MINISTÉRIO DA CIÊNCIA E TECNOLOGIA
INSTITUTO NACIONAL DE PESQUISAS ESPACIAIS

INPE-16715-TDI/1653

SIMULADOR DE SATÉLITES PARA VERIFICAÇÃO DE PLANOS DE OPERAÇÕES EM VOO

Jun Tominaga

Dissertação de Mestrado do Curso de Pós-Graduação em Computação Aplicada, orientada pelos Drs. José Demisio Simões da Silva, e Mauricio Gonçalves Vieira Ferreira, aprovada em 16 de abril de 2010.

URL do documento original:
<http://urlib.net/8JMKD3MGP7W/37HL3J8>

INPE
São José dos Campos
2010

PUBLICADO POR:

Instituto Nacional de Pesquisas Espaciais - INPE
Gabinete do Diretor (GB)
Serviço de Informação e Documentação (SID)
Caixa Postal 515 - CEP 12.245-970
São José dos Campos - SP - Brasil
Tel.:(012) 3208-6923/6921
Fax: (012) 3208-6919
E-mail: pubtc@sid.inpe.br

MINISTÉRIO DA CIÊNCIA E TECNOLOGIA
INSTITUTO NACIONAL DE PESQUISAS ESPACIAIS

INPE-16715-TDI/1653

SIMULADOR DE SATÉLITES PARA VERIFICAÇÃO DE PLANOS DE OPERAÇÕES EM VOO

Jun Tominaga

Dissertação de Mestrado do Curso de Pós-Graduação em Computação Aplicada, orientada pelos Drs. José Demisio Simões da Silva, e Mauricio Gonçalves Vieira Ferreira, aprovada em 16 de abril de 2010.

URL do documento original:
<http://urlib.net/8JMKD3MGP7W/37HL3J8>

INPE
São José dos Campos
2010

Dados Internacionais de Catalogação na Publicação (CIP)

Tominaga, Jun.

T595si Simulador de satélites para verificação de planos de operações em voo / Jun Tominaga. – São José dos Campos : INPE, 2010.

xx + 154 p. ; (INPE-16715-TDI/1653)

Dissertação (Mestrado em Computação Aplicada) – Instituto Nacional de Pesquisas Espaciais, São José dos Campos, 2010.

Orientadores : Drs. José Demisio Simões da Silva, e Mauricio Gonçalves Vieira Ferreira.

1. Simuladores. 2. Satélites artificiais. 3. Operações em voo. 4. Inteligência artificial. 5. Representação de conhecimento. 6. Sistemas especialistas. 7. Sistemas nebulosos. I.Título.

CDU 004.383.4:629.783

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

Computação Aplicada

Dr. **José Carlos Becceneri**

Presidente / INPE / SJCampos - SP

Dr. **José Demisio Simões da Silva**

Orientador(a) / INPE / SJCampos - SP

Dr. **Mauricio Gonçalves Vieira Ferreira**

Orientador(a) / INPE / SJCampos - SP

Dr. **Stephan Stephany**

Membro da Banca / INPE / SJCampos - SP

Dr. **Edgar Toshiro Yano**

Convidado(a) / ITA / São José dos Campos - SP

Aluno (a): Jun Tominaga

São José dos Campos, 16 de abril de 2010

AGRADECIMENTOS

Agradeço a todos que tiveram paciência e boa vontade, que colaboraram de alguma forma para a confecção e conclusão deste trabalho. Em especial, gostaria de registrar a importância das discussões sobre arquiteturas de simuladores de satélites que tive com Ana Maria Ambrósio, e seu convite para participar do workshop para estudos sobre simuladores de satélites, organizado por ela. Mas, principalmente, agradeço ao colega e orientador Maurício Gonçalves Vieira Ferreira, pelas inúmeras reuniões, discussões, revisões, sugestões e críticas, sempre construtivas e pertinentes, bem como pelos diversos encontros que organizou para apresentações e discussões de trabalhos desenvolvidos dentro do projeto de automação do sistema de planejamento de operações de satélites. Estas contribuições foram de vital importância para a elaboração deste trabalho e, sem a sua orientação, ele não teria sido concluído.

RESUMO

Este trabalho faz parte de um projeto de automação de planejamento de operações de satélites em desenvolvimento no Centro de Controle de Satélites do Instituto Nacional de Pesquisas Espaciais. Este projeto tem como objetivo adequar o Centro de Controle de Satélites ao aumento do número de satélites a controlar, conforme planejado pela direção do instituto. Entretanto, há uma resistência justificável por parte de especialistas em planejamento de operações quanto à automação de todo o processo, o que torna o sistema susceptível a erros de software. Para resolver este problema, propõe-se a inclusão de simuladores de satélites para a verificação de planos de operação em vôo, configurados e mantidos por estes próprios especialistas. Neste trabalho, serão descritos o problema do planejamento de operações em vôo de satélites, a motivação que levou à necessidade de elaboração de um simulador de satélite para a verificação de planos, uma pesquisa referente a usos e arquiteturas de simuladores de satélites, a justificativa de uma escolha de uma arquitetura altamente configurável para atender às necessidades do usuário, a descrição de sua implementação em software e, por fim, possibilidades de trabalhos futuros vislumbrados a serem incorporados ao projeto.

SATELLITES SIMULATOR FOR VERIFICATION OF FLIGHT OPERATIONS PLANS

ABSTRACT

This work is part of a satellite operations planning automation project under development at the *Centro de Controle de Satélites* (Satellite Control Center) of the *Instituto Nacional de Pesquisas Espaciais* (Brazilian National Institute for Space Research). This project aims at adapting the *Centro de Controle de Satélites* to the increase of the number of satellites to be controlled, as planned by the institute direction. However, there is justifiable resistance from operations planning experts regarding the automation of the whole process, which makes the system susceptible to software errors. To solve this problem, the inclusion of satellite simulators configured and maintained by these experts themselves, are proposed to verify flight operations plans. In this work shall be described: the satellites flight operations planning problem, the motivation leading to the need for the elaboration of a satellite simulator for verification of plans, research about uses; architectures of satellite simulators, the justification for choosing an highly configurable architecture to satisfy user needs, the description of its software implementation and, finally, possibilities for future works to be incorporated into this work.

LISTA DE FIGURAS

Pág.

LISTA DE TABELAS

Pág.

LISTA DE SIGLAS E ABREVIATURAS

AIT	*Assembly, Integration and Tests* (Montagem, Integração e Testes)
CBERS	*China Brazil Earth Resources Satellite* (Satélite Sino-Brasileiro de Recursos Terrestres)
CRC	Centro de Rastreio e Controle de Satélites
CSV	*Comma Separated Values* (Valores Separados por Vírgulas)
DJM	Data Juliana Modificada
DOD	*Depth of Discharge* (Profundidade de Descarga)
ECL	Previsão de Eclipses
ECSS	*European Cooperation on Space Standardization* (Cooperação Européia para Padronização Espacial)
ERMAC	Encontro Regional de Matemática Aplicada e Computacional
ESA	*European Space Agency* (Agência Espacial Européia)
FMEA	*Failure Modes and Effects Analysis* (Análise de Modos de Falha e Efeitos)
FMECA	*Failure Modes, Effects and Criticality Analysis* (Análise de Modos de Falha, Efeitos e Criticidade)
INPE	Instituto Nacional de Pesquisas Espaciais
POV	Plano de Operações em Vôo
PVP	Previsão de Passagens, equivalente a SCP
SCD	Satélite de Coleta de Dados
SCP	*Satellite Contact Prediction* (Predição de Contato de Satélite), equivalente a PVP
SPACEOPS	*International Committee on Technical Interchange for Space Mission Operations and Ground Data Systems* (Comitê Internacional em Intercâmbio Técnico para Operações de Missões Espaciais e Sistemas de Dados de Solo)
TT&C	*Telemetry, Trackingand Commanding* (Telemetria, Rastreio e Comando) ou, alternativamente, *Telemetry, Tracking and Control* (Telemetria, Rastreio e Controle)
W3C	*World Wide Web Consortium* (Consórcio *World Wide Web*, ou Consórcio Rede Mundial Ampla)
WORCAP	Workshop do Curso de Computação Aplicada
XML	*Extended Markup Language*

SUMÁRIO

Pág.

1 INTRODUÇÃO

Desde 1993, o Centro de Rastreio e Controle de Satélites (CRC) tem executado operações de controle de satélites desenvolvidos total ou parcialmente pelo Instituto Nacional de Pesquisas Espaciais (INPE). Estas operações de controle são executadas por meio de atividades de controle. Atividades de controle rotineiras podem ser automatizadas para se melhorar a eficiência. No CRC, planejamentos de operações envolvendo atividades rotineiras mais simples encontram-se automatizados. Porém, a tendência atual é de aumento, tanto da quantidade de satélites controlados, quanto da complexidade das operações de missões. Atualmente, o CRC conta com um centro de controle, responsável pelo planejamento e execução de operações de controle, e duas estações terrenas, equipadas com antenas de rastreio a partir do qual se controlam três satélites. O mais antigo destes satélites, o Satélite de Coleta de Dados Um (SCD1), foi lançado em 1993 e ainda permanece operacional. Caso o planejamento de lançamentos futuros ilustrado na figura a seguir se concretize sem perdas de missões antigas, este número poderá vir a quintuplicar em menos de dez anos.

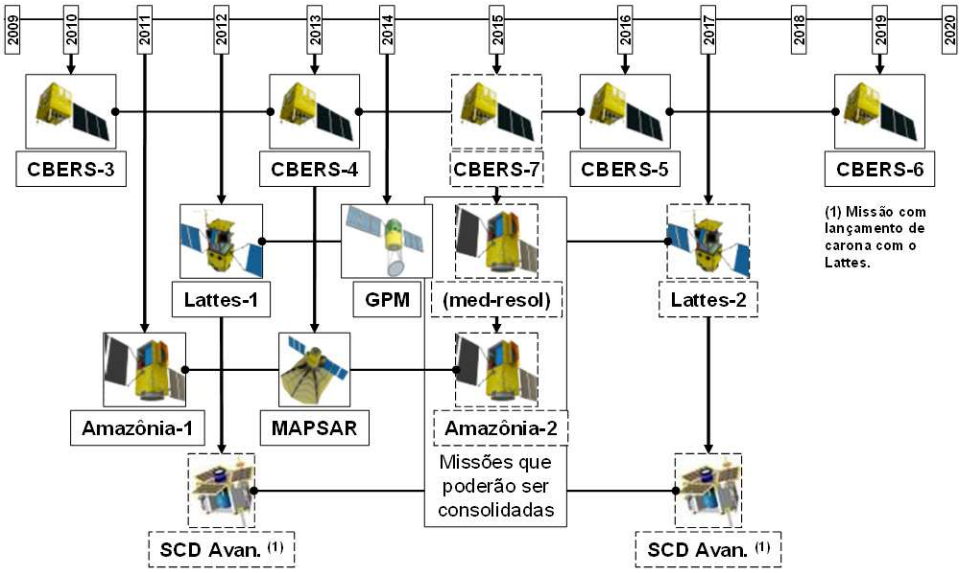


Figura 1.1 - Cronograma de novos lançamentos
Fonte: Adaptado de Câmara (2008)

Devido à preocupação quanto à possibilidade do sistema atual não conseguir gerenciar satisfatoriamente missões futuras, há um projeto em andamento que propõe a automação de planejamentos de operações mais complexas. Porém, especialistas de planejamentos de operações têm ressalvas quanto à adoção de produtos derivados deste projeto. Um simples erro no planejamento de operações pode levar à execução de procedimentos falhos, causando danos irreversíveis em satélites que podem comprometer os seus funcionamentos e os investimentos em suas missões.

Assim, propõe-se uma ferramenta de validação de planejadores por meio de verificação de planos de operações em vôo. Isto pode ser realizado por meio de um simulador de estados internos de satélites. Uma ferramenta deste tipo é capaz de permitir o diagnóstico de anomalias causadas por planos de operações em vôo falhos, através da geração de um histórico de valores de parâmetros simulados que compõem estes estados internos. Desta forma, impede-se a execução de planos de operações que possam vir a danificar o satélite.

Neste trabalho, foram levantados os requisitos para a elaboração de um simulador de satélites como ferramenta para verificação de planos de operações em vôo. Primeiramente, analisou-se o sistema de planejamento de operações atualmente implementado no INPE, para se definirem o ambiente e o escopo de funcionamento do simulador. O sistema de planejamento de operações requer constante adequação conforme a situação corrente do satélite, característica esta que deve também fazer parte do simulador. Em seguida, realizou-se um estudo acerca de simuladores de satélites, a fim de se encontrar uma arquitetura apropriada para a finalidade proposta. Após a conclusão de que simuladores existentes não atendem aos interesses do CRC, é elaborada uma nova arquitetura para um simulador voltado especificamente para a finalidade proposta. As funcionalidades requeridas para este novo simulador são explicitadas através de um cenário típico de planejamento de

operações. Em seguida, são propostas estruturas de dados para o armazenamento de modelos configuráveis do simulador, constituídas essencialmente por regras causais. A partir das especificações do modelo baseado em regras, torna-se possível implementar um simulador capaz de interpretar as estruturas de dados armazenadas em bancos de dados, e processá-las conforme as regras de dinâmica do modelo. Este simulador pode se tornar plenamente operacional após a implantação de ferramentas de edição de seus modelos.

No próximo Capítulo será apresentado um breve resumo do planejamento de operações, objetivando permitir uma melhor compreensão do problema a ser resolvido, possibilitando assim a implementação de uma solução.

2 PLANEJAMENTO DE OPERAÇÕES

Neste Capítulo, o planejamento de operações será descrito brevemente, em um nível de detalhamento suficiente para dar um melhor entendimento das necessidades e dos objetivos do verificador de planos proposto neste trabalho.

2.1. Operações de controle de satélites

Operações de controle de satélites incluem dinâmica de vôo, planejamento de operações, e execução de procedimentos. Estas operações são realizadas periodicamente, de forma cíclica, separadamente e independentemente para cada satélite. Esta relação entre operações de controle de satélites é ilustrada na figura abaixo.

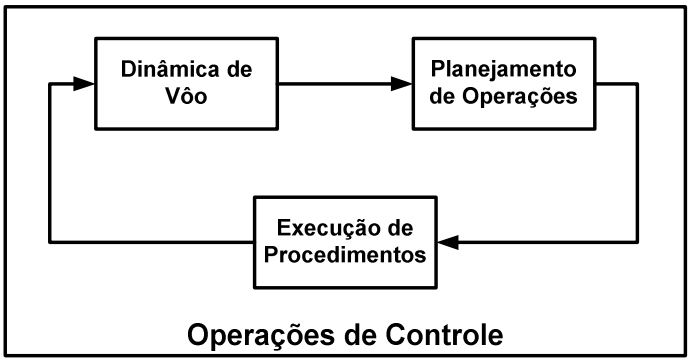


Figura 2.1 - Ciclo de operações de controle

A dinâmica de vôo é responsável pela determinação e propagação de informações referentes a posicionamento orbital e apontamento de atitude de satélites. Ela utiliza medidas de rastreio e calibração, obtidas por meio de execução de procedimentos em passagem, para determinação e propagação de parâmetros orbitais e de atitude.

O planejamento de operações agenda tarefas que devem ser desempenhadas para possibilitar a execução de operações de missão para cada satélite. Este

processo é realizado em cima de previsões de contato entre o satélite e antenas de controle, fornecidos pela dinâmica de vôo.

A execução de procedimentos consiste na realização, em tempo real, de tarefas agendadas conforme o planejamento de operações. É através desta execução que equipamentos de carga-útil dos satélites são ligados para a obtenção de dados missão, e medidas de posição e apontamento são obtidas para serem processadas pela dinâmica de vôo.

Antenas de controle equipam estações terrenas de telemetria, rastreio e controle (TT&C), conhecidas também como estações de rastreio. Estas antenas são apontadas para satélites de acordo com previsões de apontamento geradas pela dinâmica de vôo. Isto é feito durante janelas de tempo, conhecidas como passagens, dentro do qual é possível a comunicação entre satélites e antenas de controle. Com a troca de sinais de rádio entre estação e satélite, estabelece-se o enlace de comunicação, que compreende um enlace de subida, ativo, e um enlace de descida, passivo. O estabelecimento destes enlaces é necessário para a execução de procedimentos em tempo real. O enlace de descida possibilita a recepção de telemetrias, ao passo que o enlace de subida permite o envio de telecomandos e disparo de medidas de rastreio. Procedimentos são executados seqüencialmente, conforme agendados num plano de operações em vôo.

A geração de planos de operações em vôo constitui o objetivo principal do planejamento de operações. A fim de se realizar o planejamento de operações, é necessário conhecer com precisão a posição e a trajetória do satélite. Isto é realizado pela dinâmica de vôo, a partir de medidas de rastreio. A partir destas informações, é possível calcular os ângulos de apontamento de antenas de controle.

2.2. Atividades de controle e modos de operação

Atividades de controle de satélites são específicas para cada satélite e mudam, mesmo para a execução de uma mesma operação de controle, conforme o modo de operação em que se encontra o satélite. Elas se dividem primeiramente entre atividades planejadas e atividades emergenciais. Atividades planejadas englobam tarefas executadas segundo diretrizes estabelecidas por requisitos de operações. Já atividades emergenciais incluem tarefas cujas execuções não são planejadas a priori, que devem ser executadas conforme a necessidade.

Entre as atividades planejadas, destacam-se aquelas que são executadas periodicamente, repetindo um padrão cíclico num curto intervalo de tempo. Estas atividades são denominadas atividades rotineiras. Em condições normais de operação, a maior parte das atividades de controle de satélites é composta por atividades rotineiras.

Atividades planejadas especiais são elaboradas por projetistas de satélite ou especialistas de sistema, a partir de análise de estado corrente do satélite. Estas atividades têm como finalidade ou manter parâmetros controláveis dentro da faixa nominal de operação, ou levar o satélite de um modo degradado para um modo de operação normal. A execução bem-sucedida de atividades especiais leva o satélite a um modo de operação em rotina, regida por atividades rotineiras. Já uma execução mal-sucedida pode levar o satélite a um modo de operação emergencial.

Atividades emergenciais são causadas por alguma anomalia, sendo normalmente resultantes de uma ou mais falhas. Procedimentos de atividades emergenciais são elaborados e documentados por projetistas de equipamentos embarcados com base em previsões de falhas. Estas atividades têm como finalidade conter uma maior degradação do satélite e levá-lo rapidamente a um

modo seguro de operação. A execução de atividades emergenciais leva invariavelmente à elaboração de atividades planejadas especiais.

Para uma facilitar a compreensão da classificação das atividades de controle, a figura abaixo ilustra como estas atividades se subdividem.

Figura 2.2 - Classificação de atividades de controle

A figura a seguir explicita ações e eventos inerentes à execução de determinadas atividades de controle e aquelas que disparam transições. Tais ações e eventos caracterizam o que se denominam modos de operação.

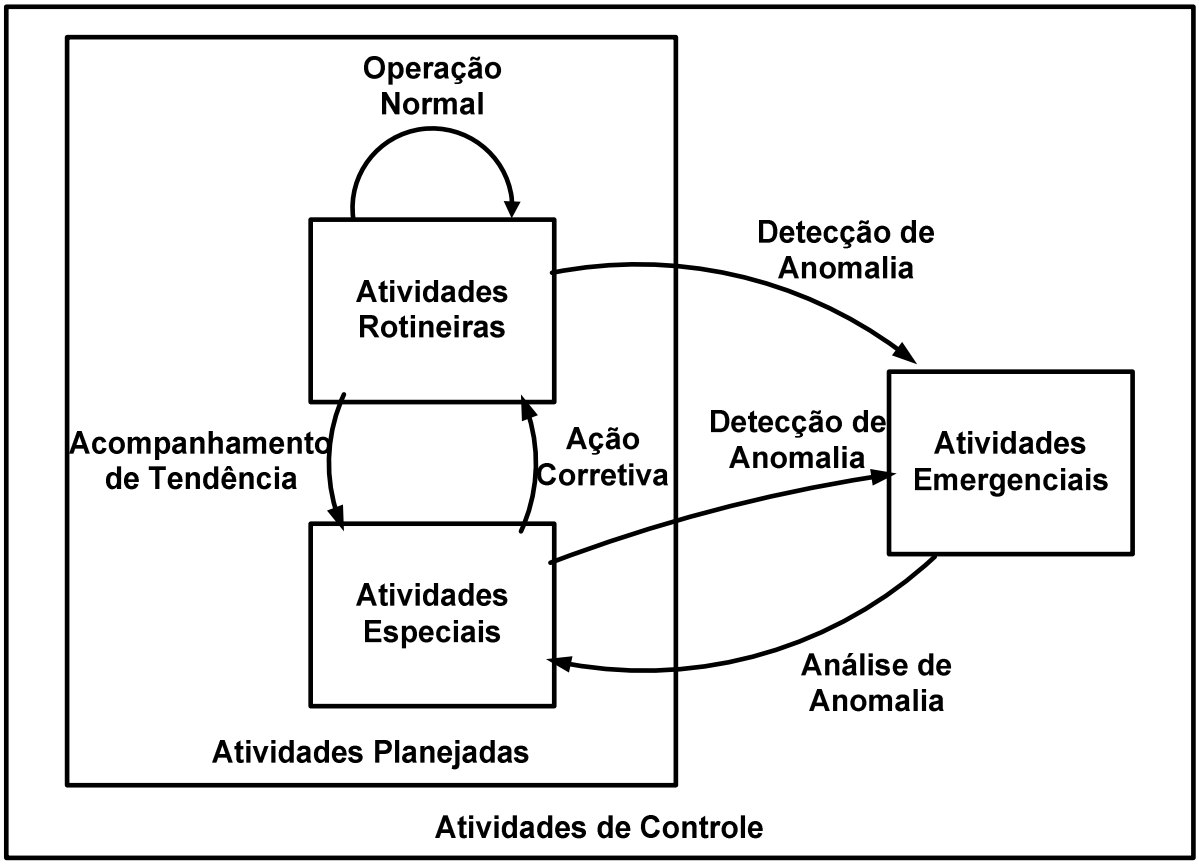


Figura 2.3 - Diagrama de transição de atividades de controle

Cada satélite possui seus próprios modos de operação, diferenciados conforme especificidades de cada missão. Porém, de forma geral, todos os modos de operação podem-se enquadrar dentro de quatro categorias básicas. A figura abaixo resume o relacionamento entre as atividades de controle e os modos de operação a que se aplicam.

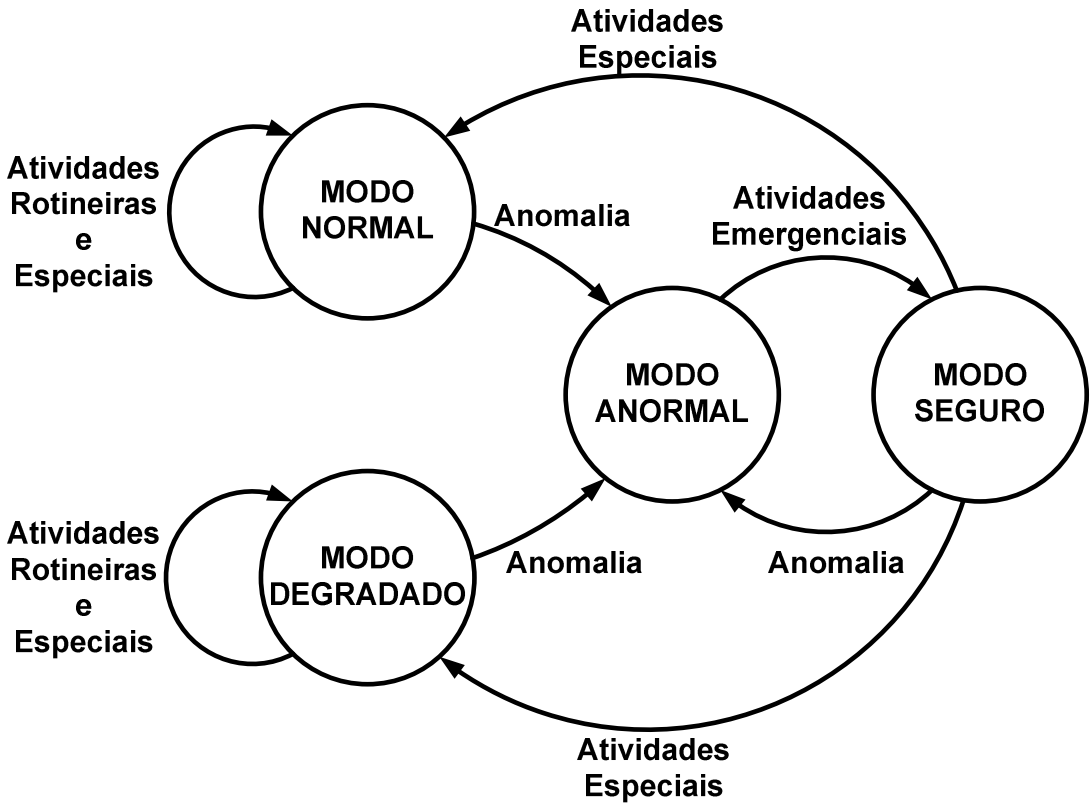


Figura 2.4 - Diagrama de transição de modos de operação

No modo de operação normal, tem-se uma situação estável, dentro do qual o satélite é operado plenamente conforme as especificações de projeto da missão. Neste modo, executam-se operações de controle que são compostos majoritariamente por atividades rotineiras. Exemplos incluem programação e execução de medidas de rastreio, recepção de telemetrias, e telecomandos para ativação e desativação de carga-útil de acordo com visibilidade de estações terrenas de recepção de dados. Atividades especiais são também ocasionalmente executadas, conforme parâmetros sob monitoração aproximam-se de seus valores limite. Exemplos incluem programação e execução de correção de tempo de computador embarcado, e manobras de correção orbital ou de atitude.

O modo de operação anormal é uma situação instável, causada por uma ou mais anomalias em bordo. A entrada e permanência neste modo constituem

uma situação indesejável que, caso não seja corrigida em tempo, pode pôr em risco a saúde do satélite. Portanto, assim que a execução de procedimentos de monitoração de telemetria identifica a entrada do satélite neste modo, atividades emergenciais devem ser disparadas para levar o satélite a um modo de operação seguro.

O modo de operação seguro consiste em uma situação estável alcançada após a execução de atividades emergenciais. Neste modo, o satélite é mantido vivo, porém inoperante. Em outras palavras, somente são mantidos ligados equipamentos embarcados com funções de manutenção de plataforma em vôo. Equipamentos de carga-útil, para desempenho de funções de missão, são mantidos desligados. Enquanto o satélite encontra-se neste modo, realizam-se análises para se buscar as causas das anomalias que trouxeram o satélite para o modo anormal, e buscam-se soluções para levá-lo de volta ao modo normal. Ou então, caso o retorno ao modo normal seja comprovadamente impossível, tenta-se encontrar um modo degradado em que o satélite possa operar.

O modo de operação degradado é outra situação estável, em que as cargas-úteis do satélite são operadas, porém sem satisfazer inteiramente as especificações de projeto da missão. O satélite entra neste modo após uma ou mais anomalias terem causado dano permanente em um ou mais equipamentos em bordo. Neste modo, atividades de controle são desempenhadas de forma semelhante ao que ocorre no modo normal. Porém, atualizam-se atividades de controle para refletir as alterações dos requisitos de missão. Por exemplo, a perda de um equipamento de carga-útil sem redundância leva necessariamente ao cancelamento de operações que dependem dele. Dependendo do caso, uma falha deste tipo pode ocasionar um uso mais agressivo das cargas-úteis restantes, devido ao acúmulo de energia excedente em bordo, ou na tentativa de tirar máximo proveito da vida remanescente do satélite antes que se alcance seu fim.

2.3. Planejamento em rotina

Planejamentos em rotina, ou planejamentos rotineiros, constituem uma categoria de operações de controle de satélites, sendo composta por planejamentos de operações contendo atividades rotineiras de controle de satélites, em modo de operação normal ou degradado. No CRC, planejamentos em rotina vêm sendo realizados via software, de forma automatizada.

No planejamento em rotina, primeiramente são obtidos da dinâmica de vôo arquivos de previsões de contato de satélites (SCP), também conhecidos como previsões de passagens (PVP). Estes arquivos contêm uma lista seqüencial de passagens, correspondente a cada satélite para cada estação terrena de TT&C, dentro de um intervalo de tempo futuro estabelecido.

Em seguida, conflitos entre passagens nas previsões de contato de satélites são gerenciados. Neste processo, são eliminadas superposições temporais e espaciais entre passagens. Uma estação terrena de TT&C não deve realizar dois rastreios simultâneos com uma mesma antena. E um mesmo satélite não deve estabelecer enlace de subida simultaneamente com mais de uma estação terrena de TT&C. Atualmente, o gerenciamento de conflitos é realizado por meio de alocação de prioridades a previsões de contato de satélites, e cortes de passagens consideradas de baixa prioridade. Estes cortes podem ser realizados parcialmente, que resultam em passagens encurtadas.

Após o gerenciamento de conflitos, as passagens remanescentes em cada previsão de contato de satélite são divididas em unidades de tempo, às quais podem se atribuir atividades de controle planejadas. Este passo constitui a geração do plano de operações em vôo (POV). Atividades de rotina são agendadas automaticamente via software de plano de operações em vôo, ao passo que atividades especiais devem ser inseridas manualmente.

O preenchimento de lacunas de tempo é realizado conforme requisitos de operação pré-estabelecidos. Estes requisitos de operação encontram-se codificados na forma de padrões de operações de satélites. Estes padrões são ocasionalmente atualizados, conforme alterações de modos de operação sofridas pelos satélites. Os padrões de operações de satélites objetivam primariamente o uso de recursos embarcados de maneira segura, não havendo preocupação com sua otimização.

A configuração de padrões de operação de satélites, conforme a necessidade, e o fornecimento de arquivos de previsões de passagens, contendo previsões de contato de satélites, são realizados pela equipe de dinâmica de vôo. Os padrões de operação de satélites são traduzidos em atividades rotineiras. Conflitos entre arquivos de previsões de passagens são resolvidos por meio de atribuição de uma escala de prioridades a estes arquivos e encurtamento de passagens de baixa prioridade. A partir da alocação de atividades às passagens, obtêm-se os planos de operação em vôo, que são submetidos à execução de procedimento.

2.4. Histórico e perspectivas

A primeira versão do software de geração dos planos de operações em vôo em uso no CRC foi desenvolvida pelo gerente de dinâmica de vôo, na década de 1990, para satélite SCD1. Com o tempo, versões específicas foram sendo criadas, na medida em que novos satélites foram sendo lançados. Atualmente, a responsabilidade por manter seu código-fonte encontra-se com a equipe de geração de previsões da dinâmica de vôo. Esta equipe é a mesma responsável pelas previsões de contato de satélites e, portanto, detém controle total sobre as atividades de controle do planejamento de operações. Em caso de mudança do modo de operação de um satélite, o código-fonte do gerador de planos de operações em vôo é alterado por esta equipe, por meio de inserção de novos padrões de operações de satélites

Atualmente, a maior parte dos os requisitos de operação para a codificação dos padrões de operações de satélites é constituída por disparos de medidas de rastreio e telecomandos para operação de carga-útil. Medidas de rastreio devem ser disparadas durante passagens, que são determinadas pela dinâmica de vôo. Para os satélites controlados atualmente pelo CRC, requisitos de operação têm estabelecido ou a operação permanente de equipamentos de carga-útil, ou então, seu ligamento no início de passagens sobre uma estação terrena de recepção de dados de carga-útil, e desligamento ao fim destas passagens. Desta forma, há uma forte correlação entre a dinâmica de vôo e o planejamento de operações.

Novos acordos para projetos de satélites indicam um aumento expressivo no número de missões a serem controladas pelo CRC. O CRC opera atualmente três satélites, e recebe dados de carga-útil de um quarto satélite, por meio de duas estações terrenas de TT&C. A não ser que haja perdas sucessivas de satélites, prevê-se que o número de satélites cresça exponencialmente nos próximos anos. A aquisição de uma nova estação terrena de TT&C está prevista para se atender esta demanda. Esta estação deve ser uma estação transportável, a ser mantida normalmente em princípio na região amazônica. (AEB, 2005)

O sistema atual de planejamento de operações é adequado para as operações de rotina das missões controladas, tendo sido validado através de anos de experiência e de uso continuado. O sistema atual permite o gerenciamento de conflito de até vinte previsões de contato de satélites. Esta é uma característica inerente do sistema legado. Este sistema permite o planejamento de operações de dez satélites por duas antenas de controle, ou de seis satélites por três antenas de controle. Porém, espera-se um aumento na complexidade dos requisitos de operações das novas missões. Acordos de cooperação internacionais estão sendo firmados para novos satélites, de forma a dividir tanto custos de desenvolvimento quanto retornos de investimento. Centros de

missão procuram atender a demandas de um número crescente de usuários, ao passo que usos abusivos dos equipamentos embarcados devem ser evitados. Novas tecnologias adicionam cada vez mais recursos aos equipamentos de carga-útil, o que contribui para o aumento de complexidade em suas operações. Os requisitos operacionais de satélites evidenciam a complexidade crescente das novas missões, comparadas a seus antecessores. (AEB, 2005) (CAST; INPE, 2007) (INPE, 1988) (INPE, 2008) (XSCC, 2000)

Para resolver este problema, encontra-se atualmente em desenvolvimento um projeto objetivando a automação do planejamento de operações no CRC. Entretanto, devido a restrições administrativas, políticas e financeiras, o projeto em andamento para a modernização do sistema de controle de satélites tem um forte caráter acadêmico. Entre os trabalhos incluídos neste projeto há planejadores de operações rotineiras, que utilizam técnicas de inteligência artificial, cujo objetivo é substituir o sistema legado gerado e operado pela dinâmica de vôo do CRC. (BIANCHO et al., 2006) (GONÇALVES, 2006) (CARDOSO et al., 2006)

Do ponto de vista da dinâmica de vôo do CRC, há uma forte resistência contra a substituição de seu sistema legado, de desenvolvimento próprio e validado por décadas de uso contínuo, por caixas-pretas de forte caráter experimental e de confiabilidade duvidosa. Para compatibilizar o uso dos novos planejadores com as preocupações do CRC, propõe-se uma ferramenta de verificação de planos de operações em vôo. Esta ferramenta deve ser capaz de permitir a identificação e a rejeição de planos de operações em vôo que possam por em risco a saúde dos satélites controlados.

Uma abordagem apropriada para a elaboração desta ferramenta consiste em desenvolver simuladores de satélites, construídos com o intuito de simular o comportamento de satélites reais após a execução dos procedimentos inclusos nos planos de operações em vôo sob teste. Caso uma simulação revele

situações consideradas perigosas para um satélite, o plano de operações sob teste que levou a este resultado é rejeitado. Caso contrário, este plano é considerado seguro, podendo ser encaminhado para a execução de procedimentos.

Desta forma, no próximo Capítulo será apresentado um estudo sobre simuladores de satélites, com o intuito de identificar as características necessárias e desejáveis para a elaboração de uma ferramenta de teste para planos de operações em vôo.

3 SIMULADORES DE SATÉLITES

Este trabalho propõe o uso de uma ferramenta baseada em simuladores de satélites para verificação de planos de operações em vôo. Neste Capítulo, será apresentado um breve estudo realizado acerca de simuladores de satélites, objetivando uma melhor compreensão de suas aplicações, técnicas de construção e processamento interno de parâmetros. Com base nestas informações, serão levantados requisitos que auxiliarão na escolha da arquitetura mais adequada para um simulador de satélite para verificação de plano de operações.

3.1. Aplicações e usos de simuladores

O INPE tem experiência em desenvolvimento e especificação de simuladores de satélites. Historicamente, o foco destes simuladores tem sido ligado majoritariamente à validação do sistema de execução de procedimentos. Estes simuladores têm sido projetados vislumbrando testes de aceitação de sistemas de controle e treinamento de seus operadores. (AMBROSIO et al., 2006) (AMBROSIO et al., 2007) (INPE, 1990) (INPE, 2000) (INPE, 2007b)

Estes simuladores requerem a implementação de interfaces com aplicativos de monitoramento de telemetrias, envio de telecomandos e disparo de medidas de rastreio, para simular a comunicação entre satélite e estação terrena, e entre estação terrena e centro de controle. Estes simuladores devem possuir mecanismos de recepção de mensagens, de telecomandos e solicitações de medidas de rastreio, a fim de retornar valores de telemetrias e mensagens de medidas. Protocolos de comunicação são implementados, de tal forma que, efetivamente, este tipo de simulador incorpora funções do satélite e da estação terrena. Isto é visível na figura a seguir, que ilustra uma arquitetura proposta para um simulador do Satélite Sino-Brasileiro de Recursos Terrestres Três (CBERS-3), desenvolvido no INPE para o treinamento de operadores do CRC.

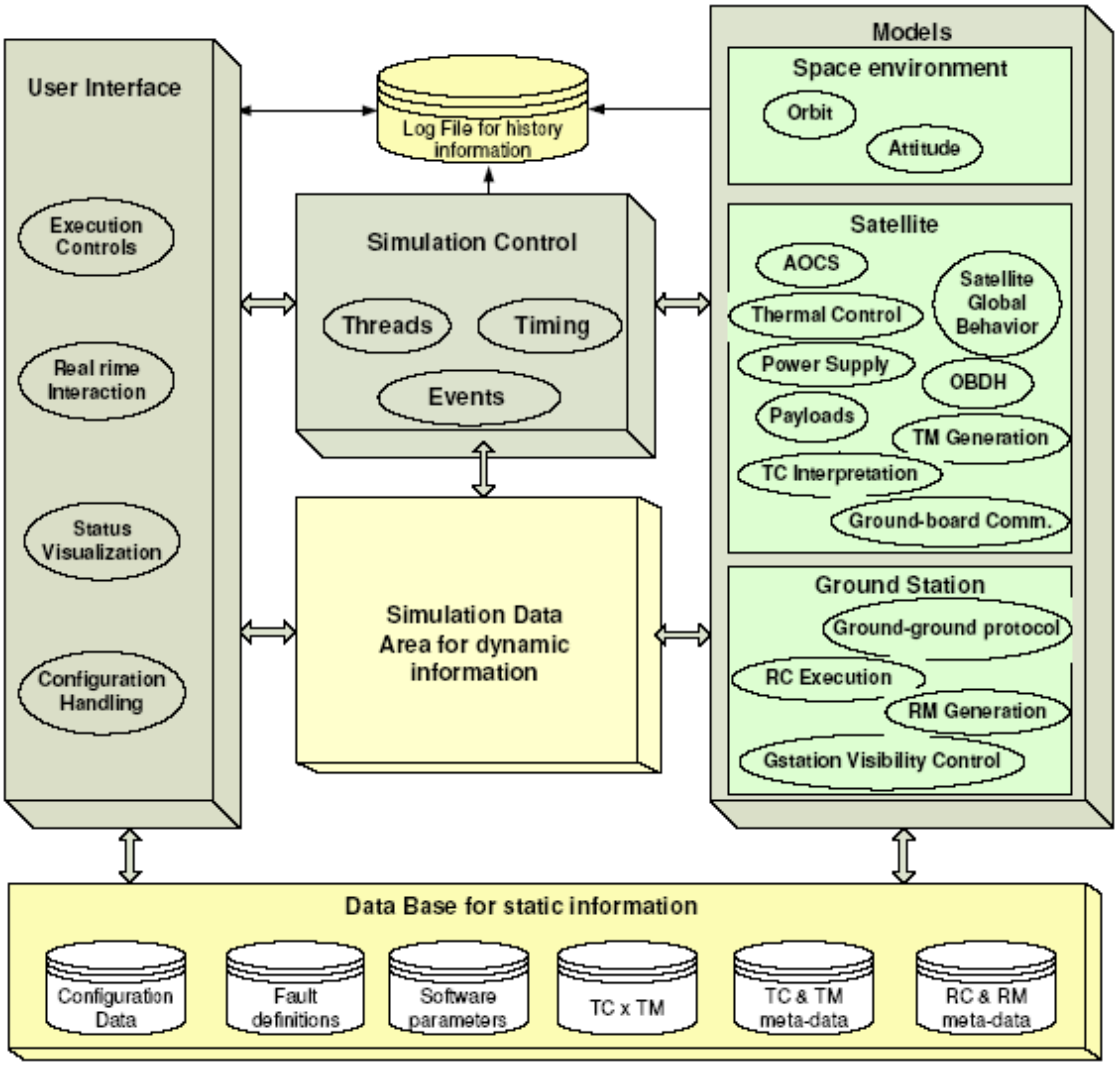


Figura 3.1 - Arquitetura de um simulador de satélite para treinamento de operadores
Fonte: Ambrosio et al. (2006)

A fidelidade requerida para o modelo interno do satélite não é alta nestes simuladores, bastando refletir efeitos de telecomandos e anomalias em bordo, de forma coerente, em valores de telemetrias. O bom funcionamento do simulador depende em parte da experiência da equipe responsável por cadastrar os parâmetros. (INPE, 2007b)

Usos semelhantes de simuladores, ou seja, para treinamento de operadores e validação de execução de procedimentos, podem ser observados também fora

do INPE. (INNORTA; WILLIAMS, 2007) (LEE et al., 2005) (MATHESON, 2008) (MO et al., 2000) (REGGESTAD et al., 2004) (SUN et al., 2000)

Outro exemplo de uso, mais próximo à proposta de validação de planos de operações, consiste na automação de execução de procedimentos em vôo. Nesta abordagem, utiliza-se um simulador de satélite para a validação de atividades a serem executadas. Um simulador deste tipo é elaborado de acordo com modos de operação previstos e documentados em fase de projeto. Devido à complexidade, modos de operação degradados causados por anomalias complexas não são consideradas. (MATHESON, 2008)

Simuladores de satélites têm sido utilizados também para dar suporte a projetos de novos satélites. O custo de implementação em software para computadores de modelos numéricos previamente construídos é relativamente baixo, especialmente em termos de tempo de desenvolvimento, ao se comparar à fabricação de equipamentos especializados. Desta forma, utilizam-se modelos de alta fidelidade para se validar projetos de hardware. Esta abordagem tem sido aplicada comumente em missões espaciais. (HENDRICKS; EICKHOFF, 2005)

Estes simuladores podem ser empregados também para dar suporte a operações de montagem, integração e testes (AIT). Durante a execução de testes integrados de satélites, comandos são enviados simultaneamente para a implementação sob teste e para um simulador. Respostas obtidas por meio de telemetria são comparadas para verificar se há discrepâncias entre valores esperados e medidos, permitindo a verificação se especificações de projeto estão sendo atendidas ou não. Este tipo de simulador possibilita inclusive sua aplicação posterior à automação de operações de execução de procedimentos, apresentando a vantagem de terem sido exaustivamente testados e validados com o satélite em vôo, já que os comportamentos de ambos foram considerados coincidentes nos casos de teste aplicados ao satélite.

(HENDRICKS; EICKHOFF, 2005) (KANG et al., 1995) (LYON et al., 2002) (SUN et al., 2000)

No caso do simulador proposto para verificação de planos, seu objetivo é simular um satélite em vôo, a fim de garantir a sua segurança. Sabe-se que, ao longo a vida útil e estendida do satélite, falhas em bordo levam a alterações nos modos de operação. Isto determina alterações nos critérios de seleção de atividades de controle. Algumas combinações de falhas mais complexas podem não ser previstas durante as fases iniciais de projeto da missão. Sua modelagem deve ser feita, portanto, em vôo. Algumas atividades de controle podem ser seguras ou inseguras, dependendo do modo operacional em que se encontra o satélite. Portanto, este simulador deve permitir a adequação de seus modelos internos, por meio de atualizações.

3.2. Arquiteturas de construção de simuladores

Simuladores de satélites são tipicamente constituídos por um núcleo de processamento ligado a diversos módulos, que podem rodar numa mesma máquina, ou então diversos módulos rodando de forma distribuída. Tradicionalmente, estes módulos são separados de acordo com funções de equipamentos, organizados conforme os subsistemas componentes dos satélites. (AMBROSIO et al., 2007) (ARGÜELLO; MIRÓ, 2000) (INPE, 1990) (REGGESTAD et al., 2004) (SCHUM et al., 2006) (SEBASTIÃO et al., 2008) (SUN et al., 2000) (XSCC; INPE, 2000)

Um simulador deste tipo pode ser construído a partir de interconexão de modelos matemáticos independentes. A transferência de dados ocorre da mesma forma que em um satélite real. Um subsistema de gerenciamento de dados embarcados processa dados de telemetrias e telecomandos. Valores de telemetrias são enviados por equipamentos de diferentes subsistemas que, por

sua vez, recebem e executam telecomandos. A figura a seguir ilustra este fluxo de dados num satélite e sua transposição para um modelo de simulador.

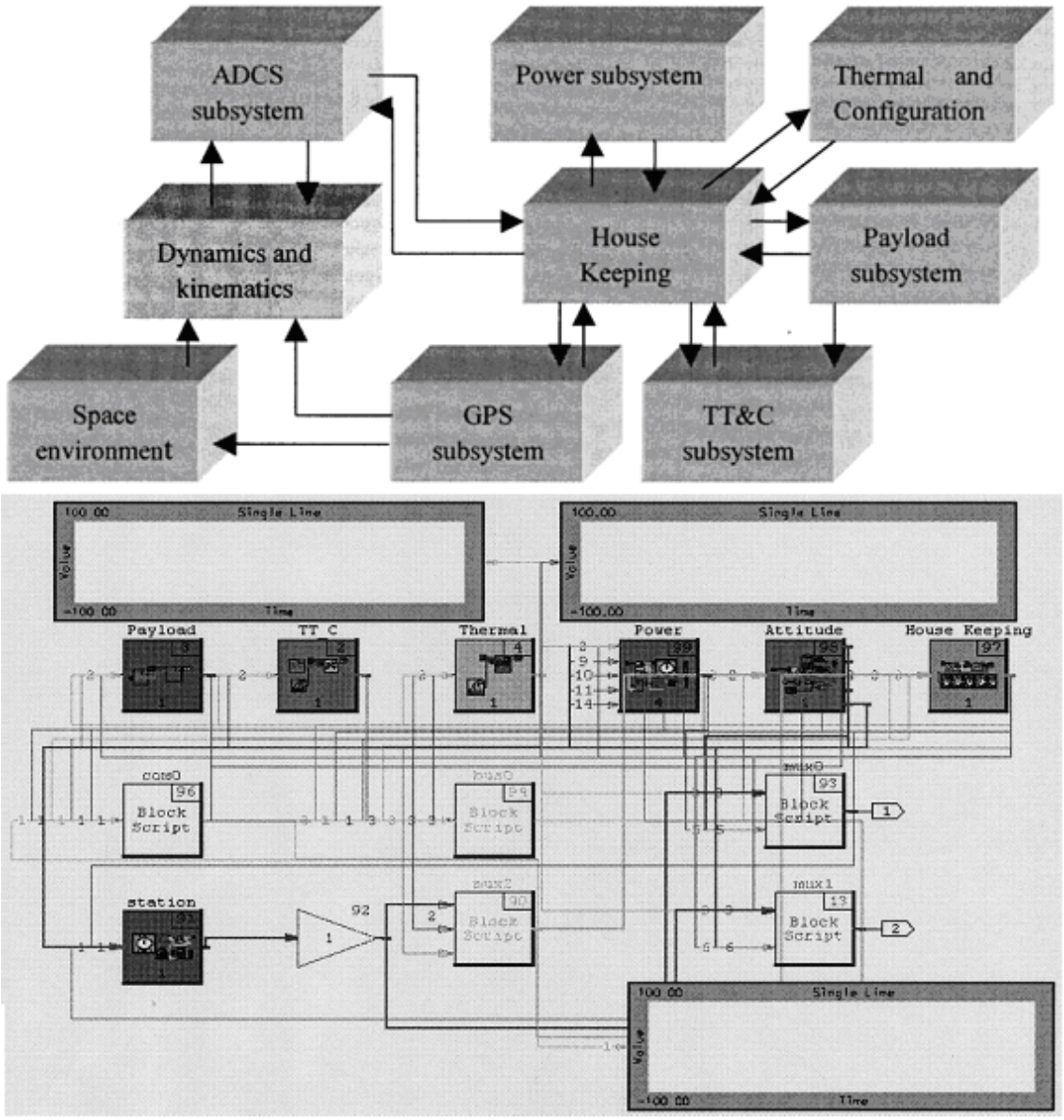


Figura 3.2 - Interconexão de modelos matemáticos em um simulador de satélites
Fonte: Sun et al. (2000)

Na parte superior da figura acima, tem-se o diagrama de blocos do modelo de sistema do simulador explicitando seus componentes, altamente correlacionados aos subsistemas do satélite. As setas indicam fluxos de

informação entre estes componentes. Na parte de baixo, mostra-se a estrutura de sua implementação em software, no caso, em MATLAB©. (SUN et al., 2000)

Custos de desenvolvimento de simuladores de satélites podem ser diluídos ao se aplicarem modificações incrementais em cima de uma arquitetura básica de referência, ao invés de se iniciarem novos projetos de simulador para cada missão, a partir do nada. Subsistemas de satélites diferentes desempenham funções semelhantes, apesar de não exatamente iguais. A codificação de detalhes de cada missão pode ser realizada em cima de uma base referencial genérica. Assim, o código de um simulador projetado para uma missão pode ser reaproveitado para missões subseqüentes, com as devidas alterações. Este tipo de enfoque, adotado pelo INPE na construção de simulador do satélite SCD2 a partir do simulador do SCD1. A figura a seguir ilustra uma seqüência de desenvolvimento de simuladores, em que arquitetura básica de referência foi adaptada progressivamente para adequar a diversas missões da Agência Espacial Européia (ESA). (SEBASTIÃO et al., 2008)

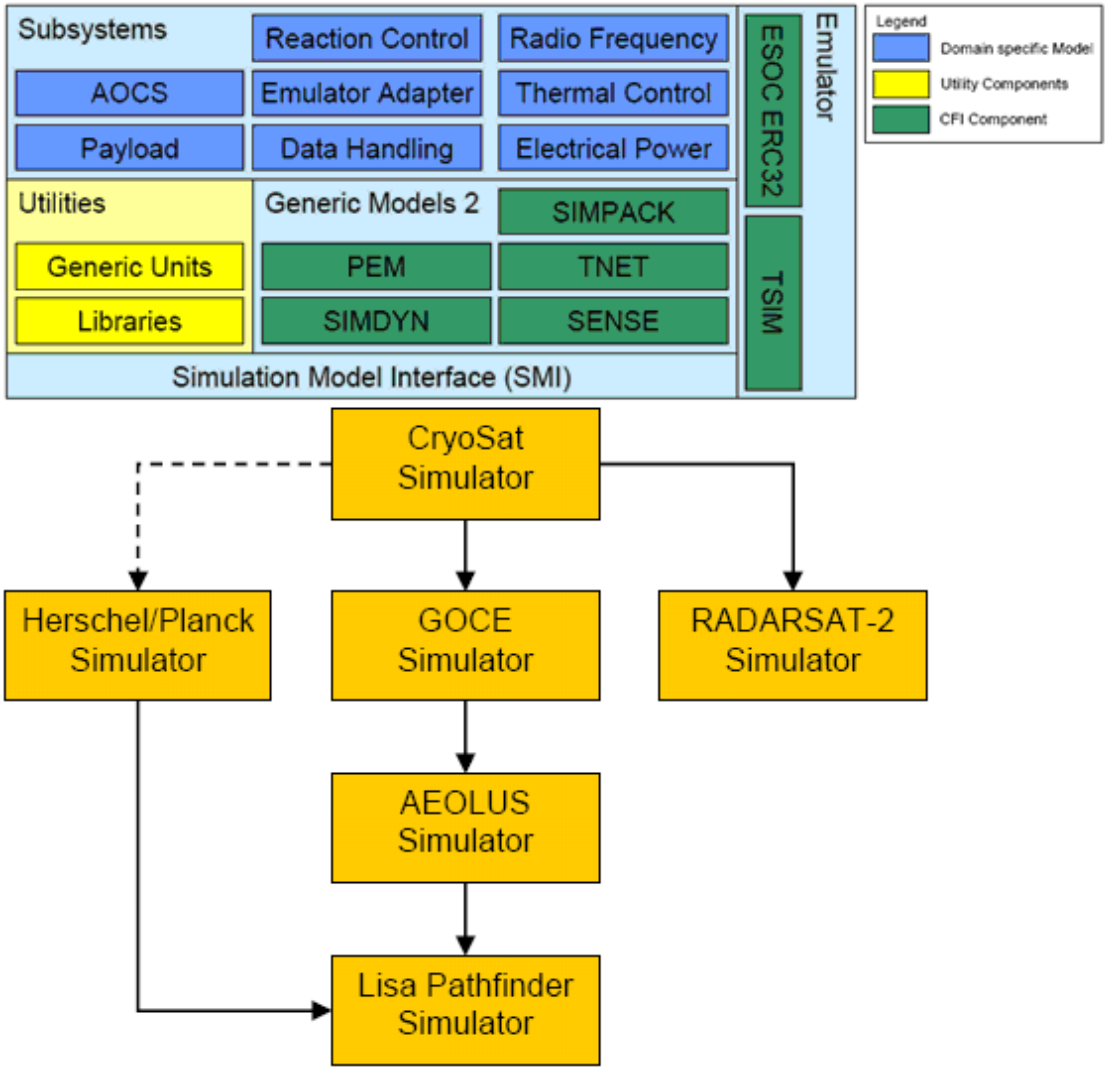


Figura 3.3 - Exemplos de simuladores de missões espaciais desenvolvidos a partir de uma arquitetura base
Fonte: Sebastião et al. (2008)

Tradicionalmente, projetos de simuladores de satélites têm se baseado em modelos de alta fidelidade, construídos artesanalmente por especialistas de subsistemas com anos de experiência. Recentemente, instituições têm se movido em busca de projetos padronizados para simuladores. Isto possibilita a construção de simulador de novos satélites com menor esforço e custo. Recomendações voltadas à padronização de simuladores de satélites têm sido desenvolvidas para este fim, por exemplo, pela Cooperação Européia para

Padronização Espacial (ECSS). (AMBROSIO et al., 2007) (GOLDFINGER et al., 2000) (SEBASTIÃO et al., 2008)

Em geral, a tendência atual quanto à construção de simuladores de satélites consiste em montá-los a partir de modelos pré-existentes, preferencialmente fornecidos por fabricantes de equipamentos embarcados, e integrá-los através de um núcleo de processamento adaptável para diversas missões. Esta abordagem tem a vantagem de reduzir custos. Porém, os modelos empregados são baseados em especificações de projeto da missão. Para o simulador de satélites proposto para verificação de planos de operações em vôo, os modelos devem refletir o estado corrente do satélite que, dependendo de combinações de falhas embarcadas, podem não ter sido previstas durante a fase de projeto. Assim, mecanismos de atualização de seus modelos tornam-se necessários.

3.3. Processamento de parâmetros de simulação

Dos simuladores de satélites pesquisados, a maioria daqueles que revelam um com pouco mais detalhes o seu funcionamento interno processa telemetrias e telecomandos. Isto facilita a modelagem do simulador, quando baseada no projeto do satélite real. Nesta abordagem, bancos de dados utilizados para atividades de controle podem ser utilizados direta ou indiretamente pelo simulador, garantindo sua compatibilidade com o satélite. (AMBROSIO et al., 2006) (INPE, 1990) (SUN et al., 2000) (XSCC; INPE, 2000)

O conjunto de telemetrias não necessariamente inclui todas as informações necessárias para a obtenção de um panorama geral do estado do satélite. Em projetos de satélites reais, informações relevantes são sacrificadas em detrimento de outras consideradas mais importantes durante o processo de codificação de quadros de telemetria, devido à limitação dos canais de comunicação. Alguns parâmetros devem ser processados a partir de telemetrias a fim de fornecer valores. Por exemplo, na ausência de sensores

para sua medição direta, o estado de carga de baterias deve ser calculado a partir de telemetrias de correntes e tensões. Neste processo, variáveis auxiliares se fazem necessárias para a solução de equações de balanço de potência. Além de telecomandos, há outros fenômenos e eventos externos que podem afetar um satélite. Um exemplo típico é o efeito de estado de iluminação ou sombra do satélite sobre a geração de energia nos painéis solares. Efeitos podem ser causados também por combinações de valores de parâmetros, conforme o exemplo da comutação automática do circuito de carga de bateria, disparado por extremos de temperatura. (CAST; INPE, 2006) (INPE, 2006)

Desta forma, um simulador de satélites cujo objetivo é permitir a análise dinâmica de seus estados internos não pode se restringir ao processamento de telemetrias e telecomandos. Ele deve modelado de forma mais apropriada, através de parâmetros de simulação que incluem, mas não se restringem apenas a, telemetrias e telecomandos. (INPE, 2007b) (AMBROSIO et al., 2007)

3.4. Interfaces de simuladores

As interfaces de simuladores desenvolvidos com participação do INPE para testes sistemas de controle em solo e treinamento de operadores têm contado com dois tipos de interfaces. Uma interface para o sistema de controle em solo, e outra para o condutor da simulação. A interface para o sistema de controle em solo permite o monitoramento de telemetrias e envio de telecomandos utilizando o mesmo ambiente utilizado para execução de procedimentos em tempo-real num satélite de verdade. A outra interface permite a um condutor configurar e disparar o processo de simulação, além de monitorar e alterar seus estados internos. Um mecanismo de sincronização controla o passo de simulação, garantindo a geração de telemetrias de acordo com a taxa de transmissão real do satélite. (AMBROSIO et al., 2006) (INPE, 1990) (INPE, 2007b) (XSCC; INPE, 2000)

Diferentemente de simuladores projetados para treinamento de operadores, o simulador para verificação de planos não requer interfaces para processamento de eventos em tempo real. O plano de operações a ser testado já contém naturalmente uma lista pré-programada de procedimentos para execução. Bastaria o simulador extrair seus tempos de execução a partir do plano e dispará-los de acordo com o tempo de simulação. Eventos externos ser previamente avaliados segundo modelos de ambiente espacial, e fornecidos pela equipe de dinâmica de vôo na forma de previsões de eventos orbitais. Isto permite que fenômenos externos e procedimentos a serem executados possam ser agrupados em uma única fila de eventos temporizada, a ser carregada antes da simulação. A análise da dinâmica de seus estados internos também não requer uma interface de tempo-real. De fato, o resultado da simulação deve ser gerado antes da execução dos procedimentos planejados. Isto elimina a necessidade de mecanismos de sincronização do passo de simulação, bastando gerar uma seqüência de parâmetros de estados internos, ao fim da simulação, com carimbos de tempo correspondentes aos passos simulados. Desta forma, o simulador proposto requer a implementação de apenas duas interfaces: uma para sua configuração, e outra para exportação de estados internos para análise.

3.5. Avaliação dos simuladores estudados

Conforme visto neste Capítulo, um simulador de satélite para verificação de planos de operação em vôo tem como objetivo permitir a análise da dinâmica do sistema. Para isto, o simulador deve ser construído de forma a refletir o comportamento do sistema, de acordo com o conhecimento de especialistas no assunto. Entretanto, algumas combinações de falhas e anomalias são imprevisíveis, portanto seus modelos devem ser projetados de forma a permitir atualizações, preferencialmente de maneira fácil.

As arquiteturas recentes de simuladores, baseada em um núcleo de processamento comum acoplado a modelos associados a missões específicas, trazem vantagens significativas ao facilitar as reconfigurações de modelos de missões existentes para novas missões. Esta vantagem deve ser válida também para reconfigurações executadas dentro de uma mesma missão, a fim de refletir alterações nos seus modos de operação. Portanto, esta arquitetura deve ser também adotada pelo simulador para verificação de planos.

Devido à necessidade de adaptação do modelo, o simulador para verificação de planos deve ser projetado tendo-se em mente a configuração de seus modelos. Alguns dos simuladores estudados permitem a configuração de parâmetros armazenados em bancos de dados. Porém, em nenhum deles foi constatada a possibilidade de introduzir mudanças mais radicais no modelo como, por exemplo, a substituição de fórmulas e expressões de modelos numéricos. (AMBROSIO et al., 2006) (AMBROSIO et al., 2007) (INPE, 1990) (INPE, 2007b) (XSCC; INPE, 2000)

Para permitir uma análise do modelo interno, os parâmetros de simulação do simulador para verificação de planos não podem se restringir apenas a telemetrias e telecomandos. Isto porque no simulador proposto, a capacidade de reconfiguração deve se estender não somente aos parâmetros de simulação, mas também às equações matemáticas que regem o seu processamento. Desta forma, parâmetros auxiliares representando valores numéricos que estariam presentes apenas dentro do núcleo de processamento nos simuladores apresentados, têm de ser portados para o banco de dados do modelo. Portanto, estruturas de dados devem ser buscadas, tais que permitam o armazenamento de todas estas informações em bancos de dados, desacoplando-os no núcleo de processamento.

Por fim, as interfaces do simulador devem possibilitar a configuração de seu modelo antes da execução da simulação e, posteriormente, a análise de seus

estados internos. Diferentemente de simuladores de tempo-real, o simulador para verificação de planos não requer interfaces para processamento de eventos que não estejam previamente previstos. A interface para processamento de eventos do simulador para verificação de planos, portanto, pode se resumir numa entrada para uma seqüência pré-programada de eventos temporizados. (AMBROSIO et al., 2006) (INPE, 1990) (INPE, 2007b) (XSCC; INPE, 2000)

No próximo Capítulo, estes requisitos serão discutidos em detalhes para se chegar a uma arquitetura de simulador para verificação de planos que atende todos eles.

4 ARQUITETURA DO SIMULADOR

Conforme visto no Capítulo anterior, a construção do simulador de satélites para verificação de planos de operação em vôo deve possuir algumas características para atender alguns requisitos. Primeiramente, a representação do conhecimento no modelo deve ser tal que permita facilmente a sua atualização devendo, portanto, ser configurável. Em seguida, sua arquitetura deve ser tal que seu modelo de sistema em banco de dados contenha também representações de modelos matemáticos e parâmetros auxiliares, separados do seu núcleo de processamento. Por fim, sua interface deve permitir sua configuração antes da simulação, e posteriormente a análise de estados internos de seus modelos, não sendo necessário processar eventos externos em tempo-real. Neste Capítulo, será proposta uma arquitetura para simulador que visa ao atendimento estes requisitos.

4.1. Representação do conhecimento do sistema

Tradicionalmente, projetos de simuladores utilizam arquiteturas baseadas em modelos matemáticos analíticos, organizados por divisão funcional em nível de subsistemas, para a representação das estruturas de dados de seus modelos. (AMBROSIO et al., 2007) (INPE, 1990) (REGGESTAD et al., 2004) (SCHUM et al., 2006) (SEBASTIÃO et al., 2008) (SUN et al., 2000) (XSCC; INPE, 2000)

Para minimizar erros na transferência de informações, a base de dados do simulador deve replicar o conhecimento dos especialistas de forma mais fiel possível. Idealmente, os próprios especialistas deveriam ser capazes de, facilmente, cadastrar as informações necessárias no modelo. Se usuários perceberem dificuldades na migração de seus conhecimentos para os modelos, muito provavelmente eles delegarão esta tarefa a outras pessoas. Se usuários não-especialistas realizarem as tarefas de configuração mecanicamente, sendo incapazes de enxergar eventuais erros e analisá-los criticamente, poderão vir a comprometer a validade do modelo.

Dinâmicas internas de satélites são usualmente documentadas na forma de conjuntos de regras causais. Manuais de satélites descrevem o funcionamento de subsistemas e equipamentos, informando faixas de operação a serem mantidos para telemetrias, e telecomandos necessários para obter o efeito desejado, bem como descrições de causa e efeito dos mecanismos de controle autônomo embarcados. Planos de execução de procedimentos listam passos a serem seguidos por operadores, informando as pré-condições necessárias e os efeitos esperados associados à execução de cada etapa do procedimento. E documentos de processamento de falhas indicam ações a serem tomadas, por meio de tabelas de Análise de Modos de Falha e Efeitos (FMEA) e/ou Análise de Modos de Falha, Efeitos e Criticidade (FMECA), listando possíveis causas de falhas e sintomas observáveis. (CAST; INPE. 2007) (INPE, 1988) (XSCC, 2000) (XSCC; CAST, 2009)

Desta forma, um simulador que represente em sua base de dados a dinâmica do sistema na forma de regras causais facilita a transferência de conhecimento do especialista para a base de dados durante seu processo de configuração inicial.

Durante o desenvolvimento de simuladores de satélites, são levados em consideração cenários e modos de operação vislumbrados por projetistas. Entretanto, anomalias podem ocorrer em combinações inesperadas, e modos de operação reais podem diferir bastante daquelas inicialmente previstas. (INPE, 2007a) (MATHESON, 2008)

Para garantir a segurança das missões, o simulador deve permitir a reconfiguração de seu modelo. Cada modelo deve refletir o conhecimento de especialistas a respeito das dinâmicas que regem os estados internos da missão que representam. Para garantir correspondência constante entre o modelo do simulador e o estado interno do satélite, especialistas responsáveis

por este gerenciamento deverão ter acesso irrestrito aos mecanismos internos do simulador. Execução e manutenção, incluindo configuração de bases de conhecimento e modificação de software, devem ficar a cargo destes especialistas. A fim de facilitar atualizações freqüentes por especialistas, os modelos de missão devem ser configuráveis e independentes.

O INPE conta com uma equipe de especialistas, que podem fornecer modelos precisos de subsistemas e equipamentos componentes. Porém, caso um satélite entre em algum modo anormal não previsto em projeto, a implementação de comportamentos associados em modelos matemáticos analíticos e a calibração de parâmetros relevantes pode demorar além da vida útil do satélite. A mudança de modo de operação de um satélite requer que atividades de controle a serem executadas sejam alteradas de acordo para aplicação imediata. Assim, o planejamento de operações requer urgência na atualização do modelo do simulador para verificação de planos. Esta urgência pode demandar um simulador modelado de forma simplificada, a ser utilizada em caráter emergencial. A adoção de modelos baseados em regras permite a implementação rápida de alterações para reconfiguração do simulador.

4.2. Desacoplamento entre modelo e processamento

Procedimentos de atualização de modelos são preferíveis à confecção de novos modelos a partir do zero. O mesmo se aplica no caso de cadastramento de novas missões, e de alteração de modo operacional de missões existentes.

Usualmente, especialistas de modelos não são programadores de software profissionais, e vice-versa. Se a representação de regras de sistema for descrita em nível de código-fonte, cada atualização do modelo demandará trabalho coordenado entre especialistas de sistema e programadores de software. Isto é mais oneroso, em termos de tempo requerido e alocação de recursos humanos, do que delegar esta função exclusivamente a especialistas

responsáveis pelo modelo. Para isto, torna-se necessária a elaboração de um ambiente de reconfiguração de regras que seja familiar ao especialista, e que não requeira a intervenção do programador de software.

Uma solução para este problema consiste na elaboração de simuladores com desacoplamento entre núcleo de processamento e bancos de dados. Cada simulador é constituído por dois componentes: um núcleo de processamento compilado comum, e um conjunto de bases de dados configurado independentemente para cada missão. Uma vez compilado o núcleo de processamento, termina o trabalho dos programadores de software. O simulador é capaz de trabalhar com diversos modelos, cuja criação e manutenção dependem apenas dos especialistas dos modelos.

O desacoplamento entre modelo e processamento é implementado em outros simuladores, porém não da maneira como se requer para a construção do simulador para verificação de planos. Nos outros casos, parâmetros de telemetria e telecomando que fazem parte do modelo são armazenados em bancos de dados. Porém, modelos matemáticos e parâmetros auxiliares são incorporados ao núcleo de processamento, sendo programados em nível de código-fonte. (AMBROSIO et al., 2007) (INPE, 1990) (XSCC; INPE, 2000)

O nível de desacoplamento entre modelo e processamento desejado para o simulador para verificação de planos requer que estes modelos matemáticos sejam armazenados em banco de dados. Uma maneira de fazer isto consiste em definir modelos matemáticos baseados inteiramente em estruturas do tipo regras causais. A construção de simuladores segundo esta abordagem é possível, conforme será mostrado no próximo Capítulo.

4.3. Estruturas de dados do modelo

As estruturas de dados do modelo devem ser capazes de refletir a dinâmica do estado interno do satélite, cujos parâmetros mudam conforme a execução de procedimentos contidos num plano de operações sob teste. A dinâmica do estado interno deve ser regida por regras causais, que expressam as condições e os efeitos causados sobre o estado interno do sistema. Devem fazer parte do modelo também eventos externos, cujas ocorrências condicionam efeitos aplicados sobre o sistema.

O estado interno instantâneo do simulador pode ser representado por meio de um conjunto de variáveis numéricas reais, que representam parâmetros do estado interno em um dado instante de tempo. Neste caso, a dinâmica interna do sistema seria caracterizada através de uma seqüência temporizada de estados internos instantâneos. Parâmetros de estado incluem telemetrias e outras variáveis relevantes para a descrição do sistema.

Perturbações no sistema podem ser causadas por disparos de eventos, que levam a alterações em valores de parâmetros que compõem o estado interno. Estes eventos incluem execução de telecomandos e fenômenos ambientais. Seus disparos ocorrem em instantes de tempo pré-programados, conforme explicitados em planos de operações em vôo e previsões de eventos orbitais. Estes eventos podem ser configurados numa única fila de eventos, a ser consultada a cada passo, ao longo de uma sessão de simulação.

Processamento de valores de parâmetros de estados e atribuições de efeitos a disparos de eventos podem ser controladas por meio de regras lógicas. Estas regras de controle são disparadas ou não, dependendo do resultado da avaliação de uma expressão de condição que associa eventos e parâmetros de estado. Se a condição for satisfeita, valores de parâmetros de estado são

modificados conforme os efeitos especificados pela regra. Estas regras refletem o conhecimento do especialista acerca da dinâmica do sistema.

A figura a seguir resume o relacionamento entre as estruturas de dados conforme a descrição anterior.

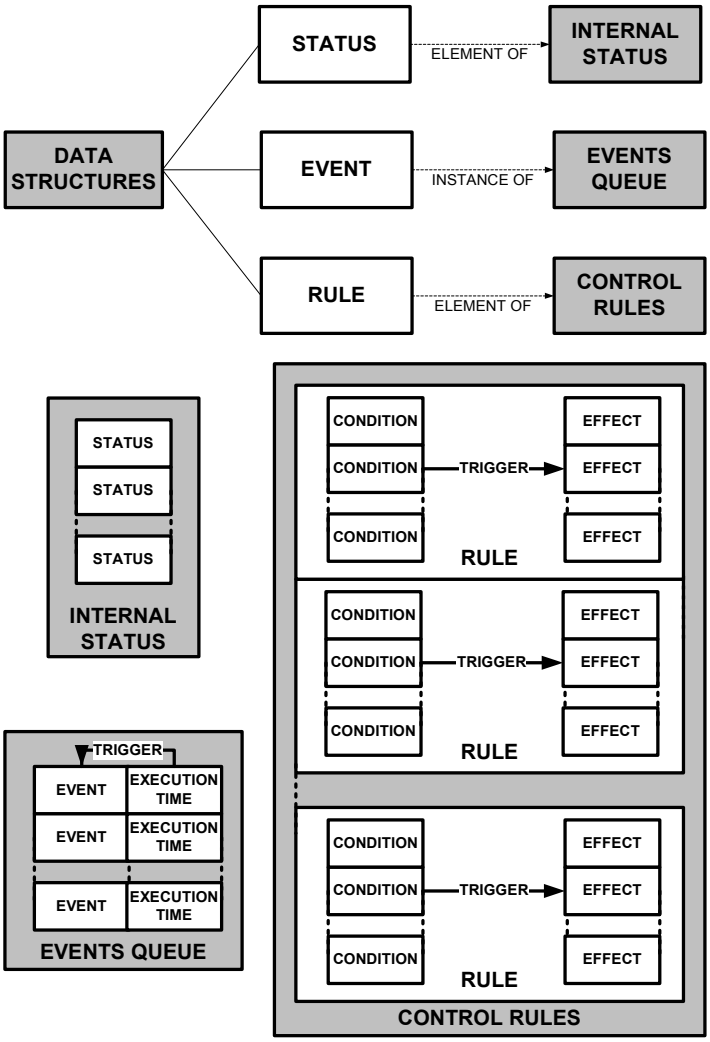


Figura 4.1 - Estruturas de dados para representação da dinâmica interna do modelo
Fonte: Adaptado de Tominaga et al. (2008)

Um detalhamento dos relacionamentos entre as estruturas de dados mostradas acima será objeto de estudo no próximo Capítulo.

Em termos de estrutura interna, o modelo proposto ignora as divisões em nível de subsistema, tradicionalmente adotadas em função de estratégias de projeto. Em seu lugar, é utilizado um conjunto único de regras, atuantes em nível de sistema. Este enfoque pode levar a uma estrutura de dados confusa à primeira vista. Porém, interações críticas entre equipamentos num satélite real dificilmente são confinados em um nível de subsistema. Isto é verdade especialmente para anomalias mais complexas, do tipo que não são previstas em projeto. Caso seja necessário ou desejável manter a classificação de regras, estados e eventos em nível de subsistemas, isto pode ser feito de maneira transparente para o simulador, através de um banco de dados auxiliar associado a uma ferramenta separada de configuração do modelo.

4.4. Interfaces para configuração e análise

O simulador a ser utilizado para verificação de planos de operação em vôo não requer a implementação de protocolos de comunicação entre satélite, estação terrena e centro de controle, que podem ser omitidos para efeitos de simplificação. Entretanto, o simulador requer que suas estruturas de dados sejam configuradas antes da simulação, e seus resultados devem ser exportados para análise.

Devido à ausência de requisitos de monitoração de estados e inserção de eventos em tempo-real, não há necessidade de integração das interfaces de edição de configuração de base de dados e análise de estados internos ao núcleo de processamento do simulador. Desta forma, uma maneira simples de implementar o simulador consiste em gerar um único aplicativo de programa que constitui um núcleo de processamento. Este programa obtém da base de dados as estruturas de dados configurados a fim de executar uma sessão de simulação. Ao fim da sessão, um histórico dos estados internos simulados é exportado na forma de um arquivo de saída. Segundo esta abordagem, as únicas interfaces que devem ser codificadas no núcleo de processamento são

aquelas de comunicação com arquivos, de entrada e de saída. Esta abordagem cria, no entanto, a necessidade de criação de aplicativos editores e visualizadores externos à parte, ou adequação de formatos para uso de software pré-existente para este fim.

4.5. Arquitetura de simulador para atender os requisitos

A arquitetura do simulador de satélites proposto deve ser capaz de verificar planos de operações em vôo, extraindo deles informações referentes a disparos de eventos que afetam a dinâmica do estado interno do satélite. O conhecimento de especialistas referentes a esta dinâmica é expresso na forma de regras causais lógicas. O modelo do simulador de satélites proposto deve ser de fácil manuseio por estes especialistas. Desta forma, é recomendável que o modelo do simulador obedeça à estruturação de dados de acordo com o conhecimento do usuário, ou seja, baseado em regras lógicas. O simulador deve permitir acesso ao histórico de seus estados internos ao longo de sua sessão de simulação, a fim de permitir a análise de sua dinâmica. Como não há necessidade de acompanhamento da simulação em tempo real, ferramentas de configuração do modelo e visualização de estados internos podem ser desacopladas do simulador, sendo implementadas na forma de aplicativos externos.

Uma arquitetura apropriada para o simulador de satélites proposto consiste em um sistema especialista baseado em regras. Um sistema especialista consiste numa ferramenta de inteligência artificial composta por uma máquina de inferência associada a uma base de conhecimento. Esta arquitetura permite a implantação de modelos de sistema, representando as bases de conhecimento, de forma desacoplada do núcleo de processamento, que seria constituído pela máquina de inferência. (NIKOLOPOULOS, 1997)

Um sistema especialista baseado em regras consiste num software inteligente, que utiliza regras de inferência para resolução ou diagnóstico de problemas. Máquinas de inferência deste tipo têm aplicação em diversos campos, incluindo-se entre eles o planejamento e controle de processos. No caso, estipulam-se ações a serem executadas por meio de regras lógicas, que especificam os efeitos a serem aplicados conforme condições são satisfeitas. Isto é realizado segundo o modelo do sistema, construído com base no conhecimento de um ou mais especialistas. (BERNARD, 1988) (LIAO, 2005)

A figura a seguir ilustra a arquitetura proposta para o simulador de satélite constituído por um sistema especialista baseado em regras. Este simulador é capaz de atender os requisitos levantados até o momento.

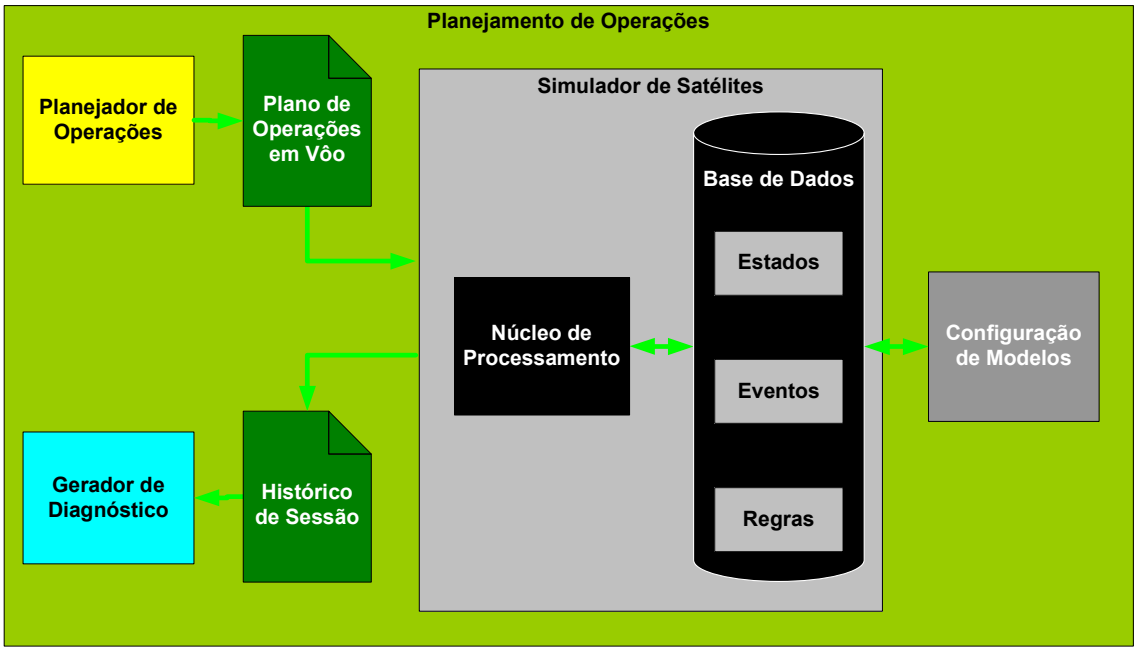


Figura 4.2 - Arquitetura de simulador para atender os requisitos levantados

A separação do simulador em um núcleo de processamento comum para diversas missões associado a bases de dados dedicadas reduz os custos de desenvolvimento. No INPE, tem-se procurado seguir esta abordagem para se

gerar simuladores com núcleo de processamento comum, para satélites da família CBERS e Amazônia. (AMBROSIO et al., 2006)
Esta abordagem será adotada também no simulador proposto para verificação de planos. No entanto, neste simulador o ponto de desacoplamento entre o núcleo de processamento e o modelo em banco de dados foi alterado de forma a conferir controle quase total do modelo ao usuário, conferindo assim maior rapidez e flexibilidade na sua atualização.

A adoção de aplicativos externos para interfaces de configuração de modelos e análise da saída para diagnóstico permite a simplificação do simulador. Torna também possível a elaboração de novas ferramentas para edição de modelos e análises de diagnósticos, conforme surjam novas necessidades.

A tabela abaixo resume as feições consideradas para a escolha desta arquitetura de simulador, relacionando-os às características que justificaram sua adoção.

Tabela 3.1 - Características das abordagens que levaram à adoção da arquitetura

Feições	Características
Modelos de inferência baseada em regras	- Descreve adequadamente a dinâmica interna do sistema. - Reflete o conhecimento do especialista. - Transferência de conhecimento facilitado entre especialistas e modelos.
Desacoplamento entre processamento e modelo	- Reuso de núcleo de processamento em várias missões. - Atualização de regras não requer intervenção de programador. - Atualização de interfaces facilitada.
Aplicativos externos de interfaces	- Simplificação do simulador - Possibilita criação de novos aplicativos independentemente do simulador

No próximo Capítulo, será descrito o funcionamento esperado de um simulador projetado conforme arquitetura escolhida.

5 FUNCIONAMENTO DO SIMULADOR

O simulador de satélite proposto para verificação de planos de operações em vôo consiste em uma máquina de inferência baseada em regras. Este simulador é composto por um núcleo de processamento, que se associa a um banco de dados que contém o modelo do sistema. A configuração do modelo se faz através de editores externos, que não fazem parte do simulador propriamente dito. Neste Capítulo, será descrito um cenário típico de planejamento de operações, a fim de ilustrar o funcionamento deste tipo de simulador. Será evidenciado que para o modelo proposto, a arquitetura proposta é necessária e suficiente para a sua descrição. Porém, será mostrado também que, dependendo da complexidade do modelo, os requisitos levantados podem não ser suficientes para a verificação de planos de operações.

5.1. Verificação dos planos

Conforme visto anteriormente, planejadores a serem validados geram saídas na forma de arquivos de planos de operações em vôo. O simulador deve reproduzir o efeito das atividades contidas nos planos de operações em vôo sobre o estado interno modelado do satélite. A figura a seguir explicita os processos componentes e os dados processados durante a verificação de planos de operação em vôo. Os processos encontram-se representados por meio de retângulos, e os dados, por paralelogramos.

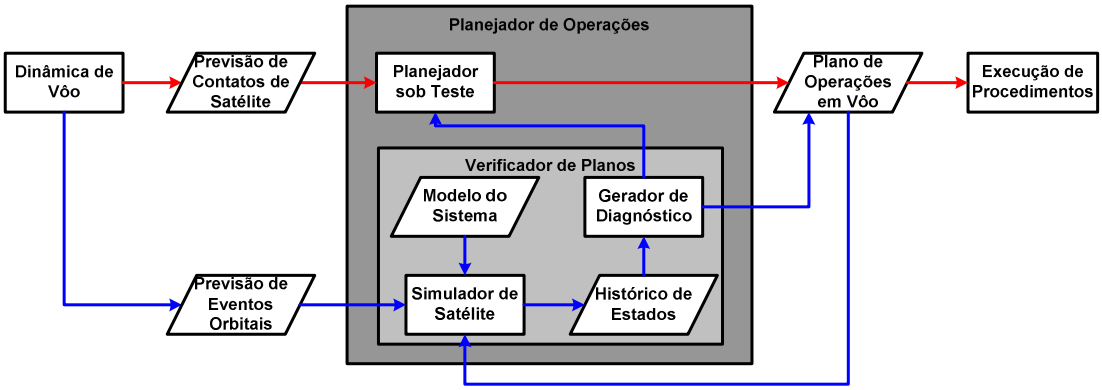


Figura 5.1 - Processos e dados componentes da verificação de planos de operações

Conforme descrito anteriormente, a dinâmica de vôo fornece previsões de contatos de satélites, que são utilizados para a geração de planos de operações em vôo. Planejadores sob teste, concebidos utilizando ferramentas de inteligência artificial, geram planos de operações em vôo de forma mais eficiente do que os planejadores correntes. Se este plano de operações não contiver problemas, ele será encaminhado para a execução de procedimentos. Este caminho representa o fluxo de dados e processos de planejamento, sendo indicado na figura por setas vermelhas.

As setas em azul indicam o caminho da verificação de planos. A dinâmica de vôo fornece, além de previsões de contatos de satélites, arquivos de previsões de eventos orbitais. Aqui se encontram informações adicionais que são externos ao satélite, mas que afetam seu estado interno. A previsão de eventos orbitais inclui, por exemplo, previsões de eclipses solares, que afetam o suprimento de energia por células solares, ou programações de saltos de tempo, que causam perda de sincronismo entre o relógio de bordo e a hora real. O simulador de satélite executa procedimentos constantes no plano de operações e na previsão de eventos para gerar um histórico de estados internos simulados, conforme regras estabelecidas por seu modelo de sistema. Este histórico de estados é analisado por meio de um gerador de diagnóstico, cuja função consiste em aprovar ou reprovar o plano de operações, segundo os requisitos de operação vigentes. O planejador sob teste deverá fornecer planos

de operação alternativos, até que um deles seja considerado seguro ou, em outras palavras, até que se encontre um plano de operações que leve a um histórico de estados que respeite todos os requisitos de operação. Neste caso, a verificação é finalizada e o plano de operações é aprovado para execução.

5.2. Etapas de funcionamento do simulador

O simulador funciona em três etapas executadas seqüencialmente, a começar pela configuração do modelo, seguido da inicialização do simulador e, por fim, a sessão de simulação. A figura a seguir mostra o funcionamento do simulador, indicando como se dá o processamento de informações em cada uma destas etapas.

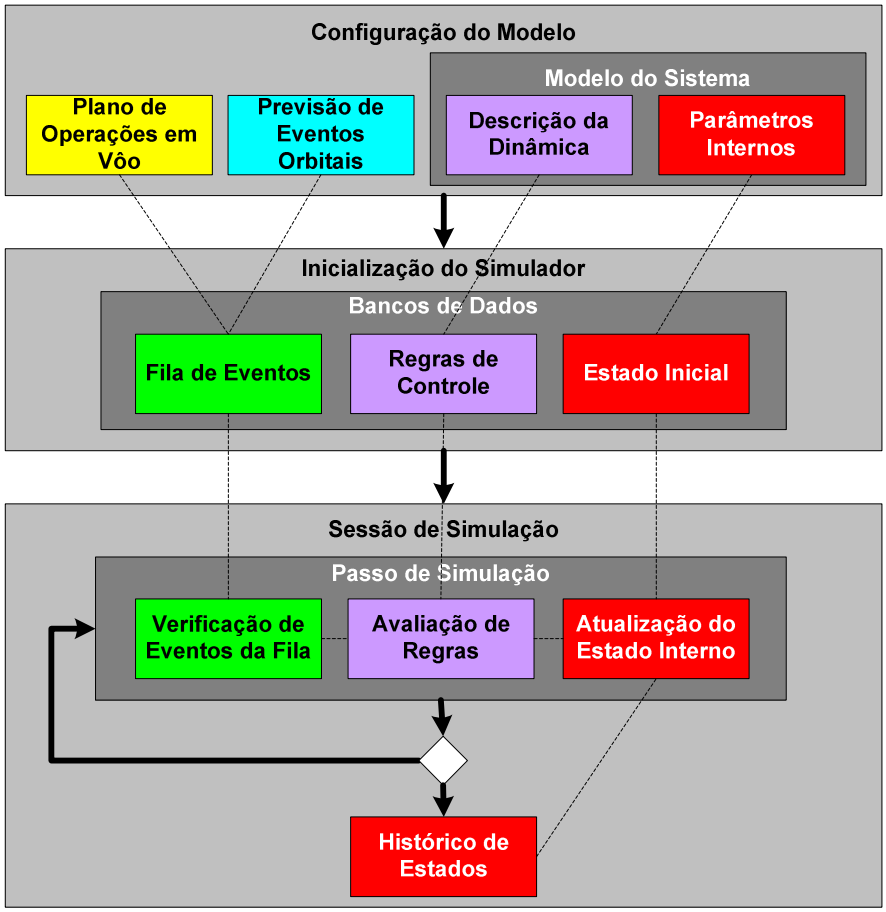


Figura 5.2 - Funcionamento do simulador

A primeira etapa consiste na configuração do modelo, durante o qual os dados de entrada são pré-processados em formatos convenientes para alimentar o simulador. Procedimentos contidos no plano de operação em vôo e eventos orbitais na previsão de eventos são extraídos para se gerar uma fila de eventos. O modelo do sistema contém uma descrição da dinâmica e um conjunto de parâmetros internos que representam seu estado interno. A descrição da dinâmica é representada na forma de regras de controle. Dos parâmetros internos se obtém estado inicial, que representa a situação em que se encontra o sistema no início da simulação. Ao fim desta etapa geram-se arquivos de banco de dados que alimentam o simulador.

Na etapa de inicialização do simulador, a fila de eventos, as regras de controle e o estado interno são lidos dos arquivos de banco de dados que alimentam o simulador. Estruturas de dados correspondentes são armazenadas em memória, sendo acessadas e modificadas durante a execução da sessão de simulação.

Durante a sessão de simulação, executam-se passos de simulação até que se atinja uma condição de parada, após o qual o histórico de estados torna-se disponível para análise e geração de diagnóstico. No primeiro passo, o tempo de simulação corresponde ao tempo inicial e o estado interno corresponde ao estado inicial inicializado. Ao início de cada passo, são verificados se eventos na fila são disparados. Durante cada passo, todas as regras de controle são avaliadas seqüencialmente. Dependendo de disparos de eventos na fila ou valores assumidos por parâmetros de estado que condicionam regras naquele passo, valores de parâmetros de estados afetados por regras disparadas são alteradas, levando à atualização do estado interno naquele passo. O histórico de estados contém uma seqüência de valores de parâmetros que representam o estado interno correspondente a cada passo. Ao final da sessão, o simulador armazena num arquivo o histórico de estados contendo todos os estados internos dos passos simulados.

5.3. Um exemplo simples

Um exemplo ilustrativo de aplicação do simulador consiste num satélite virtual simples, cuja operação se resume a ligar e desligar duas cargas úteis. A modelagem deste exemplo inclui apenas o subsistema de suprimento de energia do satélite, sendo isto necessário e suficiente para caracterizar as atividades rotineiras desta missão. (TOMINAGA et al., 2008a) (TOMINAGA et al., 2009)

A figura abaixo representa o diagrama de blocos do satélite virtual considerado. A energia em bordo é produzida por meio de um gerador composto por conjuntos de células solares (SAG). Este conjunto alimenta o controlador de carga da bateria (BCC) que, por sua vez, alimenta uma bateria recarregável (BAT) e um regulador de barramento principal (MBR). O barramento principal fornece energia para as cargas-úteis (PL1 e PL2) e para equipamentos de serviço (SL).

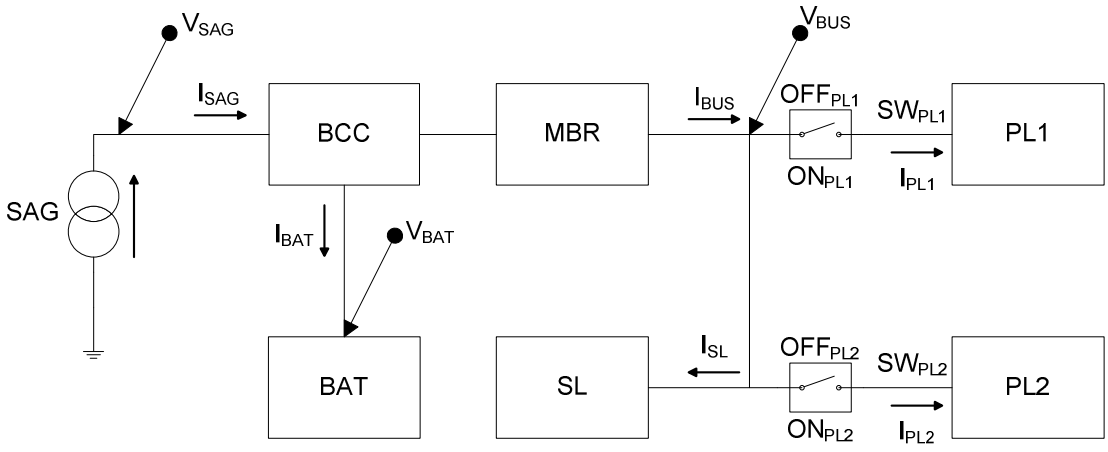


Figura 5.3 - Diagrama de blocos do subsistema de suprimento de energia do satélite
Fonte: Adaptado de Tominaga et al. (2008)

As cargas-úteis do satélite, PL1 e PL2, são operadas de forma independente. Os telecomandos ONPL1 e ONPL2 ligam, respectivamente, as cargas-úteis PL1 e PL2. Os telecomandos OFFPL1 OFFPL2 as desligam, de forma análoga.

As telemetrias do satélite incluem corrente do gerador (ISAG), tensão de bateria (VBAT), tensão e corrente do barramento (VBUS e IBUS), corrente de entrada das cargas-úteis (IPL1 e IPL2) e estado da chave de alimentação das cargas-úteis (SWPL1 e SWPL2).

De forma simplificada, as atividades rotineiras desta missão consistem em temporizar a execução de comandos de liga e desliga das cargas-úteis (ONPL1, ONPL2, OFFPL1 e OFFPL2). O objetivo da simulação deste tipo de operação consiste em monitorar a profundidade de descarga (DOD = *Depth of Discharge*) da bateria. Operacionalmente, é requisitado que o DOD da bateria não ultrapasse o limite de segurança correspondente a 20%.

5.4. Simulação deste exemplo

Conforme descrito anteriormente, a execução do simulador de satélite divide-se em três etapas funcionais. Na primeira etapa, bancos de dados contendo o modelo do simulador são configurados. Na segunda etapa o seu conteúdo, constituído por fila de eventos, regras de controle e estado inicial é inicializado. Por fim, na sessão de simulação, os estados internos simulados são exportados na forma de um arquivo de histórico de estados. Se as atividades levarem o estado interno do simulador a uma situação segura, conclui-se que o plano de operações em vôo não deverá causar problemas ao ser enviado para a execução de procedimentos. Caso contrário, o plano de operações em vôo deverá ser rejeitado para dar início à análise do histórico de estados para um diagnóstico de erros. Apenas planos de operações em vôo aprovados após verificação deverão ser encaminhados para a execução de procedimentos.

A figura a seguir resume de forma ilustrativa as interfaces do simulador após passar pela etapa de configuração. Entradas do simulador (SIM) incluem fila de eventos (EVQ), estado inicial (INIT) e regras de controle (CTRL). Estas entradas são inicializadas. Em seguida, na etapa de sessão, estados internos

são simulados e exportados na forma de um arquivo de saída (OUT). Esta saída corresponde ao histórico de estados.

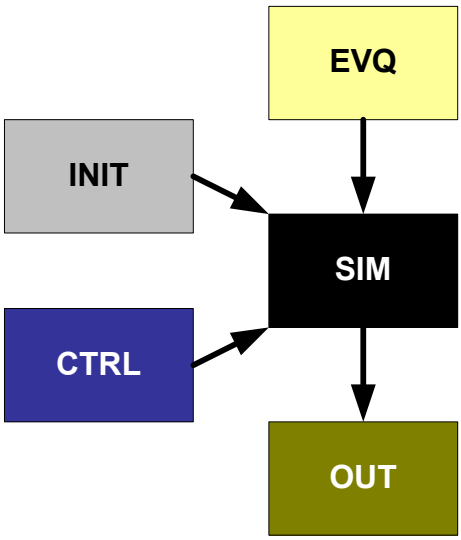


Figura 5.4 - Configuração de inicialização e sessão do simulador de satélite

No estudo a ser considerado neste Capítulo, serão considerados dois planos de operações em vôo para a missão exemplificada, correspondente às operações planejadas para um intervalo de tempo de um dia. O objetivo deste estudo consiste em executar duas sessões de simulação para verificar cada plano de operações. O primeiro plano deve resultar numa saída rejeitada, e o segundo plano, uma saída aceitável. As configurações do simulador para este estudo encontram-se representadas na figura a seguir.

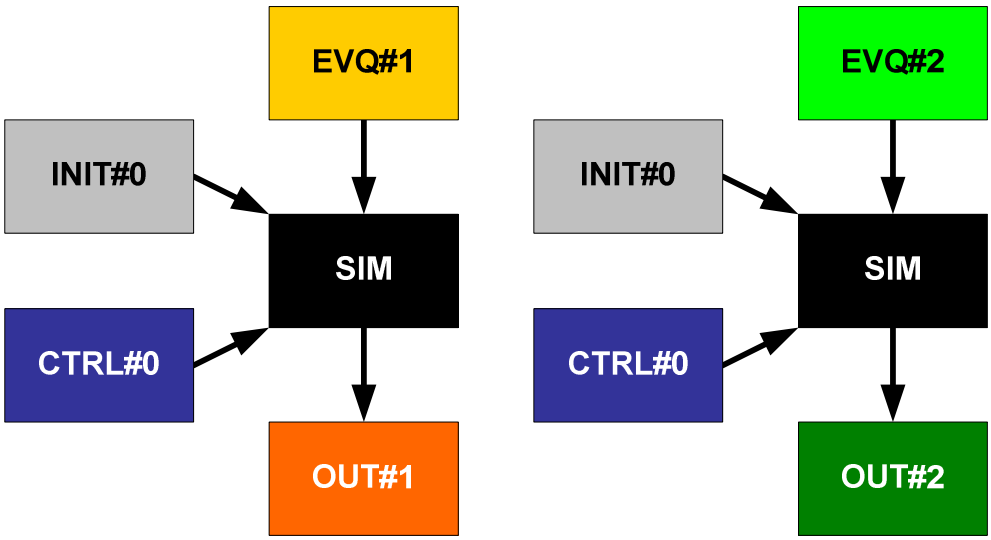


Figura 5.5 - Configurações de inicialização e sessão para estudo do simulador

Um mesmo simulador (SIM) será configurado para ser inicializado com um mesmo estado inicial (INIT#0) e as mesmas regras de controle (CTRL#0), porém filas de eventos diferentes (EVQ#1 e EVQ#2). As filas são geradas a partir de dois planos de operações em vôo e uma mesma previsão de eventos orbitais. Na sessão de simulação, estados internos simulados serão exportados na forma de arquivos de saídas diferenciados para cada fila de eventos (EVQ#1 → OUT#1 e EVQ#2 → OUT#2). Estas saídas serão analisadas mais à frente.

5.5. Construção das filas de eventos

Os candidatos para planos de operações em vôo para esta missão, dentro de um intervalo de um dia, são apresentados na tabela a seguir. Em ambos os planos consideraram-se apenas a execução temporizada dos telecomandos, desprezando-se outras atividades de controle em solo normalmente executadas em missões reais.

Tabela 5.1 - Dois exemplos de planos de operações

PLAN#1	
Execution Time	Telecommand
0:10:00	ONPL1
0:20:00	OFFPL1
1:00:00	ONPL2
1:10:00	OFFPL2
1:50:00	ONPL1
2:00:00	OFFPL1
2:40:00	ONPL2
2:50:00	OFFPL2
3:30:00	ONPL1
3:40:00	OFFPL1
4:20:00	ONPL2
4:30:00	OFFPL2
5:10:00	ONPL1
5:20:00	OFFPL1
6:00:00	ONPL2
6:10:00	OFFPL2
6:50:00	ONPL1
7:00:00	OFFPL1
8:30:00	ONPL1
8:40:00	OFFPL1
10:10:00	ONPL1
10:20:00	OFFPL1
11:50:00	ONPL1
12:00:00	OFFPL1
13:30:00	ONPL1
13:40:00	OFFPL1
15:10:00	ONPL1
15:20:00	OFFPL1
16:50:00	ONPL1
17:00:00	OFFPL1
18:30:00	ONPL1
18:40:00	OFFPL1
20:10:00	ONPL1
20:20:00	OFFPL1
21:50:00	ONPL1
22:00:00	OFFPL1
23:30:00	ONPL1
23:40:00	OFFPL1

PLAN#2	
Execution Time	Telecommand
0:10:00	ONPL1
0:20:00	OFFPL1
1:00:00	ONPL2
1:10:00	OFFPL2
1:50:00	ONPL1
2:00:00	OFFPL1
3:30:00	ONPL1
3:40:00	OFFPL1
5:10:00	ONPL1
5:20:00	OFFPL1
6:50:00	ONPL1
7:00:00	OFFPL1
7:40:00	ONPL2
7:50:00	OFFPL2
8:30:00	ONPL1
8:40:00	OFFPL1
10:10:00	ONPL1
10:20:00	OFFPL1
11:50:00	ONPL1
12:00:00	OFFPL1
13:30:00	ONPL1
13:40:00	OFFPL1
14:20:00	ONPL2
14:30:00	OFFPL2
15:10:00	ONPL1
15:20:00	OFFPL1
16:50:00	ONPL1
17:00:00	OFFPL1
18:30:00	ONPL1
18:40:00	OFFPL1
20:10:00	ONPL1
20:20:00	OFFPL1
21:00:00	ONPL2
21:10:00	OFFPL2
21:50:00	ONPL1
22:00:00	OFFPL1
23:30:00	ONPL1
23:40:00	OFFPL1

A duração total da operação de cargas-úteis é a mesma nos dois planos. A diferença está na alocação das atividades. No primeiro plano (PLAN#1), as operações das cargas-úteis estão concentradas no início do intervalo de tempo considerado. Já no segundo plano (PLAN#2), as operações estão mais bem

distribuídas ao longo do tempo. A figura abaixo ilustra graficamente as operações de cargas-úteis em cada plano de operações.

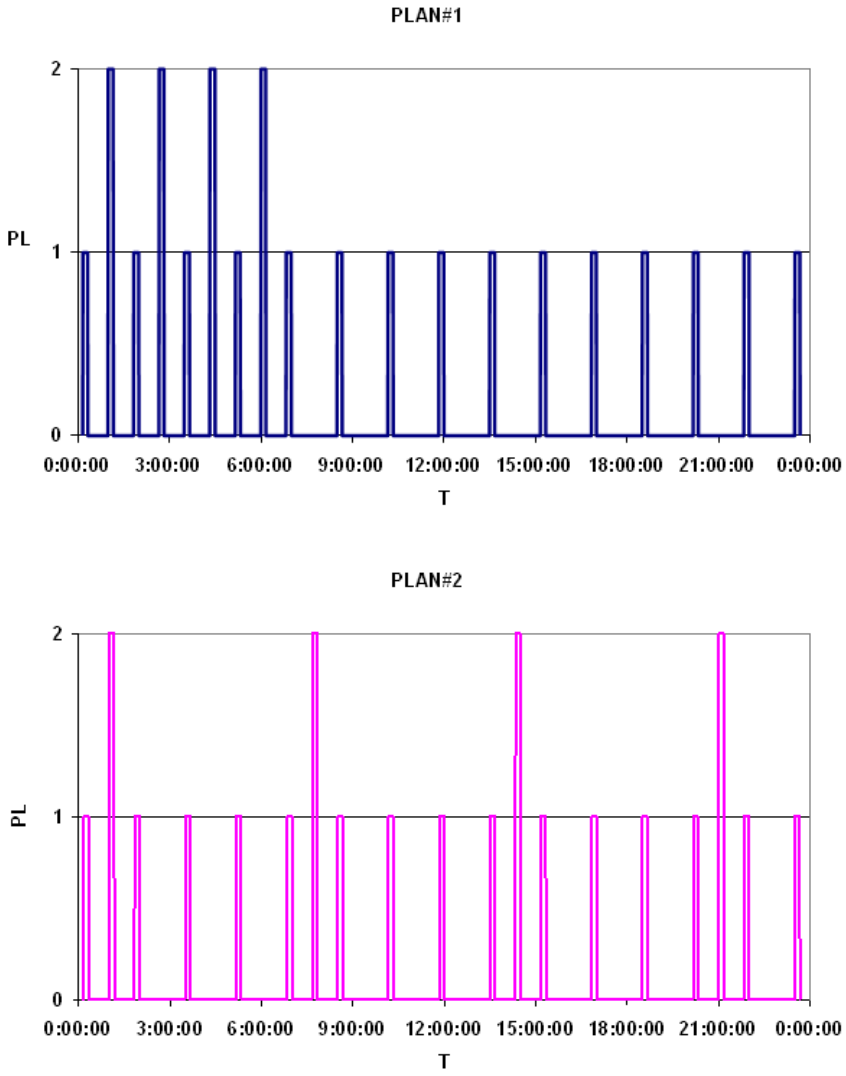


Figura 5.6 - Operações planejadas de cargas-úteis em função do tempo

Na figura acima, o gráfico superior corresponde ao plano PLAN#1 e o gráfico inferior, ao plano PLAN#2. O valor indicativo de nível no eixo Y (PL) representa o estado de operação das cargas úteis durante o tempo correspondente ao eixo X (T). Representa-se por nível zero os intervalos de tempo em que não há operação de carga-útil, nível um quando ocorre operação apenas de PL1, e nível dois quando ocorre operação apenas de PL2. Embora este caso não

esteja nos planos sob análise, se houvesse operação simultânea de ambas as cargas-úteis, esta teria sido representada por nível três.

A tabela abaixo exemplifica uma previsão de eventos orbitais (ECL#0). Ela contém horários previstos para início de fases orbitais iluminada pelo Sol (SUN) e eclipsado pela Terra (ECL) dentro do período abrangido pelos dois planos de operações. A modelagem destes eventos é vital para a simulação da geração de energia embarcada.

Tabela 5.2 - Exemplo de previsão de eventos orbitais

ECL#0	
Start Time	Event
0:20:00	SUN
1:26:40	ECL
2:00:00	SUN
3:06:40	ECL
3:40:00	SUN
4:46:40	ECL
5:20:00	SUN
6:26:40	ECL
7:00:00	SUN
8:06:40	ECL
8:40:00	SUN
9:46:40	ECL
10:20:00	SUN
11:26:40	ECL
12:00:00	SUN
13:06:40	ECL
13:40:00	SUN
14:46:40	ECL
15:20:00	SUN
16:26:40	ECL
17:00:00	SUN
18:06:40	ECL
18:40:00	SUN
19:46:40	ECL
20:20:00	SUN
21:26:40	ECL
22:00:00	SUN
23:06:40	ECL
23:40:00	SUN

Conforme comentado anteriormente, a duração total da operação de cargas-úteis é igual nos dois planos e, além disto, os horários de operação da carga-útil PL1 também são idênticos. Entretanto, a diferença no padrão de uso da carga-útil PL2 trará efeitos significativos no consumo de carga da bateria, conforme será mostrado a seguir.

Instâncias temporizadas de parâmetros do tipo evento constituem uma fila de eventos. Neste estudo, foram geradas duas filas de eventos (EVQ#1 e EVQ#2). A figura a seguir ilustra o método de geração destas filas de eventos.

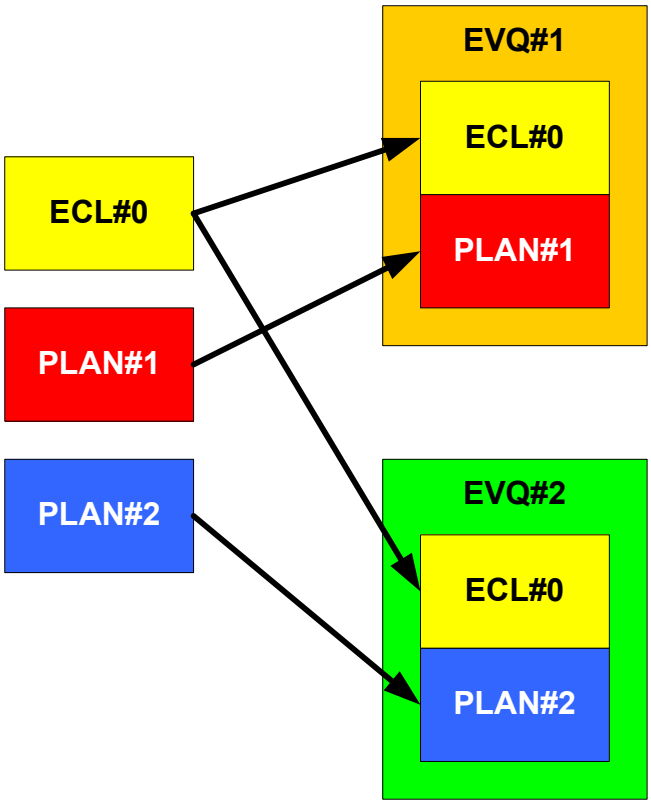


Figura 5.7 - Composição das filas de eventos de simulação

Ambas as filas baseiam-se na mesma previsão de eventos orbitais (ECL#0). Adicionam-se a elas um conjunto de telecomandos temporizados. A primeira fila (EVQ#1) contém agendamento especificado num dos planos de operações

(PLAN#1), enquanto a segunda fila (EVQ#2) é baseada no outro plano (PLAN#2).

5.6. Modelamento das regras de controle

O objetivo da simulação consiste em avaliar passo a passo a profundidade de descarga da bateria (DOD), com o intuito de verificar se ela não ultrapassa o limite estabelecido. Para a determinação do valor de DOD, são necessários cálculos de carga armazenada e balanço de potência. O conjunto de regras (RULE#0) que descreve este procedimento para o sistema exemplificado encontra-se expressa na tabela abaixo.

Tabela 5.3 - Lista de regras do sistema

#	Identifier	Description	Condition	Effect
1	*RDOD*	*DOD* Update	*TRUE*	*DOD = 1 - QBAT /QMAX*
2	*RQBAT*	*QBAT* Update	*TRUE*	*QBAT = MIN(QBAT + IBAT * SIMSTEP, QMAX)*
3	*RIBAT*	*IBAT* Update	*TRUE*	*IBAT = PBAT / VBAT*
4	*RVC*	*BAT* Charging	*PBAT >= 0*	*VBAT = VC*
5	*RVD*	*BAT* Discharging	*PBAT < 0*	*VBAT = VD*
6	*RPBAT*	*PBAT* Update	*TRUE*	*PBAT = PSAG - PBUS*
7	*RPBUS*	*PBUS* Update	*TRUE*	*PBUS = VBUS * IBUS* *IBUS = IPL1 + IPL2 + ISL*
8	*RONPL1*	*PL1* Power On	*ONPL1*	*SWPL1 = OFF* *IPL1 = IPL1ON*
9	*ROFFPL1*	*PL1* Power Off	*OFFPL1*	*SWPL1 = OFF* *IPL1 = 0*
10	*PONPL2*	*PL2* Power On	*ONPL2*	*SWPL2 = ON* *IPL2 = IPL2ON*
11	*ROFFPL2*	*PL2* Power Off	*OFFPL2*	*SWPL2 = OFF* *IPL2 = 0*
12	*RPSAG*	*PSAG* Update	*TRUE*	*PSAG = VSAG * ISAG*
13	*RECL*	Eclipse Phase Start	*ECL*	*ISAG = 0*
14	*RSUN*	Sun Phase Start	*SUN*	*ISAG = ISGMAX*
15	*RFULL*	BAT Full	*ISAG >0;* *QBAT/QMAX = 1*	*PBAT = 0* *ISAG = MIN((PBUS / VSAG, ISGMAX)*
16	*RNEXT*	Simulation Update	*TRUE*	*SIMTIME = SIMTIME +SIMSTEP*
17	*RSTOP*	Simulation Stop	*SIMTIME >= SIMEND*	*SIMSTOP = TRUE*

Fonte: Adaptado de Tominaga et al. (2008)

A primeira regra da tabela (RDOD) diz respeito ao cálculo da profundidade de descarga da bateria. A profundidade de descarga (DOD) é função da carga corrente da bateria (QBAT) e da carga máxima armazenável (QMAX). A carga

máxima armazenável (QMAX) é um valor de projeto. O valor da carga corrente (QBAT) muda conforme a condição de carga e descarga da bateria, indicada por sua corrente de carga (IBAT).

A carga corrente da bateria (QBAT) é dada pela integral no tempo da corrente de carga da bateria (IBAT). Este cálculo pode ser aproximado conforme a segunda regra (RQBAT), multiplicando-se a corrente de carga (IBAT) pelo intervalo de tempo correspondente ao tempo de passo da simulação (SIMSTEP). Este valor de carga (QBAT) aumenta ou decrementa conforme o valor da corrente (IBAT). O tempo de passo da simulação (SIMSTEP) é um parâmetro de controle cujo valor é definido pelo usuário condutor da simulação. Deve se observar que a carga corrente da bateria (QBAT) assume um valor intermediário entre zero e seu valor máximo (QMAX). Entretanto, nesta regra de modelo, a restrição de limite inferior da carga (QBAT > 0) não foi considerada. Esta implementação permite simplificar um pouco a estrutura da regra. Níveis exagerados de descarga da bateria não fazem parte de um cenário normal de operação em rotina, exceto em caso de uma falha grave em bordo. (INPE, 2007a)

O valor da corrente de carga (IBAT) da bateria é obtido segundo a terceira regra (RIBAT), dividindo-se a potência de carga da bateria (PBAT) pela tensão da bateria (VBAT). A tensão (VBAT) é obtida ou pela quarta regra (RVC) ou pela quinta (RVD), ao passo que a potência é fornecida pela sexta regra (RPBAT).

Por questão de simplicidade, neste modelo considerou-se que a tensão da bateria (VBAT) assume valores constantes distintos durante o processo de carga (RVC: VBAT = VC) e de descarga (RVD: VBAT = VD). O processo de carga (RVC) ocorre se a potência de carga da bateria for positiva (PBAT > 0), ou seja, há sobra de energia em bordo que pode ser direcionada à carga da bateria. Se houver equilíbrio entre consumo e suprimento (PBAT = 0), a tensão

da bateria é mantida (VBAT = VC). Já se houver falta de energia em bordo, a parte faltante terá que ser suprida por meio de remoção de carga bateria. Neste caso, tem-se o processo de descarga (RVD), durante o qual a potência de carga da bateria torna-se negativa (PBAT < 0).

A potência de carga da bateria (PBAT) é obtida a partir da equação de balanço de potência (RPBAT). Segundo esta equação, a potência de carga da bateria (PBAT) é igual à potência fornecida pelo gerador (PSAG) subtraído da potência consumida no barramento (PBUS). O cômputo da potência do barramento (PBUS) é realizado por meio da sétima regra (RPBUS) e, a do gerador (PSAG), pela 12ª regra (RPSAG).

A potência consumida no barramento (PBUS) é dada pela tensão no barramento (VBUS) multiplicada pela corrente no barramento (IBUS). A tensão no barramento regulado (VBUS) é um valor de projeto e considerada constante. A corrente no barramento (IBUS) é dada pela soma das correntes consumidas pelas cargas-úteis (IPL1 e IPL2) e pelos equipamentos de serviço (ISL).

Neste modelo simplificado, o valor da corrente de serviço (ISL) é considerado constante. Já as correntes consumidas pelas cargas-úteis (IPL1 e IPL2) mudam de acordo com a execução de telecomandos (ONPL1, OFFPL1, ONPL2 e OFFPL1), conforme explicitadas pelas regras numeradas de oito a onze (RONPL1, ROFFPL1, RONPL2 e ROFFPL2). Os tempos de execução destes telecomandos encontram-se explicitados nos planos de operações.

O valor da corrente de alimentação da carga-útil PL1 (IPL1) é determinado pelo estado da chave SWPL1, conforme a oitava e a nona regras. O telecomando ONPL1 leva a chave da carga-útil PL1 à posição liga (SWPL1 = ON). Como conseqüência, a corrente de consumo desta carga-útil (IPL1) assume o valor especificado em projeto (IPL1ON). O telecomando OFFPL1 faz o contrário,

levando a chave SWPL1 à posição desliga (SWPL1 = OFF). Neste caso, a corrente de consumo (IPL1) cai para zero.

Analogamente, as regras de número 10 e 11 definem o estado da chave SWPL2, que determina o valor da corrente de alimentação da carga-útil PL2 (IPL2). O telecomando ONPL2 faz a corrente de consumo subir (IPL2 = IPL2ON) e o seu par OFFPL2 a faz baixar (IPL2 = 0).

Conforme a 12ª regra (RPSAG), o valor da potência fornecida pelo gerador (PSAG) é dado pelo produto da tensão no gerador (VSAG) pela corrente gerada por ele (ISAG). A tensão no gerador (VSAG) é um valor de projeto considerado constante. Já a corrente do gerador (ISAG) depende dos estados de sombra (ECL) ou iluminação (SUN) do gerador solar, e também do estado de carga da bateria (QBAT/QMAX). Estas dependências são descritas respectivamente pelas regras de número treze, quatorze e quinze.
Dependendo da sua configuração orbital, um satélite orbitando ao redor da Terra poderá, em um dado instante de tempo, estar posicionado num ponto que lhe possibilita estar iluminado diretamente pelo Sol, ou ofuscado pela Terra, em situação de eclipse. Os instantes de início das fases iluminadas e de eclipse (SUN e ECL) são eventos orbitais que podem ser calculados e fornecidos pela equipe de dinâmica de vôo.

Quando o satélite entrar em eclipse (ECL), o painel solar deixará de estar iluminado. Neste caso, o gerador torna-se incapaz de produzir energia, portanto toda energia demandada deverá ser suprida pela bateria, que entra naturalmente em modo de descarga. A corrente do gerador (ISAG) nesta situação assume o valor zero, conforme explicitado na 13ª regra (RECL).

Ao ser iluminado (SUN), o painel solar torna-se capaz de suprir energia, fornecendo uma corrente de gerador (ISAG) cujo valor pode atingir até um máximo de projeto (ISGMAX). O fornecimento de corrente máxima (ISAG

ISGMAX) ocorre principalmente durante o início do processo de carga da bateria, durante o qual a energia fornecida pela bateria em descarga é reposta (QBAT/QMAX < 1). Nesta situação, a corrente do gerador (ISAG) assume um valor máximo constante (ISGMAX), conforme diz a 14ª regra (RSUN).

Enquanto o painel solar estiver iluminado (ISAG > 0) e a bateria plenamente carregada (QBAT/QMAX = 1), o suprimento de carga da bateria é cortado (PBAT = 0) e a corrente do gerador (ISAG) assume um valor que procura satisfazer a equação de balanço de potência. Se o consumo de potência do barramento for menor ou igual à potência máxima que pode ser suprida pelo gerador (PBUS <= VSAG*ISGMAX), então a corrente do gerador é ajustada para o valor de equilíbrio de carga (ISAG = PSAG/VSAG = PBUS/VSAG). Porém, dependendo do consumo no barramento, o gerador é forçado a operar em carga máxima do gerador (PBUS > VSAG*ISGMAX), o gerador passa a fornecer potência máxima (PSAG = VSAG*ISGMAX) e a parte faltante é extraída da bateria. Em outras palavras, a corrente do gerador (ISAG) é o menor valor entre a corrente máxima permitida (ISGMAX) e a corrente de equilíbrio de potência. Esta situação é descrita por meio da 15ª regra (RFULL).

Todas as regras acima devem ser executadas uma vez a cada novo passo em uma sessão de simulação. A execução de uma regra consiste em verificar a sua condição de disparo, para aplicar ou não seus efeitos conforme o resultado da avaliação desta condição. O controle dos passos de sessão de simulação é feito também por meio de regras, mais especificamente aquelas numeradas de 16 (RNEXT) e 17 (RSTOP).

A cada renovação de passo de uma sessão de simulação (RNEXT), o tempo de simulação (SIMTIIME) é incrementado de um tempo de passo (SIMSTEP). Os parâmetros de tempo de simulação (SIMTIME) e de passo (SIMSTEP) fazem parte da configuração da simulação, devendo ser definidos pelo usuário

antes do início da sessão. Neste caso, o tempo de simulação (SIMTIME) corresponde ao tempo de início da sessão.

A condição de parada (RSTOP) é determinada por um sinalizador de parada (SIMSTOP) que é ativado quando o tempo de simulação (SIMTIME) atinge o tempo final (SIMEND). A atribuição de valor verdadeiro ao parâmetro sinalizador de parada (SIMSTOP) leva à finalização da sessão de simulação. Este mecanismo será detalhado mais adiante, em outro Capítulo.

Desta forma, foi definido um conjunto de regras necessário para o cálculo da profundidade de descarga da bateria. Este conjunto de regras permite o levantamento da lista de parâmetros necessários para alimentar o modelo do simulador.

A tabela abaixo lista o conjunto de telemetrias e telecomandos do sistema, cujos valores caracterizam um subconjunto dos parâmetros de simulação necessários para caracterizar o modelo.

Tabela 5.4 - Lista de telemetrias e telecomandos associados a parâmetros

Identifier	Description	Type	Remark
ISAG	*SAG* Current	Status	*ISAG* = 0 if satellite not illuminated by sun.
VBAT	*BAT* Voltage	Status	*VBAT* = *VC* if charging, *VBAT* = *VD* if discharging.
VBUS	Main Bus Voltage	Status	*VBUS* = constant.
IBUS	Main Bus Current	Status	*IBUS* = *IPL1*+ *IPL2*+ *ISL*.
IPL1	*PL1* Input Current	Status	*IPL1* = 0 if *SWPL1* = OFF, *IPL1*= *IPL1ON* otherwise.
IPL2	*PL2* Input Current	Status	*IPL2* = 0 if *SWPL2* = OFF, *IPL2*= *IPL2ON* otherwise.
SWPL1	*PL1* Power Switch Status	Status	*SWPL1* = ON or OFF.
SWPL2	*PL2* Power Switch Status	Status	*SWPL2* = ON or OFF.
ONPL1	*PL1* Power On	Event	*SWPL1* becomes ON.
ONPL2	*PL2* Power On	Event	*SWPL2* becomes ON.
OFFPL1	*PL1* Power Off	Event	*SWPL1* becomes OFF.
OFFPL2	*PL2* Power Off	Event	*SWPL2* becomes OFF.

Fonte: Adaptado de Tominaga et al. (2008)

Conforme observado a partir do conjunto de regras, a lista de parâmetros acima é insuficiente para descrever completamente o modelo do simulador. A

tabela abaixo apresenta o conjunto dos parâmetros de simulação adicionais necessários para compor o modelo.

Tabela 5.5 - Lista de parâmetros adicionais para caracterização do modelo

Identifier	Description	Type	Remark
DOD	*BAT* Depth-Of-Discharge	Status	*DOD* = 1 - *QBAT* /*QMAX*
QBAT	*BAT* Charge	Status	*QBAT* = sum over time of *IBAT*
QMAX	Maximum *BAT* Charge	Status	*QMAX* = constant, 0 < *QBAT* <= Q*MAX*
IBAT	*BAT* Charging Current	Status	*IBAT* = *PBAT* / *VBAT*
PBAT	*BAT* Stored Power	Status	*PBAT* = *PBUS* - *PSAG*
VC	*BAT* Charging Mean Voltage	Status	*VC* = constant
VD	*BAT* Discharging Mean Voltage	Status	*VD* = constant
PSAG	*SAG* Power	Status	*PSAG* = *VSAG* * *ISAG*
VSAG	*SAG* Voltage	Status	*VSAG* = constant
ISGMAX	Maximum *SAG* Current	Status	*ISGMAX* = constant, 0 < *ISAG* <= *ISGMAX*
ISL	*SL* Input Current	Status	*ISL* = constant
IPL1ON	*PL1* Operating Input Current	Status	*IPL1ON* = constant
IPL2ON	*PL2* Operating Input Current	Status	*IPL2ON* = constant
PBUS	Total Load Power at Main Bus	Status	*PBUS* = *VBUS* * *IBUS*
SUN	Sun Illuminated Phase	Event	Satellite becomes illuminated by the sun
ECL	Eclipse Phase	Event	Satellite becomes shadowed by the earth
SIMTIME	Simulation Current Time	Status	Simulation time counter
SIMSTEP	Simulation Step Time	Status	*SIMTIME* = *SIMTIME* + *SIMSTEP*
SIMEND	Simulation End Time	Status	*SIMEND* >= *SIMTIME*
SIMSTOP	Simulation Stop Condition	Event	Close session; exit simulator

Fonte: Adaptado de Tominaga et al. (2008)

5.7. Definição do estado inicial

Antes de se iniciar uma sessão de simulação, um conjunto de parâmetros compondo o estado inicial deve ser alimentado e eventos temporizados carregados na forma de fila de eventos.

O estado in inicial caracteriza o estado interno do sistema simulado no instante de tempo correspondente ao início da sessão de simulação. A tabela a seguir mostra valores de inicialização atribuídos aos parâmetros de estado que constituem do estado inicial (INIT#0_ST).

Tabela 5.6 - Lista de parâmetros de estado componentes do estado inicial

Identifier	Description	Value	Unit	Remark
ISAG	*SAG* Current	0	Amperes	*ISAG* = 0 if satellite not illuminated by sun
VBAT	*BAT* Voltage	18.5	Volts	*VBAT* = *VC* if charging, *VBAT* = *VD* if discharging
VBUS	Main Bus Voltage	26.5	Volts	*VBUS* = constant
IBUS	Main Bus Current	0.8	Amperes	*IBUS* = *IPL1*+ *IPL2*+ *ISL*
IPL1	*PL1* Input Current	0	Amperes	*IPL1* = 0 if *SWPL1* = OFF, *IPL1*= *IPL1ON* otherwise
IPL2	*PL2* Input Current	0	Amperes	*IPL2* = 0 if *SWPL2* = OFF, *IPL2*= *IPL2ON* otherwise
SWPL1	*PL1* Power Switch Status	OFF	N/A	*SWPL1* = ON or OFF
SWPL2	*PL2* Power Switch Status	OFF	N/A	*SWPL2* = ON or OFF
DOD	*BAT Depth-Of-Discharge*	0.0367	-	*DOD = 1 - (QBAT /QMAX)*
QBAT	*BAT Charge*	6.936	Ampere-hour	*QBAT = sum over time of IBAT*
QMAX	*Maximum BAT Charge*	7.2	Ampere-hour	*QMAX = constant, 0 < QBAT <= QMAX*
IBAT	*BAT Charging Current*	-1.146	Amperes	*IBAT = PBAT / VBAT*
PBAT	*BAT Stored Power*	-21.2	Watts	*PBAT = PBUS - PSAG*
VC	*BAT Charging Mean Voltage*	21	Volts	*VC = constant*
VD	*BAT Discharging Mean Voltage*	18.5	Volts	*VD = constant*
PSAG	*SAG Power*	0	Watts	*PSAG = VSAG * ISAG*
VSAG	*SAG Voltage*	30	Volts	*VSAG = constant*
ISGMAX	*Maximum SAG Current*	1.6	Amperes	*ISGMAX = constant, 0 < ISAG <= ISGMAX*
ISL	*SL Input Current*	0.8	Amperes	*ISL = constant*
IPL1ON	*PL1 Operating Input Current*	0.2	Amperes	*IPL1ON = constant*
IPL2ON	*PL2 Operating Input Current*	5	Amperes	*IPL2ON = constant*
PBUS	*Total Load Power at Main Bus*	21.2	Watts	*PBUS = VBUS * IBU*
SIMTIME	*Simulation Current Time*	0	Day	*Simulation time counter*
SIMSTEP	*Simulation Step Time*	30	Seconds	*SIMTIME = SIMTIME + SIMSTEP*
SIMEND	*Simulation End Time*	1	Day	*SIMEND >= SIMTIME*

Fonte: Adaptado de Tominaga et al. (2008)

A fila de eventos contém telecomandos extraídos do plano de operações em vôo e eventos contidos na previsão de eventos orbitais. A definição de parâmetros de eventos utilizados no modelo é realizada no estado inicial. A tabela a seguir lista os parâmetros de eventos definidos no estado inicial (INIT#0_EV). Conforme pode se observar, o valor de inicialização de cada parâmetro de evento é igual a zero.

Tabela 5.7 - Lista de parâmetros de evento componentes do estado inicial

Identifier	Description	Value	Unit	Remark
ONPL1	*PL1* Power On	0	N/A	*SWPL1* becomes ON
ONPL2	*PL2* Power On	0	N/A	*SWPL2* becomes ON
OFFPL1	*PL1* Power Off	0	N/A	*SWPL1* becomes OFF
OFFPL2	*PL2* Power Off	0	N/A	*SWPL2* becomes OFF
SUN	Sun Illuminated Phase	0	N/A	Satellite becomes illuminated by the sun
ECL	Eclipse Phase	0	N/A	Satellite becomes shadowed by the earth
SIMEND	Simulation End Time	0	N/A	Close session; exit simulator

Fonte: Adaptado de Tominaga et al., (2008)

O estado inicial (INIT#0) é formado a partir da composição dos valores iniciais dos parâmetros de estado (INIT#0_ST) e de evento (INIT#0_EV), conforme ilustrado na figura abaixo.

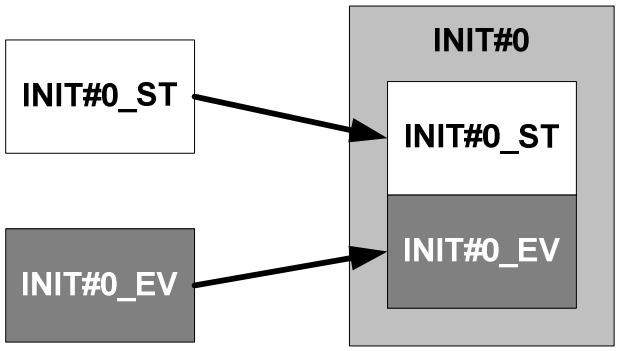


Figura 5.8 - Composição do estado interno de simulação

5.8.Análise da saída para geração de diagnóstico

Ao final de cada sessão de simulação, obtém-se um histórico de estados que descreve o comportamento de todos os parâmetros de simulação durante todos os passos que compõem a sessão. O gerador de diagnóstico lê este histórico de estados e gera um parecer de aceitação ou rejeição do plano de operações em vôo.

O requisito de operação deste exemplo especifica que o DOD da bateria deve ser mantido abaixo de 20%. O gerador de diagnóstico deve, portanto, aceitar planos de operações que mantenham o DOD sempre abaixo de 0.2, e rejeitar

aqueles que resultem num DOD acima deste valor em algum instante da simulação.

As execuções do simulador utilizando filas de eventos (EVQ#1 e EVQ#2) baseadas nos dois planos (PLAN#1 e PLAN#2) resultaram na saída de dois históricos de estados (OUT#1 e OUT#2), cujos comportamentos de valores de DOD encontram-se apresentadas graficamente na figura a seguir.

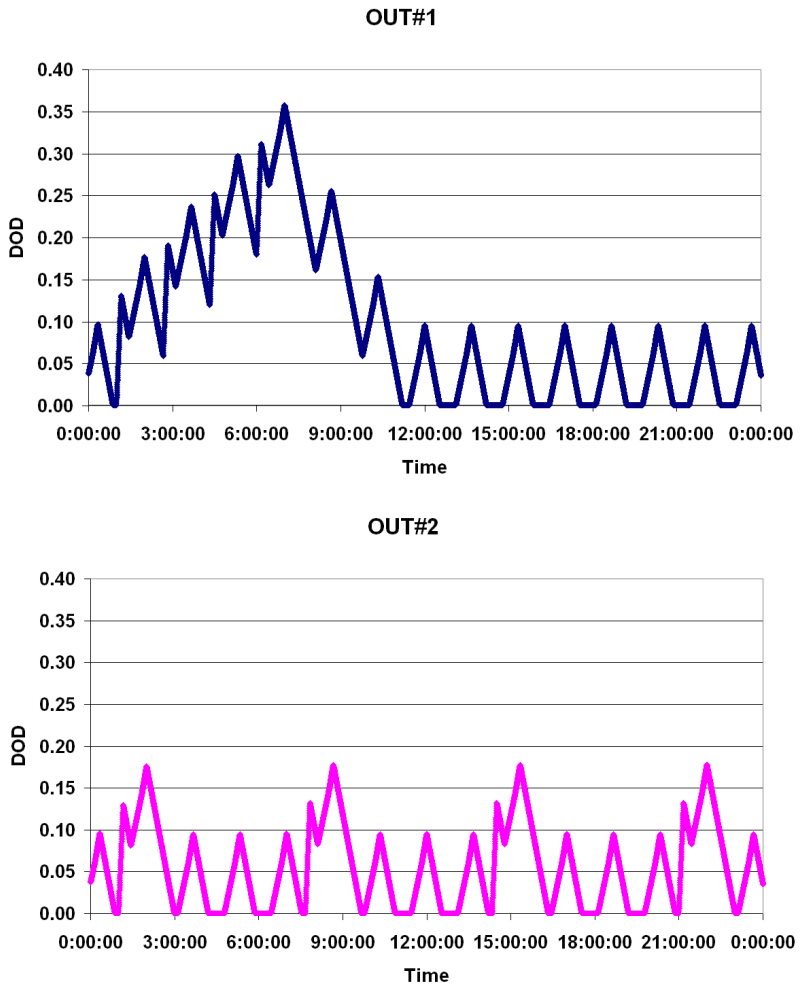


Figura 5.9 - Profundidades de descarga da bateria obtidas nas duas sessões

Desta forma, segundo o critério de diagnóstico baseado no DOD, o primeiro plano (PLAN#1) deve ser rejeitado, e o segundo plano (PLAN#2), aprovado.

5.9. Um exemplo mais complexo

A tensão de uma bateria é um parâmetro importante que indica o seu nível corrente de carga. Sua variação durante os ciclos de carga e descarga, ao longo de sua vida útil, fornece um quadro de seu estado de degradação. No modelo simplificado, a tensão de bateria assume dois níveis de valores, com um valor alto durante a carga, e outro valor mais baixo durante descarga. Modelos mais realistas de tensões de bateria associam modelos de resistência interna da bateria e comportamento capacitivo para o cálculo de tensões de carga e descarga. (AMBROSIO et al, 2010) (CHEN; RINCÓN-MORA, 2006) (MANZO et al., 2001)

A figura abaixo mostra duas curvas de tensão de bateria ao longo de certo intervalo de tempo. A curva em amarelo corresponde a valores simulados segundo um modelo simplificado baseado nas regras utilizadas para descrever o modelo exemplificado neste Capítulo. A curva em rosa corresponde a valores de telemetria recebidos de um satélite real durante o período simulado. Descontinuidades nesta curva representam períodos sem dados, durante o qual o satélite esteve fora de visada de estações de rastreio.

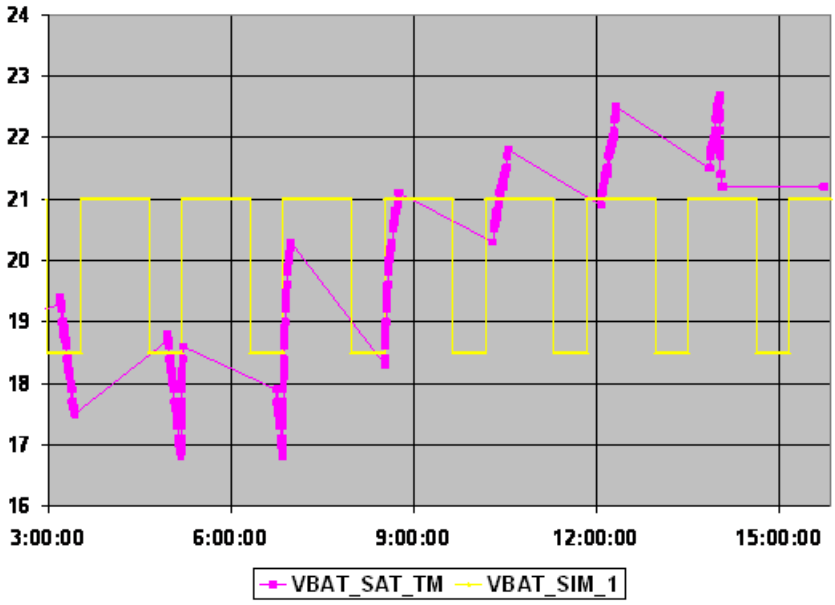


Figura 5.10 - Comparação de curvas de tensão de bateria em função do tempo

A tensão de uma bateria real é ligada intimamente com o seu estado de carga, conforme ilustrado na figura a seguir. A curva da tensão em função da capacidade de carga apresenta um comportamento não-linear.

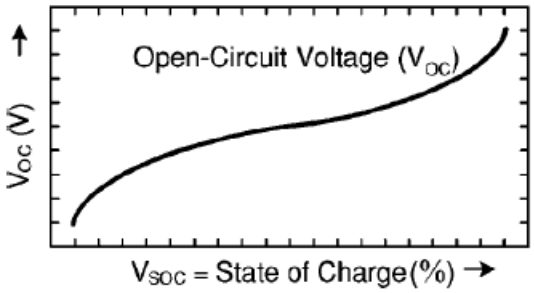


Figura 5.11 - Curva de tensão típica de bateria em função da carga
Fonte: Chen e Rincón-Mora (2006)

Desta forma, um modelo mais realista para o simulador requer a representação de uma função não-linear da tensão em função da carga da bateria. Para este tipo de problema, fabricantes e projetistas costumam fornecer curvas de calibração, que representam pontos de amostra medidos em laboratório, a partir dos quais a função pode ser reconstituída por interpolação ou extrapolação.

A figura a seguir mostra uma curva de calibração criada para representar a tensão de uma célula componente da bateria em função da carga, para uso em um modelo melhorado para o simulador de satélite baseado em regras.

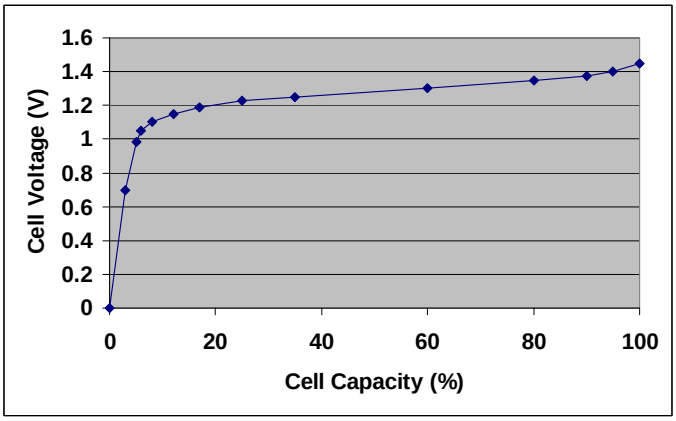


Figura 5.12 - Curva da função de tensão de uma célula de bateria em função da carga

A figura abaixo mostra as curvas de tensão da bateria obtidas a partir do modelo original e do modelo melhorado utilizando a curva de calibração. O acompanhamento do comportamento da tensão da bateria obtida através do simulador melhorado, em azul, comparada à telemetria, em magenta, é visivelmente melhor do que a saída do simulador antigo, em amarelo.

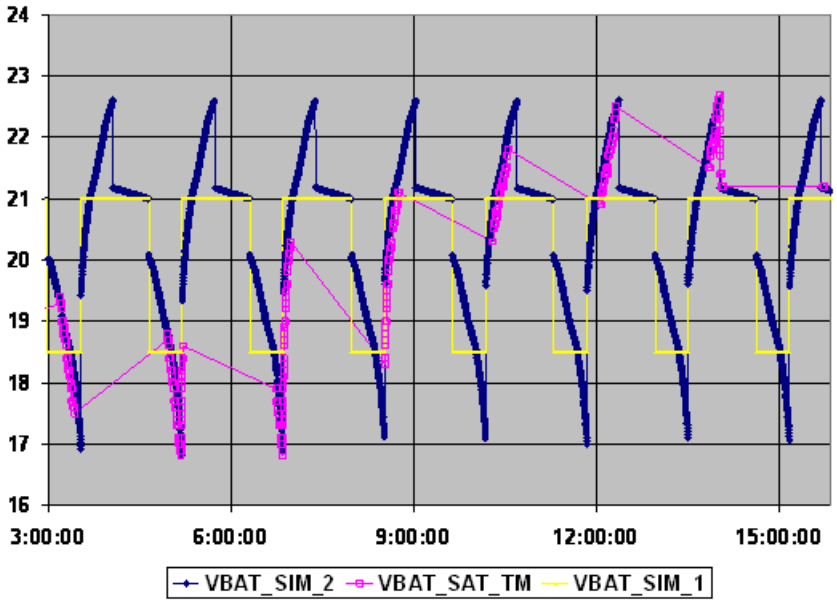


Figura 5.13 - Comparação de curvas de tensão de bateria em função do tempo

Um modelo ainda mais fiel do comportamento de carga da bateria levaria em consideração ainda uma componente térmica, cuja implementação não foi realizada devido à complexidade do modelamento. (MANZO et al., 2001)

No Capítulo seguinte, serão analisadas de forma um pouco mais aprofundada as estruturas de dados adotadas pelo simulador para a representação da informação contida no seu modelo.

6 ESTRUTURAS DE DADOS

Neste Capítulo, as estruturas de dados processados pelo simulador serão descritos em um nível maior de detalhamento. Este detalhamento deve permitir a implementação do banco de dados do simulador, que será tratado no Capítulo seguinte.

6.1. Visão geral

O simulador de satélite consiste numa máquina de inferência baseada em regras, composta por um núcleo de processamento associado a um banco de dados que contém o modelo de sistema específico de cada missão a ser simulada. Este banco de dados, gerado a partir da configuração do modelo do sistema, contém as estruturas de dados utilizadas pelo simulador.

Estas estruturas de dados incluem parâmetros de estado que caracterizam o estado interno do sistema, parâmetros de evento que são instanciadas na fila de eventos e que causam perturbações no estado interno, e um conjunto de regras que controlam a dinâmica do sistema. No final do Capítulo anterior, foi introduzida uma nova estrutura de dados a ser incluída no modelo do simulador, a partir da necessidade de se processarem curvas de funções. A nova arquitetura resultante, proposta para a verificação de planos de operações por meio de simuladores de satélites, encontra-se ilustrado na figura a seguir.

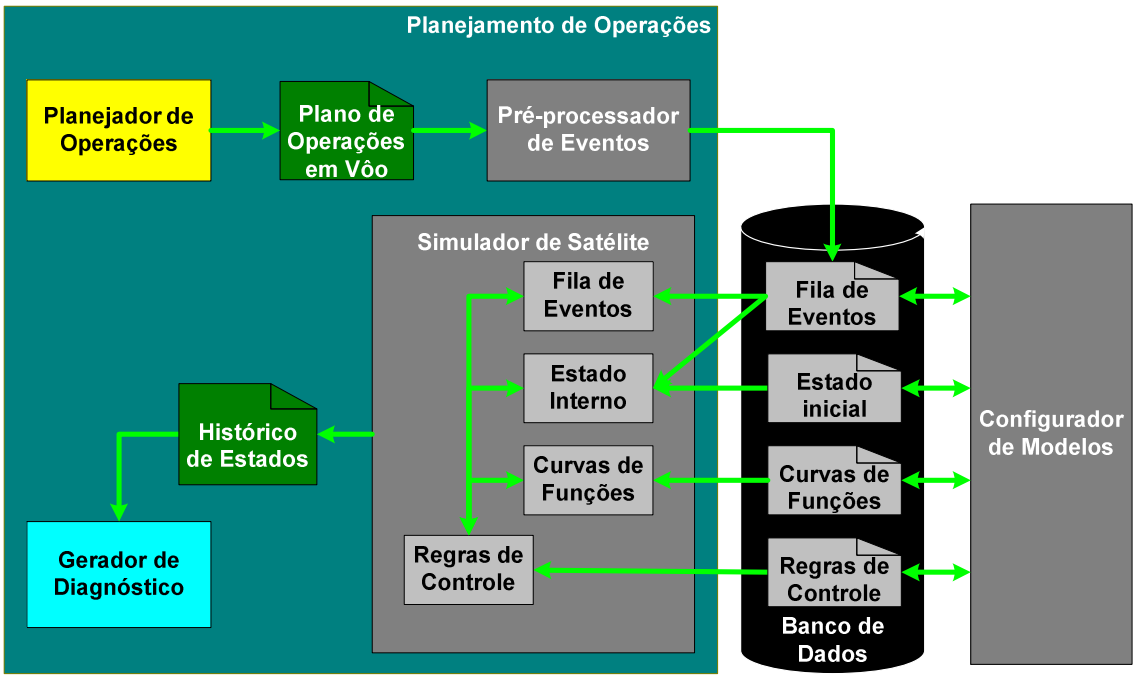


Figura 6.1 - Arquitetura do sistema de simulação para verificação de planos

O estado interno do simulador em um dado instante de tempo é representado por meio de valores assumidos por um conjunto de parâmetros de estado. O estado inicial informa os valores destes parâmetros no instante de tempo correspondente ao início da simulação. O estado interno muda dinamicamente ao longo da sessão de simulação, conforme disparos de eventos e avaliações de regras, que ocorrem a cada atualização do tempo de simulação.

A fila de eventos contém a agenda de eventos pré-programados para serem disparados durante a sessão de simulação. Como o objetivo consiste na simulação de execução de planos de operações, não há necessidade de processar eventos externos disparados em tempo de simulação. A leitura das ações a serem executadas a partir do plano de operações e dos eventos externos previstos permite a criação de uma lista temporizada destas ocorrências. A associação entre eventos e seus efeitos é realizada por meio de regras de controle.

As regras de controle codificam as relações de causa e efeito que regem a dinâmica do estado interno do sistema simulado. Uma regra é composta por uma condição e um ou mais efeitos. Dependendo do resultado da avaliação da condição, resultados de efeitos podem ser aplicados ou não sobre seus respectivos parâmetros alvos. Expressões de condições e efeitos consistem em seqüências de operandos e operações, cujas avaliações resultam em valores numéricos. Expressões de regras serão detalhadas mais à frente.

Curvas de funções são utilizadas para o mapeamento de valores, cuja representação seria difícil ou impossível por meio de expressões. Elas retornam um valor de saída calculado de acordo com o valor fornecido para a entrada, conforme uma tabela de pontos de curvas. Os pontos de curvas têm como finalidade armazenar os conjuntos de pontos que compõem as curvas de funções. Uma curva de função é definida de acordo com uma tabela de pontos. Cada ponto de uma curva corresponde a um par de valores numéricos reais de entrada e saída. Se a entrada da curva for correspondente ao valor de um ponto tabelado, então a saída será dada pela saída deste ponto. Caso contrário, o valor da saída será calculado por meio de interpolação ou extrapolação linear.

6.2. Regras de controle

No simulador de satélite proposto neste trabalho, o modelo da dinâmica interna do sistema simulado é descrito por meio de regras lógicas. A avaliação destas regras lógicas leva à atualização do estado interno sendo, portanto, responsável pelo controle de toda a simulação.

Cada regra lógica é composta por uma expressão de condição que, caso seja satisfeita, leva à aplicação de um ou mais efeitos. Cada efeito é constituído por um parâmetro alvo e uma expressão. Dependendo do resultado da avaliação da expressão de condição, efeitos podem ser aplicados ou não. A aplicação

completa de um efeito consiste na substituição do valor do parâmetro alvo pelo resultado da avaliação da expressão de efeito. Isto ocorre quando o resultado da avaliação da expressão de condição resulta no valor numérico 1, o que corresponde ao valor lógico VERDADEIRO. A não-aplicação de um efeito consiste na manutenção do valor corrente do parâmetro alvo, independentemente do resultado da avaliação da expressão de efeito. Isto ocorre quando o resultado da avaliação da expressão de condição leva à obtenção do valor numérico 0, correspondente ao valor lógico FALSO. Aplicações parciais de efeitos são determinados por valores intermediários entre 0 e 1, resultantes da avaliação da expressão de condição. Neste caso, a aplicação de efeito é ponderada entre manutenção do valor corrente e substituição total pelo resultado da expressão de efeito, conforme o valor resultante da avaliação da condição.

A figura abaixo ilustra um exemplo de regra, onde se explicitam seus componentes. Na regra representada pela figura, LIGAR AQUECEDOR é um parâmetro de evento. Ele assume valor VERDADEIRO apenas nos passos de simulação correspondentes ou imediatamente após os instantes programados na fila de eventos, mantendo-se FALSO durante o restante do tempo. Se este evento for disparado enquanto o valor do parâmetro de estado de AQUECEDOR ALIMENTADO corresponder ao valor VERDADEIRO, então a condição é satisfeita. Neste caso, todos os efeitos da regra são aplicados no mesmo passo de simulação. O valor correspondente ao estado corrente do parâmetro AQUECEDOR LIGADO torna-se VERDADEIRO e o valor de CONTADOR DE COMANDO é incrementado de um. Mas se a condição não for satisfeita, seja pelo fato de o evento LIGAR AQUECEDOR não ter sido disparado, seja pelo fato do estado AQUECEDOR ALIMENTADO não corresponder ao valor VERDADEIRO, ou mesmo ambos, então neste caso a regra não será executada.

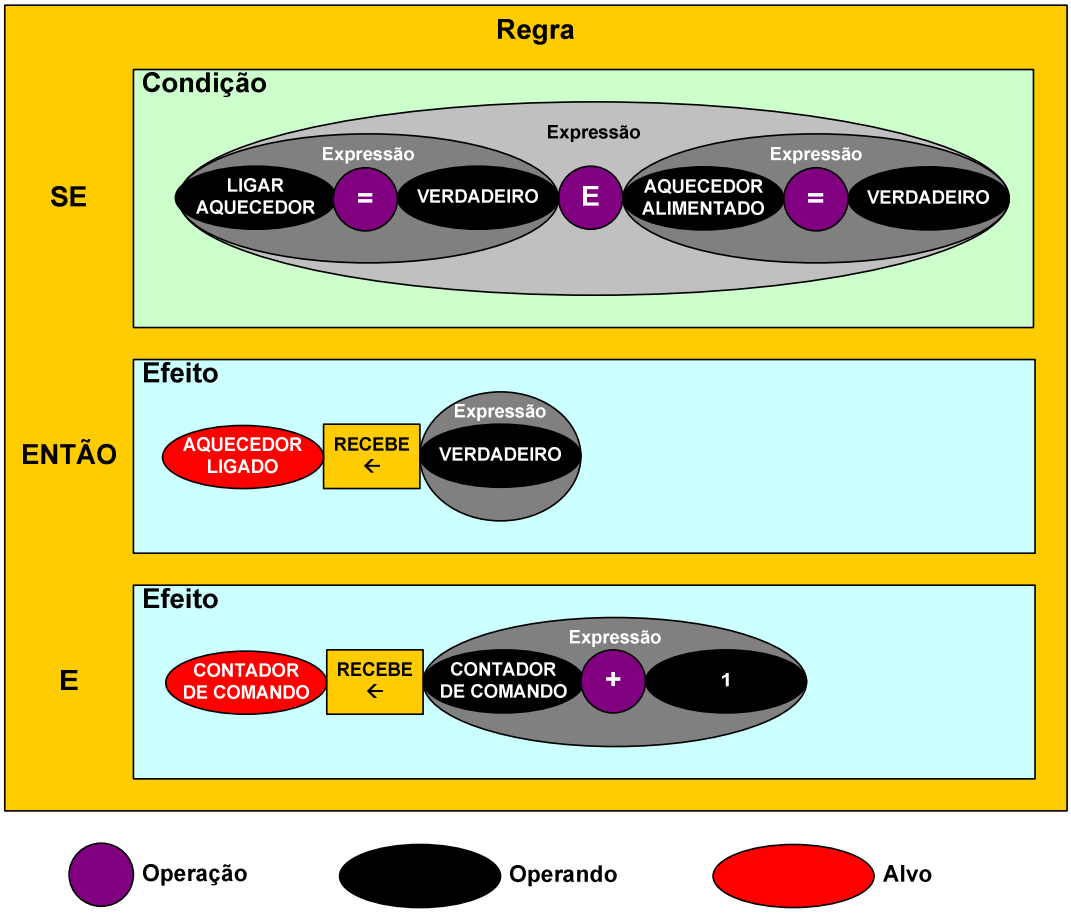


Figura 6.2 - Composição de uma regra lógica

6.3. Expressões de regras

Elementos que compõem uma expressão podem ser subdivididos em operandos e operações. Uma expressão na sua forma mais simples é composta por um único operando. Operandos de expressões podem ser identificadores associados a valores ou estados de parâmetros, constantes numéricas, ou expressões aninhadas. Operações de expressões incluem operações lógicas, aritméticas e funções. O seqüenciamento de operandos e operações definem o tipo de saída da expressão. A avaliação de uma expressão numérica retorna um valor numérico, e a de uma expressão lógica retorna um valor lógico. Exemplos de expressões são listados na tabela a seguir.

Tabela 6.1 - Exemplos de expressões

Expressão	Tipo	Operandos	Operações	Sub-Expressões
X OU (Y < Z)	Lógica	X, Y, Z	OU, <	(Y < Z)
(X OU Y) < Z	Lógica	X, Y, Z	OU, <	(X OU Y)
((A + B) = 0) OU ((C ^ 2) < (EXP(D / 3)))	Lógica	A, B, 0, C, 2, D, 3	+, =, OU, ^, EXP, /	(A + B)
				(C ^ 2)
				(D / 3)
				(EXP(D / 3))
				((A + B) = 0)
				((C ^ 2) < (EXP(D / 3)))
LOG(ABS(-2))	Numérica	-2	LOG, ABS	ABS(-2)
(LOG(ABS))-2	Numérica	ABS, 2	LOG, -	(LOG(ABS))
X É QUENTE	Lógica nebulosa	X, QUENTE	É	

Expressões de regras podem ser representadas através de operandos, operações e sub-expressões. Sub-expressões nada mais são do que expressões aninhadas. As duas primeiras expressões na tabela acima são compostas pela mesma seqüência de operandos e operações, diferindo entre si apenas na ordem em que as operações são aplicadas. Esta ordenação, ou priorização das operações, é o que define as sub-expressões. A terceira expressão representa um caso mais complexo, onde sub-expressões encontram-se dentro de sub-expressões. De forma geral, sub-expressões mais internas devem ser processadas prioritariamente ou, em outras palavras, operações cercadas por um número maior de parênteses devem ser avaliadas primeiro. Uma maneira de representar isto consiste em associar um valor numérico inteiro às operações, que identifique a prioridade de cada operação a partir da profundidade que ocupa dentro da estrutura de expressões aninhadas. A figura abaixo ilustra a atribuição de profundidades para operações que compõem a terceira expressão. Expressões com profundidade maior são avaliadas antes.

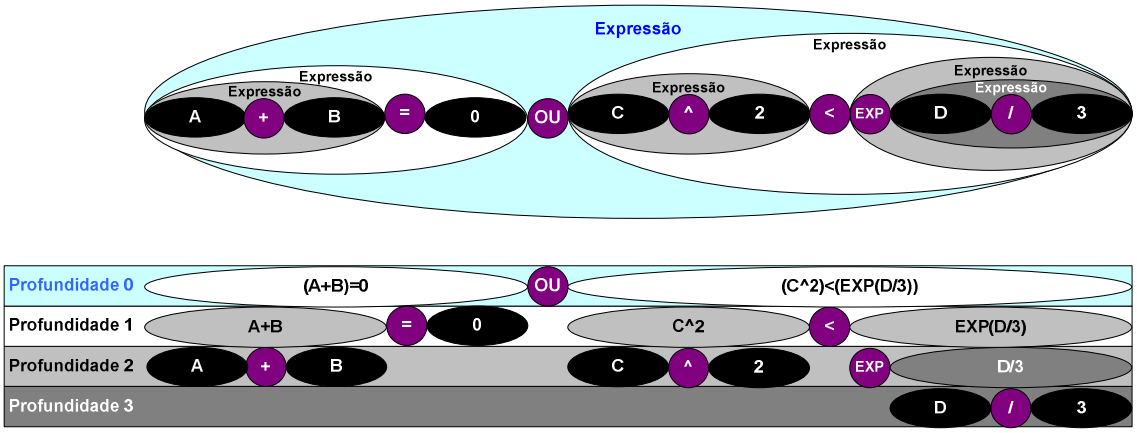


Figura 6.3 - Atribuição de profundidade a operações em uma expressão aninhada

Na quarta e quinta expressões da tabela anterior, os parênteses indicam mais do que apenas profundidade de operações e sub-expressões. Na quarta expressão, a função LOG é aplicada sobre o resultado da sub-expressão resultante da aplicação da função ABS sobre o valor numérico -2. Já na quinta expressão, a função LOG é aplicada sobre o valor de um parâmetro identificado por ABS, cujo resultado é subtraído do valor 2. Neste caso, o aninhamento da expressão evidenciou a existência de operações e operandos identificados pelo mesmo nome. A fim de evitar ambigüidades, uma expressão deve representar operandos e operações de forma inequívoca.

A sexta expressão representa uma expressão lógica diferente. Tradicionalmente, expressões lógicas utilizam a lógica booleana, segundo a qual sua avaliação pode retornar um de dois possíveis valores, ou VERDADEIRO, ou FALSO. No caso da lógica nebulosa, o resultado da avaliação da expressão pode assumir qualquer valor intermediário entre estes dois valores, inclusive. Por exemplo, um resultado de avaliação da expressão exatamente à metade entre os valores lógicos VERDADEIRO e FALSO representa uma situação em que a asserção X-É-QUENTE é simultaneamente VERDADEIRO e FALSO, na proporção de um para um. Este tipo de resultado se aplica a um valor de temperatura que, perguntado a certo número de

pessoas, levaria metade a responder que está quente, e a outra metade responder que não está.

6.4. Lógicas de controle

O simulador proposto é uma máquina de inferência, onde a decisão acerca da aplicação ou não de efeitos se dá conforme o resultado da avaliação de expressões de condição. A lógica que rege este mecanismo de condicionamento pode seguir a abordagem tradicional, baseado na asserção de expressões lógicas que resultam em um de dois valores possíveis, VERDADEIRO ou FALSO. Numericamente, atribui-se o valor inteiro 1 para representar o valor lógico VERDADEIRO, e 0 para representar FALSO. Esta representação define a lógica tradicional, conhecida também por lógica booleana.

Entretanto, podem ocorrer casos onde há interesse em representar situações com condições incertas. Situações deste tipo podem ser observadas quando condições são definidas por estados qualitativos, ao invés de quantitativos, de um ou mais parâmetros. Esta abordagem pode representar de forma mais adequada o raciocínio humano do que a lógica binária. (JANTZEN, 1998)

A figura abaixo mostra um exemplo de parâmetro com três estados qualitativos associados.

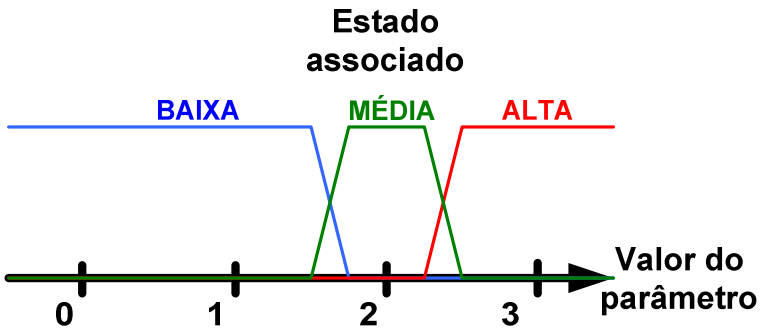


Figura 6.4 - Exemplo de associação de estados qualitativos a um parâmetro

Na figura anterior, o parâmetro em questão é classificado como BAIXO quando assume valores muito abaixo de 2, MÉDIO para valores próximos de 2, e ALTO para valores muito acima de 2. Para valores entre 1,75 e 2,25, este parâmetro encontra-se num estado classificado indiscutivelmente como MÉDIO. À medida que seu valor se afasta desta região, esta classificação torna-se cada vez menos apropriada. Entre 1,5 e 1,75 ele se torna MÉDIO tendendo a BAIXO, e entre 2,25 e 2,5 ele se torna MÉDIO tendendo a ALTO. Abaixo de 1.5 o parâmetro é puramente BAIXO e, acima de 2.5, puramente ALTO. Uma possível solução para representar uma situação deste tipo pode ser alcançada através do emprego de lógica nebulosa em substituição à lógica booleana.

A lógica nebulosa é uma ferramenta de inteligência artificial que permite o modelamento de tomadas de decisões baseadas em informações incertas. Nelas, resultados de avaliação de expressões lógicas assumem valores reais intermediários entre 0 e 1. A figura abaixo ilustra uma máquina de inferência baseada em lógica nebulosa.

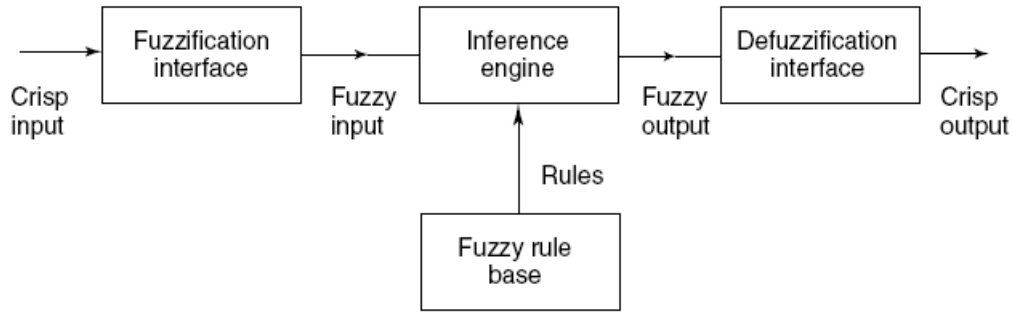


Figura 6.5 - Arquitetura de uma máquina de inferência nebulosa
Fonte: Abraham (2005)

Na figura, uma base de regras nebulosas (*fuzzy rule base*) controla a máquina de inferência (*inference engine*). Suas entradas e saídas são nebulosas (*fuzzy input* / *fuzzy output*). Interfaces de nebulização e desnebulização (*fuzzification interface* / *defuzzification interface*) são necessárias a fim de se trabalhar com entradas e saídas não-nebulosas, ou cristalinas (*crisp input* / *crisp output*).

A fim de ilustrar melhor a descrição acima, consideremos a regra nebulosa representada na figura abaixo.

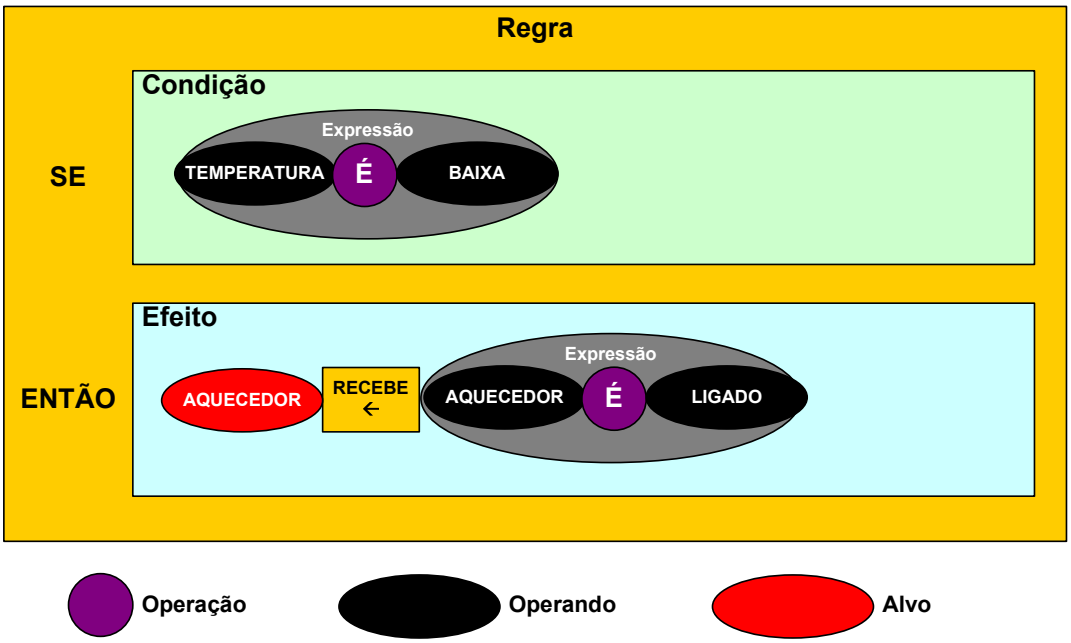


Figura 6.6 - Exemplo de uma regra lógica nebulosa

A regra nebulosa descrita acima faz parte de um banco de dados, a base de regras nebulosa, devendo ser processada por um núcleo de processamento, representado pela máquina de inferência. A entrada nebulosa corresponde à expressão de condição nebulosa TEMPERATURA-É-BAIXA, que avalia se o parâmetro TEMPERATURA se encaixa na classificação BAIXA. A avaliação desta expressão leva a uma saída nebulosa AQUECEDOR-É-LIGADO. Interfaces de nebulização e desnebulização são necessárias para converter o valor cristalino assumido pelo parâmetro TEMPERATURA para a categoria nebulosa BAIXA, e para reconverter a classificação nebulosa LIGADO para um valor cristalino a ser recebido pelo parâmetro AQUECEDOR, respectivamente.

As conversões realizadas por estas interfaces são definidas através de funções de pertencimento nebulosas. A figura a seguir mostra alguns exemplos de funções de pertencimento.

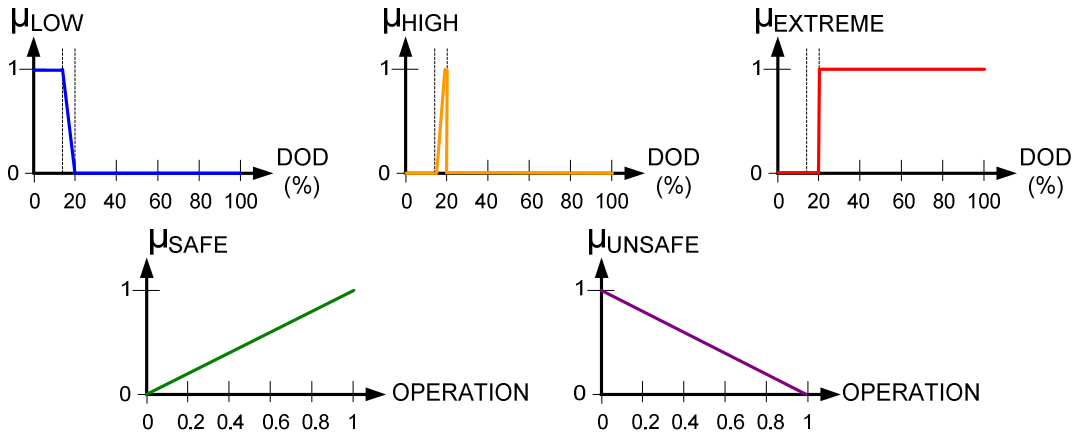


Figura 6.7 - Exemplos de funções de pertencimento
Fonte: Tominaga et al. (2009)

Das cinco funções de pertencimento representadas na figura acima, as três superiores representam interfaces de nebulização associadas a um parâmetro DOD, enquanto que as duas inferiores representam interfaces de desnebulização associadas a um parâmetro OPERATION. Estes parâmetros se relacionam através de regras nebulosas, dentre os quais um exemplo encontra-se representado na figura abaixo.

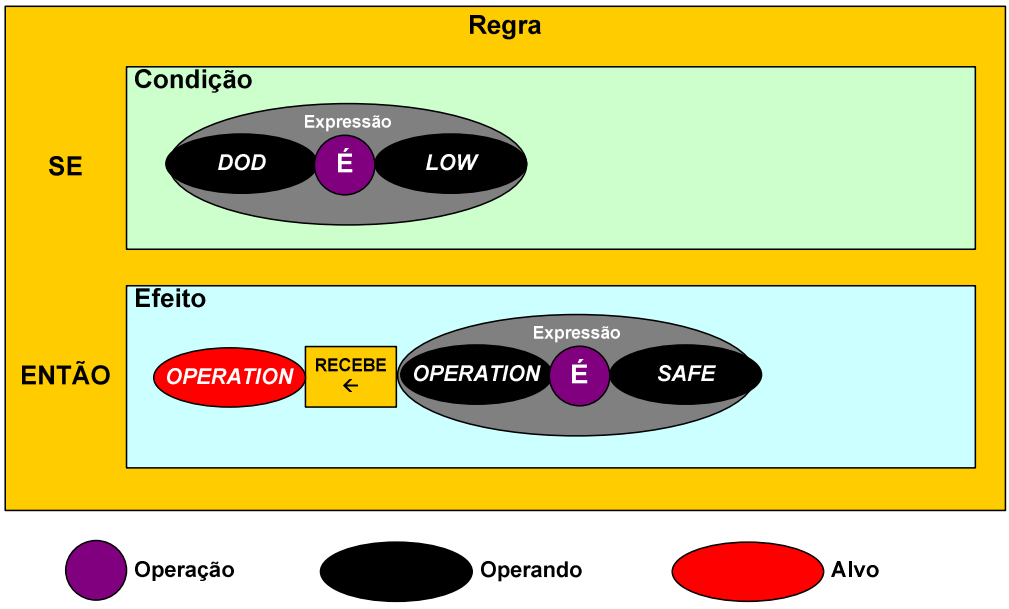


Figura 6.8 - Regra lógica nebulosa para decisão de operação em função do DOD

A regra anterior faz parte de um modelo de simulador de satélite, sendo responsável por decidir pela operação de carga-útil através da avaliação do DOD da bateria se o valor de OPERATION for maior que 0,5. Para ilustrar melhor o funcionamento desta regra considere-se, por exemplo, uma entrada cristalina DOD igual a 18%. Nestas condições, a interface de nebulização fornece uma entrada nebulosa μ_{LOW} igual a 0,4. Nestas condições, a saída nebulosa μ_{SAFE} assume valor 0,4 que leva a um valor cristalino para OPERATION igual a 0,4. Nestas condições, a operação é considerada insegura e não deve ser executada. (TOMINAGA et al., 2009)

6.5. Curvas de funções

Funções de pertencimento nebulosas podem ser expressas através de curvas de funções associadas a um parâmetro de estado. Tais curvas podem ser utilizadas para classificar o estado corrente de um parâmetro de estado, conforme seu valor numérico. Um parâmetro pode ter mais de uma curva de função associada. A figura abaixo mostra um exemplo de parâmetro de estado com três curvas de funções associadas que representam seu estado corrente.

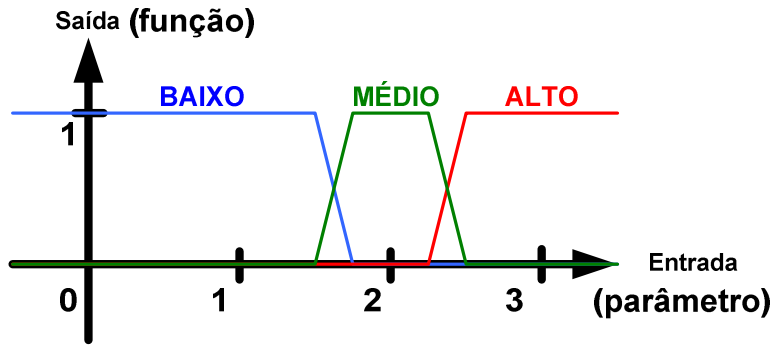


Figura 6.9 - Exemplo de associação de estados qualitativos a um parâmetro

Na figura acima, encontram-se definidas três curvas de funções associadas ao parâmetro de estado ALIMENTAÇÃO. Este parâmetro é classificado como INSUFICIENTE para valores muito abaixo de 2, OK para valores próximos de

2, e EXCESSIVO para valores muito acima de 2. A avaliação da expressão ALIMENTAÇÃO-É-OK retorna valor 1 para valores de ALIMENTAÇÃO próximos de 2. Esta saída diminui gradualmente até atingir valor 0, à medida em que a entrada se afasta do valor 2. Este tipo de expressão é útil para se gerarem regras de controle baseadas em lógica nebulosa. A tabela abaixo mostra a representação tabular das funções BAIXO, MÉDIO e ALTO, explicitando os pontos compostos por pares de valores de entrada e saída.

Tabela 6.2 - Representação tabular de funções de pertencimento

BAIXO		MÉDIO		ALTO	
Entradas	Saídas	Entradas	Saídas	Entradas	Saídas
0.00	1	0.00	0	0.00	0
1.50	1	1.50	0	2.25	0
1.75	0	1.75	1	2.50	1
3.00	0	2.25	1	3.00	1
		2.50	0		
		3.00	0		

Aplicações vislumbradas para o emprego de estruturas de dados do tipo curva de funções incluem, além da representação de funções de pertencimento nebulosas, a decomutação de telemetrias utilizando curvas de calibração. Telemetrias de satélites recebidas em solo, após processamento por equipamentos de rádio-freqüência, resultam em trens de bits. Destes trens de bits são extraídas unidades de dados que representam valores brutos de telemetrias. Estes valores brutos devem ser decomutados, ou seja, convertidos para formatos inteligíveis por operadores e especialistas humanos, a fim de permitir sua análise.

Algumas telemetrias codificam o estado operacional de equipamentos embarcados. Por exemplo, uma telemetria pode indicar por meio de um único bit se um equipamento está ligado ou desligado, ou se ele opera no modo principal ou no redundante. Outras telemetrias representam valores assumidos por algum parâmetro físico que descreve em parte o estado no interior do satélite. Curvas de calibração de telemetrias analógicas são utilizadas para

decomutação de dados brutos para valores processados em unidades de engenharia: graus centígrados, ampères, decibéis, etc. Tipicamente, tais curvas incluem tabelas de conversão de entradas brutas para saídas de sensores em unidades de engenharia, e destas para entradas de sensores unidades de engenharia. Tais curvas são obtidas e documentadas durante montagem, integração e testes, sendo representadas tipicamente na forma de tabelas de valores, com ou sem expressões de funções e gráficos associados.

A tabela abaixo exemplifica duas curvas de calibração associadas a uma mesma telemetria. A curva C-001 é associada a um sensor que fornece dados de temperatura quando um equipamento embarcado opera no principal. Esta mesma telemetria fornece dados baseadas na curva C-002 quando o equipamento é comutado para redundante. Tabelas deste tipo são encontradas freqüentemente em documentos de missão de satélites reais. (INPE, 1988) (XSCC; CAST, 2009) (XSCC; INPE, 2000)

Tabela 6.3 - Exemplo de representação tabular de curvas de calibração

	C-001		C-002	
T (°C)	Rth (Ω)	Vth (V)	Rth (Ω)	Vth (V)
-20.0	30251.3	3.01	32626.0	3.10
-10.0	28076.9	2.92	30291.7	3.01
0.0	23478.3	2.70	25372.1	2.80
10.0	18610.0	2.41	20189.7	2.51
20.0	13333.3	2.00	14602.1	2.11
30.0	9325.5	1.59	10378.5	1.71
40.0	6315.8	1.20	7218.3	1.33
50.0	4630.5	0.94	5453.1	1.07
60.0	3419.2	0.73	4186.1	0.87

Neste exemplo, valores de resistência de termistores (R_{th}) são medidos em laboratório variando-se a temperatura (T). Cada valor de resistência é associado a um valor bruto de tensão analógica, medida em volts (V_{th}). O valor bruto é codificado digitalmente em um byte, na forma de um inteiro sem sinal de 8 bits, associando-se nela quantas de 0.02 V a cada bit. Para uma melhor

ilustração, a tabela a seguir apresenta alguns valores digitais codificados e suas respectivas conversões para tensões analógicas, em volts.

Tabela 6.4 - Amostras de conversão de valor bruto de tensão analógica

Valor codificado (hexadecimal)	Valor codificado (decimal)	Tensão analógica (V)
00H	0	0.00
01H	1	0.02
02H	2	0.04
1FH	31	0.62
FFH	255	5.10

Curvas de calibração de resistência obtidas em laboratório podem ser documentadas em forma tabular, conforme mostrado acima, ou na forma de gráficos e fórmulas, conforme a figura abaixo. Nestes casos, os pontos devem ser medidos a partir das escalas de eixos dos gráficos, ou calculados para um conjunto amostrado de pontos de estrada. Os dados assim coletados são armazenados em planilhas para serem cadastradas no banco de dados. A interface de edição de curvas pode obter suas entradas a partir de tais planilhas.

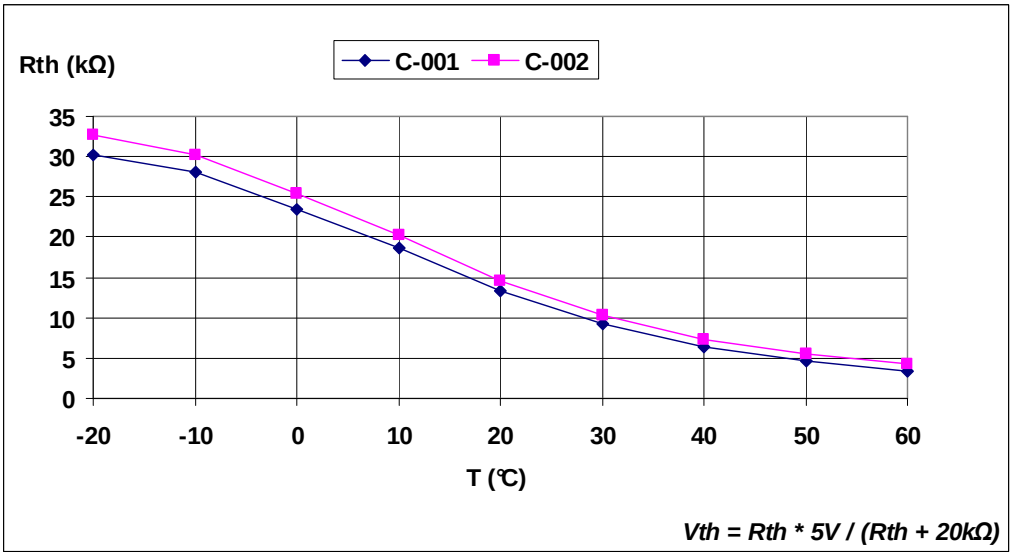


Figura 6.10 - Exemplo de representação gráfica de curvas de calibração

Na figura anterior, observam-se amostras de valores de resistência dos termistores (R_{th}) medidos em função da variação de temperatura (T). Da fórmula expressa no canto inferior direito na figura, infere-se que a tensão de valor bruto da telemetria (V_{th}) é calculada a partir de uma expressão que relaciona R_{th} a uma tensão de referência (V_{ref} = 5 V) e uma resistência de referência. (R_{ref} = 20 kΩ).

Para a simulação de satélites com a finalidade de verificar planos de operações em vôo, apenas o modelamento do parâmetro de temperatura (T) seria suficiente para a codificação das regras, sob condições normais. Porém, a simulação de curvas de calibração torna-se importante quando se há necessidade de recalibração do modelo do simulador de acordo com valores de telemetrias recuperados do histórico de missão. Neste caso, torna-se necessário o modelamento da decomutação de valores de tensão bruta. Dependendo da complexidade do diagnosticador, até mesmo as resistências dos termistores (R_{th}) podem ser necessárias, conforme será visto em um outro Capítulo, mais à frente.

6.6. Operandos e operações

Conforme visto até agora, as regras que controlam o modelo do simulador de satélite são constituídas por uma expressão de condição e um conjunto de efeitos, cada qual contendo um parâmetro de simulação alvo e uma expressão de efeito. Cada expressão é formada a partir de uma seqüência de operandos e operações. Operações representam funções ou operações lógicas e aritméticas. Operandos podem ser parâmetros de simulação ou valores constantes, estados nebulosos, ou expressões aninhadas. O aninhamento de expressões é definido pelo valor da profundidade associada a cada operação. A avaliação da condição é realizada segundo uma lógica, que pode ser booleana ou nebulosa.

Uma lista de operações lógicas, aritméticas e funções elementares para a definição de um núcleo de processamento encontram-se tabeladas abaixo.

Tabela 6.5 - Lista de operações elementares

Operações Elementares					
Lógicas		Aritméticas		Funções	
Símbolo	Descrição	Símbolo	Descrição	Símbolo	Descrição
<	Menor que	+	Soma	ABS	Valor absoluto
≤	Menor ou igual a	-	Subtração	LOG	Logaritmo natural
=	Igual a	*	Multiplicação	EXP	Exponenciação
≥	Maior ou igual a	/	Divisão	SEN	Função seno
>	Maior que	^	Potenciação	COS	Função co-seno
≠	Diferente de	\	Resto de divisão	TAN	Função tangente
&	Operação lógica E			ARCSEN	Função seno inverso
|	Operação lógica OU			ARCCOS	Função co-seno inverso
¬	Operação lógica NÂO			ARCTAN	Função tangente inverso
				É	Operação nebulosa É

Funções mais complexas do que as tabeladas acima podem ser implementadas a partir do uso de regras específicas ou de curvas de funções. Quando utilizadas como curvas de calibração ou outro tipo de função de usuário, a curva de função torna-se uma operação. Neste caso, ela deve ser associada a um operando de entrada.

Outro uso para uma curva de função é permitir o uso de lógica nebulosa por regras de controle do simulador. Quando utilizadas como funções de pertencimento nebulosas, curvas de funções devem ser referenciadas pelas regras através do seu identificador e do parâmetro associado, unidas por uma operação indicativa deste uso, no caso, a operação É.

Parâmetros de simulação incluem parâmetros de estado e parâmetros de evento. Ambos são referenciados por regras de controle, como operandos de expressões através de um identificador. Ou seja, parâmetros de simulação necessitam de um identificador para que possam ser referenciados por regras. Parâmetros de simulação armazenam um valor numérico. A forma como a informação é armazenada, entretanto, muda conforme o tipo do parâmetro.

Parâmetros presentes no estado interno armazenam valores que indicam o estado daquele parâmetro durante um passo de simulação. Já parâmetros presentes na fila de eventos guardam o tempo de disparo de uma instância de eventos. Portanto, parâmetros de estado e de evento devem ser diferenciados através de um campo de tipo.

A forma exata como estruturas de dados representando estes parâmetros de simulação, curvas de função e regras de controle são armazenadas no banco de dados do modelo do simulador será mostrado no próximo Capítulo.

7 ARQUIVOS DE ENTRADA E SAÍDA

A arquitetura do simulador é tal que um núcleo de processamento comum constituído por uma máquina de inferência permite a simulação de diversas missões conforme o modelo a ser alimentado. Este modelo consiste em uma base de regras, que é armazenada na forma de arquivos de banco de dados. Em Capítulos anteriores foram discutidas as estruturas de dados e a arquitetura de interfaces a serem utilizadas pelo simulador. Este Capítulo apresentará uma definição de formatos de arquivos do banco de dados contendo o modelo configurado e do histórico de estados a ser exportado ao fim da sessão de simulação.

7.1. Visão geral

Conforme anteriormente citado, o projeto do simulador é tal que uma mesma máquina de inferência pode utilizar diferentes bases de regras para simular diversos sistemas. Desta forma, um arquivo de configuração se faz necessário para permitir a identificação dos bancos de dados do modelo a serem utilizados na simulação. Este arquivo de configuração deve ser gerado na etapa de configuração e disponibilizado na inicialização.

Os arquivos de banco de dados que constituem o modelo do simulador são acessados para leitura na etapa de inicialização. O modelo do simulador encontra-se distribuído em quatro arquivos codificados em formato *Extensible Markup Language* (XML). Dois arquivos contêm listas de parâmetros, um representando o estado inicial do sistema, e o outro uma fila de eventos. Um terceiro arquivo destina-se ao armazenamento de regras de controle. Um quarto arquivo contém as curvas de funções, que podem ser invocadas por algumas regras. Por serem essencialmente arquivos de texto, o XML possibilita uma codificação flexível de rotinas de interface de entrada e saída na implementação do núcleo de processamento, além de permitir a visualização e configuração de seu conteúdo por meio de editores de texto.

Ao fim da etapa de sessão, o arquivo de saída contendo um histórico de estados internos simulados é exportado em formato Valores Separado por Vírgulas (CSV). Este arquivo segue um padrão semelhante ao formato utilizado pelo software de tempo-real de TT&C em uso atualmente no CRC, para exportação de telemetrias de histórico de missão. Este formato possibilita a manipulação e visualização de dados via Excel© e confere maior facilidade na comparação com telemetrias de missões reais.

Tabela 7.1 - Lista de interfaces do simulador

Descrição	Formato	Etapa	Categoria
Arquivo de configuração	Texto	Configuração	Configuração
Estado inicial	XML	Inicialização	Modelo
Fila de eventos	XML	Inicialização	Modelo
Regras de controle	XML	Inicialização	Modelo
Curvas de funções	XML	Inicialização	Modelo
Histórico de estados	CSV	Sessão	Saída

7.2. Formato XML

O XML é uma linguagem especificada e regulada pelo Consórcio *World Wide Web* (W3C). A especificação completa da linguagem, segundo disponibilizada *online* na página desta organização (www.w3.org/xml), é demasiadamente extensa e foge do escopo deste trabalho. Portanto, os bancos de dados utilizados pelo simulador de satélite utilizam na verdade um formato compatível com um subconjunto da linguagem XML. Porém, por questão de simplicidade, o formato adotado pelos arquivos de bancos de dados do simulador será referenciado ao longo deste trabalho como formato XML.

Os dados que compõem o modelo do simulador são armazenados na forma de estruturas de conteúdo (*contents*) escritos em arquivos de texto. Estes conteúdos são formados a partir de elementos (*elements*) estruturados e delimitados através de etiquetas (*tags*). Etiquetas são compostas por textos marcadores (*markups*) cercados por caracteres de delimitação, e servem como

chaves do banco de dados. Um elemento é delimitado por meio de duas etiquetas, uma de abertura, e outra de fechamento. Uma etiqueta de abertura é composta por um marcador antecedido por um caractere menor que (<), e sucedido por um caractere maior que (>). Uma etiqueta de fechamento é composta pelo mesmo marcador, antecedido por uma seqüência de dois caracteres formada por um menor que e uma barra invertida (<\), e é sucedido por um caractere maior que (>). Caracteres de espaço, tabulação, quebra de linha e retorno de alimentação são ignorados pelo analisador sintático (*parser*), o que permite certa liberdade na codificação de arquivo, visando melhorar sua legibilidade por um ser humano. A figura abaixo ilustra um exemplo de arquivo no formato XML.

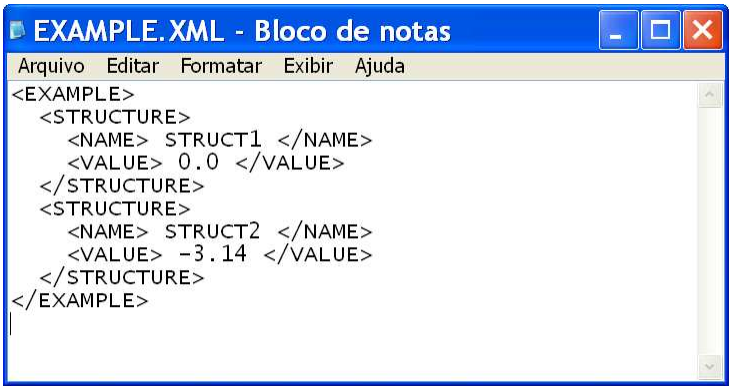


Figura 7.1 - Exemplo de arquivo em formato XML

No exemplo acima, temos um conteúdo etiquetado pela marcação EXAMPLE, composto por dois conteúdos-filhos. Cada um destes conteúdos-filhos, identificados pela marcação STRUCTURE, contém respectivamente mais dois conteúdos-filhos, indicados pelas etiquetas de abertura <NAME> e <VALUE>. Cada conteúdo NAME tem associado um elemento (STRUCT1 e STRUCT2), assim como o conteúdo VALUE (0.0 e -3.14). Este exemplo (EXEMPLE) representa o armazenamento de duas estruturas (STRUCTURE) do mesmo tipo, compostas por dois campos, nome (NAME) e valor (VALUE). A primeira estrutura de nome STRUCT1 tem valor 0.0, e o valor atribuído à segunda estrutura, de nome STRUCT2, é -3.14.

7.3. Formato CSV

O formato CSV adotado neste trabalho consiste num arquivo de texto para representação de tabelas, onde linhas são separadas por caracteres de quebra de linha e retorno de alimentação, e colunas delimitadas por caracteres vírgula (,). Valores numéricos reais são expressos através do uso de ponto decimal, a fim de reservar o uso de vírgulas para a separação de colunas. A figura abaixo exemplifica um arquivo em formato CSV.

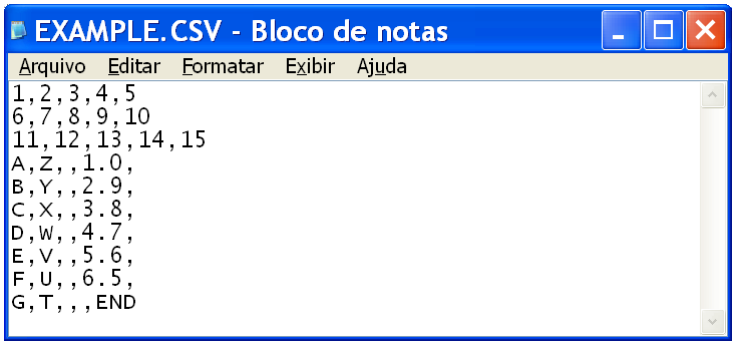


Figura 7.2 - Exemplo de arquivo em formato CSV

A figura acima representa uma tabela de dez linhas e cinco colunas. Vírgulas consecutivas ou seguidas que quebra de linha indicam células em branco. A representação deste exemplo em forma de planilha é ilustrada na figura abaixo.

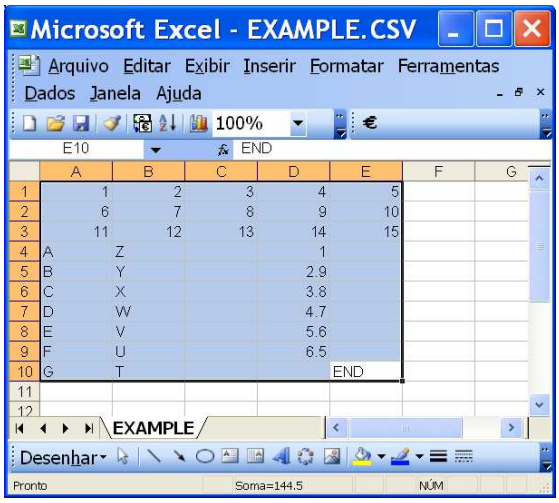


Figura 7.3 - Representação em forma de planilha do arquivo CSV exemplificado

Diferentemente do XML, o CSV não possui um órgão regulador. Um formato alternativo de CSV, utilizado por default pelo Excel© em sistemas operacionais configurado regionalmente para português brasileiro, utiliza caracteres ponto-e-vírgula (;) para separação de colunas. Esta adaptação permite o uso de vírgulas em valores numéricos reais para a separação unidades fracionárias. A figura abaixo ilustra como ficaria o exemplo acima na formatação alternativa.

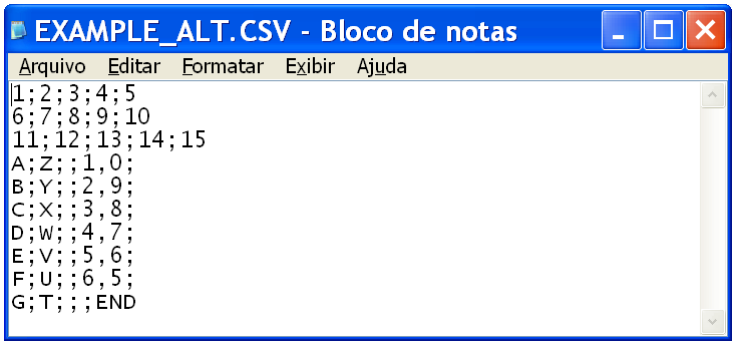


Figura 7.4 - Arquivo CSV exemplificado em formato alternativo

Entretanto, esta abordagem foi deixada de lado por questões de compatibilidade com o sistema legado, que utiliza o formato padrão.

7.4. Arquivo de configuração

No início da inicialização, um arquivo de configuração é lido pelo simulador. Este arquivo de configuração informa os arquivos de bancos de dados a serem utilizados na simulação.

São quatro arquivos de entrada a serem carregados na inicialização e um arquivo de saída para exportação de dados. Os nomes destes arquivos devem ser explicitados na configuração. O formato escolhido para a implementação do arquivo de configuração é um arquivo texto contendo cinco linhas. Cada linha deve conter um caminho para cada arquivo de dados a ser utilizado pela simulação. O conteúdo de cada linha é explicitada na tabela a seguir.

Tabela 7.2 - Conteúdo do arquivo de configuração

Linha	Mnemônico	Descrição
1	InitStat	Estado inicial
2	EvQueue	Fila de eventos
3	CtrlRules	Regras de controle
4	StatHis	Histórico de estados
5	FuncCurv	Curvas de funções

Após a leitura das cinco linhas iniciais, linhas excedentes devem ser ignoradas pelo núcleo de processamento, portanto podem conter informações adicionais na forma de comentários. A figura abaixo ilustra um exemplo de arquivo de configuração.

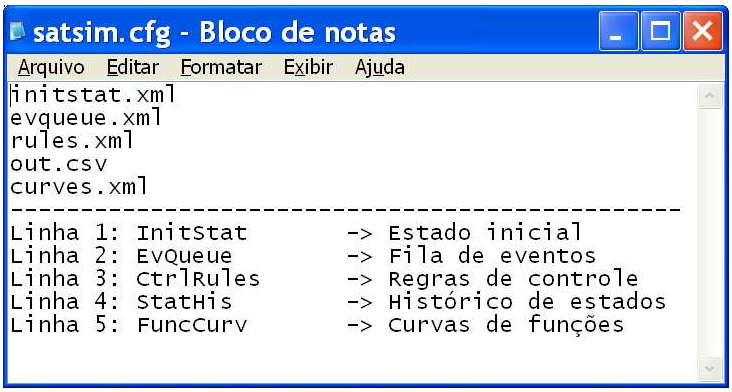


Figura 7.5 - Exemplo de arquivo de configuração

7.5. Estado inicial

O estado inicial do sistema simulado corresponde a um conjunto de parâmetros de simulação que definem o estado interno do sistema no passo de simulação inicial da sessão de simulação. Além de estabelecer valores que caracterizam o sistema no início da sessão, o estado inicial é responsável também por definir o conjunto de parâmetros existentes no modelo. Desta forma, todos os parâmetros que fazem parte do modelo devem ser declarados neste arquivo, uma única vez.

O estado interno é definido através de valores (ParValue) assumidos por cada parâmetro. Parâmetros de simulação são referenciados por regras através de um identificador (ParId). Parâmetros de estado e parâmetros de evento devem ser diferenciados através de um indicador de tipo (ParType). A tabela a seguir mostra a estrutura de campos adotada para representar um parâmetro de simulação.

Tabela 7.3 - Estrutura de um parâmetro de simulação

Mnemônico	Chave	Descrição
ParId	<id>	Identificador do parâmetro
ParType	<typ>	Tipo do parâmetro
ParValue	<val>	Valor numérico

A segunda coluna na tabela acima (<id>, <typ> e <val>) indica etiquetas de abertura XML utilizadas como chaves de identificação de campos. O estado inicial é constituído por uma lista de estruturas de parâmetros identificada pela chave <stat>. Cada parâmetro é indicado pela chave <par>. A figura abaixo ilustra um exemplo de arquivo XML contendo o estado inicial.

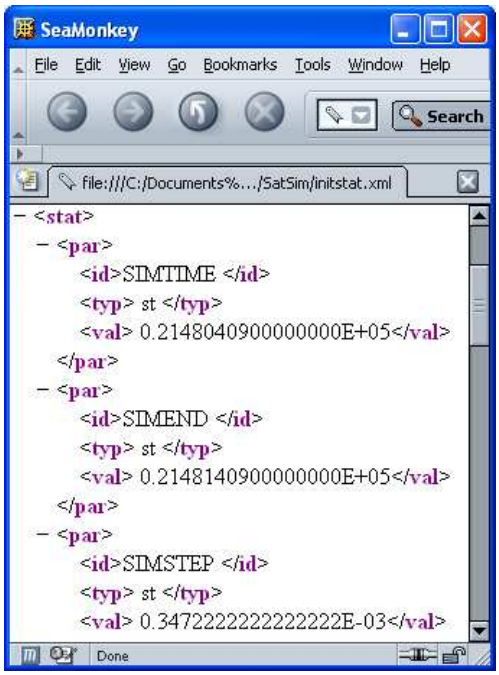


Figura 7.6 - Exemplo de arquivo de estado inicial

Na figura anterior são mostrados três parâmetros localizados no início do arquivo. Os parâmetros são identificados pelos nomes SIMTIME, SIMEND e SIMSTEP. Os três são parâmetros do tipo estado (ParType = st). Os valores de SIMTIME e SIMEND correspondem a duas datas expressas como o tempo transcorrido desde zero hora do dia primeiro de janeiro do ano de 1950, medido em dias. Esta representação de tempo é denominada Data Juliana Modificada (DJM), sendo empregada pela dinâmica de vôo do CRC para cálculos de determinação e previsão de órbitas de satélites. O valor de SIMSTEP corresponde a trinta segundos, expressa como fração de dia, de forma compatível com o sistema DJM. Estes parâmetros são parâmetros importantes de controle da simulação. O valor de SIMTIME define o tempo inicial, SIMEND o tempo final, e SIMSTEP, o tempo incremental de passo de simulação.

7.6. Fila de eventos

A fila de eventos é uma compilação de eventos orbitais previstos e atividades contidas no plano de operações. Ela é composta por uma lista temporizada de instâncias de parâmetros de simulação, do tipo evento.

Um arquivo de fila de eventos é constituído por uma lista de parâmetros identificada pela chave <queue>. Cada parâmetro é representado da mesma forma que é feita no arquivo e estado inicial, o que torna estes dois arquivos muito semelhantes. Entretanto, enquanto no estado inicial parâmetros são declarados uma única vez, na fila de eventos os parâmetros podem ser instanciados mais de uma vez. Além disto, todos os parâmetros contidos numa fila de eventos são todos do tipo evento (ParType = ev).

As semelhanças visuais e estruturais entre um estado inicial e uma fila de eventos, bem como as diferenças, são nitidamente perceptíveis ao se compararem a figura anterior com a que será mostrada a seguir.

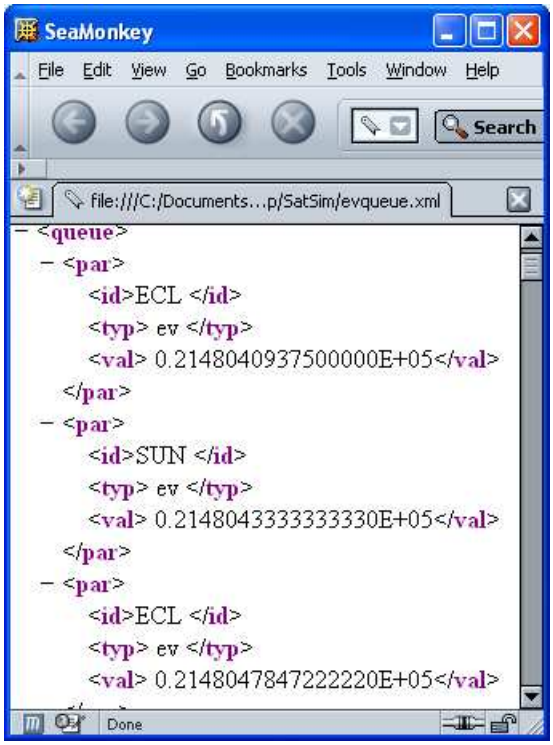


Figura 7.7 - Exemplo de arquivo de fila de eventos

Na figura acima, são mostrados três parâmetros localizados no início do arquivo. Os parâmetros são identificados pelos nomes ECL e SUN. O parâmetro ECL é instanciado na primeira e terceira posições. Embora não seja mostrado na figura, ocorrem mais instâncias destes mesmos parâmetros, em tempos diferentes, ao longo deste arquivo. Os valores dos parâmetros correspondem a datas expressas em DJM que correspondem aos instantes de início de cada evento.

7.7. Regras de controle

As regras de controle explicitam como parâmetros de simulação contidos no estado interno devem ser alterados em cada passo de simulação, ao longo da sessão de simulação.

Cada regra possui uma condição (RuleCond) que, caso seja satisfeita, dispara um ou mais efeitos (RuleEffect) associados. Uma condição consiste em uma

única expressão lógica (CondExpr). Um efeito é composto por um parâmetro alvo (EffectTarget) e uma expressão numérica (EffectExpr). Apesar de não ser necessária para o processamento da simulação, cada regra possui também um identificador (RuleId) utilizada na etapa de configuração.

A tabela abaixo mostra a estrutura de uma regra descrita acima.

Tabela 7.4 - Estrutura de uma regra de controle

Mnemônico	Chave	Descrição
RuleId	<id>	Identificador da regra
RuleCond	<cond>	Condição da regra
CondExpr	...	Expressão de condição da regra
RuleEffect	<effect>	Efeito da regra
EffectTarget	<target>	Parâmetro alvo do efeito da regra
EffectExpr	...	Expressão de efeito da regra

De acordo com a segunda coluna na tabela acima, identificadores de regras (RuleId), condições (RuleCond), efeitos (RuleEffect) e parâmetros alvos (EffectTarget) têm chaves XML bem definidos (<id>, <cond>, <effect> e <target>, respectivamente). Entretanto, expressões de condição (CondExpr) e de efeito (EffectExpr) possuem chaves variáveis, devido à forma pelo qual expressões são definidas.

Expressões são compostas por operandos e operações, conforme descrito no Capítulo anterior. Operandos podem ser constituídos por parâmetros (ExprPar), valores (ExprVal), curvas (ExprCurve) ou sub-expressões (ExprNest). Operações podem ser elementares (ExprOper) ou funções definidas pelo usuário (ExprFunc). A tabela a seguir mostra uma lista de componentes de uma expressão.

Tabela 7.5 - Componentes de uma expressão de regra

Mnemônico	Chave	Tipo	Descrição
ExprPar	<par>	Operando	Parâmetro
ExprVal	<val>	Operando	Valor numérico
ExprCurve	<curv>	Operando	Curva associada a parâmetro
ExprOper	<oper>	Operação	Operação elementar
ExprFunc	<func>	Operação	Função definida por curva
ExprNest	<expr>	Sub-expressão	Aninhamento de expressão

A presença de caracteres reservados pelo padrão XML impede a utilização de alguns símbolos para representar operações. Alternativas existem no XML padrão, como por exemplo, o uso de códigos de escape (*escape codes*) e referências a entidades (*entity references*). Entretanto, a complexidade introduzida na análise sintática levou ao descarte de sua adoção na definição do formato para a representação de regras no modelo. (HAROLD; MEANS, 2002)

Desta forma, operações elementares são representadas por códigos compostos por até três caracteres, conforme tabelados a seguir.

Tabela 7.6 - Lista de operações elementares

Operações Elementares					
Lógicas		**Aritméticas**		**Funções**	
Código	**Descrição**	**Código**	**Descrição**	**Código**	**Descrição**
LT	**Menor que**	**+**	**Soma**	**ABS**	**Valor absoluto**
LE	**Menor ou igual a**	**-**	**Subtração**	**LOG**	**Logaritmo natural**
EQ	**Igual a**	*****	**Multiplicação**	**EXP**	**Exponenciação**
GE	**Maior ou igual a**	**/**	**Divisão**	**SIN**	**Função seno**
GT	**Maior que**	**^**	**Potenciação**	**COS**	**Função co-seno**
NE	**Diferente de**	**MOD**	**Resto de divisão**	**TAN**	**Função tangente**
AND	**Operação lógica E**			**ASI**	**Função seno inverso**
OR	**Operação lógica OU**			**ACO**	**Função co-seno inverso**
NOT	**Operação lógica NÂO**			**ATA**	**Função tangente inverso**
				IS	**Operação nebulosa É**

Um arquivo de regras de controle é constituído por uma lista de parâmetros identificada pela chave <ctrl>. Cada regra é indicada pela chave <rule>. A

representação de um exemplo de regra em formato XML é mostrada na figura abaixo.

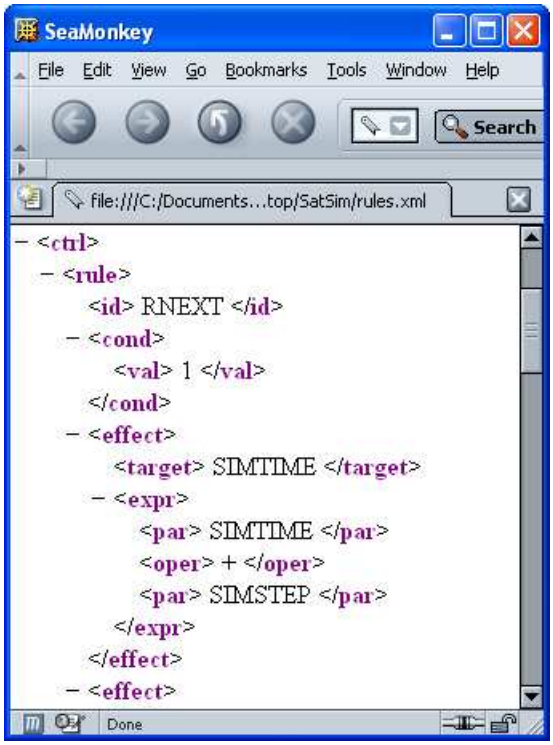


Figura 7.8 - Exemplo de arquivo de regras de controle

Na figura acima, a regra RNEXT é condicionada a uma expressão representada por um valor verdadeiro (CondExpr = ExprVal = 1). Em outrass palavras, a condição da regra é tal que ela sempre é satisfeita. Este tipo de condicionamento é útil para representar cálculos que devem ser executados em todos os passos de simulação da sessão. A regra possui um único efeito, onde se atribui a um parâmetro alvo SIMTIME o resultado de uma expressão (SIMTIME + SIMSTEP). Esta regra é responsável pelo controle do passo de simulação. A cada passo, o valor do tempo de simulação SIMTIME é incrementado do tempo de passo SIMSTEP.

7.8. Curvas de funções

Curvas de funções armazenam estruturas de dados destinadas ao mapeamento de valores, que retornam um valor de saída mapeado conforme o valor da entrada. O arquivo de curvas de funções tem como finalidade informar ao núcleo de processamento as curvas definidas no modelo. Portanto, cada curva deve ser declarada uma única vez.

Cada curva de função é constituída por um conjunto de pontos (CurvePoints), que associam um valor de saída (PointInput) a um valor de entrada (PointOutput). Valores não especificados são obtidos via interpolação ou extrapolação linear. Elas são referenciadas em regras por meio de um identificador (CurveId) e podem ser associadas a um parâmetro (CurvePar). A tabela abaixo apresenta a estrutura de uma curva de função.

Tabela 7.7 - Estrutura de uma curva de função

Mnemônico	Chave	Descrição
CurveId	<id>	Identificador da curva
CurvePar	<par>	Parâmetro associado à curva
CurvePoints	<pt>	Pontos da curva
PointInput	<in>	Entrada de um ponto
PointOutput	<out>	Saída de um ponto

A identificação de uma curva é feita a partir da combinação destes dois campos (CurveId e CurvePar). Desta forma, é possível que mesmo arquivo possa conter mais de uma curva com o mesmo nome (CurveId), porém associado a parâmetros (CurvePar) diferentes. Isto permite que parâmetros distintos possuam funções de pertencimento nebulosas com o mesmo nome. Uma curva que representasse uma função de usuário, sem parâmetro associado, poderia ser representada por meio de um valor vazio no campo correspondente (CurvePar).

Cada conjunto de pontos de uma curva (CurvePoints) deve conter uma seqüência de pares de entrada e saída (PointInput e PointOutput) maior ou igual a dois. Dois pontos definem uma reta, e três ou mais pontos definem uma curva linear por partes. A representação de uma curva de funções no formato de arquivo XML é mostrada na figura abaixo.

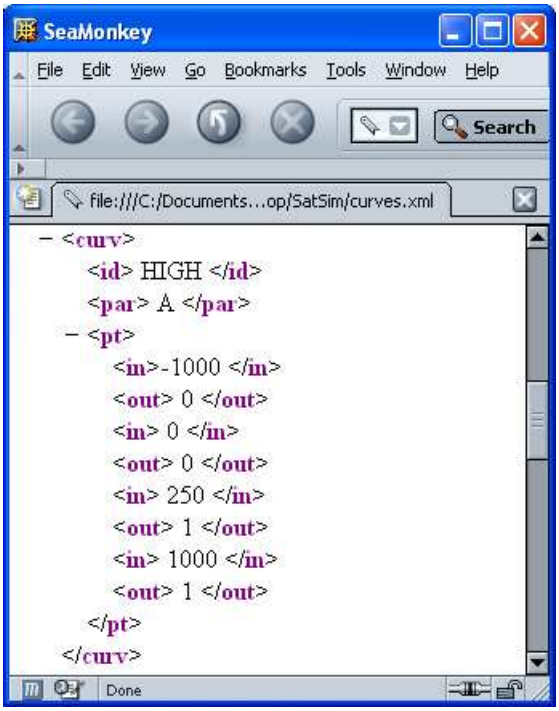


Figura 7.9 - Exemplo de arquivo de curvas de funções

Na figura acima, uma curva HIGH associada a um parâmetro A é definida através de quatro pontos. Os dois primeiros pontos definem um segmento de reta, segundo o qual para entradas maiores ou iguais a 0, o valor de saída torna-se 0. O segundo e terceiro pontos definem uma rampa de subida linear que associa a uma entrada entre 0 e 250 uma saída correspondente a 1/250 deste valor. Os dois últimos pontos definem que para entradas maiores que 250, a saída é constante igual a 1. A curva exemplificada na figura define uma função de pertencimento nebulosa HIGH para um parâmetro A. O mesmo arquivo de curvas pode conter, por exemplo, outra curva HIGH associada a outro parâmetro B.

7.9. Histórico de estados

O arquivo de saída contendo o histórico de estado é utilizado para geração de diagnóstico e aceitação de planos de operações. Este arquivo não faz parte do banco de dados, mas seu conteúdo pode ser incorporado ao modelo durante a configuração, conforme será visto em outro Capítulo.

Os parâmetros do histórico de estados são exportados em forma de tabela, com colunas representando a evolução temporal de valores de um parâmetro de estado, e linhas representando o estado interno num determinado passo de simulação. O formato deste arquivo é CSV.

Os parâmetros componentes do estado interno são dispostos em colunas no histórico de estados. Cada coluna corresponde a um parâmetro de simulação. As duas primeiras linhas do arquivo contêm um cabeçalho, que identifica (ParId) e tipifica (ParType) cada parâmetro. As linhas seguintes representam valores assumidos pelos parâmetros durante a sessão de simulação. Cada linha corresponde ao estado interno em cada passo de simulação da sessão. Um exemplo de arquivo de saída é mostado a seguir, para efeito de ilustração.

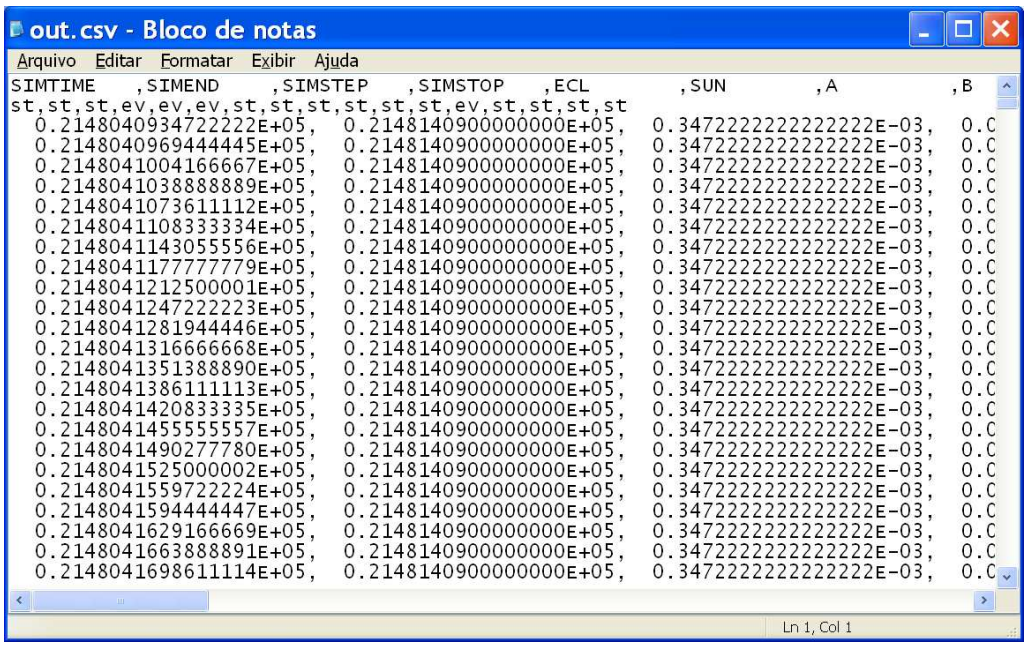
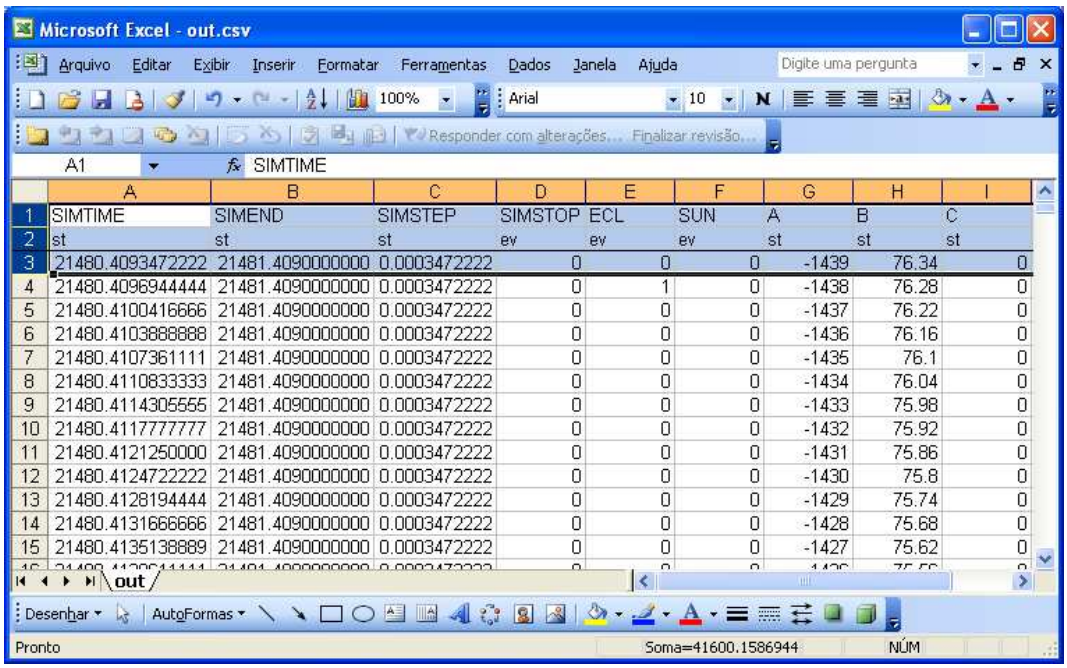


Figura 7.10 - Exemplo de arquivo de saída

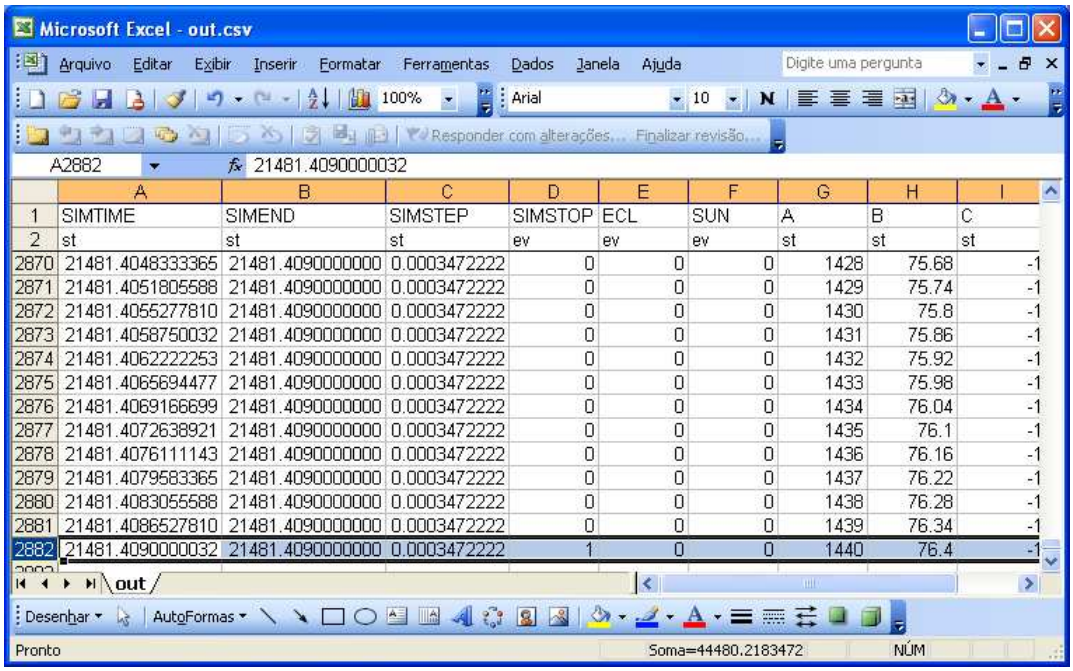
Na figura anterior, identificam-se dezessete parâmetros. Podem-se reconhecer oito deles por meio de seus identificadores (ParId), na primeira linha. A quantidade total de parâmetros no estado interno pode ser inferida através da segunda linha, que lista treze parâmetros do tipo estado (ParTyp = st) e quatro do tipo evento (ParTyp = ev). Conforme a posição em que são representadas, observa-se que SIMTIME, SIMEND, SIMSTEP, A e B são parâmetros de estado, enquanto SIMSTOP, ECL e SUN são parâmetros de evento. As linhas seguintes mostram o início de um histórico de valores em três colunas e apenas parte do valor na quarta coluna. Conforme a figura, o valor de SIMTIME incrementa a cada passo de simulação, enquanto SIMEND e SIMSTEP mantêm-se constantes. Uma análise mais cuidadosa revela que este o valor de SIMTIME incrementa a cada passo de simulação, de um valor constante equivalente ao valor de SIMSTEP. Este comportamento é ditado pela regra RNEXT, descrito anteriormente.

A figura a seguir mostra o mesmo arquivo aberto em forma de planilha. Através dela, é possível perceber que os eventos SIMSTOP e SUN se mantêm inativos nos primeiros quinze passos de simulação da sessão, enquanto o evento ECL é disparado no segundo passo. Nestes mesmos passos, os parâmetros de estado A e B têm seus valores modificados, enquanto C mantém-se igual a zero.



	A	B	C	D	E	F	G	H	I
1	SIMTIME	SIMEND	SIMSTEP	SIMSTOP	ECL	SUN	A	B	C
2	st	st	st	ev	ev	ev	st	st	st
3	21480.4093472222	21481.4090000000	0.0003472222	0	0	0	-1439	76.34	0
4	21480.4096944444	21481.4090000000	0.0003472222	0	1	0	-1438	76.28	0
5	21480.4100416666	21481.4090000000	0.0003472222	0	0	0	-1437	76.22	0
6	21480.4103888888	21481.4090000000	0.0003472222	0	0	0	-1436	76.16	0
7	21480.4107361111	21481.4090000000	0.0003472222	0	0	0	-1435	76.1	0
8	21480.4110833333	21481.4090000000	0.0003472222	0	0	0	-1434	76.04	0
9	21480.4114305555	21481.4090000000	0.0003472222	0	0	0	-1433	75.98	0
10	21480.4117777777	21481.4090000000	0.0003472222	0	0	0	-1432	75.92	0
11	21480.4121250000	21481.4090000000	0.0003472222	0	0	0	-1431	75.86	0
12	21480.4124722222	21481.4090000000	0.0003472222	0	0	0	-1430	75.8	0
13	21480.4128194444	21481.4090000000	0.0003472222	0	0	0	-1429	75.74	0
14	21480.4131666666	21481.4090000000	0.0003472222	0	0	0	-1428	75.68	0
15	21480.4135138889	21481.4090000000	0.0003472222	0	0	0	-1427	75.62	0

Abaixo é mostrada a porção final do mesmo arquivo, aberto em forma de planilha.



	A	B	C	D	E	F	G	H	I
1	SIMTIME	SIMEND	SIMSTEP	SIMSTOP	ECL	SUN	A	B	C
2	st	st	st	ev	ev	ev	st	st	st
2870	21481.4048333365	21481.4090000000	0.0003472222	0	0	0	1428	75.68	-1
2871	21481.4051805588	21481.4090000000	0.0003472222	0	0	0	1429	75.74	-1
2872	21481.4055277810	21481.4090000000	0.0003472222	0	0	0	1430	75.8	-1
2873	21481.4058750032	21481.4090000000	0.0003472222	0	0	0	1431	75.86	-1
2874	21481.4062222253	21481.4090000000	0.0003472222	0	0	0	1432	75.92	-1
2875	21481.4065694477	21481.4090000000	0.0003472222	0	0	0	1433	75.98	-1
2876	21481.4069166699	21481.4090000000	0.0003472222	0	0	0	1434	76.04	-1
2877	21481.4072638921	21481.4090000000	0.0003472222	0	0	0	1435	76.1	-1
2878	21481.4076111143	21481.4090000000	0.0003472222	0	0	0	1436	76.16	-1
2879	21481.4079583365	21481.4090000000	0.0003472222	0	0	0	1437	76.22	-1
2880	21481.4083055588	21481.4090000000	0.0003472222	0	0	0	1438	76.28	-1
2881	21481.4086527810	21481.4090000000	0.0003472222	0	0	0	1439	76.34	-1
2882	21481.4090000032	21481.4090000000	0.0003472222	1	0	0	1440	76.4	-1

A figura anterior revela que a sessão de simulação foi encerrada assim que foi disparado o evento SIMSTOP. Neste último passo, percebe-se também que os valores de SIMTIME e SIMEND coincidem a menos de um pequeno erro. Este comportamento é definido através de um mecanismo de controle da simulação incorporada ao núcleo de processamento que utiliza estes parâmetros, conforme será descrito no próximo Capítulo.

8 IMPLEMENTAÇÃO DO SIMULADOR

Neste Capítulo será tratada a implementação do núcleo de processamento do simulador de satélite, capaz de processar modelos em bancos de dados e gerar uma saída, conforme especificados e descritos nos Capítulos anteriores.

8.1. Visão geral

Conforme visto anteriormente, o simulador funciona em três etapas executadas em seqüência. A figura abaixo mostra estas etapas, atualizada de acordo com novas informações identificadas e discutidas nos últimos Capítulos.

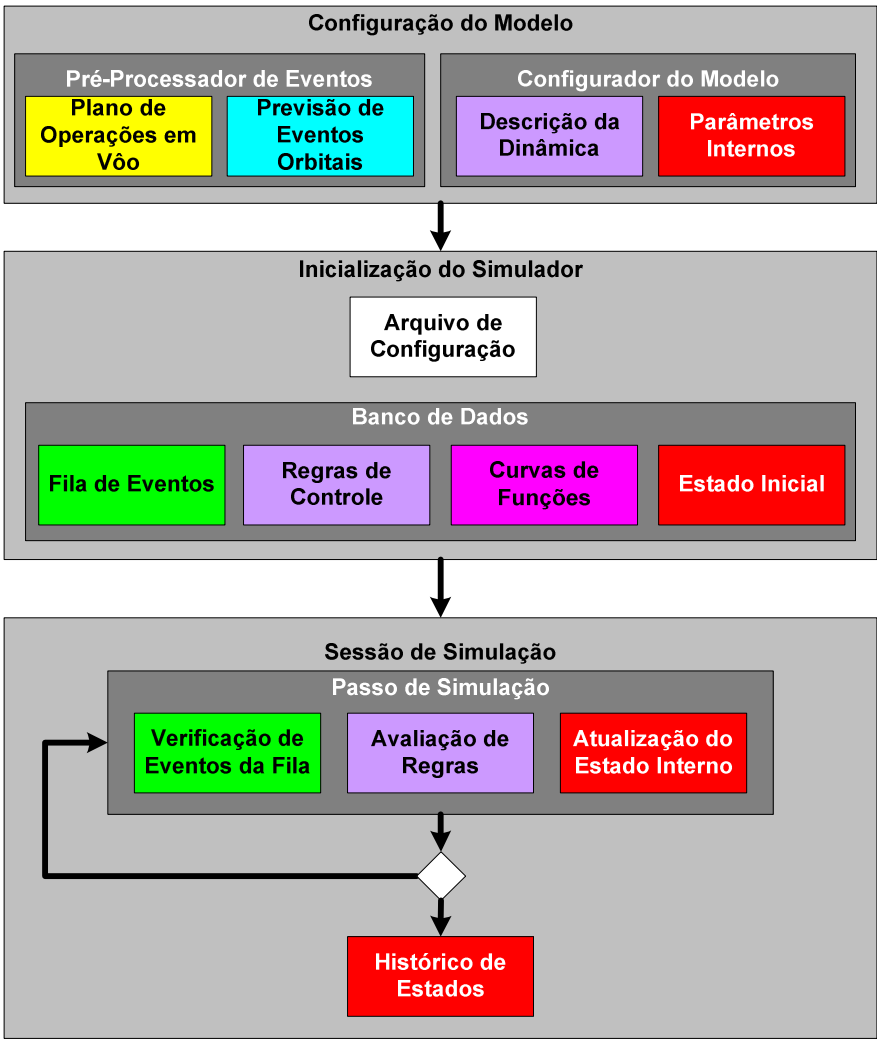


Figura 8.1 - Funcionamento do simulador, atualizado

A primeira etapa, de configuração do modelo, é realizada por meio de ferramentas auxiliares, mais especificamente através de um configurador de modelos e de um pré-processador de eventos. Cabe ao simulador propriamente dito executar as etapas restantes, desde a sua inicialização até a sessão de simulação.

Assim, uma vez configurado, a execução do simulador divide-se em duas etapas. Na etapa de inicialização do simulador, estado inicial, fila de eventos, regras de controle e curvas de funções são carregados do banco de dados. Em seguida, o simulador entra em modo de sessão, durante o qual o estado interno é atualizado a cada passo e, ao fim, exportado na forma de arquivo de histórico. A figura a seguir mostra o diagrama de atividades do simulador, onde as atividades de cada etapa são apresentadas segundo a ordem de execução.

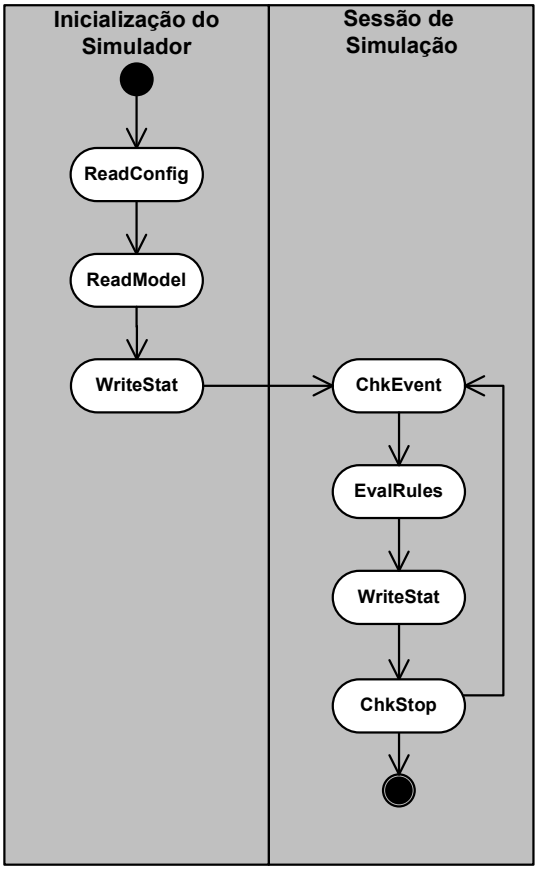


Figura 8.2 - Diagrama de atividades do simulador

A primeira atividade executada pelo simulador durante a sua inicialização consiste na leitura de um arquivo de configuração (ReadConfig). Ao ser alimentado com os modelos devidamente configurados, um mesmo núcleo de processamento pode ser utilizado para simular diferentes missões. Assim, o simulador deve primeiramente identificar qual modelo carregar a fim de começar sua inicialização. Isto é feito a partir da leitura de um arquivo de configuração, cujo conteúdo foi descrito no Capítulo anterior.

A segunda atividade na inicialização consiste na leitura de arquivos de bancos de dados que compõem o modelo (ReadModel). São lidos os arquivos contendo o estado inicial, a fila de eventos, as regras de controle e, por fim, as curvas de funções. O simulador carrega as estruturas de dados destes arquivos de bancos de dados uma única vez. Elas serão mantidas em memória e atualizadas durante a execução da sessão de simulação.

A terceira atividade, executada no fim da etapa de inicialização, consiste na escrita de parâmetros do estado inicial, que consiste no estado interno antes da execução do primeiro passo, num arquivo de saída (WriteStat). Ela objetiva a inicialização do arquivo de saída, ao qual serão anexados estados internos atualizados a cada passo de simulação da sessão a ser executada, compondo a saída do simulador. Nesta atividade são escritos no arquivo de saída os campos de identificação (ParId) e tipo (ParType) dos parâmetros, o que compõe as duas primeiras linhas, de cabeçalho, do arquivo do histórico de estado.

Terminada a etapa de inicialização, o simulador entra em modo de sessão de simulação. Neste modo realizam-se ciclicamente passos de simulação. Em cada passo, executam-se seqüencialmente as atividades de verificação de evento (ChkEvent), avaliação de regras (EvalRules), escrita de estado corrente (WriteStat) e verificação de parada da sessão. A cada passo de simulação avaliam-se inicialmente os disparos de eventos. Com base nos eventos

disparados e valores de parâmetros do estado interno corrente, avaliam-se condições de execução das regras de controle. Caso alguma regra seja disparada, aplicam-se seus efeitos sobre os parâmetros alvos que compõem o estado interno corrente. Após a avaliação de todas as regras, o estado interno atualizado é anexado ao arquivo de saída, que constitui o histórico de sessão da simulação, para atualizá-lo. Ao fim de cada passo, é verificada a condição de parada da sessão de simulação.

Durante a atividade de verificação de eventos (ChkEvent), o tempo de simulação é comparado com os tempos de execução constantes na fila de eventos. Esta fila contém o agendamento de todos os eventos pré-programados que podem ser disparados durante a sessão de simulação. Como o objetivo consiste na simulação de execução de planos de operações, não há eventos externos a serem incluídos em tempo real. Todos os procedimentos a serem executados são obtidos por meio do plano de operações. Eventos externos são obtidos a partir da previsão de eventos orbitais, o que permite a criação de um arquivo de fila de eventos que consiste numa lista temporizada destas ocorrências. A associação entre eventos e seus efeitos é realizada por meio de regras de controle.

Através da atividade de avaliação de regras (EvalRules) ocorre a atualização de parâmetros que compõem o estado interno, em cada passo de simulação. Cada regra é composta por uma condição que, caso seja satisfeita, leva à aplicação de um ou mais efeitos. Cada efeito é constituído por um parâmetro alvo e uma expressão. Dependendo do resultado da avaliação da expressão de condição, efeitos podem ser aplicados ou não. A aplicação completa de um efeito consiste na substituição do valor do parâmetro alvo pelo resultado da avaliação da expressão de efeito. Isto ocorre quando o resultado da avaliação da expressão de condição resulta no valor numérico 1, o que corresponde ao valor lógico VERDADEIRO. A não-aplicação de um efeito consiste na manutenção do valor corrente do parâmetro alvo, independentemente do

resultado da avaliação da expressão de efeito. Isto ocorre quando o resultado da avaliação da expressão de condição resulta no valor numérico 0, correspondente ao valor lógico FALSO. Aplicações parciais de efeitos são determinados por valores intermediários entre 0 e 1 resultantes da avaliação da expressão de condição. Neste caso, a aplicação de efeito é ponderada entre manutenção do valor corrente e substituição total pelo resultado da expressão de efeito, conforme o valor resultante da avaliação da condição.

Na atividade de escrita ao fim de cada passo (WriteStat), os valores de parâmetros que compõem o estado interno atualizado é exportado para o histórico de estados. Em cada passo, o estado interno é atualizado depois de avaliadas todas as regras. Os valores de seus parâmetros são anexados na forma de linha ao fim do arquivo CSV a cada execução desta atividade.

Ao fim de cada passo de simulação, a condição de parada da sessão é verificada (ChkStop). Dependendo desta avaliação, ou a sessão é continuada através da avaliação do próximo passo de simulação, ou o simulador é finalizado. Em caso de continuação, executam-se novamente a verificação de evento (ChkEvent), a avaliação de regras (EvalRules), a escrita de estado interno (WriteStat) e uma nova verificação da condição de parada. Em caso de finalização, o arquivo de histórico de estados é fechado e liberado para análise pelo gerador de diagnóstico.

As atividades apresentadas acima se encontram sumarizadas na tabela abaixo. Observa-se que uma atividade (WriteStat) é executada nas etapas de inicialização e sessão. A sua implementação será descrita a seguir.

Tabela 8.1 - Resumo de atividades do simulador

Etapa	Mnemônico	Atividade
Inicialização	ReadConfig	Leitura de arquivo de configuração
	ReadModel	Leitura de parâmetros, curvas e regras do modelo
	WriteStat	Escrita de estado para inicialização de histórico
Sessão		Escrita de estado para atualização de histórico
	ChkEvent	Verificação de disparo de eventos
	EvalRules	Avaliação de regras de controle
	ChkStop	Verificação de condição de parada de simulação

8.2. Implementação

A máquina de inferência que constitui este núcleo de processamento foi construída utilizando como linguagem de programação o FORTRAN 77, segundo o mesmo padrão adotado para o desenvolvimento de programas de dinâmica de vôo atualmente em uso no CRC. Estes programas são estruturados em sub-rotinas e utilizam interface tipo console, com entradas e saídas em forma de arquivos.

O software do simulador é composto por um programa principal associado a onze sub-rotinas. A figura a seguir mostra a hierarquia de componentes do simulador.

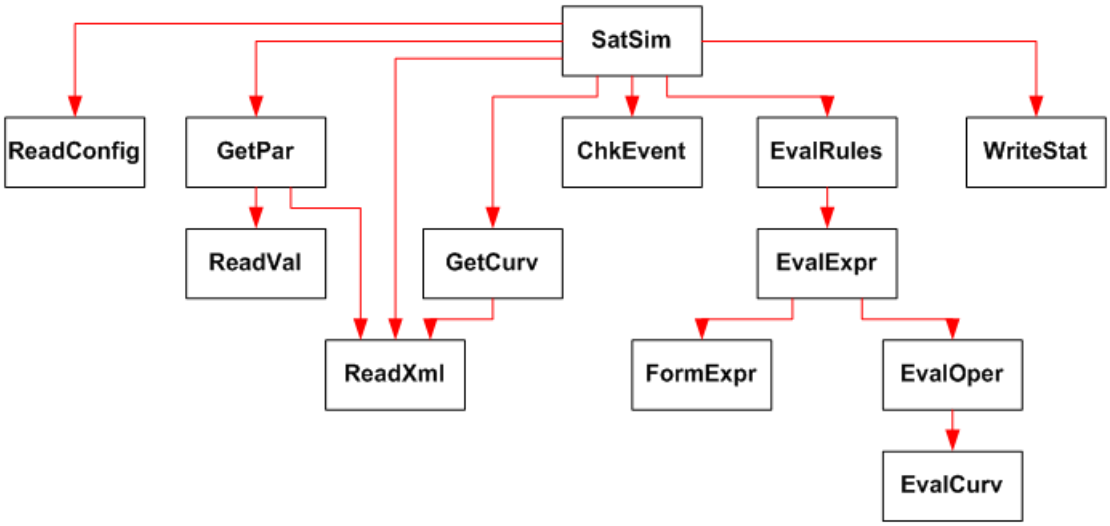


Figura 8.3 - Hierarquia de componentes do simulador

Durante a atividade de inicialização, são invocadas pelo programa principal SatSim as sub-rotinas ReadConfig para leitura do arquivo de configuração, GetPar para obtenção de parâmetros da fila de eventos, ReadXml para leitura de regras, GetCurv para obtenção de curvas, e WriteStat para inicialização da saída. As sub-rotinas GetPar e GetCurv realizam sub-chamadas a ReadXml, para análise sintática dos arquivos XML. GetPar conta ainda com a sub-rotina ReadVal para a formatação de valores numéricos a serem lidos.

Já na sessão de simulação, são chamadas pelo programa principal SatSim as sub-rotinas ChkEvent, EvalRules e WriteStat. ChkEvent realiza a verificação de eventos, EvalRules a avaliação de regras, e WriteStat exporta valores de estado interno para a saída. EvalRules chama EvalExpr para a avaliação de expressões. No início do processo de avaliação, expressões são re-formatadas através da separação de operandos e operações, realizada pela sub-rotina FormExpr. O cálculo de valores correspondentes aos resultados de operações sobre operandos é feito pela sub-rotina EvalOper. Em caso de operações envolvendo curvas de funções, EvalOper realiza uma sub-chamada a EvalCurv para a obtenção de valores de saída a partir de entradas fornecidas. Depois disto, a condição de parada da simulação é testado pelo próprio programa principal SatSim, através do qual é decidido se o passo de simulação seguinte é executado ou não.

As funções desempenhadas por cada sub-rotina que compõe o simulador encontram-se descritas brevemente na tabela a seguir, juntamente com as atividades em que se enquadram.

Tabela 8.2 - Lista de sub-rotinas componentes do simulador

Sub-rotina	Função	Atividade
ReadConfig	Leitura de caminhos do arquivo de configuração	ReadConfig
GetPar	Obtenção de parâmetros do modelo	ReadModel
ReadVal	Leitura e formatação numérica de valores de parâmetros	
ReadXml	Leitura de parâmetros a partir de arquivos XML	
GetCurv	Leitura de curvas de função a partir de arquivos XML	
ChkEvent	Verificação de disparo de eventos da fila	ChkEvent
EvalRules	Avaliação de regras de controle	EvalRules
EvalExpr	Avaliação de expressões componentes de regras	
FormExpr	Formatação de estruturas de dados de expressões	
EvalOper	Avaliação de operações de expressões	
EvalCurv	Avaliação de curva de função associada a expressões	
WriteStat	Escrita de parâmetros em histórico de estados	WriteStat

Observa-se que na tabela acima, a atividade de verificação da condição de parada (ChkStop) não se encontra na lista, por ser executada pelo programa principal SatSim.

8.3. Estruturas de dados

A sub-rotina ReadConfig executa a atividade de mesmo nome, no início a etapa de inicialização. Ela é responsável pela identificação dos arquivos de banco de dados que compõem o modelo da dinâmica do simulador. Ao fim desta atividade, o simulador obtém os nomes e caminhos dos arquivos de interface, conforme apresentados na tabela abaixo.

Tabela 8.3 - Lista de arquivos de interface do simulador

Arquivo	Descrição	Estrutura de dados
InitStat	Estado inicial	stat
EvQueue	Fila de eventos	queue
CtrlRules	Regras de controle	ctrl
StatHis	Histórico de estados	stat
FuncCurv	Curvas de funções	curv

Durante a atividade ReadModel, estruturas de dados correspondentes ao estado inicial, à fila de eventos, às regras de controle, e aos pontos de curvas são obtidas dos arquivos de banco de dados indicados pelo arquivo de configuração. Estas estruturas de dados são ilustradas na figura a seguir.

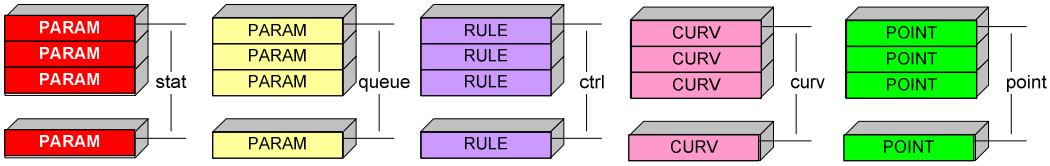


Figura 8.4 - Estruturas de dados do simulador

As estruturas de dados correspondentes ao estado interno (stat) e à fila de eventos (queue) são vetores de estruturas do tipo parâmetro (PARAM), conforme pode ser visto na figura acima. Da mesma forma, a estrutura de dados de regras de controle (ctrl) é constituída por um vetor de regras (RULE), a de curvas de função (curv) por um vetor de curvas (CURV), e a de pontos de curva (point), um vetor de pontos (POINT).

8.4. Parâmetros

No início da atividade ReadModel, a sub-rotina GetPar é invocada duas vezes. Na primeira vez, ela é usada para a construção do estado interno (stat) a partir do arquivo de estado inicial (InitStat). Na segunda vez, é gerada a estrutura de fila de eventos (queue) através da leitura do arquivo de fila de eventos (EvQueue). Conforme visto no Capítulo anterior, estes dois arquivos (InitStat e EvQueue) são estruturalmente idênticos, diferindo apenas no conteúdo. A execução da sub-rotina, em ambos os casos, resulta na obtenção de vetores de estruturas do tipo parâmetro (PARAM).

A figura a seguir mostra a estrutura de dados adotada para representar parâmetros de simulação (PARAM). Esta estrutura visa a associar uma identificação (id) e um tipo (typ) a um valor numérico (val).

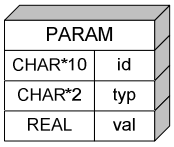


Figura 8.5 - Estrutura de dados representativa de um parâmetro de simulação

Cada parâmetro (PARAM) é composto por três campos. O identificador (id) consiste numa *string* de dez caracteres que deve permitir referenciar inequivocamente um parâmetro no modelo. O campo de tipo (typ), composto por dois caracteres, diferencia parâmetros do tipo estado daqueles do tipo evento. O campo de valor (val) corresponde a uma variável de ponto flutuante de 64 bits. Avaliações de expressões que referenciam parâmetros são realizadas com base nos valores atribuídos a este campo. A sub-rotina ReadXML é chamada para extrair as estruturas de parâmetros de dentro de arquivos XML (InitStat e EvQueue). A obtenção de valores numéricos (val) neste processo é realizada através da sub-rotina ReadVal.

O estado interno (stat) corresponde a um vetor de parâmetros de comprimento variável, carregada na memória a partir da leitura do arquivo de estado inicial (InitStat). Sua estrutura de dados encontra-se ilustrada na figura abaixo.

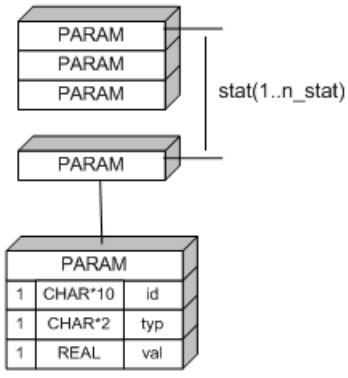


Figura 8.6 - Representação estrutural do estado interno

Cada elemento do estado interno deve possuir um identificador único. Em outras palavras, um mesmo identificador não deve ser utilizado por mais de um

parâmetro no estado interno. O estado interno contém todos os parâmetros utilizados pelo modelo, incluindo os tipo estado e os do tipo evento. Valores de parâmetros de estado indicam o estado interno do sistema, ao passo que valores de parâmetros de evento indicam o estado de disparo de eventos. O conjunto dos valores de todos os parâmetros componentes do estado interno representa o estado atual no passo de simulação da sessão corrente.

A fila de eventos (queue) consiste num vetor de parâmetros de evento, de comprimento variável, obtido do arquivo de fila de eventos (EvQueue). A estrutura da fila de eventos (queue) é idêntica à do estado interno (stat), conforme evidencia a comparação entre as figuras anterior e a seguir.

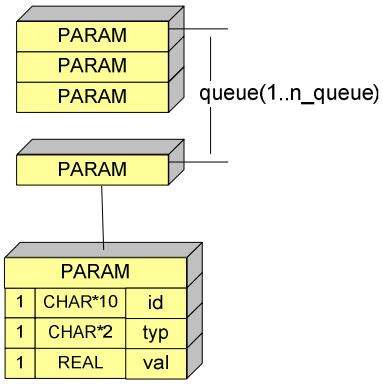


Figura 8.7 - Representação estrutural da fila de eventos

Cada elemento da fila de eventos corresponde a uma instância de disparo de um evento temporizado. Portanto, identificadores de parâmetros de evento podem se repetir numa mesma fila de eventos. O valor de cada instância de um parâmetro na fila de eventos informa o tempo de execução deste disparo, enquanto no estado interno, o valor do parâmetro correspondente representa a situação de disparo do evento no passo de simulação corrente. Este relacionamento entre parâmetros de eventos presentes no estado interno e instanciados na fila de eventos encontra-se ilustrado na figura seguinte.

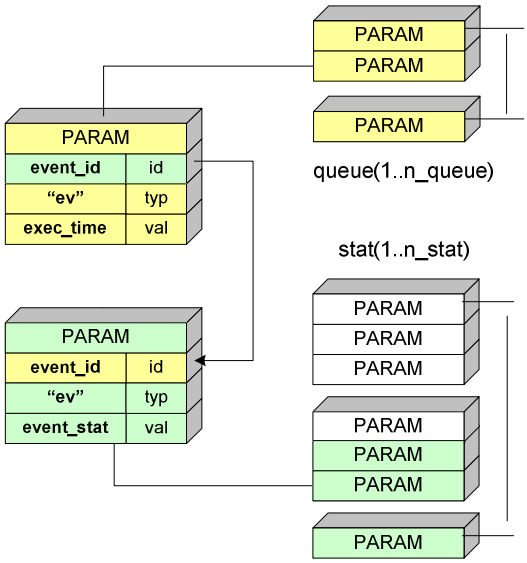


Figura 8.8 - Relacionamento entre parâmetros no estado interno e na fila de eventos

Na figura acima, o estado interno é representado em branco, e a fila de eventos, em amarelo. Parâmetros de evento são declarados no estado interno (verde), e instanciados na fila de eventos (amarelo). O relacionamento entre eles é feito através dos identificadores de parâmetro (id = event_id). A cada passo de simulação na etapa de sessão, os parâmetros de eventos declarados no estado interno (verde) recebem valores numéricos (val) que indicam seus estados de disparo (event_stat). Por default, este valor é igual a zero, o que corresponde ao valor lógico FALSO (event_stat = 0). Se o tempo de simulação do passo corrente corresponder ao tempo de execução de algum evento contido na fila (exec_time), o valor do parâmetro correspondente no estado interno assume valor numérico um, ou VERDADEIRO (event_stat = 1).

O tempo de simulação descrito acima é um parâmetro reservado, utilizado no mecanismo de controle da simulação. Alguns parâmetros têm seus identificadores (ParId e id) reservados para uso interno do simulador, devendo ser sempre declarados no estado inicial (InitStat). Estes parâmetros são listados na tabela a seguir.

Tabela 8.4 - Lista de identificadores de parâmetros reservados pelo simulador

Identificador	Tipo	Descrição
SIMTIME	Estado	Tempo de simulação associado ao passo corrente
SIMSTEP	Estado	Tempo de passo de simulação incremental
SIMSTOP	Evento	Evento de parada da sessão de simulação

Os dois primeiros parâmetros (SIMTIME e SIMSTOP) controlam a atividade de verificação de eventos (ChkStat), ao passo que o último parâmetro (SIMSTOP) é responsável pela verificação de condição de parada (ChkStop).

A atividade de verificação de disparos de eventos (ChkEvent) é realizada no início de cada passo de simulação através da sub-rotina de mesmo nome (ChkEvent). Conforme descrito anteriormente, nesta atividade os valores de parâmetros de evento contidos no estado interno (stat) são todos inicializados com valor zero (event_stat = 0). O valor do parâmetro de tempo de simulação (SIMTIME) é comparado aos valores dos tempos de execução (exec_time) associados a cada parâmetro de evento instanciado na fila de eventos (queue). Caso o tempo de execução de algum evento instanciado (exec_time) esteja dentro do intervalo de tempo definido pelo tempo de simulação (SIMTIME) e o tempo de passo de simulação (SIMSTEP), o valor do parâmetro de evento correspondente no estado interno recebe valor um (event_stat = 1). Este processo é ilustrado na figura abaixo.

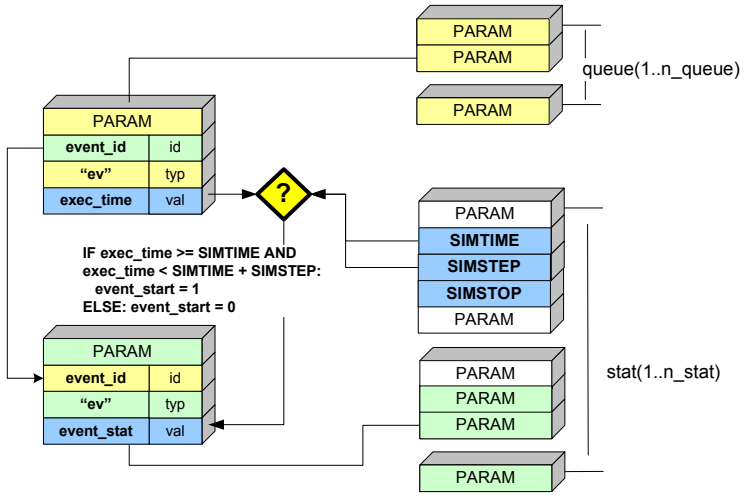


Figura 8.9 - Avaliação de disparo de eventos

O processo descrito acima pode ser utilizado para temporizar o disparo do último parâmetro reservado (SIMSTOP). Este parâmetro de evento é responsável pela sinalização da condição de parada da simulação à máquina de inferência. Durante a sessão de simulação, executam-se passos de simulação ciclicamente até que este evento seja disparado, conforme ilustrado através da figura a seguir. Neste caso, a parada ocorre quando o valor associado à instância temporizada de SIMSTOP na fila de eventos encontra-se na faixa de valores definida por SIMTIME e SIMSTEP.

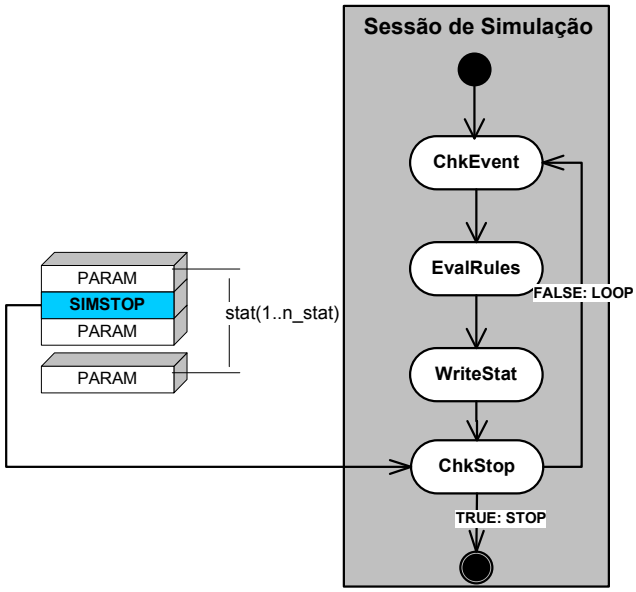


Figura 8.10 - Avaliação de condição de parada de sessão

Alternativamente, a condição de parada pode ser alcançada através de atribuição de valor ao parâmetro de evento SIMSTOP por meio da aplicação de um efeito de uma regra. Esta abordagem foi adotada para se gerar o arquivo de saída descrito no final do Capítulo anterior, sendo útil quando se deseja definir a condição de parada no arquivo de estado inicial, ao invés de fazê-lo na fila de eventos. Uma explicação sobre este tipo de situação será detalhada no próximo Capítulo, ao se descrever a configuração do modelo.

8.5. Regras

Eventos disparados e valores de parâmetros de estado corrente são utilizados durante a atividade de avaliação de regras de controle (EvalRules). A cada passo de simulação de uma sessão, todas as regras (RULE) que compõem a estrutura de regras de controle (ctrl) são avaliadas uma a uma, segundo a ordenação definida dentro do arquivo de regras de controle (CtrlRules).

Cada regra é constituída por uma condição e um ou mais efeitos. Uma condição é uma expressão lógica. Se a avaliação da condição resultar num valor verdadeiro, os efeitos são aplicados, um a um. Cada efeito consiste numa expressão numérica associada a um parâmetro alvo. O valor de cada parâmetro alvo é substituído pelo resultado da avaliação da expressão associada. A estrutura adotada para a representação de regras de controle no simulador é mostrada na figura a seguir.

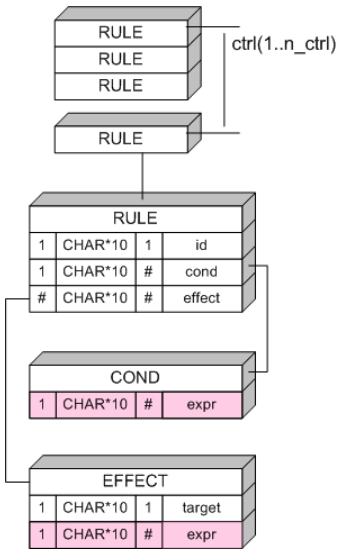


Figura 8.11 - Estrutura de regras de controle

Cada estrutura do tipo regra (RULE) é composta por um identificador (id) de 10 caracteres, uma condição (cond) composta por uma expressão lógica (expr), e um ou mais efeitos (effect). Cada efeito (effect) é composto por um identificador de parâmetro alvo (target), mais uma expressão numérica (expr) associada. A

implementação do simulador é tal que a estrutura de expressão é comum (expr) para representação de expressões lógicas e numéricas empregadas nas condições e efeitos, conforme destacado em rosa na figura acima. Uma estrutura de expressão (expr) é representada por um vetor de comprimento variável cujos elementos são *strings* de 10 caracteres.

O mecanismo de avaliação de regras de controle é ilustrado na figura a seguir.

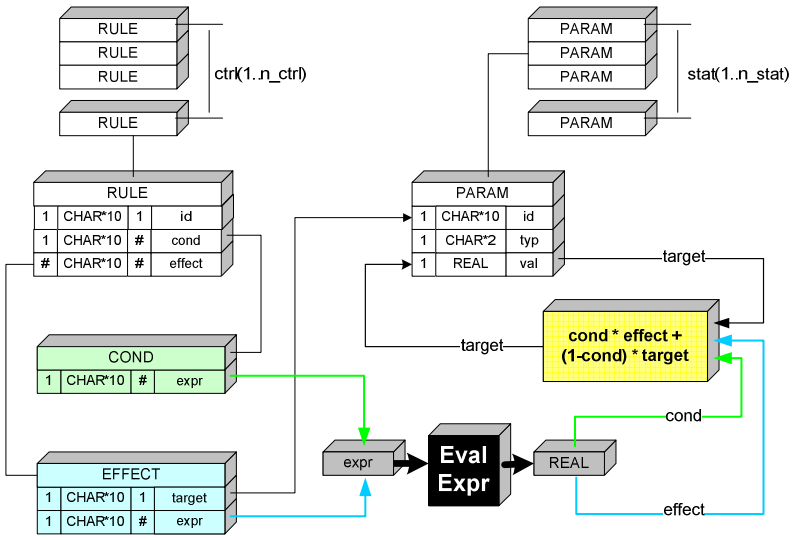


Figura 8.12 - Avaliação de regras de controle

A avaliação seqüencial de regras (RULE) que compõem a estrutura de regras de controle (ctrl) é desempenhada pela sub-rotina EvalRules. A avaliação de uma regra (RULE) consiste na decisão pela manutenção ou substituição de valores de parâmetros alvos (target) por valores (effect) resultantes da avaliação de expressões (expr) de efeito (EFFECT). Esta decisão é realizada com base nos valores (cond) resultantes da avaliação da expressão (expr) de condição (COND). Resultados de condição (cond) e efeito (effect) são valores numéricos reais obtidos por meio de avaliação de suas expressões (expr), realizadas pela sub-rotina EvalExpr.

Segundo a lógica booleana, uma regra é disparada caso a avaliação de sua expressão de condição resulte no valor VERDADEIRO. Convencionou-se a atribuição de valor lógico VERDADEIRO ao valor numérico 1, e de valor lógico FALSO ao valor numérico 0. Quando uma regra é disparada (cond = 1), os valores dos parâmetros alvos são substituídos por resultados de expressões de efeito (target = effect), e mantidos caso contrário. A implementação do simulador é tal que se permite a atribuição de um valor ponderado a parâmetros alvos, se a avaliação da condição resultar num valor intermediário entre 0 e 1. Esta implementação permite simular situações de transição entre um passo de simulação e outro. Um exemplo deste tipo de situação inclui o estado de penumbra que ocorre na transição entre condições de sombra e iluminação de um sensor ou painel solares. Em termos de lógica nebulosa, esta implementação corresponde à implementação de uma função de pertencimento do tipo rampa, de coeficiente 1 para 1, definida no intervalo de 0 a 1, associada às saídas de regras nebulosas.

8.6. Expressões

Expressões de regras são utilizadas para descrever em uma seqüência de procedimentos a serem executados a fim de se obter um valor numérico de saída. Esta descrição é realizada por meio de um vetor de elementos de expressão, que podem ser subdivididos em operandos e operações. Operandos de expressões podem ser variáveis, constantes numéricas, expressões aninhadas, ou curvas de funções definidas pelo usuário. Operações de expressões incluem operações lógicas e aritméticas, e funções elementares ou definidas por usuário.

A estrutura adotada para representar uma expressão (expr) é representada através de um vetor de tamanho variável de elementos, conforme ilustrado na figura abaixo.

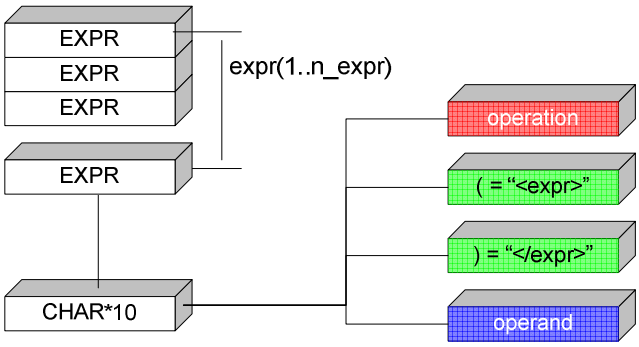


Figura 8.13 - Estrutura de expressões

Cada elemento de uma expressão (EXPR) consiste numa *string* composta por 10 caracteres, podendo representar uma operação (operation), um operando (operand), ou sinalizações de início (parêntese esquerdo = <expr>) e fim (parêntese direito =</expr>) de aninhamento de expressões.

A avaliação de uma estrutura de expressão (expr) é executada pela sub-rotina EvalExpr, retornando na saída um valor numérico real. Antes de serem avaliadas, expressões devem ser re-formatadas para compor seqüências de operandos e operações, conforme mostrada na figura seguinte.

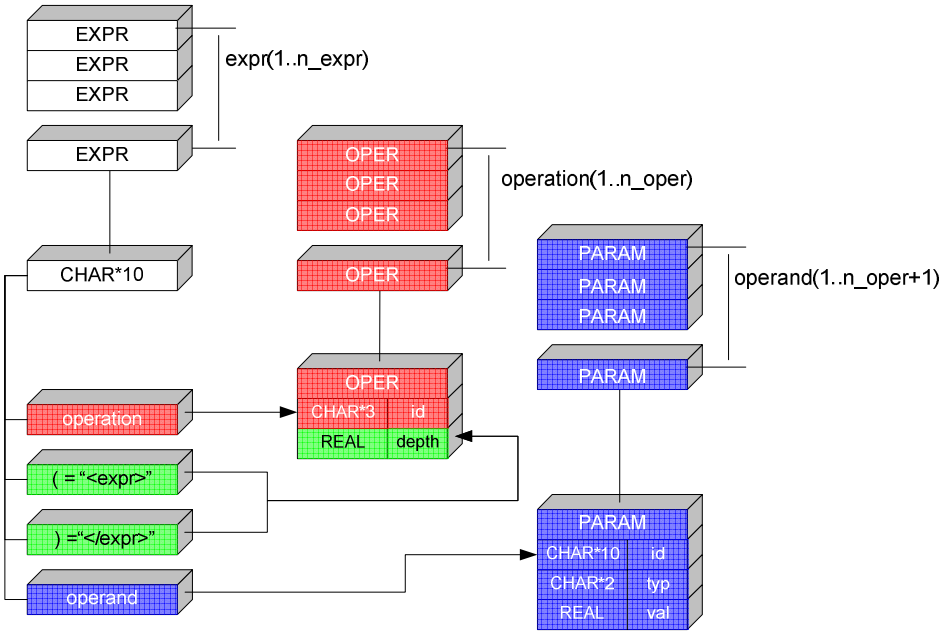


Figura 8.14 - Re-formatação de expressões

Durante a re-formatação da expressão, o simulador associa a cada elemento da estrutura de operandos (operand) um parâmetro (PARAM), e a cada elemento da estrutura de operações (operation) uma estrutura de dados específica para representar operações (OPER). Cada parâmetro associado a operandos (PARAM) é preenchido para poder representar um parâmetro de simulação ou uma curva de funções através de seu identificador (id), ou uma constante numérico através de seu valor (val). Operações (OPER) são identificadas através de códigos de três caracteres, tendo associados campos de profundidade (depth) atribuídos de acordo com o nível de aninhamento dentro da expressão.

De acordo com a figura anterior, a re-formatação de uma expressão deve gerar uma estrutura de operandos (operand) contendo um elemento a mais do que a estrutura de operações (operation). Durante a re-formatação, os elementos de uma expressão são alocados em estruturas de operandos e operações. Uma forma de visualização destas estruturas útil para a compreensão do processo de avaliação de expressões consiste em compor uma seqüência alternada de operandos e operações, iniciadas e finalizadas por um operando, com intercalação de operações. Segundo esta visualização, uma expressão é composta por um número de elementos igual a 2*n_oper+1, sendo n_oper operações e n_oper+1 operandos. Dado um número qualquer n no intervalo de 1 a n_oper, a operação n encontra-se cercada pelos operandos adjacentes de número n e n+1. Exemplos representativos de expressões segundo esta visualização são ilustrados na figura a seguir.

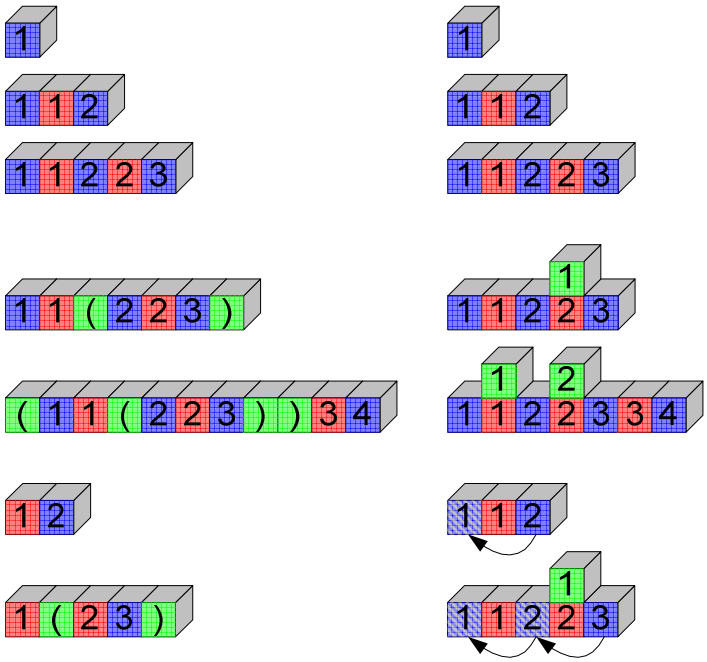


Figura 8.15 - Exemplos de re-formatação de expressões

Na figura acima são mostradas sete exemplos de expressões, à esquerda conforme elas são representadas dentro da estrutura de regras, e à direita como são dispostas após a re-formatação. Cada bloco colorido representa um elemento de expressão (EXPR), com operandos sendo mostrados em azul, operações em vermelho, e aninhamentos em verde. Uma expressão é formada por elementos ordenados seqüencialmente da esquerda para a direita. Os números atribuídos a blocos azuis e vermelhos correspondem à posição de cada elemento a ser atribuído na estrutura de operandos e operações, respectivamente, após a re-formatação. Os números atribuídos a blocos verdes no lado direito, localizados acima de blocos vermelhos, correspondem a valores de profundidade (depth) atribuídos a operações (OPER). Deste lado da figura, blocos vermelhos sem blocos verdes visíveis acima representam operações com profundidade zero.

Os três primeiros exemplos, localizados na porção superior da figura, representam casos em que as expressões não apresentam aninhamento de

expressões, e são compostas apenas por operandos e operações de dois termos. Nestes casos, a re-formatação é direta e trivial.

Os dois exemplos seguintes, na posição intermediária da figura, representam casos em que as expressões apresentam aninhamento de expressões, mas são compostas apenas por operandos e operações de dois termos. Nestes casos, a re-formatação deve ser feita levando em consideração a profundidade das operações. A profundidade de uma operação é calculada a partir da contagem de sinalizações de inícios de aninhamento (parênteses esquerdos = “(” verdes) subtraída da contagem fins de aninhamento (parênteses direitos = “)” verdes), contados até se chegar à operação.

Os dois últimos exemplos mostrados na parte inferior da figura representam casos de uso de operações unárias. Nestes exemplos, utilizam-se operandos com valores reproduzidos a partir de seu vizinho, para manter a integridade da estrutura alternada. Os valores de operandos indeterminados são preenchidos a posteriori, depois que se tornam conhecidos os valores que eles devem assumir.

O processo de re-formatação é realizado pela sub-rotina FormExpr, conforme procedimento descrito acima. Depois disto, operações são aplicadas sobre operandos adjacentes em ordem decrescente de profundidade. O resultado de uma operação substitui os valores de ambos os operandos adjacentes, que a cercam. A figura abaixo ilustra graficamente o processo descrito acima, aplicando-o sobre os exemplos da figura anterior.

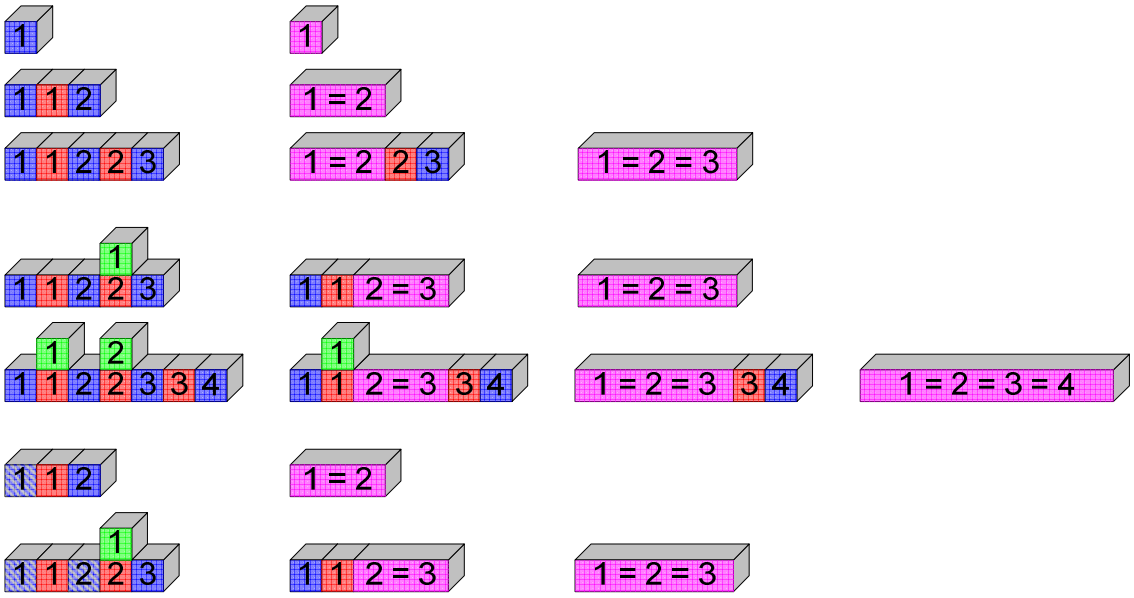


Figura 8.16 - Exemplos ilustrativos de avaliação de expressões

Na figura acima, a evolução da execução dos processos de avaliação ocorre da esquerda para a direita. Resultados de operações são representados em rosa. Este processo de cálculo de operações é realizado através da sub-rotina EvalOper. Ao fim de todo o processo, o resultado após a aplicação de todas as operações encontra-se replicado em todos os operandos da expressão, conforme figura acima. Este é o valor final retornado pela sub-rotina EvalExpr. Em caso de erro de avaliação, o valor retornado é uma combinação de bits correspondente a infinito.

8.7. Curvas

No início do projeto, o objetivo principal das curvas de funções era fornecer um mecanismo preliminar de diagnóstico, por meio de disparo de alarmes. Entretanto, ao longo do desenvolvimento foi percebida a possibilidade de inclusão de funções de usuário, permitindo-se a simulação de conversores de dados embarcados e curvas de calibração de telemetrias. Além disto, estas curvas poderiam ser representar funções de pertencimento nebulosas, a fim de configurar regras de controle nebulosas.

A inicialização de estruturas de dados associadas a curvas de função é realizada através da sub-rotina ReadCurv. O cálculo de operações envolvendo curvas de ambos os tipo descritos acima é executada pela sub-rotina EvalCurv, quando invocados pela sub-rotina EvalOper. A estrutura de dados adotada para a representação de curvas de funções é mostrada na figura abaixo, juntamente com a estrutura de dados auxiliar elaborada para armazenar seus pontos.

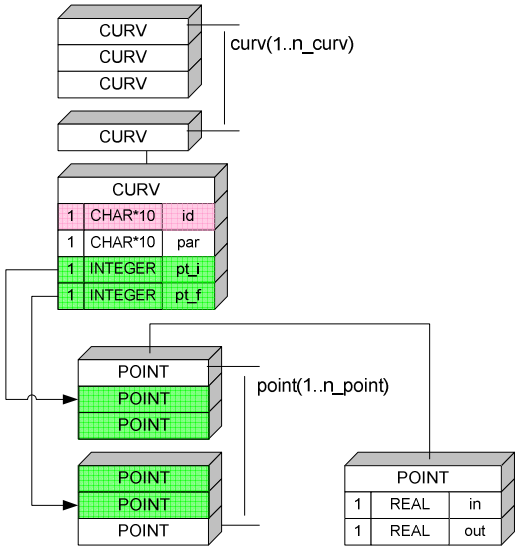


Figura 8.17 - Estrutura de curvas de funções e pontos de curvas

Uma curva de função (CURV) é composta por um identificador de curva (id) de 10 caracteres, um identificador de parâmetro associado (par) opcional, também de 10 caracteres, e dois ponteiros de endereço de pontos, um inicial (pt_i) e um final (pt_f), representados por variáveis inteiras de 32 bits. Uma estrutura de dados representando os pontos de curvas (points) contém os pontos (POINT) empregados por todas as curvas (CURV) presentes no modelo. A tabela de curvas empregada por uma curva de função é obtida a partir dos pontos de curvas a partir dos ponteiros de endereço de pontos inicial (pt_i) e final (pt_f).

Dentro de uma expressão, uma curva de função pode ser utilizada como um operando, ou como uma operação. A figura abaixo mostra dois exemplos de expressões com uso de curvas de funções.

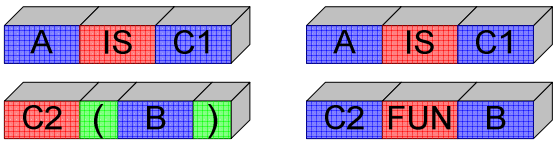


Figura 8.18 - Exemplos de re-formatação de expressões com curvas de funções

No primeiro exemplo, mostrado na parte superior da figura, a curva C1 é utilizada como função de pertencimento de uma regra nebulosa, sendo associada ao parâmetro A. No segundo exemplo, na parte inferior da figura, a curva C2 é usada como uma função de usuário para mapeamento de saída dada uma entrada B. Observe-se que, durante a re-formatação da expressão no lado direito da figura, ambas as curvas são atribuídas a operandos. No caso específico do segundo exemplo, seria mais correto afirmar que a curva é desmembrada em um operando e uma operação, conforme será detalhado adiante.

Curvas de funções associadas a parâmetros são expressas na forma de seqüências re-formatadas conforme mostrado na figura a seguir. A avaliação da expressão é invocada a partir da operação IS, que identifica uma curva de função no segundo operando, associada ao parâmetro indicado no primeiro operando.

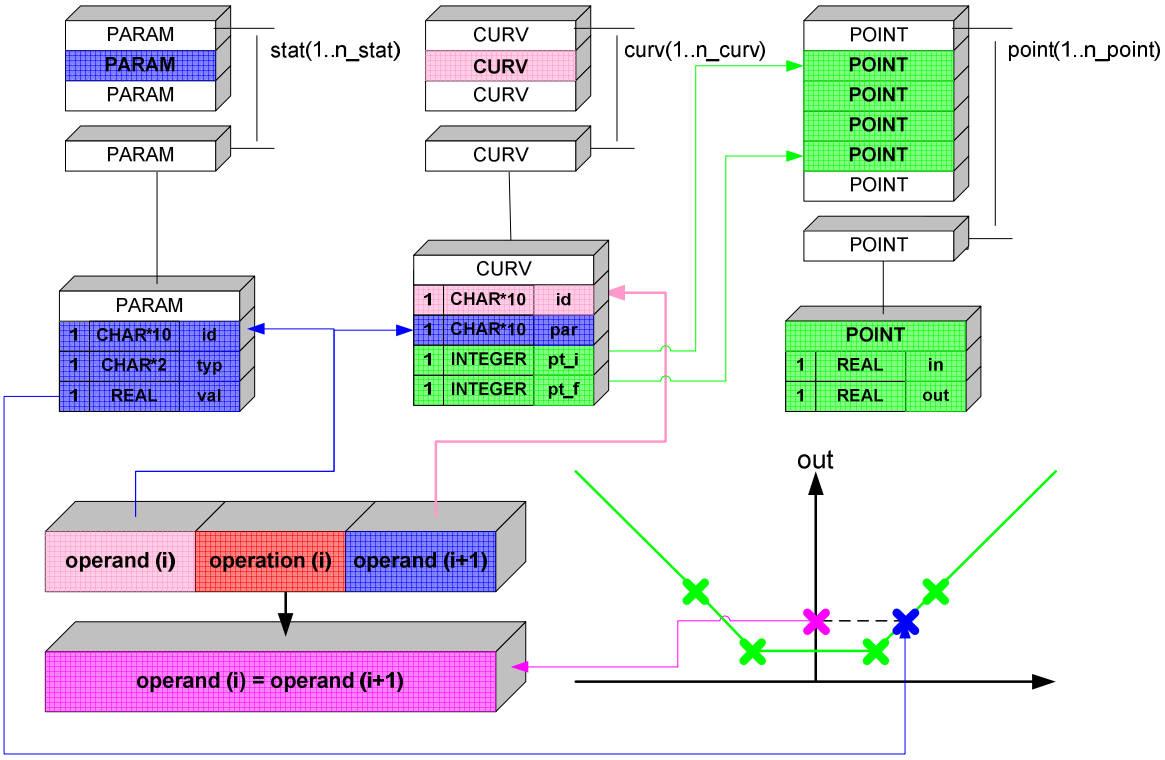


Figura 8.19 - Exemplo ilustrativo de operação com curva associada a parâmetro

Curvas de funções utilizadas para mapeamento, independentes de parâmetros, são re-formatadas conforme mostrado na figura seguinte. Neste caso, a informação referente à curva de função é armazenada num operando, e a avaliação da expressão é realizada por meio de uma operação implícita FUN, cuja existência não é reconhecida em nível de regras de controle. Esta operação calcula o resultado a partir de uma curva de função identificada no primeiro operando, utilizando como entrada o segundo operando.

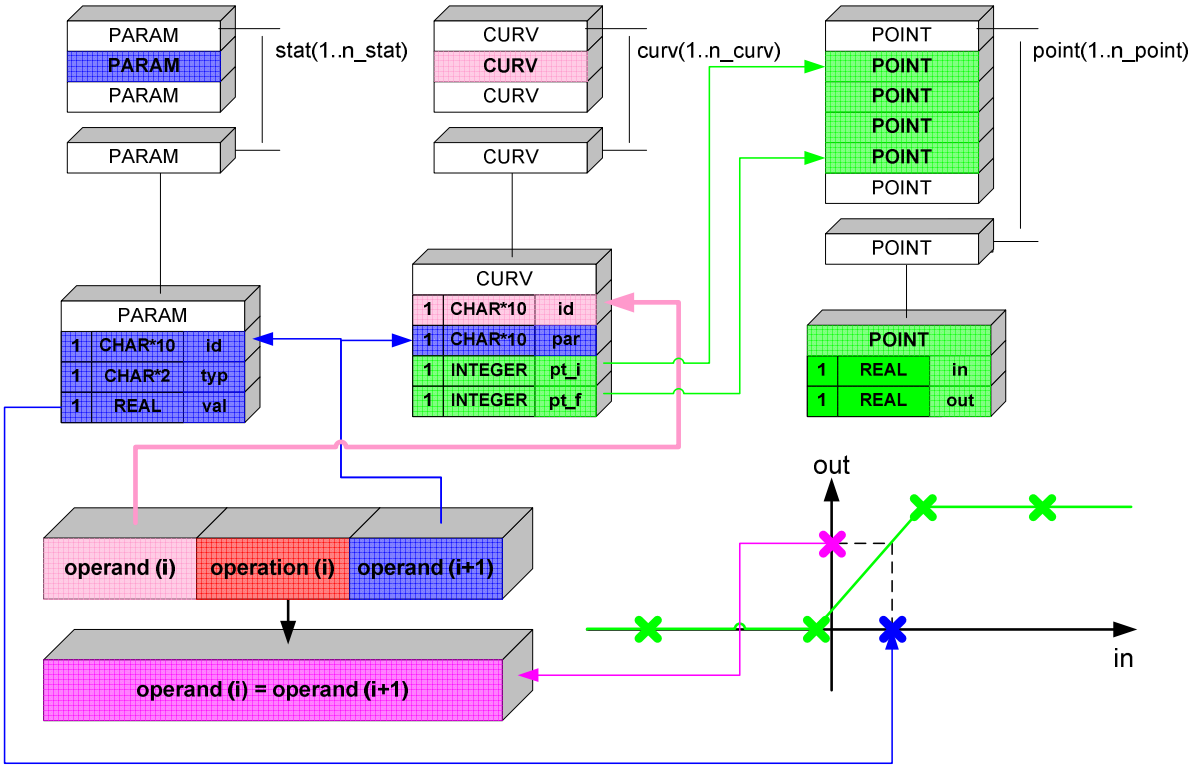


Figura 8.20 - Exemplo ilustrativo de operação com curva não associada a parâmetro

8.8. Interfaces

As interfaces do simulador com os arquivos de entrada e saída são realizadas através de três sub-rotinas. A leitura do arquivo de configuração é realizada por ReadConfig, o acesso a estruturas de banco de dados em formato XML via ReadXml, e a escrita em formato CSV através da sub-rotina WriteStat. O formato de cada arquivo segue a descrição do Capítulo anterior.

Para correto funcionamento, o arquivo de configuração deve estar presente na mesma área onde se localiza o aplicativo do núcleo de processamento do simulador. Seu nome deve ser SATSIM.CFG.

Os arquivos de configuração e os bancos de dados em XML são criados através de ferramentas auxiliares durante a etapa de configuração do modelo. Eles serão objeto de descrição no próximo Capítulo.

9 CONFIGURAÇÂO DO MODELO

O simulador de satélite proposto funciona em três etapas, dos quais dois foram tratados em Capítulos anteriores. Até o momento foram descritos a inicialização dos arquivos de banco de dados que caracterizam o modelo do simulador, e a sessão de simulação executada pela máquina de inferência que constitui o núcleo de processamento. O tema deste Capítulo centra-se na configuração do modelo, a começar pela descrição do pré-processador de eventos e um estudo acerca do configurador do modelo.

9.1. Visão geral

O simulador utiliza modelos armazenados em banco de dados, descritos na forma de arquivos XML. Este modelo contém informações referentes a parâmetros de estados iniciais e fila de eventos, regras de controle do sistema, e curvas de funções associadas. A edição manual de arquivos XML é uma abordagem válida para a implantação de sistemas simples, descritos por um número reduzido de parâmetros e regras. Este enfoque atende as necessidades do CRC para a validação de software experimental de planejamento de operações de satélites. Entretanto, no caso de modelos mais complexos, torna-se comum dividir o sistema em unidades menores, ou subsistemas, para melhor gerenciamento de informações. Enquanto o modelo do simulador reflete de forma adequada o conhecimento de especialistas de sistema, sua reconfiguração freqüente conforme informações atualizadas providas por especialistas de subsistemas pode se tornar complicado e susceptível a erros. (TOMINAGA et al., 2010)

Documentos de especificação de requisitos de projeto de satélites são elaborados antes do lançamento. Normalmente, eles mantêm-se imutáveis ao longo da vida da missão. Já seus requisitos operacionais mudam conforme o estado de funcionamento do satélite, o que por sua vez afeta os requisitos de missão. Por exemplo, uma falha permanente num equipamento embarcado

pode levar a alterações nas prioridades de operações de missão das cargas-úteis, conforme atualizações das margens de segurança para operações seguras. A fim de permitir a validação correta de planos de operações, o simulador deve ter seu modelo constantemente calibrado, em conformidade com as observações atuais que refletem o estado interno corrente do satélite.

Desta forma, propõe-se uma ferramenta visual para auxiliar na configuração de parâmetros, regras, e curvas do modelo do simulador, tendo como objetivo auxiliar na tarefa de atualização dos modelos de simuladores. Além de viabilizar a reconfiguração mais freqüente dos simuladores, esta ferramenta permitiria a elaboração de modelos mais complexos sem prejuízo de inteligibilidade, o que poderia atender possíveis necessidades futuras que iriam além do planejamento de operações.

O modelo do simulador é composto por arquivos de estado inicial, fila de eventos, regras de controle, e curvas de funções. Todos estes arquivos são codificados num formato de arquivo XML. A fila de eventos é gerada a partir do plano de operações em vôo e da previsão de eventos orbitais, através de um pré-processador de eventos. A fim de facilitar seu uso de evitar erros humanos, propõe-se uso de um configurador de modelos para a edição do estado inicial, das regras de controle e das curvas de funções. Os casos de uso identificados para estas duas ferramentas são mostradas na figura abaixo, que representa o diagrama de casos de uso da configuração de modelo do simulador.

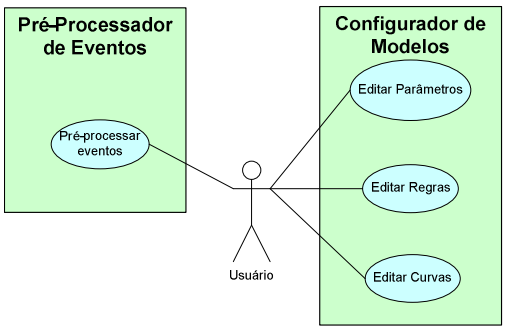


Figura 9.1 - Diagrama de casos de uso da configuração do modelo

9.2. Pré-processador de eventos

O sistema de planejamento corrente não gera planos de operações estruturados em XML, mas sim em um formato padronizado de arquivo texto, denominado POV. Um exemplo de arquivo POV encontra-se representado na figura abaixo.

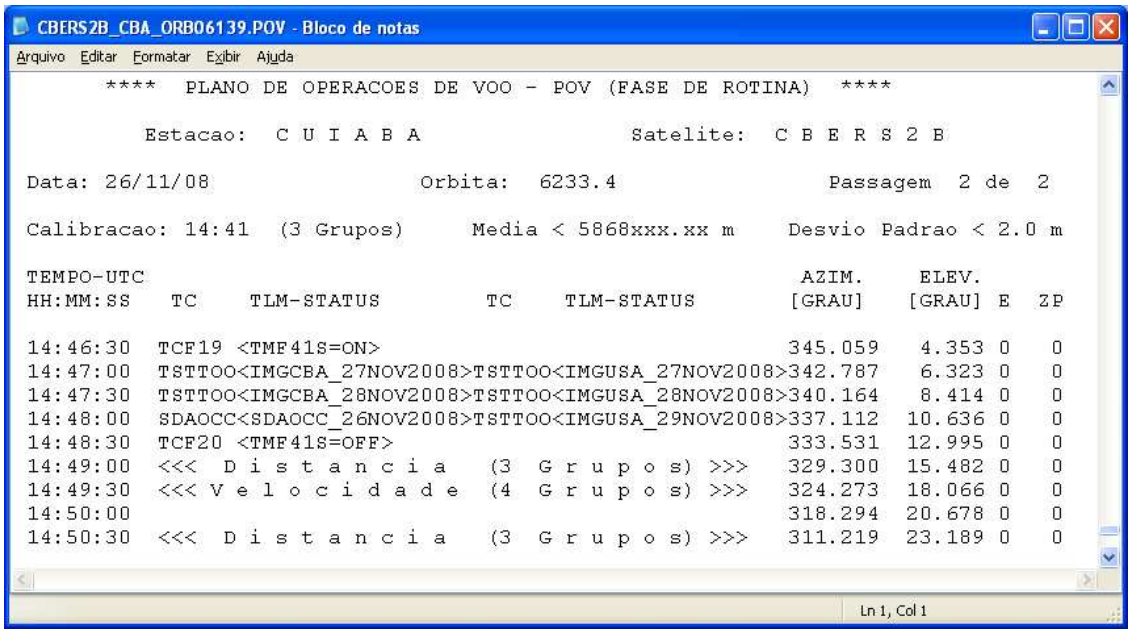


Figura 9.2 - Exemplo de plano de operações em formato POV

Observa-se na parte inferior da figura acima nove linhas de texto correspondentes ao período de tempo compreendido entre 26/Nov/2008 14h46min30s e 26/Nov/2008 14h50min30s GMT. Na porção central destas linhas observam-se atividades a serem executadas. Ás 14h46min30s há um envio de telecomando, às 14h47min00s, 14h47min30s e 14h48min00s dois envios de telecomando, às 14h48min30s o envio de um telecomando, às 14h49min00s um disparo de três grupos de medidas de distância, às 14h49min30s um disparo de quatro grupos de medidas de velocidade, e às 14h50min30s outro disparo de três grupos de medidas de distância. Para o simulador de satélite, interessa simular a execução dos telecomandos. Cabe ao

pré-processador de eventos converter estas informações contidas no POV em um formato XML, para que o simulador possa processá-lo.

Na figura anterior, os seis telecomandos planejados para serem enviados às 14h47min00s, 14h47min30s e 14h48min00s constituem telecomandos de execução temporizada, associados a arquivos externos. Neste exemplo, chamadas a telecomandos do tipo TSTTOO são associadas a arquivos de telecomandos temporizados que contém filas de telecomandos temporizados em blocos, como mostra o exemplo da figura abaixo.

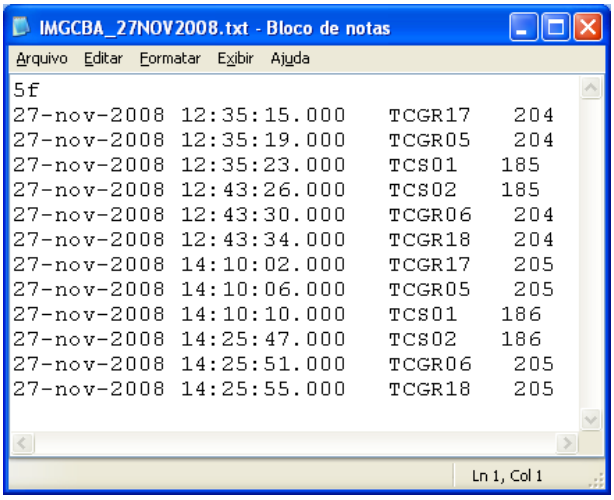


Figura 9.3 - Exemplo de arquivo de telecomandos temporizados

É importante observar que telecomandos tornam-se diferentes conforme mudam os satélites. Um pré-processador de eventos configurado para gerar filas de eventos a partir de um arquivo POV exemplificado anteriormente dificilmente seria capaz de processar um POV de outro satélite, como um exemplificado na figura a seguir.

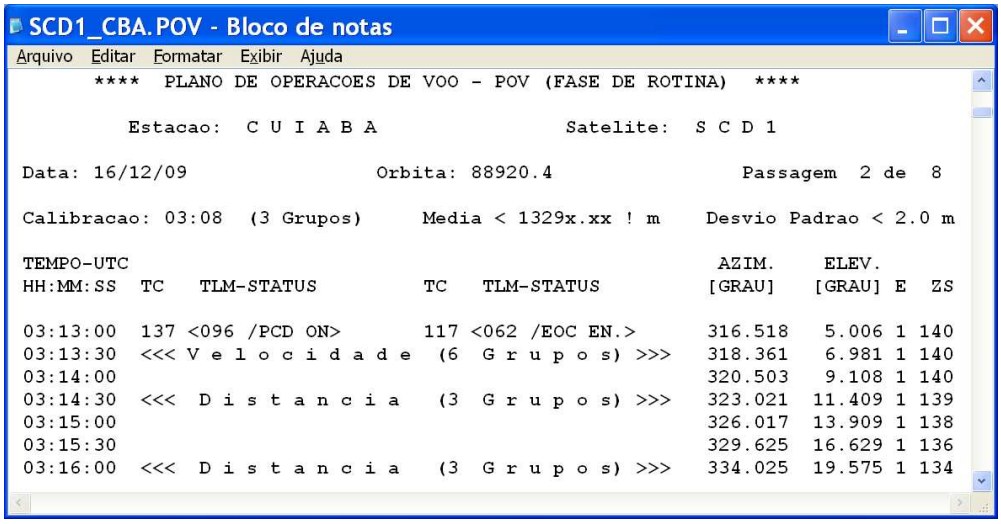


Figura 9.4 - Outro exemplo de plano de operações em formato POV

Em uma etapa imediatamente anterior ao planejamento de operações, a dinâmica de vôo gera um arquivo de previsão de eclipses. Este arquivo, também chamado arquivo ECL, contém informações referentes à situação de iluminação do satélite pelo sol, o que faz do arquivo ECL um exemplo típico de previsão de eventos orbitais. Durante a fase iluminada, o satélite é capaz de gerar energia elétrica por meio de geradores solares. Durante a fase de eclipse, cabe à bateria embarcada suprir a energia em bordo. A figura a seguir apresenta um exemplo de arquivo ECL.

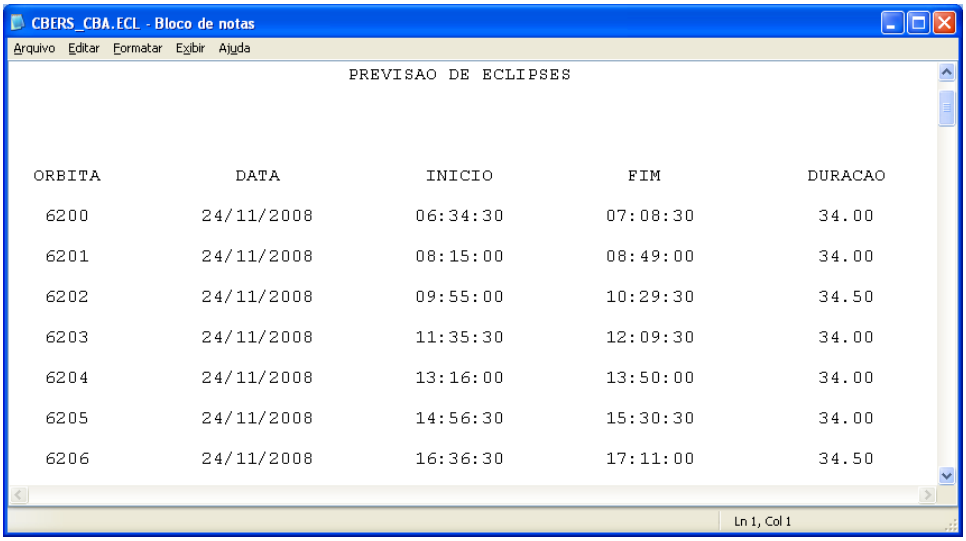


Figura 9.5 - Exemplo de arquivo de previsão de eclipses

Na figura anterior, a primeira coluna corresponde ao número de órbitas do satélite, a segunda coluna a data, a terceira coluna o horário de início da fase de eclipse e fim da fase iluminada, a quarta coluna o horário de fim da fase de eclipse e início da fase iluminada, e a última coluna, a duração do período de eclipse na órbita. Desta forma, a combinação da terceira, quarta e quinta colunas fornece os horários de disparo de eventos de início de fase de eclipse e início de fase iluminada, a serem utilizadas na simulação.

Assim, chega-se a uma especificação para o pré-processador de eventos, que deve ser capaz de gerar um arquivo de fila de eventos conforme apresentado em um Capítulo anterior, a partir da leitura de campos de dados contidos em arquivos POV e ECL. De preferência, o pré-processador de eventos deve ser configurável para permitir o processamento de arquivos POV de diferentes satélites. Dependendo da missão, a configuração pode indicar arquivos externos contendo procedimentos pré-programados. O diagrama de atividades do pré-processador de eventos proposto é apresentado na figura abaixo.

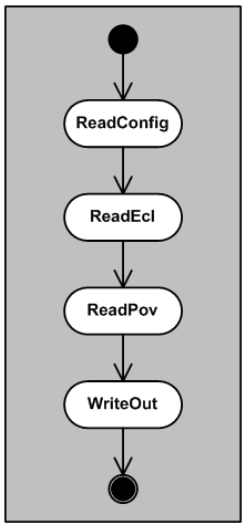


Figura 9.6 - Diagrama de atividades do pré-processador de eventos

A figura acima mostra quatro atividades executadas em seqüência. Inicialmente, é lido um arquivo de configuração que define os caminhos dos arquivos de entrada assim como informações auxiliares necessárias para a conversão de telecomandos em eventos. (ReadConfig). Em seguida, campos

de início e fim de eclipse são extraídos do arquivo ECL conforme especificado (ReadEcl). Depois disto, telecomandos são obtidos do POV e eventuais arquivos auxiliares (ReadPov). Por fim, a fila de eventos montada é exportada para o arquivo de saída, em formato XML (WriteOut). Em cada atividade, o pré-processador de eventos identifica os arquivos de entrada e os campos relevantes para leitura e processamento.

Um protótipo de pré-processador de evento foi codificado em FORTRAN 77, tirando proveito de bibliotecas geradas para a construção do núcleo de processamento do simulador de satélite e códigos legados da dinâmica de vôo. Seus componentes são mostrados na figura a seguir.

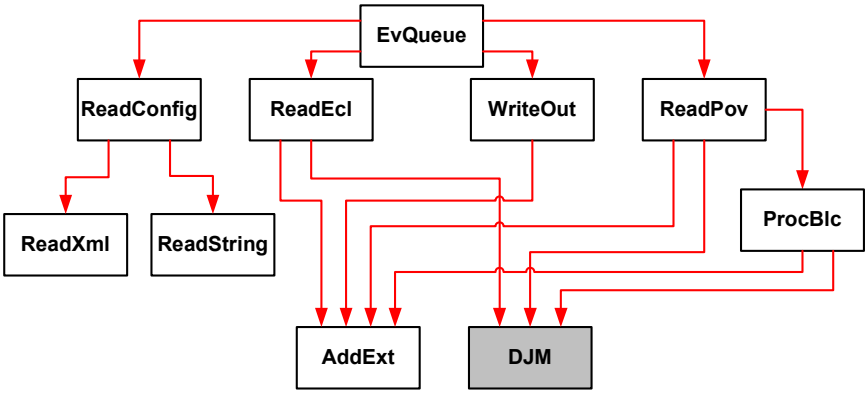


Figura 9.7 - Hierarquia de componentes do pré-processador de eventos

Para desempenhar a atividade ReadConfig, o programa principal EvQueue invoca a sub-rotina ReadConfig para leitura do arquivo de configuração. O arquivo de configuração deste programa é formato XML, conforme mostrado na figura abaixo. Por causa disto, ele faz uso da sub-rotina ReadXml, para análise sintática da estrutura XML, conforme definido para os bancos de dados do simulador. Na figura abaixo se percebe também o uso de elementos XML na forma de *strings* com uso de aspas e referências a entidades (< e >), que haviam sido excluídas modelo do simulador. Para processá-los, a sub-rotina ReadConfig conta ainda com a sub-rotina ReadString.

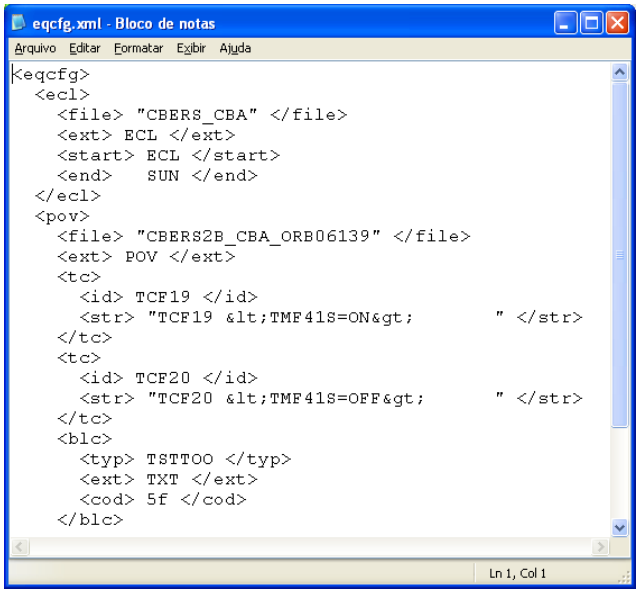


Figura 9.8 - Arquivo de configuração do pré-processador de eventos

Os campos de dados que compõem o arquivo de configuração do pré-processador de eventos são listados na tabela abaixo.

Tabela 9.1 - Lista de campos de dados componentes do arquivo de configuração

Mnemônico	Chave	Descrição
EqCfg	<eqcfg>	Arquivo de configuração do pré-processador de eventos
Ecl	<ecl>	Arquivo de previsão de eclipses ECL
EclFile	<file>	Nome do arquivo ECL
EclExt	<ext>	Extensão do arquivo ECL
EclStart	<start>	Nome do parâmetro de evento de início de eclipse
EclEnd	<end>	Nome do parâmetro de evento de fim de eclipse
Pov	<pov>	Arquivo de plano de operações em vôo POV
PovFile	<file>	Nome do arquivo POV
PovExt	<ext>	Extensão do arquivo POV
PovTc	<tc>	Telecomando
TcId	<id>	Nome do parâmetro de evento de telecomando
TcStr	<str>	String para referenciar o telecomando no POV
PovBlc	<blc>	Bloco de telecomando referenciado em arquivo
BlcType	<typ>	Tipo do bloco de telecomando
BlcExt	<ext>	Extensão do arquivo de bloco de telecomando
BlcCode	<cod>	Tipo do bloco de telecomando
Out	<out>	Arquivo de saída de fila de eventos XML
OutFile	<file>	Nome do arquivo XML
OutExt	<ext>	Extensão do arquivo XML

Após o processamento do arquivo de configuração, o programa principal EvQueue chama a sub-rotina ReadEcl para executar a atividade de mesmo nome. Ao fim desta atividade, obtêm-se os eventos EclStart e EclEnd.

A atividade ReadPov é executada em seguida, através da sub-rotina de mesmo nome. Ao fim desta atividade, obtêm-se os eventos do tipo PovTc e, se houver, eventos temporizados de blocos PovBlc processados pela sub-rotina ProcBlc.

Por fim, a atividade WriteOut é executada também pela sub-rotina de mesmo nome. Ao fim desta atividade, gera-se um arquivo XML contendo os eventos temporizados contidos nos arquivos ECL, POV e blocos associados, em formato DJM.

As sub-rotinas ReadEcl, ReadPov, ProcBlc e WriteOut fazem sub-chamadas à sub-rotina AddExt para acoplar a extensão (EclExt, PovExt, BlcExt e OutExt) aos respectivos nomes de arquivos de entrada (EclFile, PovFile, BlcFile OutFile) a fim de abri-los. Somente BlcFile é lido do POV, sendo o resto obtido do arquivo de configuração.

As sub-rotinas acima, à exceção de WriteOut, também utilizam a função DJM para formatação de tempos de eventos (EclStart, EclEnd, PovTc e BlcTc) em DJM. Somente BlcTc é lido do arquivo de bloco apropriado, sendo o restante obtido do arquivo de configuração.

A figura a seguir mostra um exemplo de arquivo de saída em XML gerado pelo pré-processador de eventos. A esquerda da figura representa um trecho obtido através do arquivo ECL, e a direita, outro trecho obtido pela leitura do POV.

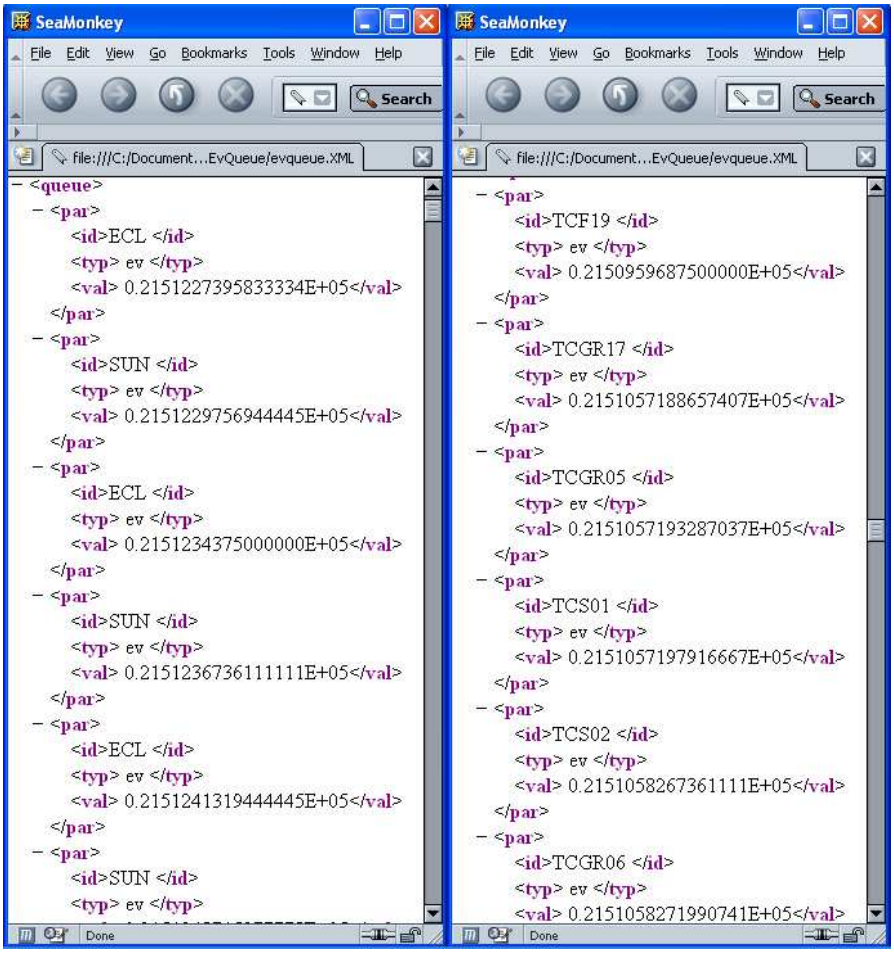


Figura 9.9 - Arquivo de saída do pré-processador de eventos

9.3. Configurador de modelos

O configurador de modelos tem como objetivo fornecer uma ferramenta visual para permitir fácil configuração de arquivos de estado inicial, regras de controle e curvas de funções, utilizados pelo simulador como bancos de dados de modelo do sistema. Um protótipo de configurador de modelos começou a ser desenvolvido em linguagem Java, visando ao atendimento destes requisitos. Sua interface de abertura é mostrada na figura seguinte.

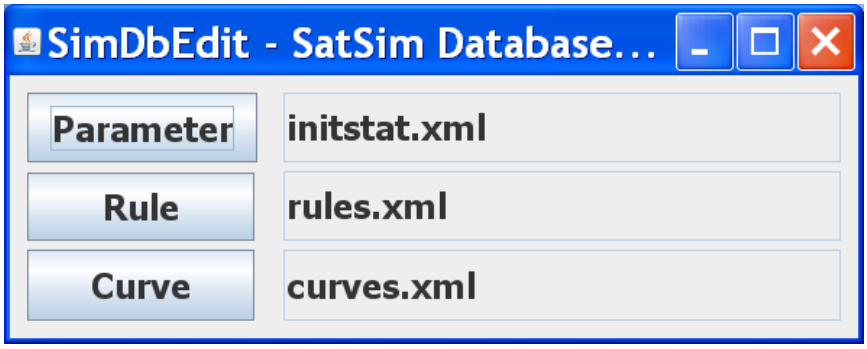


Figura 9.10 - Interface de abertura do protótipo de configurador de modelos

A interface de abertura é dotada de três botões grandes, localizados no lado esquerdo da janela. À direita de cada botão é exibido o caminho do arquivo XML associado. Um clique no botão superior leva o usuário à interface de edição de parâmetros do estado inicial, inferior à edição de curvas de funções, e o do meio, à edição de regras de controle. A interface de edição de parâmetros do estado inicial é mostrada na figura abaixo.

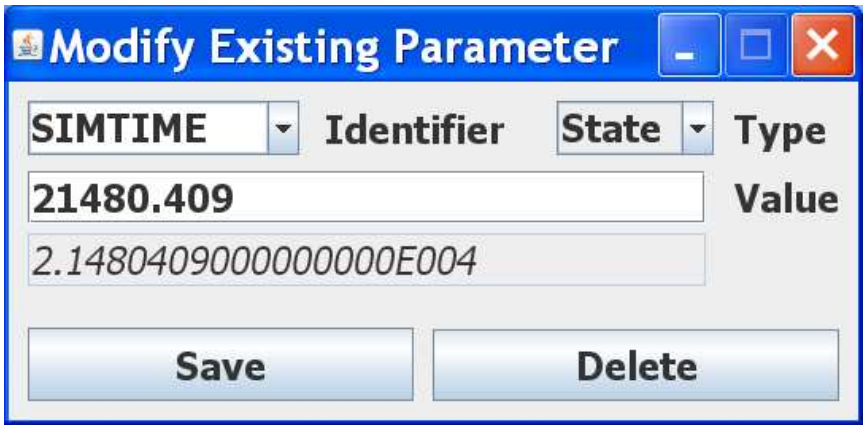


Figura 9.11 - Interface de edição de parâmetros do estado inicial

Na interface de edição ilustrada, uma caixa de combinação (*combo box*) editável localizada na parte superior esquerda serve para selecionar ou digitar um identificador de parâmetro. Parâmetros novos podem ser criados ao se entrar com um identificador inexistente no arquivo, ou um parâmetro existente pode ser modificado caso contrário. À sua direita se localiza outra caixa de combinação, utilizada para escolha do tipo do parâmetro, de estado ou de

evento. A caixa de texto abaixo delas permite a entrada de valores de inicialização do parâmetro. Abaixo dela, um rótulo de texto exibe o valor numérico na forma em que ela será escrita no arquivo XML, a fim de fornecer uma idéia ao usuário quanto à precisão esperada do valor. No canto inferior da janela, há um botão no lado esquerdo para salvar as alterações, e outro do lado direito para apagar o parâmetro do estado inicial. Para cancelar a edição ou sair da janela sem aplicar alterações, basta clicar no botão vermelho no canto superior direito para fechar a janela.

A figura abaixo ilustra a interface de edição de curvas de funções implementada protótipo de configurador de modelos.

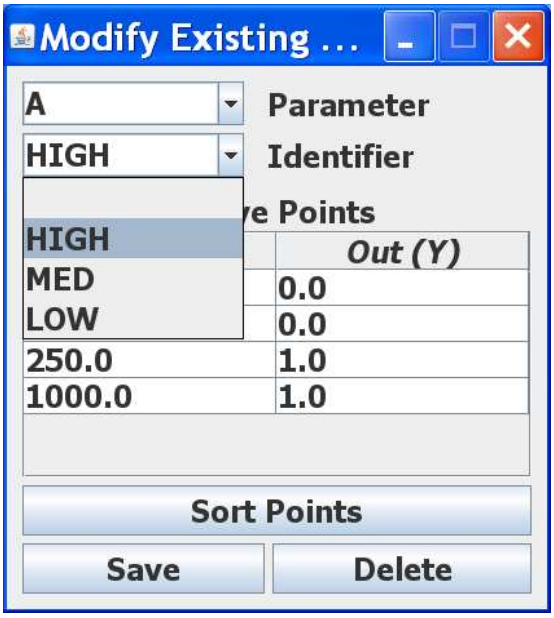


Figura 9.12 - Interface de edição de curvas de funções

Nesta interface, as caixas de combinação editáveis presentes na porção superior esquerda permitem a entrada do parâmetro associado e do identificador da curva. Na figura, a segunda caixa de combinação revela a existência de três curvas associadas ao parâmetro exibido. A tabela posicionada abaixo delas permite a manipulação pontos da curva, através da edição de valores entrada e saída. Abaixo da tabela, um botão reordena os

pontos da tabela por ordem crescente de entrada. Logo abaixo, botões de salva e apaga desempenham funções iguais às da interface de edição de parâmetros. E no canto superior direito, o botão de fecha janela serve para cancelar qualquer operação não salva.

A interface de edição de regras proposta para o protótipo de configurador de modelos é apresentado na figura abaixo.

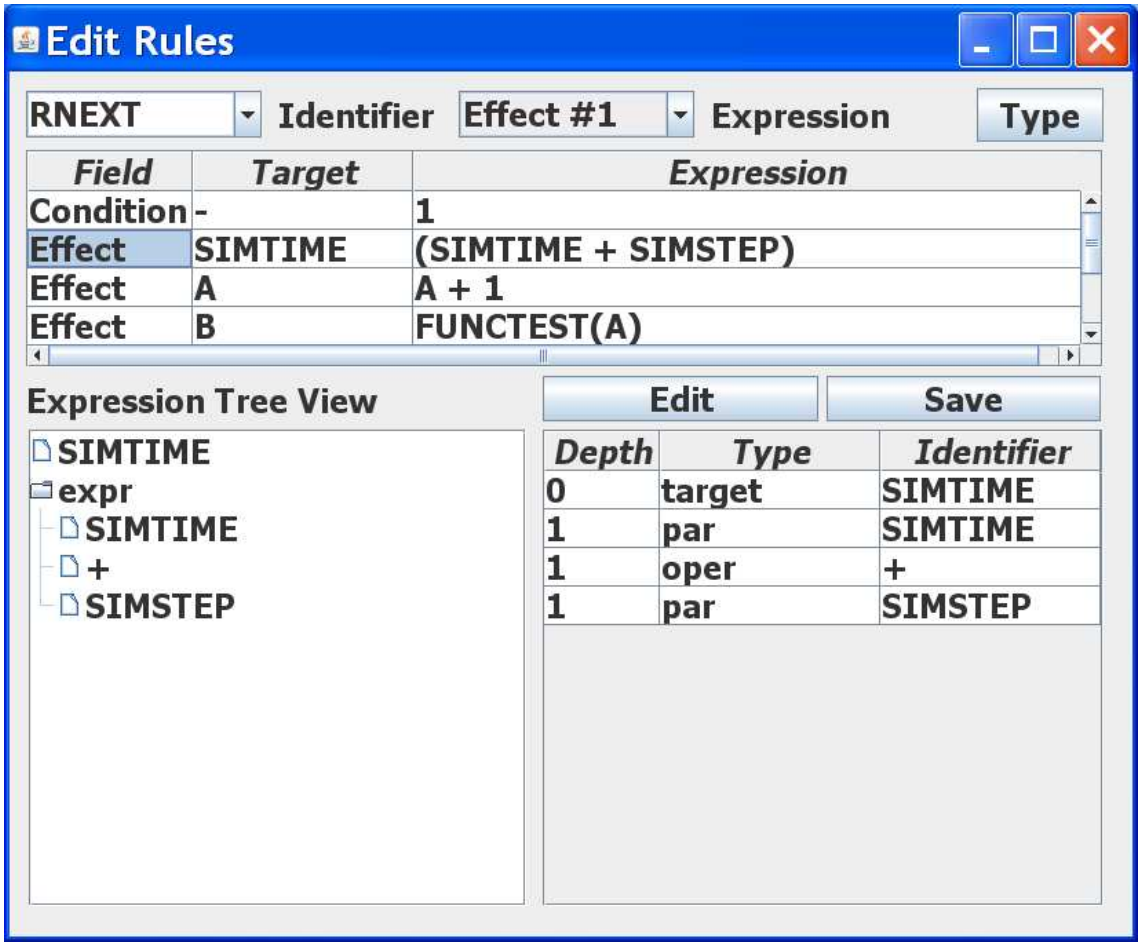


Figura 9.13 - Interface de edição de regras de controle

Na interface mostrada acima, a caixa de combinação editável na região superior esquerda permite a entrada de um identificador de regra. A caixa de combinação à sua direita permite a seleção de sua condição ou um efeito. Esta seleção é refletida na porção inferior da janela, onde é exibida a estrutura

selecionada em forma de árvore, à esquerda, e em forma de tabela, à direita. A tabela acima delas exibe a condição e os efeitos que compõem a regra, de uma forma mais intuitiva para um usuário humano. O botão acima dela, à direita, permite selecionar entre exibir ou esconder informações adicionais referentes às expressões, conforme mostrado na figura abaixo.

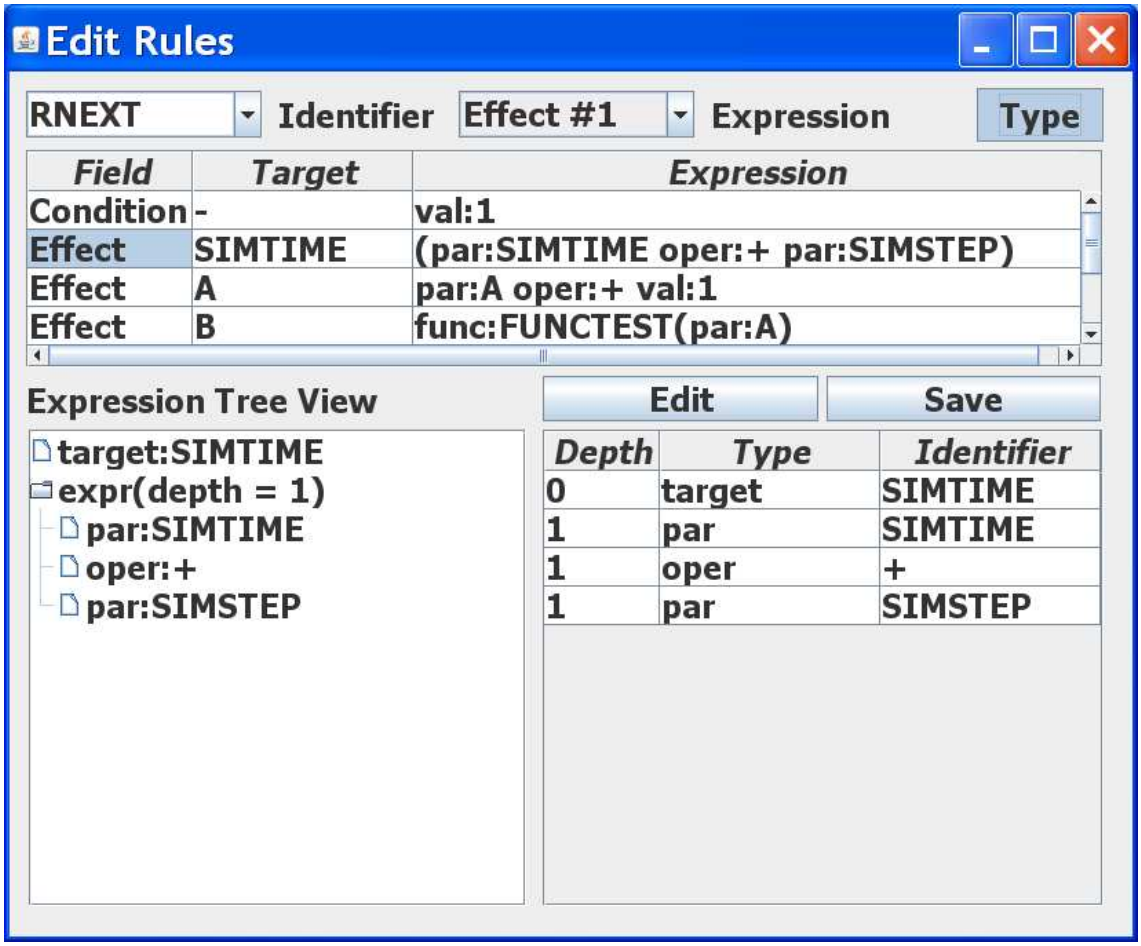


Figura 9.14 - Interface de edição de regras de controle, em modo de exibição de detalhes de expressões

Neste modo de exibição, o botão de editar, localizado próximo ao centro da janela, entre as duas tabelas, habilitaria a modificação de valores na tabela acima, sendo estas modificações salvas através do botão à sua direita. Entretanto, este recurso não foi implementado.

O desenvolvimento do protótipo do configurador de modelos foi interrompido por duas razões. A primeira delas se deve a novos estudos que levaram à elaboração de novos requisitos para o configurador de modelos, que dificilmente seriam cumpridos mantendo-se a abordagem atual de implementação de interfaces. A segunda, talvez mais importante, deve-se a um recente esforço visando à integração da configuração do modelo do simulador ao editor de domínio do planejamento de operações, dentro do escopo do projeto de arquitetura unificada de planejamento de operações. (KONO et al., 2010)

9.4. Arquitetura unificada de planejamento de operações

A figura abaixo mostra a arquitetura proposta para o novo sistema de planejamento de operações de satélites, segundo o projeto de automação atualmente em desenvolvimento no CRC.

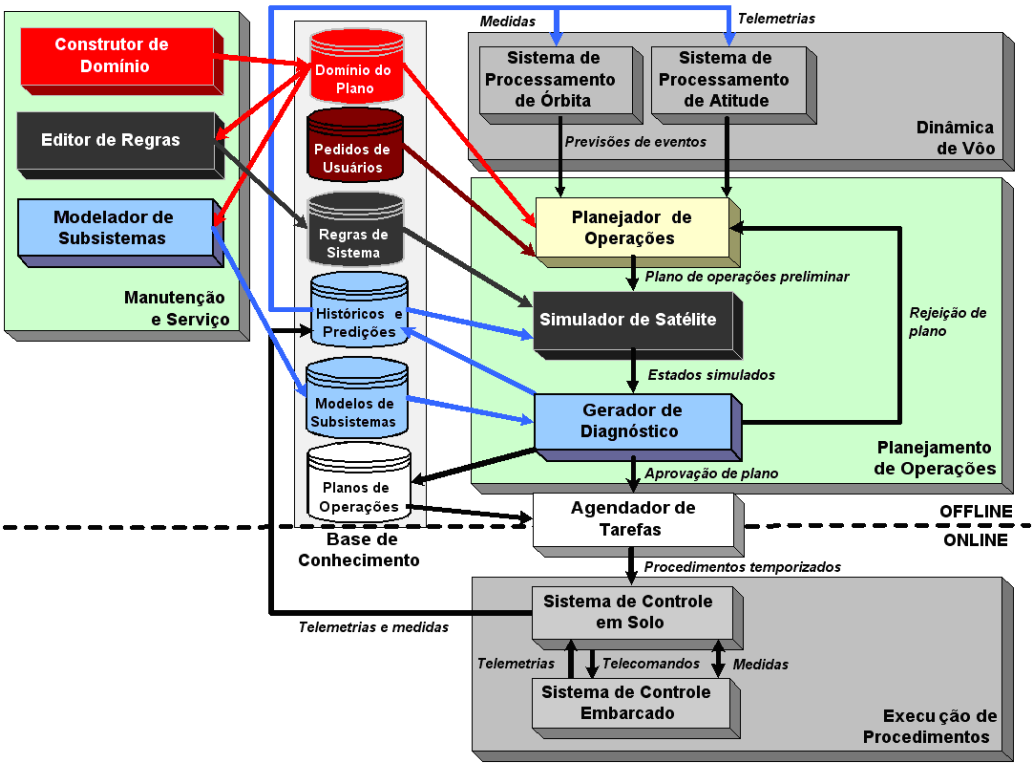


Figura 9.15 - Arquitetura do novo sistema para planejamento autônomo de operações

Na figura anterior, o simulador de satélite é indicado pela cor preta. O modelo do simulador, incluindo estados iniciais, fila de eventos, regras de controle e curvas de funções armazenadas em arquivos de banco de dados XML, são indicados na figura pelo nome de regras de sistema. O configurador de modelos e o pré-processador de eventos são agrupados sob editor de regras.

Conforme indica a figura, um dos objetivos do projeto de integração é modelar o domínio de planejamento de operações, indicado em vermelho, incluindo aspectos de modelos de sistemas utilizados pelo simulador de satélite e pelo gerador de diagnóstico. Caso seja implementado, este projeto deve permitir o compartilhamento de informações entre as diferentes bases de conhecimento. Em termos práticos do ponto de vista do simulador de satélite, a reconfiguração do modelo poderia ser feita através de realimentação de telemetrias obtidas do satélite e resultados de diagnósticos. Como um dos objetivos do projeto é a automação do sistema de planejamento, a sua implementação poderia significar a eliminação da necessidade de configuração manual de regras, ficando esta tarefa a cargo de agentes inteligentes responsáveis pela manutenção do sistema. (ROCHA, 2010)

10 CONCLUSÃO

Este trabalho apresentou uma proposta de verificação de planos de operações em vôo por meio de um simulador de satélites, projetado especificamente para este fim. Estudos realizados sobre outros simuladores de satélites revelaram deficiências quanto à sua capacidade de adequação do modelo a situações reais de operação, o que levou a uma busca por uma resposta para este problema. A solução encontrada consiste numa ferramenta de inteligência artificial conhecida como máquina de inferência, do tipo sistema especialista, associada a um modelo codificado na forma de base de regras.

Um protótipo do simulador foi construído em três etapas, conforme seu funcionamento proposto. A busca por um modelo configurável capaz de representar o conhecimento a respeito do satélite levou à elaboração de estruturas de dados constituídos por regras de controle e parâmetros de simulação divididos em estados e eventos, com a inclusão posterior de curvas de funções. O formato XML foi adotado para estes arquivos por ser de fácil manipulação na ausência de um editor específico. Em seguida, criou-se um núcleo de processamento capaz de processar os arquivos XML e gerar um histórico de estados internos em formato CSV, compatível com o sistema de telemetrias de satélites em uso no CRC. Por fim, foi iniciado o processo de construção do sistema de configuração do modelo, que se encontra em pleno desenvolvimento, incorporando conceitos que surgiram a partir de estudos ligados a outros trabalhos. Pontos em aberto, a serem endereçados em trabalhos futuros, incluem a conclusão de estudos para definir interfaces para a ferramenta de configuração de modelos, e sua integração à arquitetura de planejamento autônomo de operações atualmente em desenvolvimento.

10.1. Principais contribuições

Apresar das restrições atuais quanto à configuração do modelo, o simulador de satélite desenvolvido neste trabalho, composto pelo modelo armazenado em

base de regras XML e uma máquina de inferência no papel de núcleo de processamento, atende plenamente as necessidades do CRC para validar planejadores experimentais em desenvolvimento. Espera-se que o uso destes simuladores, devidamente configurados, permita que num futuro próximo a automação do planejamento de operações possa ser implementada operacionalmente, de forma segura conforme requisitado pela equipe do CRC. Desta forma, a conclusão deste trabalho constitui um passo fundamental para permitir a operacionalização de outros trabalhos em desenvolvimento visando à implantação do novo sistema autônomo de planejamento de operações de satélites do INPE. (KONO et al. 2010) (ROCHA, 2010) (SOUZA et al. 2009)

Uma das características principais do simulador desenvolvido é a liberdade dada aos seus usuários para a configuração de seu modelo interno, dando-lhes acesso inclusive a expressões de modelos matemáticos. A estrutura de dados adotada para a representação de regras é armazenada em banco de dados, assim como parâmetros e curvas. Uma vantagem desta abordagem é que ela permite ao usuário a configuração do modelo do simulador, seja para cadastrar novas missões, seja para recalibrar uma mesma missão existente, de uma forma bem mais abrangente do que permitiriam outros simuladores. Esta escolha possibilita, ainda, o acesso a informações contidas no modelo do simulador por meio de ferramentas externas. Esta característica permite que novos trabalhos relacionados a serem desenvolvidos futuramente, tais como o gerador de diagnósticos, possam utilizar o modelo do simulador para outros fins.

A saída gerada pelo simulador de satélite para verificação de planos consiste em um histórico de estados internos preditos. Em outras palavras, gera-se neste arquivo uma previsão do comportamento futuro do satélite simulado. Desta forma, outro emprego possível para este simulador é como ferramenta de diagnóstico de satélites em operações de tempo-real. Ao se criar uma versão modificada deste simulador, capaz de processar telecomandos

recebidos em tempo de execução, problemas em satélites podem ser detectados em tempo-real, através de comparação entre telemetrias e resultados de simulação.

Mesmo sem uma nova implementação do simulador como a descrita acima, a comparação direta entre históricos de estados e de telemetrias pode levar ao diagnóstico de falhas em bordo. Um dos protótipos deste simulador, gerados e configurados para representar satélites reais com a finalidade de validar o próprio simulador, revelou-se capaz de identificar problemas em um satélite que haviam passado despercebidos na execução de procedimentos de monitoração e análise. O satélite SCD1 sofreu uma falha num equipamento responsável pelo controle de carga da bateria em dezembro de 2009. Esta falha impede a operação do satélite em eclipse, durante o qual o suprimento de energia depende inteiramente da bateria, desta forma comprometendo seriamente a missão. Testes do simulador à época revelaram um comportamento anômalo nas telemetrias do satélite, o que levou à comutação do controlador de carga da bateria para redundante. Graças à execução deste procedimento, e pelo uso do simulador, o satélite SCD1 encontra-se plenamente operacional até o momento da escrita deste trabalho. (TOMINAGA et al., 2010)

Outros trabalhos relacionados diretamente à elaboração desta dissertação foram apresentados em congresso internacional SPACEOPS (*International Committee on Technical Interchange for Space Mission Operations; Ground Data Systems*), e nas oficinas regionais V ERMAC (Encontro Regional de Matemática Aplicada e Computacional) Vale do Paraíba e WORCAP (Workshop do Curso de Computação Aplicada), além de encontros internos de estudos de simuladores de satélites do INPE. Foram publicados artigos completos referentes às apresentações do SPACEOPS e da WORCAP. (TOMINAGA et al., 2008a) (TOMINAGA et al., 2008b) (TOMINAGA et al., 2008c) (TOMINAGA et al., 2009) (TOMINAGA et al., 2010)

O protótipo de simulador desenvolvido neste trabalho permite o uso de regras nebulosas, em adição a regras convencionais, para o modelamento de regras de sistemas. Na prática, foi constatado que o modelamento de regras de sistemas relacionados a operações de satélites faz pouco uso deste recurso. A descrição do protótipo de simulador descrito neste trabalho considera uma versão validada que implementa apenas parte do processamento de saídas nebulosas, através de funções de pertencimento em forma de rampa. Há uma versão não finalizada mais recente, que implementa a desnebulização de funções de pertencimento através de cálculos de centro de área. Contudo, as regras descritivas da dinâmica do sistema parecem apontar que a substituição pela nova versão não trará ganhos significativos, ao menos para testes de planos de operações em vôo. Entretanto, não se descarta a possibilidade de que outras aplicações, ainda não vislumbradas no momento, poderão vir a fazer melhor uso desta ferramenta de inteligência artificial no futuro.

10.2. Trabalhos futuros

Um possível passo futuro para dar continuidade ao projeto de simuladores baseados na arquitetura apresentada consiste em gerar máquinas de inferências distribuídas. No simulador apresentado, uma única máquina de inferência processa toda a informação referente ao sistema, através de um único estado interno. Nestas condições, o modelamento de sistemas mais complexos pode se tornar complicado. Uma possível solução para este problema consiste em dividir o estado interno em conjuntos de parâmetros relacionados entre si, através de regras em comum, e atribuir o processamento de cada conjunto de parâmetros a uma máquina de inferência distinta. Em outras palavras, teríamos simuladores de subsistemas sendo executadas paralelamente, trocando informações entre si para compor uma sessão de simulação. Esta abordagem permitiria teoricamente a criação de simuladores de altíssima fidelidade, através da composição de partes responsáveis pelo processamento de modelos o quão detalhista se queira.

Este trabalho abre caminho para a construção futura de simuladores, diagnosticadores, e outras ferramentas de inferência, baseadas em modelos de sistema constituídos por regras causais. Apesar de o projeto original deste trabalho objetivar a criação de um simulador de satélite para a verificação de planos de operações em vôo, a ferramenta criada possibilita modelar e gerar previsões de qualquer sistema causal. Acredita-se que o estudo apresentado neste trabalho pode ser adaptado e aproveitado para a confecção de aplicativos para os mais diversos fins, desta forma contribuindo potencialmente para a ampliação do conhecimento além da área de operações espaciais.

REFERÊNCIAS BIBLIOGRÁFICAS

ABRAHAM, A. Rule-based expert systems. In: SYDENHAM, P.; THORN, R. (Eds.). **Handbook of measuring system design.** 1. [S.l.]: Wiley, 2005. 1648 p. ISBN (978-0470021439).

AGÊNCIA ESPACIAL BRASILEIRA (AEB), **Programa nacional de atividades espaciais PNAE 2005-2014.** Brasília, 2005. Disponível em: <http://www.aeb.gov.br/download/PDF/pnae_web.pdf>. Acesso em: 17 mar. 2010.

AMBROSIO, A. M.; CARDOSO, P. E.; BIANCHI NETO, J. Brazilian satellite simulators: previous solutions trade-off and new perspectives for the CBERS program. In: INTERNATIONAL CONFERENCE ON SPACE OPERATIONS, 9TH, 2006, Rome, Italy. **Proceedings...** 2006. p. 7. CD-ROM. (INPE-14068-PRE/9237). Disponível em: <http://urlib.net/sid.inpe.br/mtc-m16@80/2006/08.21.15.01>. Acesso em: 17 mar. 2010.

AMBROSIO, A. M.; BARRETO, J. P.; GUIMARÃES, D. C. Satellite simulator requirements specification based on standardized space services. In: ISPE INTERNATIONAL CONFERENCE ON CONCURRENT ENGINEERING, 14. (CE 2007), 2007, São José dos Campos. **Proceedings...** São José dos Campos: Springer, 2007. p. 171-179. CD-ROM; On-line. ISBN 978-184628-975-0. (INPE-15259-PRE/10080). Disponível em: <http://urlib.net/dpi.inpe.br/ce@80/2007/01.22.20.39>. Acesso em: 17 mar. 2010.

ARGÜELLO, L.; MIRÓ, J. Distributed interactive simulation for space projects. **ESA Bulletin**, v. 102, p. 126-130, 2000. Disponível em: <http://www.esa.int/esapub/bulletin/bullet102/Arguello102.pdf >. Acesso em: 17 mar. 2010.

BERNARD, J. A. Use of a rule-based system for process control. **IEEE Control Systems Magazine**, v. 8, n. 5, p. 3-13, 1988.

BIANCHO, A. C.; AQUINO, A. C.; FERREIRA, M. G. V.; SILVA, J. D. S.; CARDOSO, L. S. Software agents for multiple satellite automated planning and control. In: AIAA INTERNACIONAL COMMUNICATIONS SATELLITE SYSTEMS CONFERENCE, 24TH (ICSSC 2006), 2006, San Diego, Califórnia. **Proceedings...** 2006. p. 06. Papel. (INPE-14075-PRE/9244). Disponível em: <http://urlib.net/sid.inpe.br/mtc-m16@80/2006/08.21.19.57>. Acesso em: 17 mar. 2010.

CÂMARA, G. **INPE 2008-2010**, São José dos Campos: INPE, 2008. Disponível em: <http://www.dpi.inpe.br/gilberto/present/inpe_2008_2010.ppt> Acesso em: 17 mar. 2010. (Apresentação para o Ministro Sérgio Rezende).

CARDOSO, L. S.; FERREIRA, M. G. V.; ORLANDO, V. An intelligent system for generation of automatic flight operation plans for the satellite control activities at INPE. In: INTERNATIONAL CONFERENCE ON SPACE OPERATIONS WILL BE HOSTED BY THE, 9TH, 2006, Rome, Italy. **Proceedings...** 2006. p. 9. CD-ROM. (INPE-14072-PRE/9241). Disponível em: <http://urlib.net/sid.inpe.br/mtc-m16@80/2006/08.21.18.08>. Acesso em: 17 mar. 2010.

CHINA ACADEMY OF SPACE TECHNOLOGY (CAST); INSTITUTO NACIONAL DE PESQUISAS ESPACIAIS (INPE). **CBERS 2B satellite to control segment interface specification.** 107 p. Beijing, 2006. (C-IFS-002(2B)/02)

CHINA ACADEMY OF SPACE TECHNOLOGY (CAST); INSTITUTO NACIONAL DE PESQUISAS ESPACIAIS (INPE). **CBERS 02B applicable handbook.** 179 p. Beijing, 2007. (CB-SUM-001(02B))

CHEN, M.; RINCÓN-MORA, G. A. Accurate Electrical Battery Model Capable of Predicting Runtime and I-V Performance. **IEEE Transactions on Energy Conversion**, v. 21, n. 2, p. 504-511, 2006.

GOLDFINGER, A.; SILBERBERG, D.; GERSH, J.; HUNT, J.; WEISKOPF, F.; SPISZ, T.; MOU, Z. G.; ROGERS, G.; SEMMEL, R. A knowledge-based approach to spacecraft distributed modeling and simulation. **Advances in Engineering Software,** v. 31, p. 669-677, 2000.

GONÇALVES, L. S. C. **Aplicação da tecnologia de agentes de planejamento em operações de satélites.** 2006. 167 p. (INPE-14092-TDI/1075). Dissertação (Mestrado em Computação Aplicada) - Instituto Nacional de Pesquisas Espaciais, São José dos Campos. 2006. Disponível em: <http://urlib.net/sid.inpe.br/MTC-m13@80/2006/05.04.17.36>. Acesso em: 17 mar. 2010.

HAROLD, E. R.; MEANS, W. S. **XML in a nutshell**: a desktop quick reference. 2 ed. [S.l.]: O'Reilly Media, 2002. 640 p. ISBN (978-0596002923).

HENDRICKS, R.; EICKHOFF, J. The significant role of simulation in satellite development; verification. **Aerospace Science and Technology,** v. 9, n. 3, p. 273-283, 2005.

KANG, J. Y.; KIM, J. M.; CHUNG, S. J. Design; development of an advanced real-time satellite simulator. **ETRI Journal**, v. 17, n. 3, 1995.

KONO, Y.; FERREIRA, M. G. V.; SARAIVA JÚNIOR, F. E. G.; PEREIRA, L. M. Strategies for implementation of an automated planning system. In: SPACEOPS 2010 CONFERENCE, 2010, Huntsville, USA. AIAA, 2010. **Proceedings...** [S.l.]: AIAA, 2010.

INNORTA, D.; WILLIAMS, A. New Simulation Approach for the Training of Satellite Mission Control Teams. In: ASIA INTERNATIONAL CONFERENCE ON MODELING & SIMULATION (AMS'07), 1., 2007, Phuket, Thailand. **Proceedings...** [S.l.]: IEEE, 2007. p. 562-567.

INSTITUTO NACIONAL DE PESQUISAS ESPACIAIS (INPE). **CBERS-2 attitude and battery anomaly November 2007.** São José dos Campos, 2007. 11 p.

INSTITUTO NACIONAL DE PESQUISAS ESPACIAIS (INPE). **CBERS 2B power budget.** São José dos Campos, 2006. 8 p. (CB-IFD-0010(2B)/01)

INSTITUTO NACIONAL DE PESQUISAS ESPACIAIS (INPE). **Data Collecting Satellite operation handbook.** São José dos Campos, 1988. 214 p. (A-MIN-0014)

INSTITUTO NACIONAL DE PESQUISAS ESPACIAIS (INPE). **Proposal for CBERS-2B Operations Plan in Routine Operation Phase.** São José dos Campos, 2008. 5 p.

INSTITUTO NACIONAL DE PESQUISAS ESPACIAIS. **Proposta de um simulador para o satélite SCD1**. São José dos Campos, 1990. 81 p. (OPR-RE-001)

INSTITUTO NACIONAL DE PESQUISAS ESPACIAIS (INPE). **Requisitos do software simulador CBERS3 - treinamento de operadores.** São José dos Campos, 2007. 33 p. (RT-SRS-1002/00)

INSTITUTO NACIONAL DE PESQUISAS ESPACIAIS (INPE). User requirements for the CBERS high fidelity satellite simulator equipment. In: XI'AN SATELLITE CONTROL CENTER (XSCC); INSTITUTO NACIONAL DE PESQUISAS ESPACIAIS (INPE). **CBERS simulator software design documents.** Xi'an/São José dos Campos: XSCC/INPE, 2000. Part 1, p. i-iii + p. 1-6

JANTZEN, J. **Tutorial on fuzzy logic**. Denmark: Technical University of Denmark, 1998. Disponível em: < www.iau.dtu.dk/~jj/pubs/logic.pdf >. Acessado em: 17 mar. 2010.

LEE, S.; CHO, S.; LEE, B. S.; KIM, J. Design, implementation, and validation of KOMPSAT-2 software simulator. **ETRI Journa**l, v. 27, n. 2, 2005.

LIAO, S.-H., Expert system methodologies and applications - a decade review from 1995 to 2004. **Expert Systems with Applications,** v. 28, p. 93-103, 2005.

LYON, R.; SELLERS, J.; UNDERWOOD, C. Small satellite thermal modeling and design at USAFA: FalconSat-2 applications. In: 2002 IEEE AEROSPACE CONFERENCE, 2002, Big Sky, USA. **Proceedings…** [S.l.]: IEEE, v. 7, p. 7-3391-7-3399, 2002.

MATHESON, L., METEOSAT Second Generation: automated procedures execution algorithms. In: SPACEOPS 2008 CONFERENCE, 2008, Heidelberg, Germany. **Proceedings...** [S.l.]: AIAA, 2008.

MANZO, M. A.; STRAWN, D. M.; HALL, S. W. **Aerospace nickel-cadmium cell verification - final report.** Washington, USA: National Administration for Space and Aeronautics, 2001. (NASA/TM-2001-210598)

MO, H. S.; LEE, H. J.; LEE, S. P. Development and testing of satellite operation system for Korea Multipurpose Satellite-I. **ETRI Journal,** v. 22, n. 1, 2000.

NIKOLOPOULOS, C. **Expert systems:** introduction to first and second generation and hybrid knowledge based systems. 1. New York, Taylor & Francis, 1997. 331 p. ISBN (9780824799274)

REGGESTAD, V.; GUERRUCCI D.; EMANUELLI, P. P.; VERRIER D. Simulator development: the flexible approach applied to operational spacecraft simulators. In: SPACEOPS 2004 CONFERENCE, 2004, Montreal, Canada. **Proceedings...** [S.l.]: AIAA, 2004. Disponível em: <http://www.aiaa.org/spaceops2004archive/downloads/papers/SPACE2004sp-template00139F.pdf >. Acesso em: 17 mar. 2010.

SCHUM, W. K.; DOOLITTLE, C. M.; BOYARKO, G. A. Modeling and simulation of satellite subsystems for end-to-end spacecraft modeling. In: MODELING AND SIMULATION FOR MILITARY APPLICATIONS, 2006, Kissimmee, USA. **Proceedings…** [S.l.]: SPIE, v. 6228, p. 622804-1-622804-10, 2006.

SEBASTIÃO, N.; REGGESTAD, V.; SPADA, M.; WILLIAMS, A.; PECCHIOLI, M.; LINDMAN, N.; FRITZEN, P. A reference architecture for spacecraft simulators. In: SPACEOPS 2008 CONFERENCE, 2008, Heidelberg, Germany. **Proceedings...** [S.l.]: AIAA, 2008.

SILVA, R. R. **KPlanOO:** um meta-modelo orientado a objetos para descrição de domínios e problemas de planejamento. Dissertação (Mestrado em Computação Aplicada) - Instituto Nacional de Pesquisas Espaciais, São José dos Campos. 2010.

SOUZA, P. B.; FERREIRA, M. G. V.; SILVA, J. D. S.; AMBROSIO, A. M. Decision support tool for prediction of critical data to the satellite integrity. In: WORCAP 2009, 2009, São José dos Campos, Brazil. **Anais...** São José dos Campos: INPE, 2009. Disponível em: <http://www.lac.inpe.br/cap/arquivos/pdf/ST52.pdf>. Acesso em: 17 mar. 2010.

SUN, Z., XU, G.; LIN, X.; CAO, X. The integrated system for design, analysis, system simulation and evaluation of the small satellite. **Advances in Engineering Software**, v. 31, p. 437-443, 2000.

TOMINAGA, J.; FERREIRA, M. G. V.; SILVA, J. D. S. A proposal for implementing automation in satellite control planning. In: SPACEOPS 2008 CONFERENCE, 2008, Heidelberg. **Proceedings...** [S.l.]: AIAA, 2008.

TOMINAGA, J.; FERREIRA, M. G. V.; SILVA, J. D. S. An implementation of a satellite simulator as a fuzzy rule-based inference machine. In: WORCAP 2009, 2009, São José dos Campos, Brazil. **Anais...** São José dos Campos: INPE, 2009. Disponível em: <http://www.lac.inpe.br/cap/arquivos/pdf/ST32.pdf>. Acesso em: 17 mar. 2010.

TOMINAGA, J.; FERREIRA, M. G. V.; SILVA, J. D. S.; Pereira, L. M. Reconfigurable satellite simulator modeling approach for extended mission operations. In: SPACEOPS 2010 CONFERENCE, 2010, Huntsville, USA. **Proceedings...** [S.l.]: AIAA, 2010.

TOMINAGA, J.; FERREIRA, M. G. V.; SILVA, J. D. S. **Uma proposta de simulador de satélite com modelamento em nível de sistema.** São José dos Campos, 2008. Apresentação na oficina V ERMAC Vale do Paraíba - SP, realizada no Instituto Nacional de Pesquisas Espaciais (INPE), de 26 a 28 de ago. de 2008.

TOMINAGA, J.; FERREIRA, M. G. V.; SILVA, J. D. S. Verificação de planos de operações de satélites via simulador baseado em regras. In: WORCAP 2008, 2008, São José dos Campos. **Anais...** São José dos Campos: INPE, 2008.

XI'AN SATELLITE CONTROL CENTER (XSCC). Flight operation TT&C procedure. In: XI'AN SATELLITE CONTROL CENTER (XSCC). **CBERS operation handbook - book two**: ground TT&C operation. São José dos Campos: INPE, 2000. Part 10, p. 10.1-10.122

XI'AN SATELLITE CONTROL CENTER (XSCC), CHINA ACADEMY OF SPACE TECHNOLOGY (CAST). **CBERS 02B operations handbook**. Xi'an/Beijing: XSCC/CAST, 2009. 339 p.

XI'AN SATELLITE CONTROL CENTER (XSCC); INSTITUTO NACIONAL DE PESQUISAS ESPACIAIS (INPE). CBERS simulator software requirement. In: XI'AN SATELLITE CONTROL CENTER (XSCC); INSTITUTO NACIONAL DE PESQUISAS ESPACIAIS (INPE). **CBERS simulator software design documents.** Xi'an/São José dos Campos: XSCC/INPE, 2000. Part 3, p. i-iii + p.1-46