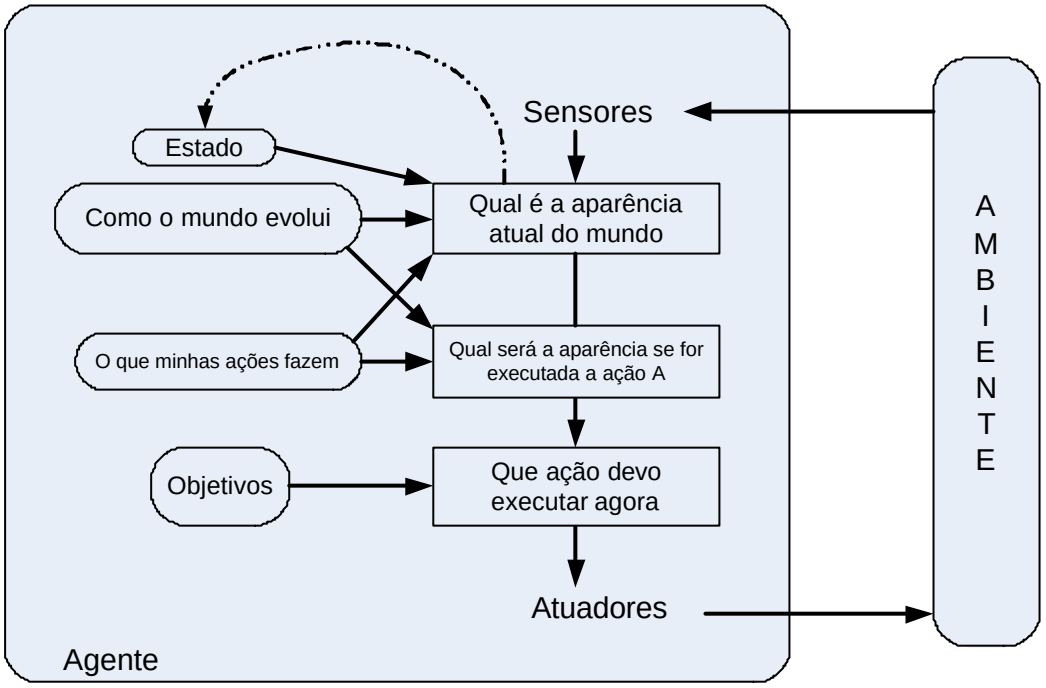

FIGURA 3.2 - Agente baseado em objetivos.

FONTE: Russell e Norvig (2004, pág. 50).

Às vezes, a seleção da ação baseada em objetivos é direta, quando a satisfação do objetivo é alcançado de imediato com a execução de uma única ação. Outras vezes ela será mais complicada, quando o agente tiver de considerar longas sequências de ações até encontrar um meio de atingir o objetivo. São nestas situações mais complexas que se encaixa a área de planejamento da IA.

O termo planejamento é definido no dicionário Aurélio como “plano de trabalho detalhado”. No contexto de agente de planejamento o termo pode ser definido como uma seqüência de ações que ao serem executadas alcançarão um objetivo.

O agente de planejamento, como o próprio nome indica, é dedicado a resolução de problemas de planejamento. A Figura 3.3 apresenta um diagrama de um agente de planejamento clássico. Segundo Feber (1999), o agente inicia com a especificação inicial das situações do ambiente (estado inicial), dos objetivos a serem atingidos com o plano e do universo de ações disponíveis. O módulo Planejador encontra um conjunto de planos {pi} do qual apenas um é selecionado pelo módulo Selecionador. Este plano é então encaminhado para o módulo Executor que se encarregará de sua execução. Em muitos sistemas, o Planejador gera um único plano e o módulo Selecionador não é necessário.

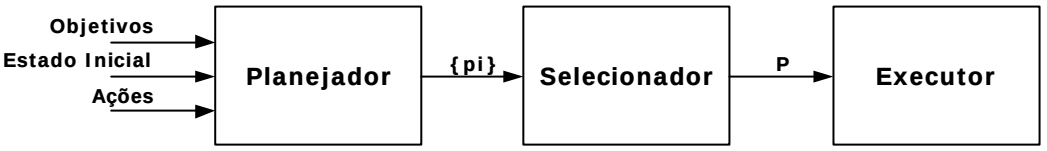

FIGURA 3.3 - Agente de Planejamento Clássico.

FONTE: Feber (1999, pág. 152).

Segundo Russell e Norvig (2004), um planejador pode ser visualizado como um programa que procura uma solução ou como um programa que demonstra (construtivamente) a existência de uma solução. Os planejadores são algoritmos poderosos de resolução de problemas que operam sobre representações explícitas proposicionais (ou de primeira ordem) de estados e ações.

Eles definem dois tipos de ambiente de planejamento: clássico e não-clássico dependendo do ambiente de tarefas descrito no item 3.1. O ambiente de um planejamento clássico é classificado como um ambiente completamente observável, determinístico, finito, estático (a mudança só acontece quando o agente age) e discreto. Em um ambiente de planejamento não-clássico, o ambiente é parcialmente observável e também pode ser estocástico. Para cada tipo de problema de planejamento existe um conjunto diferente de algoritmos e projetos de agentes.

O problema de planejamento clássico é o tipo de problema para o qual são fornecidos o estado inicial do ambiente, o conjunto de ações e um objetivo a ser atingido. A solução

é uma seqüência de ações que quando executada começa em um estado inicial e leva para uma situação na qual o objetivo é atingido. Ele assume que o planejador conhece tudo o que é verdade na situação inicial e também conhece os efeitos de cada ação.

No problema de planejamento temporal, outros fatores são levados em conta além dos elementos do planejamento clássico como: o tempo gasto para a execução do plano, o início e a duração de cada ação e em problemas mais complexos também a restrição de recursos. Um único recurso pode ser utilizado por duas ações diferentes e esta restrição deve ser levada em consideração durante a geração do plano pelo planejador. Neste tipo de problema, aspectos mais complexos sobre a geração de planos são introduzidos como ações concorrentes e/ou mutuamente exclusivas. O planejamento temporal é aplicado a problemas do mundo real.

Para os tipos de problemas de planejamento clássico e temporal, um agente pode primeiro planejar e depois executar o plano "com os olhos fechados".

Já para um ambiente não-clássico, um agente deve usar suas percepções para descobrir o que está acontecendo no ambiente enquanto o plano está sendo executado e, possivelmente, modificar ou substituir o plano se acontecer algo inesperado. Para tratar este tipo de problema existem várias abordagens como planejamento com conformação ou sem sensores, planejamento condicional e planejamento contínuo, e o problema pode ser resolvido com qualquer uma dentre diversas representações para regras do tipo condição-ação. O tratamento deste tipo de problema muitas vezes depende dos sensores e dos atuadores disponíveis para o agente.

Qualquer representação de problemas de planejamento inclui a representação de estados, ações e objetivos para tornar possível a criação de algoritmos de planejamento tirando proveito da estrutura lógica do problema. A chave é usar uma linguagem que seja suficientemente expressiva para descrever uma ampla variedade de problemas, mas restritiva o bastante para permitir que algoritmos eficientes operem sobre ela.

A técnica de planejamento usada e a linguagem de representação de problemas de planejamento formam a base da criação de agentes de planejamento. Os subitens 3.2.1 e

3.2.2 descrevem estes dois temas. As técnicas de planejamento para um ambiente não-clássico não será abordada por não fazer parte do escopo deste trabalho que tem como objetivo solucionar um problema de planejamento temporal.

3.2.1 Técnicas de Planejamento

A área de planejamento da IA tem buscado construir algoritmos planejadores que capacitem um agente a sintetizar uma seqüência de ações que atinja um objetivo. Várias técnicas de planejamento têm sido desenvolvidas nos últimos anos (Penberthy e Weld, 1992; Kautz, McAllester e Selman, 1996; Bonet e Geffner, 1998) e a área de planejamento sofreu uma grande revolução com o desenvolvimento da técnica de planejamento em grafo (Blum e Fusrt, 1995).

No início dos anos setenta, Fikes e Nilsson (1971) lançaram o famoso planejador STRIPS (*Stanford Research Institute Problem Solver*). De formulação simples, o STRIPS marca o início do estudo de planejamento automático. Os operadores STRIPS permaneceram quase duas décadas como paradigma vigente, mas, ainda assim, havia problemas difíceis para serem resolvidos por eles. A alta complexidade de seu algoritmo (busca exaustiva no espaço de estados) fez com que muitas melhorias fossem propostas.

Os anos oitenta foram marcados por certo desânimo quanto à capacidade da geração automática de planos, e alguns poucos resultados só apareceram no final da década. O planejador mais famoso do final da década de oitenta foi o Prodigy (Knoblock et al., 1989), que adotava uma engenhosa heurística de planejamento incluindo reordenação de operadores, aprendizagem automática e busca em paralelo de sub-objetivos. Mas apesar dos esforços, os pesquisadores não conseguiram um mecanismo robusto o bastante para resolver problemas genéricos.

A década de noventa começou com a busca do aumento da velocidade da geração de planos e muitos estudos sobre linguagem e técnicas alternativas ao STRIPS. Nessa época, surgiram alguns conceitos válidos ainda hoje, como planejamento em ordem

parcial, uso de variáveis na definição do esquema de ações, efeitos condicionais e linguagens baseadas em causalidade entre os operadores (Penberthy e Weld, 1992).

A partir de 1995, a evolução do planejamento automático sofreu novo impulso quando Blum e Furst (1995) lançaram o algoritmo de planejamento em grafo - GRAPHPLAN, que consiste em um mecanismo de representação de intervalos de tempo e extração de planos por meio de grafos de planejamento. A simplicidade do novo planejador, aliada ao desempenho muito superior aos planejadores da época, estimulou o desenvolvimento de muitos outros planejadores a partir de sua idéia central.

Um ano mais tarde, Kautz, McAllester e Selman (1996) lançaram o SATPLAN que recuperou as idéias associadas a planejamento por satisfatibilidade (Kautz e Selman, 1992). O SATPLAN é um planejador que traduz o conhecimento do GRAPHPLAN ao problema da satisfatibilidade. Esta abordagem se baseia em testar a satisfatibilidade de uma sentença lógica, ou seja, a sentença contém símbolos de proposições correspondentes a toda ocorrência de ação possível. Um modelo que satisfizer a sentença atribuirá verdadeiro às ações que fizerem parte de um plano correto e falso às outras. Uma atribuição que corresponder a um plano incorreto não será um modelo, porque será inconsistente com a asserção de que o objetivo é verdadeiro. Se o problema de planejamento é insolúvel, então a sentença será não-satisfatível. Tal transformação na forma de representar os problemas se mostrou mais eficiente que o GRAPHPLAN na maioria dos casos em que os dois sistemas foram comparados.

Em 1998, Bonet e Geffner (1998) propuseram o planejamento por busca heurística (HSP). Esta técnica transforma os problemas de planejamento em problemas de busca heurística pela extração automática da função heurística ao invés de introduzí-la manualmente. A grande contribuição desta técnica foi a definição de uma função de custo que permite verificar a distância dos estados intermediários ao estado objetivo. Este técnica de planejamento permitiu uma maior rapidez na solução dos problemas.

Recentemente, a área de pesquisa de planejamento teve o foco voltado para o problema de planejamento temporal que permite que planejadores manipulem problemas do mundo real. O planejamento temporal tem como requisito o tratamento de restrições de

tempo e recursos. O plano temporal é composto de uma lista tripla (hora de início da execução da ação, ação e duração), ou seja, cada ação tem uma restrição temporal. No ano de 2003, este tipo de problema de planejamento foi inserido na Competição Internacional de Planejamento (IPC, 2005).

Hoje, o grande desafio é tratar os eventos exógenos incondicionais determinísticos, que são tipicamente janelas de tempo muito comuns no mundo real como, por exemplo, período de tempo em que uma loja fica aberta, período de tempo de trabalho de uma pessoa, período em que é dia e período de visibilidade de um satélite. Estes eventos são fatos que se tornarão verdadeiros ou falsos em algum ponto do tempo, independentemente das ações que o planejador escolher para executar. Os pesquisadores estão buscando soluções usando não apenas uma técnica, mas várias para encontrar um caminho que resolva este problema. No IPC do ano de 2004 (IPC, 2005), os problemas de janelas de tempo também foram inseridos na competição. Apenas os planejadores LPG-TD - *Local Searching for Planning Graphs – Timed initial literal and Derived predicates* (Gerevini, Saetti e Serina, 2005), SGPlan – *Subgoal partitioning and Global search Planner* (Wah e Chen, 2004) e TilSAPA - *Timed Initial Literals in Temporal Planning* (Kavuluri e Khemani,2004) estavam preparados para solucionar este tipo de problema.

O planejamento em grafo será apresentado com mais detalhes no próximo item porque o planejador LPG-TD, que foi utilizado no protótipo construído por este trabalho de pesquisa, tem como base esta técnica.

3.2.1.1 Técnica - Planejamento em Grafo

Blum e Furst (1995) propuseram o planejamento automático baseado na análise de grafos e desenvolveram o algoritmo GRAPHPLAN. Este algoritmo revolucionou a área de planejamento da IA por duas razões:

- Era um algoritmo simples e robusto, que apresentava uma velocidade de planejamento bem superior que seus antecessores Prodigy (Knoblock et al., 1989) e UCPOP (Penberthy e Weld, 1992);

- Permitiu a automação da redução de problemas de planejamento ao problema de satisfatibilidade proposicional (Kautz, McAllester e Selman, 1996).

Uma ilustração de um grafo de planejamento está representada na Figura 3.4.

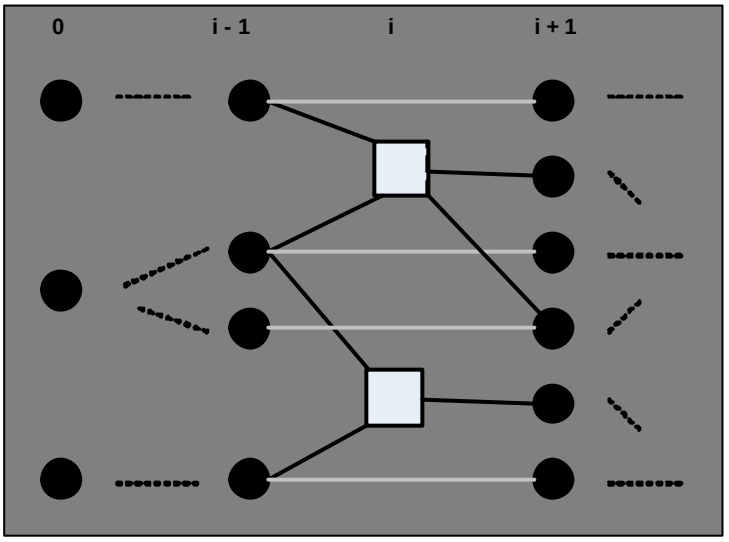

FIGURA 3.4 - Representação de um grafo de planejamento.

FONTE: Weld (1999)

Este grafo alterna entre camadas de **nodos proposicionais** (representados pelos círculos) e **nodos de ação** (representado pelos quadros). Os níveis de numeração par contêm os nodos proposicionais que representam o conhecimento do mundo em um determinado instante de tempo, dependendo das ações executadas nos instantes de tempo precedentes. O nível zero consiste precisamente dos literais que são verdadeiros no estado inicial do problema de planejamento. Os nodos dos níveis de numeração impar correspondem às instâncias das ações que são todas aquelas ações que podem ter suas precondições satisfeitas nesse instante de tempo, dependendo de quais literais são realmente válidos. Existe um nodo para cada instância de ação cujas precondições estão presentes (e são mutuamente consistente) no nível anterior. As linhas que conectam os literais com as instâncias de ações do próximo nível significam que as precondições das

ações referenciam tais literais. Adicionalmente, as linhas que conectam nodos de ações aos literais subseqüentes significam que os mesmos se tornam verdadeiros pelos efeitos das ações. As linhas cinzas e horizontais entre os nodos proposicionais representam "ações de persistência", que indicam que um nodo proposicional permanece verdadeiro ao passar de uma situação para a seguinte, se nenhuma ação o alterar.

Note que o grafo de planejamento representa ações paralelas a cada nível de ação. Isto significa que um grafo de planejamento com k níveis de ações pode representar um plano com mais que k ações. Entretanto, o fato de um conjunto de ações estar incluído em um mesmo nível de um grafo de planejamento não implica que todas as ações do conjunto serão necessariamente executadas.

A parte central da eficiência do GRAPHPLAN é relacionada à inferência de relação de exclusão mútua binária ("*mutex*") entre nodos de um mesmo nível, ou seja, ele registra conflitos entre ações, que impediriam que essas ações ocorressem juntas. Uma relação de exclusão mútua (ou seja, de uma relação *mutex*) é válida entre **duas ações** em um determinado nível, se pelo menos uma das três condições descrita a seguir são válidas:

- **Efeitos inconsistentes**: O efeito de uma ação é a negação do efeito de outra ação;

- **Interferência**: Um dos efeitos de uma ação é a negação de uma precondição da outra.

- **Necessidades concorrentes**: Uma das precondições de uma ação é mutuamente exclusiva com uma precondição da outra, ou seja, duas ações têm precondições que são mutuamente exclusivas no nível i-1.

Uma relação de exclusão mútua é valida entre **dois literais** no nível i se um literal é negação do outro, ou se suas ações geradoras forem mutuamente exclusivas. Essa condição é chamada **suporte inconsistente**

A Figura 3.5 ilustra as relações de exclusão mútua. Os círculos denotam literais, os quadrados representam ações e as linhas curvas e finas representam a relação *mutex*. As

três primeiras partes ilustram a relação *mutex* entre ações (quadrados pretos), e a última parte a relação *mutex* entre literais (círculo preto).

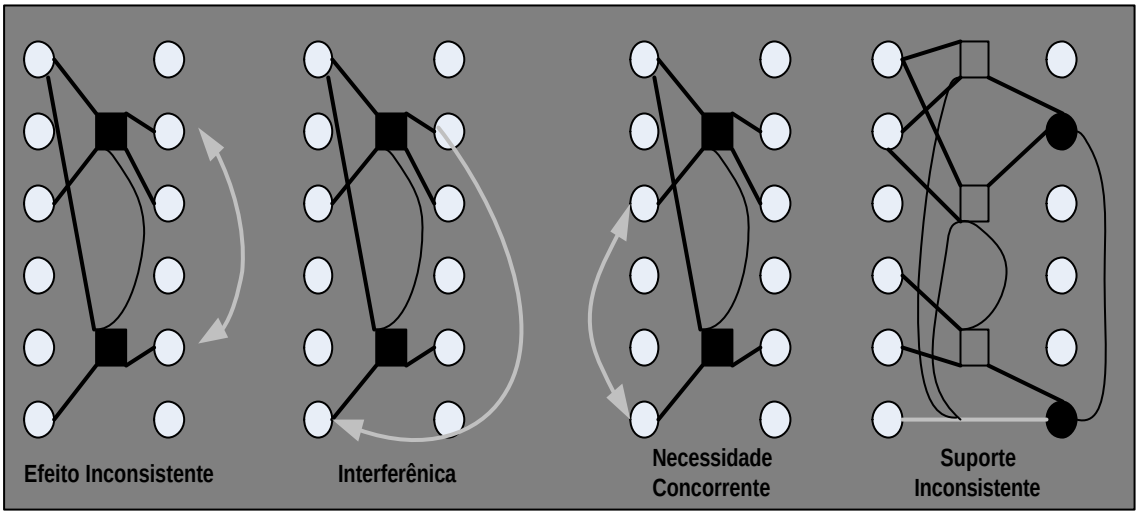

FIGURA 3.5 – Representação gráfica da definição ***mutex.***

FONTE: Weld (1999)

O algoritmo de planejamento em grafo tem duas etapas principais, que se alternam dentro de um laço repetitivo: **expansão do grafo** e **extração da solução**. Primeiro, ele verifica se todos os literais de objetivo estão presentes no nível atual, sem vínculos de exclusão mútua entre qualquer par deles. Se for o caso, então pode existir uma solução dentro do grafo atual e, portanto, o algoritmo tenta extrair essa solução. Caso contrário, ele expande o grafo adicionando as ações correspondentes ao nível atual e os literais de estados referentes ao próximo nível. O processo continua até ser encontrada uma solução ou até se concluir que não existe nenhuma solução. É importante observar que o processo de construção do grafo do planejamento não requer a escolha entre as ações, o que exigiria uma busca combinatória. Em vez disso, ele registra apenas a impossibilidade de certas escolhas com a utilização de vínculos de exclusão mútua. A complexidade da construção do grafo de planejamento é um polinômio de baixa ordem em relação ao número de ações e literais.

A Figura 3.6 apresenta o pseudocódigo do algoritmo de planejamento em grafo. A função PLANEJAMENTO-EM-GRAFO se alterna entre uma etapa de extração de

solução (EXTRAIR_SOLUÇÃO) e uma etapa de expansão do grafo (EXPANDIR_GRAFO). EXTRAIR_SOLUÇÃO verifica se um plano pode ser encontrado, começando do final e buscando no sentindo inverso. EXPANDIR_GRAFO adiciona as ações correspondentes ao nível atual e os literais relativos ao próximo nível.

```
função PLANEJAMENTO_EM_GRAFO(problema) retorna solução ou falha
grafo <- GRAFO_DE_PLANEJAMENTO_INICIAL(problema)
objetivos <- OBJETIVOS [problema]
repita
  se objetivos são todos de não –exclusão mútua no último nível de grafo então faça
     solução <- EXTRAIR_SOLUÇÃO(grafo, objetivos, COMPRIMENTO(grafo))
     se solução ? falha então retornar solução
     senão se NENHUMA_SOLUÇÃO_POSSÍVEL(grafo) então retornar falha
  grafo <- EXPANDIR_GRAFO(grafo, problema)
```

FIGURA 3.6 - Pseudocódigo do algoritmo de Planejamento em Grafo.

FONTE: Russell e Norvig (2004, pág.388).

Quando o algoritmo gera um nível com todos os literais de um objetivo e esses literais não são *mutex*, então é possível que o grafo contenha um plano válido para o problema em questão. Esta condição é necessária, mas insuficiente para afirmar que o plano existe. Para tal, é necessário que não haja conflito entre as ações que geraram as proposições do objetivo. A verificação da ausência de conflitos é a etapa de extração de solução que é um processo de busca que parte do nível mais alto do grafo ao nível mais baixo, verificando a presença de conflitos. Caso algum conflito seja detectado, o algoritmo cancela a extração da solução e retorna ao processo de expansão do grafo até que um outro nível contenha os literais do objetivo e assim por diante. Para cada proposição do nível i, o algoritmo seleciona uma ação geradora do nível i-1 e verifica se essa ação não é mutuamente exclusiva com as outras ações geradoras selecionadas no mesmo nível i-1. Caso não haja conflito, o algoritmo repete o processo com as pré-condições dessas ações selecionadas e assim por diante, até chegar ao nível 0. Caso chegue ao nível 0, o plano existe. Nota-se que um literal do objetivo pode ser gerado por

mais de uma ação, obrigando o algoritmo a considerar todas as combinações de ações geradoras.

Para ilustrar a execução do algoritmo de planejamento em grafo será utilizado o exemplo de Weld (1999) que está descrito na Tabela 3.1. O problema é a preparação de um jantar-surpresa pelo esposo enquanto sua esposa está descansando.

TABELA 3.1 – Descrição do problema do jantar-surpresa.

Estado Inicial	*lixo ? mãoslimpas ? silêncio*
Estado Objetivo	¬*lixo ? jantar ? presente*
Ações	
cozinhar	**Pré-condição**: *mãoslimpas* **Efeito:** *jantar*
embrulhar	**Pré-condição**: *silêncio* **Efeito**: *presente*
carregar	**Pré-condição**: não possui pré-condição **Efeito**: ¬*lixo ?* ¬*mãoslimpas*
arrastar	**Pré-condição**: não possui pré-condição **Efeito**: ¬*lixo ?* ¬*silêncio*

O estado inicial do ambiente é o seguinte: existe lixo acumulado (*lixo*) e o esposo tem as mãos limpas (*mãoslimpas*) e a casa está silenciosa (*silêncio*). Assume-se a hipótese de mundo fechado que significa que quaisquer proposições não mencionadas no estado inicial são consideradas falsas.

O estado objetivo, ou seja, os objetivos a serem alcançados no plano são: lixo removido (¬*lixo) e jantar* pronto e *presente* embrulhado.

Existem quatro ações aplicáveis ao problema de planejamento do jantar-surpresa, sendo:

- Ação ***cozinhar*** que requer que as mãos estejam limpas (*mãoslimpas*) e, como conseqüência produz o *jantar*.

- Ação ***embrulhar*** exige *silêncio* para evitar acordar a esposa e estragar a surpresa e tem como efeito o *presente* embrulhado.

- Ação ***carregar*** o lixo para fora satisfaz o literal *lixo* removido (¬*lixo*), mas suja as mãos (¬*mãoslimpas*).

- Ação ***arrastar*** o lixo para fora evita sujar as mãos satisfazendo o literal *lixo* removido (¬*lixo*), mas o enferrujado carrinho do lixo faz barulho (¬*silêncio*).

A Figura 3.7 apresenta o grafo de expansão do problema do jantar-surpresa.

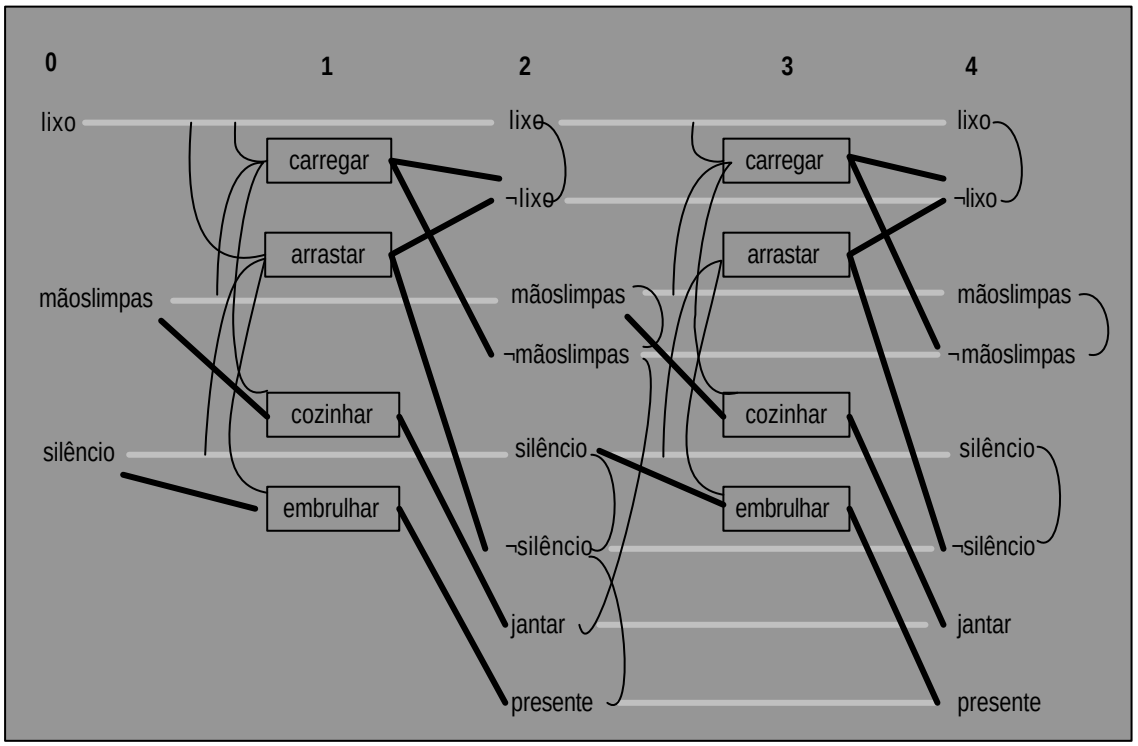

FIGURA 3.7 – Grafo de Planejamento para o problema do jantar-surpresa.

FONTE: adaptada Weld (1999)

O primeiro passo do algoritmo gera o nível 1, com todas as ações cujas pré-condições são satisfeitas no nível 0 (inicial), e o nível 2, com as conseqüências dessas ações. Observe-se que os arcos representam os conflitos entre as ações do nível 1 e o conflito

entre os literais no nível 2. A ação *carregar* é mutuamente exclusiva com a "manutenção" de *lixo* e de *mãoslimpas* porque gera a suas respectivas negações. A ação *carregar* gera mãos sujas (¬*mãoslimpas*), conflitando por interferência com a ação *cozinhar*. A ação *arrastar* é inconsistente com as "manutenções" de *lixo* e de *silêncio* por gerar as suas respectivas negações. A ação *arrastar* também causa interferência na ação *embrulhar* ao gerar a negação de sua pré-condição *silêncio*. No nível 2, as exclusões mútuas são entre os literais de sinal contrário e entre os literais causados por ações mutuamente exclusivas.

O fato de todos os literais do estado objetivo (¬*lixo ? jantar ? presente*) estarem presentes no nível 2 e o fato de não haver exclusão mútua entre eles possibilitam a existência de um plano. Nesse caso, o algoritmo passa para a segunda etapa que é a extração do plano no grafo atual (com os níveis 0, 1 e 2).

Observe-se que existem três literais do objetivo no nível 2, o literal ¬*lixo* foi gerado pela ação *carregar* e *arrastar*, o literal *jantar* foi gerado pela ação *cozinhar* e o literal *presente* foi gerado pela ação *embrulhar*. Neste momento, existem dois grupos de ações geradoras a serem considerados: {*carregar, cozinhar e embrulhar*} e {*arrastar, cozinhar e embrulhar*}. Note-se que nenhum dos grupos é consistente, pois a ação *carregar* é mutuamente exclusiva com a ação *cozinhar* e a ação *arrastar* é mutuamente exclusiva com a ação *embrulhar*. Como não há um conjunto de ações consistentes, a extração da solução falha. O algoritmo inicia uma nova expansão do grafo de planejamento gerando os níveis 3 e 4.

Embora nenhum novo literal tenha sido gerado, a diferença entre os níveis 2 e 4 está no número de exclusões mútuas entre os literais. Pode ser verificado que não há mais exclusão mútua entre o literal *jantar* e ¬*mãoslimpas*. Isso ocorre porque o literal *jantar* foi gerado por uma "ação de persistência" que não possui pré-condições conflitantes com as ações geradoras do literal ¬*mãoslimpas:* ação *carregar* e uma ação de persistência. A presença das ações de persistência reduz a ocorrência de conflitos entre as ações geradoras, uma vez que essas ações só são mutuamente exclusivas quando possuem pré-condições mutuamente exclusivas.

No nível 4, o algoritmo novamente encontra as condições que possibilitam a extração de um plano, porém com um número um pouco maior de conjuntos de ações geradoras. As ações geradoras para os objetivos são:

- Literal ¬*lixo* é gerado por {*carregar*, *arrastar* e ação de persistência};
- Literal *jantar* é gerado por {*cozinhar* e ação de persistência};
- Literal *presente* é gerado por {*embrulhar* e ação de persistência}.

Todos estes conjuntos devem ser combinados na tentativa da extração da solução, gerando 3 x 2 x 2 = 12 combinações de suporte para as ações no nível 3 ao invés de 2 x 1 x 1 = 2 combinações do nível 1. Esse aumento no número de possibilidades de ações geradoras aumenta as chances de que algum plano seja encontrado, mas aumenta também o trabalho de extração do plano.

Como exemplo, será usado o seguinte conjunto: ação *carregar* que suporta o sub-objetivo ¬*lixo*, ação de persistência que suporta o sub-objetivo *jantar* e ação *embrulhar* que suporta o sub-objetivo *presente*. Nenhuma dessas ações é *mutex* com outra, assim as escolhas para nível 3 são consistentes. A seleção dessas ações levam para os seguintes sub-objetivos do nível 2: *jantar* (precondição da ação de persistência) e *silêncio* (precondição da ação *embrulhar*), uma vez que a ação *carregar* não tem precondição, há somente 2 sub-objetivos para o nível 2 (*jantar* e *silêncio*). Todas as combinações de ações geradoras desse conjunto devem ser consideradas, mas o exemplo apresenta apenas um: ação *cozinhar* que suporta o *jantar* e ação de persistência que suporta o *silêncio*. Essas suas ações não são *mutex* e sendo assim, as seleções para o nível 1 são consistentes. As precondições dessas ações criam 2 sub-objetivos para o nível 0: *mãoslimpas* e *silêncio*. Uma vez que estas proposições estão presentes na condição inicial do problema, a seleção é consistente e uma solução de plano existe.

A Figura 3.8 ilustra o plano encontrado na extração da solução para o conjunto escolhido. As ações dentro das caixas pretas serão executadas enquanto todas as outras não. Note-se que o plano encontrado está parcialmente ordenado, pois aparecem duas

ações no nível 3 (*carregar* e *embrulhar*) que podem ser executadas em qualquer ordem tendo o mesmo efeito. No momento de relatar o resultado, o programa pode optar por {*cozinhar*, *carregar* e *embrulhar*} ou {*cozinhar*, *embrulhar* e *carregar*}.

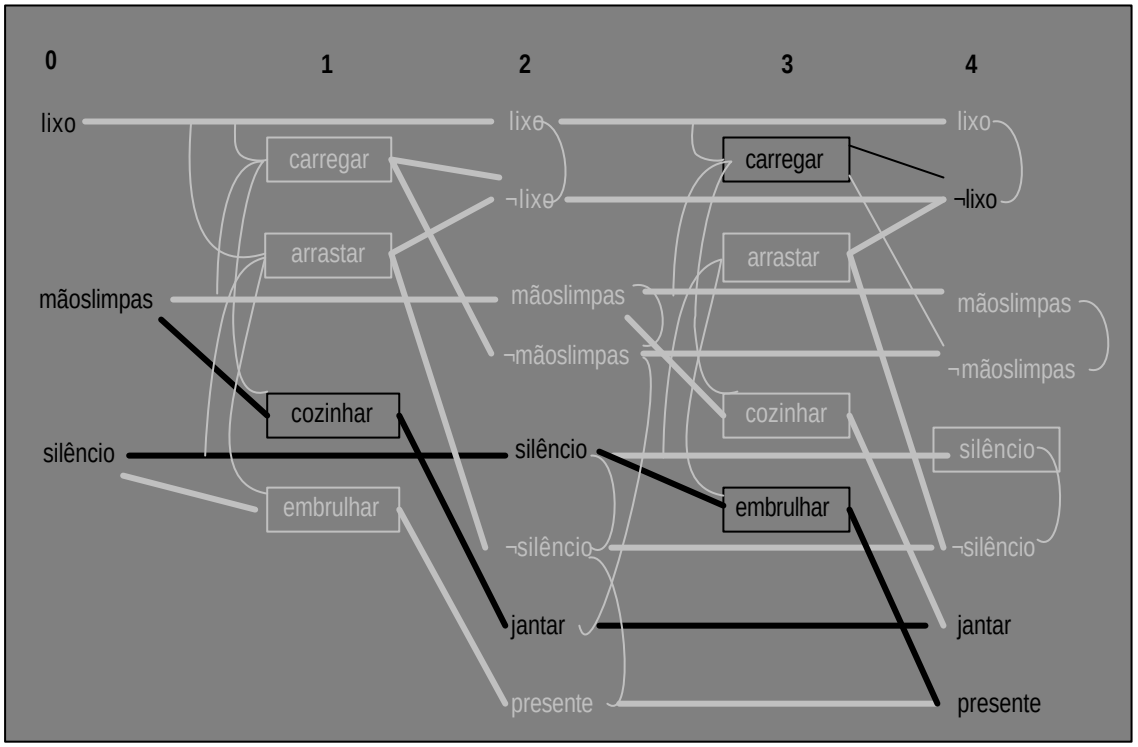

FIGURA 3.8 – Resultado de um plano encontrado na extração da solução.

FONTE: adaptada Weld (1999)

Além da estratégia original apresentada, o algoritmo GRAPHPLAN tem sido otimizado e amplamente discutido pela comunidade de IA nos últimos anos. Essas otimizações têm atingido uma grande melhoria na eficiência. Esses avanços não serão descritos aqui, pois representam otimizações e adaptações que não alteram a idéia central do planejamento em grafo.

3.2.2 Linguagens de Representação de Problemas de Planejamento

As linguagens de representação de problemas de planejamento são representações explícitas proposicionais ou de primeira ordem de estados e ações que tornam possível a derivação de heurísticas efetivas e o desenvolvimento de algoritmos poderosos e flexíveis para a resolução de problemas.

Em 1971, Fikes e Nilsson (1971) lançaram o primeiro sistema de planejamento de nome STRIPS que foi projetado como o componente de planejamento do software para o projeto de robótica Shakey no *Stanford Research Institute* (SRI). A representação de ações usada pela STRIPS teve muito mais influência que sua abordagem algorítmica. Quase todos os sistemas de planejamento desde então utilizaram uma ou outra variante da linguagem STRIPS. STRIPS é uma linguagem básica de representação de problemas de planejamento clássicos, descreve ações em termos de suas precondições e seus efeitos, e descreve o estado inicial e de objetivo como conjunção de literais positivos.

Nos últimos anos, ficou claro que a linguagem STRIPS era insuficientemente expressiva para problemas reais de planejamento. Assim foram desenvolvidas muitas variantes dessa linguagem. A linguagem ADL – *Action Description Language* (Pednault, 1989), por exemplo, é uma variação importante que relaxou algumas das restrições da linguagem STRIPS permitindo a representação de disjunção nas precondições, de condições nos efeitos, e de quantificadores universais nas precondições e efeitos tornando possível assim codificar problemas mais realistas.

Até 1998, não se tinha um formato padrão de representação a ser usado na definição de problemas entre a comunidade de planejamento. Com isso, os planejadores eram desenvolvidos usando diferentes tipos de entrada com formatos variáveis. Desta forma, era muito difícil comparar a eficiência dos planejadores para resolver problemas reais. Entretanto, em 1998, o comitê responsável pela organização da primeira Competição Internacional de Planejamento (IPC) disponibilizou a linguagem PDDL – Planning Domain Description Language para descrição de domínios de planejamento a ser utilizada na competição daquele mesmo ano – IPC/98 (McDermontt, 2000, McDermott et al., 1998). A PDDL foi introduzida como uma sintaxe padronizada e analisável por computador para representar STRIPS, ADL e outras linguagens e tem sido utilizada como a linguagem-padrão nos IPCs (IPC, 2005). Desde então, ela passou a ser para a comunidade de planejamento a linguagem-padrão para representação e troca de modelos no domínio de planejamento e tem facilitado o desenvolvimento científico na área de planejamento.

Também em 1998, iniciou-se um movimento decisivo dentro da comunidade de pesquisa para buscar aplicações de tecnologia de planejamento para problemas realistas que envolviam tempo e muitos tipos de recursos. Embora vários membros da comunidade estivessem trabalhando em aplicações de planejamento para problemas do mundo real, havia sempre um distanciamento entre os requisitos de modelagem de tais aplicações e o que podia ser expresso em PDDL. Assim, em 2002, para o terceiro IPC, uma extensão da PDDL, PDDL 2.1 foi disponibilizada com o intuito de diminuir o distanciamento entre a pesquisa e as aplicações do mundo real. A PDDL 2.1 permite modelar problema do mundo real que envolva tempo e restrições de recursos na geração de planos (Fox e Long, 2003).

Já para o IPC-2004, uma nova extensão da PDDL, PDDL 2.2, foi disponibilizada para permitir a modelagem de problemas de planejamento envolvendo eventos exógenos incondicionais determinísticos (isto é, representam fatos que se tornam verdadeiros ou falsos em pontos de tempos determinados independentemente das ações no plano) e predicados derivados (predicados que não são afetados por nenhuma das ações disponíveis no plano, mas cujos valores verdade são derivados de um conjunto de regras) (Edelkamp e Hoffmann, 2004). Esta nova versão permite modelar um número maior de problemas do mundo real.

A linguagem PDDL e suas extensões ainda não cobrem todas as necessidades da comunidade, principalmente quando são considerados os problemas de planejamento em ambientes não-clássicos (Frank, Golden e Jonsson, 2003; Berloti et al., 2003). Entretanto, o estabelecimento de um padrão de fato na representação de problema de planejamento vem possibilitando o compartilhamento de informações e troca de dados entre a comunidade, a realização dos IPCs e tem despertado o interesse do uso de agente de planejamento em aplicações reais.

3.2.2.1 Linguagem PDDL

Este item descreve a estrutura básica da linguagem PDDL (McDermontt et al., 1998, Fox; Long, 2003, Edelkamp, Hoffmann, 2004) e apresenta exemplos codificados em

linguagem PDDL 2.2 que foi a linguagem utilizada na implementação do protótipo neste trabalho.

A linguagem PDDL provê um fundamento para que padrões expressivos possam ser construídos, permitindo que modelos de domínios possam ser compartilhados entre comunidades de aplicações, e vem motivando o desenvolvimento no campo de planejamento por meio de aplicações realistas.

A PDDL é uma linguagem centrada na ação, inspirada pela bem conhecida formulação da linguagem STRIPS (Fikes e Nilsson, 1971) de problemas de planejamento. O seu núcleo é uma simples padronização de uma sintaxe para expressar semânticas familiares de ações, usando pré e pós-condições para descrever a aplicabilidade e os efeitos das ações.

A linguagem PDDL separa a descrição de ações parametrizadas, que caracterizam o comportamento do domínio de planejamento que se está modelando, da descrição das condições iniciais do ambiente e dos objetivos que se deseja alcançar no plano que caracterizam a instância do problema. Desta forma, o problema de planejamento é criado pela junção da descrição do domínio com a descrição do problema. Uma mesma descrição de domínio pode fazer par com várias diferentes descrições de problemas para produzir diferentes problemas de planejamento dentro de um mesmo domínio. A Figura 3.9 ilustra a base de conhecimento que é construída quando modelada em PDDL. Ela é composta por um arquivo de domínio e um ou mais arquivos de problemas.

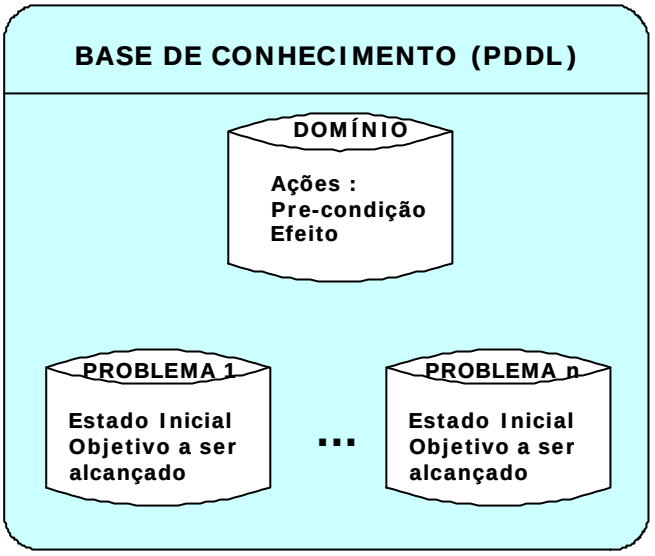

FIGURA 3.9 - Base de conhecimento em PDDL.

O arquivo de domínio consiste da descrição de todos os elementos que são necessários para modelar o comportamento do domínio para o qual o planejador deve encontrar um plano. Neste arquivo são definidos os tipos, as funções, os predicados (literais) e todas as ações do domínio. Para todas as ações é necessário definir a **precondição** para execução da ação e o **efeito** que ocorre após sua execução. A precondição é uma conjunção de literais que declara qual deve ser o estado do ambiente para que a ação possa ser executada. E o efeito é também uma conjunção de literais que declara qual será o estado do ambiente após a execução da ação. Para ações com duração, é necessário definir também a duração da execução da ação. A definição de duração foi introduzida na versão PDDL 2.1 (Fox; Long, 2003).

A Figura 3.10 ilustra um exemplo simplificado de um arquivo que mapeia um domínio de satélite para obter imagem de fenômenos espaciais. Este exemplo foi adaptado de um problema proposto no IPC do ano de 2002 (IPC, 2005) e está codificado em PDDL 2.2.

A sintaxe para a declaração de uma variável em PDDL é um símbolo precedido de um ponto de interrogação (?), seguido de um hífen (-) e do seu tipo. Este tipo é um dos tipos definidos na lista de tipos de entidades (*types*). Por exemplo, na definição do predicado (on_board ?i - instrument ?s - satellite) duas variáveis foram declaradas: **i** do tipo instrument e **s** do tipo satellite.

```
(define (domain satellite)
 (:requirements :strips :equality :typing :fluents :durative-actions)
 (:types satellite direction instrument mode)
 (:predicates
              (on_board ?i - instrument ?s - satellite)
              (supports ?i - instrument ?m - mode)
              (pointing ?s - satellite ?d - direction)
              (power_avail ?s - satellite)
              (power_on ?i - instrument)
              (have_image ?d - direction ?m - mode) )
 (:functions (slew_time ?a ?b - direction))

 (:durative-action turn_to
  :parameters (?s - satellite ?d_new - direction ?d_prev - direction)
  :duration (= ?duration (slew_time ?d_prev ?d_new))
  :condition (and (at start (pointing ?s ?d_prev)) )
  :effect (and  (at end (pointing ?s ?d_new))
            (at start (not (pointing ?s ?d_prev)))))

 (:durative-action switch_on
  :parameters (?i - instrument ?s - satellite)
  :duration (= ?duration 2)
  :condition (and (over all (on_board ?i ?s))
                 (at start (power_avail ?s)))
  :effect (and (at end (power_on ?i))
           (at start (not (power_avail ?s)))))

 (:durative-action switch_off
  :parameters (?i - instrument ?s - satellite)
  :duration (= ?duration 1)
  :condition (and (over all (on_board ?i ?s))
                 (at start (power_on ?i)))
  :effect (and (at start (not (power_on ?i)))
           (at end (power_avail ?s))) )

 (:durative-action take_image
  :parameters (?s - satellite ?d - direction ?i - instrument ?m - mode)
  :duration (= ?duration 7)
  :condition (and    (over all (on_board ?i ?s))
                 (over all (supports ?i ?m) )
                 (over all (power_on ?i))
                 (over all (pointing ?s ?d))
                 (at end (power_on ?i)))
  :effect (and (at end (have_image ?d ?m)))))
```

FIGURA 3.10 - Exemplo do arquivo de domínio de satélite.

No exemplo do arquivo de domínio de satélite (Figura 3.10), os tipos de entidades definidos para o domínio foram:

- *satellite* - nome do satélite;
- *direction* - para onde está apontando o satélite;
- *instrument* - nome do instrumento de bordo que obtém a imagem; e
- *mode* - modo de operação do instrumento de bordo para obter a imagem.

Os predicados definidos no exemplo que representam o conhecimento do domínio de satélite foram os seguintes:

- (on_board ?i - instrument ?s - satellite) - o instrumento **i** está a bordo do satélite **s**;
- (supports ?i - instrument ?m - mode) - o instrumento **i** suporta o modo de operação **m**;
- (pointing ?s - satellite ?d - direction) - o satélite **s** está apontado para a direção **d**;
- (power_avail ?s - satellite) - o satélite **s** tem energia suficiente para obter a imagem;
- (power_on ?i - instrument) - o instrumento **i** está ligado;
- (have_image ?d - direction ?m - mode) - imagem obtida da direção **d** no modo de operação **m**.

A única função definida, *slew_time*, fornece o tempo necessário para que o satélite mude de uma direção para outra.

No exemplo, o conjunto de ações necessárias para se obter a imagem de um fenômeno espacial foi o seguinte:

- *turn-to* - apontar para uma direção específica;
- *switch-on* - ligar um instrumento de bordo;
- *switch-off* - desligar um instrumento de bordo;
- *take-image* - obter uma imagem espacial.

As definições das ações utilizam os tipos, os predicados e as funções para mapear o seu comportamento. As ações serão inseridas no plano pelo planejador se forem aplicáveis ao estado do ambiente, ou seja, se sua precondição é satisfeita nesse instante de tempo. Na ação *turn-to* ilustrada na Figura 3.11 por exemplo, a precondição é que o satélite esteja apontado para uma direção prévia e o efeito da execução dessa ação será o satélite mudar seu apontamento para uma nova direção. O tempo gasto para execução dessa ação é dado pela função *slew-time.*

```
(:durative-action turn_to
 :parameters (?s - satellite ?d_new - direction ?d_prev - direction)
 :duration (= ?duration (slew_time ?d_prev ?d_new))
 :condition (and (at start (pointing ?s ?d_prev)) )
 :effect (and  (at end (pointing ?s ?d_new))
            (at start (not (pointing ?s ?d_prev)))))
```

FIGURA 3.11 - Definição da ação *turn-to.*

Como se pode perceber, o arquivo de domínio se aplica a qualquer satélite que tem como característica obter imagens. O arquivo de problema é que instancia o domínio para o ambiente desejado. Na Figura 3.12 está ilustrado um arquivo de problema definido para o arquivo de domínio de satélite apresentado na Figura 3.10.

```
(define (problem sat-example)
(:domain satellite)
(:objects
        satellite0 - satellite
        instrument0 - instrument
        image1 - mode
        spectrograph2 - mode
        thermograph0 - mode
        GroundStation2 - direction
        Phenomenon4 - direction)
(:init
    ;; Definição do valor da função
        (= (slew_time Phenomenon4 GroundStation2) 39.73)
        (= (slew_time GroundStation2 Phenomenon4) 40.00)
    ;;Estado inicial do ambiente
        (supports instrument0 thermograph0)
        (on_board instrument0 satellite0)
        (power_avail satellite0)
        (pointing satellite0 GroundStation2))
(:goal (and
        (have_image Phenomenon4 thermograph0))))
```

FIGURA 3.12 - Exemplo do arquivo de problema para o domínio de satélite.

Nos arquivos de problemas são definidas as seguintes informações: objetos (objects), estado inicial do ambiente (init) e o estado objetivo (goal).

Os objetos são instâncias dos tipos definidos no arquivo de domínio, ou seja, os artefatos reais que envolvem o domínio do problema. No exemplo, os tipos definidos no domínio são instanciados para os seguintes objetos: satellite = satellite0, instrument = instrument0, mode = image1, spectograph2 e thermograph0, e direction = GroundStation2 e Phenomenon4.

O estado inicial (*init*) descreve o estado inicial do ambiente ao iniciar a execução do planejador. Todas as informações importantes para definir o estado do ambiente no início do planejamento devem ser fornecidas. Na verdade, são definidos os predicados que são verdadeiros de forma a mapear o estado inicial do ambiente. No exemplo, o estado inicial para o problema de planejamento foi definido como:

- (supports instrument0 thermograph0) – o instrumento *instrument0* suporta o modo de operação *thermograph0*;

- (on_board instrument0 satellite0) - o instrumento *instrument0* está a bordo do satélite *satellite0*;

- (power_avail satellite0) - o satélite *satellite0* tem energia disponível; e

- (pointing satellite0 GroundStation2) – o satélite *satellite0* está apontado para a direção *GroundStation2*.

É na seção *init:* que também é definido o valor de retorno de cada função incluída no arquivo de domínio. No exemplo, a função *slew-time* definida no arquivo de domínio tem dois valores de retorno dependendo dos objetos instanciados para o tipo especificado.

O estado objetivo é o estado que se deseja alcançar após a execução do plano. Ou seja, o planejador, de posse do estado inicial do ambiente, deve encontrar uma seqüência de ações que atinja o objetivo do problema de planejamento desejado. No exemplo, o objetivo é obter a imagem do *Phenomenon4* no modo de operação *thermograph0*.

O planejador de posse do arquivo de domínio e do arquivo de problema, por meio da aplicação de algoritmos e de derivação de heurísticas, encontra uma seqüência de ações que atenda o objetivo proposto. A Figura 3.13 ilustra o ambiente necessário para a geração automática de um plano onde o planejador de posse da base de conhecimento gera um plano.

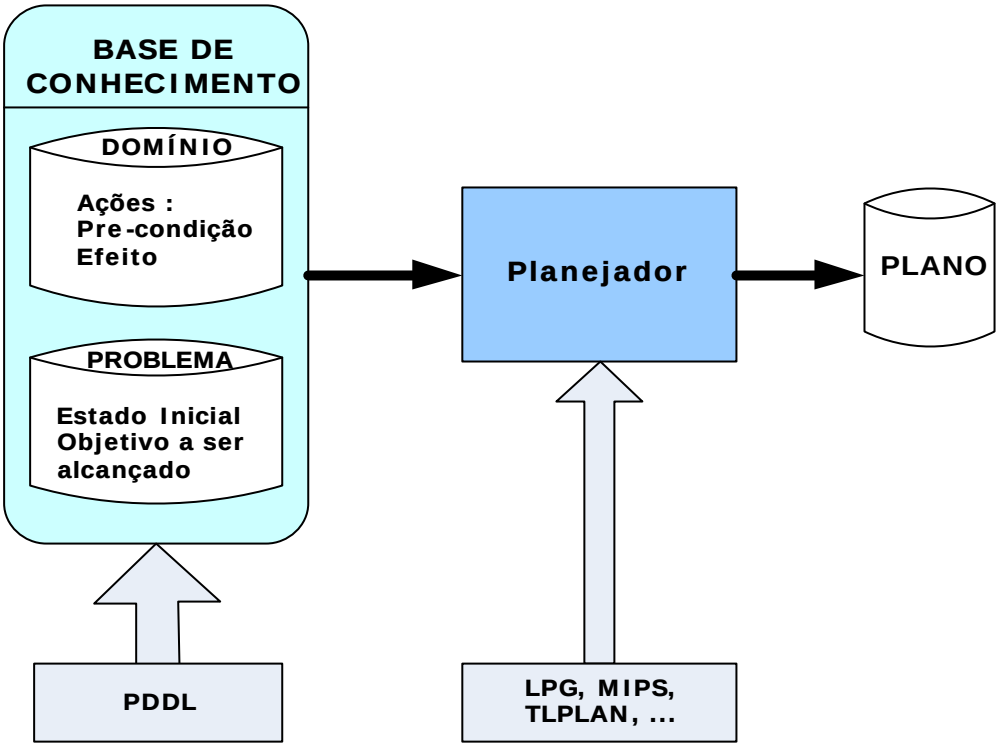

FIGURA 3.13 – Ambiente de geração automática de planos.

Alguns planejadores que suportam o formato PDDL 2.1 e geram planos para problemas de planejamentos temporais são:

- LPG - *Local Searching for Planning Graphs* (Gerevini e Serina, 2002, Gerevini, Saetti e Serina, 2003, Gerevini, Saetti e Serina, 2004);
- MIPS - *Model checking Integrated Planning System* (Edelkamp e Helmert 2001);
- TALPlanner – *Temporal Action Logics Planner* (Doherty e Kvarnstrom, 2001, Kvarnström e Magnusson, 2003);
- TLPlan – *Temporal Logical Plan* (Bacchus e Ady, 2001];
- SGPlan – *Subgoal partitioning and Global search Planner* (Wah e Chen, 2004);
- SAPA (Do e Kambhampati , 2003).

Os dois planejadores que suportam a extensão PDDL 2.2 são:

- LPG-TD (Gerevini et al., 2004, Gerevini, Saetti e Serina, 2005) e
- TilSAPA (Kavuluri e Khemani, 2004).

A seqüência de ações obtidas pelo planejador LPG-TD para a base de conhecimento exemplo do domínio de satélite (Figura 3.10 e Figura 3.12) está mostrada na Figura 3.14. Como pode ser observado, a última ação inserida no plano quando executada tem como efeito a situação do objetivo desejado no arquivo de problema (*have_image* Phenomenon4 thermograph0). Um mesmo arquivo de domínio pode ser utilizado para vários arquivos de problemas.

; Version LPG-td-1.0

Time 0.03

0.0003: (**turn_to** satellite0 phenomenon4 groundstation2) [39.7300]
0.0005: (**switch_on** instrument0 satellite0) [2.0000]
39.7308: (**take_image** satellite0 phenomenon4 instrument0 thermograph0) [7.0000]

FIGURA 3.14 - Plano gerado para o domínio de satélite.

Uma característica importante inserida na PDDL 2.2 (Edelkamp, Hoffmann, 2004), foi permitir a definição de eventos exógenos incondicionais determinísticos no arquivo de problema, ou seja, eventos que devem ocorrer em um determinado instante do tempo independente da seqüência das ações do plano. A Figura 3.15 ilustra o arquivo de problema com a definição do predicado *power-avail* como um evento exógeno. Esta definição significa que o satélite só terá energia disponível no intervalo de tempo entre 100 a 150 segundos do início do plano. Desta forma, o planejador deve respeitar a janela de tempo definida.

```
(define (problem sat-example)
(:domain satellite)
(:objects
        satellite0 - satellite
        instrument0 - instrument
        image1 - mode
        spectrograph2 - mode
        thermograph0 - mode
        GroundStation2 - direction
        Phenomenon4 - direction
)
(:init
    ;; Definição do valor da função
        (= (slew_time Phenomenon4 GroundStation2) 39.73)
        (= (slew_time GroundStation2 Phenomenon4) 40.00)
    ;;Estado inicial do ambiente
        (supports instrument0 thermograph0)
        (on_board instrument0 satellite0)
    ;;(power_avail satellite0) foi definido como um evento exógeno
        (at 100.0 (power_avail satellite0))
        (at 150.0 (not (power_avail satellite0)))

        (pointing satellite0 GroundStation2)
)
(:goal (and
        (have_image Phenomenon4 thermograph0))
)
)
```

FIGURA 3.15 - Exemplo de definição de evento exógeno no arquivo de problema.

A seqüência de ações encontradas para o arquivo de problema da Figura 3.15 pelo planejador LPG-TD está mostrada na Figura 3.16. Observe-se que a seqüência das ações é a mesma encontrada no exemplo anterior, ilustrado pela Figura 3.14, mas o tempo de execução da ação *swicth-on* que tem o predicado *power-avail* como condição para execução deve ocorrer no tempo de 100 segundos após o início do plano e não 0.0005 como no resultado anterior.

```
; Version LPG-td-1.0

Time 0.03

0.0005:      (turn_to satellite0 phenomenon4 groundstation2) [39.7300]
100.0003:    (switch_on instrument0 satellite0) [2.0000]
102.0007:    (take_image satellite0 phenomenon4 instrument0 thermograph0) [7.0000]
```

FIGURA 3.16 – Plano para o domínio de satélite respeitando o evento exógeno.

3.2.3 Planejador LPG-TD

O planejador LPG-TD (Gerevini et al., 2004, Gerevini, Saetti e Serina,2005) foi o planejador utilizado no protótipo desenvolvido do trabalho. O LPG-TD é uma extensão do planejador LPG (Gerevini e Serina, 2002, Gerevini, Saetti e Serina, 2003, Gerevini, Saetti e Serina, 2004) e tem a capacidade de interpretar as características da linguagem PDDL 2.2.

O LPG-TD é um planejador completamente automático e incremental de geração de planos para os problemas que envolvam os seguintes elementos:

- Ações STRIPS;
- Ações com duração;
- Ações e objetivos que envolvem expressões numéricas;
- Operadores com efeitos de quantificação universal;
- Operadores com precondições de quantificação existencial;
- Operadores com precondições disjuntivas;
- Operadores com precondições implicativas;
- Literais iniciais temporizados (eventos exógenos incondicionais determinísticos);

- Predicados derivados por axiomas do domínio;
- Maximização e minimização de métricas de planos complexos.

Como no LPG, o LPG-TD também está baseado em pesquisa local estocástica em um espaço particular de grafos de ações derivado da especificação do problema de planejamento. No LPG-TD, as representações de grafos têm sido estendidas para lidar com as novas características da linguagem PDDL 2.2 (predicados derivados e literais iniciais temporizados), bem como, para melhorar o gerenciamento das ações com duração e das expressões numéricas.

O espaço de busca do planejador é formado por sub-grafos particulares do grafo de planejamento, chamado de grafos de ações que representam planos parciais. Dado um grafo de planejamento *G*, o processo de busca local começa de um grafo de ação inicial *A* de *G* (isto é, um grafo parcial), e transforma-o em um grafo de solução por meio da aplicação interativa de algumas modificações no grafo *A* melhorando o plano parcial corrente. Essas modificações podem ser realizadas de duas formas: (i) adicionando um novo nodo de ação no grafo de ação corrente; (ii) ou, removendo um nodo de ação do grafo juntamente com os níveis correspondentes de nodos de precondições e efeitos do grafo de planejamento.

Em qualquer passo no processo de busca, o qual produz um novo grafo de ação, o conjunto de ações que podem ser adicionadas ou removidas é determinado pelas violações de restrições que existem no grafo de ação corrente. A violação de restrição (ou de inconsistência) é qualquer relação *mutex* que envolve nodos de ação no grafo de ação corrente, ou um literal não suportado que é precondição de uma ação no plano parcial corrente.

O esquema geral para encontrar um grafo solução (isto é, um estado final da busca) consiste de dois passos principais. O primeiro passo é a inicialização da busca na qual é construído o grafo de ação inicial. O segundo passo é o processo de busca local no espaço de todo os grafos de ação, iniciando pelo grafo de ação inicial.

Neste planejador, as técnicas de busca local básicas foram melhoradas por meio do uso de uma função parametrizada de avaliação de ação que permite especificar diferentes tipos de heurísticas. Esta função especifica o custo de inserir e de remover uma ação do plano parcial representado pelo grafo de ação corrente. Várias técnicas são usadas para definir e avaliar a busca na vizinhança de modo a explorar a estrutura de um grafo subjacente. Estas técnicas incluem os multiplicadores de Lagrange, a propagação No-op, a ordenação de ações e o tamanho do grafo incremental, e os custos de precondições heurísticas (Gerevini e Serina, 2002).

Em relação à qualidade do plano gerado, o planejador leva em consideração o custo de execução de uma ação. Os custos de execução das ações são representados como números associados com as mesmas. O planejador produz planos de boa qualidade por meio da minimização da função objetivo que inclui um termo para a avaliação do custo total de execução de um plano. Isto é feito por um processo que gera uma série de planos onde cada plano gerado é melhor do que o anterior. Similarmente, o planejador pode gerar planos de boa qualidade em termos do número de passos paralelos no plano, e pela duração total do plano.

O enfoque dado no LPG-TD para o tratamento de problemas de planejamento temporal onde as ações têm durações e devem ser executadas em uma determinada janela de tempo, foi à combinação de duas técnicas: i) algoritmo polinomial para raciocínio temporal baseado em restrições durante a busca; e ii) técnicas de busca local para planejamento que exploram informação temporal (Gerevini, Saetti e Serina, 2005).

O planejador LPG-TD obteve o segundo lugar no IPC do ano de 2004 (IPC, 2005). O primeiro lugar foi para o planejador SGPlan (Wah e Chen, 2004) que ainda não está preparado para aceitar o formato PDDL 2.2.

3.3 Engenharia de Conhecimento

Durante o desenvolvimento de um agente inteligente, uma das primeiras atividades a ser realizada é a geração da base de conhecimento que representa o conhecimento do

domínio para qual se está projetando o agente. Devido à importância desta atividade este item descreve um processo para geração de uma base de conhecimento consistente.

Primeiramente, o termo representação de conhecimento será definido. Segundo Sowa (Sowa, 1999), a representação do conhecimento é a aplicação da lógica e da ontologia na tarefa de construir modelos computacionais para algum domínio e para algum propósito.

Para Davis, Shrobe e Szolovits (1993) a representação de conhecimento pode ser definida através de cinco papéis onde cada um deles apresenta diferentes, e às vezes conflitantes, demandas nas propriedades que uma representação deve ter.

Primeiro, a representação do conhecimento é fundamentalmente um "representante", um substituto para as coisas propriamente ditas, que é usado para capacitar uma entidade a determinar conseqüências pelo pensamento e não pela ação. Ou seja, ter um raciocínio sobre o mundo ao invés de ter ações sobre ele.

Segundo, é um conjunto de comportamentos ontológicos que responde à seguinte questão: "De que maneira eu devo pensar sobre o mundo?".

Terceiro, é uma teoria fragmentada do raciocínio inteligente expressa por meio de três componentes: (i) concepção fundamental de representação do raciocínio inteligente; (ii) o conjunto de inferências que ratificam as representações e; (iii) o conjunto de inferências que é recomendado. O termo inferência é usado em um senso genérico que significa qualquer caminho para obter novas expressões a partir das antigas.

Quarto, é um meio para uma computação eficiente, isto é, o ambiente computacional no qual o pensamento é realizado. Uma contribuição para esta eficiência pragmática é fornecida pela orientação que a representação provê para organizar as informações de forma a facilitar a criação das inferências recomendadas.

Quinto, é um meio de expressão humana, isto é, uma linguagem com a qual se diz coisas sobre o mundo.

Definido o termo representação do conhecimento, definiremos agora o termo engenharia de conhecimento. Engenharia do conhecimento é o nome dado para o processo geral de construção de uma base de conhecimento. Um engenheiro de conhecimento é alguém que investiga um domínio específico, aprende quais conceitos são importantes sobre esse domínio e cria uma representação formal dos objetos e relações no domínio.

Os projetos de representação de conhecimento variam amplamente em conteúdo, escopo e dificuldades, mas todos eles podem ser desenvolvidos através das seguintes etapas, definidas por Russell e Norvig (2004), para o processo de construção da base de conhecimento:

a) **Identificação da tarefa**: Nesta etapa deve ser delineada a variedade de questões que a base de conhecimento admitirá e os tipos de fatos que estarão disponíveis para cada instância específica de problema. A tarefa determinará que conhecimento deve ser representado, com a finalidade de conectar instâncias de problemas com as respectivas respostas. Esta etapa é análoga ao processo de definição do ambiente de tarefa do agente descrita no item 3.1.

b) **Aquisição de conhecimento**: O engenheiro do conhecimento deve ser um especialista no domínio que se deseja modelar ou talvez precise trabalhar com especialistas reais para extrair o conhecimento deles sobre o domínio. Nessa etapa, o conhecimento não é representado formalmente. A idéia é compreender o escopo da base de conhecimento, determinado pela tarefa, e entender como o domínio realmente funciona.

c) **Definição da ontologia do domínio**: Nesta etapa os importantes conceitos do nível de domínio são convertidos em nomes no nível de lógica. Define-se um vocabulário de predicados, funções e constantes que é chamado de ontologia do domínio. A palavra ontologia representa uma teoria específica sobre a natureza de ser ou existir. A ontologia determina os tipos de itens que existem, mas não determina suas propriedades específicas e seus inter-relacionamentos.

d) **Codificação do conhecimento geral do domínio**: Nesta etapa, todos os termos do vocabulário são escritos em axiomas correspondentes. Isso fixa o significado dos termos, permitindo ao especialista verificar o conteúdo. Com freqüência, nessa etapa, são encontradas concepções erradas e lacunas no vocabulário que devem ser corrigidas retornando a etapa de "definição da ontologia do domínio" e repetindo-se todo o processo.

e) **Codificação da instância específica do problema**: Se a ontologia estiver bem elaborada, essa etapa será fácil. Ela envolverá a escrita de sentenças atômicas simples (predicado e termos) sobre instâncias de conceitos que já fazem parte da ontologia. Para um agente lógico, as instâncias de problemas são fornecidas pelos sensores, enquanto uma base de conhecimento "desincorporada" é suprida com sentenças adicionais, da mesma forma que os programas tradicionais são supridos com dados de entrada.

f) **Formulação de consultas ao procedimento de inferência e obtenção de respostas**: Nessa etapa é que se deixa o procedimento de inferência operar sobre os axiomas e fatos específicos do problema para derivar os fatos que se está interessado em conhecer.

g) **Depuração da base de conhecimento**: Se os axiomas descrevem correta e completamente o modo como o mundo funciona e o modo como as percepções são produzidas então, qualquer procedimento completo de inferência lógica deduzirá a descrição mais forte possível do estado do mundo, dadas as percepções disponíveis. Mas infelizmente, muitas vezes as respostas obtidas das consultas não são as esperadas. Isto significa que as respostas estarão corretas para a base de conhecimento criada, supondo-se que o procedimento de inferência seja consistente, mas não são as respostas que o usuário esperava. Nesta etapa, deve-se testar a base de conhecimento de várias maneiras, a fim de verificar que tipos de comportamento errôneos emergem. Se problemas forem encontrados, deve-se retornar às etapas anteriores para correção, repetindo-se todo o processo se necessário.

CAPÍTULO 4

SOLUÇÃO PARA GERAÇÃO DE PLANOS DE OPERAÇÕES DE VÔO PARA SATÉLITES USANDO AGENTE DE PLANEJAMENTO

Como visto no Capítulo 2, a busca por uma solução de automatização aplicada às atividades de operação de satélites artificiais é uma necessidade mundial para reduzir os custos das missões e aumentar a confiabilidade das operações. No INPE uma grande parte destas atividades é ainda feita de forma manual. Encontrar uma solução automatizada para as operações de satélites do INPE é de extrema importância para conseguir manter o bom desempenho nas operações de rastreio dos satélites levando em conta o aumento previsto do número de satélites a serem operados e a limitação de recursos financeiros disponíveis.

Com este enfoque, o trabalho propõe a utilização de técnicas de planejamento temporal da IA na geração automática de POVs para atender à fase operacional de rotina.

Para contextualizar o trabalho, primeiramente será mostrado, em linhas gerais, como atualmente é preparada e executada a operação de satélites no CRC do INPE na fase de rotina. Em seguida, será apresentada a solução proposta para a geração automática de planos de operação de vôo para satélites usando a tecnologia de agentes..

4.1 Preparação e Execução da Operação de Satélites no INPE

As principais atividades necessárias na preparação e execução do POV para fase de operação em rotina de um satélite no CRC do INPE estão representadas na Figura 4.1.

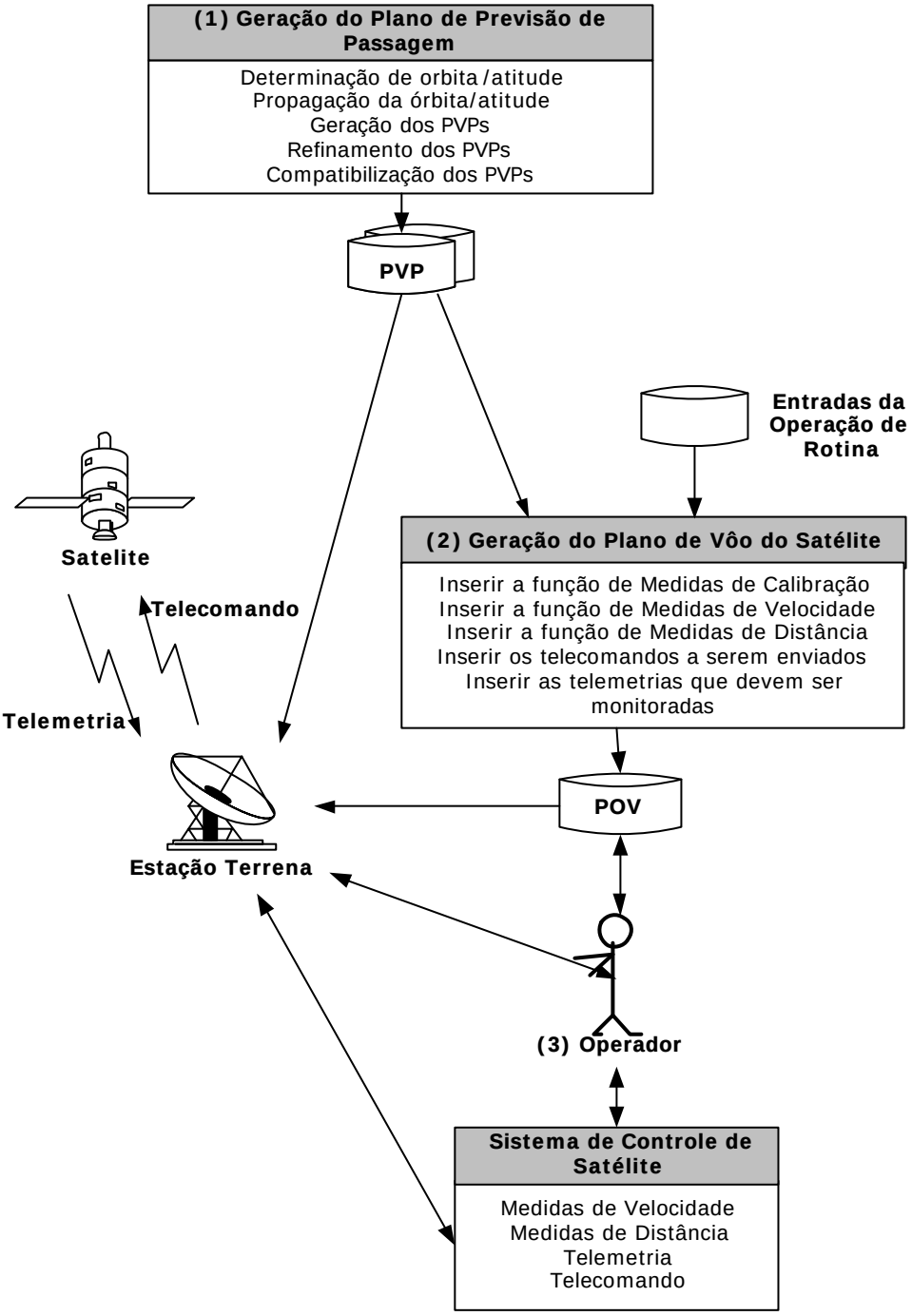

FIGURA 4.1 - Preparação e Execução do POV.

Primeiramente, um Plano de Previsão de Passagem (PVP) para um período de uma semana é gerado para cada par satélite - estação terrena (ET), a partir dos dados de órbita e atitude. Cada arquivo PVP gerado contém várias linhas de informações, cada linha contendo, entre outros dados: o instante de tempo (dia/mês/ano, hora, minuto e

segundo), os valores dos ângulos de azimute e elevação do satélite neste instante (para o apontamento da antena da ET em direção ao satélite), a distância entre a ET e o satélite, a variação temporal desta distância e o valor do desvio de Doppler na freqüência do sinal portador enviado ao satélite. O intervalo de tempo em que cada linha do PVP é gerada pode ser definido pelo usuário. O CRC utiliza em suas atividades operacionais um intervalo de trinta segundos entre cada linha de informações.

Em seguida é realizada a seleção das passagens do satélite sobre a ET que serão realmente utilizadas através da aplicação de um conjunto de critérios. Como exemplo: (i) conflitos de passagens simultâneas de mais de um satélite sobre uma mesma ET são solucionados por aplicação de regras de prioridades de rastreio dos satélites; (ii) em caso de passagens simultâneas de um satélite sobre duas ou mais ETs, a ET com maior prioridade será selecionada; e (iii) para cada ET um intervalo mínimo entre o fim da passagem de um satélite e o início da passagem de outro satélite deve ser respeitado, para que a passagem possa ser considerada útil.

Após a geração e compatibilização dos PVPs, a atividade de geração do POV é realizada para cada satélite. O período de planejamento do POV corresponde ao período coberto pelo PVP, no caso uma semana. No POV são indicadas todas as ações de operação que o operador do satélite deve executar antes, durante e após cada passagem do satélite. A indicação das atividades sempre é feita em instantes correspondentes aos das linhas de informação lidas pelo gerador do POV nos arquivos PVPs.

Os operadores do CRC controlam e monitoram o satélite executando as atividades contidas no POV. Para executar estas atividades, os operadores utilizam o Sistema de Controle de Satélites (SICS) que é o chamado software aplicativo de tempo real do CCS, composto dos subsistemas de telemetria, telecomandos, medidas de distância e medidas de velocidades. Durante a passagem de um satélite na região de visibilidade de uma ET, o SICS comunica-se em tempo real com o satélite, utilizando a rede privada de comunicação de dados que liga o CCS com as ETs, enviando telecomandos, recebendo telemetria e realizando medidas de distância e velocidade. As funções do SICS, correspondentes às atividades contidas no POV, são acionadas manualmente pelos

operadores nos respectivos instantes previstos para as mesmas. Após a execução das atividades, os operadores registram no POV, os resultados obtidos.

O cenário de operação típica dos satélites controlados pelo INPE durante uma passagem sobre uma estação terrena inclui as seguintes ações:

- Execução de medidas de calibração do equipamento de medida de distância antes do horário previsto para o início da passagem;

- Recepção de telemetria que é uma ação passiva, ou seja, o sistema de telemetria fica disponível durante toda passagem para receber automaticamente as telemetrias enviadas pelo satélite;

- Envio de telecomando. Esta ação está relacionada ao satélite sendo rastreado. Cada satélite possui um conjunto diferente de telecomandos. Os telecomandos podem ser classificados de acordo com o seu envio da seguinte forma:

    - Todas as passagens (telecomandos que devem ser enviados em todas as passagens úteis);

    - Diários (devem ser enviados todos os dias);

    - Periódicos (aqueles que são enviados de acordo com a rotina de operação estabelecida para o satélite. Por exemplo, todas as segundas-feiras, os telecomandos para as cargas úteis do CBERS devem ser enviados);

- Execução de medida de distância entre o satélite e a ET. Esta ação tem uma duração e deve ser executada quantas vezes for possível durante a passagem, mas não pode ser concorrente com a ação de envio de telecomando. As medidas coletadas são usadas na geração do próximo PVP;

- Execução de medida de velocidade do satélite em relação a ET. Esta ação também deve ser executada durante toda a passagem. As medidas coletadas também são entradas para a geração do próximo plano de previsão de passagem.

A inserção destas ações no POV deve respeitar várias restrições, como por exemplo: (i) um pedido de medidas de distância entre o satélite e a ET ou envio de telecomandos só podem ser inseridos no POV em instantes no qual o ângulo de elevação previsto do satélite é maior que um valor mínimo pré-estabelecido; (ii) os períodos de zona de silêncio (período da passagem durante o qual a comunicação do satélite com a ET é interrompida) devem ser respeitados, ou seja, nenhuma ação pode ser executada durante este período; (iii) o intervalo de tempo mínimo após o início da passagem para que o transmissor do satélite (responsável pelo envio de telemetria para o solo) possa ser ligado; e (iv) a duração mínima de uma passagem para que seja possível enviar telecomandos e/ou executar medidas de distância e de velocidade.

Para cada um dos satélites SCD1 e SCD2, são rastreadas 57 passagens em média durante o período de uma semana por estarem estes satélites em órbitas de baixa inclinação. Já para o satélite CBERS2, que tem uma órbita polar heliosíncrona, são rastreadas 27 passagens em média durante o período de uma semana. As passagens do CBERS2 têm prioridade sobre as passagens dos SCDs, ou seja, as passagens dos SCDs que são concorrentes com as do CBERS2 são descartadas. Como o ambiente do rastreio muda a cada passagem, um POV para cada passagem do satélite é gerado para cada ET utilizada em seu rastreio.

O desafio no planejamento dos POVs para a operação dos satélites do INPE está em garantir que todas as ações a serem executadas sejam inseridas no plano em instantes adequados, respeitando as janelas de visibilidade do satélite em relação a ET responsável por rastreá-lo. Além da restrição das janelas de visibilidade, todas as restrições para inserção de cada ação no plano devem ser respeitadas, considerando-se tanto restrições de tempo como de recursos.

4.2 Arquitetura do Sistema de Planejamento Inteligente de Planos de Operações de Vôo

A solução proposta para a geração de POVs para satélites usando agente de planejamento se encaixa na segunda atividade, geração de POVs, que foi descrita no item 4.1.

Os requisitos funcionais que orientaram a geração da solução estão listados a seguir:

a) Gerar POVs para os satélites controlados pelo INPE levando em consideração todas as restrições de tempo e recursos impostas pelo problema de planejamento;

b) Permitir que o operador escolha uma seqüência inteira de passagens, que corresponde a um período de uma semana de passagens de um satélite sobre uma determinada ET, ou que escolha um sub-conjunto de passagens desta seqüência para a geração do POV.

c) Permitir que o operador visualize os POVs gerados por meio da escolha da passagem desejada;

d) Permitir que os POVs gerados sejam editados pelo operador para melhor se ajustarem a situações específicas de operação do satélite em que atividades extras possam ser incluídas e/ou atividades existentes possam ser removidas;

e) Permitir que a configuração de operação seja alterada e aplicada a toda uma seqüência de passagens ou a passagens específicas durante a geração do POV.

Além dos requisitos funcionais apresentados, partiu-se também dos seguintes pressupostos:

- A tecnologia de planejamento para resolução de problemas de planejamento temporal da área da IA seria usada;

- A arquitetura de um agente de planejamento estaria inserida na arquitetura global do sistema e que o módulo Executor não seria implementado (ver item 3.2, Figura 3.3);

- A linguagem PDDL 2.2 seria utilizada para modelar o domínio de rastreio de satélite, pois o problema de planejamento do POV é um problema de planejamento temporal com restrições de tempo e de recurso. Além disso, o gerador de POV tem que ser capaz de planejar suas ações dentro de uma janela temporal bem definida (vide item 3.2.2.1). Por exemplo, o tempo de visibilidade do satélite pela ET e o tempo de zona de silêncio;

- O planejador a ser utilizado para a geração do plano deveria ser aquele que tivesse capacidade de manipular o formato da linguagem PDDL 2.2 e que implementasse a solução para o problema de planejamento que envolve eventos exógenos incondicionais determinísticos.

O sistema de Planejamento Inteligente de Planos de Operação de Vôo (PlanIPOV) é a solução proposta que responde aos requisitos descritos acima.

O digrama de contexto do sistema PlanIPOV está apresentado na Figura 4.2. O sistema tem como dado de entrada o arquivo PVP que contem o plano de previsão de passagens de um determinado satélite para uma ET específica. Os arquivos PVP são gerados pelo software aplicativo de Dinâmica Orbital existente no CCS. A atividade de geração dos arquivos PVP é a primeira atividade dentro da preparação das operações do satélite pelo CRC como descrito no item 4.1. De posse do arquivo PVP e das opções escolhidas pelo operador do CCS, o PlanIPOV é capaz de gerar um POV para cada passagem do satélite escolhida. Os POVs gerados podem ser visualizados e editados pelo operador.

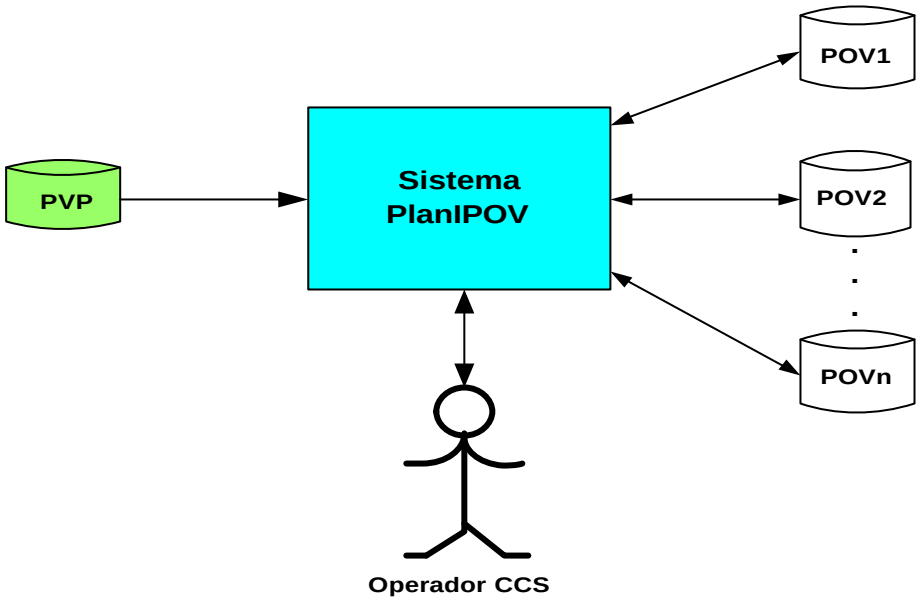

FIGURA 4.2 – Diagrama de contexto do sistema PlanIPOV.

A arquitetura do sistema PlanIPOV está ilustrada na Figura 4.3 e é composta de cinco módulos: Configurador, Gerador de Problema, Planejador, Visualizador POV e Interface Gráfica.

Os arquivos internos manipulados pelo sistema são: o arquivo de Domínio, os arquivos de Problemas e a base de dados de Configuração. O **arquivo de Domínio** é o arquivo que contém a base de conhecimento do domínio de rastreio de satélite codificada em PDDL 2.2 e é gerado manualmente. Os **arquivos de Problemas** são os arquivos que são gerados para cada passagem a ser rastreada e contém o estado inicial do ambiente da passagem e o objetivo a ser atingido no POV. Estes arquivos são gerados de forma automática pelo sistema. A **base de dados de Configuração** contém todos os parâmetros de configuração dos satélites e das ETs envolvidas no rastreio. Duas categorias de parâmetros são encontradas nesta base: (i) parâmetros fixos, ou seja, os seus valores são sempre os mesmos independentemente da passagem a ser rastreada e (ii) parâmetros configuráveis, ou seja, os seus valores podem ser alterados e essas alterações influenciam no resultado final do POV gerado. Por exemplo, um parâmetro configurável seria o status da função de Calibração que indica se esta função está ativada ou não para ser realizada durante o rastreio de um determinado satélite em uma

determinada ET. Se o parâmetro estiver ativado, o predicado que indica que a função de calibração está disponível deverá ser inserido como verdadeiro no arquivo de problemas.

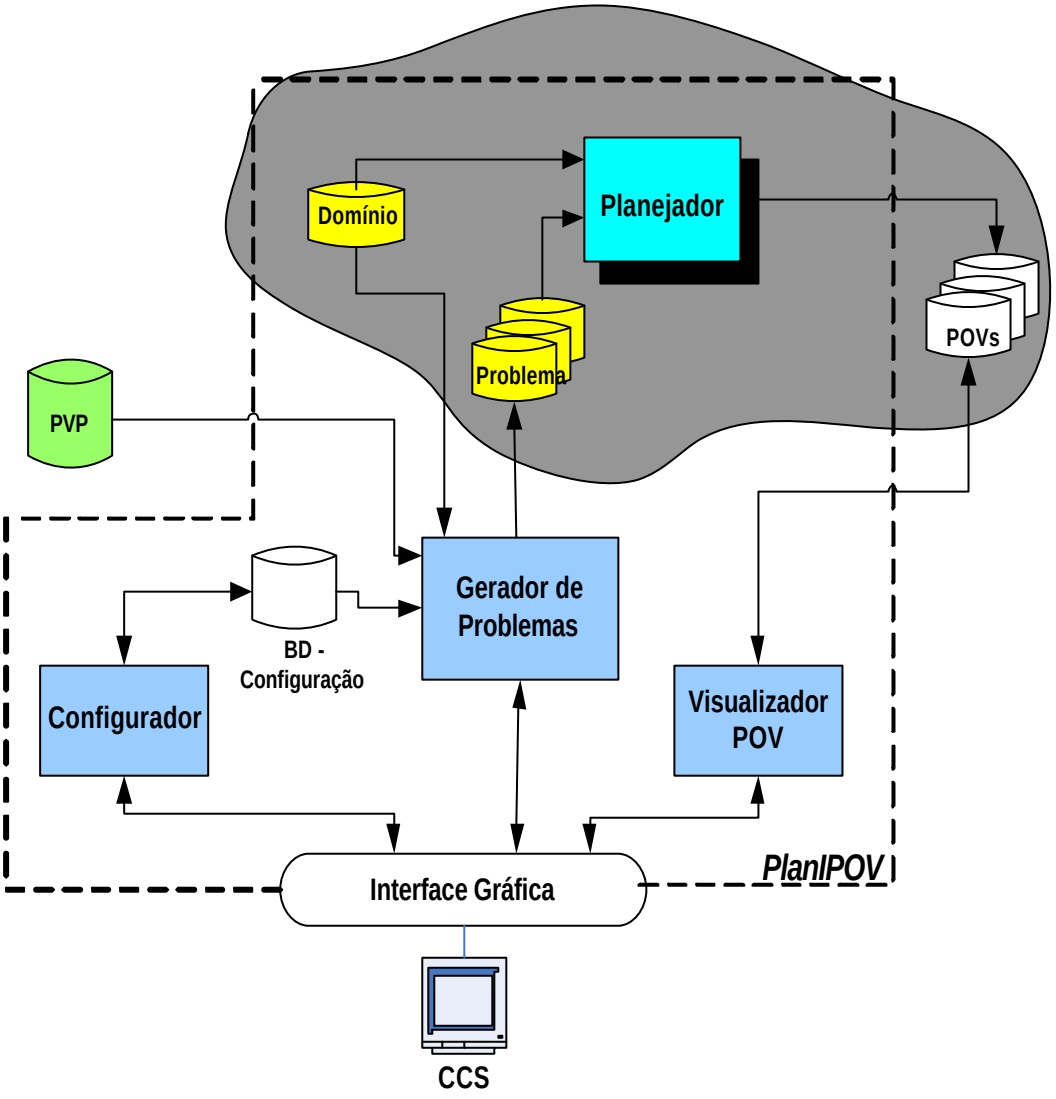

FIGURA 4.3 - Arquitetura do sistema PlanIPOV.

O módulo **Configurador** é responsável pela manipulação da base de dados de Configuração. Este módulo permite que o responsável pelo planejamento das operações de um satélite possa editar: (i) todos os parâmetros necessários para a configuração das ETs que podem ser usadas para o rastreio de satélites; (ii) os parâmetros dos satélites a serem rastreados, que inclui os parâmetros de telecomandos; e (iii) os relacionamentos

entre as ETs e os satélites, que contém os cenários de operação específicos de cada dupla (satélite, ET). Este módulo deve permitir que todos os parâmetros da base de dados possam ser inseridos, alterados e excluídos pelo responsável pelo planejamento das atividades de operação.

O módulo **Gerador de Problema** é responsável por gerar os arquivos de Problemas para cada passagem no formato PDDL 2.2. Este módulo necessita dos seguintes dados de entrada: o arquivo PVP, a base de dados de Configuração e o arquivo do Domínio. Por meio dos dados de entrada, este módulo é capaz de gerar de forma automática o estado inicial do ambiente no início do rastreio de cada passagem, os eventos que devem ocorrer em tempos pré-definido no POV (eventos exógenos) e o objetivo a ser atingido ao final da passagem.

O módulo **Planejador** é responsável por gerar o POV. Este módulo, a partir da base de conhecimento descrita em PDDL 2.2 (arquivo do Domínio e de Problemas), é capaz de encontrar uma seqüência de ações que atinja o objetivo proposto no arquivo de Problemas. A solução é encontrada pela aplicação de técnicas de planejamento que tratam problemas de planejamento com restrição de tempo e de recurso. O arquivo de saída do planejador é o POV para o satélite da passagem especificada no problema. Como pode ser observado na parte mais escura da Figura 4.3, que contêm o módulo Planejador, o arquivo de Domínio, os arquivos de Problemas e os POVs representa a arquitetura de um agente de planejamento sem o módulo Executor.

O módulo **Visualizador POV** é responsável por obter o arquivo POV gerado pelo módulo Planejador e deixar o seu conteúdo disponível para visualização pelo operador. Este módulo permite também que o conteúdo do arquivo POV seja alterado pelo operador antes que o mesmo possa ser executado.

O operador responsável pelo planejamento das operações dos satélites, por meio do módulo **Interface Gráfica** disponibilizado pela aplicação, tem a opção de executar três tipos de serviço: configuração da base de dados de Configuração, geração do POV e visualização do POV gerado.

Se escolhido o serviço de configuração da base de dados de Configuração, o operador poderá atualizar os parâmetros contidos na base de dados tendo como opções as funções de edição, alteração e exclusão de parâmetros.

Se for escolhido o serviço de geração do POV, o operador deverá escolher qual arquivo PVP deve ser entrada para a geração do POV. De posse do arquivo PVP escolhido, todas as passagens existentes neste arquivo devem ser extraídas e mostradas ao operador. O operador terá a opção de selecionar uma, algumas ou todas as passagens para que sejam gerados os respectivos POVs. Isto dá flexibilidade para que a base de dados de Configuração seja modificada de forma que um cenário específico de operação seja aplicado em uma ou mais passagens determinadas. O módulo Gerador de Problema de posse do arquivo PVP e das passagens selecionadas, executa a geração automática dos arquivos de problemas. Em seguida, o módulo Planejador é automaticamente ativado para gerar os POVs utilizando os arquivos de problemas recém criados e o arquivo de Domínio. No caso em que o planejador não encontre uma solução possível para o arquivo de problema criado, uma mensagem do ocorrido será apresentada ao operador.

A opção de visualizar os POVs gerados estará disponível após a geração dos mesmos. O operador responsável pelo planejamento terá a opção de escolher um dos planos gerados para ser visualizado. O conteúdo do POV escolhido será mostrado na tela e o usuário terá a opção de imprimí-lo e, se necessário, poderá modificar manualmente algumas ações do plano gerado

O sistema PlanIPOV como já foi salientado tem um agente de planejamento incorporado em sua arquitetura geral. Desta forma, o seu desenvolvimento passa por duas etapas bem definidas: (i) modelagem da base de conhecimento do domínio de rastreio de satélite que pertence a engenharia do conhecimento (ver item 3.3) e (ii) modelagem do sistema que incorpora a arquitetura do agente de planejamento e tem como objetivo gerar e visualizar POVs fazendo uso da base de conhecimento gerada (engenharia de software).

No restante deste Capítulo, serão apresentados: (i) um processo para a modelagem de uma base de conhecimento para um domínio de planejamento usando a linguagem PDDL (item 4.3) e (ii) um *framework* para se desenvolver um sistema de planejamento que faz uso da linguagem PDDL e que tem como objetivo gerar os arquivos de problemas de forma automática (item 4.4).

4.3 Processo para Modelagem do Domínio

Um dos desafios do trabalho é definir de forma correta a representação do conhecimento do domínio de planejamento, pois a partir desta definição é possível mapear corretamente os dados de entrada necessários para a geração automática do arquivo de problema.

Nesta seção serão apresentados os passos para modelar uma base de conhecimento para um domínio de planejamento usando o formato PDDL. Este processo foi baseado no processo geral de construção de base de conhecimento fornecido por Russel e Norvig (vide item 3.3). O processo proposto para modelagem de uma base de conhecimento em PDDL para problema de planejamento inclui as seguintes etapas:

1) Especificação clara do ambiente de tarefa do agente;

2) Aquisição de conhecimento do domínio de planejamento;

3) Definição da versão da linguagem PDDL a ser usada e do agente de planejamento;

4) Especificação da ontologia do domínio;

5) Codificação do arquivo de domínio na linguagem PDDL;

6) Validação do domínio (parcial e global).

Os sub-itens a seguir descrevem cada etapa do processo. E cada etapa descrita será exemplificada pelo resultado obtido da aplicação da mesma no domínio de rastreio de satélites que é o escopo do trabalho.

4.3.1 Especificação clara do ambiente de tarefa do agente

A primeira etapa é especificar claramente o ambiente de tarefa do agente a ser desenvolvido levantando o problema para o qual o agente deve ser a solução. Em seguida, deve-se categorizar o ambiente de tarefa obtido de forma a determinar o projeto apropriado do agente (ver item 3.1).

No caso do domínio de rastreio de satélites, os elementos levantados do ambiente de tarefa foram:

- **Medida de desempenho**: gerar planos de vôo corretos para um determinado satélite levando em consideração uma ou mais janelas de visibilidade (passagens) de uma ET;

- **Ambiente externo**: janela de comunicação do satélite com a ET;

- **Atuadores**: exibir os planos de operação de vôos gerados contendo as data/horas inicial e final da passagem e todas as ações do plano acompanhadas das data/horas de execução das mesmas;

- **Sensores**: Plano de previsão de passagem pertencente ao satélite e à ET desejada, configuração do satélite e configuração da ET.

Analisando o ambiente de tarefa definido, verificou-se que o mesmo pode ser categorizado como:

- **Completamente observável**, ou seja, os planos de previsão de passagem e as configurações do satélite e da ET permitem um acesso ao estado completo do problema do rastreio de satélite a ser solucionado pelo agente;

- **Determinístico**, ou seja, o próximo estado do ambiente de rastreio de satélite pode ser determinado pelo estado atual e pela ação a ser executada pelo agente. Não se está levando em consideração no problema a ser modelado as situações de contingências que podem ocorrer durante o rastreio do satélite. Se fossem

tratadas estas condições, o problema passaria de um planejamento clássico para um planejamento não-clássico;

- **Seqüencial,** ou seja, cada ação inserida no plano de operação de vôo afeta a decisão de inserção da próxima ação no plano;

- **Dinâmico,** ou seja, o ambiente de rastreio de uma passagem de satélite está em constante mudança. Muitas ações a serem inseridas no plano dependem do ângulo de apontamento da antena da ET que está rastreando o satélite. Este ângulo vai se alterando durante todo o rastreio;

- **Contínuo**, ou seja, cada passagem do satélite a ser rastreada tem um período de visibilidade e as ações a serem inseridas no plano também têm uma duração de execução. O plano a ser gerado deve inserir todas as ações necessárias ao rastreio da passagem dentro do período de visibilidade. Esta característica do ambiente de rastreio impõe restrição de tempo no problema de planejamento. A execução do plano durante a passagem do satélite é contínua. Vale a pena ressaltar que o domínio de rastreio de satélite também tem restrição de recurso como, por exemplo, uma ET só pode rastrear um satélite de cada vez e nenhuma ação pode ser executada entre a ET e o satélite durante o período de zona de silêncio;

- **Agente único,** ou seja, é necessário apenas um agente para resolver o problema de geração de plano de operação de vôo.

4.3.2 Aquisição de conhecimento do domínio de planejamento

Com as características gerais definidas do agente, inicia-se a etapa de aquisição de conhecimento junto aos especialistas no domínio que se deseja modelar. Nesta etapa é importante extrair do especialista e registrar todas as informações que permitam entender como o domínio realmente funciona. As seguintes informações devem ser levantadas para que o domínio de planejamento possa ser compreendido e posteriormente mapeado:

- Todas as ações que podem existir no plano;
- Em que condições tais ações podem ocorrer ou não;
- Quais as ações são temporizadas;
- Quais as ações que têm duração;
- Quais as restrições de tempo e de recurso apresentados no ambiente;
- Entidades do mundo envolvidas no ambiente;
- Parâmetros que são sempre fixos no ambiente;
- Parâmetros que se modificam durante a execução do plano;
- Procedimentos operacionais padrões se existirem;
- Restrições de operação;
- Objetivos que se deseja atingir com a execução do plano.

4.3.3 Definição da versão da linguagem PDDL a ser usada e do agente de planejamento

As duas primeiras etapas permitem definir o tipo de problema de planejamento que se deseja solucionar. Desta forma, é possível definir qual versão da linguagem PDDL deve ser utilizada na geração da base de conhecimento dependendo das características necessárias para modelar o ambiente (vide item 3.2.2.1) e qual o agente de planejamento (planejador) apropriado para o problema a ser solucionado. Neste momento, pode-se decidir por desenvolver um agente com as características levantadas utilizando técnicas de planejamento que mais se adaptam ao problema, ou fazer uso de um agente de planejamento disponível que é capaz de resolver o problema desejado que utilize a versão PDDL escolhida.

No estudo de caso para a geração de POVs, o ambiente de tarefa do agente de planejamento caracteriza um ambiente de planejamento clássico com restrição de tempo e recurso. Com base na definição do ambiente de tarefa e no entendimento do problema de planejamento do POV, foi possível definir que o agente tem como característica: (i) primeiro planejar para depois executar o plano e (ii) que o tipo do planejamento é temporal com restrições de tempo e recursos.

Entendido como o domínio de geração de POVs funciona, foi verificado que a versão 2.2 da linguagem PDDL era a que melhor atendia às características do planejamento necessárias para a geração do POV, pois esta versão é capaz de representar eventos exógenos incondicionais determinísticos que existem no problema de planejamento em estudo (como exemplo, janela de visibilidade). Como conseqüência da escolha da linguagem PDDL 2.2, o planejador LPG-TD foi selecionado por ter a capacidade de interpretar esta versão da linguagem e por seu código executável estar disponível para ser utilizado na época da construção do protótipo (ver item 3.2.3). O código do planejador TilSAPA que também seria capaz de tratar as características do domínio do planejamento proposto não estava disponível no início da implementação do protótipo (Kavuluri e Khemani, 2004).

4.3.4 Especificação da ontologia do domínio

Após a definição do agente de planejamento e da linguagem a ser utilizada, a próxima etapa é estabelecer uma ontologia do domínio onde é definido um vocabulário para o domínio sendo modelado. A definição desse vocabulário deve ser feita já orientada para os elementos existentes na estrutura da linguagem PDDL (McDermontt et al., 1998, Fox; Long, 2003, Edelkamp, Hoffmann, 2004) que inclui a definição de todos os tipos, ações, funções e predicados existentes no domínio.

Em relação aos predicados, após a definição, estes devem ser classificados em uma das quatro categorias abaixo para possibilitar a geração automática dos arquivos de problemas:

- **Estados iniciais do ambiente**, ou seja, aqueles predicados que informam o estado do ambiente para iniciar a preparação do plano. Como exemplo pode-se citar alguns predicados definidos que informam a situação inicial do ambiente da passagem a ser rastreada: ET disponível para rastreio (GSAvailable), passagem deve ser rastreada (PassTrack) e medida de distância deve ser executada para a passagem (RangeAvailable);

- **Estados intermediários**, ou seja, aqueles estados que ocorrem durante o planejamento, mas que os seus predicados não são mapeados no arquivo de problemas. Para exemplificar pode-se citar o predicado de calibração executada (CalibrationOK), que é um estado atingido após a execução da ação calibrar (Calibrate) e que é um estado necessário para a ação Executar Medida de Distância (Range);

- **Eventos exógenos**, ou seja, aqueles eventos que passam a ser verdadeiros ou falsos em um ponto de tempo determinado independentemente das ações que o planejador escolhe para executar. Este tipo foi bastante utilizado na modelagem do domínio de rastreio de satélite. O evento que informa a ocorrência de zona de silêncio durante o rastreio de uma passagem (SatSilentZoneAvailable) é um exemplo típico deste tipo de evento. Ou seja, este evento informa a janela de tempo em que ocorre uma zona de silêncio onde nenhuma ação de comunicação entre o satélite e a estação terrena pode ser executada no período.

- **Estados objetivos,** ou seja, aqueles predicados que podem fazer parte do estado objetivo informando o que se quer atingir com o plano a ser gerado. O objetivo do plano pode ser composto de sub-objetivos, isto é, muitas vezes não é possível estabelecer um único predicado objetivo para a geração do plano e o objetivo passa a ser uma conjunção de predicados. Alguns exemplos do estudo de caso são: medida de distância executada (RangeOK) e passagem eliminada (PassEliminated).

Nesta etapa, a ontologia do domínio de rastreio para a família de satélites estudada foi definida. Alguns tipos definidos para o domínio foram: Código da Passagem, Código do Satélite, Código da Estação Terrena, Código do Telecomando e Identificação da Zona de Silêncio. Foram também definidas as funções para o cálculo de duração das seguintes ações: calibração, transmissão de telecomando, execução de medidas de velocidade, execução de medidas de distância e zona de silêncio. Alguns predicados definidos para o domínio já foram citados para exemplificar a classificação de predicados.

4.3.5 Codificação do arquivo de domínio na linguagem PDDL

Depois da ontologia definida, inicia-se a etapa de codificação do arquivo de domínio na linguagem PDDL selecionada. Nesta etapa, todos os termos definidos na ontologia do domínio são escritos de acordo com a estrutura da linguagem. A codificação de cada ação faz uso dos tipos, funções e predicados definidos. Para cada ação, deve-se definir: os tipos utilizados do domínio (*parameters*), os predicados de pré-condição para a inserção da ação no plano (*condition*) e os predicados de efeito que informam o que a ação gera ao ser executada (*effect*). Se a ação tiver uma duração, o tempo necessário para a execução da mesma (*duration*) deve ser definido como um valor fixo ou o valor variável que é obtido por uma função. O valor retornado pela função deve ser definido no arquivo de problemas. A seguir são apresentados três exemplos de ações codificadas em PDDL 2.2 para o domínio de rastreio de satélite:

1. Ação **Calibrate** (ou seja, executar calibração antes da passagem)

```
(:durative-action Calibrate
   :parameters (?pass - PassNumber ?sat - SatCode ?gs - GSCode
                ?cgroup - CalibGroupNumber)
   :duration (= ?duration (CalibrationTime ?gs ?cgroup))
   :condition ( and  (at start (GSAvailable  ?gs))
                     (at start (CalibrationAvailable ?pass  ?gs))
                     (over all (PassTrack ?pass  ?sat ?gs))
                     (over all (RangeAvailable ?pass  ?sat  ?gs))
               )
   :effect (and  (at end (CalibrationOK ?pass ?cgroup))
                 (at start (not (GSAvailable ?gs)))
                 (at end (GSAvailable ?gs))
                 (at end (not (CalibrationAvailable ?pass  ?gs)))
            )
```

```
)
```

2. Ação **WaitSilentZone** (ou seja, esperar por Zona de Silêncio)

```
 (:durative-action WaitSilentZone
   :parameters (?pass - PassNumber ?sat - SatCode ?gs - GSCode
                ?ZS - ZSNumber )
   :duration (= ?duration (ZoneSilentTime ?pass ?ZS))
   :condition ( and (at start (GSAvailable  ?gs))
               (at start (SatSilentZoneAvailable ?pass ?sat ?gs ?ZS)))
   :effect (and  (at end (SilentZone ?pass ?sat ?gs ?ZS))
                 (at start (not (GSAvailable ?gs)))
                 (at end (GSAvailable ?gs)))
)
```

3. Ação **SendFirstTC** (ou seja, enviar primeiro telecomando da passagem)

```
(:durative-action SendFirstTC
   :parameters (?pass - PassNumber ?sat - SatCode ?tc - TCCode
                ?gs - GSCode ?cgroup - CalibGroupNumber)
   :duration (= ?duration (TCTime ?tc))
   :condition ( and (at start (GSAvailable  ?gs))
                (at start (FirstTc ?pass  ?sat  ?tc))
                (over all (PassTrack ?pass  ?sat ?gs))
                (over all (not (TCNotAvailable ?pass ?sat)))
                (at start (or (CalibrationOK ?pass ?cgroup)
                          (NotCalibrationAvailable ?pass  ?gs)))
                )
   :effect (and  (at end (FirstTcOK ?pass ?sat ?tc))
                 (at end (TcSent ?pass ?sat))
                 (at start (not (GSAvailable ?gs)))
                 (at end (GSAvailable ?gs))
            )
)
```

Nos exemplos apresentados pode ser observado que para todas as ações foram definidas as pré-condições (*condition*) para que a ação possa ser inserida no plano, o tempo de execução (*duration*) e os efeitos (*effect*) que estas ações produzem ao serem executadas.

4.3.6 Validação do domínio

A etapa de validação do domínio deve ocorrer em paralelo com a codificação do arquivo de domínio. A modelagem do domínio deve ser feita de forma evolutiva tendo como base a ontologia definida. No arquivo de domínio, as ações são codificadas uma a uma e a cada ação codificada, o domínio como um todo deve ser validado. A codificação de vários arquivos de problemas (vide item 3.2.2.1) se faz necessária para

permitir o teste da validade ou não do domínio em desenvolvimento. Na validação deve ser utilizado o planejador escolhido para a geração do plano. Os arquivos de problemas de testes devem ser elaborados de forma a conter o ambiente inicial e o objetivo a ser atingido no plano que são necessários para exercitar as ações que estão sendo inseridas no arquivo de domínio. Após a modelagem de todo o domínio, a base de conhecimento deve ser testada a fim de verificar se o seu comportamento está conforme o esperado. Se problemas forem encontrados, deve-se retornar às etapas anteriores para correção, repetindo-se todo o processo se necessário.

A Figura 4.4 mostra a arquitetura do agente de planejamento utilizada nos testes durante a etapa de codificação e validação da base de conhecimento do domínio de rastreio de satélites.

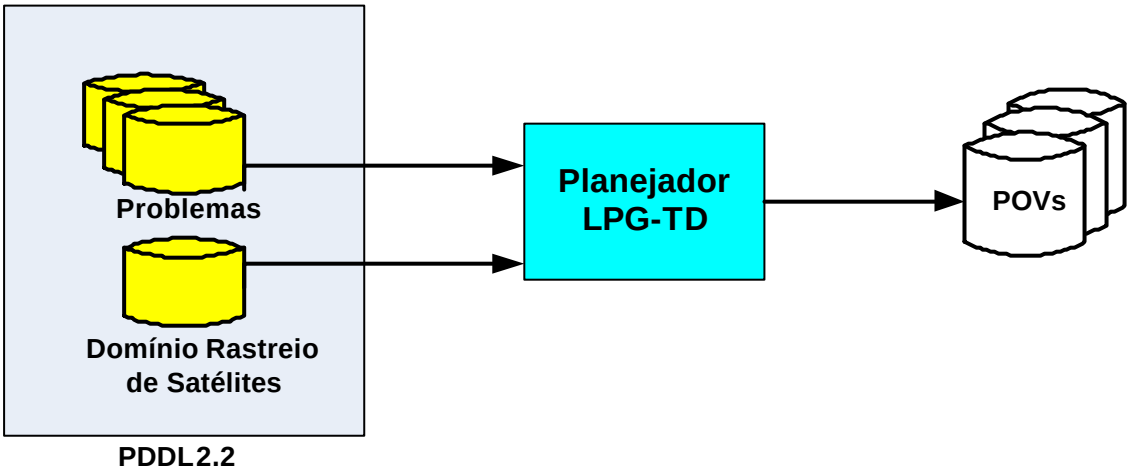

FIGURA 4.4 – Arquitetura de teste do domínio de rastreio de satélite.

Para cada ação codificada no arquivo de domínio, um ou vários arquivos de problemas foram codificados para exercitar a base de conhecimento. Para cada arquivo de problema criado para o teste, o planejador LPG-TD era executado e os resultados gerados eram analisados. Quando erros de codificação em PDDL eram encontrados, a estrutura da linguagem era analisada e eram feitas as correções necessárias no arquivo de domínio e/ou no arquivo de problema. Quando o POV era gerado pelo planejador, o conteúdo do mesmo era analisado. No caso dos resultados não serem os esperados, ajustes na ontologia e na definição das condições e dos efeitos das ações foram feitos

até que os planos fossem gerados conforme o desejado. O arquivo de domínio modelado para o domínio de rastreio de satélite encontra-se no Apêndice A.

Muitas vezes, dependendo do problema encontrado durante a validação da base de conhecimento, se faz necessário repetir uma ou várias etapas para se obter o plano desejado.

É importante ressaltar que durante todo este processo, os especialistas do domínio devem ser consultados várias vezes ajudando no entendimento do problema de forma a possibilitar a geração correta dos planos, pois são os mesmos que retêm o conhecimento do domínio sendo modelado.

4.4 *Framework* para Geração de Planos Usando a Linguagem PDDL

A criação dos arquivos de problemas se apresenta como uma grande dificuldade na utilização de agentes de planejamento para a geração de planos em um ambiente onde o estado inicial e o objetivo mudam a cada plano a ser gerado. A geração do arquivo de problemas em PDDL de forma manual para problemas reais é muito trabalhosa. Na verdade, sem a automatização na geração desses arquivos de problemas, o trabalho manual do responsável pela geração dos planos seria transferido da fase de geração dos planos para a fase de preparação dos mesmos e assim, não se teria nenhuma vantagem em aplicar esta técnica de planejamento em aplicações do mundo real.

Este item apresenta um *framework* de classes para solucionar esta dificuldade. Este *framework* pode ser usado por qualquer sistema que tem como objetivo gerar de forma automática os arquivos de problemas na linguagem PDDL para a geração de planos. A Figura 4.5 mostra o diagrama de classes que compõem o *framework* proposto.

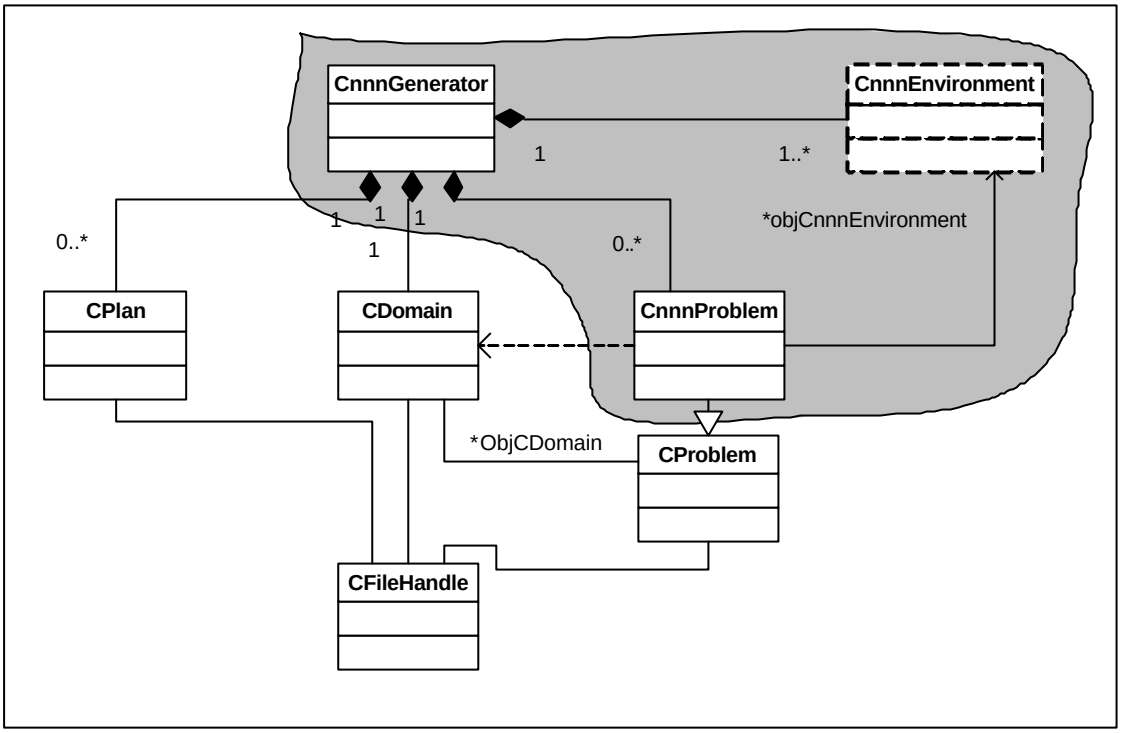

FIGURA 4.5 – Diagrama de Classes.

Como pode ser visto no diagrama de classes, existem dois conjuntos de classes. Um conjunto, que está fora da delimitação, que pode ser aplicado a qualquer sistema que usam a estrutura da linguagem PDDL. E um outro, que está dentro da parte escura do diagrama, que precisa ser preparado de acordo com o problema de planejamento sendo modelado.

As classes exteriores à área delimitada precisam ser implementadas uma única vez e podem ser reutilizadas para qualquer domínio. Os escopos destas classes estão descritos a seguir:

- CDomain -> Responsável por ler o arquivo de domínio gerado em PDDL e extrair todas as informações do domínio que serão entradas para a geração automática do arquivo de problema. Esta classe faz um *parse* no arquivo de domínio. Como pode ser observado na Figura 4.6, os atributos da classe correspondem às informações extraídas do arquivo de domínio que serão usadas na geração do arquivo de problemas e os métodos que estão precedidos do símbolo + são os que serão utilizados para obter o conteúdo dos atributos.

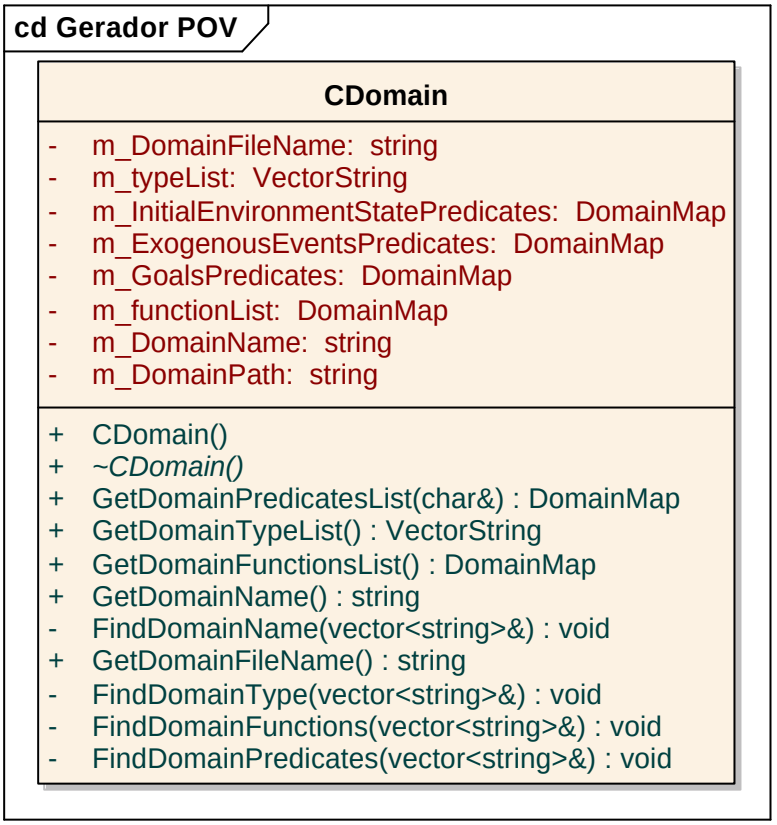

FIGURA 4.6 – Definição da classe CDomain.

- CProblem -> Responsável por gerar o arquivo de problemas em PDDL. É uma classe virtual pura que conhece a estrutura do arquivo de problemas que deve ser gerado em PDDL e que tem todos os serviços necessários para criá-lo. A Figura 4.7 apresenta a definição desta classe. Os métodos que têm a palavra *pure* na frente devem ser definidos na classe filha que irá ser responsável por tratar um problema específico. Os atributos da classe são todas as informações que devem ser escritas no arquivo de problemas. O conteúdo desses atributos é variável, ou seja, a cada novo arquivo de problema a ser criado os seus valores são gerados de acordo com o estado inicial do ambiente e do objetivo a ser atingido. Deve ser observado que esta classe contem um ponteiro para a classe CDomain, isto porque é na classe do domínio que são encontradas todas as definições necessárias para a geração do arquivo de problema. Fazendo uma analogia da estrutura definida na classe CDomain com o conceito de classes e objetos, pode-se dizer que os atributos da mesma definem classes cujos objetos são os

atributos da classe CProblem. Por exemplo, um tipo *código-satélite* definido na classe CDomain (m_TypeList) pode ser instanciado em um objeto *SCD1* na classe CProblem (m_Objects). Esta mesma idéia é aplicada aos outros atributos das duas classes.

FIGURA 4.7 – Definição da classe CProblem.

- CPlan -> Responsável por ler o arquivo de plano gerado pelo planejador e extrair todas as informações necessárias para serem apresentadas ao usuário. Esta classe fornece os métodos de acesso às informações do plano. Ela deve ser

implementada levando em conta o formato do arquivo gerado pelo planejador pois faz um *parse* nas informações contidas neste arquivo.

- CFileHandle -> Classe de uso comum que é responsável por manipular arquivos, fornecendo serviços de abertura, leitura, escrita e fechamento de arquivos. Esta classe não precisa ser implementada caso se use uma classe que implemente os serviços citados. Outra opção é incluir os serviços de manipulação de arquivos nas classes CDomain, CProblem e CPlan.

As classes CnnnGenerator, CnnnEnvironment e CnnnProblem (vide Figura 4.5) que estão dentro da delimitação são específicas ao problema de planejamento sendo considerado. Para a definição correta destas classes, é fundamental que primeiramente se tenha a base de conhecimento do domínio de planejamento definida e testada como descrito no item 4.3. Isto garante que todas as informações necessárias para modelar o ambiente tenham sido corretamente especificadas e possam ser utilizadas como fonte de informações para definição dessas classes.

A classe CnnnGenerator é responsável por gerenciar a geração dos planos de acordo com o pedido do usuário. O planejador deve ser acionado por ela depois que o arquivo de problema tenha sido criado.

A classe CnnnEnvironment é responsável por fornecer todos os dados necessários do ambiente para permitir a geração correta das informações que estarão contidas no arquivo de problemas. Esta classe manipula as informações dinâmicas do ambiente. Como pode ser visto na Figura 4.5, ela está tracejada porque pode ser desmembrada em mais classes dependendo da complexidade do ambiente de planejamento sendo tratado.

A classe CnnnProblem é responsável por preparar as informações que vão ser entrada para a geração do arquivo de problemas. Esta classe é filha da classe CProblem descrita anteriormente. Esta classe obtém as informações que são variáveis no ambiente utilizando os métodos fornecidos pela classe CnnnEnvironment, e de posse desses dados, instancia para cada atributo da classe CDomain os seus objetos, ou seja: (i) os tipos definidos no domínio são instanciados para os objetos reais do problema, (ii) os

predicados de estado inicial, eventos exógenos e objetivos definidos no domínio são instanciados se tiverem as condições desejadas no ambiente e recebem os valores necessários para sua criação, (iii) e as funções definidas no domínio também serão instanciadas de acordo com o ambiente sendo mapeado. Os métodos da classe pai CProblem que devem ser definidos estão mostrados na Figura 4.7 precedidos da palavra *pure*.

A aplicação deste *framework* de classe será exemplificada utilizando-se do domínio de rastreio de satélite. A Figura 4.8 apresenta o diagrama de classes gerado para este domínio com base no *framework* apresentado na Figura 4.5.

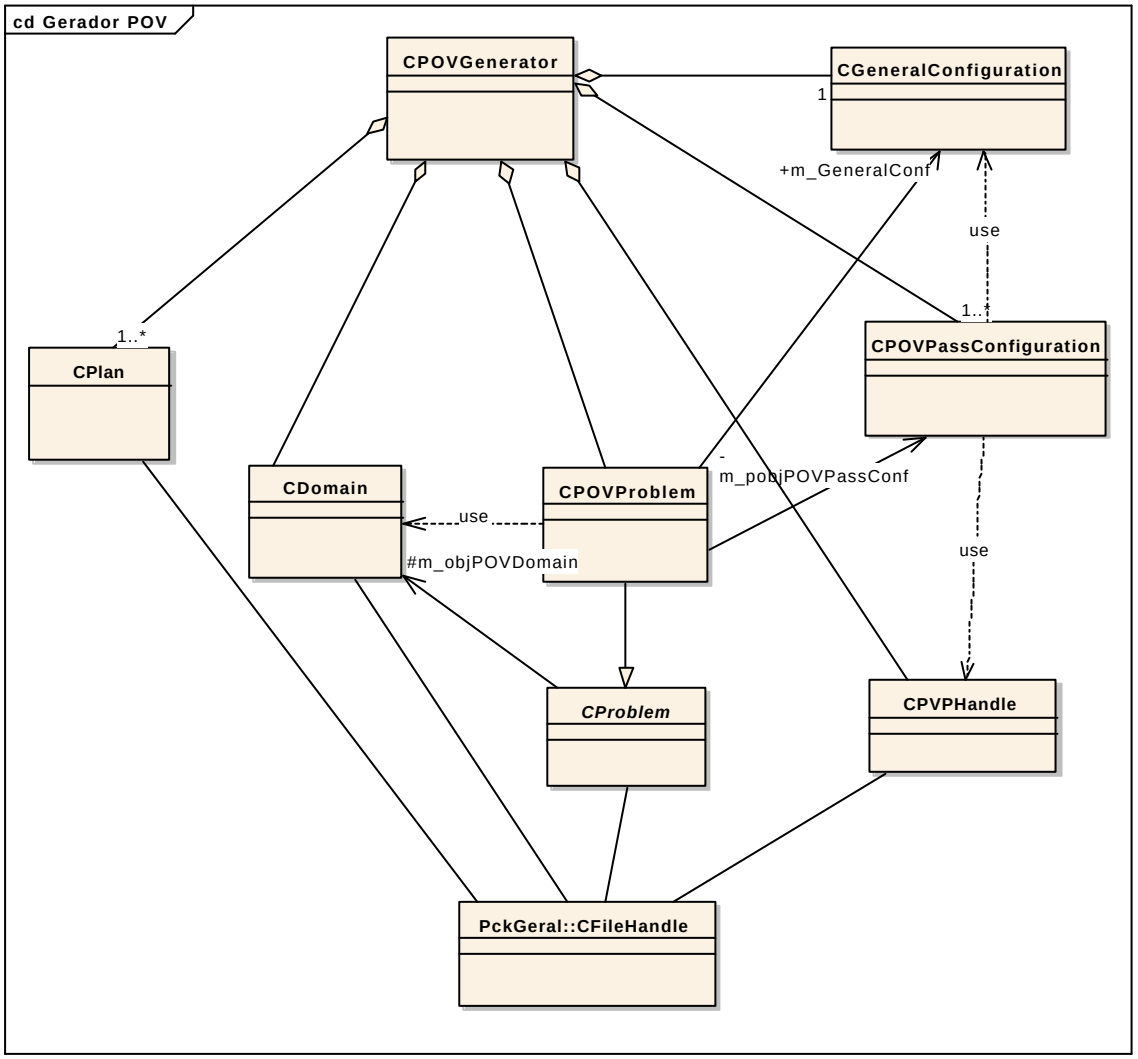

FIGURA 4.8 – Diagrama de classes do domínio de rastreio de satélite.

Comparando o diagrama de classes da Figura 4.5 e o diagrama de classes do domínio de rastreio de satélite da Figura 4.8, percebe-se que a diferença está na classe responsável por mapear o ambiente (CnnnEnvironment) que foi desmembrada em três classes (CGeneralConfiguration, CPOVPassConfiguration e CPVPHandle).

No ambiente de rastreio de satélite foram identificadas duas categorias de informações. A primeira categoria é composta das informações de configuração dos satélites e das ETs que independem das passagens a serem rastreadas, isto é são aquelas que se aplicam especificamente ao satélite e à ET envolvida no rastreio. A classe CGeneralConfiguration é responsável pelo gerenciamento destas informações.

A segunda categoria identificada corresponde às informações de configuração das passagens, ou seja, são as informações que variam de acordo com cada passagem a ser rastreada e é a partir destas informações que o estado inicial da passagem e o objetivo a ser atingido pelo plano podem ser obtidos. A classe CPOVPassConfiguration é responsável por gerenciar as informações de configuração da passagem. Esta classe necessita das informações das classes CGeneralConfiguration (descrita acima) e da CPVPHandle para encontrar a configuração de uma passagem específica. A classe CPVPHandle é responsável por ler o arquivo PVP que contém o plano de previsão de passagens e extrair dele todas as informações que auxiliam a configuração das passagens. De posse das informações da passagem para a qual se deseja gerar o plano, obtidas do arquivo PVP através da classe CPVPHandle e das configurações fixas do ambiente, fornecidas pela classe CGeneralConfiguration, a classe CPOVPassConfiguration é capaz de preparar os parâmetros de configuração da passagem de forma a permitir a geração do arquivo de problemas. A Figura 4.9 apresenta a definição da classe CPOVPassConfiguration. Os atributos desta classe identificam a configuração da passagem para qual se deseja gerar o POV.

FIGURA 4.9 – Definição da classe CPOVPassConfiguration.

A classe CPOVProblem é responsável por criar os objetos, os predicados (estado inicial, eventos exógenos e objetivo) e as funções de acordo com a configuração do ambiente da passagem a ser rastreada. Esta classe de posse das informações de tipos, predicados e funções definidos no arquivo de domínio de rastreio de satélite (classe CDomain), decide pela inserção ou não dos mesmos no arquivo de problema fazendo uso dos serviços disponibilizados pelas classes CPOVPassConfiguration e CGeneralConfiguration.

Para exemplificar como a classe CPOVProblem trabalha será mostrado como é feita a instanciação do predicado SatSilentZoneAvailable do domínio de rastreio de satélite para o arquivo de problema. Este predicado é um evento exógeno que informa o período de zona de silêncio na passagem, ou seja, o período em que a comunicação do satélite com a ET é interrompida. No arquivo de domínio, este predicado é definido da seguinte forma em PDDL 2.2:

(SatSilentZoneAvailable ?pass - PassNumber ?sat - SatCode ?gs - GSCode ?ZS - ZSNumber).

O pseudo-código mostrado a seguir descreve os passos que são dados para inserir ou não o predicado SatSilentZoneAvailable no arquivo de problemas:

Se *(predicado obtido do domínio (classe CDomain) = "SatSilentZoneAvailable")*
Então
Verificar se existe zona de silêncio para a passagem (classe CPOVPassConfiguration)
Se *existir zona de silêncio*
Então
Enquanto *existir zona de silêncio,* ***faça****:*
Obter os tipos que são utilizados pelo predicado (classe CDomain)
Obter os objetos que são instâncias destes tipos
(classes CPOVPassConfiguration e CGeneralConfiguration)
Obter hora inicial e final da zona de silêncio na passagem
(classe CPOVPassConfiguration)

Criar um código para a zona de silêncio

Inserir predicado no arquivo de problemas

Fim Enquanto

Fim Se

Fim Se

Se a passagem que está sendo configurada tivesse o código PASS60485-7 que faz parte do rastreio do satélite SCD1 pela estação terrena de CBA e que tem uma zona de silêncio entre os segundos 540 e 600 depois do início da passagem, o resultado para o exemplo seria a inserção de um evento exógeno do tipo SatSilentZoneAvailable no arquivo de problemas. As linhas de código em PDDL 2.2 geradas seriam as seguintes:

(at 540.0 (SatSilentZoneAvailable PASS60485-7 SCD1 CBA ZS01))

(at 600.0 (not (SatSilentZoneAvailable PASS60485-7 SCD1 CBA ZS01)))

Observe-se que os tipos definidos no arquivo de domínio foram instanciados para os objetos reais do ambiente (PassNumber = PASS60485-7, SatCode = SCD1, GSCode = CBA e ZSNumber = ZS01).

Se não existisse uma zona de silencio na passagem PASS60485-7, este predicado não seria inserido no arquivo de problemas.

Como pode ser observado, a classe CPOVProblem faz uso do conhecimento do ambiente da passagem que é propriedade da classe CPOVPassConfiguration para decidir o que deve ser ou não colocado no arquivo de problemas. Para cada predicado e função definido no arquivo de domínio, um código específico para tratamento do mesmo deve ser criado na classe CPOVProblem.

A Figura 4.10 apresenta um diagrama de seqüência de geração de POVs que faz uso do modelo de classes gerado para o domínio de rastreio de satélite. Esta seqüência parte do princípio que o arquivo PVP, que contém as passagens para as quais devem ser gerados os planos, já foi especificado.

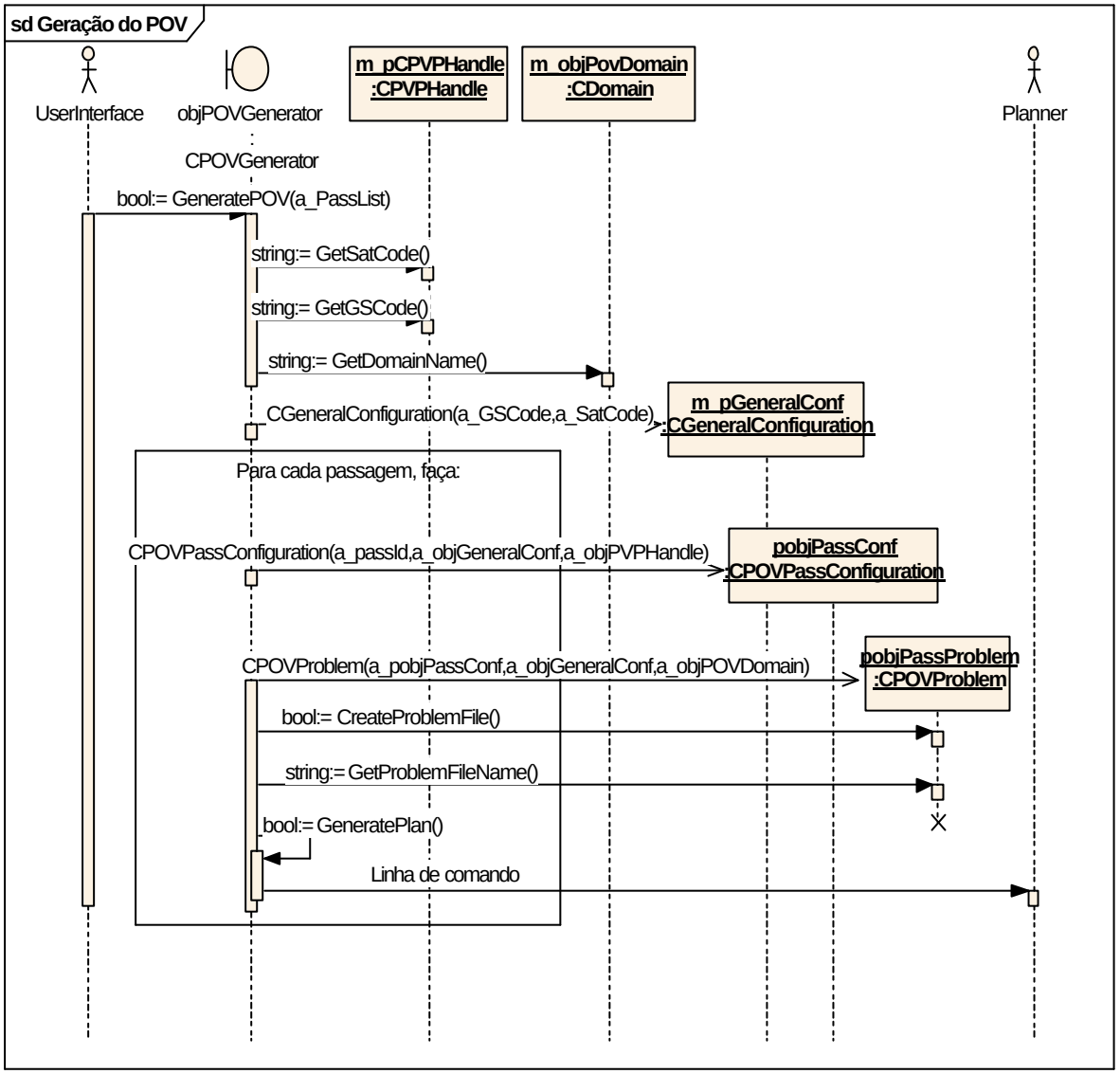

FIGURA 4.10 – Diagrama de seqüência de geração do POV.

O usuário ativa a geração de planos passando os códigos das passagens selecionadas do arquivo PVP (serviço GeneratePOV da classe CPOVGenerator). Neste momento as informações gerais para todas as passagens são obtidas (classes CPVPHandle e CDomain) e com base nas informações obtidas do arquivo PVP, um objeto da classe CGeneralConfiguration é criado para o satélite e a ET especificada no PVP. Para cada passagem selecionada pelo usuário, os seguintes passos devem ser realizados:

a) Preparação de todas as informações que configuram o ambiente da passagem a ser rastreada (objeto da classe CPOVPassConfiguration é criado);

b) Geração automática do arquivo de problemas para a passagem configurada (objeto da classe CPOVProblem é criado); e

c) Geração do POV para a passagem por meio da execução do planejador que é ativado por meio de linha de comando no código

Após a geração dos POVs, o usuário poderá visualizar os planos gerados fazendo uso do serviço GetGeneratedPlan disponibilizado pela classe CPOVGenerator que por sua vez utiliza o serviço da classe CPlan para obter o conteúdo do POV desejado.

CAPÍTULO 5

IMPLEMENTAÇÃO DE UM PROTÓTIPO DA ARQUITETURA

Neste Capítulo será apresentado o protótipo que foi implementado baseado na arquitetura do sistema PlanIPOV, descrita no item 4.2, e no *framework* proposto para a geração de planos utilizando a linguagem PDDL, descrito no item 4.4.

A implementação do protótipo do sistema PlanIPOV (ver Figura 4.3) utilizou as seguintes ferramentas e linguagens no desenvolvimento:

- Gerenciador de base de dados Microsoft Access2003;
- Microsoft Visual Studio 6.0;
- Linguagem C++;
- Planejador LPG-TD-1.0.exe que suporta a linguagem PDDL 2.2.

A Tabela 5.1 mostra a correspondência entre os artefatos implementados/usados no protótipo e os módulos propostos no sistema PlanIPOV, bem como, a ferramenta e linguagem de software utilizada no desenvolvimento de cada artefato.

TABELA 5.1 – Artefatos do protótipo x módulos do PlanIPOV.

Artefato	**Módulo do PlanIPOV**	**Ferramenta/linguagem**
POVGenerator.exe	Interface Gráfica Gerador de Problemas Visualizador POV	Microsoft Visual Studio 6.0 Linguagem C++
LPG-TD-1.0.exe	Planejador	Linguagem C
POVConf.mdb	Configurador	Microsoft Access 2003

A Figura 5.1 apresenta o diagrama de componente de execução do programa executável POVGenerator gerado para o protótipo.

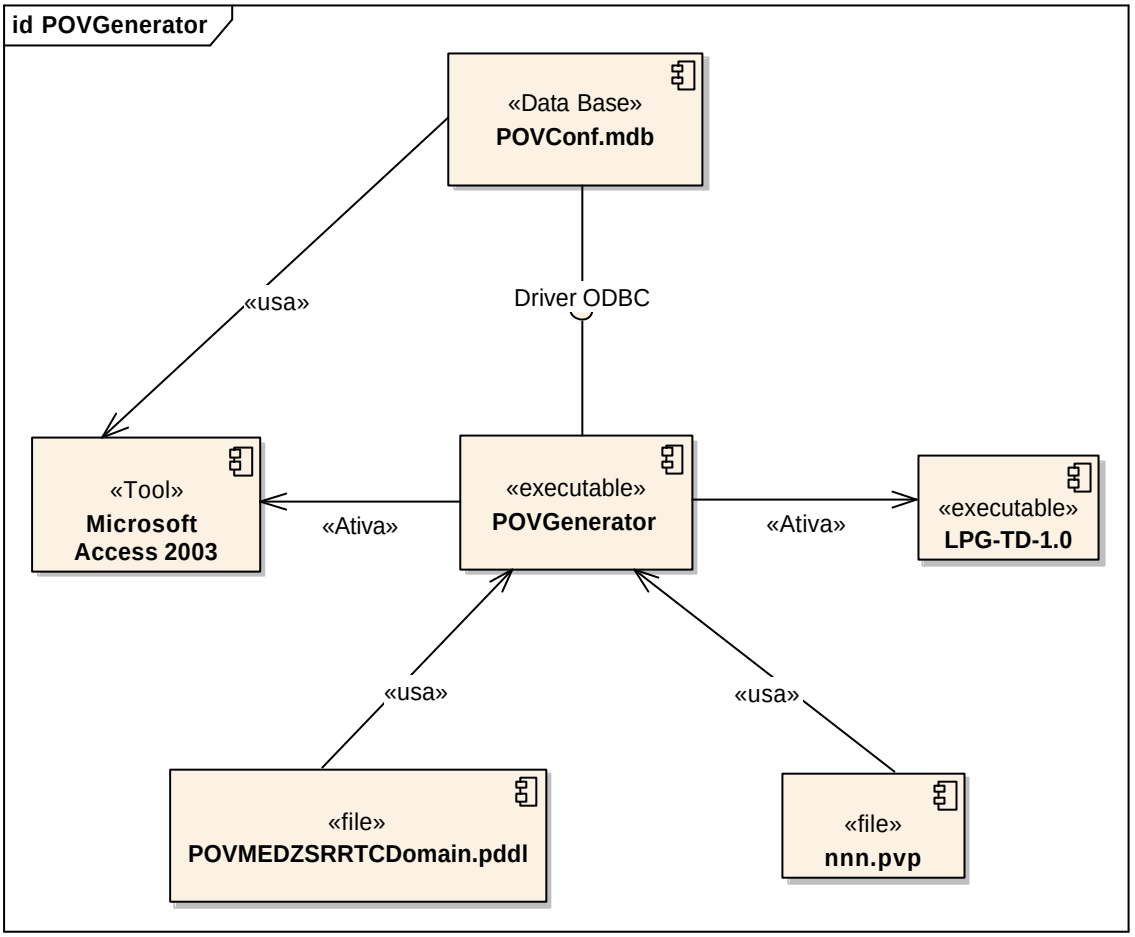

FIGURA 5.1 – Diagrama de componente de execução do POVGenerator.

Este executável utiliza os seguintes arquivos de entrada:

- Arquivo nnn.pvp, que contem o Plano de Previsão de Passagem para o qual se deseja gerar os POVs (nnn, é o nome do arquivo PVP gerado pelo CCS);
- Arquivo POVMEDZSRRTCDomain.pddl, que é o arquivo que contém o domínio de rastreio de satélite codificado em PDDL 2.2.

Para a geração dos POVs, o POVGenerator ativa o programa executável LPG-TD-1.0 por meio de linha de comando.

Para obter os parâmetros de configuração, o POVGenerator faz uso do *driver* ODBC para acessar a base de dados de configuração POVConf.mdb. Por meio da interface gráfica fornecida pelo POVGenerator, é possível ativar o Microsoft Access 2003 que por sua vez ativa a aplicação responsável pela edição da base de dados POVConf.mdb.

Como pode ser observado no diagrama de componente de construção do programa executável POVGenerator apresentado na Figura 5.2, as classes desenvolvidas para o protótipo são agrupadas em quatro pacotes de classes, sendo: PckGeradorPOV, PckUserInterface, PckGeral e PCKAcessoBDados.

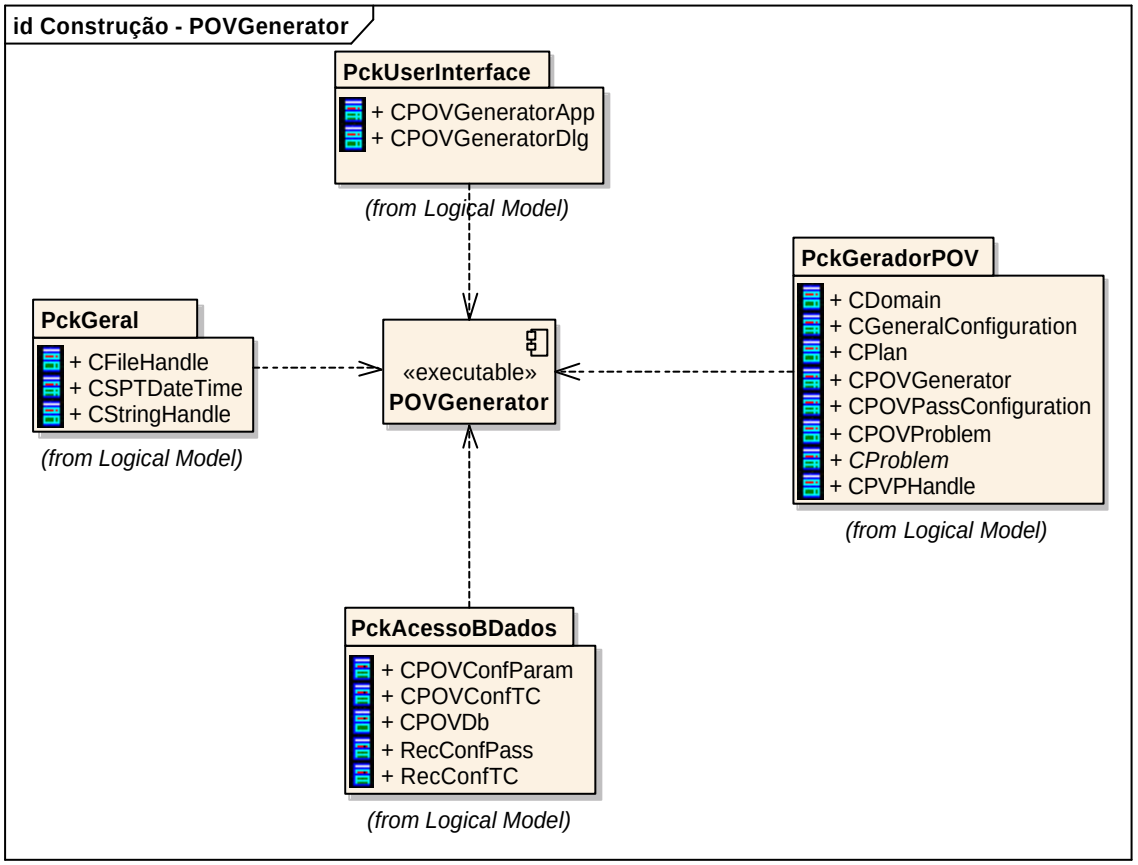

FIGURA 5.2 – Diagrama de componente de construção do POVGenerator.

O pacote de classes PCKGeradorPOV foi baseado no *framework* para Geração de Planos (vide item 4.4). As classes contidas neste pacote tratam especificamente do domínio do problema, ou seja, a geração do POV para os satélites do INPE. O diagrama de classes deste pacote foi apresentado na Figura 4.8.

Os outros três pacotes de classes são de suporte, sendo:

a) PCKAcessoBDados - pacote de classe de acesso às tabelas da base de dados de configuração POVConf.mdb;

b) PckGeral - pacote de classes responsáveis por manusear data e tempo, arquivos e *string*; e

c) PckUserInterface - pacote de classes de interface com o usuário.

A interface gráfica desenvolvida para interação com o usuário está ilustrada na Figura 5.3. Por meio da tela principal, o usuário tem a opção de editar a base de dados de configuração, selecionando o botão *Configuração*, ou a opção de geração de POVs, selecionando o botão *Escolher PVP*.

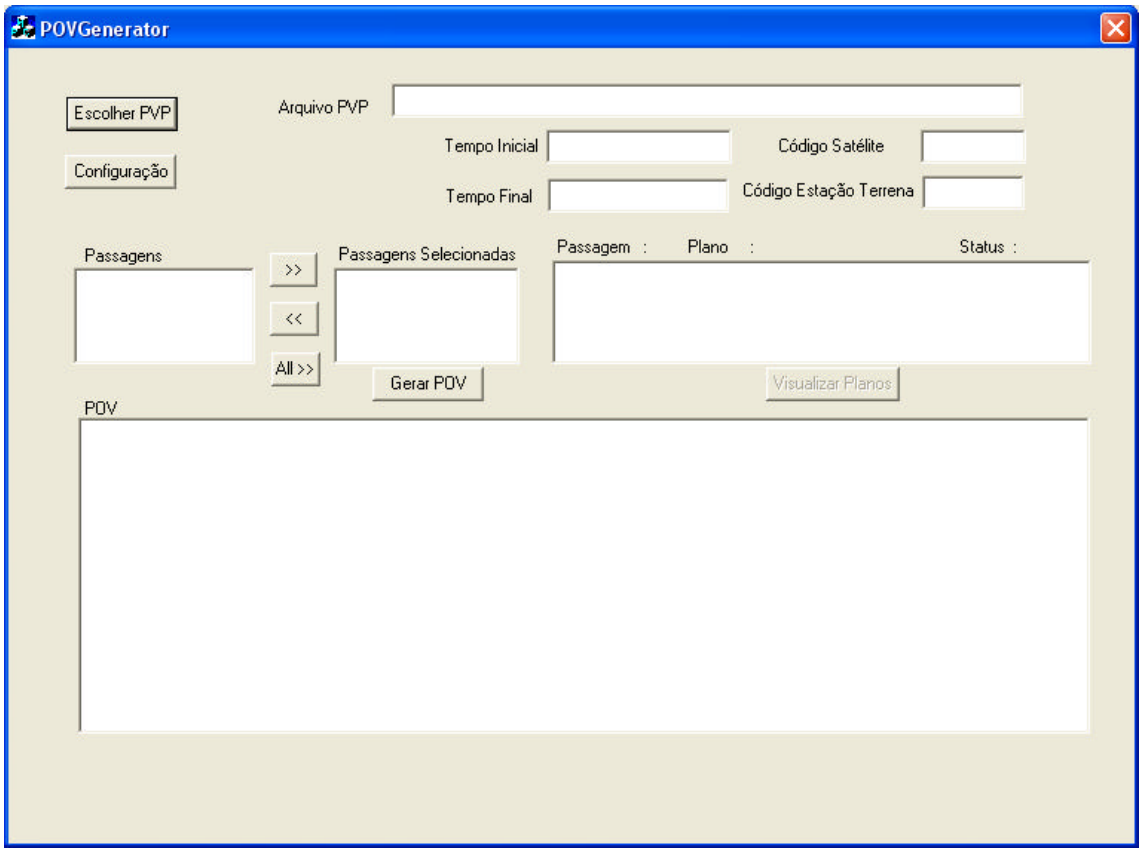

FIGURA 5.3 – Tela principal do protótipo.

Os sub-itens 5.1 e 5.2 a seguir descrevem a opção de configuração da base de dados e a opção de geração de POVs, respectivamente.

5.1 Opção de Configuração da Base de Dados

Se o botão *Configuração* for selecionado, o programa POVGenerator ativará a ferramenta Microsoft Access 2003 que ativa o editor da base de dados POVConf.mdb. A tela principal do editor está apresentada na Figura 5.4. O usuário poderá editar, alterar e/ou excluir os seguintes tipos de parâmetros contidos na base de dados: configuração de estações terrenas, configuração de satélites, configuração de parâmetros que se aplicam à passagem de um satélite sobre uma determinada estação terrena e configuração dos telecomandos de cada satélite.

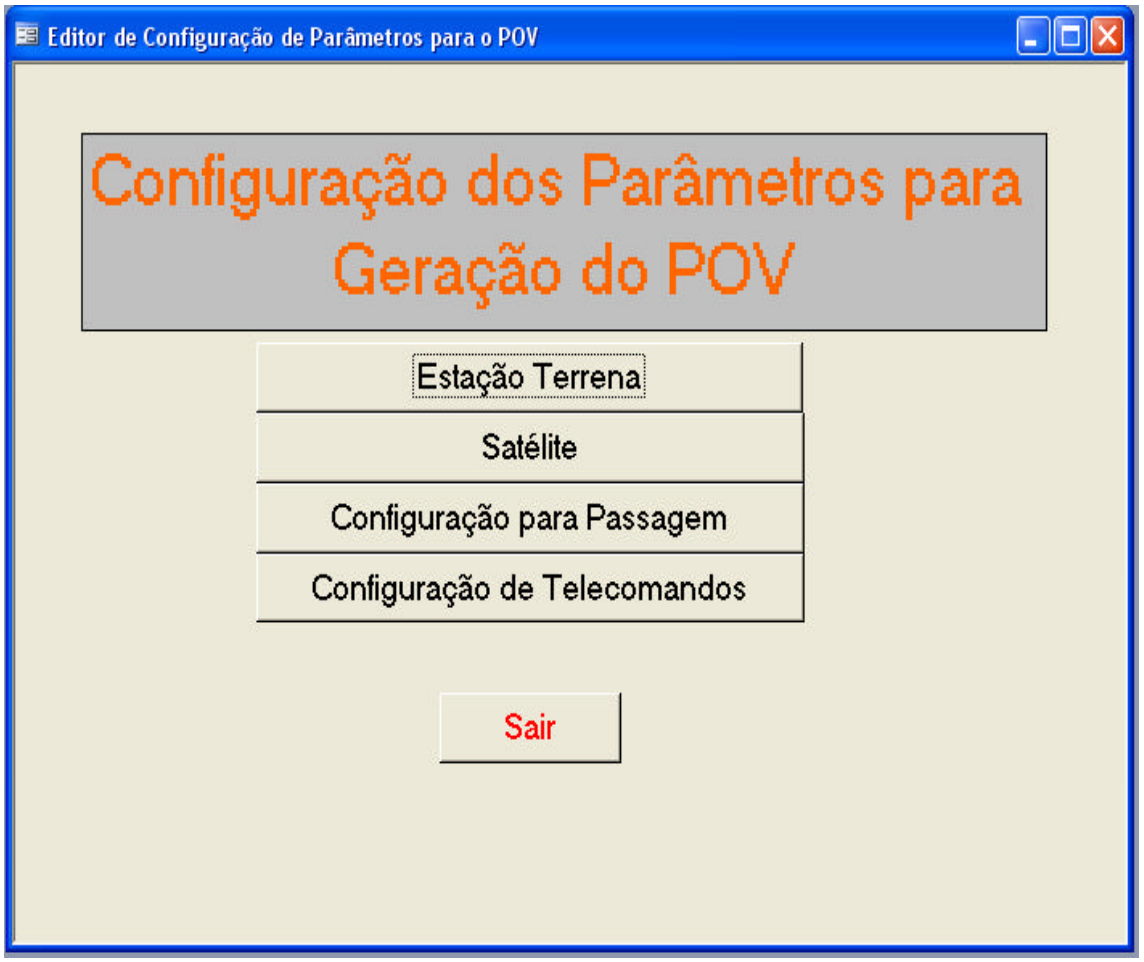

FIGURA 5.4 – Tela do Editor da base de dados de Configuração.

A Figura 5.5 ilustra a tela que permite a configuração de parâmetros relacionados à passagem. O usuário poderá atualizar a base de dados de configuração de acordo com o cenário específico do PVP ou da passagem a ser rastreada. Uma vez que os parâmetros de configuração tenham sido definidos, a mudança nos seus valores não é freqüente, há não ser que um novo cenário operacional seja definido.

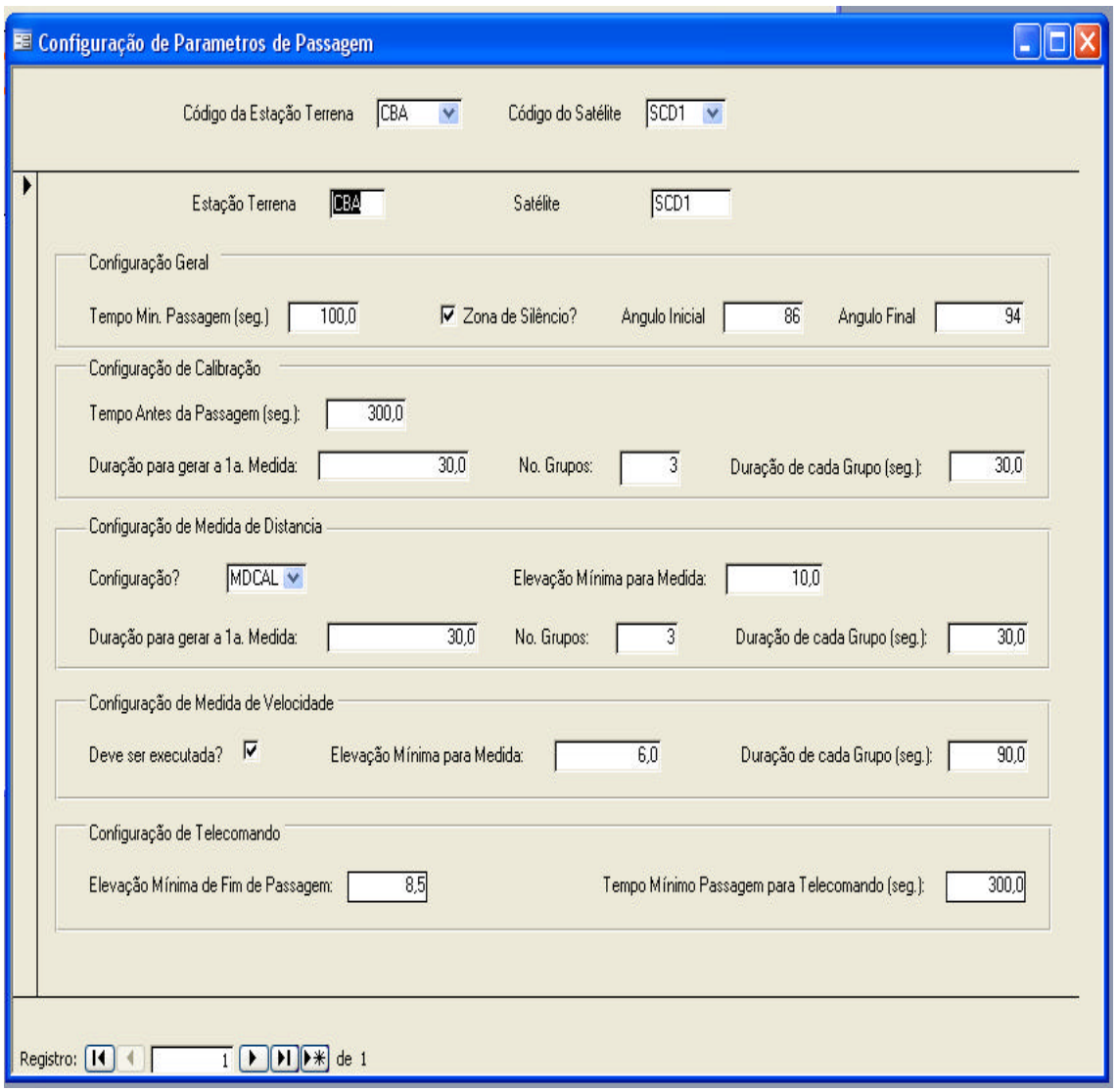

FIGURA 5.5 – Tela de Configuração de Parâmetros de Passagem.

5.2 Opção de Geração de POVs

Se o usuário desejar gerar POVs, primeiramente o botão *Escolher PVP* deve ser selecionado. Após a seleção deste botão, um diálogo será apresentado ao usuário com a relação dos nomes de todos os arquivos PVP disponíveis. O usuário deverá escolher o arquivo PVP desejado.

Escolhido o arquivo PVP, o sistema extrai as informações contidas no mesmo e as apresenta na tela. A Figura 5.6 mostra a tela atualizada com as informações extraídas do arquivo scd1_cba_60480.pvp, onde observa-se os campos data/hora da primeira passagem (*Tempo Inicial*) e da última passagem (*Tempo Final*) da seqüência de passagens fornecidas, código do satélite, código da estação terrena e identificação de todas as passagens existentes na seqüência de passagens fornecida (*Passagens*).

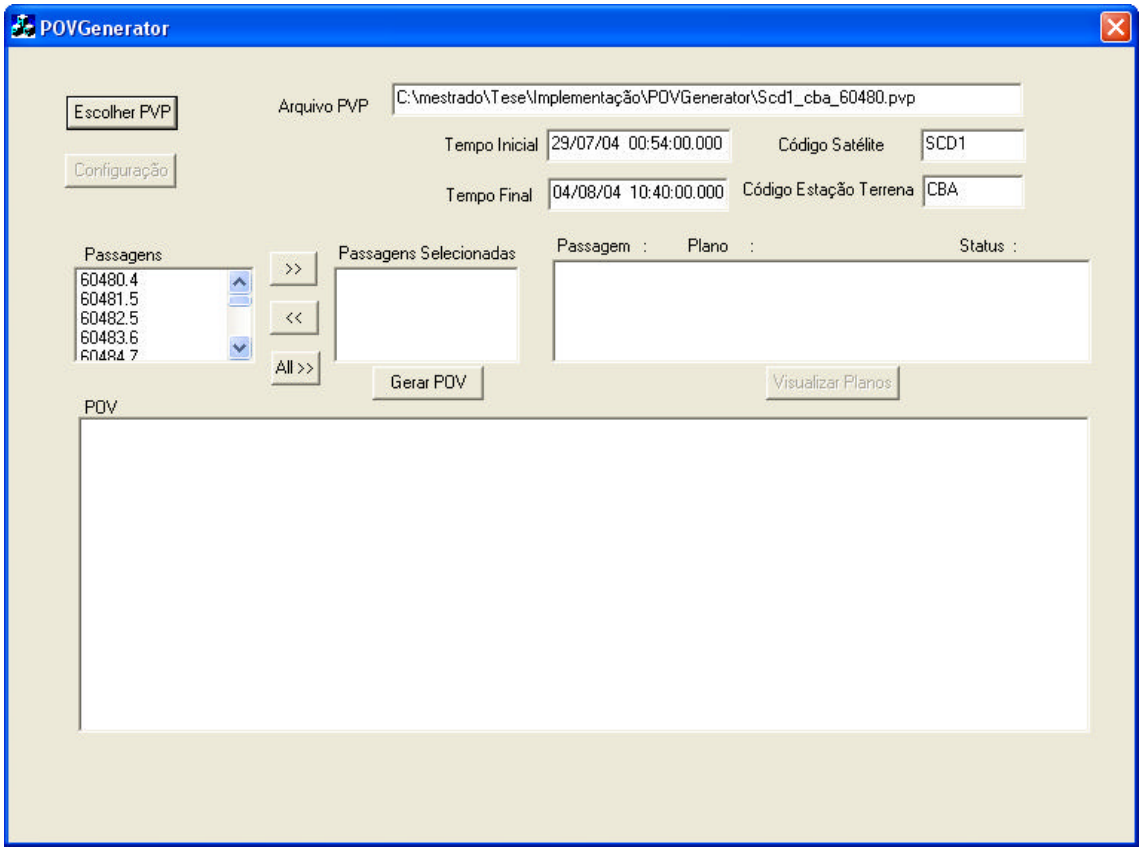

FIGURA 5.6 – Tela atualizada com as informações extraídas do arquivo PVP fornecido.

Com a tela atualizada com as informações do arquivo PVP escolhido, o usuário deverá escolher as passagens para as quais se deseja gerar o POV. O usuário tem a opção de escolher uma, algumas ou todas as passagens listadas no campo *Passagens* da tela. Após a escolha das passagens, o usuário deverá selecionar o botão *Gerar POV* para iniciar a geração do POV para cada passagem escolhida. O resultado dos POVs gerados vão sendo apresentados no lado direito da tela. A Figura 5.7 apresenta a tela atualizada com os resultados da geração dos POVs para as passagens escolhidas. Como pode ser observado, três passagens foram selecionadas para que fossem gerados os respectivos POVs (*Passagens Selecionadas*). Após o botão *Gerar POV* ter sido acionado, o sistema disparou a geração do POV para cada passagem selecionada e o resultado obtido em cada geração foi apresentado na tela informando o número da passagem, o nome do arquivo POV gerado e o status de geração do mesmo. No exemplo, todos foram gerados corretamente.

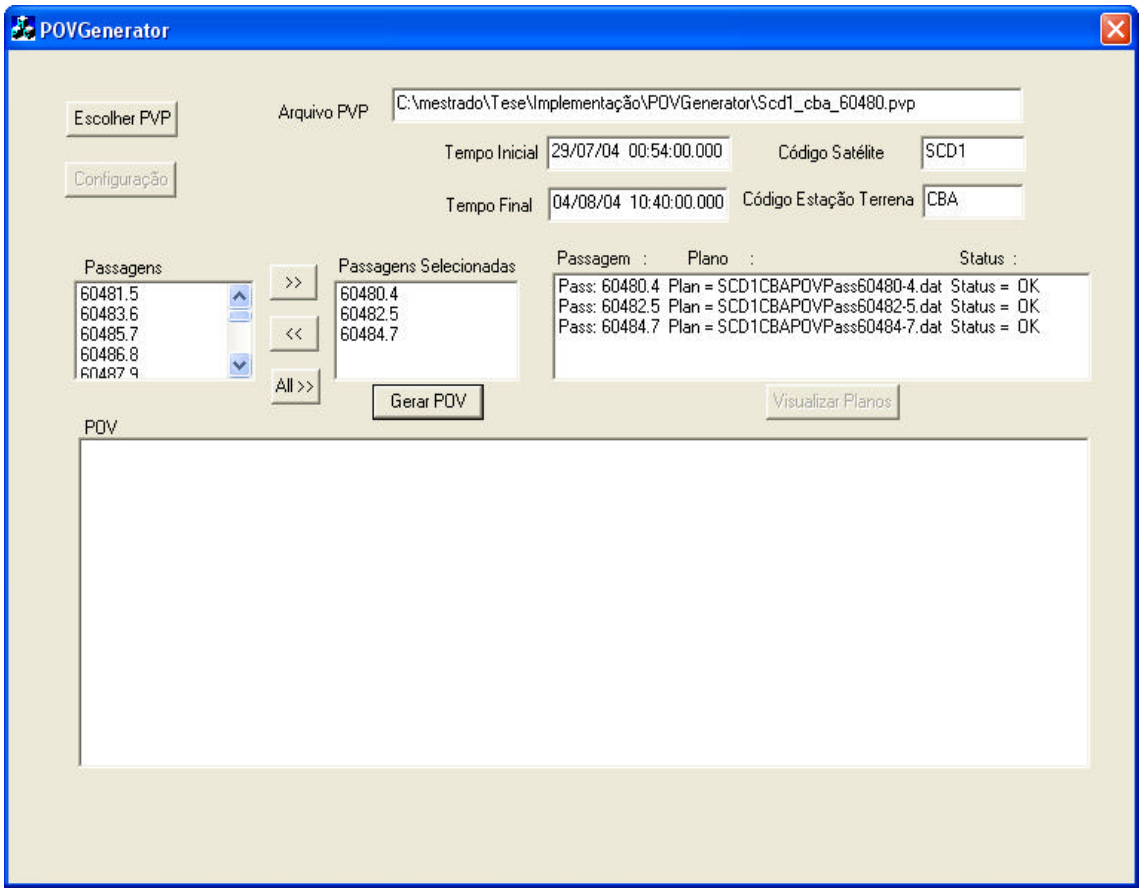

FIGURA 5.7 – Tela com o resultado dos POVs gerados para as passagens selecionadas.

Concluída a geração dos POVs, o usuário tem a opção de visualizar os planos gerados. Para isto, deve-se selecionar os nomes dos arquivos POVs que se deseja visualizar e em seguida selecionar o botão *Visualizar Planos*. O conteúdo dos planos selecionados será apresentado na parte inferior da tela. A Figura 5.8 apresenta uma tela com o conteúdo do plano, gerado para a passagem 60484.7, que foi selecionado pelo usuário.

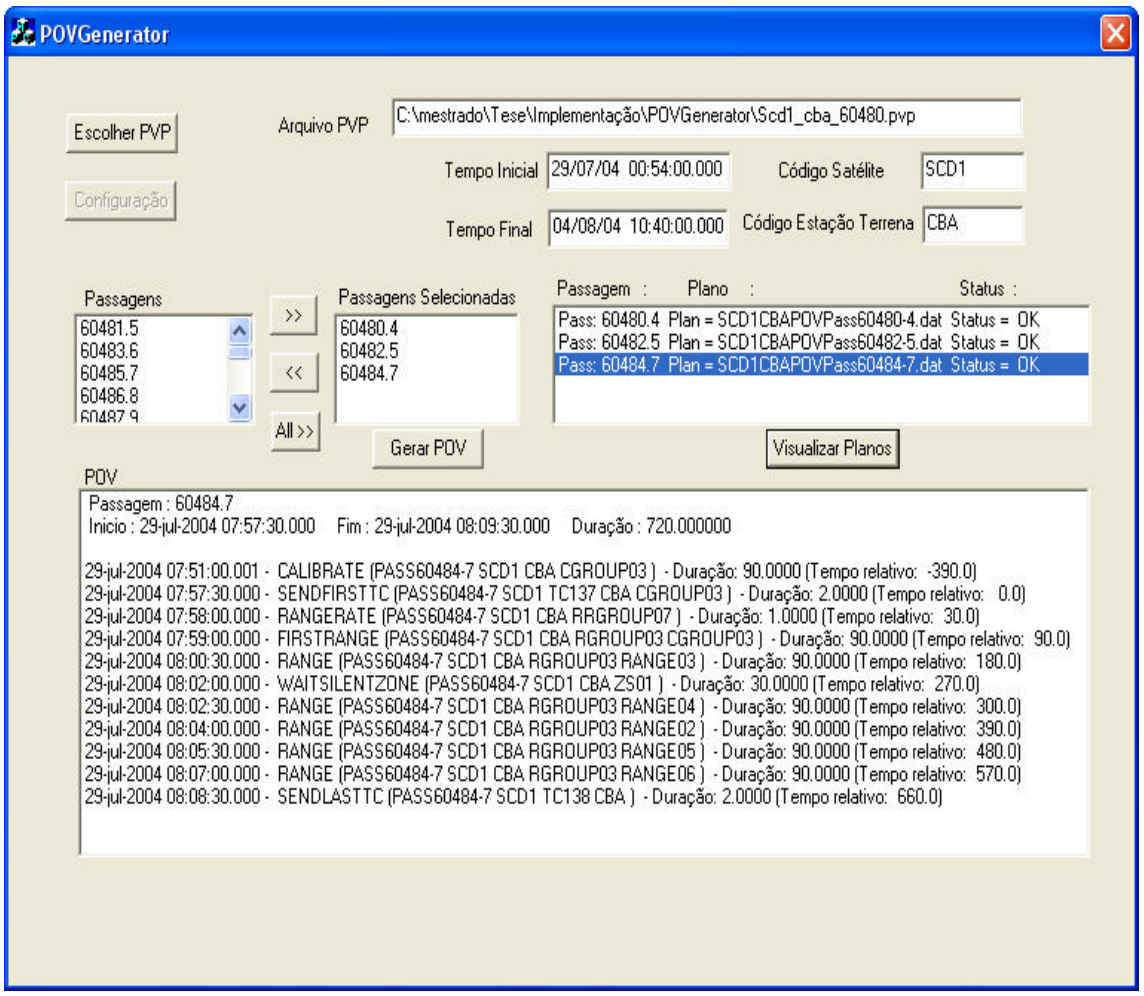

FIGURA 5.8 – Tela com o conteúdo do POV selecionado.

CAPÍTULO 6

APRESENTAÇÃO E ANÁLISE DOS RESULTADOS OBTIDOS

Neste Capítulo são apresentados os resultados obtidos durante os testes realizados para a validação do protótipo. As limitações encontradas na aplicação da tecnologia de planejamento para implementação do protótipo e as soluções adotadas para resolver estas limitações também são apresentadas.

6.1 Teste e Resultados Obtidos

O protótipo desenvolvido foi testado para a geração dos POVs para os satélites SCD1, SCD2 e CBERS2 que atualmente são rastreados pelo INPE.

Para a realização dos testes foi necessária a preparação de uma base de dados de Configuração que refletisse o cenário operacional dos satélites do INPE. O Editor de Configuração de Parâmetros para o POV (vide item 5.1) foi utilizado nesta fase. Os conteúdos dos parâmetros a serem colocados na base de dados foram obtidos dos especialistas responsáveis pelo controle de satélite do CCS. Os seguintes parâmetros foram cadastrados: as estações terrenas que fazem parte do rastreio, os satélites que são rastreados, os telecomandos pertencentes a cada satélite, bem como, a configuração de operação da passagem de cada satélite para cada estação terrena.

Vários arquivos PVP utilizados na operação real dos satélites foram fornecidos pelo CCS. Estes arquivos continham a seqüência de passagem que cobre sete dias específicos de rastreio. Estes arquivos foram utilizados como entradas para o sistema durante a realização dos testes.

O arquivo POVMEDZSRRTCDOMAIN.PDDL, que contém o domínio de rastreio de satélite codificado em PDDL 2.2, também foi uma entrada para o sistema. Este arquivo,

que foi gerado durante a fase de modelagem do domínio da engenharia do conhecimento descrita no item 4.3, se encontra listado no Apêndice A.

De posse da base de dados de configuração, dos arquivos PVP e do arquivo de domínio de rastreio de satélite, os testes foram realizados para cada um dos satélites rastreados pelo INPE.

A primeira fase de testes visou testar a montagem correta do arquivo de problemas pelo protótipo para os vários estados iniciais do ambiente da passagem, os eventos exógenos, os objetivos e as funções. Nesta fase, os arquivos PVP fornecidos pelo CCS foram modificados de forma a permitir testar as condições idealizadas nos casos de testes.

Durante a aplicação dos testes, vários ajustes foram necessários para que o arquivo de problema fosse gerado de acordo com as condições da passagem fornecida pelo arquivo PVP. Alguns ajustes também foram necessários no arquivo de domínio, ou seja, mudanças nas pré-condições e efeitos de algumas ações e criação de novos predicados.

Depois de obtida a geração automática correta dos arquivos de problemas e consequentemente a geração "correta" dos POVs, passou-se para uma segunda fase de teste. Nesta fase, o objetivo era gerar os POVs utilizando os arquivos PVP fornecidos pelo CCS sem modificações via o protótipo e, em seguida, comparar os resultados obtidos com os POVs reais utilizados no rastreio dos satélites do INPE.

Um programa foi desenvolvido para comparar de forma automática o conteúdo dos POVs gerados pelo protótipo e dos POVs reais. Este programa usou os seguintes critérios de comparação:

- Número de ações existentes nos planos;
- Ordem das ações inseridas nos planos;
- A duração de execução de cada ação;
- Os tempos predefinidos (eventos exógenos);

- O tempo inicial e final de cada passagem; e
- Os telecomandos inseridos nos POVs reais para implementar ações fora da rotina de operação foram desprezados.

Nesta fase, várias baterias de testes foram realizadas para cada satélite rastreado pelo INPE tendo como base o arquivo PVP fornecido. Alguns ajustes foram necessários no código do protótipo até que o arquivo de problemas fosse gerado corretamente.

Vários arquivos PVP foram utilizados durante o teste para cada um dos satélites. Como o cenário operacional de cada um dos satélites para a fase de rotina não se modifica muito em função das características das passagens, os resultados obtidos na aplicação dos testes tendo como entrada os vários arquivos PVP não diferiram muito entre si.

A Tabela 6.1 apresenta os resultados obtidos na comparação entre os POVs gerados pelo protótipo e os POVs reais fornecidos pelo CCS para um conjunto de passagens extraído de arquivos PVP. Cada arquivo PVP cobre um período de sete dias de passagens.

TABELA 6.1 – Resultado da Comparação entre os POVs gerados e os POVs reais.

Código Satélite	**Arquivo PVP**			**Resultados**		
	Cód. Passagem	**Período**	**No. de Passagens**	**POVs Iguais**	**POVs Diferentes**	**Sem solução**
SCD1	60480	29/07/04 a 04/08/04	57	36 64%	14 24%	07 12%
SCD2	30444	29/07/04 a 04/08/04	57	46 81%	11 19%	00 0%
CBERS2	9668	24/08/05 a 31/08/05	28	22 78%	06 22%	00 0%

Três tipos de resultados foram encontrados durante a comparação do conjunto de POVs gerados a partir dos arquivos PVP para cada satélite, sendo:

- POVs gerados e reais com conteúdos **iguais**, ou seja, todos os critérios de comparação foram satisfeitos;

- POVs gerados e reais com conteúdos **diferentes**. As diferenças encontradas entre os POVs foram na ordem das ações de pedido de execução de medidas de distâncias e de medidas de velocidade e a quantidade de ações de pedido de execução de medidas de distâncias inseridas no plano. Estas diferenças não invalidam os planos gerados porque todos os eventos exógenos foram inseridos de forma correta no plano. A ação de pedido de execução de medidas de distância não é crucial, o critério aplicado pela operação corrente é fazer o máximo de medida de distância possível durante uma passagem. Desde que os eventos exógenos sejam respeitados, fazer uma medida a mais, ou a menos, e/ou a ordem que as mesmas são executadas não prejudicam a operação do satélite. Esses POVs gerados pelo protótipo poderiam ser usados sem problemas na operação real dos satélites;

- Um plano para o arquivo de problemas gerado pelo protótipo não foi encontrado pelo planejador, ou seja, arquivo de problema **sem solução**. Os planos que não conseguiram ser gerados pelo planejador estão relacionados ao fato de que os objetivos inseridos no arquivo de problemas ultrapassavam o intervalo de tempo disponível na passagem. Esta é uma limitação da linguagem PDDL2.2 e que está descrita no item 6.2.

Como pode ser observado na Tabela 6.1, os resultados encontrados foram diferentes para cada tipo de satélite. O motivo é que existe um cenário operacional diferente para cada satélite.

O satélite SCD1 está operando em modo degradado, isto significa que no início de cada passagem deve ser enviado um telecomando para ligar o transmissor de bordo e no final da mesma um telecomando para desligá-lo.

Para a operação de rotina do satélite SCD2 não é necessário enviar telecomandos, os mesmos são enviados apenas durante operações de manobras.

A operação do satélite CBERS2 requer que telecomandos sejam enviados somente em uma única passagem em três dias específicos da semana (segunda-feira, quarta-feira e sexta-feira), nas outras passagens, o envio de telecomando não é necessário.

Desta forma, o pior resultado obtido foi justamente com os POVs gerados para o satélite SCD1. Os arquivos de problemas em que o planejador não conseguiu encontrar solução foram relacionados à inserção de ações de medidas de distância no plano. Para este satélite dois telecomandos sempre devem ser inseridos em instantes de tempo predeterminados em relação ao início e fim da passagem, reduzindo o intervalo de tempo disponível na passagem para a realização das medidas de distâncias. Consequentemente, nem sempre o número de medidas calculadas para serem realizada naquela passagem encaixava no intervalo de tempo disponível.

A seguir são apresentados alguns trechos de códigos, na linguagem PDDL 2.2, dos eventos exógenos definidos e do objetivo (composto de sub-objetivos) a ser atingido no plano do arquivo de problemas e o plano gerado pelo planejador LPG-TD a partir deste arquivo. O arquivo de problemas foi gerado automaticamente, pelo protótipo, para a passagem 60509.4 do satélite SCD1 sobre a estação terrena de Cuiabá. O código completo do arquivo de problemas gerado pelo protótipo se encontra no Apêndice B.

```
;;Definição dos Eventos exógenos
    (at 0.0 (CalibrationAvailable PASS60509-4 CBA))
    (at 90.0 (not (CalibrationAvailable PASS60509-4 CBA)))
;;
    (at 480.0 (FirstRange PASS60509-4 SCD1   CBA))
    (at 570.0 (not (FirstRange PASS60509-4 SCD1   CBA)))
;;
    (at 390.0 (FirstTc PASS60509-4 SCD1   TC137))
    (at 392.0 (not (FirstTc PASS60509-4 SCD1   TC137)))
```

```
;;
    (at 1050.0 (LastTC PASS60509-4 SCD1   TC138))
    (at 1052.0 (not (LastTC PASS60509-4 SCD1   TC138)))
;;
    (at 420.0 (RRateAvailable PASS60509-4 SCD1   CBA))
    (at 421.0 (not (RRateAvailable PASS60509-4 SCD1   CBA)))
;;
    (at 870.0 (SatSilentZoneAvailable PASS60509-4 SCD1   CBA ZS01))
    (at 900.0 (not (SatSilentZoneAvailable PASS60509-4 SCD1   CBA ZS01)))
;;
;;Definição dos Objetivos a serem atingidos no Plano
(:goal (and
    (LastTCOK PASS60509-4 SCD1   TC138)
    (RRateOK PASS60509-4 SCD1   CBA RRGroup07)
    (RangeOk PASS60509-4 RGroup03  Range02)
    (RangeOk PASS60509-4 RGroup03  Range03)
    (RangeOk PASS60509-4 RGroup03  Range04)
    (RangeOk PASS60509-4 RGroup03  Range05)
    (SilentZone PASS60509-4 SCD1   CBA ZS01)
))
```

POV gerado pelo planejador:

0.0003: (CALIBRATE PASS60509-4 SCD1 CBA CGROUP03) [90.0000]

390.0005: (SENDFIRSTTC PASS60509-4 SCD1 TC137 CBA CGROUP03) [2.0000]

420.0008: (RANGERATE PASS60509-4 SCD1 CBA RRGROUP07) [1.0000]

480.0010: (FIRSTRANGE PASS60509-4 SCD1 CBA RGROUP03 CGROUP03) [90.0000]

570.0012: (RANGE PASS60509-4 SCD1 CBA RGROUP03 RANGE02) [90.0000]

660.0015: (RANGE PASS60509-4 SCD1 CBA RGROUP03 RANGE03) [90.0000]

750.0018: (RANGE PASS60509-4 SCD1 CBA RGROUP03 RANGE04) [90.0000]

870.0020: (WAITSILENTZONE PASS60509-4 SCD1 CBA ZS01) [30.0000]

900.0023: (RANGE PASS60509-4 SCD1 CBA RGROUP03 RANGE05) [90.0000]

1050.0024: (SENDLASTTC PASS60509-4 SCD1 TC138 CBA) [2.0000]

Cada linha do plano gerado pelo planejador é composta dos seguintes itens: (i) tempo relativo ao início da passagem no qual a ação deve ser executada; (ii) a ação a ser executada acompanhada dos parâmetros que a identificam; e (iii) tempo necessário para a execução da ação (duração).

A Figura 6.1 apresenta a tela principal do protótipo com o conteúdo deste mesmo plano formato para ser visualizado pelo usuário.

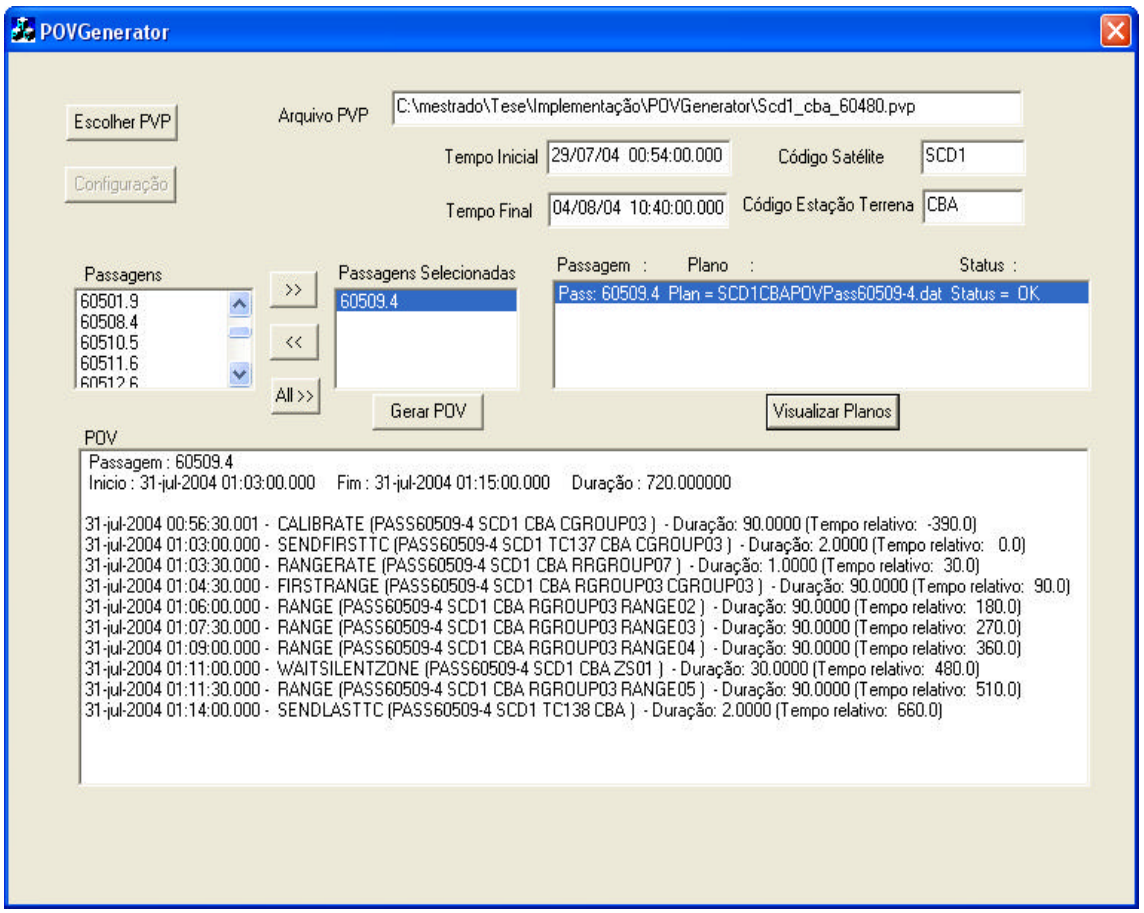

FIGURA 6.1 – POV gerado para a passagem 60509-4 do satélite SCD1.

6.2 Limitações Encontradas na Utilização da Linguagem PDDL 2.2 e Soluções Adotadas

Durante o processo de desenvolvimento e validação da base de conhecimento do domínio de rastreio de satélite e do protótipo do sistema PlanIPOV proposto, foram observadas algumas limitações da linguagem PDDL 2.2.

A primeira limitação encontrada foi em relação à dificuldade na geração dos arquivos de problemas a cada mudança do ambiente durante a fase de modelagem da base de conhecimento do domínio de rastreio de satélite. Após a geração correta do arquivo de domínio constatou-se que uma grande parte do trabalho se concentrava na geração dos arquivos de problemas. A cada mudança do ambiente, que no caso do domínio de rastreio de satélite muda a cada passagem a ser rastreada, era necessário re-programar as funções, os eventos iniciais do ambiente, os eventos exógenos e os objetivos a partir do arquivo de entrada PVP. A geração manual dos arquivos de problemas tinha um custo muito alto.

Esta dificuldade foi solucionada com a implementação dos módulos de Configuração e de Gerador de Problemas do sistema PlanIPOV que permitem a geração automática dos arquivos de problemas para cada passagem a ser rastreada.

Durante a fase de desenvolvimento do protótipo observou-se que a estrutura oferecida pela linguagem PDDL 2.2 não permitia a classificação dos predicados no arquivo de domínio. Sem esta classificação, a geração automática do arquivo de problemas seria muito trabalhosa já que para cada tipo de predicado deve-se dar um tratamento diferente, uma vez que os mesmos devem ser inseridos em lugares diferentes dentro do arquivo de problemas com estruturas diferentes. Por exemplo, um predicado que é de objetivo deve ser colocado após a linha fixa *:goal* no arquivo de problemas. Sendo assim, o código a ser desenvolvido não poderia dar uma solução genérica para os tipos de predicados e sim, deveria tratar cada predicado de forma individual.

A solução adotada foi a definição de categorias de predicados e o nome de cada categoria definida foi inserido no arquivo de domínio como uma linha de comentário.

Os predicados então foram inseridos no arquivo de domínio abaixo da linha de comentário que representasse sua categoria. As categorias de predicados definidas foram: estados iniciais do ambiente (*Initial Environment States*), estados intermediários (*Internal Events*), eventos exógenos (*Exogenous Events*) e estados objetivos (*Goals*). A explicação de cada um dos tipos definidos se encontra no item 4.3. A seguir é apresentado um trecho do código do arquivo de domínio onde foram inseridas as linhas de comentários para a classificação dos predicados. As linhas de comentários estão em negrito.

```
(:predicates
        ;;Internal Events
        (TCSent ?pass - PassNumber ?sat - SatCode)
        (CalibrationOK ?pass - PassNumber ?cgroup - CalibGroupNumber)
        ;;Initial Environment States
        (GSAvailable  ?gs - GSCode)
        (PassTrack ?pass - PassNumber ?sat - SatCode ?gs - GSCode)
        ;;Exogenous Events
        (CalibrationAvailable ?pass - PassNumber  ?gs - GSCode)
        (FirstTc ?pass - PassNumber ?sat - SatCode ?tc - TCCode)
        (LastTC ?pass - PassNumber ?sat - SatCode ?tc - TCCode)
        ;;Goals
        (FirstRangeOk ?pass - PassNumber ?rgroup - RangeGroupNumber)
        (PassEliminated ?pass - PassNumber ?sat - SatCode ?gs - GSCode)
        (LastTCOK ?pass - PassNumber ?sat - SatCode ?tc - TCCode)
 )
```

Esta classificação permitiu que o código de geração automática de problemas desse um tratamento genérico na geração e montagem de cada predicado por meio de seu tipo. Os eventos intermediários não foram tratados pelo programa.

Uma outra limitação encontrada na linguagem é de não permitir definir prioridades nos eventos exógenos. Isto significa que se dois eventos exógenos tem o mesmo horário de ocorrência, o planejador não consegue encontrar um plano válido. Para exemplificar, se

um telecomando deve ser enviado no início de certa passagem em um horário predeterminado e se no início desta mesma passagem ocorre uma zona de silêncio, o envio de telecomando deveria ser deslocado para depois da zona de silêncio, pois a zona de silêncio tem prioridade sobre o envio de telecomando.

Para solucionar este problema, foi implementada uma rotina que estabelece prioridades entre os eventos exógenos. Após a geração de todos os eventos exógenos, antes de serem inseridos no arquivo de problemas, esta rotina é chamada. No código desta rotina foram estabelecidas as prioridades para cada um dos eventos exógenos. De posse destas prioridades e da lista dos eventos definidos para a passagem, os tempos dos eventos que são coincidentes são alterados de acordo com suas prioridades. Para cada mudança, os tempos relativos de início e fim dos eventos exógenos devem ser recalculados.

A Figura 6.2 ilustra um exemplo da aplicação do critério de prioridades em eventos exógenos superpostos em uma passagem.

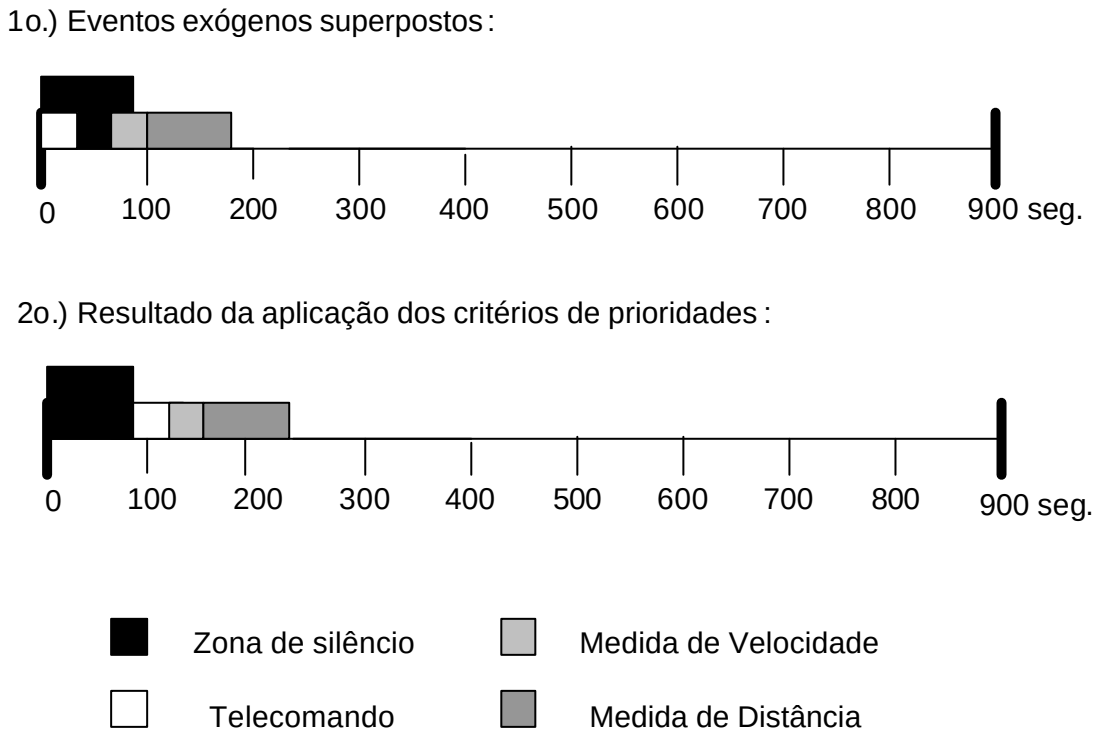

FIGURA 6.2 – Prioridades entre os eventos exógenos.

No primeiro gráfico é apresentado o início da passagem com 3 eventos exógenos superpostos (zona de silêncio, primeiro telecomando a ser enviado na passagem e medida de velocidade a ser executada). Após a aplicação dos critérios de prioridades, a situação final da definição dos eventos fica do modo mostrado no segundo gráfico. O evento Zona de silêncio é mantido por ter uma maior prioridade e os outros eventos são deslocados para frente também respeitando a prioridade. O envio de telecomando tem prioridade sobre a execução das medidas de distância e de velocidade.

Uma outra limitação da linguagem, também relacionada à prioridade, é de não permitir o estabelecimento de prioridades nos sub-objetivos a serem alcançados no plano. Por exemplo, quando o arquivo de problema tem cinco sub-objetivos a serem atingidos, que ultrapassam o intervalo de tempo disponível para a geração total do plano, o planejador não consegue encontrar um plano válido. Seria interessante ser possível definir prioridades para os sub-objetivos, porque, desta forma, um plano poderia ser gerado cumprindo parte do objetivo proposto. Os sub-objetivos de baixas prioridades seriam descartados quando não fosse possível atingir todos os objetivos no plano a ser gerado, ao invés de não encontrar nenhuma solução para o objetivo proposto.

Para esta limitação foi implementada uma rotina para descartar os sub-objetivos que ultrapassassem o intervalo de tempo disponível para a passagem. Os arquivos de problemas gerados para o satélite SCD1 em que o planejador não conseguiu encontrar um plano válido (ver Tabela 6.1, do item 6.1) se encaixam nesta limitação. Isto se deu porque a rotina implementada não tratou todas as situações possíveis em que poderia ocorrer a inserção de mais pedidos de medida de distância que o intervalo disponível na passagem seria capaz de suportar.

A Figura 6.3 ilustra o uso de prioridade para descartar sub-objetivos. O primeiro gráfico apresenta dois eventos exógenos superpostos (primeira medida de distância e zona de silêncio) e o restante do intervalo de tempo da passagem comporta mais cinco medidas de distância, já que cada medida tem a duração de noventa segundos.

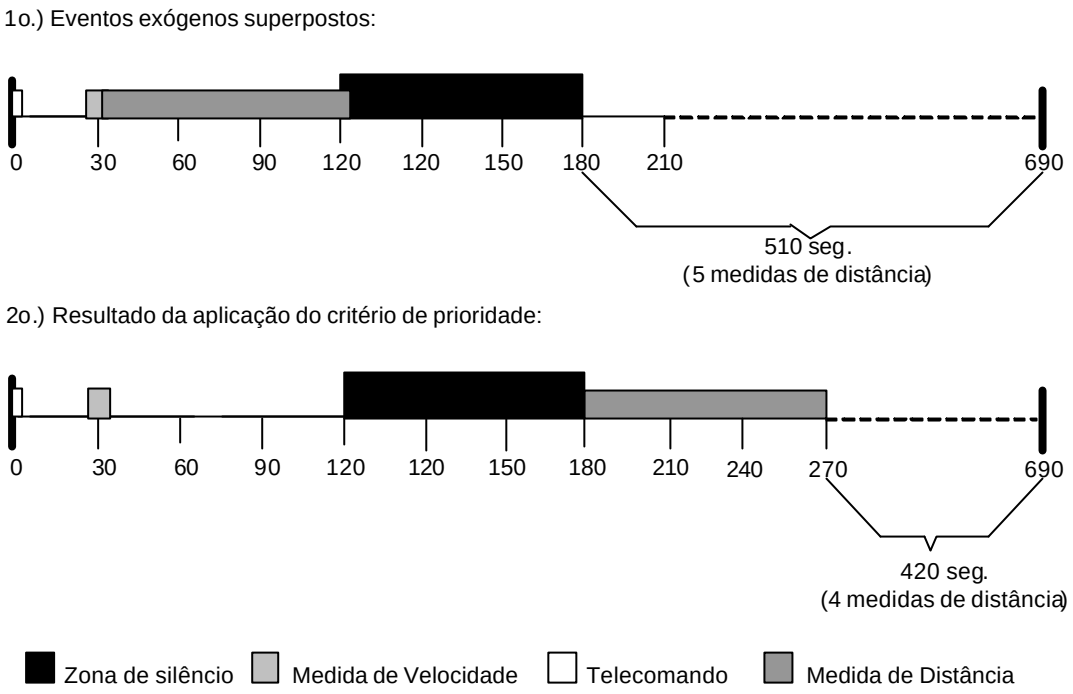

FIGURA 6.3 – Prioridades nos sub-objetivos.

Após a aplicação do critério de prioridade entre os eventos exógenos, ilustrado no segundo gráfico, o evento de zona de silêncio foi mantido por ter maior prioridade enquanto o evento de primeira medida de distância foi deslocado para frente. Com isso, o intervalo de tempo da passagem disponível para execução de medidas diminuiu e consequentemente, o número de medidas a serem feitas nesse intervalo também. Se fosse possível definir prioridades para os sub-objetivos, o arquivo de problema não precisaria ser modificado em relação à quantidade de medidas de distância a serem inseridas no plano, ou seja, o sub-objetivo continuaria a ser a inserção de seis pedidos de medidas de distância no plano, e o resultado seria a inserção de apenas cinco pedidos no plano pelo planejador.

Um grande número de linhas de código poderia ter sido evitada para permitir a geração automática do arquivo de problemas se a linguagem permitisse a definição de prioridades nos eventos exógenos e nos sub-objetivos.

CAPÍTULO 7

CONCLUSÃO

A criação de um sistema de planejamento inteligente de POVs, para a fase operacional de rotina dos satélites, pode ser considerada o primeiro estágio na direção de uma maior automatização das operações de satélite do INPE.

A geração automática dos POVs neste trabalho baseou-se na tecnologia de agente de planejamento da IA. O caminho encontrado durante a pesquisa, e posteriormente aplicado na implementação do protótipo, foi a utilização da linguagem padrão de planejamento PDDL 2.2 na geração da base de conhecimento do domínio de rastreio de satélite.

O protótipo implementado, como proposto pela arquitetura PlanIPOV (vide item 4.2), incorpora a arquitetura de um agente de planejamento e gera de forma automática o arquivo de problemas em PDDL 2.2 para cada passagem do satélite a ser rastreado. As limitações encontradas na linguagem PDDL 2.2 para o problema de planejamento dos POVs foram tratadas internamente pelo protótipo possibilitando assim a geração de POVs para os satélites rastreados pelo INPE.

A seguir são apresentadas as principais contribuições do trabalho destacando os pontos favoráveis e desfavoráveis encontrados durante a aplicação da tecnologia de agente de planejamento e os trabalhos futuros vislumbrados.

7.1 Contribuições

7.1.1 Pontos Favoráveis

Ao longo do desenvolvimento deste trabalho, pôde-se identificar pontos favoráveis na utilização da tecnologia de agente de planejamento aplicada a problemas do mundo real, que são detalhados a seguir:

- Um ponto positivo nesse tipo de abordagem é que a forma de pensar no domínio do problema é próxima à forma de pensar dos especialistas do domínio. Como é possível tratar as pré-condições e efeitos de cada ação a troca de informações com os especialistas fica mais fácil e eles conseguem dar sugestões e encontrar problemas ou conflitos de uma forma mais rápida;

- Esta técnica orienta, ou melhor, "obriga" a documentação do conhecimento por meio da criação da base de conhecimento. Nesta base são registradas todas as informações necessárias e importantes que representam o conhecimento do problema de planejamento extraído do especialista. Este aspecto da abordagem evita que o conhecimento do domínio possa ser em algum momento perdido por falta de documentação. Se a base de conhecimento for bem construída e testada, ela conterá o registro fiel e completo do conhecimento do domínio do problema que está sendo tratado;

- A arquitetura do agente de planejamento utilizando a linguagem PDDL permite a desvinculação do arquivo do domínio de planejamento do arquivo de problema (ambiente inicial e objetivo a ser atingido no plano). Esta característica dá uma maior flexibilidade ao sistema, permitindo variações do ambiente inicial e dos objetivos sem afetar o domínio. Ou seja, o mesmo domínio de um problema de planejamento pode ser aplicado a várias instâncias do problema sem que seja necessário fazer mudanças no arquivo de domínio para cada caso.

7.1.2 Pontos Desfavoráveis

Os pontos observados que dificultam a aplicação da tecnologia de agente de planejamento em problemas do mundo real são:

- A limitação da linguagem PDDL 2.2 em não permitir a definição de prioridades em eventos exógenos e em objetivos (vide item 6.2) dificulta a geração dos arquivos de problemas. Tarefas que poderiam ser executadas pelo planejador são transferidas para a etapa de geração do arquivo de problemas. Desta forma, uma

parte da solução do plano passa para a geração do arquivo de problema onde devem ser definidos a ordem dos eventos e os sub-objetivos que possam realmente ser atingidos. Esta limitação insere uma maior complexidade no código de geração automática dos arquivos de problemas que além de instanciar o problema para o ambiente desejado (ambiente inicial e objetivo) deve se preocupar com problemas de escalonamento que deveria ser tarefa do planejador.

- A idéia inicial do trabalho, era gerar uma arquitetura que permitisse trocar o planejador a ser utilizado na solução do problema sem modificar as interfaces definidas entre o sistema e o planejador. Isso não foi possível porque embora existam, até o momento da conclusão deste trabalho, apenas dois planejadores preparados para trabalhar com a linguagem PDDL 2.2, os dados de entrada e de saída dos mesmos não são padronizados, exigindo assim uma adaptação na interface do sistema para cada um deles.

- A criação dos arquivos de problemas se apresenta como uma grande dificuldade na utilização de agentes de planejamento para a geração de planos em um ambiente onde o estado inicial e o objetivo mudam a cada plano a ser gerado. Sem a automatização na geração desses arquivos de problemas, não se teria nenhuma vantagem em aplicar esta técnica de planejamento em aplicações do mundo real pela grande quantidade de trabalho manual acarretado.

- A geração automática do arquivo de problemas insere uma outra dificuldade na utilização da técnica. O código implementado para a geração automática é totalmente dependente do arquivo de domínio, ou seja, se o arquivo de domínio for alterado, estas alterações deverão ser refletidas no código. Desta forma, para um grupo de satélites com as mesmas características de operação, o arquivo de domínio é o mesmo. Se um novo satélite tiver que ser rastreado e o mesmo apresentar características totalmente novas, o arquivo de domínio e o sistema para a geração automática dos arquivos de problemas devem ser alterados para atender os novos requisitos. O custo e benefício de atualizar o que já existe ou de

se desenvolver um novo arquivo de domínio e um sistema específico para este domínio devem ser considerados.

7.1.3 Principais Contribuições

Além dos pontos favoráveis e desfavoráveis apresentados na utilização da tecnologia de agente de planejamento, as principais contribuições obtidas com a pesquisa e a implementação de uma solução para a geração automática de POVs para os satélite do INPE foram:

- Definição da arquitetura do sistema PlanIPOV que incorpora em sua estrutura a arquitetura de um agente de planejamento. O protótipo implementado baseado nesta arquitetura gerou resultados que mostram a viabilidade do uso da tecnológica de planejamento nas operações espaciais (vide item 4.2). Os resultados obtidos com o protótipo mostraram que é possível alterar o cenário de operação dos satélites do INPE sem que seja necessário modificar o arquivo de domínio e o código. No caso do controle de um novo satélite com os mesmos requisitos de operação dos satélites anteriores será suficiente inserir seus dados na base de dados de configuração do sistema. De forma análoga, para inserir uma nova estação terrena de rastreio no planejamento será suficiente inserir seus dados na mesma base de dados. Em ambos os casos, é necessário registrar na base de dados de configuração a conexão entre satélite e as estações terrenas que o rastreiam. Toda a alteração se concentra na atualização da base de dados de configuração e nenhum código precisa ser modificado. Esta característica do sistema é muito útil durante a fase de treinamento dos operadores onde as variações do estado inicial do ambiente e dos objetivos são importantes para a simulação dos ambientes de operação.

- Definição de um processo de modelagem que pode ser aplicado na criação de uma base de conhecimento para qualquer domínio de aplicação que necessite resolver um problema de planejamento utilizando a linguagem PDDL (vide item

4.3). Esse processo foi aplicado com sucesso na definição da base de conhecimento do domínio de rastreio de satélites do INPE;

- Definição de um *framework* para a geração de planos usando a linguagem PDDL (vide item 4.4). Este *framework* orienta a definição do projeto de um sistema que tenha como objetivo gerar de forma automática os arquivos de problemas na linguagem PDDL. O projeto do protótipo desenvolvido teve como base o *framework* proposto;

- Classificação dos predicados em quatro categorias (estado inicial do ambiente, estado intermediário, eventos exógenos e estado objetivo) no arquivo de domínio para facilitar a geração automática dos arquivos de problemas;

7.2 Trabalhos Futuros

A conclusão deste trabalho não encerra, de forma alguma, as considerações a serem feitas em relação ao uso da tecnologia de agente de planejamento na automatização de operações de satélites. Na verdade, este trabalho abre uma série de questões que podem ser exploradas, como por exemplo:

- Execução automática dos POVs gerados. Este item enfoca o módulo Executor da arquitetura do agente de planejamento. Este módulo deve atuar no ambiente de operação por meio da ativação das funções do sistema de controle de satélite que correspondem às ações definidas no plano. Aqui se abre uma questão: é suficiente o agente tratar separadamente as funções de planejamento e execução do plano ou é necessário a utilização de re-planejamento automático?

- Desenvolvimento de um planejador para tratar os problemas específicos encontrados no planejamento de operações de satélites. Este item abre um leque de dúvidas a serem respondidas, como por exemplo, qual o melhor algoritmo, adoção ou não da linguagem PDDL e que características da linguagem são realmente importantes para este tipo de domínio de planejamento.

Um outro nicho de pesquisa que começa a despontar na área de planejamento da IA está relacionado à aplicação da engenharia do conhecimento juntamente com a tecnologia de planejamento para o desenvolvimento de sistemas eficientes. Isto pôde ser observado na Conferência Internacional de Planejamento e Escalonamento Automático (ICAPS, 2005), onde foi inserida pela primeira vez a competição de engenharia de conhecimento que tem como objetivos: (i) fomentar os aspectos de modelagem do domínio baseado no conhecimento para o problema de planejamento; (ii) acelerar a pesquisa na engenharia do conhecimento dentro da área de planejamento da IA e (iii) encorajar o desenvolvimento e o compartilhamento de protótipos de ferramentas ou plataformas de software que propiciem caminhos mais rápidos, acessíveis e efetivos para a construção de sistemas de planejamento e escalonamento eficientes e confiáveis. O termo de engenharia do conhecimento voltada para a área de planejamento, recebeu a seguinte definição no ICAPS-2005: "o processo que trata da aquisição, validação e manutenção dos modelos do domínio de planejamento e otimização do maquinário de planejamento apropriado para trabalhar sobre eles".

7.3 Considerações Finais

Por meio do desenvolvimento deste trabalho, e especialmente após a modelagem da base de conhecimento do domínio de rastreio de satélite e da construção do protótipo, pôde-se constatar que a aplicação da tecnologia de agente de planejamento para a automatização de operações de controle de satélite é um caminho bastante promissor para redução de custos de operação das missões espaciais no INPE na fase operacional de rotina.

A automatização nas operações de satélite do INPE é uma necessidade iminente. Os resultados obtidos com este trabalho mostram que a tecnologia de planejamento da IA é sem dúvida um dos caminhos viáveis e promissores para responder a esta necessidade de automatização. Apesar das limitações mostradas, a tecnologia de planejamento utilizando IA está em grande expansão e novas soluções estão sendo encontradas de forma a facilitar sua utilização nos sistemas do mundo real.

As idéias deste trabalho já foram publicadas, até o momento, em Cardoso (2005) e Cardoso (2005b). E o artigo *An Intelligente System for Automatic Flight Operation Plans Generation for to Satellite Control Activities at INPE* foi aceito e será apresentado no SPACEOPS 2006 que se realizará no período de 19 a 23 de junho.

Espera-se, por meio do desenvolvimento deste trabalho, colaborar, mesmo que de forma modesta para o avanço da pesquisa no Brasil, e para o sucesso da missão espacial brasileira, oferecendo uma possível solução de automatização das operações espaciais visando à redução dos custos totais com o controle de satélite. Adicionalmente se espera ter colaborado com a comunidade de planejamento por meio da aplicação da tecnologia em um problema do mundo real.

REFERÊNCIAS BIBLIOGRÁFICAS

Ai-Chang, M.; Bresina, J.; Charest, L.; Chase, A.; Cheng-jung, J.; Jonsson, A.; Kanefsky, B.; Morris, P.; Rajan, K.; Yglesias, J.; Chafin, B.; Dias, W.; Maldague, P. MAPGEN: Mixed-initiative planning and scheduling for the Mars Exploration Rover Mission. **IEEE Intelligent Systems**, v. 19, n.1, p. 8-12, Jan/Feb 2004.

Bacchus, F.; Ady, M. Planning with resources and concurrency: a forward chaining approach, In: International Joint Conference on Artificial Intelligence, 17. (IJCAI), Aug 04 – 10 2001, Seatle, Washington, USA. **Proceedings**… San Francisco: Morgan Kaufmann, 2001. (ISBN 1-55860-777-3).

Berloti, P.; Cimatti, A.; Dal Lago, U.; Pistore, M. Extending PDDL to nondeterminism, limited sensing and iterative conditional plans. In: International Conference on Automated Planning and Scheduling Workshop on PDDL, 13. (ICAPS), June10 2003, Trento, Italy. **Proceedings** … Trento, Italy, AAAI Press, p.15-24, 2003. (ISBN 978-1-57735-187-0).

Blum, A.; Furst, M. Fast planning through planning graph analysis, **Artificial Intelligence**, v. 90, p.:281-300, 1997. Special issue on the 1995 Fourteenth International Joint Conference on Artificial Intelligence (IJCAI'95).

Bonet, B.;Geffner, H. HSP: Heuristic Search Planner. **AI Magazine**, v.21, n.2, p. 13 – 33, Summer, 2000. (International Conference on Artificial Planning Systems, 4. (AIPS), June 7 – 10 1998, Pittsburg, Pennsylvania, USA).

Cardoso, L.; Ferreira, M.; Orlando, V.; Biancho, A. Aplicação da tecnologia de agentes de planejamento em operações de satélites. In: Simpósio Brasileiro de Automação Inteligente, 7. (SBAI), 20 – 23 set 2005, São Luís, Maranhão, Brasil. **Anais**... São Luis, 2005 (ISBN 85-85048-55-7).

Cardoso, L.; Ferreira, M.; Orlando, V. Aplicação da tecnologia de agentes de planejamento em operações de satélites. In: Workshop dos Cursos de Computação

Aplicada do INPE, 5. (WORCAP), out 2005, São José dos Campos, São Paulo, Brasil. **Anais**...São José dos Campos: INPE, 2005.

Chien, S.; Rabideau, G.; Knight, R.; Engelhardt, B.; Mutz, D.; Estlin, T.; Smith, B.; Fisher, F.; Barret, T.; Stebbins, G.; Tran, D. ASPEN – Automated planning and scheduling for space mission operation. In: International Conference on Space Operations, 7. (SPACEOPS), Oct 9 – 12 2002, Houston, Texas, USA. **Proceedings**… Disponível em: <http://www.spaceops2002.org/conference.html>. Acesso em: 31 de março de 2006.

Davis, R.; Shrobe, H.; Szolovits, P. What is a knowledge representation? **AI Magazine**, v. 14, n. 1, p. 17 a 33, Spring 1993.

Do, M.; Kambhampati, S. SAPA: A Multi-objetive metric temporal planner. **Journal of Artificial Intelligence Research**, v. 20, p. 155-194, Dec 2003. (ISSN-1076-9757).

Doherty, P.; Kvarnstrom, J. TALPlanner: a Temporal Logic based Planner. **AI Magazine**, v. 22, n. 3, p. 95 – 102, Fall 2001.

Donati, A.;Romani, E. INTELMOD – An intelligent system for capturing operations knowledge and providing advanced operations support. **European Space Agency Bulletin**, n. 102, May 2000.

European Cooperation for Space Standardization. **ECSS-E-70:** Space Engineering – Ground Systems and Operations – Part 1: Principles and Requirements, ESA-ESTEC, Requirements & Standards Division. Noordwijk, The Netherlands, Apr 2000.

Edelkamp, S.; Helmert, M. MIPS: The Model-checking Integrated Planning System. **AI Magazine**, v. 22, n. 3, p. 67 – 72, Fall 2001.

Edelkamp, S.;Hoffmann, J. **PDDL 2.2:** the language for the classical part of the 4th International Planning Competition. Freiburg, Germany, Albert Ludwigs Universität Institüt fur Informatik: Technical Report n.195, Jan 2004.

Ercolani, A.; Ferri, P.; Simonic, A.; Kowalczyk, A.; Ulriksen, T. Operations automation for Rosetta Mission. In: International Conference on Space Operations, 8. (SPACEOPS), May 17 – 21 2004, Montreal, Quebec, Canada. **Proceedings**…Montreal, Canada: American Institute of Aeronautics and Astronautics, 2004.

Feber, J. **Multi-agent systems an introduction to distributed artificial intelligence**. London: Addison-Wesley (ed). 1999. 509p. (ISBN 0-201-36048-9)

Ferri, P.; Pecchioli, M.; Pena, A. Introducing automation in ground segment operations In: International Conference on Space Operations, 8. (SPACEOPS), May 17 – 21 2004, Montreal, Quebec, Canada. **Proceedings**… Montreal, Canada: American Institute of Aeronautics and Astronautics, 2004.

Fikes, R. E.; Nilsson, N. J. STRIPS: a new approach to the application of theorem proving to problem-solving, **Artificial Intelligence**, v. 2, n. 3 - 4, p.:189 -208, 1971.

Frank, J.; Golden, K.; Jonsson, A. The loyal opposition comments on plan domain description languages. In: International Conference on Automated Planning and Scheduling Workshop on PDDL, 13. (ICAPS), June 10 2003, Trento, Italy. **Proceedings**… Menlo Park, California: AAAI press, 2003. (ISBN 978-1-57735-187-0).

Franklin, S.;Graesser, A. Is it an agent, or just a program? A taxonomy for autonomous agents. In: International Workshop on Agent Theories, Architectures, and Languages, 3. (ATAL), 1996, Berlin, Germany. **Proceedings**… [s.l.]: Springer-Verlag, 1996. (ISBN 3-540-62507-0).

Fox, M.; Long, D. PDDL2.1: An extension to PDDL for expressing temporal planning domains, **Journal of Artificial Intelligence Research**, v.20, p. 61-124, Dec 2003. (ISSN-1076-9757).

Gerevini, A.; Serina, I. LPG: a planner based on planning graphs with action costs. In: International Conference on AI Planning and Scheduling, 6. (AIPS), Apr 23 – 27 2002, Toulouse, France. **Proceedings**… Menlo Park, California: AAAI Press, p.13-22, 2002. (ISBN 978-1-57735-142-9).

Gerevini, A.; Saetti, A.; Serinia, I. Planning through stochastic local search and temporal action graphs in LPG. **Journal of Artificial Intelligence Research**, v.20, p. 239-290, Dec 2003. (ISSN-1076-9757).

Gerevini, A.; Saetti, A.; Serinia, I. An empirical analysis of aome heuristic features for local search in LPG. In: International Conference on Automated Planning and Scheduling 14. (ICAPS), June 3-7 2004, Whistler, British Columbia, Canada. **Proceedings**… Menlo Park, California: AAAI press, 2004. (ISBN 978-1-57735-200-6).

Gerevini, A.; Saetti, A.; Serina, I.; Toninelli, P. LPG-TD: a fully automated planner for PDDL2.2 domains. In: International Conference on Automated Planning and Scheduling 14. (ICAPS), June 3-7 2004, Whistler, British Columbia, Canada. **Proceedings**… Whistler: ICAPS, 2004. (booklet of the system demo section).

Gerevini, A.; Saetti, A.; Serina, I. Integrating planning and temporal reasoning for domains with durations and time windows. In: International Joint Conference on Artificial Intelligence, 19. (IJCAI), July 30 – Aug 05 2005, Edinburgh, Scotland, UK **Proceedings**… Edinburgh, Scotland, UK: IJCAI-Inc, 2005.

International Conference on Automated Planning and Scheduling (**ICAPS**). Disponível em: <http://icaps05.icaps-conference.org/>. Acesso em: 03 de abr de 2006.

International Planning Competition (**IPC**). Disponível em: <http:\\ipc.icaps-conference.org/>. Acesso em: 03 de abr de 2006.

International Organization for Standardization **ISO 14950:2004**: Space Systems – Unmanned Spacecraft Operability, ACE/68, Apr 02 2004. p.34. (ISBN 0-580-43607-1).

Kautz, H.; Selman, B. Planning as satisfiability. In: European Conference on Artificial Intelligence, 10. (ECAI), Aug 1992, Vienna, Austria. **Proceedings**… Vienna, Austria: John Wiley & Sons, p. 359 – 363, 1992.

Kautz, H.; McAllester, D.; Selman, B. Encoding plans in propositional logic. In: International Conference on Principles of Knowledge Representation and Reasoning, 5.

(KR), Nov 5 – 8 1996, Cambridge, MA USA. **Proceedings**… San Francisco: Morgan Kaufmann, p.374-385.. 1996. (ISBN 1-55860-421-9).

Kavuluri, B.; Khemani, D. Timed initial literals in temporal planning. In: International Conference on Knowledge based Computer Systems, 5. (KBCS), Dec 19 – 22 2004, Hyderabad, India. **Proceedings**…<http://aidb.cs.iitm.ernet.in/students/bharat/.> Acesso em 23 de ag de 2004.

Knoblock, M.; Kuokka, G.; Joseph, R.; Carbonell, J. **Prodigy 2.0:** the manual and tutorial. School of Computer Science, University of Carnegie Mellon, Pittsburg: Technical. Report CMU-CS-89-146, 1989.

Ko, A.; Maldague, P.; Page, D.; Bixler, J.; Lever, S.; Cheung, K. Design and architecture of planning and sequence system for Mars Exploration Rover (MER) operations. In: International Conference on Space Operations, 8. (SPACEOPS), May 17 – 21 2004, Montreal, Quebec, Canada. **Proceedings**… Montreal, Canada: American Institute of Aeronautics and Astronautics, 2004.

Kvarnström, J.; Magnusson, M. TALplanner in IPC-2002: Extensions and control rules. **Journal of Artificial Intelligence Research**, v.20, p. 343-377, Dec 2003. (ISSN-1076-9757).

Kuga, H.; Kondapalli, R. Satellite orbit determination: a first-hand experience with the first brazilian satellite SCD1. In: International Astronautical Federation Congress, 44. (IAF), Oct 16-22 1993, Graz, Austria. **Proceedings**… [s.l.]: IAF,1993.

Larson, W.; **Space mission analysis and design**. 2a. Ed. Torrance, California: Wertz, J. Microcosm, Inc. and Kluwer Academic Publishers. 1996. 827 p. (ISBN 1-881883-01-9)

Maes, P. Artificial life meets entertainment: lifelike autonomous agents. **Communications of the ACM,** v.38, n.11, p. 108-114, Nov 1995.

McDermott, D.; Ghallab, M.; Howe, A.; Knoblock, C.; Ram, A.; Veloso, M.; Weld, D.; Wilkins, D. **PDDL –** the planning domain definition language. Yale University: Yale

Center for Computational Vision and Control. Oct 1998. (Technical Report n. CVC TR-98-003/DCS TR-1165).

McDermott, D. The 1998 AI Planning Systems Competition. **AI Magazine**, v.21, n. 2, p. 35 – 55, Summer 2000.

Monham, A.; Rudolph, A. Controlled approach to automated operations. In: International Conference on Space Operations, 8. (SPACEOPS), May 17 – 21 2004, Montreal, Quebec, Canada. **Proceedings** … Montreal, Canada: American Institute of Aeronautics and Astronautics, 2004.

Pednault, E. ADL: exploring the middle ground between STRIPS and the situation calculus. In: International Conference on Principles of Knowledge Representation and Reasoning, 1. (KR), May 15-18 1989, Toronto, Canada. **Proceedings**… San Francisco, CA: Morgan Kaufmann, 1989, p. 324-332. (ISBN 1-55860-032-9).

Penberthy, J.; Weld, D. UCPOP: a sound, complete, partial order planner for ADL. In: International Conference on Principles of Knowledge Representation and Reasoning, 3. (KR), Oct 25-29 1992, Cambridge, Massachusetts, USA. **Proceedings**… San Mateo, CA: Morgan Kaufmann, 1992, p. 103-114. (ISBN 1-55860-262-3).

Russell, S.; Norvig, P. **Inteligência artificial.** 2ª. Ed. Rio de Janeiro, RJ, Brasil: Elsevier Editora Ltda, 2004. 1021 p. (ISBN 85-352-1177-2).

Sherwood, R.; Chien, S.; Tran, D.; Cichy, B.; Castano, R.; Davies, A.;Rabideau, G. Operating the autonomous sciencecraft experiment. In: International Conference on Space Operations, 8. (SPACEOPS), May 17 – 21 2004, Montreal, Quebec, Canada. **Proceedings**… Montreal, Canada: American Institute of Aeronautics and Astronautics, 2004.

Sowa, J.. **Knowledge representation:** logical, philosophical, and computational foundations. Pacific Grove, CA: Brooks Cole Publishing Co, 2000. (ISBN 0-534-94965-7).

Wah, B.;Chen, Y. Subgoal partitioning and global search for solving temporal planning problems in mixed space. **International Journal of Artificial Intelligence Tools**. v. 13, n. 4, p. 767-790, Dec 2004. World Scientific Publishing Co. Pte. Ltd.

Weld, D. S. Recent Advances in AI Planning. **AI Magazine**, v. 20, n. 2, p. 93 -123, Oct 1999.

Wooldridge, M.; Jennings, N. Agent theories, architecture, and language: a survey. In: European Conference on Artificial Intelligence Workshop on Agent Theories, Architectures and Languages, 11. (ECAI), Aug 8 – 9 1995, Amsterdam, the Netherlands. **Proceedings**… Berlin: Springer-Verlag, 1995, p. 1-22.

GLOSSÁRIO

Ações: também chamadas operadores, são esquemas que representam regras de inferência não instanciadas. Uma ação é especificada em termos das precondições que devem ser válidas antes de ser possível executá-la e dos efeitos que resultam quando ela é executada. Para ações com duração, é necessário definir também a duração da execução da ação.

Arquivo de Domínio: arquivo gerado na linguagem PDDL que consiste da descrição de todos os elementos que são necessários na representação do conhecimento do domínio do problema de planejamento. Neste arquivo são definidos os tipos, as funções, os predicados e as ações que modelam o domínio.

Arquivo de Problema: arquivo gerado na linguagem PDDL que fornece o problema para o qual o planejador deve encontrar uma solução. Este arquivo é composto de duas partes principais:a situação inicial do ambiente (estado inicial) e o objetivo que se deseja atingir no plano.

Axioma: provérbio que encerra uma verdade indiscutível. Ele fornece informações factuais básicas a partir das quais podem ser derivadas conclusões úteis.

Domínio: é alguma parte do mundo real sobre o qual se deseja expressar algum conhecimento. Para o planejamento, o domínio é composto pela enumeração dos predicados e do conjunto das ações aplicáveis a um determinado ambiente.

Inferência: é o processo de extrair uma conclusão baseada somente no que já se conhece. Em IA, é a derivação de novas sentenças a partir de sentenças antigas.

Medida de distância: medida do tempo de propagação de um sinal eletromagnético emitido por um equipamento instalado, por exemplo, em uma estação terrena de rastreio, no percurso total de ida até um transponder instalado, por exemplo, em um satélite, e retorno até o local de onde foi emitido, após ser retransmitido de volta pelo transponder. Como o sinal eletromagnético se propaga a velocidade da luz, uma vez obtida a medida do tempo de propagação do sinal no percurso entre a estação terrena e o

satélite, em um dado instante, basta multiplicar o valor dessa medida pela velocidade da luz para transformá-lo no valor de uma medida da distancia da estação ao satélite, nesse instante.

Medida de velocidade: o processo de geração baseia-se na medição do desvio Doppler de freqüência sofrido por um sinal eletromagnético no percurso entre um transmissor e um receptor, que é proporcional à velocidade radial relativa entre o transmissor e o receptor. Esse tipo de medida é dito ser de um caminho, quando o transmissor se encontra no satélite e o receptor em solo, onde é feita a medição. Quando tanto o transmissor quanto o receptor se encontram em solo, e o satélite é equipado com um transponder que apenas recebe o sinal enviado de solo e o retransmite de volta (o que implica em uma medida de velocidade baseada na medição do desvio Doppler total sofrido pelo sinal nos percursos de subida ao satélite e retorno ao solo), a medida é dita ser de dois caminhos.

Ontologia: é uma especificação rigorosa de um conjunto de termos de vocabulário especializado e seus relacionamentos que seja suficiente para descrever e raciocinar sobre uma faixa de situações de interesse em algum domínio. Em outras palavras, é uma representação conceitual de entidades, eventos e seus relacionamentos que compõem um domínio específico.

Plano: é a seqüência ordenada de ações instanciadas por um planejador que ao ser executada permite a transição do estado inicial ao estado objetivo de um problema.

Planejamento automático: é a enumeração do conjunto de instâncias de ações aplicáveis ao domínio de um problema que permitem a transição do estado inicial (S_0) para o estado desejado e conhecido como estado objetivo (S_n) deste problema.

Telemetria: dados transmitidos pelo satélite para o solo que informam o estado dos seus subsistemas.

Telecomando: comandos remotos enviados de solo para os subsistemas do satélite.

APÊNDICE A
ARQUIVO DE DOMÍNIO CODIFICADO EM PDDL 2.2 PARA O DOMÍNIO DE RASTREIO DE SATÉLITE DO INPE

```
;; Domínio – Rastreio de satélite do INPE
;; Autora: Luciana Sêda Cardoso
(define (domain POVMEDZSRRTC)

(:requirements :equality :typing :fluents :durative-actions :timed-initial-literals)

(:types PassNumber SatCode GSCode CalibGroupNumber RangeGroupNumber
RangeNumber ZSNumber RRateGroupNumber TCCode)

(:predicates
    ;;Internal Events
    (TCSent ?pass - PassNumber ?sat - SatCode)
    (CalibrationOK ?pass - PassNumber ?cgroup - CalibGroupNumber)
    (FirstTcOK ?pass - PassNumber ?sat - SatCode ?tc - TCCode)
    ;;Initial Environment States
    (GSAvailable  ?gs - GSCode)
    (PassTrack ?pass - PassNumber ?sat - SatCode ?gs - GSCode)
    (NotPassTrack ?pass - PassNumber ?sat - SatCode ?gs - GSCode)
    (NotCalibrationAvailable ?pass - PassNumber  ?gs - GSCode)
    (RangeAvailable ?pass - PassNumber ?sat - SatCode ?gs - GSCode)
    (TCNotAvailable ?pass - PassNumber ?sat - SatCode)
    ;;Exogenous Events
    (FirstRange ?pass - PassNumber ?sat - SatCode ?gs - GSCode)
    (CalibrationAvailable ?pass - PassNumber  ?gs - GSCode)
    (RRateAvailable ?pass - PassNumber ?sat - SatCode ?gs - GSCode)
    (SatSilentZoneAvailable ?pass - PassNumber ?sat - SatCode ?gs - GSCode ?ZS -
ZSNumber)
```

```
        (FirstTc ?pass - PassNumber ?sat - SatCode ?tc - TCCode)
        (LastTC ?pass - PassNumber ?sat - SatCode ?tc - TCCode)
        (TimeTaggedTc  ?pass - PassNumber ?sat - SatCode ?tc - TCCode)
        ;;Goals
        (FirstRangeOk ?pass - PassNumber ?rgroup - RangeGroupNumber)
        (RangeOk ?pass - PassNumber ?rgroup - RangeGroupNumber ?rnum -
RangeNumber)
        (RRateOK ?pass - PassNumber ?sat - SatCode ?gs - GSCode ?RRGroup -
RRateGroupNumber)
        (TimeTaggedTCOK ?pass - PassNumber ?sat - SatCode ?tc - TCCode)
        (TCSendOK ?pass - PassNumber ?sat - SatCode ?tc - TCCode)
        (SilentZone ?pass - PassNumber ?sat - SatCode ?gs - GSCode ?ZS - ZSNumber)
        (PassEliminated ?pass - PassNumber ?sat - SatCode ?gs - GSCode)
        (LastTCOK ?pass - PassNumber ?sat - SatCode ?tc - TCCode)
 )

(:functions
         (CalibrationTime ?gs - GSCode ?cgroup - CalibGroupNumber)
         (RangeTime ?gs - GSCode ?rgroup - RangeGroupNumber)
         (PassTime ?pass - PassNumber)
         (ZoneSilentTime ?pass - PassNumber ?ZS - ZSNumber)
         (TcTime ?tc - TCCode)
 )

;; Eliminação de Passagem
(:durative-action PassEliminate
   :parameters (?pass - PassNumber ?sat - SatCode ?gs - GSCode)
   :duration (= ?duration (PassTime ?pass))
   :condition ( and (over all (NotPassTrack ?pass  ?sat ?gs))
                        (at start (GSAvailable  ?gs)) )
   :effect (and  (at end (PassEliminated ?pass  ?sat ?gs))
```

```
                    (at start (not (GSAvailable ?gs)))
                    (at end (GSAvailable ?gs)))
)

;; Calibração antes da passagem
(:durative-action Calibrate
  :parameters (?pass - PassNumber ?sat - SatCode ?gs - GSCode ?cgroup -
CalibGroupNumber)
  :duration (= ?duration (CalibrationTime ?gs ?cgroup))
  :condition ( and (at start (GSAvailable  ?gs))
                    (at start (CalibrationAvailable ?pass  ?gs))
                    (over all (PassTrack ?pass  ?sat ?gs))
                    (over all (RangeAvailable ?pass  ?sat  ?gs)) )
  :effect (and  (at end (CalibrationOK ?pass ?cgroup))
             (at start (not (GSAvailable ?gs)))
             (at end (GSAvailable ?gs))
             (at end (not (CalibrationAvailable ?pass  ?gs))))
)

;; Primeira Medida de Distância
(:durative-action FirstRange
  :parameters (?pass - PassNumber ?sat - SatCode ?gs - GSCode ?rgroup -
RangeGroupNumber ?cgroup - CalibGroupNumber)
  :duration (= ?duration (RangeTime ?gs ?rgroup))
  :condition ( and (at start (GSAvailable  ?gs))
                    (at start (FirstRange ?pass  ?sat  ?gs))
                    (over all (PassTrack ?pass  ?sat ?gs))
                    (over all (RangeAvailable ?pass  ?sat  ?gs))
                     (at start (or (CalibrationOK ?pass ?cgroup) (NotCalibrationAvailable
?pass  ?gs))) )
  :effect (and  (at end (FirstRangeOK ?pass ?rgroup))
```

```
            (at start (not (GSAvailable ?gs)))
            (at end (GSAvailable ?gs)))
)

;; Próxima Medida de Distância
(:durative-action Range
  :parameters (?pass - PassNumber ?sat - SatCode ?gs - GSCode ?rgroup -
RangeGroupNumber ?rnum - RangeNumber)
  :duration (= ?duration (RangeTime ?gs ?rgroup))
  :condition ( and (at start (GSAvailable  ?gs))
                   (over all (PassTrack ?pass  ?sat ?gs))
                   (over all (RangeAvailable ?pass  ?sat  ?gs))
                   (over all (FirstRangeOK ?pass ?rgroup)) )
  :effect (and  (at end (RangeOk ?pass ?rgroup ?rnum))
            (at start (not (GSAvailable ?gs)))
            (at end (GSAvailable ?gs)))
)

;; Esperar por Zona de Silêncio
(:durative-action WaitSilentZone
  :parameters (?pass - PassNumber ?sat - SatCode ?gs - GSCode ?ZS - ZSNumber )
  :duration (= ?duration (ZoneSilentTime ?pass ?ZS))
  :condition ( and (at start (GSAvailable  ?gs))
                   (at start (SatSilentZoneAvailable ?pass ?sat ?gs ?ZS)) )
  :effect (and  (at end (SilentZone ?pass ?sat ?gs ?ZS))
                (at start (not (GSAvailable ?gs)))
                (at end (GSAvailable ?gs)))
)

;; Medida de Velocidade
(:durative-action RangeRate
```

```
  :parameters (?pass - PassNumber ?sat - SatCode ?gs - GSCode ?RRgroup -
RRateGroupNumber)
  :duration (= ?duration 1.0)
  :condition ( and (at start (GSAvailable  ?gs))
                    (at start (RRateAvailable ?pass  ?sat  ?gs ))
                    (over all (PassTrack ?pass  ?sat ?gs)))
  :effect (and  (at end (RRateOK ?pass  ?sat  ?gs  ?RRGroup))
                 (at end (not(RRateAvailable ?pass  ?sat ?gs)))
                 (at start (not (GSAvailable ?gs)))
                 (at end (GSAvailable ?gs)))
)

;; Enviar Primerio Telecomando da passagem
(:durative-action SendFirstTC
  :parameters (?pass - PassNumber ?sat - SatCode ?tc - TCCode ?gs - GSCode ?cgroup
- CalibGroupNumber)
  :duration (= ?duration (TCTime ?tc))
  :condition ( and (at start (GSAvailable  ?gs))
                    (at start (FirstTc ?pass  ?sat  ?tc))
                    (over all (PassTrack ?pass  ?sat ?gs))
                    (over all (not (TCNotAvailable ?pass ?sat)))
                     (at start (or (CalibrationOK ?pass ?cgroup) (NotCalibrationAvailable
?pass  ?gs))) )
  :effect (and  (at end (FirstTcOK ?pass ?sat ?tc))
                 (at end (TcSent ?pass ?sat))
                 (at start (not (GSAvailable ?gs)))
            (at end (GSAvailable ?gs)))
)

;; Enviar Último Telecomando da passagem
(:durative-action SendLastTC
```

```
   :parameters (?pass - PassNumber ?sat - SatCode ?tc - TCCode ?gs - GSCode)
   :duration (= ?duration (TCTime ?tc))
   :condition ( and (at start (GSAvailable  ?gs))
                    (at start (LastTc ?pass  ?sat  ?tc))
                    (over all (PassTrack ?pass  ?sat ?gs))
                    (over all (TcSent ?pass ?sat)))
   :effect (and  (at end (LastTcOK ?pass ?sat ?tc))
                 (at start (not (GSAvailable ?gs)))
                 (at end (GSAvailable ?gs))
                 (at end (not (TCSent ?pass ?sat)))
                 (at end (not (PassTrack ?pass  ?sat ?gs))) )
)

;; Enviar Telecomando Temporizado em solo
(:durative-action SendTimeTaggedTC
   :parameters (?pass - PassNumber ?sat - SatCode ?tc - TCCode ?gs - GSCode)
   :duration (= ?duration (TCTime ?tc))
   :condition ( and (at start (GSAvailable  ?gs))
                    (at start (TimeTaggedTC ?pass  ?sat  ?tc))
                    (over all (PassTrack ?pass  ?sat ?gs)))
   :effect (and  (at end (TimeTaggedTCOK ?pass ?sat ?tc))
                 (at start (not (GSAvailable ?gs)))
                 (at end (GSAvailable ?gs)))
)

;; Enviar Telecomando não temporizado
(:durative-action SendTC
   :parameters (?pass - PassNumber ?sat - SatCode ?tc - TCCode ?gs - GSCode ?cgroup
- CalibGroupNumber)
   :duration (= ?duration (TCTime ?tc))
   :condition ( and (at start (GSAvailable  ?gs))
```

```
                    (over all (PassTrack ?pass  ?sat ?gs))
                    (over all (TCSent ?pass ?sat))
                     (at start (or (CalibrationOK ?pass ?cgroup) (NotCalibrationAvailable
?pass  ?gs))))
  :effect (and  (at end (TCSendOK ?pass ?sat ?tc))
              (at start (not (GSAvailable ?gs)))
              (at end (GSAvailable ?gs)))
)
)
```

APÊNDICE B
EXEMPLO DE UM ARQUIVO DE PROBLEMAS CODIFICADO EM PDDL 2.2 GERADO AUTOMATICAMENTE PELO PROTÓTIPO

O conteúdo do arquivo de problemas que foi gerado automaticamente pelo protótipo para a passagem 60509.4 do satélite SCD1 e estação terrena de Cuiabá está apresentado a seguir. Em negrito estão feitas algumas observações para melhor entendimento que não fazem parte do código original.

;;Arquivo de problema para a passagem:SCD1CBAProblemPass60509-4
(define (problem SCD1CBAProblemPass60509-4)
(:domain POVMEDZSRRTC) ? ? **Nome do arquivo de domínio**
;;Definição dos Objetos
(:objects ? ? **Objetos instanciados dos tipos definidos no arquivo de domínio**
PASS60509-4 - PassNumber
SCD1 - SatCode
CBA - GSCode
CGroup03 - CalibGroupNumber
RGroup03 - RangeGroupNumber
Range02 - RangeNumber
Range03 - RangeNumber
Range04 - RangeNumber
Range05 - RangeNumber
ZS01 - ZSNumber
RRGroup07 - RRateGroupNumber
TC137 - TCCode
TC118 - TCCode
TC138 - TCCode
TT140 - TCCode
TT150 - TCCode
)

;;Definição das Funções ? ? **Os valores aplicados a cada função foram calculados**

(:init

;;

(= (CalibrationTime CBA CGroup03) 90.0)

(= (PassTime PASS60509-4) 720.0)

(= (RangeTime CBA RGroup03) 90.0)

(= (TcTime TC137) 2.0)

(= (TcTime TC118) 2.0)

(= (TcTime TC138) 2.0)

(= (TcTime TT140) 3.0)

(= (TcTime TT150) 2.0)

(= (ZoneSilentTime PASS60509-4 ZS01) 30.0)

;;

;;Definição do Estado inicial do ambiente ? ? **Estado inicial da passagem**

;;

(GSAvailable CBA) ? ? **Estação terrena CBA disponível para rastreio**

(PassTrack PASS60509-4 SCD1 CBA) ? ? **Passagem 60509-4 a ser rastreada**

(RangeAvailable PASS60509-4 SCD1 CBA) ? ? **Méd. de distância disponível nesta passagem**

;;

;;Definição do Eventos exógenos ? ? **Definição dos eventos com tempo predeterminados da passagem**

;;

(at 0.0 (CalibrationAvailable PASS60509-4 CBA)) ? ? **Calibração**

(at 90.0 (not (CalibrationAvailable PASS60509-4 CBA)))

;;

(at 480.0 (FirstRange PASS60509-4 SCD1 CBA)) ? ? **1a. Med. de distância**

(at 570.0 (not (FirstRange PASS60509-4 SCD1 CBA)))

;;

(at 390.0 (FirstTc PASS60509-4 SCD1 TC137)) ? ? **1o. telecomando**

(at 392.0 (not (FirstTc PASS60509-4 SCD1 TC137)))

;;

(at 1050.0 (LastTC PASS60509-4 SCD1 TC138)) ? ? **último telecomando**

(at 1052.0 (not (LastTC PASS60509-4 SCD1 TC138)))

;;

(at 420.0 (RRateAvailable PASS60509-4 SCD1 CBA)) ? ? **Med. de velocidade**

(at 421.0 (not (RRateAvailable PASS60509-4 SCD1 CBA)))

;;

(at 870.0 (SatSilentZoneAvailable PASS60509-4 SCD1 CBA ZS01)) ? ? **Zona de silêncio**

(at 900.0 (not (SatSilentZoneAvailable PASS60509-4 SCD1 CBA ZS01)))

)

;;

;;Definição Objetivos a serem atingidos no Plano

;;

(:goal (and ? ? **Objetivo composto de sub-objetivos**

(LastTCOK PASS60509-4 SCD1 TC138)

(RRateOK PASS60509-4 SCD1 CBA RRGroup07)

(RangeOk PASS60509-4 RGroup03 Range02)

(RangeOk PASS60509-4 RGroup03 Range03)

(RangeOk PASS60509-4 RGroup03 Range04)

(RangeOk PASS60509-4 RGroup03 Range05)

(SilentZone PASS60509-4 SCD1 CBA ZS01)

)))